FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Answerer Ranking Framework for Community-Driven Question and Answer Platforms

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur/in

im Rahmen des Studiums

Software Engineering/Internet Computing

eingereicht von

Karl Stary
Matrikelnummer 0406241

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung
Betreuer: Univ.Prof. Dr. Schahram Dustdar
Mitwirkung: Dr. Benjamin Satzger

Wien, 08.05.2012

(Unterschrift Verfasser) (Unterschrift Betreuer)

Erklärung zur Verfassung der Arbeit

Karl Stary
Lanzendorf 12, 3071 Böheimkirchen

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

(Ort, Datum) (Unterschrift Verfasser)

Danksagung

Ich möchte mich bei Gott bedanken, für meine Talente und für das großartige Umfeld in dem wir hier in Mitteleuropa leben. Weiters bedanke ich mich bei meinen Eltern, meiner Großmutter, meinen lieben Schwestern, meinen Freunden, und bei allen, die mich unterstützt haben. Danken möchte ich auch Professor Dr. Schahram Dustdar, der diese Arbeit ermöglicht hat. Ein besonderer Dank gilt Dr. Benjamin Satzger, der mit seinem schnellen und ausführlichen Feedback den letztlich raschen Abschluss der Arbeit ermöglich hat.

"Das ist mein Gebot: Liebt einander, so wie ich euch geliebt habe."Johannes 15:12

Acknowledgements

I want to thank God for all my talents and for the great environment in which we are living in Central Europe. Further, I want to thank my parents, my grandmother, my dear sisters, my friends and all of those who supported me. I would also like to thank Prof. Dr. Schahram Dustdar, who enabled this work. I would particularly like to thank Dr. Benjamin Satzger, who provided me with fast and extensive feedback, which made the ultimately fast completion of my thesis possible.

"This is my commandment, that you love one another as I have loved you." John 15:12

Abstract

Web-based Question and Answer communities have become very popular within the last decade. Today, there are huge communities, where tens of thousands of questions are posted every single day. This work aims at improving such communities via automatically forwarding new questions to appropriate answerers. In addition, askers should not be required to categorize their questions. To achieve these goals a framework is presented which contains several components in order to estimate how suitable users are for answering a specific question. The three core components are Expertise/Knowledge, Authority and Availability, which are used to predict the accuracy, trustworthiness and response time of a user's potential answer regarding a particular new question. These components are crucial in the process of determining if a certain user is likely to give a satisfying answer to a specific question. The framework implements different approaches of these components, which can be combined for the user ranking calculation. Concrete realizations of the Expertise component are based on the Vector Space Model (VSM) and on the Query Likelihood Language Model. VSM is improved via Term Frequency, Inverse Document Frequency (TF-IDF), where IDF is based on the user collection. To the best of our knowledge, this is a novel interpretation, because throughout the literature, IDF is based on the question collection. Realizations of the Authority component are, for example, InDegree, PageRank, and ZScore. User activity is realized via a novel Activity Filter, which, in all conscience, has not been used by related works. It removes inactive users prior to the actual ranking determination of potential answerers. User profiles, which are the foundation for all these components, are built from previously best answered questions. The implementation of the Ranking Engine, which produces the ranking of potential answers in relation to their likelihood of answering specific new questions, is optimized for modern multi-core processors by leveraging concurrent programming. Based on a dataset, provided by the Yahoo! Research Alliance Webscope program, different ranking variants are evaluated and the most accurate one is determined.

Kurzfassung

Internetbasierte Frage und Antwort Gemeinschaften haben sich innerhalb der letzten zehn Jahre zu sehr populären Plattformen entwickelt. Heute gibt es riesige Gemeinschaften, in denen täglich mehrere Zehntausende Fragen gestellt werden. Das Ziel dieser Arbeit ist es, solche Gemeinschaften zu verbessern, indem neue Fragen automatisch zu passenden Mitgliedern weitergeleitet werden. Weites soll es nicht notwendig sein, dass Benutzer ihre Fragen kategorisieren müssen. Um diese Ziele zu erreichen wird ein System vorgestellt, welches verschiedene Komponenten enthält, mittels welcher die Wahrscheinlichkeit geschätzt wird, dass ein spezieller Benutzer solch einer Gemeinschaft fähig ist, eine spezielle Frage zu beantworten. Die drei Kern-Komponenten sind Fachkenntnis/Wissen, Autorität, und Verfügbarkeit. Sie werden verwendet, um die Richtigkeit, die Vertrauenswürdigkeit und die Antwortzeit von potentiellen zukünftigen Antworten eines speziellen Benutzers zu schätzen. Diese Komponenten sind essentiell um zu ermitteln, wie wahrscheinlich ein spezieller Benutzer eine spezielle Frage zufriedenstellend beantworten könnte. Konkrete Realisierungen der Wissens-Komponente basieren auf dem Vektor-Raum Modell (VRM) und auf dem Frage-Wahrscheinlichkeits-Sprachen-Modell (Query Likelihood Language Model). VRM verwendet Term Frequenz - Inverse Dokumenten Frequenz (TF-IDF), wobei IDF auf der Benutzersammlung basiert. Nach bestem Wissen ist das ein neuartiger Ansatz, da in der Literatur IDF immer mittels Fragensammlung ermittelt wird. Die Realisierungen der Autoritätskomponente sind zum Beispiel InDegree, PageRank, und ZScore. Benutzer-Aktivität ist mittels neuartigem Aktivitätsfilter berücksichtigt, welcher, nach bestem Wissen und Gewissen, noch bei keinen ähnlichen Arbeiten verwendet wurde. Dieser entfernt alle inaktiven Benutzer vor der eigentlichen Berechnung der Rangliste von potentiellen Antwortenden. Benutzer-Profile, welche die Grundlage für alle Komponenten bilden, werden von bisherigen, am besten beantworteten Fragen eines speziellen Benutzers erstellt. Die Implementierung des Systems ist für die Verwendung von modernen Multi-Kern Prozessoren ausgelegt. Basierend auf einem vom Yahoo! Research Allicance Webscope Programm bereitgestellten Datensatz werden verschiedene Varianten der Reihungsberechnung untersucht und die passendste ermittelt.

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

1.1 Motivation

The population of our planet is rising and now there are already living about 7 billion people [104] on it. More than 2 billion [73] of them are using the Internet and all of them have questions from time to time. Web search engines like Google are very important for finding information. However, since they are keyword based, it is difficult to get answers to more complex questions. Question and Answer (Q&A) communities are more suitable for asking questions which require some explanation and context, in order to be properly answered. There are many Q&A communities, where people can ask any question they have in mind. In contrast to search engines, where information is provided via content of documents, Q&A platforms offer the possibility to communicate with humans, to ask them for their opinions and their suggestions. The potential answerers also have the possibility to ask for clarification of the question, if there are uncertainties. People who provide answers are in some communities volunteers, in others they are paid answerers. For huge communities like Yahoo! Answers [107] with more than 90 million unique users worldwide according to [109] (stats from 2010), there is a huge amount of questions posted every single day. Not all of these questions receive answers, since the answers are provided only by volunteers and hence there is no guarantee that each question will ever receive an answer. There is plenty of space for improvements of asker satisfaction via improving response rate and also response time. Furthermore, potential answerers could be supported via automatic routing of new questions, according to their interests.

1.2 Contribution

For potential answerers it is difficult to find appropriate questions to answer, since in big Q&A communities there are so many questions available. Despite question categorization as provided by, for example, Yahoo! Answers there are still too many questions per category so that it is quite difficult to find questions that are interesting to the answerers. Some questions simply get

lost in this huge amount of new questions, because nobody notices them in time and so there are already a lot of new questions which move them down in the stream of new questions. Therefore, they become more and more unlikely to receive answers at all.

For askers the categorization of questions is sometimes quite difficult and inconvenient, because specific questions may belong to several different categories. In some communities, there are administrators, who try to clean up this mess via re-categorizing misplaced questions. However, this seems rather unfeasible for big Q&A communities, because of the already mentioned huge amount of questions.

This work aims for improving user satisfaction by making question categorization via the asker obsolete. Further, it is desired to improve the response time via routing questions to appropriate answerers and therefore the asker's satisfaction. In addition, this will increase the convenience of the answerers, since they do not have to look for suitable questions anymore.

The main contributions of our work are:

- Development of a framework for answerer ranking, which supports several state of the art technologies in the context of the core components for ranking users, which are:
  - Expertise/Knowledge - refers to the ability of a specific user of giving an accurate answer to a certain new question.
  - Authority - indicates the trustworthiness and reliability of a user.
  - Availability/Activity - crucial, since all knowledge is not helpful if it is not applied, which means mainly recently and frequently active users matter in the context of forwarding questions.
- Improvements of specific technologies/methods as follows:
  - User based term frequency - inverse document frequency (TF-IDF) applied to the Vector Space Model. All previously mentioned methods in the literature use TF-IDF weights based on the question collection which contains all questions within a community.
  - User activity is normally used via a kind of linear combination with other techniques. In this work, an Activity Filter is presented, by which all inactive users are removed prior to the actual calculation of the ranking of potential answerers of a Q&A community.
- The evaluation, which is based on a huge data set provided by Yahoo! Answers, compares different algorithms which are related to the core components of the framework and combinations of them.

1.2.1 Methodology

The methodology of this work is as follows: After a detailed analysis of the state of the art techniques, and an investigation of related work, a theoretical concept is developed. Based on this concept a framework will be implemented, that allows new questions and potential potential

answerers to be matched in order to produce a ranking of potential answerers according to their likelihood of answering a specific new question. Based on the calculated ranking of potential answerers a fraction of the top ranked users is meant to receive the new questions. Finally, the different components and their combinations of the framework will be evaluated via a huge test data set with more than 4 million questions provided by Yahoo! Answers.

1.3 Structure of this Work

The remainder of this work is organized as follows: The next chapter presents state of the art in the area of information retrieval. Further, there is an extensive investigation of popular community Q&A web sites which can be found in section 2.2. Chapter 3 summarizes already existing approaches of routing questions to appropriate answerers which are users who have knowledge related to the question and who have some kind of authority.

The proposed answerer ranking framework is provided in chapter 4, the implementation is described in chapter 5. The evaluation of the different components and their combinations that are supported by the framework is presented in chapter 6. The summary of this work can be found in chapter 7.

CHAPTER 2

State of the Art

This chapter summarizes state of the art information retrieval techniques which are applied by this work like the Query Likelihood Language Model (QLLM) and the Vector Space Model and there is also an extensive overview of the most famous English-language Community Question Answering (CQA) platforms in the section about web based question answering.

2.1 Information Retrieval

Information Retrieval (IR) has a long history and evolved over centuries and even over millennia as Singhal [42] describes it. Clearly, what matters to this work is the way it is interpreted nowadays. According to Manning et al. [30], IR is defined as the process of searching for information of a formless kind like text. This text is usually located within a document which itself is part of a huge document collection. The data is assumed to be persisted on computers.

Wikipedia [91] has a more comprehensive description for IR where it is explained as scientific domain dealing with searching for documents and content within them, or meta data about documents. Furthermore, IR includes also searching for information in structured storage, relational databases or online. IR spans over a number of disciplines like mathematics, information science, library science, linguistics, statistics etc. with the common foundation which is computer science.

2.1.1 TF-IDF

According to Manning et al. [30] and [100], TF-IDF (Term Frequency - Inverse Document Frequency), which combines TF and IDF, is used to weight terms of documents. In this work it is used to weight terms of user profiles. As a result, these profiles can be more effectively compared to search queries or simply to questions.

The Term Frequency (TF) weights terms equal to their count that they occur in a specific document. TF of a term t in a document d is denoted as $tf_{t,d}$. This way of interpreting a docu-

ment is also referred to as the Bag of Words Model [90] where the word order does not matter but only the number of times they occur is preserved. The idea is that documents with similar bag of words notations are also similar in content.

Inverse Document Frequency (IDF) is intended to overcome the problems TF alone would cause. TF on its own does not distinguish between more and less important terms of documents related to a search query. Some terms have almost no distinguishing ability like, for example, a specific term which occurs in nearly all documents. Such a term cannot be used to decide what document is relevant to a search query. IDF is therefore used to diminish the influence of terms t with a high document frequency df_t that is the number of documents from the document set in which t occurs, because they do not have a significant impact on the result for relevance determination. IDF is defined as follows:

$$idf_t = log\frac{N}{df_t}$$

where N represents the number of documents in the collection. The logarithm used by Manning et al. is to the base 10, however, they mention that the base of the log function does not affect the ranking. To summarize IDF, terms that occur only in few documents receive high weights whereas terms which occur in many documents are weighted lower.

The combination of TF and IDF which is used to calculate the weight of a specific term t from document d is given by:

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

The following three properties of TF-IDF are highlighted by Manning et al.:

- The weight of term t is highest when it occurs only in few documents but many times within them. Such a term is therefore very valuable in distinguishing relevant from non relevant documents.
- A term's weight is lower if it either has a low $tf_{t,d}$ value or it occurs in many documents.
- A term is powerless to determine the relevance of a document if it occurs in each document of the entire collection.

2.1.2 Vector Space Model

The Vector Space Model, as described by [30, 101], is used to represent documents and also queries as vectors within one vector space. Each distinct term of document d corresponds to one dimension of the vector $\vec{d}$. In general, the value of a dimension is determined by the frequency the corresponding term occurs in the document. Of course, weighting mechanisms as for example TF-IDF can be used to replace the plain term frequencies which is exactly how it is done by this work. As described above in the paragraph about TF, the vector representation does not preserve the word order from the original document.

In the context of this work, the Vector Space Model is used to determine the relevance of documents related to queries via the calculation of the similarity of their vector representations.

$$sim(\vec{d_1}, \vec{q}) = cos\Theta = \frac{\vec{d_1} \cdot \vec{q}}{\left\|\vec{d_1}\right\| \|\vec{q}\|}$$

The dot product of the two vectors $\vec{d_1} \cdot \vec{q}$ is defined as $\sum_{i=1}^{N} d_{1,i} q_i$, where N represents the number of dimensions. The Euclidean length or simply the norm vector is denoted via $\left\|\vec{d_1}\right\|$ and $\|\vec{q}\|$ which is calculated as:

$$\left\|\vec{d_1}\right\| = \sqrt{\sum_{i=1}^{N} (d_{1,i})^2}$$

and for $\vec{q}$ respectively. The calculation has the effect that the vectors become unit vectors via length normalization by $\frac{\vec{d_1}}{\|\vec{d_1}\|}$ and for $\vec{q}$ accordingly.

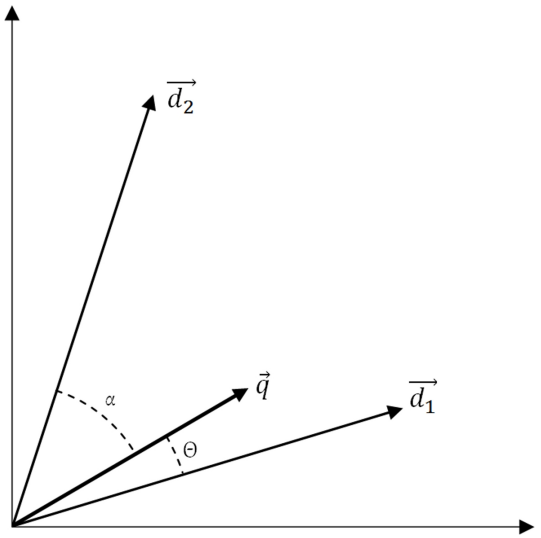

Figure 2.1: Cosine similarity $sim(\vec{d_1}, \vec{q}) = cos\Theta$

Figure 2.1 shows the angle Θ between the vectors $\vec{d_1}$ and $\vec{q}$. It further indicates that $\vec{d_1}$ is more relevant to the original query q than $\vec{d_2}$. The smaller the angle between question vector and document vector, the higher the cosine value becomes. $cos(0) = 1$ is equal to a perfect match, whereas $cos(90) = 0$ means that question and document vector are orthogonal because they are totally different with zero similarity.

Based on the similarity scores of the documents related to a query, the most relevant ones can be found because they have the highest values. In this work user profiles act as documents. They are matched against questions of other users in order to determine the relatedness of potential answerers to a particular question. High similarity values are supposed to indicate that the user has knowledge about and also to some degree interest in answering the question.

2.1.3 Language Model

Language Models (LM) have already been present in the context of Information Retrieval (IR) for more than a decade as the works of Ponte and Croft [36] and Song and Croft [44] show. [92] describes a LM as a model based on a statistical foundation that uses a probability distribution through which probabilities are assigned to word sequences which are in fact questions in the context of this work. Applied to IR, LMs are based on documents from a collection. For each document a LM is generated. Via these LMs the likelihood that a specific query or question is generated can be used to rank the documents in order to receive the most relevant documents for the query. According to Manning et al. [30], the probability that a query q is produced by a probabilistic language model M_d based on the document d which is denoted as $P(q \mid M_d)$, is high if the document contains the words of the query often.

There are different kinds of LMs. According to [92], so called unigram language models are used quite commonly in the context of IR. As Manning et al. describe, unigram models estimate the probability of each term independently:

$$P_{uni}(t_1 t_2 t_3) = P(t_1) \cdot P(t_2) \cdot P(t_3)$$

Bigram LMs are more complex because they consider prior terms:

$$P_{bi}(t_1 t_2 t_3) = P(t_1) \cdot P(t_2 \mid t_1) \cdot P(t_3 \mid t_2)$$

There are still more sophisticated LMs, however as Manning et al. mention, unigram models usually are sufficient in the context of IR. Further, it is mentioned that the potential improvement via using more complex models is often limited because of data sparsity. All LMs used by this work are unigram Language Models.

2.1.4 Query Likelihood Language Model

The Query Likelihood Language Model (QLLM) [30, 98] is the application of the basic LM in order to calculate the relevance of documents from a collection related to a search query. As already described above, for each document of the collection a language model M_d is generated. The probability $P(d \mid q)$ is understood as the likelihood that a specific document d is relevant to q. It is calculated via the application of Bayes' rule:

$$P(d \mid q) = \frac{P(q \mid d)P(d)}{P(q)}$$

Since $P(q)$ is equal for all documents it has no impact on the ranking and can therefore be ignored. The prior probability $P(d)$ of document d is generally interpreted to be uniform throughout all documents d. Therefore, it could also be ignored, but Manning et al. mention that $P(d)$ might be used to extend the model with a kind of prior like authority as it is done in the context of this work. Further, other document related attributes could be used which are user related attributes since user profiles represent the documents in this work. The probability $P(q \mid d)$ alone is left to rank the documents which can be seen as approach to model the query generation process.

According to a combination of Liu et al. [26], where the equation is slightly different from Manning et al., but more comprehensible and Manning et al., $P(q \mid d)$ is calculated as follows:

$$P(q \mid d) = K_q \prod_{t \in V_q} P(t \mid M_d)^{tf_{t,q}}$$

The multinomial coefficient K_q, which is related to a specific query q, is calculated, in compliance with [87], as follows:

$$K_q = \frac{L_q}{tf_{t_1,q}! \cdot tf_{t_2,q}! \cdot \ldots \cdot tf_{t_M,q}!}$$

Further, K_q, which is according to Manning et al. constant for a specific query, has no effect on the overall ranking process. Therefore, K_q is simply dropped. The absolute term frequency of term t from query q is denoted as $tf_{t,q}$. The probability $P(t \mid M_d)$ is the likelihood that t is generated out of the Language Model M_d, which itself is based on document d. The process of query generation is treated to be random. V_q represents the vocabulary of query q. $L_q = \sum_{i=1}^{M} tf_{t_i,q}$ is the number of tokens from the query q which is also called length of q, and M refers to the amount of distinct terms from q. [87] which is based on Manning et al. describes the model almost identically to the combination described here.

In conclusion, three steps are performed in order to achieve document ranking depending on the relevance to a specific query:

- For each document d from the collection a LM (M_{d_i}) needs to be generated
- $P(q \mid M_{d_i})$ needs to be calculated for each M_d
- Ranking of the documents related to their probabilities

Based on the unigram assumption, $P(q \mid M_d)$ is estimated via Maximum Likelihood Estimation (MLE):

$$\hat{P}(q \mid d) = \prod_{t \in V_q} \hat{P}_{MLE}(t \mid M_d)^{tf_{t,q}} = \prod_{t \in V_q} \left(\frac{tf_{t,d}}{L_d} \right)^{tf_{t,q}}$$

The ˆsymbol on P indicates the estimation of the model. As L_q for query q, L_d is the length of document d. The term frequency $tf_{t,d}$ is the same as $tf_{t,q}$, but based on a document d. The MLE estimation takes the term frequency of a specific term t and divides it by the total document length L_d.

One of the problems of this approach is that words which occur in the query q, but not in the document would lead to a zero probability for that particular document of being relevant to q. Only documents which contain all query terms would have a probability greater than zero. This is especially a problem when the training data used for generation of M_d is scarce. In the context of this work, user profiles based upon which the models are built may be based on only one best answer which clearly does not lead to an extensive vocabulary. Therefore, an effective solution for this problem is mandatory for this work. Another problem mentioned by Manning et al.,

is that words which occur only once in a document are in general over estimated because it is assumed that their appearance is partially random.

Smoothing, which has already been investigated by several researchers like [8, 50, 43, 3] , is applied in order to overcome these problems. It does not only help avoiding zero probabilities, but it also adds a kind of term weighting. On the one hand, non-zero probabilities are discounted and on the other hand, unseen words receive the weight. It is quite common that the probability for an unseen word is estimated via the Language Model M_c which is based on the entire document collection. Unseen words t are therefore estimated to be less than or equal to the probability of being generated by M_c.

$$\hat{P}(t \mid M_d) \leq \frac{cf_t}{\sum_{d \in D} L_d}$$

The term frequency cf_t is the raw collection frequency of term t. D represents the document collection and $\sum_{d \in D} L_d$ calculates the number of items contained in D. [87] mentions that this kind of assumption for an initial probability distribution over terms is denoted as Prior.

There are several approaches to combine term probabilities from documents with those of the entire collection. Linear Interpolation LM, which is also called Jelinek-Mercer smoothing [17], uses linear combination.

$$\hat{P}(t \mid d) = \lambda \hat{P}_{MLE}(t \mid M_d) + (1 - \lambda)\, \hat{P}_{MLE}(t \mid M_c) \tag{2.1}$$

The parameter λ is set to a value between 0 and 1. If λ is set to 1, no smoothing is applied, whereas a value of 0 yields only collection based probabilities. The value of λ needs to be set very carefully in order to efficiently apply smoothing. According to the results of Zhai and Lafferty [50], for small queries a λ value around 0.9 seems to be best. For long queries λ is better to be about 0.3. It is mentioned that these values indicate that more smoothing is required for long queries. (In order to avoid confusion it is mentioned that Zhai and Lafferty apply λ and $(1-\lambda)$ exactly the opposite way as for example Manning et al.. The equation for Jelinek-Mercer smoothing of this work is based on Manning et al..)

Since Amiri et al. [3] conclude in their work that tuning λ via the method proposed by Witten and Bell [48] yields promising results, it is also investigated by this work. λ is calculated as follows:

$$\lambda_{Witten} = \frac{\sum_{t \in d} tf_{t,d}}{\sum_{t \in d} tf_{t,d} + N_{DOC}}$$

where N_{DOC} refers to the number of distinct terms in the document d. Further, Amiri et al. point out that $\lambda \geq 0.5$ always holds.

Dirichlet Prior smoothing as described in [43, 87, 30] combines the multinomial probability distribution over words based on the document d with the multinomial probability distribution over words based on the entire document collection M_c via a Bayesian updating process where the probability for a term t based on the document model M_d becomes partially updated via $t's$ probability based on M_c. The individual amount of smoothing is determined by the document length, whereas γ acts as global smoothing adjustment. Bigger documents are relatively less smoothed than small documents. Dirichlet Prior smoothing is calculated as follows:

$$\hat{P}(t \mid d) = \frac{tf_{t,d} + \gamma \hat{P}(t \mid M_c)}{L_d + \gamma} \tag{2.2}$$

The prior probability of the term t is denoted via $\hat{P}(t \mid M_c)$ and γ is an adjustment parameter. According to Manning et al., the larger the value of γ, the more smoothing is applied. They further mention that short queries require only a little smoothing, whereas long queries need more smoothing. The parameter γ could therefore be based on the query size as suggested by Manning et al..

Smucker and Allan [43] explained that the linear interpolation model 2.1 can be extended via a Dirichlet prior which is in fact a document dependent extension. If λ is calculated as

$$\lambda = \frac{\sum_{t \in d} tf_{t,d}}{\sum_{t \in d} tf_{t,d} + \gamma}$$

and applied to 2.1, it is equal to Dirichlet prior smoothing 2.2. This is exactly the way how smoothing is applied by Liu et al. [26].

2.1.5 PageRank

PageRank, introduced by Page et al. [34], is a link analysis method [96] which has been used by the popular Google web search engine. It is used to calculate a weight for each web page which indicates the relative importance of it based on the link structure of the web. In the context of this work PageRank weights are calculated for each user in order to estimate the relative user authority.

The web graph is based on the links between web pages. Links to a particular web page are interpreted as recommendations for that page. Thus, many such links which are called "in links" indicate that the page has to have a high authority. The actual PageRank weight of, for example, page A depends on the number of in links and the PageRank value of those pages which link to A. The algorithm is defined recursively which means that the weights are propagated throughout the entire link structure. According to [96], PageRank is vulnerable to manipulation. The application in this work tries to overcome the spam problem where one page links many times to another page in order to boost the weight of that page via counting only distinct links between two specific web pages.

The algorithm distributes an equal initial probability to each page. Via an iterative process the actual values are calculated as follows [96]:

$$PR(p_i, t = 0) = \frac{1}{N}$$

It allocates an initial probability to each page. The number of pages considered for the PageRank calculation is denoted as N. The parameter t refers to the $t-th$ iteration.

$$PR(p_i, t+1) = \frac{1-d}{N} + d \cdot \sum_{p_j \in M(p_i)} \frac{PR(p_j, t)}{L(p_j)}$$

The PageRank value of page p_i after iteration $t+1$ is performed is denoted as $PR(p_i, t+1)$. The collection $M(p_i)$ refers to the pages that link to p_i. The number of outgoing links of p_j, to which $PR(p_j, t)$ is equally distributed is denoted as $L(p_j)$. The damping factor d, which is normally set to 0.85, is used in order to consider the probability that, for example, a random person who

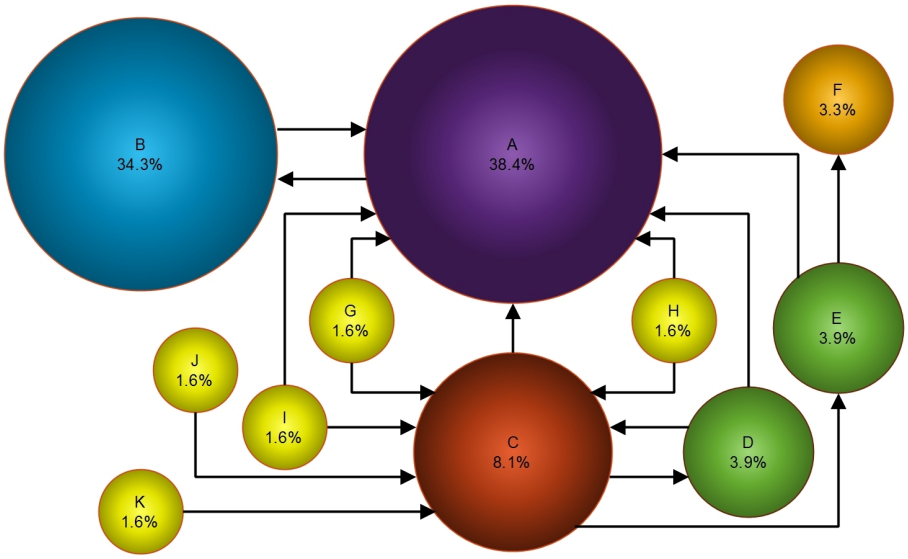

Figure 2.2: PageRanks represented via percentages, based on the figure presented by [96]

surfs through the internet (web surfer) will eventually stop surfing the web. The likelihood of continuing surfing the web is d.

The principle of PageRank is illustrated via figure 2.2. Page A receives the highest score, because it has strong support via in links of pages which themselves have some support from other pages. B receives a higher value than C even though it has just one in link, but this one is the most valuable one which is in this example coming from page A. Pages with no in links like G or H receive just a very small value depending on the damping factor since no pages support their score. Without that damping factor those pages would have a zero probability. Therefore, damping gives every page some probability of randomly being chosen from the web surfer. It is further mentioned that a random web surfer who starts at a particular page has an 85% likelihood of choosing a link from that page and the remaining 15% indicate the chance that a random page from the entire web is chosen by the web sufer where the likelihood that for example page B is chosen is 34.3%. The 85% and 15% respectively depend on the value of the damping factor.

2.2 Web based Question Answering

There are many different possibilities of how to find answers to questions. This section provides an overview of today's most relevant web based Question Answering (QA) systems. In general, web based search for answers can be divided into two main fields, namely Social Search and Non Social Search. The focus of this work is on Community Question Answering (CQA) which is a special kind of Social Search.

According to [9], the term "Social Search" refers to search that involves social interaction in a broad context. [33] describes this term as searching for information via the help of friends, reference librarians or other people online.
The Wikipedia entry for Social Search [99] defines it in a different way: Social Search is described as to involve the Social Graph of the user who tries to find an answer to a question. The Social Graph represents the social network of the user. In this work Social Search is referred to as search for information based on general human interaction.

Traditional Search Engines like Baidu [63], Bing [64] "basic" Google [71] (basic, because Google recently launched a kind of Social Search extension) or Yahoo! [106] are very important in today's usage of the web in order to find quick solutions for problems or just to answer almost every kind of question. They are considered to be "Non Social" because answers are in general provided via documents and not from other people and so there is no interaction between people. Non Social QA solutions are beyond the scope of this work.

According to Gyongyi et al. [12], web forums and discussion boards mostly target specific topics. They are not explicitly made for question answering, but also for data sharing and so on. Often, these systems use a kind of reputation mechanism to verify content and user expertise. There are lots of forums for nearly every topic available. A main problem within these platforms is trust. Everybody could answer a question and therefore it is important that there are measurements for trust, reliability and expertise. Even if these systems can be considered related to Social Search they do not explicitly focus on the question answer pattern and are therefore beyond the scope of this work.

Emails are obviously one of the simplest ways to ask questions. Basically, it is only required that the asker knows the email address of the person to be asked. Of course it is important that the questioner knows about the expertise as well as the degree of reliance from the potential answerer.

In accordance to Gyongyi et al. [12], online chat rooms are places where people communicate about nearly everything and of course they are also asking questions sometimes. It is simply not sure that there is someone available to answer a specific question. Emails and chat rooms are also outside the scope of this work.

2.2.1 Community Question Answering Platforms

This section provides an overview of the most important and popular CQA platforms which are all based on Social Search. They all focus on the question answer pattern which is the foundation of this work.

Yahoo! Answers

Yahoo! Answers [12, 107] is a question answering community where users can ask and answer questions on almost any topic they want. According to [109] it has more than 90 million unique users worldwide and it is therefore the largest knowledge-sharing online community.

Gyongyi et al. [12] point out, that Yahoo! Answers also aims to create a knowledge base which is used to preserve useful answers for future retrieval to other users with similar questions.

In order to ask questions users have to sign in. This can be achieved via a Yahoo! account or by using an external Facebook or Google account. After the successful login, a question statement is required and optionally details can be added. The next step asks for distinct categorization of the question. Once the question is submitted other users have 4 days to answer the question and there is an option to extend the available time to respond up to 8 days [108].

If there is at least one answer available for at least one hour to a specific question, the asker can choose a best answer, or if there are at least two answers available the questioner could pass this decision to the community for a two days voting period. If the asker does not take any action, the community voting process will start after the answering period expires and if there is at least one answer. This process also offers the so called "No Best Answer" choice. If an answer is selected to be the best one, the question changes its status to resolved and it becomes persisted. Otherwise, if the No Best Answer is chosen either by the asker or through the voting process, the question will be deleted. Questions with no answer will be deleted after the answer period expires.

As already mentioned above, questions are organized into topics with at most three levels and about 28 subjects in the top level. According to [2] this structure has more than 1000 lower level nodes. Using this categorization it is easier to find questions about a specific topic that the user is interested in.

In order to motivate users to participate in this system Yahoo! Answers offers incentives via "a scoring scheme" [12]. It is also used to "reward great answers" [108]. This scheme, which is based on points and levels, also shows others how active a specific user participates in the system. New users start with 100 points. Asking a question withdraws 5 points, but choosing a best answer returns three points. If the community vote leads to No Best Answer, 5 points are returned. Answering a question is worth two points, deleting it removes these two points. For a simple login one point per day is added. Voting for an answer adds one point, except the No Best Answer choice which does not change the points. If a user has a best answer, ten points are added. Clicking the thumbs-up icon on a best answer adds one point to the user who wrote that answer. Not more than 50 points can be received this way.

The distinction between levels addresses different concerns. On the one hand, new users are restricted in order to avoid misuse the system, on the other hand, users which are already on a higher level are obviously more participating in the question answering process and are therefore rewarded with this status. A key restriction is that users can access the rating functionality only from level two and higher. New users first have to earn enough points to reach this level. The higher the level the more questions a user can ask and the more votes can be made. Rating is done via clicking the thumbs-up for a good answer and the thumbs-down for a not so good answer. Users also can rate interesting or high quality questions via the star button which is a

measure for the popularity of a question. Further, users also have the possibility to comment on best answers. This allows to share opinions even after the question is resolved.

Yahoo! Answers focuses strictly on the question answer pattern in a sophisticated way. However, it does not support a discussion process to find an answer to a question. Additionally, Yahoo! Answers provides a feature called "Suggested Questions". This feature forwards new posted questions to potential answerers except those who are in the question author's blocked users list. The suggested questions are supposed to match with the potential answerers interests. There seems to be no detailed information available on the process of the suggestion of questions which could be compared to this work.

Askville

Askville [61, 62] is a typical question and answer website where people help each other with their knowledge and opinions. It is owned by Amazon.com. The sign-up process is slightly different from [62] in the point that a person who wants to join does not only need Internet access and an email address, but also a mobile or an amazon account with a purchase history of one of the following countries: USA, Canada, Germany, France, UK or Japan. Once the registration process is finished, new questions can be asked or answers can be given. New questions are automatically categorized, but this still could be changed by the user. There is also the possibility to add details even if the question is already submitted. Users have the possibility to follow specific topics, similar as following somebody on Twitter, in order they only see questions from this categories on their Askville homepage. Once a question occurs which the user wants to answer, the reply is added in the answer box below the question. There is also an option to attach files.

If a question receives an answer, the questioner can express his appreciation via hitting the "thanks" button. Another way of expressing emotions about the response is to make compliments. Good answers can be emphasized via voting "yes". The best answers will then be moved to the top in order to be found easier. Compared to Yahoo! Answers, which uses a system of points, Askville uses so called "Askville Achievements". These are awards - one of them is, for example, the "Highly Complimented Award" which is given for receiving many compliments or the "Inquisitive Award" for asking good questions. Another difference is the lack of restrictions for new users. Overall, Askville seems to be less complicated in terms of points and levels and related restrictions.

AnswerBag

Answerbag [53, 55] is another CQA website which does not only facilitate questions, but also polls and debates. According to [54], questions are answered by the community and from professional researchers. As usually, a login is required. A new user has two possibilities: One is to create an Answerbag account and another one is to use a Facebook account. To ensure that the question in mind is unique, it is suggested to search first whether the question has already been asked. In addition to the normal question text, there is also a possibility to add details if the available 140 characters for the question are not enough. In the next step the question needs to be categorized in order people can find it. This can be done via entering a specific word and

choosing a proposed category or by manually browsing the categories. There is also a choice between asking for advice, facts, discussion or fun. The question can also be posted to the friends on Facebook.

Answerbag also uses a motivation system with points and levels [55] which is similar to the scoring scheme of Yahoo! Answers but not so sophisticated. This system is also used to rate questions and answers in order the best ones appear on top of the results list. The authors of questions and answers earn points for each "Like" from other users. Users receive points for flagging questions, answers and comments which is done via clicking on "Report" and choosing an option like "Spam", "Offensive", "Nonsense" and so on, as soon as it is approved to be correct feedback. For great answers, which are determined from staff, moderators or community leaders, five points can be earned.

Obviously, the more points a user collects the higher the level of this user will be. These levels have specific names to point out the achievements of the user and are therefore called "ACHIEVEMENT LEVEL". Rating questions and answers is simply done via clicking the "Like" button. If a user changes his mind there is the possibility to unlike content that has previously been liked. Answerbag also offers the possibility to add other users as friends. A request is then sent to the other user and once it is confirmed the requester can keep track of her activities.

AOL Answers

AOL Answers [58] is a CQA site owned by AOL. It is based on Yedda, an Israeli startup, which was acquired by AOL a few years ago.

Login is possible via an account from AOL, Facebook or Twitter. If a Facebook account is used, AOL Answers first ask to invite the Facebook friends. Next, the user can choose the topics of interest. Finally, there is a chance to claim a Yeddai badge in memory to the acquired company named Yedda.

As a new question is entered, there is immediately a drop down box with automatically detected similar questions in order to receive already available answers and to avoid that the same questions are asked again and again. Once the question is accepted videos and pictures can be added. Finally, a topic and a subtopic needs to be chosen or a new topic could be created. Then the question can be posted. According to [58], the question will then be forwarded to somebody who has specified to have interest in the related topic.

Answers can also be extended via videos or pictures and comments can be added. To point out that a response might be good the "Helpful?" button can be clicked. Additionally, there is also a reporting system in order to detect spam or oder abuse of the system.

To motivate users and to thank them for great participation badges like "Day Runner", "Fastest Gun in The West", "Yeddai", or "On Call" are given and to figure out how involved a user is, there is a summary about his activities.

Answers.com

Answers.com [56] contains WikiAnswers [89] and also ReferenceAnswers [81]. The first one is a question and answer platform which uses wiki-based technologies, whereas the second one

provides answers to millions of subjects via reference publishers.

There is a significant difference in the approach between especially Askville and Answers.com in the way that Askville enforces real people behind each account whereas Answers.com does not even require an account for asking questions. However, there are several advantages of having an account, for example, to establish reputation, a user profile, the possibility to follow questions via putting them on a watch list, following categories of interest or just to customize some settings. To sign in there are a number of possibilities. A user can choose between an account of Facebook. Twitter, Yahoo!, Google and LinkedIn or a new Answers.com account can be created.

If similar questions are already existing, they are presented in order the same question is not asked several times and to offer immediate answers. Next, the question needs to be checked for question wording etc. For logged in users there is the option to add the question to their watch list. Additionally, new questions have to be categorized. Each question has to be assigned to not more than three topics. This can be accomplished by either searching for categories, choosing suggested ones or by browsing them.

Already submitted questions can be further edited to make them more precise or to improve spelling and grammar. Another aspect is to make very specific questions more generic in order they are interesting for more people. All this can be done only by contributors who are signed in and have already made enough contributions. Compared to other CQA sites, which support question details and attachments, Answers.com focuses on simple questions which are basically single sentences.

Even after submitting answers they can be further improved. There is also an answering policy which consist among others a plagiarism policy. To edit answers, the user needs to be signed in. Answers need to be changed or deleted for various reasons such as removing personal remarks, deleting plagiarized answers, deleting useless answers like "I don't know" etc. Questions with incomplete answers can be flagged for further improvement as to add more details or to correct errors or vandalism etc.

To further improve questions or answers links to related websites can be added. They are not included into the body of the answer, but added as "Related Links". Abuse of this feature needs to be corrected by the community via deleting spam links etc. Similar to external links, there can also be links to related questions within WikiAnswers. This is called "Add Related Links"; It helps to avoid the production of duplicate content.

WikiAnswers supports alternate questions where the same question is asked in different ways, so-called "Alternate Wordings". The main advantage of this idea is that it makes it easier to find questions and it avoids that the same question gets answered several times. In certain cases such as vandalism or spam answers can or actually should be removed. In other cases it might be sufficient to simply improve answers. Another specificity of WikiAnswers is "Pre-answering questions". People with knowledge about something that has not been asked before can create and answer a question themselves. In this case it is not about getting an answer to a question, but about sharing knowledge which might be interesting to others.

Question popularity is determined via votes which themselves depend on the opinion of the users. Questions with the most votes move up to the top of the category. This is also the way how question become Frequently Asked Questions (FAQ). There is no such system for

answers. Via "Trust Points" users can vote for others who have made valuable contributions. Anyway, WikiAnswers points out that this system is not a good indicator about how trustworthy an answer by a specific user is. Furthermore, there is a feature called "Share Button" [57] which is used to share answered questions with social networking sites like Facebook etc.

LinkedIn Answers

LinkedIn Answers [76] is part of the social network called LinkedIn [75]. It is about answering business related questions via knowledge, experience and opinions.

To get started an account needs to be created. The basic account is free. After signing in a question can be asked by simply entering the text and choosing one or two suitable categories. Additionally, there is an option in order to send the question only to selected contacts in a private manner. There is also an option to specify the region that matters to the question. If the question is not asked privately, it will appear on the asker's profile, on the LinkedIn homepage of his contacts, it will be listed beneath the Answers tab and in emails if also specific recipients are selected [77].

Answering questions is a good chance to present the own knowledge to potential employers and business partners, because answers are shown on the replier's profile and therefore a kind of prove about expertise. Answers will be visible beneath the question, on the answerer's profile, via email to the asker and on the websites of the replier's connections.

Expertise can be improved via answering questions. If the reply is chosen from the questioner as best, the answerer will earn one point in the specific category. Experts are recognized from LinkedIn via a list of experts in each area depending on the collected points.

Facebook Questions

Facebook offers an extension to support asking questions [69]. Members of Facebook can ask questions and create polls. Questions are shown via a user's News Feed. Friends of this user can answer and follow the question which is then included in their own News Feed and therefore their friends can also access and answer the question. Via privacy settings the ability of others to comment on questions can be restricted but not the visibility itself. Therefore questions are potentially viewable for everybody on Facebook. Anonymous questions are not supported.

There are still many more question and answer websites like, for example, Ask MetaFilter [60], ChaCha [65], Fluthter [70] etc. and therefore this is supposed to be an overview of the most famous, important and interesting ones rather than a complete survey.

Core Feature matrix

The core feature matrix summarizes the features of the described CQA platforms and is pretty much self-explanatory.
The complete feature matrix can be found in the appendix, tables A.1 and A.2.

Feature \ Q&A System	Yahoo! Answers	Askville	Answerbag	AOL Answers	Answers.com	LinkedIn Answers	Facebook Questions
Automatic question routing/forwarding/suggesting	x			x			
Manual question routing/forwarding		x					x
Reputation system	x	x	x	x	x		
Rating questions	x		x		x		
Rating/rewarding answers	x	x	x	x		x	x
Automatic question categorization		x					
Manual question categorization	x	x	x	x	x	x	
Following question categories		x			x		
Question expiration	x						
Question details	x	x	x			x	
Question constraints						x	
Posting questions on Social Networks (SN)	x		x	x		x	x
Sharing answered questions with SN					x		x
Anonymous questions (or answers)	x	x	x	x	x		
Private Questions						x	
Discussion support			x				x
Manually specifying user interests				x			x
Blocking other users	x	x					x
Spam and abuse reporting system	x		x	x	x	x	

Table 2.1: Core feature matrix summarizing characteristics of popular CQA platforms

CHAPTER 3

Related Work

This chapter provides an extensive overview of previous work on potential answerer ranking approaches related to CQA systems. The focus is on knowledge estimation via Vector Space Models and different interpretations of the Query Likelihood Language Model (QLLM), several authority calculation methods based on link structures of user networks and also on user activity consideration.

3.1 Ranking experts via the Vector Space Model and PageRank

Jiao et al. [18] presented ExpertRank which is used to find experts in online communities which are based on discussion groups. Their proposed system uses a vector space model and a form of PageRank [34] to rank users. It is distinguished between domain driven and domain independent information where the latter can be seen as the authority of a user which is calculated via an adapted version of PageRank. The domain driven information is modeled via user profiles which are filled with the content of discussion threads the user participates in. The Microsoft Discussion Groups are used as data source for the evaluation process.

The idea of this approach is that the expertise of a user can be calculated as a function of the similarity between a query and the user profile as one parameter and the authority of that user as another one. The user profile of a particular user is the result of merging all his posts from the discussion groups.
For the authority part user relations are extracted from user participations in threads. If, for example, a user A starts a thread and users B and C reply to it then A has a directed link to B and C. The problem for this authority approach is that users who just participate in small groups would gain a high authority score. This behaviour is considered to be spam and therefore a modification to the calculation is added.
The ranking score is calculated in two parts as already mentioned. First, the expert relevance is

calculated as follows:

$$RE\left(ca_i, q\right) = \frac{\sum_{j=1}^{t} w_j d_{ij}}{\sqrt{\sum_{j=1}^{t} \left(d_{ij}\right)^2 \sum_{j=1}^{t} \left(w_j\right)^2}}$$

$RE\left(ca_i, q\right)$ is the relevance score which is calculated via cosine similarity between the profile of candidate user ca_i and the query q. w_j represents the TF-IDF weight of the jth query term. The TF-IDF calculation is based on the question set which is significantly different to the approach presented in this work, where it is based on user profiles which makes it significantly more accurate as the results presented by this work show. d_{ij} is the term frequency of the jth term in the profile of user i. Since there is no information about any normalization of the term frequency the absolute term count is assumed for the reference implementation.

The authority of a user is the second component of the ranking score. The calculation is based on two kinds of relations namely the user - start topic relation and the user - reply to topic relation. Based on this information the user network is generated and the PageRank algorithm is applied which is adapted in order to avoid spam as mentioned above. In contrast to the approach of Jiao et al., in this work there is at most one link considered from user A to user B no matter how many questions of user A user B has answered. Therefore, two specific users can not significantly manipulate each others authority score.

The evaluation is based on crawled data from Microsoft's Office Discussion Groups which include a predefined expert list suitable for the evaluation of that approach. There are two strategies how to combine the expertise and the authority of a user:
The linear combination $ER(e, q) = a \cdot ER(e, q) + (1 - a) \cdot AU(e)$ shows that the result is best when the authority score alone ($a = 0$) is used for ranking. It significantly drops as a increases to 0.1. As the results from this work show, the linear combination of expertise and authority leads to significant improvements, whereas the question remains why their approach fails to do so. It could be that the value of a was not carefully enough checked between 0 and 0.1.
The second strategy is a cascade approach. First, the expertise score is used for ranking alone and then the top n users are picked and re-ranked using the authority score. This leads to a substantial improvement over the odd results of the linear combination. The used metrics among others are precision and recall, which are also considered by this work, however, different amount of data, in this work more than ten times of the amount of questions are used, makes it difficult to compare the results.

3.2 Question selection bias based on the existing question value

Pal and Konstan [35] introduced the concept of question selection bias which is based on the idea that expert users choose questions where they are more likely to produce valuable replies. These questions are assumed to be lacking of preexisting good answers. This user behaviour is the source for identifying experts. The data set is collected from TurboTax Live Community which is a tax related Q&A community. There exists a list of experts which is manually created by employees of TurboTax which is used for the evaluation.

First, the value of a question needs to be determined. This is based on several question attributes such as votes and answer status like best answer or helpful. The question value V_q is based on the value of its answers V_a:

$$V_q = \sum_a V_a = \sum_a w_0 \cdot votes(a) + w_1 \cdot status(a)$$

The status of an answer is 2 if it has a special one like best answer or helpful, otherwise it is 1. The more answers a question receives, the higher its value will be in general, but the value decreases if it has received negative votes. The values w_0 and w_1 are both set to 1 since this works best. The question value is further transformed into discrete values ranging from 0 to 5 which has the advantage that the learning process does not require a lot of data.

The question selection bias is measured as the degree to which a user's question selection preference is related to the given question value. The hypothesis says that experts tend to choose questions with low existing values in order to have a higher chance of making a valuable contribution, whereas normal users may not have the knowledge to answer them.

$$P(V_{uq} \mid A_{uq} = 1) = \begin{cases} 1 & \text{if } V_{uq} = v_0 \\ 0 & \text{otherwise} \end{cases}$$

All answers of a user are considered to calculate his selection bias. If user u selects a question with previous value v_0 where V_{uq} is a discrete random variable representing the value just before u adds his answer to question q, it indicates that this user prefers to answer that kind of questions. The random variable A_{uq} with the possible values $\{0, 1\}$ informs if the question q was answered by user u. With Bayes rule, the selection bias of a user is calculated in the following way:

$$P(V_u \mid A_u = 1) = \sum_q P(V_{uq} \mid A_{uq} = 1) \cdot P(A_{uq} = 1 \mid A_u = 1)$$

$P(A_{uq} = 1 \mid A_u = 1)$ represents the previous likelihood hat question q is selected by user u for answering, which is in general equal for all questions that are answered by u.
If, for example, $P(V = 0 \mid A = 1)$ is high for a specific user, it means that he prefers to answer questions with no previous value, whereas a high value of $P(V = 1 \mid A = 1)$ indicates that questions with a little existing value are preferred.

The machine learning model is based on a feature vector x_u which is defined for each user u.

$$x_u = [P(V_u = v \mid A_u) : \forall v, v \neq max(V_u)]^T$$

A user gets classified as expert or normal user via his feature vector using Ridge regression, Logistic regression and a generative model which is based on a Gaussian distribution.

Pal and Konstan used a 10-fold cross-validation [39] for the learning process. The results on selection bias show considerable differences between experts and normal users on all values of v, especially for $v = 0$ where the question value is zero indicating no or bad answers in which case experts have a higher probability to answer them than normal users. If $v = 1,\ 2,\ 3,\ 4$ *or* 5, normal users have a higher probability to choose those questions for answering. In conclusion, experts are more focused on selecting questions with no previous question value.

The comparison to other models especially the ZScore, which is also used by this work, shows that precision for ZScore with 0.60 is twice as high as for the bias based model where the value is 0.28, whereas recall is almost twice as high for the bias model with 0.92 where it is just 0.47 for ZScore. As the precision value is not very high additional experiments with the answer count threshold, that means only users with at least a specific amount of answers are considered, are performed. If the required amount of answers increases, the results of the selection bias based model improve significantly to a precision of 0.7 for a required answer count per user of 100. That tendency is almost the opposite compared to the results of this work where a higher diversity related to the answer count leads to better results. It could be argued that the results get better because the higher the answer threshold the less normal users are considered and therefore the precision has to improve. The recall value is high because normally experts provide lots of answer and are therefore not affected by an answer threshold. This is also shown in the presented figure where the recall value does not change in the presented range of the answer threshold. The results also indicate that experts can be identified with only the data of the first month of their membership in the community. Furthermore, the selection bias does not significantly change over time.

3.3 Question recommendation

Kabutoya et al. [20] proposed a question recommender focused on the knowledge and interests of potential answerers. The aim is to perform better than existing category oriented solutions. Unanswered questions should be (re-)recommended and also users who have not answered any questions before should be considered for question recommendation. To accomplish this question and answer histories of each user are used via the combination of collaborative filtering and content-based filtering models.

First, Kabutoya et al. started by using a questionnaire to more than 2000 Q&A users in order to find out about user satisfaction. It turned out that about 90% of the people are unsatisfied with the answers and 50% of them are not satisfied with the quality. To improve this situation the question recommender should find suitable users to answer questions and therefore it should help to improve the response time because potential answerers would not have to look for questions they might be able to answer. This is also based on the results of the questionaire where about 80% say that they would try to solve questions related to their knowledge and interests. Another reason for the usefulness of the proposed system is the vast amount of categories and subcategories with even similar categories which all makes it difficult for users to find suitable questions.

In the proposed method, the top-n questions related the the interests of a potential answerer are ranked accordingly. In the calculation process TF-IDF is used based on the the question set for IDF calculation which is significantly different to the approach presented in this work where the IDF calculation is based on the set of user document vectors. The questions q of the question set Q are represented via keyword vectors. The probability that a question q gets answered by a specific user u is calculated via the following logistic regression model:

$$P(r = 1 \mid u, q) = \frac{1}{1 + exp(-\sum_{f \in F} \lambda_f P_f(r = 1 \mid u, q) - \lambda_0)}$$

The unknown parameters $\lambda = \{\lambda_f\}_{f \in F}$ are received via the usage of optimization techniques like quasi-Newton methods. The other parameters are f which represents a feature whereas F is the feature set. r indicates if a user replies to a question.

The feature set contains six features:

- Category based probability of question answering

$$P_{CategoryA}(r = 1 \mid u, q) = P(r = 1 \mid u, c_q)$$

  where c_q represents the category of q. This method uses the probability of a user u in being able to answer a question of a specific category.

- User based collaborative filtering

$$P_{UserCF}(r = 1 \mid u, q) = \frac{1}{|U_q|} \sum_{u' \in U_q} cos(Q_u^R, Q_{u'}^R)$$

  It is based on the recommender system of Resnick et al. [40]. It does not consider content information of the question. Q_u^R is the set of questions answered by u whereas U_q is the set of users replying to q. cos refers to cosine similarity.

- Item based collaborative filtering

$$P_{ItemCF}(r = 1 \mid u, q) = \frac{1}{|Q_u^R|} \sum_{q' \in Q_u^R} cos(U_q, U_{q'})$$

  This method is based on Sarwar et al. [41] and it also does not need question content.

- Content based filtering via the usage of answer histories

$$P_{ContentA}(r = 1 \mid u, q) = cos(q, \Theta_u^R)$$

  This method is from Mooney and Roy [32] where the terms of the question content are weighted using TF-IDF weights. That method is able to suggest new questions without any preexisting answers whereas collaborative filtering techniques are not able to do so. Θ_u^R represents the document vector of $u's$ answer history.

- Content based filtering via question histories

$$P_{ContentQ}(r = 1 \mid u, q) = cos(q, \Theta_u^A)$$

  where Θ_u^A is the document vector of $u's$ question history. This method has the advantage of the possibility of recommending questions to users who have only asked questions, but not yet answered any questions of other users.

- Likelihood of submitting a question in a specific category

$$P_{CategoryQ}(r = 1 \mid u, q) = P(a = 1 \mid u, c_q)$$

  where a indicates if a user posts a question. This method uses the information about the categories from the user's question history.

The evaluation is based on a data set of the Japanese Q&A community named Oshiete goo [80]. The top-n accuracy, which is the fraction of users to whom the right questions are suggested, is used as metric. Further, the newest question a user u has replied to is used as test data $\bar{q}_u$ whereas the older ones are used as training data of u. Via the recommendation methods the relevancy between u and $\bar{q}_u$ and all the other question which u did not answer is calculated. The recommendation is successful, if $\bar{q}_u$ is included in the top n questions recommended to u.

The results show that the proposed method performs better for top-n accuracies than the second best method which is CategoryA. For users who have not answered any questions CategoryQ performs best, ContentQ is second and the method of Kabutoya et al. is third. Recommenders which are based on collaborative filtering do not work with unanswered questions since there is no useful information to work with; This is called the "Cold-Start" problem.

Qu et al. [38] described a different approach to recommend new questions to potential answerers which is based on Probabilistic Latent Semantic Analysis PLSA [15, 37]. Because of the popularity of CQA platforms like Yahoo! Answers, which receives thousands of questions each day, it is important that the questions are found and answered quickly from users who are able to answer these questions in order to satisfy the askers. User interests are modeled via PLSA by using the user's history of previously asked (maybe mistaken and it should mean answered) questions. This is quite different to other approaches because normally the answer history is used to generate user profiles. The details of PLSA are out of scope of this work.

The Model produces a sorted list of users and the question is forwarded according to this ranking to the top-n users of it. The evaluation is based on data sets of three categories from Yahoo! Answers. All questions with only one answer are removed. In all three categories together, there are about 26,000 questions available for testing. 85% of the questions from each category are used for training of the PLSA model. The remaining ones are used as test data. Qu et al. explained that the Precision metric is not suitable in the context of CQA. This is because users who have the knowledge still can only answer a few of those many questions they would probably be able to answer and not only because there are so many questions, but also because it is difficult for potential answerers to access or simply to find those quesitons which would be interesting to them. The metric proposed by Qu et al. uses only the set of actual answerers of a specific question from the test data set. It is aimed to predict who answered the question best, based on the ground truth which is the actual best answerer. The best answerer is chosen by either the asker or by the community. Therefore, there is no guarantee that the real best answer is chosen. According to Adamic et al. [2], who investigated this question based on questions of different categories from Yahoo! Answers, the conclusion is that the choices for best answers are mostly accurate. The proposed metric is calculated as follows:

$$accuracy = \frac{|R| - R_B - 1}{|R| - 1}$$

$|R|$ represents the amount of answers from the question under consideration which is also the length of the recommendation list. R_B is the actual best answerer's rank. This approach is very different to the one presented in this work. It is the aim to recommend questions without the knowledge about the users who tried to answer it. Therefore, no such metric is used in this

work. However, the explanations from Qu et al. point out the difficulties in evaluating the results. Precision is still a metric which can be applied, but it needs to be modified as for example P@100 which could measures if the actual best answerer is within the top 100 of the ranked list and the results need to be interpreted according to the context. P@1 is definitely not very useful since there is not only one unique user who is able to answer a specific question; There are many of them - especially in a huge community as Yahoo! Answers.

As comparison Qu et al. used an implementation of cosine similarity between user and the test question:

$$s(u,q) = \frac{\sum_w tf.idf(u,w)tf.idf(q,w)}{\sqrt{\sum_w tf.idf(u,w)^2}\sqrt{\sum_w tf.idf(q,w)^2}}$$

$tf.idf(q,w)$ denotes the TF-IDF weight of a word from the question q, whereas $tf.idf(u,w)$ represents the sum of the TF-IDF weights of w from questions which the user u asked or answered. There is no indication that the user document vector based approach for the TF-IDF calculation presented by this work was used by Qu et al. [38]; It seems that the usual question set based TF-IDF calculation variant is used.
The results show that PLSA performs a bit better than cosine similarity.

3.4 Answerer prediction via Language Model, user activity etc.

Liu et al. [26] suggested a system to predict best answerers for new questions in the context of community question answering. A probabilistic framework via a combination of a Language Model and the Latent Dirichlet Allocation based on a user's answer history and his interests is used. Additionally, activity and authority of users are also part of the framework. The aim is to improve answer quality in the way that experts answer the question rather than users who randomly happen to find and pick the question for answering. Also it is desired to improve the response time via suggesting questions to the right experts and to consider the potential answerer activity status. More active users are considered to have a higher possibility in answering new questions. The user profile which reflects the expertise of a particular user on which the relationship to a new question is measured is the result of the user's best answers collection. This is very similar to the basis for the user profiles used by the proposed model of this work where different combinations of information from the question with the best answer of the user are used.

The following framework is used to determine the top-k users. Liu et al. described the likelihood that user u is the best answerer for question q based on Bayes' rule:

$$P(u \mid q) = \frac{P(u)P(q \mid u)}{P(q)}$$

$P(u)$ represents the previous probability of user u which is composed of authority and activity of that user. $P(q)$ is the likelihood that question q is created which is equal for all potential answerers. $P(q \mid u)$, which can be interpreted as the interest of user u in question q, is the probability that q is derived from $u's$ profile. The ranking of potential answerers is achieved via the score of the combination of $P(u)$ and $P(q \mid u)$.

Liu et al. assumed that a user who gives a best answer to specific questions also has a kind of interest in it. There are two models which are used to calculate the interest of a user u in a particular question q.

First, a unigram model, which is a specific kind of Language Model is used. In this model, each word of a question is treated to be generated independently. A multinomial probability distribution over words is used to represent the interest of user u in question q.

$$P(q \mid u) = \prod_{w} P(w \mid \Theta_u)^{n(w,q)}$$

Θ_u represents the profile of u which is filled with questions that u answered best, whereas $P(w \mid \Theta_u)$ is the likelihood of a specific word w to be generated out of $u's$ profile. $n(w, q)$ is the absolute term count of w in q. A simple way of calculating $P(w \mid \Theta_u)$ is to calculate the fraction of w in Θ_u, however smoothing is required to catch the cases where w does not occur in Θ_u which would turn the probability for this user to zero.
The model is calculated as follows:

$$P_{LM}(w \mid \Theta_u) = \lambda\, P(w \mid \Theta_u) + (1 - \lambda)P(w)$$

where

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

is the background Language Model which is generated on the entire question collection Q via a maximum likelihood estimation. $n(w, Q)$ is the term frequency of w in Q, whereas $|Q|$ refers to the total amount of words in Q. $\lambda \in [0,\ 1]$ is used to handle the influence from the background model in a flexible way based on each user's profile size.

$$\lambda = \frac{\sum_{w \in \Theta_u} tf(w, \Theta_u)}{\sum_{w \in \Theta_u} tf(w, \Theta_u) + \mu}$$

where μ is set to 1000 and $tf(w, \Theta_u)$ is the frequency of term w in $u's$ profile. For users with a huge profile it means that the background model will have little impact on the calculation results, whereas for users with a smaller profile the influence of $P(w)$ increases. In the approach used by this work the λ calculation is also investigated.

The second model uses Latent Dirichlet Allocation (LDA) introduced by Blei et al. [4] which is a generative model based on latent topics. It is used to overcome the lexical gap amongst user profiles and new questions. Guo et al. [11] also used LDA for their system to recommend answer providers. However, topic related techniques are outside the scope of this work.

The two interest or expertise modeling techniques are combined linearly:

$$P_{LM+LDA}(w \mid \Theta_u) = \pi P_{LM}(w \mid \Theta_u) + (1 - \pi)P_{LDA}(w \mid \Theta_u)$$

where π is simply set to 0.5.

As for the approach towards modeling user interests there are also two components in the model about the previous information from a potential answerer $P(u)$. First, the authority part is explained. Users with higher authority are considered to be more likely to produce reliable

answers. In the process of selecting best answers more authoritative users are assumed to be preferred from either the author himself or by the community voting to select best answers. This is exactly the way how best answers are chosen in Yahoo! Answers which is used to provide the evaluation data set for this work. The authority of a user u is calculated as

$$authority(u) = log(1 + numans_u)$$

The log function is used to smooth the authority of u and therefore to control the influence of the authority score for the prediction process, whereas $numans_u$ represents the best answer sum of user u. In the approach presented by this work the best results are achieved via the so called InDegree method which considers the amount of distinct people a specific user has helped and the calculated score is in the range from 0 to 1.

The second part of the prior information about a user is represented via his activity. Obviously, an asker prefers quick responses and therefore potential answerers who are more active in answering questions are more likely to answer soon compared to users who have not participated in this process for a long time.

$$activity(u) = exp^{-(t_q - t_u)}$$

t_u is the last time a user u has answered a question, whereas t_q refers to the time when the question q was submitted. The previous information about a user is calculated as:

$$P(u) = authority(u) \times activity(u)$$

A user's probability to provide the best answer to a question is the combination of interest and prior information:

$$P(u \mid q) \propto P(u)P(q \mid u)$$

According to the score of each user the users are ranked and the top ones are considered to be the potential best answerers for the particular question.

The evaluation is based on a data set from the Chinese CQA site Iask. The experiments are performed upon two main categories. Similar as in the evaluation process of this work, Liu et al. used only users with at least $N = 10$ best answers for their experiments. The difference is that in this work only 2 best answers per user are required. A segmentation tool for Chinese terms is used to segment all questions; Further stop words are removed and questions with fewer than two terms are dropped. A time stamp is used to decide between training data and test data where the data after the time stamp is used as test data. Additionally, not all questions with the best answer from a specific user are used as test questions. At most 10 of them are used. This is different to the approach of this work, because there is no limitation in this work. It could be argued that the limitation influences the prediction results in an unnatural and negative way especially related to the authority score which basically uses the best answer count.
The two test data sets for the two categories contain 3,126 and 1,693 questions. In this work much more training and test data including all available categories is used. As in the approach of this work, Liu et al. ranked all possible users according to their likelihood of answering the question and compare the result to the position of the actual best answerer. The interpretation is different because Liu et al. checked only if the actual best answerer is in the top 1, 10 or 100 of

the prediction result list and if so, the prediction is considered to be successful, whereas in this work a normalized error value is calculated and the mean reciprocal rank MRR are used too. The results show that the Language Model alone is in 11 of 14 cases slightly better than the Latent Dirichlet Allocation model. The combination of both improves the result. User activity and authority improve the results significantly. Overall, the combination of Latent Dirichlet Allocation with user activity and authority performed better than the combination of all four methods.

Li and King [24] presented a system to route questions to suitable answerers within CQA systems. A so called Question Routing (QR) framework is introduced which contains four steps: First performance profiling, second expertise estimation, third availability calculation and fourth ranking of the potential answerers.
In order to investigate the situation of the answering process within Yahoo! Answers and also from Baidu Zhidao (a Chinese CQA website) Li and King tracked 3000 questions from the time they were posted. The results show that in Yahoo! Answers about 17.6% of the questions received useful replies within the first two days which is a relatively low rate. 20% of that questions with no satisfying answers received no answers at all. For Baidu Zhidao the values are 22.7% and 42.8%. These results imply the necessity of improvements in the question answering process . QR aims to rout questions to potential answerers who are most likely to reply swiflty in a satisfactory way.

In the performance profiling phase a so called performance profile or simply a user profile is created for each user who has given at least one answer. For those users all previously answered questions are used to build the profile.

There are three methods proposed for expertise calculation. First the Query Likelihood Language Model (QLLM) with Jelinek-Mercer smoothing [50] is used.

$$E(u_i, q_r) = P(q_r \mid q_{u_i})$$

where $E(u_i, q_r)$ represents the expertise of u_i related to the new question q_r. q_{u_i} represents the collection of all previously answered questions by u_i.

$$P(q_r \mid q_{u_i}) = \prod_{w \in q_r} P(w \mid q_{u_i})$$

denotes the probability that q_r is generated out of q_{u_i}.

$$P(w \mid q_{u_i}) = (1 - \lambda)P(w \mid q_{u_i}) + \lambda P(w \mid C)$$

where C represents the entire question collection. λ is used to control the influence of the background model and it is set to 0.8 according to the results from Zhai and Lafferty.

$$P(w \mid q_{u_i}) = \frac{tf(w, q_{u_i})}{\sum_{w' \in q_{u_i}} tf(w', q_{u_i})}$$

$tf(w, q_{u_i})$ represents the frequency of term w in the user profile q_{u_i} and C respectively.

$$P(w \mid C) = \frac{tf(w, C)}{\sum_{w' \in C} tf(w', C)}$$

where $P(w \mid C)$ refers to the background model.

Based on QLLM there are two extensions which consider answer quality. In this work the answer quality is assumed to be satisfying since only best answers are considered to create user profiles. This seems to be different in the approach of Li and King, because in their approach all answered questions of a user seem to be used to create his profile. The combination of QLLM and the answer quality is calculated via linear combination using α as weighting factor.

$$E(u_i, q_r) = \alpha \cdot P(q_r \mid q_{u_i}) + (1 - \alpha) \cdot Q(u_i, q_r)$$

User $u_i's$ answer quality according to q_r is denoted as $Q(u_i, q_r)$. In the basic model the answer quality of user u_i is simply the weighted average quality of similar questions to q_r answered by this user. The similarity is calculated via cosine similarity, the questions are represented via a vector space model and the terms of the questions are weighted using TF-IDF weights. The Smoothed Model applies similarity fusion [46]. Due to data sparsity of the actual user under consideration the answer quality of alike answers from similar users is used. The similarity of two users is estimated via the amount of points the user owns, sum of answers provided, amount of best answers, total of asked questions and the amount of stars received by the user under consideration.

A logistic regression model is used to determine the answer quality of a user's prior given answers. The answer quality is based on several features of the given answers like length of the answer, ration between length of question and answer, total answers for the question, number of both up and down rates by other users for the answer, the total points of the author of the answer and the best answer ratio of the answerer.

The other main part of the ranking function is the availability estimation. Based on the assumption that a user is available when he logs in to Yahoo! Answers, a model is used to forecast if a user will log on after a new question is posted. An autoregressive model is used to tackle this trend analysis problem. $A(u_i, t)$ represents the likelihood of user u_i being available at time t where t usually represents one particular day to answer the question. $A(u_i, t)$ is 1 if on day t user u_i posted at least one answer, otherwise it is set to 0.

$$A(u_i, T) = 1 - \prod_{j=1}^{s} (1 - A(u_i, t_j))$$

T represents a period of time $\{t_1, ..., t_s\}$ for which the availability is calculated.

The potential answerer ranking is calculated as following:

$$QR(u_i, q_r, T) = \gamma \cdot E(u_i, q_r) + (1 - \gamma) \cdot A(u_i, T)$$

Availability and expertise are considered to be independent and are therefore linear combined.

The evaluation is based on a data set from Yahoo! Answers which is from the Computers & Internet category collected between April and May 2010. The data set used in this work is spanning over all categories and approximately collected over several years. A certain date is used to distinguish between archive or training and test data. The comparison is based on the actual best answerer which is the way it is done in this work. The test set includes about 1,700 questions. Answers of the training data set are marked as good if they were selected as best

answer or if they received more than 50% of rate-ups. Answers are considered to be bad if they received more than 50% rate-downs. T is set to 3 because the longest time for a question of the training data set to receive an answer is 2.16 days. According to another parameter, the records of three days are used to calculate a user's availability. As metric, MRR (mean reciprocal rank) is used.

The results show that the usage of answer quality improves QR. The basic availability model improves QLLM (MRR = 0.0389) at about 27% with MRR = 0.0494 whereas the smoothed avaiability model yields 33.68% improvement with MRR = 0.052. Both models are based on QLLM as mentioned above. The smoothed avaiability model combined with the availability component leads to the best result with MRR = 0.0541.

Li et al. [25] presented a question routing system where potential answerers are ranked via category-sensitive Language Models (LM). Since the approach presented by this work especially tries to overcome the necessity of topics or categories, there will be just a brief summary of that work.

As most of the other works mentioned above Li et al. aimed to improve CQA systems because of the vast amount of questions where it is increasingly difficult for potential answerers to find suitable questions. The evaluation process is based on a data set with more than 400,000 questions collected between June and October 2010 from the categories Computer & Internet and Entertainment & Music of Yahoo! Answers. The advantage of their test data is that there is information to each answer about the author of it, whereas in this work there is just the information about the author of the best answer.

The basic category-sensitive LM (BCS-LM) is based on questions where the asker defines the category to which it belongs. The idea is to calculate QLLM based on the same leaf category assumption which means that only users who have knowledge about the same category as the question belongs to are considered to be potential answerers. Users who belong to different leaf categories are not considered at all.

The transferred category-sensitive QLLM (TCS-LM) is based on the assumption that a user who has expertise in a certain leaf category might also be able to answer questions from similar leave categories. This means that the weight from the leave node with expertise is partially transfered to similar ones.

Additionally to these two methods, a Cluster Based LM (CBLM) and a sort of Latent Dirichlet Allocation combined with LM (LDALM) are used in the evaluation process in order to compare the methods.

The results show that for QLLM the MRR value is 0.146, for BCS-QLLM it is 0.1893, for TCS-QLLM MRR = 0.1965, for LDALM it is 0.1695 and for CBLM it is only 0.0031. This shows that the TCS-QLLM method improves the results significantly.

Liu et al. [28] aimed to find experts in CQA systems via the usage of Language Models (LM). In order to improve the answer performance of such platforms it is important that questions are targeted at user who are able to answer them. Such users have to have some expertise about the topic of the question and they are therefore to some extent considered to be experts in the domain of the question.

The evaluation is based on a data set from Wondir.com which was a CQA platform that ceased to exist a few years ago. The test data contains more than 850,000 question answer (QA) pairs. From this data five different data sets are created based on different requirements for the minimum amount of questions a user has to have answered. The data set D2 for example contains only QA pairs that are related to users who answered at least two questions. The evaluation process of this work is intended to use a similar approach. The test data for each of the data sets is created via randomly picking one question per user. The other questions are considered as training data. Via the training data sets the user profiles are created. In the evaluation process, the calculation results are compared to the users who actually answered the questions; This is the way it is done in this work, too.

New questions are matched with the user profiles. It is assumed that users who answered similar questions in the past are likely to be able to answer such questions. The user profile which is generated based on the questions a specific user has answered is considered as document, whereas new questions are treated as queries. User rankings are calculated via Language Models like the query likelihood model QLLM [31], the relevance model RM [23] and the cluster-based Language Model CBLM [27].

There are different versions of user profiles based on the used data. "All QA pairs" includes question and answer text of all previously answered questions from a specific user, "All Qs" considers also all prior answered questions, but only the question text is used, "Single QA pair" uses answer and question text from only one prior answered question and "Single Q" uses only the question text from one prior answered question. Via the last two different versions of user profiles for one user could be generated. Users are rated according to the results of the Language Model calculation based on the value of the best profile version.

The MRR values for Single QA pair and Single Q are very similar when using QLLM. The results of All Qs are a bit better than the ones for All QA pairs. The results are better when the requirements, how many questions a user has to be related to, are lower which is exactly how it was observed via experiments by this work. Since All Qs performs best the comparison of the different Language Models is based on it.
QLLM and CBLM perform better than RM and based on D2 where only two questions per user are required the MRR value for all LMs is higher than 0.11 which means that the actual answerer is within the top 9 ranked users. It is further argued why it is difficult to predict a specific answerer of a certain question. One reason is that it is very likely that there are several users who are potentially able to answer the question, but who have not actually answered it. There is no perfect user for answering a question, but many people might be able to answer it.

There is no clear information if a question of the used data set could have several answers and if there are best answers. Therefore, it is difficult to compare this work to their work since in this work the best answerer is to be predicted which is harder than to try to predict one of many answerers related to a question. One fact indicates that a question might have several answerers, because the statistic about the data sets shows that even if there are 37,723 users ("experts") only 23,949 questions are in the test data set. But [28] described that for each user one question is selected to be in the test data set.

Zhou et al. [52] addressed the problem of routing questions to appropriate answerers in fo-

rum systems with the aim that questions are faster answered with more accurate answers and therefore to increase user satisfaction. The suggested framework produces a ranked list of potential answerers from which the question is routed to the top-k users. The method uses content for expertise estimation and user relations of the forum to calculate the probability of each user to answer a given question.

A main difference to this work is that forums are more discussion based whereas Yahoo! Answers used by this work focuses on the question answer pattern. A similarity to the proposed approach of this work is that Zhou et al. mention that all new questions can be processed by their framework without prior knowledge about the topic from the question. It is also mentioned that most works in the field of question routing/recommending tend to focus on effectiveness and not so much on efficiency. In this work as in the work of Zhou et al., efficiency is to some extent also under consideration.

The first part of the answerer probability estimation is to calculate the user's expertise related to a given question via considering the user's previous participation in the forum. In the second part user authority is determined on a graph based approach as described by Zhang et al.. The two parts are combined to a final score which determines the ranking of each user.

The user expertise is calculated via three different approaches. The profile based model uses all answers given by a specific user and also the related questions to create his profile. In the thread based model every thread is treated as latent topic which is out of the scope of this work. And third, the cluster-based model, where threads with similar content are put into clusters which is also out of the scope of this work. All three models are based on the Language Model.

User authority is estimated via a modified PageRank algorithm. The links between users are weighted based on the frequency the users are helping each other. For the profile based approach the user network is built upon all threads of the forum.

The Language Model (LM) is used to calculate the expertise of a specific user. A potential answerer is represented via a multinomial probability distribution over the collection of words of his profile. There are differences to the described LM used by Liu et al. [26]. The foundation is the same:

$$P(q \mid u) = \prod_{w} P(w \mid \Theta_u)^{n(w,q)}$$

Also Jelinek-Mercer smoothing is applied and $p(w)$ refers to the same approach with the background model as in Liu et al. [26]. Furthermore, Liu et al. describe $P(w \mid \Theta_u)$ as the simple fraction of w in Θ_u, which is different to Zhou et al. where it is calculated as follows:

$$p(w \mid \Theta_u) = (1 - \lambda)P(w \mid u) + \lambda p(w)$$

with

$$p(w \mid u) = \sum_{td} p(w \mid td_u) con(td, u)$$

$p(w \mid td_u)$ denotes the likelihood that term w is produced by thread td where user u participated and $con(td, u)$ is the contribution level of u on td.

The authority score based on PageRank is used for re-ranking like $p(q \mid u)p(u)$ where $p(u)$ denotes the authority of user u and $p(q \mid u)$ is the likelihood of query q being generated out of user $u's$ profile which is the result from the Language Model calculation.

The evaluation is based on a data set from a forum called Tripadvisor [86]. The used metrics which are also considered by this work are MRR which is the mean reciprocal rank [94] and Precision@N [97] where only the top N users from the ordered list are considered to check if the actual best answerer is in that part of the list. This is a kind of modification to precision since in the case of this work there is only one known, correct user in the list and the overall precision is calculated as the fraction of correct top N rankings of the actual best answerers out of all processed questions so it is a different level of abstraction to the approach where it is checked how many of the top N entries of the sorted list are correct entries. The results are almost not comparable since the metrics are applied differently and the foundation which is a forum on the one side and CQA platform on the other side is also different.

3.5 Question routing in the context of social networks

Aardvark, presented by Horowitz and Kamvar [16], is a Social Search engine. Users enter questions which are routed within the social graph of the user and therefore trust is provided via intimacy. Compared to traditional search engines, which focus on finding the right document, Aardvark tries to find the most suitable users to answer questions.

While traditional search engines like Google, Yahoo, Bing or Baidu are based on the library paradigm, Aardvark focuses on another important principle for information retrieval called "the village paradigm" [16]. This paradigm focuses on information proliferation via face to face communication. If somebody has a question, the challenge lies in finding the right person. In difference to search within a library, where search is based on keywords, Social Search uses natural language. Questions can be much more specialized and surrounded with context information. Answers are therefore much more tailored to the needs of the questioner than using a "normal" keyword based web search engine.

In the initial procedure of a new user several indexing steps are performed. One of them takes care about the social graph of the user which represents the friendship relations. To accomplish this step, there is the possibility to sign in with an existing account from social networks like Facebook or LinkedIn which is further used to import the friend list. Aardvark's main source of new users are invitations from existing users to their friends and contacts which are not yet members.

To quantify the knowledge of a user Aardvark uses topics. Users can define their knowledge about a specific topic, their friends could also do this or it is accomplished via the topic parser which parses given information or documents from the user. Furthermore, there is a dynamic progress concerning topic rating depending on the users's behaviour within the question answering process.

A query is started by a user who asks a question via one of various interfaces. Further the question becomes normalized and it is checked to be a correct question. The so called Question Analyzer identifies the topics that are related to the question and after that the user has the opportunity to edit them. In the same time, the Routing Engine accesses the user's Social

Graph and determines a ranked list of users who could probably answer the question best. This information is sent to the Conversation Manager which organizes asking users of the list related to the Routing Policy. Finally the Conversation Manager delivers the answer to the asker and offers the possibility of direct communication between asker and answerer.

Aardvark is based on a statistical model, the so called "aspect model" [15]. It is used for question routing. There are two main points: On the one hand the probability that a user u_i will answer a question q successfully depends on if the user has enough knowledge about the topics t of which the question is about.

$$p(u_i \mid q) = \sum_{t \in T} p(u_i \mid t) p(t \mid q)$$

On the other hand, there is a probability if a user can answer a question which does not depend on the question itself, but on the relationship between asker u_j and answerer u_i:

$$p(u_i \mid u_j)$$

The result of both parts combined is the scoring function which is used to rank possible answerers:

$$s(u_i, u_j, q) = p(u_i \mid u_j) \cdot p(u_i \mid q)$$

Aardvark is learning over time through the behaviour of a user. It is also considered if a user is the only person in the group who has knowledge on a specific topic. If so, there is less confidence in the user's knowledge than if the others also know about it. There are also several other considerations about connectedness like, for example, social connection, demographic, profile and behaviour similarity.

The purpose of the Question Analyzer is quite different from a traditional search engine. It is only necessary to understand the question in a way to be able to forward it to a user who can answer it. For web search engines the understanding of the query has to lead to a document which can answer the question itself. In the case of a Social Search engine the answering is done by a human. Aardvark question analysis also tries to identify simple questions and answers them via available services. The Aardvark Ranking Algorithm does not only take into account the knowledge about topics and the relationships between users, but also the availability of them. Therefore, Aardvark tries to find an available person but in the same time load balancing is also considered that not only one person receives all questions. Other points are guidelines and frequency settings which are also considered during selecting potential answerers.

Aardvark offers several user interfaces which are based on the available communication channels like IM, email, SMS, iPhone, Twitter etc. Anyway, it turned out that the chat-like interface works best because it establishes intimacy like in a real one to one communication. During the question process, the asked user has several possibilities like to answer, to forward the question to a friend or simply to pass. There is also the possibility to just access Aardvark's "Answering" tab and to check if there are unanswered questions in the user's network.

The analysis part of Horowitz and Kamvar shows that questions are answered quite quickly. More than 87% of the questions receive an answer and more than 57% have one answer already within under 10 minutes. With Yahoo! Answers most questions are not answered within 10

minutes and on Facebook only 15.7% are answered within 15 minutes. In the evaluation it turned out that the median time to satisfactory response for the search engine Google [71] was 2 minutes, but for Aardvark it was 5 minutes.

3.6 User authority via link analysis

Jurczyk and Agichtein [19] described the problem of diverse answer quality in community question answering. Whereas some users give great answers, others abuse the CQA for advertisement and commercial stuff or they even insult the asker or other repliers. Feedback as a source to distinguish answer quality is not always sufficiently available. Therefore, Jurczyk and Agichtein proposed a link based user authority estimation for particular question categories which can be used for answer ranking. Even if answer ranking is not part of this work authority measures are an important part in ranking potential answerers too. In this work the authority calculation is based on the link structure including all users from the CQA platform rather than users linked within certain question categories.

The generation of the link structure is based on questions. A question has an author and replies which themselves are created by specific users. Therefore, there are links between the question author and the answerers. Of course users may answer several different questions and ask how many they want to and therefore, users are connected to many others who are themselves connected to many other users and so on. From this setting two specific users may be connected via several links. In the approach of this work there is always at most one link between two distinct users under consideration for authority calculation.

Jurczyk and Agichtein assumed that low quality questions will receive just very few answers leading to a low outdegree. Serious questions, on the other hand, are assumed to receive more answers with higher quality which leads to a high outdegree. Those users who answer lots of questions from serious user will therefore have a high indegree. Users who ask lots of questions act as hubs while users who answer many of them act as authorities. Based on that information the HITS algorithm [21] is applied.

For the evaluation user feedback is used to investigate if users who are estimated to be authorities really become better feedback than normal answerers. This is based on several quality measures like on the thumbs up or down function as a voting for answers or the fraction of answers from a users which are best answers. However, in this work this information is not available in the used data set. There is only the information about the author of the question, the author of the best answer and a collection of not best answers without the information about the author.

The used data set is crawled from Yahoo! Answers containing about 495,000 questions. In this work a data set with more than 3 million questions is used. The results show the effectiveness of HITS for some categories while in others it does not perform so well which is explained by the probability of the existence of local structures in the graph.

Zhang et al. [51] studied expertise in online communities such as the Java Forum based on social network analysis. The algorithms which are used for ranking include PageRank, which is investigated by this work too and HITS. The aim of their work is to help finding users with

the right expertise to answer questions. It is also mentioned that expertise of a person does not necessarily imply that the person is an expert, but rather he has some knowledge in a certain domain which is also referred to as expertise as a relative concept [1]. Everybody knows something, which is in detail investigated by Adamic et al. [2], but only a few people are real experts. CQA platforms require all users to share their knowledge, because otherwise the experts would be completely jammed with the huge amount of questions.

Similar as Jurczyk and Agichtein, Zhang et al. describe the network based on the asker replier relationship, the so called post-reply network, within the context of thread based discussions. There are some noteworthy features of it: The links between users are not intentionally created like connections within social networks like Facebook [68] or Google+ [72]. They indicate that the users have similar interests. There is still more information in the link between asker and answerer. The answerer can be assumed to have better knowledge about the subject of the question than the asker, otherwise it would not be possible to create a useful answer. Of course this is quite simplistic. Communities are much more complex. Answerers do not necessarily indeed answer the question. They could ask for clarification of the question itself, or they could extend the question. There are also users who abuse these communities for advertisement and so on. Zhang et al. also mention the possibility of weighting the links between the users. Community members who are often selected positively are regarded to be prestigious. This becomes even more valuable if positive choices are not reciprocated [47].

In the empirical study conducted by Zhang et al. a sub forum of the Java Forum dealing with general Java related programming questions with more than 333,000 messages in nearly 50,000 threads is used. This data is used to create a network which contains more than 13,000 nodes and more than 55,000 edges. The network is characterized via a bow tie structure. There are lots of users in the Java Forum, however, neither do all of them ask nor do all of them answer questions. This can be reflected via the bow tie structure, first presented from researchers at IBM, AltaVista and Compaq. It is based on the assumption that the web consists of four components which are Core, In, Out, "Tendrils" and "Tubes" as in Broder et al. [7]. Applied to CQA the Core is the group of users who often help one another and who are therefore tightly connected via question answer links. In fact, every user can reach every other user by following this link structure within the core component. Users from the In group normally only ask questions while users from the Out fraction normally only answer questions from Core users. "Tendrils" and "Tubes" are components containing users who answer questions posted from users of the In group or whose questions get answered only by Out users. These users are therefore only connected via users from the In and/or the Out group or both, but not with users from the Core. According to this classification, the Java Forum has many users in the In group where more than 50% participate and a much smaller Core where only about 12% of the users are members. This means that only few people actively ask and answer each others questions. 13% of the users of the Java Forum only answer questions. This shows that CQAs, like the Java Forum, are less based on reciprocity, but they are more like places for people who ask for help from volunteers. A further characteristic from the Java Forum is that users with lower expertise are more likely to answer questions from askers with lower expertise. Top answerers on the other hand answer questions from everyone. This outcome also supports the idea on which this work calculates the authority of a user which can also be seen as a kind of quality of knowledge. In this work author-

ity is calculated with several approaches: One of them is PageRank where users are rewarded if they answer questions of other users with high authority.

The ranking algorithms presented by Zhang et al., used to calculate the expertise level of the users, are described in the following section. First, basic statistical measures are considered. It is assumed that users who answer lots of questions of a specific topic or category have much knowledge about it except spammers or trolls of course. This leads to the "AnswerNum" measure which simply counts the questions answered by a specific user. In this work we apply two forms of this measure, namely BAC which is the overall count of best answers given by a specific user and CBAC which is the same, but calculated based on leaf categories.

The "Indegree" measure or indegree of a node is different in the way that it counts the number of distinct people a user helps. Users with high "AnswerNum" may have cheated in receiving a high score simply by reciprocal answering a friend's questions. "Indegree" on the other hand rewards users who answer questions from lots of different askers with a higher score. "Indegree" is also used by this work.

"Z-score" is another measure based on the assumption that users who answer lots of questions have high expertise, while asking many questions indicates the lack of knowledge.

$$z = \frac{a - q}{\sqrt{a + q}}$$

where a is the number of answers and q is the number of questions from a specific user. For users with equal answer and question count, the z-score will be zero, for users with more answers than questions it will be positive and for users with more questions than answers, the z-score will be negative. The z-score is applied to the amount of answers and questions from a user as "Z_number" and also to the amount of users the current user has replied to and received answers from which is denoted as "Z_degree". The "Z_number" which is referred to as "ZScore" is used in this work too.

The "ExpertiseRank" algorithm is intended to improve the weakness of the simplistic z-sore approach. Z-sore does not distinguish between a user who helps for example 100 Java learners and another user who helps 100 Java experts but it can be reasonably argued that the latter one has greater expertise. Zhang et al. propoed "ExpertiseRank" which is based on PageRank in oder to consider who a user helped. For example, if user A helps user B and user C helps B then the expertise level of C is based not only on B but because B has helped A therefore C's expertise also depends on A. This way expertise is propagated throughout the question-answer link network.

$$ER(A) = (1 - d) + d(\frac{ER(U_1)}{C(U_1)} + ... + \frac{ER(U_n)}{C(U_n)})$$

where $ER(A)$ denotes the ExpertiseRank score of user A and $U_1...U_n$ represent the askers to whom user A replied. $C(U_i)$ represents the count of users who answered questions of user U_i. The damping factor d is as usual set to 0.85. The algorithm is applied via an iterative calculation. Zhang et al. describe several possibilities of weighting the edges of the expertise network, however, there were no improvements so the network used is unweighted. In this work where it is called "user authority" instead of expertise, the calculation via PageRank is based on the question - best answer network rather than on all question answer links.

HITS (Hypertext induced topic selection) [22], as already described above is another link based ranking algorithm. In the work of Zhang et al. the authority score of HITS "HITS_Auth" is used to represent a user's expertise.

For the evaluation process a "gold standard" needed to be created because there is no expertise ranking data from the Java forum available for comparison. Therefore, independent consultants rated 135 randomly selected users who have at least 10 posts because otherwise it would not be possible for human raters to judge. The raters decided between 5 distinct levels of expertise, because it is quite difficult to rate 135 users in a list based on lots of posts of each user. As evaluation metrics Spearman's rho and Kendall's tau are used [10, 14] to calculate the correlation between human ratings and the results produced by the proposed algorithms. Additionally, a so called "TopK" metric is used where a Kendall's tau is calculated only for the top 20 ranks. Before the actual comparison of the algorithms 11 of the 135 samples were removed because the human raters classified them too different or have not had enough data to rate them at all. So a weak point is that there are only 124 users in the test data set.

The correlation between human ratings and the different algorithms shows for all metrics and for all algorithms a correlation value greater than 0.6. This indicates the effectiveness of the algorithms to calculate user expertise or authority based on structural information.
The results show that the Z_num and the Z_degree performed best for all three metrics. The PageRank based ExpertiseRank follows each time on the third position even if there is almost no difference to the left algorithms. AnswerNum, Indegree and HITS_Auth are almost equal. The differences between all of the algorithms is quite small. Only the z-score based algorithms perform better for all used metrics. The more sophisticated algorithms do not outperform the simpler ones, however, the approach presented by this work which uses PageRank to estimate the authority of users has the advantage that it is quite difficult to manipulate the score and therefore it is more robust than simple z-score measures.

Bouguessa et al. [6] used Yahoo! Answers in the context of finding authoritative users. Besides the link based approaches to calculate scores in order to produce ranked user lists, there is still the question how many of them that should be considered as authorities.

According to a simplification of Zhang et al., Bouguessa et al. describe three types of users in CQA platforms which are askers only, answerers only and users who both ask and answer questions. Repliers usually do not expect any "return on their investment" [6, 45]. It is obvious that an asker prefers answers from experts in order they can rely on the answer. It is important to identify authoritative users with high expertise because the knowledge about authorities in CQA platforms like Yahoo! Answers can be used to forward new questions to these experts in order to improve the response time and also to enhance the answer quality and therefore to increase the satisfaction of the askers.

In the following section, Bouguessa et al. describe the different link based approaches to determine the relative authority score of each user. First, PageRank is discussed. Based on the example by Zhang et al., which states that in case B is helping A and C is helping B then C has to receive a better score since C helps someone who himself is helping by answering the question of A. Well, this sounds promising, but Bouguessa et al. point out that especially for Yahoo! Answers this is not usually true. The example to support this statement is that if user

B answers a question of user A from the category "Programming & Design" which is in detail about Java and user C answers a question from B which is in the same category, but about C++, then it is not valid to say that user C is more expert than user B because their knowledge is simply different. It is further mentioned that the example above is frequent in Yahoo! Answers because the main categories cover large fields. Therefore, PageRank is supposed to be more suitable in communities with one coherent domain such as the Java Forum as investigated by Zhang et al..

Another approach is HITS introduced by Kleinberg [21] as already mentioned above. The speciality of this approach are so-called hubs which are in fact link collections which point to authorities. In the context of CQA askers act as hubs while users with best answers are considered as authorities. Experiments of Bouguessa et al. show that HITS does not yield satisfying results with test data from Yahoo! Answers. The following example, based on the work of Borodin et al. [5], explains why HITS is not suitable to calculate the authority score in the context of CQA. H refers to hubs which ask questions and A refers to authorities which answer questions. According to the HITS algorithm the nodes A1 to A6 of figure 3.1 receive high authority scores

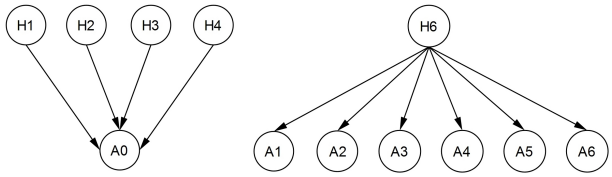

Figure 3.1: Negative example of HITS

because in this example H6 is the best hub. If there are more authorities (A1 to A6) than hubs (H1 to H4), which is here the case, then all the weight is given to the authorities (A1 to A6) while A0 is weighted zero. The reason is that H6 is assumed to be the best hub thus giving more weight to the authorities it links to. It is obvious that the authority A0 should be higher ranked than the other authorities (A1 to A6) because A0 answers questions of four different users (hubs H1 to H4) while the other authorities answer each just one question, but HITS ranks the authorities different and therefore it is not suitable to determine authorities in the context of CQA.

Next, Bouguessa et al. investigate z-score, or more specific the "Z_number" introduced by Zhang et al. which is already explained above. The reason why z-score is not an appropriate authority estimation is explained by the same example as given in relation to PageRank. In a big domain like "Programming & Design" one user might have knowledge about one subject where he answers lots of questions and in the same time this user might lack of knowledge about another topic in the same domain and therefore he asks many questions. Via z-score such a user would have almost no authority, but in fact this user might be a good authority for one specific topic.

InDegree, which is also mentioned by Zhang et al., is best suitable for authority estimation of user from Yahoo! Answers according to Bouguessa et al.. To measure the authority of a user A the score is calculated by simply summing up the amount of people which are referred to as in-links of A that user A has answered questions from. Users with a high in-link count are therefore likely to be authorities. Even if InDegree is quite a simple approach, it represents the choices

that people make about answers from others. By selecting best answers, it is indicated that the author of the best answer provided useful, interesting and a kind of authoritative information. A link between an asker and a best answerer affirms the quality of the answer. InDegree, used by Bouguessa et al., is a simple best answer count per user. This is clearly vulnerable to spam. This is mentioned via the hint that the community usually identifies such spammers which are then removed from moderators of Yahoo! Answers. The InDegree version used by this work uses at most on link from one specific asker Q to one specific answerer A, no matter how many questions of Q have been answered by A which means distinct inlinks and therefore manipulation is more difficult.

Based on the best answer count per user Bouguessa et al. proposed a category based model to identify authoritative users. Category based approaches are outside the scope of this work.

CHAPTER 4

Answerer Ranking Framework

Expertise or Knowledge, Authority, and Availability (EAA) are the three core components in the process of ranking potential answerers. In this chapter a framework is presented which is used to combine the various realizations of the mentioned components. Further, this framework is the foundation for the evaluation presented in chapter 6. There are several attempts to achieve such a ranking as described in chapter 3.

Jiao et al. [18] use a cascading approach to combine the expertise estimation which is accomplished via the Vector Space Model with TF-IDF weights and the user authority which is based on a modified PageRank calculation. User activity is not part of their model. In this work the TF-IDF weight calculation is improved, as shown in the evaluation 6.

Liu et al. [26] combine the Query Likelihood Language Model with Dirichlet smoothing (QLLM-D) with Latent Dirichlet Allocation (LDA) for knowledge estimation. Further, they calculate user authority as the logarithm of the best answer count. User activity is, simplified, the difference between the question date and the date a specific user has lately answered a question. Authority and activity scores are multiplied. The result is multiplied with the knowledge estimation score. The difference to this work lies mainly in the user activity consideration, where in this work inactive users are excluded before the actual calculation takes place. Also the authority part is different, because by this work there are several possibilities offered to consider user authority. This work does not use LDA, but there is the option to choose between the Vector Space Model and QLLM with several smoothing methods for knowledge estimation.

Li and King [24] calculate the knowledge estimation of a specific user via considering all previous answers given by that user. Therefore, they combine QLLM based on Jelinek Mercer smoothing with an answer quality component. In this work only best answers are considered for user profile generation; Therefore, answer quality is not an issue. User availability is based on a the user log in time. It is aimed to forecast when the user will be online again. Knowledge estimation and user activity forecast are combined via linear combination. The main differences to this work are the different smoothing method for QLLM, the lack of user authority estimation in their approach and the prior removal of inactive users which is used by this work.

Zhou et al. [52] introduce a forum based approach of routing questions to potential answerers. Basically, they use QLLM with Jelinek Mercer smoothing in a linear combination with a PageRank based authority score. The lack of user activity consideration is definitely the most important difference of their approach compared to this work.

Jurczyk and Agichtein [19] use HITS as a link based user authority estimation in the context of determining answer quality. As Bouguessa et al. [6] pointed out, HITS is not suitable 3.1, especially in the context of Yahoo! Answers it is not used by this work. Zhang et al. [51] describe several link based measures for expertise/authority estimation like ZScore and PageRank which are also used by this work. Bouguessa et al. [6] compares several authority estimation methods like HITS, PageRank, ZScore and InDegree. InDegree is mentioned to be most suitable. Except HITS they are all used in this work too.

Compared to all of these approaches of finding suitable answerers for new questions the most significant difference is the "Activity Filter" presented by this work which applies user activity via a cascading approach where inactive users are removed prior to the actual calculation. Further, the framework structure, which offers flexible adjustment of authority measures and expertise estimation methods related to the context in which this system is applied, is also new.

Figure 4.1 shows the structure of the framework. The "User Profile Generator" builds the user profiles with an optional component which is intended to consider changing user interest over time. The "Activity Filter" removes inactive users prior to the actual calculation of the ranking of potential answerers for a new question. The "Authority" interface offers several authority estimation methods. Via the "Expertise Estimation" interface it can be chosen between the "Vector Space Model" and the Query Likelihood Language Model (QLLM) as methods for user knowledge estimation. The "Question Value" component just plays a minor role, however, there is still potential to improve it. The details of this components are described in the following sections.

4.1 Data Model

Before the actual components are explained the data model, on which these components depend, is described. Basically, there are users, questions and the according answers.

- A user refers to a member of a Question and Answer (Q&A) community. He can ask and answer questions. The time stamp when he gave his last best answer is used to determine his activity which is used by the Activity Filter.

- A question which is asked by a user who is referred to as author of that question has several attributes. The subject contains the actual question. The content offers additional explanation about the context of the question. Further, questions receive answers. From these answers one is selected to be the best one.

- An answer obviously refers to a question. A best answer is the answer of all answers a particular question received which is selected, by either the author of the question or the community, to be most appropriate.

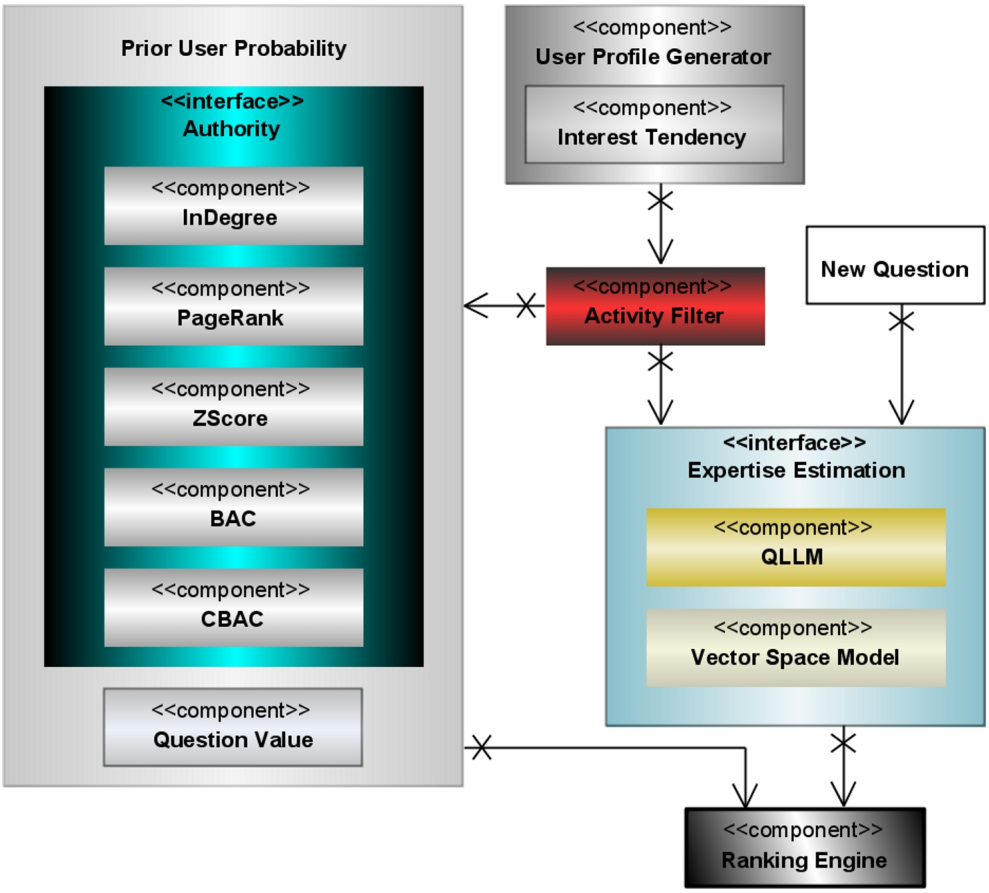

Figure 4.1: Answerer Ranking Framework

The training data set contains the best answered questions of the users which are considered for the ranking calculation. It is a prior information source for the prediction process, a kind of history about the members of the Q&A community. The data is expected to be available in an XML format.

The test data set contains all questions which are asked after the last question of the training data set. These questions are used to simulate new questions which are used for the evaluation of the framework which is presented in chapter 6.

4.2 User Profile Generator

The User Profile Generator builds user profiles with data from the training data set. Each profile of a user contains a document vector which is filled with words from the best answers given by that user. Via tagging only nouns and foreign words are considered and stemming it is aimed to decrease the amount of data in order to improve accuracy and to decrease processing time. Details about tagging and stemming are provided in chapter 5. In addition, there are

several parameters to restrict the data from which the document vectors are generated. They are described in detail in chapter 6. Furthermore, user profiles contain a categorized best answer count (CBAC) which counts the best answer given by a specific user based on the category of the related question. The best answer count (BAC) counts all best answer provided by a specific user. In addition, there is a variable which stores the user's last activity.

4.2.1 Interest Tendency

Interest Tendency (IT) aims to consider the possibility that a user changes interests over time. It is assumed that the last few answers are more important in order to estimate the knowledge than, for example, the best answers given a year ago. The second thought is that people tend to forget over time. It is not entirely clear, but probably it is a combination of both, changing interests and forgetfulness, which are assumed to improvement the answerer prediction process, if it is applied appropriately. The term frequencies tf of best answered questions by user u, which are added to $u's$ document vector, are penalized. It is calculated as follows:

$$tf = \begin{cases} tf \cdot \frac{\delta}{q_{age}} & \text{if } \frac{\delta}{q_{age}} \leq 1 \\ tf & \text{otherwise} \end{cases}$$

The empirical parameter δ is set to 100, which means that the best answers given within approximately the last 100 days, are not penalized. The question age q_{age} specifies the date when the question was asked.

4.3 Activity Filter

The Activity Filter (AF) is used to filter out low-activity users who have not contributed via giving best answers within a specific period of time. Normally, users who gave best answers within the last week, or up to the last month are considered to be potential answerers for new questions. There are two positive effects expected by such a filter: First, users, who have been very active a long time ago, but who are almost not participating in the question answer process any more, would heavily affect the prediction result in a negative way. Such users would be ranked quite high, but they would almost never answer any of those questions suggested to them. This would be bad for response time and for user satisfaction. Therefore, they are not considered for the actual prediction process. However, if they start giving best answers again, they would be considered in the future, since the user profiles would be updated from time to time. This updating process is not part of this work.

The second point is that because of the huge community, especially the one of Yahoo! Answers, with millions of members the computation of the ranked list of potential answerers is expensive. The Activity Filter can effectively reduce the processing time via exclusion of inactive members.

4.4 Expertise Estimation

In general, the expertise estimation matches user profiles and a new question and estimates the probability of users in being able to answer the question. Via the Expertise Estimation interface the Query Likelihood Language Model (QLLM) and the Vector Space Model (VSM) can be selected to model user expertise, or, as it is called by other related works, user interest in specific questions. Bayes' rule:

$$P(d_{u_i} \mid q) = \frac{P(q \mid d_{u_i})P(d_{u_i})}{P(q)}$$

as already explained in chapter 2, section 2.1.4, with the only difference, that document d is replaced with the user profile d_{u_i} of user u_i, is applied. The probability $P(q \mid d_{u_i})$, which can be seen as approach to model the query generation process is calculated via QLLM and VSM.

4.4.1 QLLM

The Query Likelihood Language Model, as described in section 2.1.4, is applied as follows:

$$\hat{P}(q \mid d_{u_i}) = \prod_{t \in V_q} \hat{P}_{MLE}(t \mid M_{d_{u_i}})^{tf_{t,q}} = \prod_{t \in V_q} \left(\frac{tf_{t,d_{u_i}}}{L_{d_{u_i}}} \right)^{tf_{t,q}} \tag{4.1}$$

The background LM is calculated via Maximum Likelihood Estimation as follows:

$$\hat{P}_{MLE}(t \mid M_c) = \frac{cf_t}{L_c}$$

where M_c represents the LM based on the entire collection of user profiles and L_c is the amount of tokens from the entire collection. Based on these equations there are three smoothing methods applied, as also described in section 2.1.4.

Jelinek Mercer Smoothing with Query Length

Jelinek Mercer smoothing, which is also referred to as Linear Interpolation [30], is applied as follows:

$$\hat{P}_{JM}(t \mid d_{u_i}) = \lambda \hat{P}_{MLE}(t \mid M_{d_{u_i}}) + (1 - \lambda)\, \hat{P}_{MLE}(t \mid M_c) \tag{4.2}$$

Via equation 4.2 applied to equation 4.1 the likelihood of user u_i being a suitable answerer for question q is calculated as follows:

$$\hat{P}_{JM}(q \mid d_{u_i}) = \prod_{t \in V_q} \hat{P}_{JM}(t \mid d_{u_i})^{tf_{t,q}}$$

The Query Length impact is described in the following section. He and Ounis [13] mention in their work, referring to Zhai and Lafferty [49], that the query length is important in the adjustment of smoothing in the context of Language Models. In this work the query length is composed of all parts of the questions except the answers of course. The words are stemmed and

stop words are removed. The length represents the remaining number of words (also duplicates). According to Zhai and Lafferty [50], which is very similar to, but more extensive than Zhai and Lafferty [49], average short queries contain about 2.5 terms, whereas long queries contain between 50 and 60 terms. In their work stop words are not removed since they argue that language modeling methods should take care of them, but stemming is applied. Here it is argued, that this would unnecessarily waste processing time, therefore stop words are removed.

Based on the findings from Zhai and Lafferty [50], where in the context of Jelinek Mercer smoothing, as already mentioned in chapter 2, section 2.1, a very little smoothing is applied to short queries with $\lambda = 0.9$ and much more smoothing for long queries with $\lambda = 0.3$. A function to consider the query length is developed as follows:

$$\lambda_{L_q} = \frac{\lambda_1 \cdot \chi}{\chi + L_q}$$

where λ_{L_q} refers to the amount of smoothing depending on the query length L_q. λ_1, which is set to 0.99, defines the smoothing for a query of the length 1. The empirical adjustment parameter χ is set to 300 in order to avoid that the value of λ drops too fast as the query length increases.

Witten Bell Smoothing

Witten Bell smoothing, as described in section 2.1.4, is applied in the following way:

$$\lambda_{Witten} = \frac{\sum_{t \in d_{u_i}} tf_{t,d_{u_i}}}{\sum_{t \in d_{u_i}} tf_{t,d_{u_i}} + N_{DOC}} \tag{4.3}$$

Via equation 4.3 applied to equation 4.2, which is further applied to equation 4.1, the likelihood of user u_i being a suitable answerer for question q is calculated as follows:

$$\hat{P}_W(q \mid d_{u_i}) = \prod_{t \in V_q} \hat{P}_W(t \mid d_{u_i})^{tf_{t,q}}$$

Dirichlet Prior Smooting

Last but not least, Dirichlet Prior smoothing, as described in section 2.1.4, is applied. It is calculated as follows:

$$\hat{P}_D(t \mid d_{u_i}) = \frac{tf_{t,d_{u_i}} + \gamma \hat{P}(t \mid M_c)}{L_{d_{u_i}} + \gamma} \tag{4.4}$$

The empirical adjustment parameter γ is set to $16,000$ which is quite high compared to the value used by [50], but it performs best. Equation 4.4 is applied to equation 4.1. The likelihood of user u_i being a suitable answerer for question q is calculated as follows:

$$\hat{P}_D(q \mid d_{u_i}) = \prod_{t \in V_q} \hat{P}_D(t \mid d_{u_i})^{tf_{t,q}}$$

4.4.2 Vector Space Model

The Vector Space Model (VSM), as described in section 2.1.2, is used as a kind of baseline approach. However, there is an adaption to the way it is applied by Jiao et al. [18] where the TF-IDF calculation, as described in section 2.1.1, is based on the question collection. In this work it is based on the user document vector collection. It is assumed that this more intuitive calculation leads to improvements which is investigated in chapter 6. The probability of user u_i being a suitable answerer for question q is calculated via cosine similarity as follows:

$$P_{VSM}(\vec{d}_{u_i}, \vec{q}) = \frac{\sum_{j=1}^{N}(w_{j,u_i} \cdot w_{j,q})}{\sqrt{\sum_{j=1}^{N} w_{j,u_i}^2} \cdot \sqrt{\sum_{j=1}^{N} w_{j,q}^2}}$$

The weight of each term w_{j,u_i} and $w_{j,q}$ is respectively calculated via TF-IDF which is based on the user collection.

4.5 Authority

User Authority is a measure to estimate the trustworthiness, the reliability and especially the quality of answers which can be expected of a specific user. Therefore, users with high authority are assumed to give more satisfying answers and therefore also more often best answers compared to users with low authority score. It is one part of information about a user which is available before a new question arrives and is therefore called prior probability of user u. The other parts of prior information are user activity, as described in the next section and the Question Value (QV), which plays only a minor role in the context of this work, because the data set does not provide sufficient information to model QV more sophisticated.

The Authority interface offers several realizations of user authority estimation like InDegree or PageRank which are explained in the following sections that can be selected in accordance with the context in which this framework is applied. Bouguessa et al. [6] highlight the pros and cons of the different approaches quite well. Their conclusion indicates that simple approaches are sufficient and are even able to outperform more sophisticated methods.

4.5.1 InDegree

InDegree is mentioned by Zhang et al. [51]. Bouguessa et al. even denote it to be most suitable to model user authority. The score is calculated by simply summing up the amount of distinct people which are referred to as in-links to whom the current user has given best answers. Users with a high in-link count are likely to be authorities. Best answers represent the choices that people made about answers from others. By selecting best answers, it is indicated that the author of the best answer provided useful, interesting and a kind of authoritative information. A link between an asker and a best answerer affirms the quality of the answer.

$$InDegree(u_i) = \log\left(\sum inlink(u_i)\right)$$

The higher the value the more authority is assumed. At most one link from one specific user is considered, even if the current user gave several best answers to questions of that user.

4.5.2 PageRank

PageRank, as described in section 2.1.5, uses the link structure created via the question answer links in a more sophisticated way than InDegree. Users, who both ask and answer questions, connect the link structure so that a graph can be derived. However, as Bouguessa et al. pointed out, it is not always a valid assumption in the context of Yahoo! Answers to assume that a user A who helps user B has more knowledge than B, but he could also have different knowledge. The effectiveness of PageRank is investigated via the following iterative calculation:

$$PR(u_i, t = 0) = \frac{1}{N}$$

It allocates an initial probability to each user. The number of users considered for the PageRank calculation is denoted as N. The parameter t refers to the $t-th$ iteration.

$$PR(u_i, t + 1) = \frac{1 - d}{N} + d \cdot \sum_{u_j \in M(u_i)} \frac{PR(u_j, t)}{L(u_j)}$$

The PageRank value of user u_i, after iteration $t+1$ is performed is denoted as $PR(u_i, t+1)$. The set $M(u_i)$ refers to the users that are the authors of the questions to which u_i gave the best answers. They are considered as $u_i's$ in-links. As already indicated by "set", it does not contain duplicates in order to avoid misuse between two users via answering many questions reciprocally which would affect the authority score dishonestly. The number of outgoing links of u_j to which $PR(u_j, t)$ is equally distributed is denoted as $L(u_j)$. The damping factor d is set to 0.85.

4.5.3 Best answer count

The best answer count (BAC) is simply the amount of best answers given by a specific user u_i normalized by the global best answer amount. It is used as a kind of baseline.

$$BAC(u_i) = \frac{\sum ba(u_i)}{\sum_{j \in U} \sum ba(u_j)}$$

The set of users is denoted by U, whereas ba is the abbreviation for best answer.

4.5.4 Categorized best answer count

Despite the fact that this works aims to avoid question categorization in order to increase user convenience, the categorized best answer count (CBAC) provides a good comparison to the determined best setting. CBAC is almost the same as BAC except that it counts the best answers related to leaf categories. Leaf categories are the categories which are not further divided into sub categories.

$$CBAC(u_i, c) = \frac{\sum ba(u_i, c)}{\sum_{j \in U} \sum ba(u_j)}$$

A best answer of a specific category c of user u_i is denoted as $ba(u_i, c)$. CBAC is also normalized via the global best answer count.

4.5.5 ZScore

ZScore, as introduced by Zhang et al. [51], is also used to estimate user expertise. It is described in section 3.6.

$$ZScore(u_i) = \frac{\sum ba(u_i) - \sum q(u_i)}{\sqrt{\sum ba(u_i) + \sum q(u_i)}}$$

A question of user u_i is denoted as $q(u_i)$. ZScore penalizes asking questions. As also described by Bouguessa et al. it is not a good measure for authority, because, similar as for PageRank, it is not generally valid to assume that asking questions reduces authority. A user might have great knowledge in one specific domain, whereas he is very eager in asking questions about another topic. In such a case ZScore would almost be zero or even negative.

4.6 Question Value

The Question Value (QV) indicates the value of those questions which are answered best by user u_i. Since the used data set does not provide much data to create a more sophisticated model it is simply calculated as follows:

$$QV = \log\left(\sum_{j \in Q_{u_i}} answers(j)\right)$$

The Question Value is calculated as the natural logarithm of the total amount of answers that the questions answered best by u_i, received. The set of question, which received best answers from u_i, is denoted as Q_{u_i}, whereas $answers(j)$ refer to the amount of answers a specific question received. It is only used to decide the ranking between users who receive the exact same score by the used ranking methods.

4.7 Ranking Engine

The Ranking Engine (RE) combines the different measures and produces a list with the ranking of potential answerers. The question could be forwarded to the top N users of that list, for example 10, since they are most likely of being able to answer the question. Alternatively, in order to avoid jamming the top users, the question could be suggested randomly to 20 out of the top 50. However, this is beyond the scope of this work.

The aim of the ranking engine is to predict the best answerer. Since this is not a deterministic process, which means that there are potentially many users who are able to give a satisfying answer, or even a better answer than the actual best answerer, the result of the Ranking Engine is a ranking of users who are available who have some authority and who have some kind of knowledge related to the question.

CHAPTER 5

Implementation

This chapter describes the implementation of the framework which is explained in the previous chapter. First, a brief overview of the used technologies is given. The following part deals with data cleansing of the used data set via, for example, stemming and stop word removal. Next, the user profile generation process, which is realized via the User Profile Generator component, is explained. Finally, the calculation of the ranked list of potential answerers via the Ranking Engine, which is realised via multi-threading, is described.

5.1 Used Technology

As development environment Eclipse is used in combination with the programming language Java. The data set, which is encoded in the XML format, is processed via SAX, an XML parser. The evaluation output is gathered via LOG4J which is a Java based logging framework [103].

5.1.1 Eclipse

Since Eclipse [67] is freely available and has been successfully used throughout the study, it seems appropriate to be used for this work, too. There are frequently new versions of this platform available. For this work Helios and Indigo are used. Besides, it supports several programming languages, also Java, of course.

5.1.2 Java

The programming language Java [74], version 1.6 is used here, because it is a modern, state of the art programming language for Object Oriented Programming (OOP).

5.1.3 XML

The data set is encoded via eXtensible Markup Language (XML) [105]. It is a data format which is both, human and machine readable.

5.1.4 SAX

According to the data model defined in 4.1, the data of which the user profiles are to be built is expected to be available in an XML format. In order to read that data a SAX parser is used (Simple API for XML) [82]. According to [82], it is "an event-based sequential access parser". Since the data file may be huge a sequential parser seems to fit best. It was also successfully used during the university education. The XML DOM (Document Object Model) parser [66], which builds an object tree and the StAX parser (Streaming API for XML) [85] where also considered.

5.1.5 Logging

In order to collect and persist the data produced throughout the evaluation process, LOG4J [78], which is an easy to use logger from "The Apache Software Foundation" [59], is applied.

5.2 Implementation

The implementation description organisation is based on the package diagram of the project 5.1. The main packages are "data" which deals with data cleansing and data selection, "learning" which contains the "UserProfileGenerator" and the prediction package which contains the "RankingEngine".

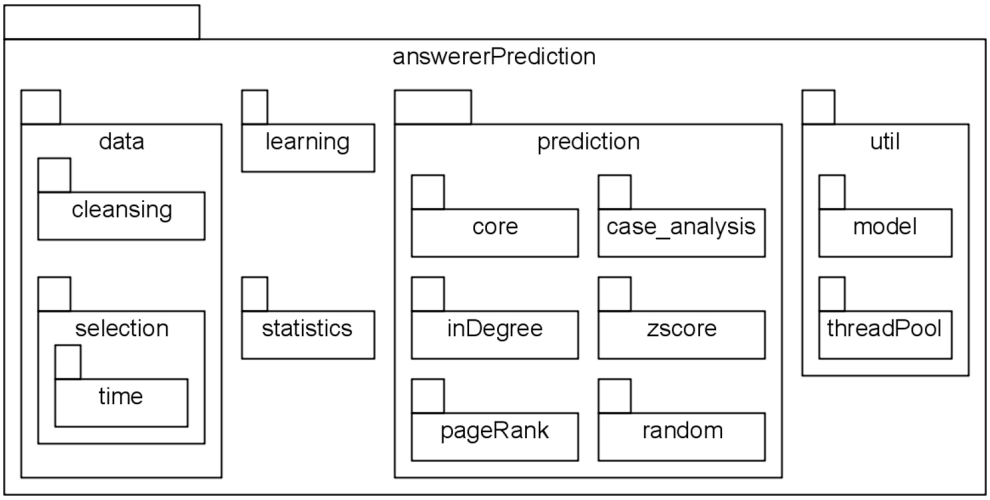

Figure 5.1: Package Structure of the Implementation

5.2.1 Data cleansing

Data cleansing [29, 102] is used to remove or correct corrupted data records. In the context of this work data records are questions. If, for instance, the question author or the best answerer identification are missing then this particular question is dropped. Further, questions which

are not specified to be written in English, are also dropped. Question data, which is not used throughout the best answerer prediction process, is also removed.

First, the text of the question records is altered via a regular expression which changes all upper-case letters to lower-case letters. Next, specific tags and symbols are removed. All character sequences with only one or two tokens are removed. Finally, white spaces are reduced in order that there is only a single space between two character sequences. After that procedure, stop words are removed via tagging. At the last step, Stemming is applied.

Stemming

For Stemming, the Java version of the SnowballStemmer [83] (English language) is used. It is aimed to merge words which are very similar in meaning, but slightly different in spelling. The process of Stemming is iterative. Three iterations are performed where each time, according to the stemming rules, the words where applicable, are reduced.

Stop word removal via Tagging

Even after Stemming there are still many almost meaningless words in the question and answer text of the question record. Since they are valueless for the actual prediction process they need to be removed in order to increase processing speed and also to improve prediction accuracy. Via the Standford-PosTagger [84] (left3words-wsj-0-18.tagger) all words that are not nouns or foreign words are discarded.

The whole cleansing process is realized via parallel processing of question records. Multithreading is applied the same way as in the ranking calculation where it is described in detail which follows later on in this chapter.

5.2.2 Data selection

The data selection is only used for the evaluation of the framework since the distinction between training and test data would not be necessary for normal operation. The evaluation process needs training data to build the user profiles and test data to investigate the efficiency of the described models. The user profiles are based on best answered questions of a particular user. In order to create profiles, users have to provide at least $n = 1$ best answers. Other users are not considered for the prediction process since there is just no data upon which a ranking could be established.

As figure 5.2 shows the training data contains the majority of the data, whereas the test data contains only the questions from the "current" day, which means that the test data set contains the questions from the last day from the overall data set. "Current" indicates that the system tries to simulate that the "new" questions from the test data set are considered to be questions from "today" where suitable answerers need to be found. The test data set contains all questions from the particular day, whereas the training data set contains only those questions where the best answerer of a particular question is the best answerer of at least n questions from questions within the training data set.

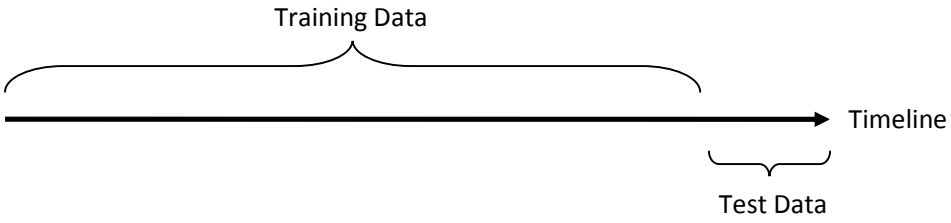

Figure 5.2: Training and test data related to time

5.2.3 User Profile generation

The "User Profile Generator" component which is located in the "learning" package, takes the training data set as input and creates user profiles based on the best answered questions of each user. There are several possibilities which differ in the parts of the question that are considered for profile generation. They are described in the course of presenting the evaluation in chapter 6. If the Interest Tendency feature is active the term count of each distinct term from the question text is penalized as described in 4.2.1.

For each question its content, which is represented by words, is merged into a vector representation, the so called document vector. For the implementation of Q-TF-IDF the raw term frequency is used since in Jiao et al. [18] there is no indication that any kind of term frequency normalization is used. Q-TF-IDF is only used to compare the TF-IDF related improvements of this work to the variant used by Jiao et al.. The acronym "Q-TF-IDF" refers to TF-IDF weights for cosine similarity calculation where the IDF part is based on the entire Question set. The modification presented by this work calculates the IDF part based on the User document vector collection and is therefore called "U-TF-IDF". Additionally, term frequencies of questions are normalized before they are added to the user document vector.

The user profile generation is not parallelized, however, as it turned out it would be relatively easy to establish. Each thread just needs to create separate user profiles. After all questions are processed the results of the threads need to be merged simply by adding the term counts of equal terms of the different user profiles from one specific user. However, since the current implementation is far less time consuming compared with the ranking process it is sufficient without multi-threading.

Term frequency normalization

There are three variants of normalization considered by this work:

1. Norm vector normalization [88]: The term frequencies of all distinct terms from the question document vector are divided by the norm vector of the question which leads to a unit vector. Therefore, all questions are treated to be equal. One could argue that longer questions also contain more information. On the other hand, short answers to questions can reveal even greater knowledge than long ones which might be lacking of precision. Since

it performed slightly better than the max term count normalization it is used in the context of U-TF-IDF calculation.

2. Max term count normalization: Each term frequency is divided by the term count of the most frequent term of the question vector. In this case there is still a distinction between longer and shorter questions.

3. Absolute term count normalization: All term frequencies are divided by the sum of all tokens from the question.

For QLLM the raw term counts are used since normalization of the term frequencies leads to a significant loss of prediction accuracy. Further, the best answer count, the categorized best answer count, the last activity and the activity history are determined by the User Profile Generator during processing the questions from the training data set.

5.2.4 Ranking Engine

The Ranking Engine is located in the sub package "core" of the package "prediction". It takes the user profiles, generated via the User Profile Generator component and a new question and calculates the ranking of users related to their likelihood of being suitable answerers for a given question. To perform this calculation several combinations of the proposed components from chapter 4 are supported:

- Authority only, where user authority alone is used to determine the ranking of the users. It can be chosen between InDegree, PageRank, ZScore, BAC and CBAC. The last one is only supported as baseline to be compared to the results of the more sophisticated methods.

- Expertise Estimation only, in which case one of the two components, the Vector Space Model (VSM) and the Query Likelihood Model (QLLM), can be chosen. For QLLM, it can be further selected, which smoothing method to apply. For VSM, there are two implementations, namely Q-TF-IDF and U-TF-IDF as mentioned in section 5.2.3.

- Combinations of realizations of Authority with Expertise Estimation are also supported. It is not supported to combine two or more Authority estimation methods, or two Expertise Estimation methods. For QLLM only a combination with InDegree is supported by the current implementation where InDegree is applied via a cascading approach. Since the QLLM scores do not appear to be compatible with the scores of InDegree via normal linear combination, an authority level is applied before QLLM is calculated. This means that all users who have less than a specific amount of in-links, which is specified via the mentioned authority level, are removed from the list of potential answerers. For combinations of VSM with an implementation of the Authority interface, linear combination is applied.

- The Activity Filter can be combined with Authority or Expertise Estimation, or with a combination of both of them.

- The Question Value can be used in combination with all previously mentioned variants.

Multi-Threading and the Abstract Factory Pattern

Like the data cleansing (DC) implementation the Ranking Engine (RE) uses multi-threading. Via application of the Abstract Factory Pattern [79], as shown in figure 5.3, the basic implementation of the `TheadPool` which manages threads, the so called `WorkerThreads`, can be used for several realizations of the `ThreadPool`. The implementations of `REThreadPool` and `DCThreadPool` only differ in the realization of the inherited method named termination-logic().

Via the realizations of the interface `IWorkerFactory`, which are `REFactory` and `DCFactory`, instances of `WorkerThreads` are created. These instances are called `RankingEngine` and `DataCleansing`. Both have access to the `ThreadPool` instance which is in the case of the `RankingEngine` threads called `REThreadPool`.

The implementation of the `ThreadPool` contains a question queue. A question parser reads question records from a data set and puts them in that queue. The `WorkerThreads` dequeue one question at a time and process the particular question. The `RankingEngine` threads calculate the rankings, whereas the `DataCleansing` threads cleanse questions, one at a time and store them within a new data file which is the foundation for the data selection as described in section 5.2.2. The mentioned data set is the raw data file in the context of `DataCleansing` and the test data set in the context of the `RankingEngine`.

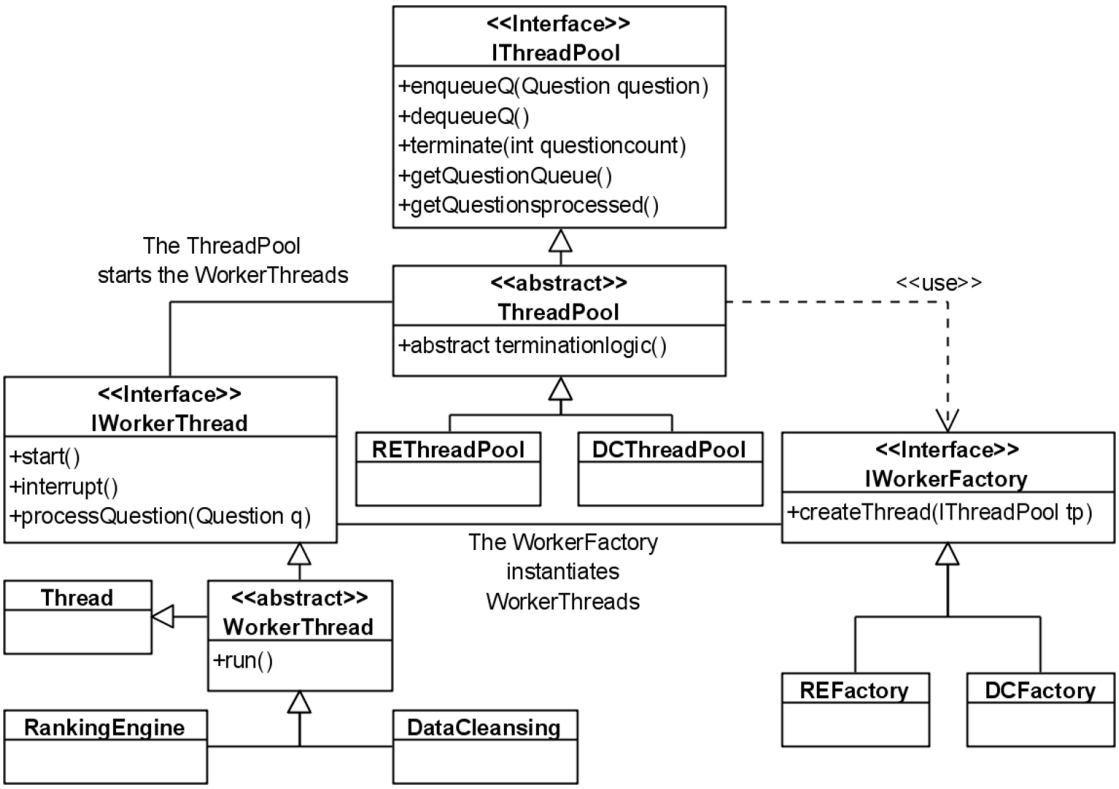

Figure 5.3: Factory pattern applied to multi-threading

Simplified Prediction Process with Multi threading

The prediction process as shown in figure 5.4 is based on a simplified evaluation test scenario, because the complete process would not fit into one diagram. The actual calculation of the ranked list of potential answerers is not part of the diagram. However, the whole process is successfully implemented and tested.

The Evaluation `TestCase` starts the `TestExecutor` and tells it via properties and a test case name with which settings the calculations are to be performed. First, the `TestExecutor` initiates the user profile generation via calling the related method of the `UserProfileGenerator`. The actual prediction process is launched via calling the static prediction method of the `RankingEngine` with the user profiles, the properties and the test data location as parameters. From that static method the `ThreadPool`, which is in this case an instance of `REThreadPool`, is created. Further, the `ThreadPool` instantiates the `WorkerThreads` which are in this case instances of the `RankingEngine`.

Then, the `Question Parser` is started which starts filling the question queue of the `ThreadPool`. The `WorkerThreads` are already waiting for questions to arrive within that queue. As soon as questions are available the `WorkerThreads` start dequeuing questions with which the ranking is calculated within the process(question) method of the `RankingEngine` instance. After all questions are processed the `ThreadPool` knows about the overall amount of questions from the `Question Parser` which tells the `ThreadPool` to terminate with the parameter "questioncount", which refers to the overall amount of questions that have been enqueued for processing, the `ThreadPool` interrupts all waiting `WorkerThreads`. As soon as they are interrupted they set their activity status to be inactive and therefore the loop in which they had been waiting which checks this flag, ends and the threads terminate.

Finally, the `ThreadPool` calls the termination logic method which is indicated via the `finished()` method of the figure. The results of the `RankingEngine` are analysed by the `TestExecutor` and via the logging system persisted.

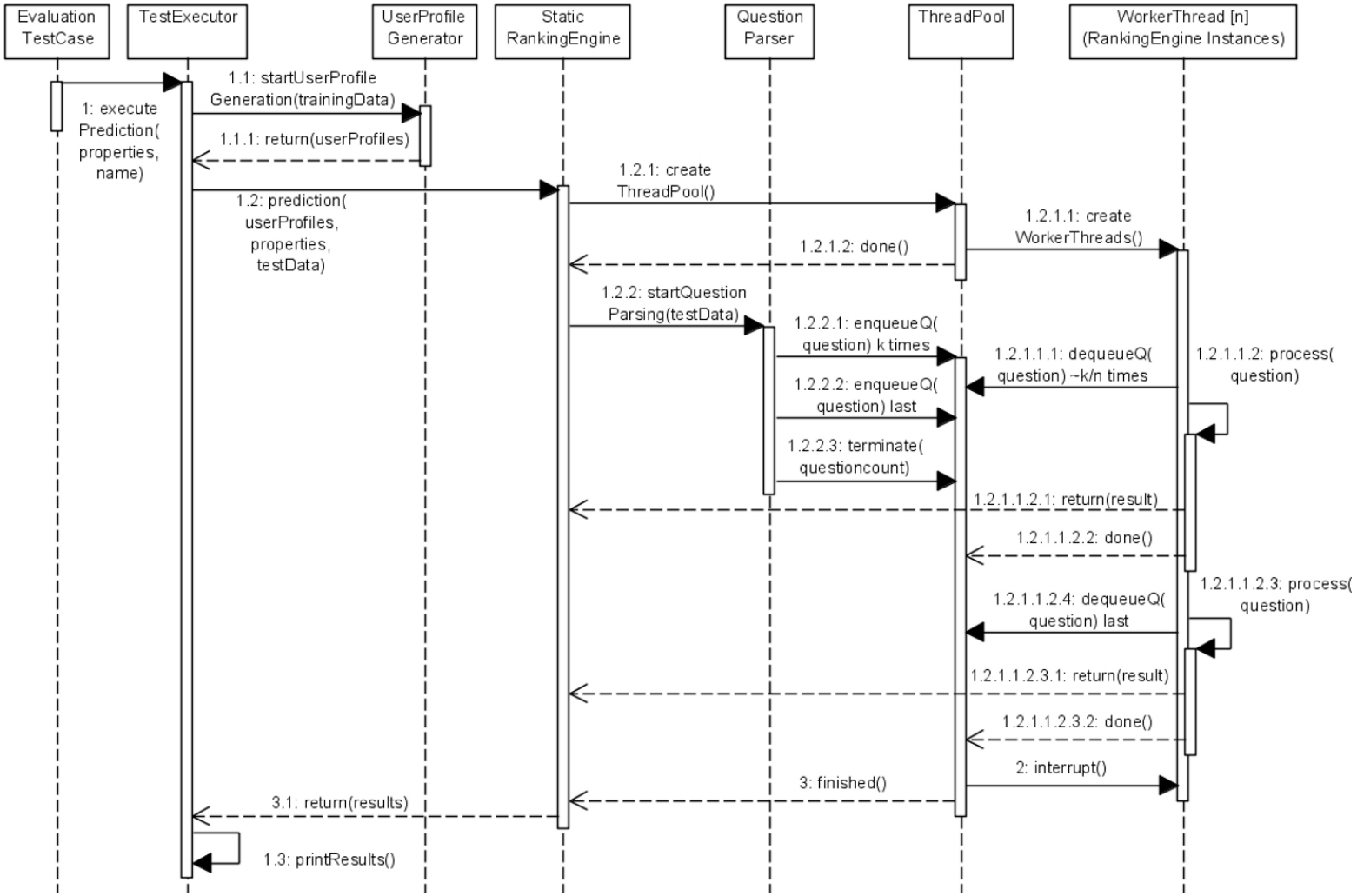

Figure 5.4: Simplified Prediction Process with Multi-Threading

CHAPTER 6

Evaluation

In this chapter, the ability to rank answerers of the different algorithms and combinations of them, in the context of the used test data provided by Yahoo! Answers, is examined via the usage of the presented framework. The main difference to comparable works in the field of ranking potential answerers in the context of Q&A communities is that this work uses a data set containing questions from a broad variety of categories. Basically, it contains all questions of all categories of Yahoo! Answers which are provided by their data set. All other related works consider only specific categories, or they focus on communities like, for example, the Java Forum which are dedicated to a specific knowledge domain. Further, none of the other works apply user activity via prior filtering which, as shown in this chapter, is quite significant in the process of ranking potential answerers.

The evaluation starts with a short description of the test environment. Next, the used data set which is provided by Yahoo! Answers is described. It is followed by a description of the used metrics. Then, the impact of the Activity Filter is investigated. Next, the different Authority estimation methods are analysed. After that the different QLLM smoothing methods are compared. Then, the application of the Vector Space Model and its improvements are discussed. Finally, the different combinations of the components of the framework are compared and a best setting is determined.

6.1 Test Environment

The evaluation is conducted via a remote desktop connection to a university computer with the following specifications:

- Operating system: Windows Server ®Enterprise without Hyper-V, Service Pack 2
- Processor: Intel(R) Xeon(R) CPU X5450 @ 3.00GHz 2.99GHz (2 processors) (overall 8 cores)
- Memory (RAM): 32.0 GB

- System type: 64-bit Operating System

6.2 Data Set Description

The data set is from the Yahoo! Research Alliance Webscope program [110]. The data, which consists of question records, are stored in an XML format. The statistics are shown in table 6.2. The used structure of the question records is show in figure 6.1. A description of the general data model is given in section 4.1.

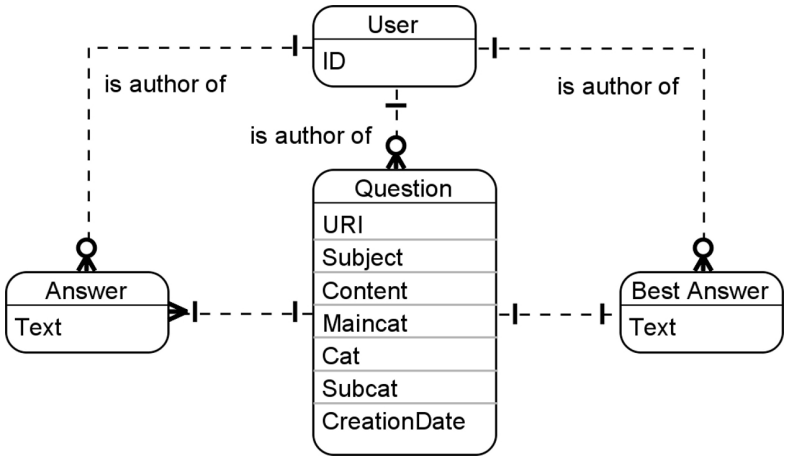

Figure 6.1: Data Model

The following meta data about the questions are available:

- Each question is identified via a uniform resource identifier (URI)
- A subject element, which specifies the query, which is referred to during the evaluation with "`S`".
- A query can be extended via the optional content field, indicated by "`C`".
- The best answer is stored within a specific element; it is referred to as "`BA`".
- All answers, a specific question received, are available too. However, just the number of answers is used in the context of the Question Value.
- The fields "maincat", "cat", and "subcat" together referred to via "`MCS`", represent the categorization of the question which are optional. Even if this work aims at overcoming the necessity of question categorization it is assumed that even if categories would not be required users still have to specify the question and their context somehow. Therefore, these three elements are treated as part of the query text.
- The author of the question is specified via the "id" element.

- The best answer is related to its author via the "best_id" field.

- The elements "qlang" and "language" specify the language of the question where "qlang" is used in the context of data cleansing in order to eliminate all questions with a different language than English. The field "qintl" specifies the location where the question was created. It is not used by this work.

- The element "date" specifies the date when the question was posted. It is used to separate training data from test data. Further, it is applied as estimation for the best answer creation data on which basis the user activity is calculated, because there is no exact information according to the creation data of the best answer available. The fields "lastanswerts" which indicates the data when the last answer of the question was posted, the optional field "res_date" which refers to the date when the question was resolved and the "vote_date" element which refers to the date when the best answer was determined via voting are not used by this work.

Total questions	4,483,032
Total questions after data cleansing	3,418,691
Distinct askers	1,363,459
Max amount of asked questions of one user	1,304
Average amount of asked questions	2.5
Distinct best answerers	427,765
Max amount of answered questions of one user	3,763
Average answer count	7.99
Answerers who also ask questions	274,977
Maximum query length	316
Average query length	10.8
Minimum query length	2
Questions within the training data set (BAC* per user >= 2)	3,195,096
Date of the first question of the training data set	Tue Jul 05 09:00:00 CEST 2005
Date of the last question of the training data set	Thu Nov 30 08:59:51 CET 2006
Distinct askers	1,300,066
Distinct best answerers	234,315
Answerers who also ask questions	166,571
Total amount of questions asked by answerers	1,301,517
Questions within the test data set (BAC* per user >= 2)	27,073
Date of the first question of the test data set	Thu Nov 30 09:00:07 CET 2006
Date of the last question of the test data set	Fri Dec 01 08:59:58 CET 2006

Table 6.1: Statistics about the used data; *BAC = best answer count

6.3 Evaluation Metrics

The following metrics are used throughout the evaluation of the framework.

6.3.1 Precision@N

Precision@N [97] (P@N), is adapted by this work. The P@N of a specific test case, states in how many rankings (a ranking per test question, and many test questions per test case are used) the actual best answerer is ranked, via the prediction process, within the top N. For example, there are 110 test questions, from which 100 are processed. The other ten cases are not considered because there is no information about the actual best answerer in the training data set, therefore no profile is built and hence it is impossible to verify the ranking result, because the actual best answerer would not be in the ranked list. If P@10 is used and the actual best answerer is 60 times within the top ten of the produced 100 rankings (as above, a ranking per test question), then P@10 would be 60%. Additionally, P@100 is used, which works exactly the same way, except that the top 100 are considered.

6.3.2 Recall

In this work, Recall [97] is interpreted differently compared to the standard definition, where Recall is referred to as the fraction of relevant items which are retrieved. We define it as the fraction of questions from the test data set, which are processed. A question can only be processed, if the actual best answerer of that question has a profile, based on the training data set, and is therefore part of the ranking.

6.3.3 Mean Reciprocal Rank

Mean Reciprocal Rank (MRR) [94] is quite frequently applied in the context of ranking potential answerers. It is calculated as follows [94]:

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{rank_i}$$

The amount of questions processed in a specific test scenario is denoted as $|Q|$. Therefore MRR represents the average reciprocal rank of the actual best answerer within the calculated ranking.

6.3.4 Mean Absolute Error

The Mean Absolute Error (MAE) [93] states the average difference of the actual best answerer within the calculated ranking to the first place. It is calculated as follows:

$$MAE = \frac{1}{|Q|} \sum_{i=1}^{|Q|} (rank_i - 1)$$

6.3.5 Mean Squared Error

The Mean Squared Error (MSE) [95] is the average squared absolute error. It is used to analyse the variance of the calculated rankings in relation to the position of the actual best answerer. Bad predictions affect this metric score heavily, whereas for MRR, a few bad rankings of the actual best answerer are almost meaningless, as long as there are many good rankings related to the position of the best answerer. MSE is calculated as follows:

$$MSE = \frac{1}{|Q|} \sum_{i=1}^{|Q|} (rank_i - 1)^2$$

6.3.6 Normalized Error

The Normalized Error (NERROR) appears to be a new measure. It is the normalization of the mean absolute error by the length of the ranking of potential answerers. It helps comparing ranking results of rankings with different amount of users. It is calculated as follows:

$$NERROR = \frac{MAE}{length(R)}$$

The amount of users, which are ranked, is denoted by $length(R)$. The lower the value, which is always between 0 and 1, the better the method under consideration.

6.4 Activity Filter

The Activity Filter, as described in section 4.3 improves the accuracy of the best answerer prediction via removing all inactive users prior to the actual calculation, and reduces the processing time significantly. The prediction calculation for the following figures is based on QLLM with Dirichlet smoothing (QLLM-D). The following diagrams show the effect of the activity filter via various metrics.

Figure 6.2 shows the prediction results, where via the Activity Filter (AF), 1 to 20 days are considered for users to be relevant for the ranking. The test case "all" refers to the case, where the AF is turned off. If all users are considered, the Recall is 100%, since no users are excluded from the ranking calculation, therefore all actual best answerers have user profiles, since in the test data set, there are only questions from users which already gave best answers within the training data set. If AF is set to 1, which means, that only users are considered for the ranking, which have been active via giving best answers within the last 24 hours, the P@10 and P@100 values improve significantly. However, the Recall drops to 60%. It can be concluded that the majority of questions from the test data set receives best answers from users who have been online within the last 24 hours. Since the aim of this work is to decrease response time, the Activity Filter is set to consider only the last 24 hours for all further test cases.

As shown by figure 6.3, the very positive side effect of the Activity Filter is its enormous reduction of the processing time per question. In this case, where all users are considered who gave at least one best answer within the training data set, the processing per question to produce

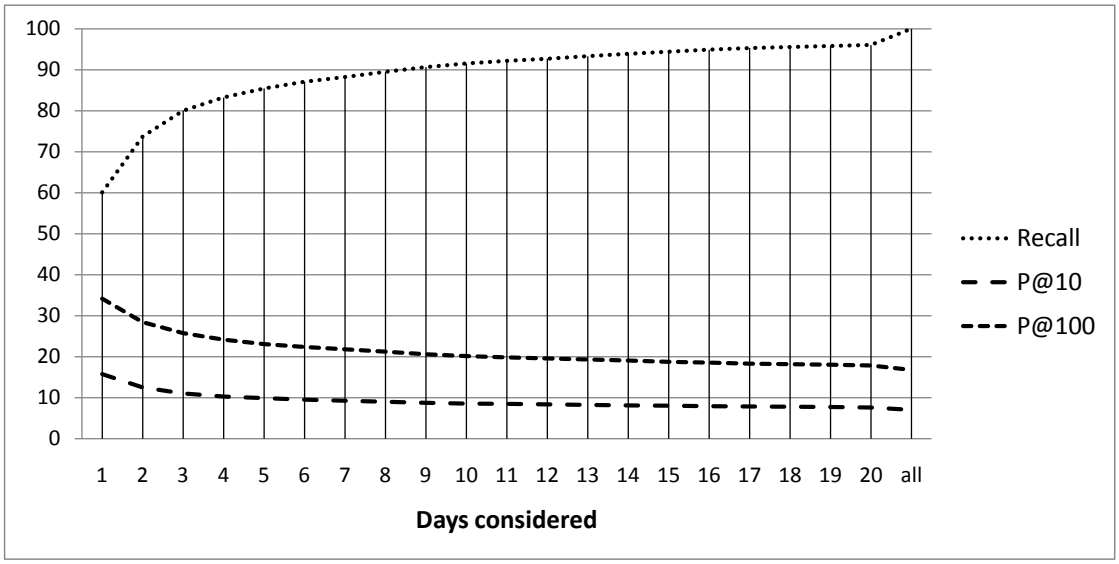

Figure 6.2: Effect of Activity Filter related to Recall and Precision; user considered who gave at least one best answer

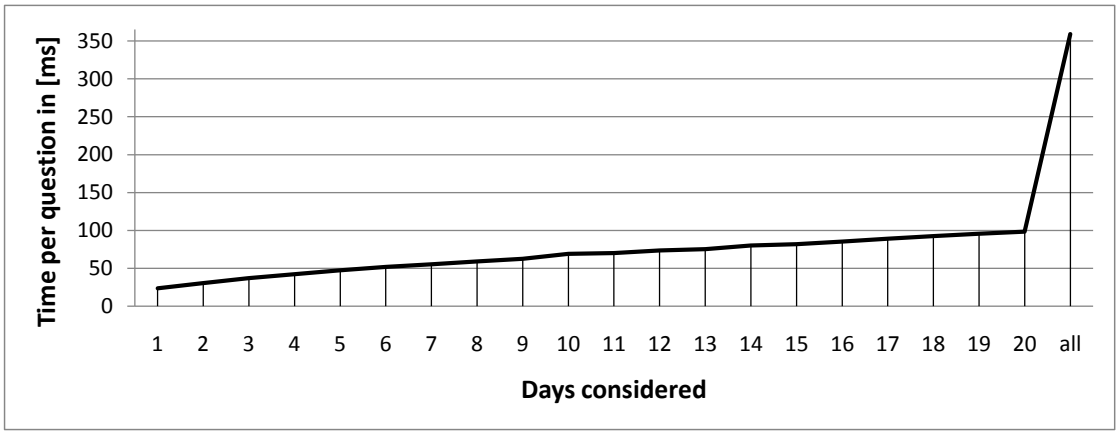

Figure 6.3: Effect of Activity Filter related to calculation time per question; users considered who gave at least one best answer

the ranking, if the AF is turned off, takes more than 350 ms. If the AF is set to 1, it is less than 25 ms which is a reduction of more than 90%.

The decision to select all users who gave at least one best answer within the training data set leads to a significant problem. As figure 6.4 shows not only the MRR value rises up as the days which are considered via the AF decrease, but also the NERROR increases after the sharp decline between all and 20. The assumption is that the less days that are considered, the more the effect of new users, which join the community, affects the NERROR value. Especially if only users which have been active within the last 24 hours which have at least one best answer

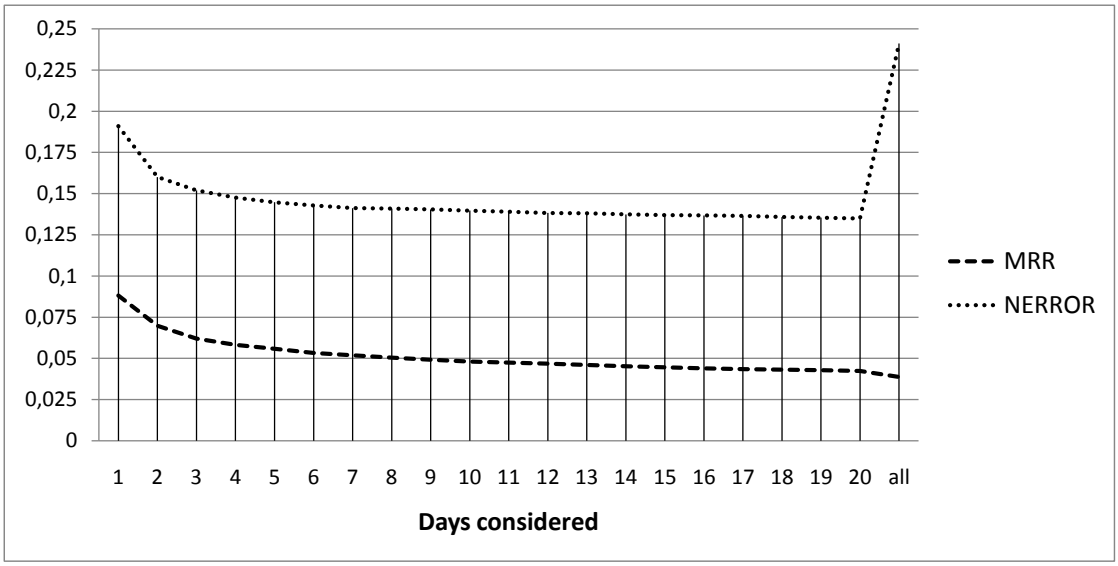

Figure 6.4: Effect of Activity Filter to MRR and Normalized Error; user considered who gave at least one best answer

are considered it is likely that there are many new users who just gave their first best answer. They still might experiment with different areas to which they want to give answers. Therefore, it can be concluded that the requirement that users have to provide one best answer in order to be considered for the prediction process is not restrictive enough. Therefore, user profiles need to be based on more than one best answer. Overall, the requirement is increased to at least 2 best answers per user. The effect of only one best answer can also be seen in figure 6.6 where only active users from the last day are considered, and in figure A.1 of the appendix, where all users are considered who gave at least one best answer. Most actual best answerers are predicted on the left side of the prediction which is good, however, a significant portion of results is located on the right end of the distribution, as shown by the bins 10,000 to 16,500 of the histogram from figure 6.6 and the approximate bins 300,000 to 450,000 of histogram A.1 respectively. It is assumed that those results are based on users who gave one best answer in one domain on which their user profile is built and in the future, on the test day, they answer a question from a completely different domain. This behavior seems to be likely for new users who are discovering the Q&A platform.

In figure 6.5 only users with at least two best answers within the training data set are considered for the ranking calculation. In this case the MRR value still acts as assumed and the NERROR now also performs as expected which means that it should decrease since only users are considered who are active and who are assumed to answer questions from the test day. The more inactive users are removed the smaller the NERROR value should become since the inactive users could receive a high rank, but are not likely to answer questions. Further, this seems to support the assumption made above about new users. Figure 6.7 also supports the theory since the sorting effect does not occur as seen in figure 6.6, if users need to provide at least two best

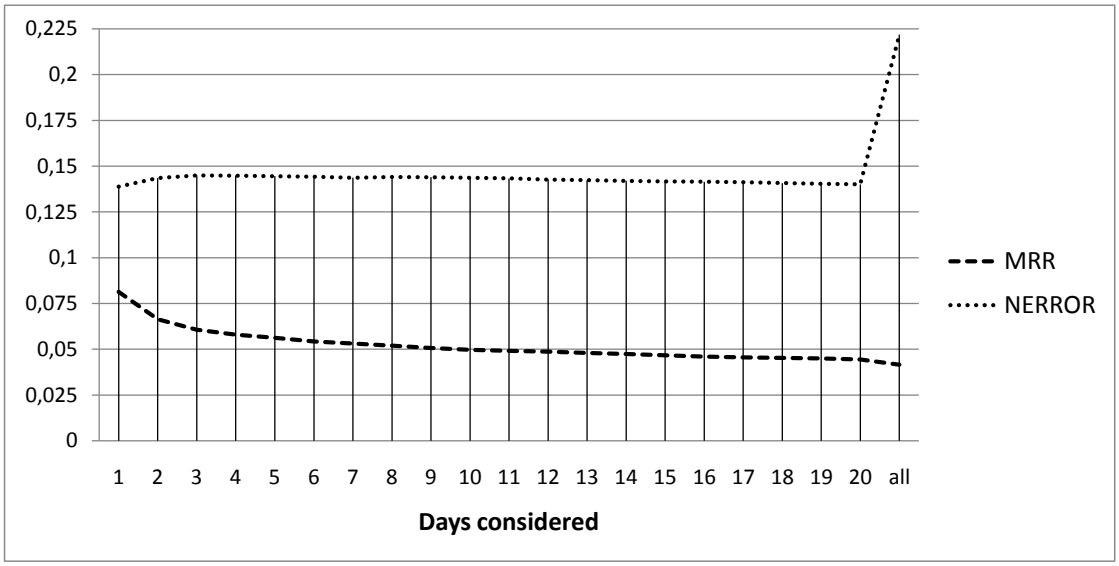

Figure 6.5: Effect of Activity Filter related to MRR and Normalized Error; Users considered who gave at least two best answers

answers within the test data set. The scores for the bins for high ranks (10,000 to 16,000) are very low, as desired.

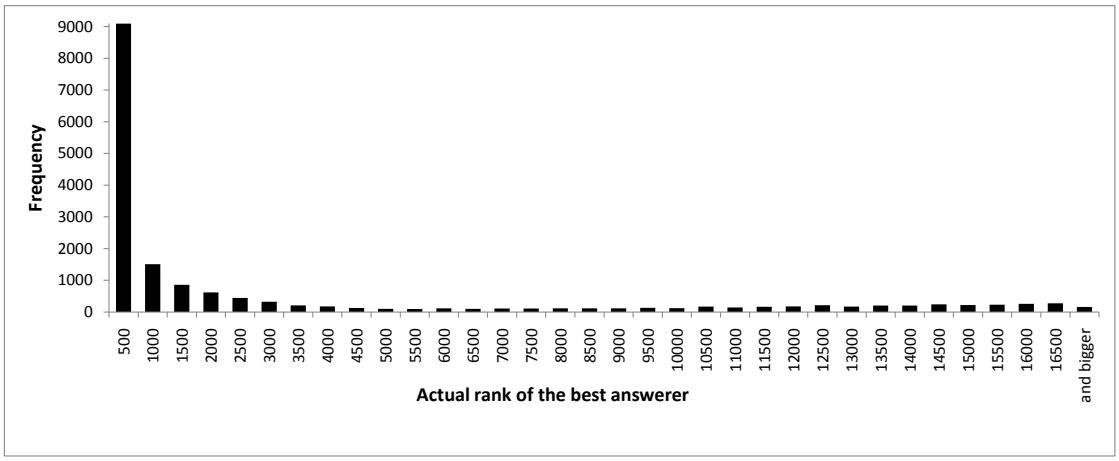

Figure 6.6: Histogram of QLLM-Dirichlet based prediction with users who have been active approximately within the last 24 hours and who gave at least 1 best answer

In conclusion, the time period is set to 24 hours in order to receive fast answers. This means that only users who have been active within the last 24 hours are considered for the best answerer prediction process. This reduces the amount of test questions, because only test questions are considered of which the actual best answerer is known. However, the aim is to increase the

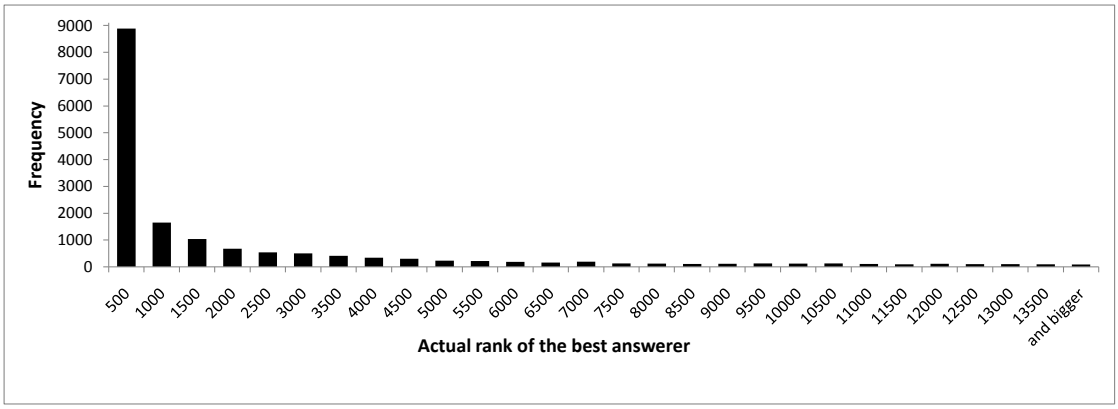

Figure 6.7: Histogram of QLLM-Dirichlet based prediction with users who have been active approximately within the last 24 hours and who gave at least 2 best answers

askers' satisfaction. This can be improved via answers which are not only satisfying because of the content that they produce, but also because of being provided within a short period of time. The assumption is that users who have been online recently will provide quicker responses than users who have not been active for a few days which seems to be reasonable. Since the MRR value increases the shorter the time period for users to be considered becomes it indicates that recently active users are more likely to provide best answers than users who have not been online for a while. This does not mean that those users have less knowledge, but that they just do not participate as much as the more active ones.

6.5 Authority

The proposed framework offers several concrete implementations of the Authority interface as described in section 4.5. The results of their performance according to several metrics can be found in figure 6.2.

The "Random" example is only used to demonstrate the difference of the actual Authority techniques to the case where the ranking is an unordered list of potential answerers from which one user is randomly selected to be the assumed best answerer. The NERROR value of 0.5 proves that the users are selected randomly, because on average the assumed best answerer is always in the middle of the list. The actual distribution can be found in the appendix A.2.

The global best answer count "BAC" 4.5.3 sorts the users based on the amount of best answers they have given. The distribution of the actual best answerer produced via this technique is shown in the appendix A.6. The MRR value compared to the random example significantly improves and the NERROR value decreases which also indicates the effectiveness of this method.

The categorized best answer count, as described in 4.5.4, is only used as baseline for the best setting of the framework, presented in 6.9. Since this work aims to avoid the necessity of question categorization in order to increase user convenience it is not valid to use CBAC. However, since this method performs quite well, there could be an improvement of the framework,

where topics are extracted from the question in order to apply the CBAC score. It is not part of the current version of the framework.

ZScore, as presented by Zhang et al. [51] and additionally described in section 4.5.5 is another method of authority estimation. Since Bouguessa et al. mentioned that it is not to a valid user authority estimation method, it is not used in the context of the proposed best setting, presented at the end of this chapter, however, its MRR score compared to BAC and InDegree is surprisingly good. The overall sorting ability of ZScore, indicated by NERROR, is worse than the ones of BAC and InDegree. The distribution of ZScore can be found in the appendix A.5.

PageRank, as described in section 2.1.5 and 4.5.2 has the worst performance of the MRR value which means that the rankings produced by PageRank contain less top ranks of the actual best answerers compared to all the other methods. Its NERROR value is comparable to the one of ZScore. The distribution of PageRank can be found in appendix A.4. The calculation of PageRank is based on 5 iterations of the algorithm.

InDegree, as mentioned by Bouguessa et al. to be best suitable in the context of Q&A communities like Yahoo! Answers, is described in section 4.5.1. Its MRR value is almost equal to the ones of BAC and ZScore, whereas its NERROR value together with BAC is the best one. InDegree is further used in combination with the Expertise Estimation methods in order to find a best setting.

Input	Technique	Random	BAC	CBAC	ZScore	PageRank	InDegree
Id, Best_id, Maincat, Cat, Subcat	Total users profiles (TUP)	234315	234315	234315	234315	234315	234315
	Recently active users (RAU)	13969	13969	13969	13969	13969	13969
	Total test questions (TTQ)	27073	27073	27073	27073	27073	27073
	Analysed test questions (ATQ)	16887	16887	16887	16887	16887	16887
	Recall	62,38%	62,38%	62,38%	62,38%	62,38%	62,38%
	Precision@10 (P@10)	0,09%	1,26%	16,00%	1,80%	0,59%	1,36%
	Precision@100 (P@100)	0,65%	6,76%	36,76%	6,85%	5,00%	6,67%
	Mean Absolute Error (MAE)	7,042E3	3,832E3	2,532E3	4,080E3	4,077E3	3,847E3
	Mean Square Error (MSE)	6,579E7	2,810E7	2,162E7	3,245E7	3,029E7	2,826E7
	Mean reciprocal rank (MRR)	0,0008	0,0058	0,0856	0,0066	0,0040	0,0060
	Normalized Error (NERROR)	0,5041	0,2744	0,1812	0,2921	0,2919	0,2754

Table 6.2: Evaluation of the user authority estimation methods; Users considered who gave at least two best answers and have been active within the last 24 hours; BAC = best answer count, CBAC = categorized best answer count

6.5.1 Cascading InDegree (CID)

The calculated values of InDegree appear to be incompatible with the extremely small values produced by QLLM. Therefore, a linear combination leads to no improvement of the overall ranking, hence a cascading approach is used. This means that users who are considered as potential answerers for new questions not only have to fulfill the requirements of the Activity Filter, but they must also provide a certain amount of in-links in order to be considered as authoritative users. Figure 6.8 shows the effect of the required in-link count on Recall and Precision. As the in-link requirement rises P@10 and P@100 improve. On the other hand, the Recall value drops since less questions can be processed, because not all questions are answered by the selected, presumably more authoritative users.

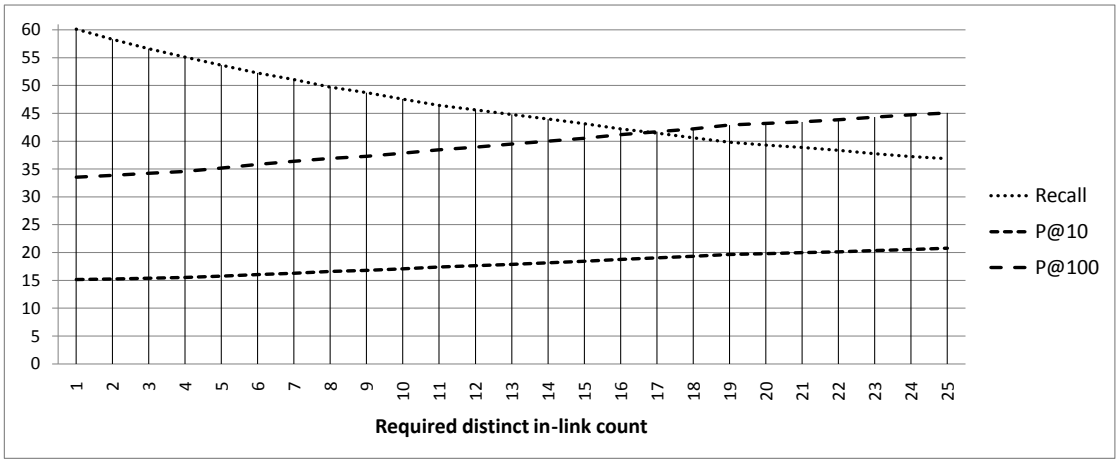

Figure 6.8: Cascading InDegree in combination with QLLM-D, Recall and Precision

Figure 6.9 shows that the NERROR drops the higher the in-link requirement is set. The MRR value is also better the higher the in-link count is set. One interpretation for the rising MRR value is that the higher the in-link requirement is set, in general, the more best answers are provided by each user. Therefore, the user profiles are based on lots of questions. Hence, QLLM can match new queries more accurately to the user profiles. Another point is that the higher the in-link count the less users are considered for the ranking. Therefore, the value has to rise. However, since the list of potential answerers at a required in-link count of, for example, 10 is still huge it does not appear to be the main reason for the improvement of the MRR value. Another explanation is that there could be users who's profiles match quite well with the query under consideration, but who just answer questions very seldom indicated by their low in-link score. These users would be ranked quite hight, but the cases in which they might really be the best answerer of a test questions are rare. These users would definitely have a negative impact on the MRR value.

To improve InDegree the in-links could be weighted. In this work the link weights were meant to be based on profile similarity between asker and answerer in order to solve the problem indicated by Bouguessa et al. related to PageRank where user might have different knowledge

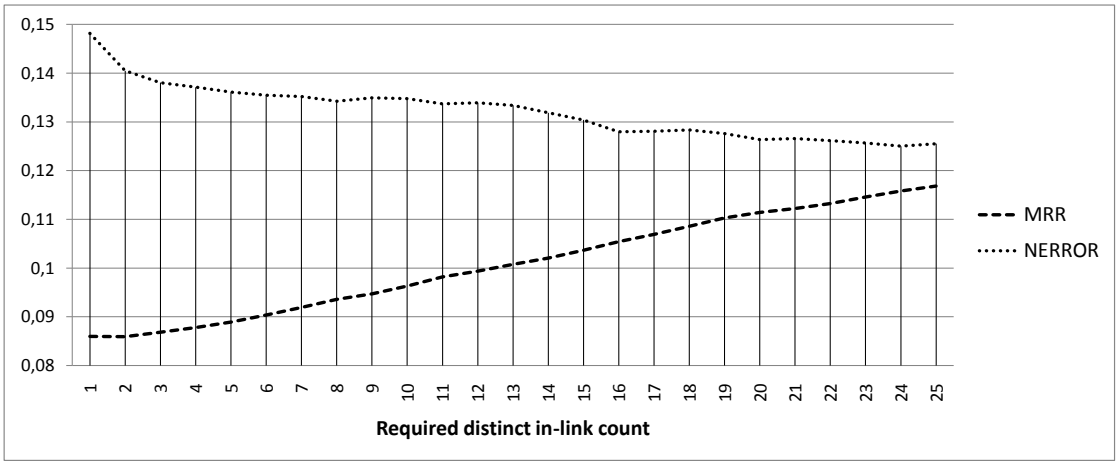

Figure 6.9: Cascading InDegree in combination with QLLM-D, MRR and NERROR

and therefore it can not be concluded that the answerer has in general more knowledge/authority than the asker. However, since the majority of users only asks questions, but never answers any of them there is just nothing that could be compared therefore, weighting of in-links is not applied. The experiments with in-link weights have not improve the ranking therefore, weighting of in-links is not used.

6.6 QLLM smoothing methods compared

Before the actual comparison of the different smoothing methods the potential improvement of Jelinek Mercer smoothing via determining the parameter λ based on the query length is investigated.

6.6.1 Query Length impact

The query length impact, as described in section 4.4.1, is assumed to improve Jelinek Mercer smoothing, if for short queries only a little smoothing is applied whereas for long queries more smoothing is assumed to yield better results. As shown in figure 6.11, the value of λ decreases asymptotically via the proposed calculation method of section 4.4.1.

The statistics about query length, presented in table 6.2 and figure 6.10, are based on a question collection with 3,418,691 elements from a dataset provided by Yahoo! Answers. The frequency distribution of the query length shows that the vast majority of questions is between a length of 2 and 20. Since the average query length is quite low and the variance appears to be quite small the effect of query length consideration is limited. Via the calculated λ, shown in figure 6.11, it is obvious that the majority is treated as short queries with low smoothing.

The results of the comparison of the different smoothing methods, as described in 2.1.4 and 4.4.1, are shown in table 6.6.1. It is quite obvious that QLLM with Dirichlet smoothing, which

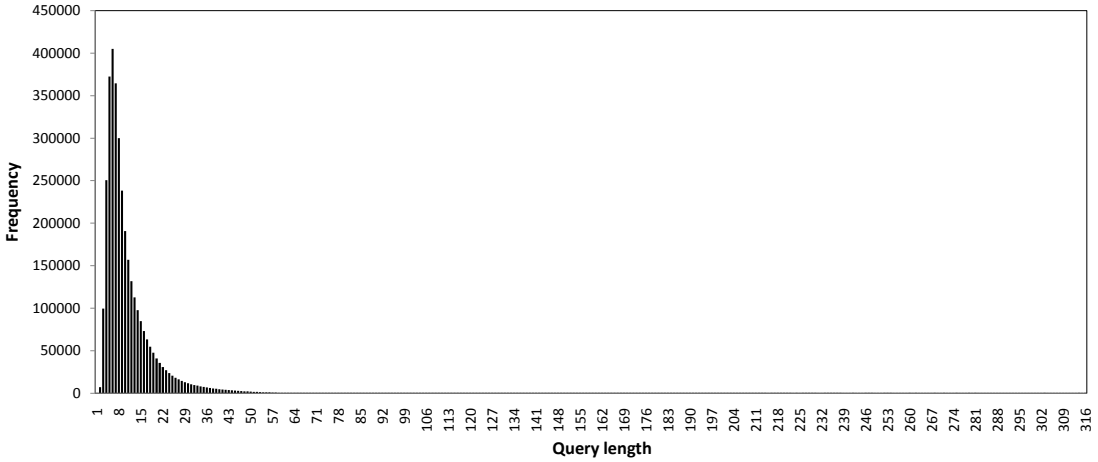

Figure 6.10: Distribution of the query length

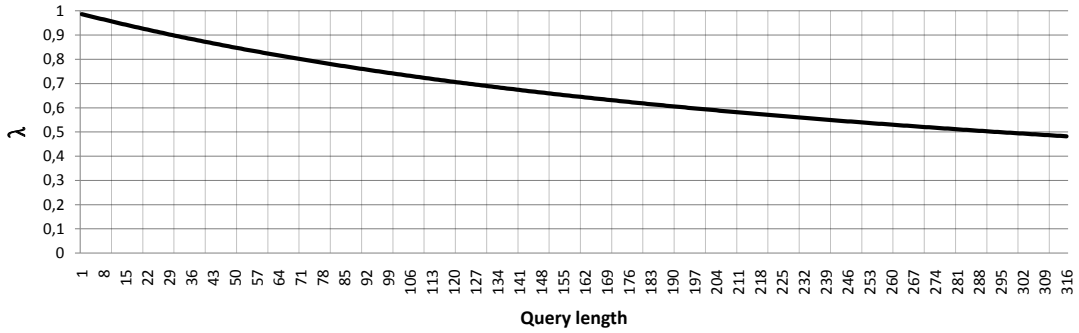

Figure 6.11: λ depending on the query length L_q

will henceforth be referred to as QLLM-D, performs best. It has the highest P@10 value, the by far best MRR value, only the NERROR is almost equal to Jelinek Mercer smoothing, with and without Query Length consideration. WittenBell performs worst for all considered metrics. It is obviously not suitable in the context of this work. As conclusion, QLLM-D is used for the final determination of the best setting.

Input	Technique	Dirichlet	JM $\lambda = 0.9$	JM QL-λ	WittenBell
Subject, Content, Best Answer, Maincat, Cat, Subcat	TUP	234315	234315	234315	234315
	RAU	13969	13969	13969	13969
	TTQ	27073	27073	27073	27073
	ATQ	16887	16887	16887	16887
	Recall	62,38%	62,38%	62,38%	62,38%
	P@10	14,93%	13,10%	13,09%	11,20%
	P@100	33,28%	32,39%	31,91%	28,23%
	MAE	1,940E3	1,827E3	1,852E3	2,307E3
	MSE	1,363E7	1,223E7	1,235E7	1,655E7
	MRR	0,0844	0,0665	0,0659	0,0572
	NERROR	0,1389	0,1308	0,1326	0,1652

Table 6.3: Evaluation of different smoothing methods for QLLM, JM = Jelinek Mercer, QL = Query Length; Users considered who gave at least two best answers and have been active within the last 24 hours; Metric acronyms as in table 6.2 and section 6.3

6.7 Vector Space Model Variants

The Vector Space Model, described in section 2.1.2 and 4.4.2, is one of the realizations of the Expertise Estimation interface of the proposed framework from chapter 4. All variants are based on the cosine similarity between document vectors of the user profiles and the query document vector. The difference, as already mentioned, lies in the interpretation of the term weights. For Q-TF-IDF, TF-IDF weights are based on the entire question collection from which the user profiles are built. If only users are considered who gave at least two best answers then only those questions that are answered by these users and which are part of the training data set are within the entire question collection. Q-TF-IDF, which is used by Jiao et al. [18], is compared to the proposed variant of this work, namely U-TF-IDF. For U-TF-IDF, the TF-IDF weights are based on the user profile collection.

The results, presented in table 6.7, show that U-TF-IDF significantly outperforms Q-TF-IDF. The values of all six used metrics (P@10, P@100, MAE, MSE, MRR, and NERROR) are significantly better than Q-TF-IDF.

The other tests investigate the effect of the Question Value (QV), the Interest Tendency (IT) and a combination with QV and InDegree (ID). The QV, in combination as U-TF-IDF-QV slightly improves the NERROR metric, but it is not significant. The IT extension which indicated improvements with very little test data is also not able to produce significant improvements. Even if all metrics except the MRR value are slightly better than of plain U-TF-IDF the differences are quite small. The combination of U-TF-IDF-QV with the Authority instance InDegree significantly improves the P@10 and the MRR values. MAE, MSE, and NERROR on the other hand get worse. This means that this combination is able to improve top rankings, but

the overall ranking is not so good. Since only accurate top rankings matter, the new question should be forwarded to the best suitable users not to all of the ranking; The overall ranking ability of the used technique is not crucial. It is only important that approximately the top 100 are really capable of answering the query. For the final determination of the best setting U-TF-IDF itself and in combination with QV and ID is used.

Input	Technique	Q-TF-IDF	U-TF-IDF	U-TF-IDF-QV	U-TF-IDF-IT	U-TF-IDF-QV-ID
Subject, Content, Best Answer, Maincat, Cat, Subcat	TUP	234315	234315	234315	234315	234315
	RAU	13969	13969	13969	13969	13969
	TTQ	27073	27073	27073	27073	27073
	ATQ	16887	16887	16887	16887	16887
	Recall	62,38%	62,38%	62,38%	62,38%	62,38%
	P@10	12,53%	14,07%	14,07%	14,09%	15,61%
	P@100	32,72%	35,41%	35,41%	35,54%	33,36%
	MAE	1,825E3	1,767E3	1,763E3	1,757E3	2,207E3
	MSE	1,191E7	1,180E7	1,171E7	1,173E7	1,562E7
	MRR	0,0613	0,0702	0,0702	0,0702	0,0836
	NERROR	0,1306	0,1265	0,1262	0,1258	0,1580

Table 6.4: Evaluation of different realizations of the Vector Space Model (VSM); Users considered who gave at least two best answers and have been active within the last 24 hours; Metric acronyms as in table 6.2 and section 6.3

6.8 Vector Space Model compared to Query Likelihood Language Model

Via the final tests, where the two Expertise Estimation realizations in combination with InDegree as user Authority estimation are compared where further several input combinations related to the different elements of the question 6.2 are investigated, the best setting of the framework is determined. For the Vector Space Model U-TF-IDF and U-TF-IDF-QV-ID are used where the latter combines cosine similarity with user based TF-IDF weights, the Question Value and InDegree as user Authority measure. The Query Likelihood Language Model (QLLM) is used in combination with Dirichlet smoothing (QLLM-D). Further, QLLM-D is combined with the cascading realization of InDegree (CID) and also with both, QV and CID. CID is set to consider all users as authoritative who have at least 10 distinct in-links. Others are not considered as potential answerers in the test cases where CID is applied.

The results, presented in table 6.8, show that QLLM-D-CID performs best where subject, content, best answer, maincat, cat, and subcat of the question are used for building the user

Input	Technique	U-TF-IDF	U-TF-IDF-QV-ID	QLLM-D-CID-QV	QLLM-D-CID	QLLM-D
Subject, Best Answer	TUP	234315	234315	234315	234315	234315
	RAU	13969	13969	7554	7554	13969
	TTQ	27073	27073	27073	27073	27073
	ATQ	16887	16887	13768	13768	16887
	Recall	62,38%	62,38%	50,86%	50,86%	62,38%
	P@10	6,37%	8,40%	12,35%	12,36%	10,33%
	P@100	17,56%	18,91%	25,88%	25,97%	21,77%
	MAE	3,607E3	3,171E3	2,059E3	2,118E3	4,861E3
	MSE	2,969E7	2,332E7	9,839E6	1,034E7	4,925E7
	MRR	0,0337	0,0463	0,0721	0,0731	0,0610
	NERROR	0,2582	0,2270	0,2726	0,2803	0,3480
Subject, Content, Best Answer	P@10	7,70%	9,90%	13,33%	13,42%	11,30%
	P@100	21,20%	20,74%	28,07%	28,32%	23,95%
	MAE	3,014E3	3,040E3	1,682E3	1,686E3	3,335E3
	MSE	2,311E7	2,221E7	7,227E6	7,273E6	2,896E7
	MRR	0,0408	0,0532	0,0752	0,0766	0,0645
	NERROR	0,2157	0,2176	0,2227	0,2232	0,2387
Subject, Content, Best Answer, Maincat, Cat, Subcat	P@10	14,07%	15,61%	17,51%	17,60%	14,57%
	P@100	35,41%	33,36%	38,65%	38,89%	32,68%
	MAE	1,767E3	2,207E3	1,007E3	1,004E3	1,935E3
	MSE	1,180E7	1,562E7	3,521E6	3,519E6	1,331E7
	MRR	0,0702	0,0836	0,0984	0,0989	0,0824
	NERROR	0,1265	0,1580	0,1333	0,1329	0,1386
Maincat, Cat, Subcat	P@10	8,18%	13,71%	16,22%	16,22%	13,28%
	P@100	30,49%	33,87%	40,27%	40,27%	33,40%
	MAE	1,868E3	1,790E3	8,950E2	8,949E2	1,843E3
	MSE	1,273E7	1,224E7	3,145E6	3,145E6	1,377E7
	MRR	0,0398	0,0758	0,0868	0,0868	0,0711
	NERROR	0,1337	0,1281	0,1185	0,1185	0,1319

Table 6.5: Comparison of VSM (U-TF-IDF) with QLLM-D; Users considered who gave at least two best answers and have been active within the last 24 hours; Metric acronyms as in table 6.2 and section 6.3

profiles and where subject, content, maincat, cat, and subcat of the "new" questions from the test data set are used for the ranking calculation. As already mentioned, the categorization is treated to be part of the query despite the fact that by the proposed framework question categorization is not used users have to specify their queries properly. U-TF-IDF has the advantage that it is compatible with InDegree via linear combination. Therefore, the related Recall value does not

decrease, whereas the application of CID yields a decrease of the Recall value. QLLM-D-CID compared with CBAC from table 6.2 shows that QLLM-D-CID performs better since its MRR value is higher. On the downside the Recall value is lower because of the already mentioned reason.

The diagram 6.12 highlights the results of the different combinations of techniques for the ranking calculation via the MRR metric. It reveals that the Question Value does not improve the results of QLLM-D-CID. It also demonstrates the significant difference between plain U-TF-IDF and plain QLLM-D where the MRR value of QLLM-D is for each input type far better than the one of U-TF-IDF. The advantage of U-TF-IDF is its ability to be linearly combined with InDegree which leads to significant improvements over plain U-TF-IDF.

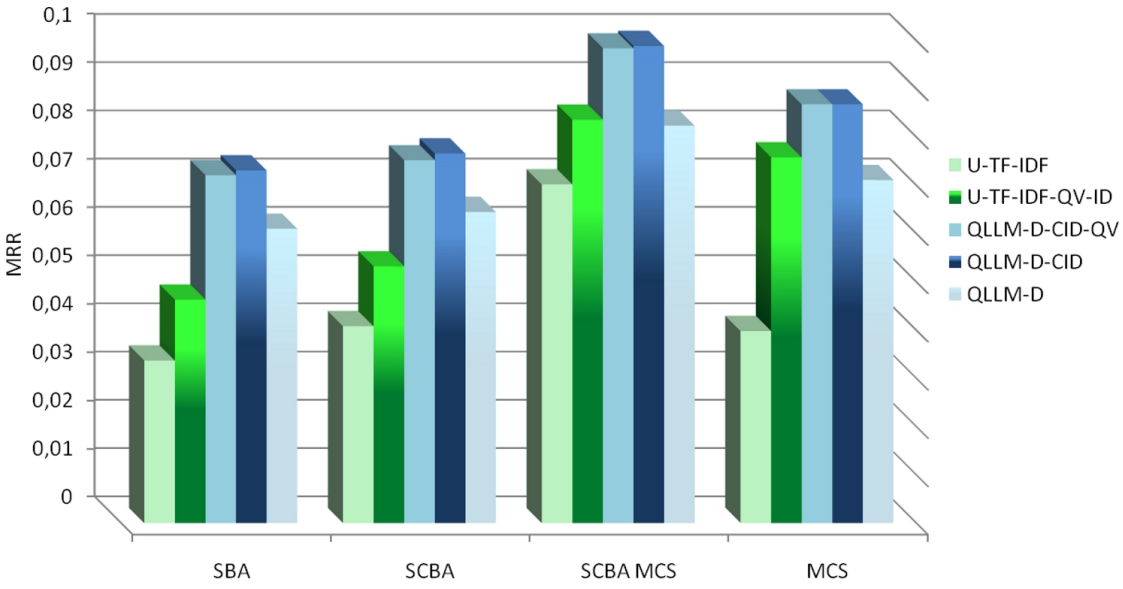

Figure 6.12: Comparison of VSM with QLLM via the MRR metric with users who have been active approximately within the last 24 hours (via Activity Filter) and who gave at least 2 best answers; The acronyms on the bottom refer to the kind of input data as described in section 6.2

The diagram with the NERROR metric can be found in the appendix A.7 The histograms of CBAC, U-TFIDF-QV-ID and QLLM-D-CID can be found in the appendix A.8, A.9 and A.10. The proposed best setting is described in the following section.

6.9 Best Setting Discussions

The best setting which has been determined via the various experiments presented above combines the three main components of the framework which are Knowledge/Expertise, Authority and Activity. As expertise estimation the Query Likelihood Language Model with Dirichlet Prior smoothing P_D is used. The user activity is applied via the Activity Filter which is set to consider users who have been online within the last 24 hours. This setting has to be adapted to

the context in which the framework is applied since it is important that enough users are considered as potential answerers in order to avoid jamming the top users with all new questions. The Activity Filter removes all inactive users prior to the actual calculation of the ranking.

As user authority measure $InDegree$ is used via a cascading approach similar to the user activity. From the list of potential answerers, after removing all inactive ones, all users who have less than a specific amount of in-links, for example 10, are removed. Furthermore, Interest Tendency is turned off since it does not seem to be compatible with QLLM. The Question Value which is intended to be used to decide the ranking of users with an equal score based on the previously mentioned measures is also turned off since it does not improve the ranking. The probability that question q can be answered satisfactorily by user u_i which is denoted as $P_{E_{AA}}$ (Expertise, Authority and Activity), is calculated as follows:

$$P_{E_{AA}}(q \mid u_i) \propto \prod_{t \in V_q} \left(\frac{tf_{t,d_{u_i}} + \gamma \hat{P}(t \mid M_c)}{L_{d_{u_i}} + \gamma} \right)^{tf_{t,q}}$$

The denotation E_{AA} indicates that authority and activity are considered prior to this calculation as described above.

CHAPTER 7

Conclusion

Question and Answer communities have become very popular. There are huge communities, like Yahoo! Answers where tens of thousands of questions are posted every single day. Users who are willing to answer new questions have to spend a significant amount of their time with searching for appropriate questions. This wastes time and is rather frustrating.

In this work we presented an answerer ranking framework in order to tackle the challenge mentioned above, caused by the enormous amount of new questions in such communities. It is aimed at increasing answerers' satisfaction via suggesting appropriate questions to them and further via an assumed improvement of the response time as a result of the automatic question suggestion system an increase of askers' satisfaction. The three core components of the framework are Expertise/Knowledge, Authority and Availability which are used to predict the accuracy, trustworthiness and response time of a user's potential answer regarding a particular new question.

As concrete realizations of the Expertise/Knowledge component an improved variant of the Vector Space Model (VSM) is used. In contrast to applications of VSM by other works where term frequency - inverse document frequency (TF-IDF) weights are applied based on the entire question collection from the community we apply user based TF-IDF weights. The second realization of Expertise/Knowledge uses the Query Likelihood Language Model (QLLM) with different smoothing techniques in order to overcome the data sparsity problem. Since user profiles are on average built from only a few best answered questions these profiles simply can not contain all possible terms of a language. If only one term of a specific query is missing in a specific user profile, the QLLM based probability for this user profile of being potentially related to the question turns to zero. Therefore, a background model is applied via smoothing in order to solve this problem through giving terms that do not occur in specific user profiles the average probability based on all user profiles.

The functionality of the Authority component can be realized based on different approaches. InDegree, a distinct in-link based measure, where the authority of a user is determined by the amount of different people he has helped. PageRank is also link based, but authority scores are distributed throughout this link structure. ZScore basically uses the difference between the

amount of answered and asked questions where user who ask lots of questions are assumed to be less authoritative, since asking is assumed to indicate lack of knowledge. The best answer count (BAC) simply counts the amount of best answers given by a specific user. The categorized best answer count (CBAC) is the same as BAC, but on category level. This approach is only used to compare the results of the other methods since categories are not assumed to be present in the context where the framework is applied. User Activity is considered via an Activity Filter with which all inactive users are removed from the list of potential answerers prior to the actual calculation of the ranking.

7.1 Results

The results of this work show that not only knowledge and authority of potential answerers are important, but especially in the context of huge communities with hundreds of thousands of potential answerers user activity plays a crucial role in the context of forwarding question to appropriate answerers. The Activity Filter presented in section 4.3 which, to the best of our knowledge has not been used by related work, ensures that only recently active users are considered for the actual calculation of the ranking of potential answerers. This has several advantages. Users who are almost not active within the community would affect the overall assumed response time negatively since questions which are forwarded to not only active, but inactive users too, would in general receive fewer answers and later at least by the suggested answerers. Via removing those users who are not active the overall amount of users decreases and therefore, the calculation time per question drops significantly. Questions which are forwarded to recently active users have a much higher chance of receiving fast answers.

User based TF-IDF, in all conscience a novel interpretation of the IDF part, combined with VSM performs significantly better than the normal approach where IDF is based on the question collection. In general, QLLM yields better results, especially when it is applied in combination with Dirichlet smoothing which considers the profile size in order to determine the amount of smoothing. The other smoothing methods perform worse in the context of the evaluation based on a data set provided by Yahoo! Answers than this method.

The evaluation also showed that user profiles require a minimum amount of data in order to produce meaningful rankings. If the profiles are based on only one best answer and if only users are considered who have been active within the last 24 hours the sorting becomes unstable. It is assumed that this effect is mainly based on new users who just discover the community and who are assumed to give answers to not only one specific kind of questions. This means that new users gave already one best answer in, for example, public health and on the next day, which is in our case the test day, they give a best answer in the area of science fiction movies. This is a case which is not predictable. Therefore, user profiles have to be based on more than one best answered question. The results show that user profiles based on two best answered questions significantly improve ranking results. The overall sorting indicated by histograms is much better.

For knowledge estimation in combination with user authority and activity the results show that the Query Likelihood Language Model with Dirichlet smoothing in combination with cascading InDegree (CID) significantly outperforms the Vector Space Model based approach which also uses InDegree as authority component, but via linear combination. CID is applied via re-

moving all users who have less than a specified distinct in-link count prior to the actual calculation of the ranking of potential answerers.

The setup of the framework needs to be adapted to the context in which it is used. But further, it also has to be adapted to the dynamics of the Q&A community for which it is applied since the amount of as active considered users and the expected number of new questions need to be matched. This means that the Activity Filter should generally consider only a very short period of time in order to forward questions only to users who are likely to provide fast answers with the important constraint that there have to be still enough users under consideration in order to avoid jamming only a few recently active users with all new questions. Therefore, there has to be a balance between users considered for the ranking process and the number of new questions that are expected, established on a daily basis for example. The system is assumed to be updated within such a time period, too.

7.2 Future Work

The linear combination of the VSM based technique called U-TF-IDF with the authority estimation method called InDegree has a significant advantage, because QLLM-D does not appear to be compatible with the values of InDegree. The cascading InDegree (CID) has the disadvantage that it does not contribute to the actual ranking of potential answerers, because it only removes users who are assumed to lack authority. Therefore, a linear combination of QLLM-D with InDegree is desirable. Future work could further investigate the reasons and provide solutions.

A further challenge lies in avoiding spam caused by inappropriate content of best answered questions which would lead to misleading user profiles. If, for instance, several users answer their questions reciprocally and if they put specific terms very frequently in the question or answer text, it would affect the QLLM based ranking calculation since the raw term frequency is used for it. This way, users could manipulate their profiles in order to appear to be experts where in fact they might lack relevant knowledge. This is definitely something that needs attention.

APPENDIX A

Appendix

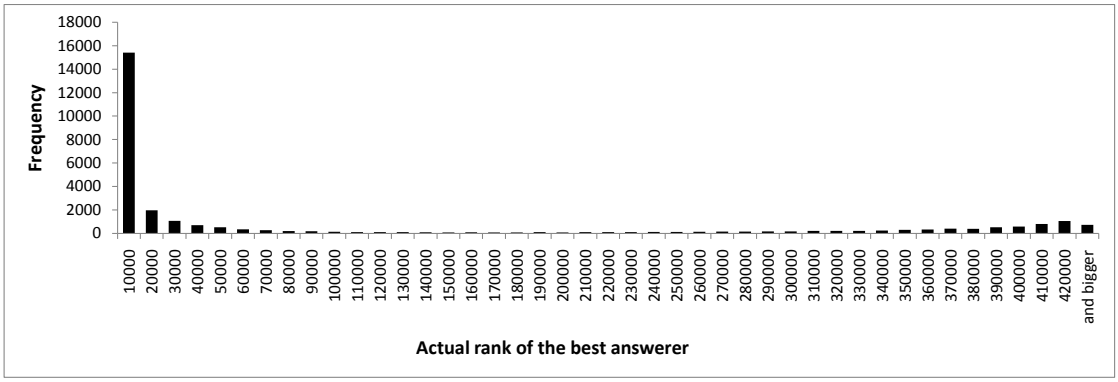

Figure A.1: Histogram of QLLM-Dirichlet based prediction with all users who gave at least one best answer

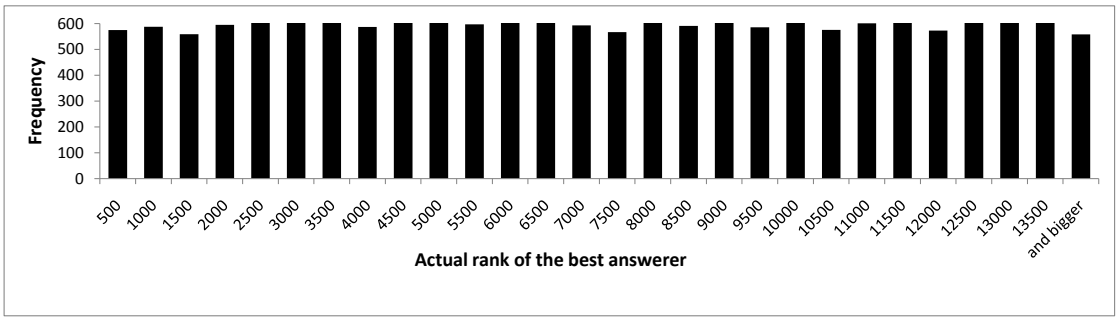

Figure A.2: Distribution of the actual best answerer via random prediction

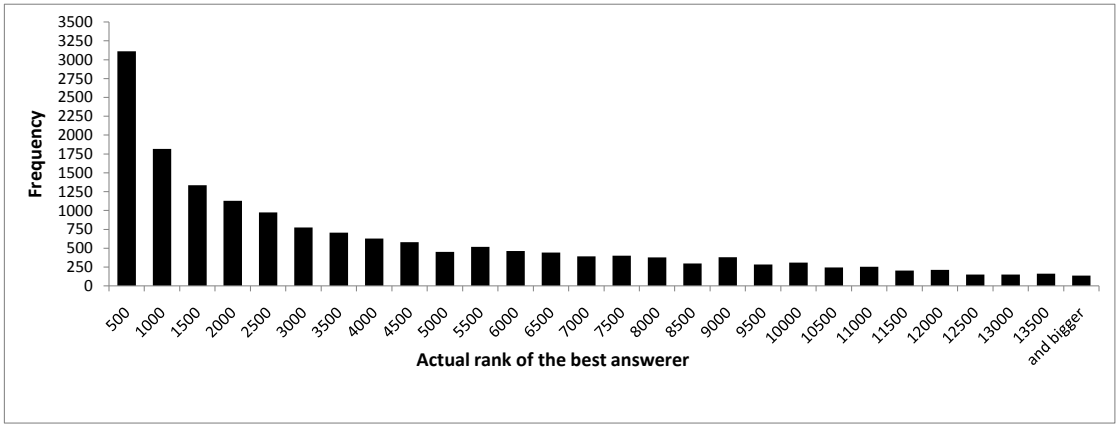

Figure A.3: Distribution of the actual best answerer via InDegree prediction

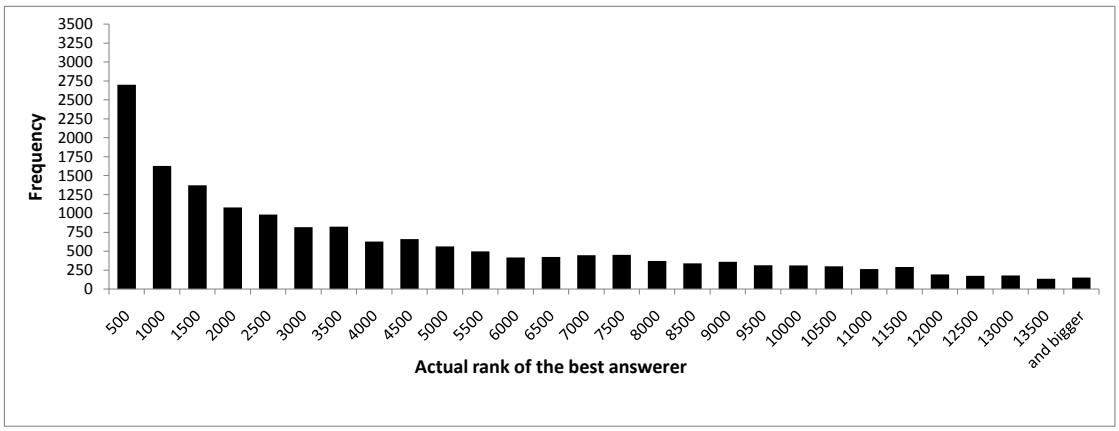

Figure A.4: Distribution of the actual best answerer via PageRank prediction

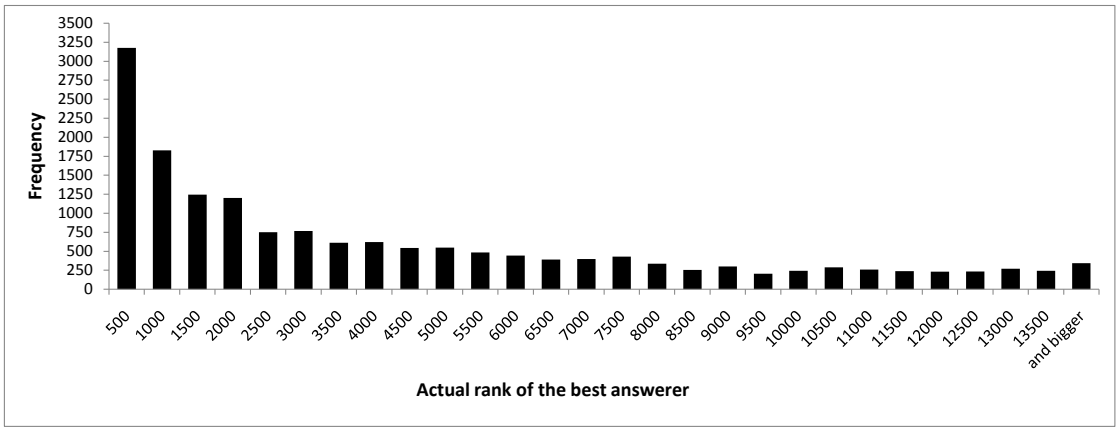

Figure A.5: Distribution of the actual best answerer via ZScore prediction

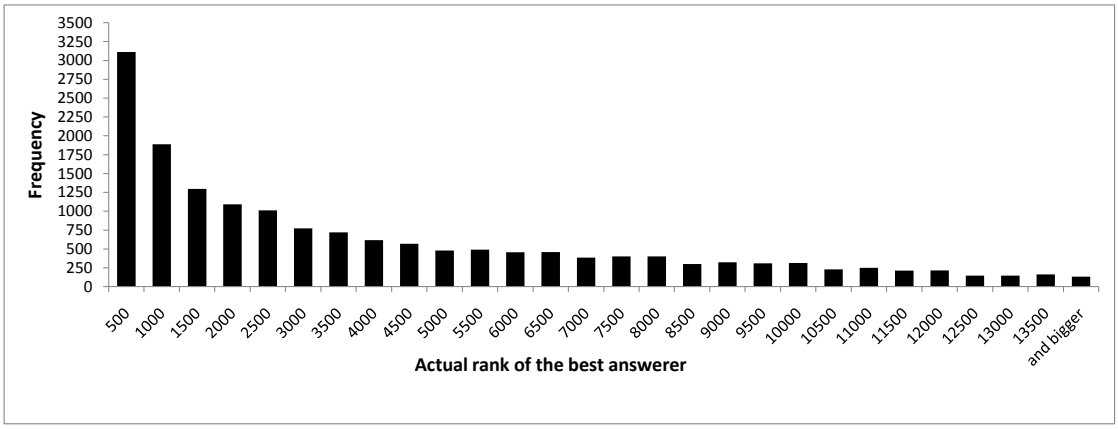

Figure A.6: Distribution of the actual best answerer via BAC prediction

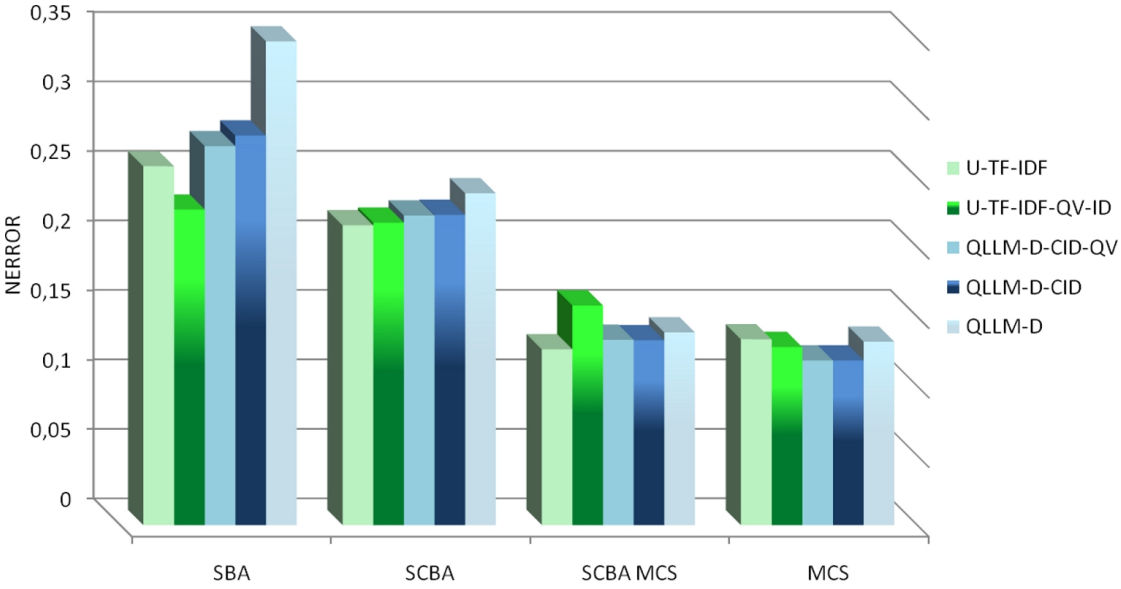

Figure A.7: Comparison of VSM with QLLM via the NERROR metric with users who have been active approximately within the last 24 hours (via Activity Filter) and who gave at least 2 best answers

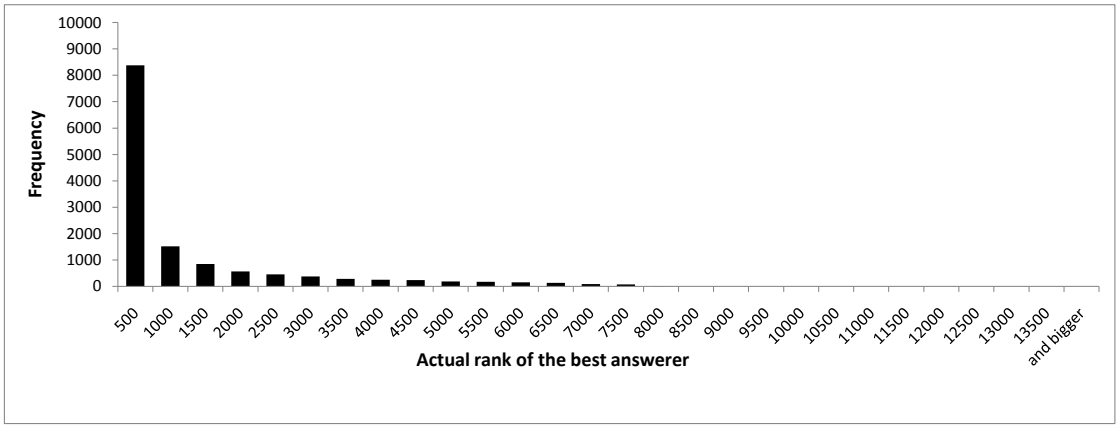

Figure A.8: Distribution of the actual best answerer via QLLM-D-CID prediction calculation

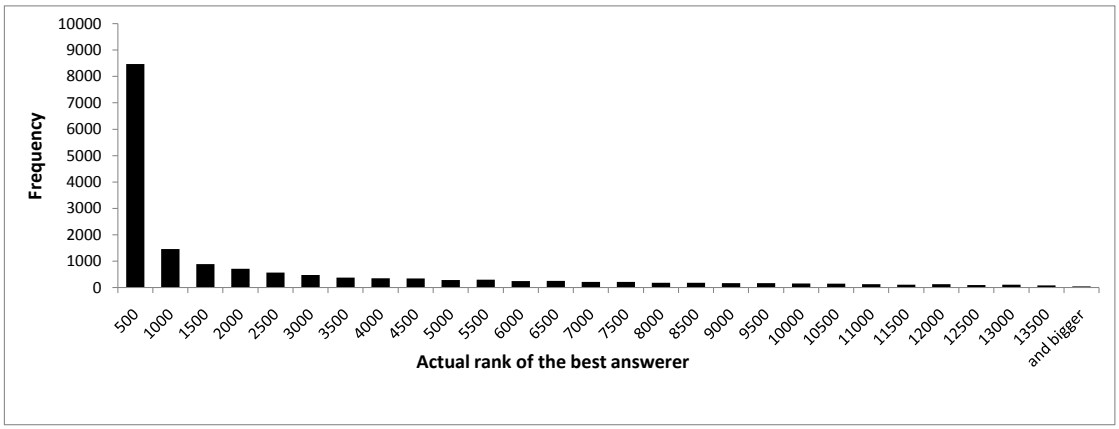

Figure A.9: Distribution of the actual best answerer via U-TFIDF-QV-ID prediction calculation

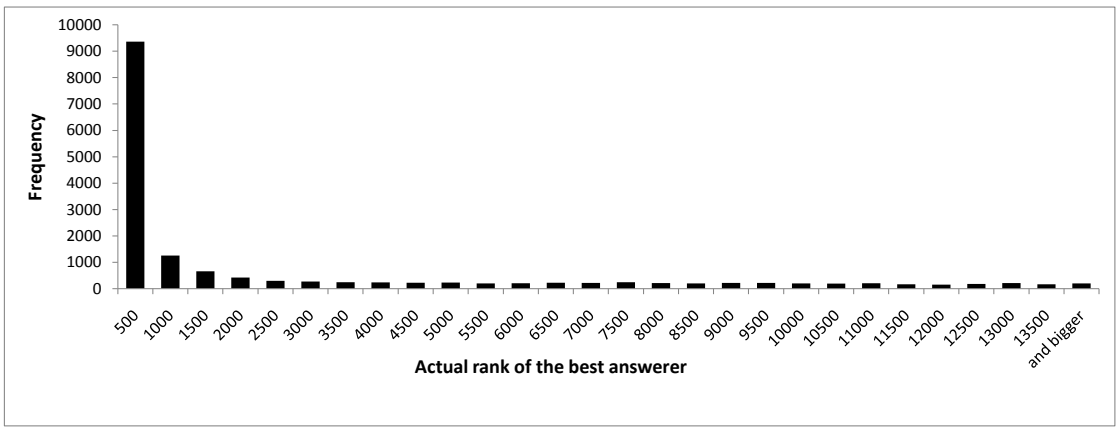

Figure A.10: Distribution of the actual best answerer via CBAC prediction calculation

Feature \ Q&A System	Yahoo! Answers	Askville	Answerbag	AOL Answers	Answers.com	LinkedIn Answers	Facebook Questions
Q&A support	x	x	x	x	x	x	x
Discussion support			x				x
Support for polls			x				x
Developing a knowledge base - searching	x	x	x	x	x	x	
Similar questions are shown when asking				x	x		
Account/login required to participate	x	x	x	x		x	x
Enforcing real people behind accounts		x					
Support of existing accounts	x		x	x	x	x	x
Check for question wording					x		
Manual question categorization	x	x	x	x	x	x	
Automatic question categorization		x					
Question expiration	x						
Anonymous questions (or answers)	x	x	x	x	x		
Private Questions						x	
Public questions	x	x	x	x	x	x	x
Question details	x	x	x			x	
Question constraints						x	
Question attachments				x			
Related links				x	x	x	
Automatic question routing/forwarding/suggesting	x			x			
Manual question routing/forwarding		x					x

Table A.1: Feature matrix 1/2

Feature \ Q&A System	Yahoo! Answers	Askville	Answerbag	AOL Answers	Answers.com	LinkedIn Answers	Facebook Questions
Reputation system	x	x	x	x	x		
Rating questions	x		x		x		
Rating/rewarding answers	x	x	x	x		x	x
Manual expertise specification/system				x		x	
Support for alternate questions					x		
Pre-answering questions					x		
Commenting on questions		x					
Commenting on answers	x		x	x			x
Direct communication							x
Posting questions on Social Networks (SN)	x		x	x		x	x
Sharing answered questions with SN					x		x
Editing submitted question and answers					x		
Removing questions or answers					x		x
Answer attachments		x	x	x			
Manually specifying user interests				x			x
Adding other users to friend list	x	x	x	x		x	x
Inviting friends (from a social network)				x		x	x
Blocking other users	x	x					x
Following question categories		x			x		
Following/watching questions	x				x		x
Spam and abuse reporting system	x		x	x	x	x	

Table A.2: Feature matrix 2/2

Bibliography

[1] Ackerman, M. S., Wulf, V., and Pipek, V. Sharing Expertise: Beyond Knowledge Management. MIT Press, Cambridge, MA, USA, 2002.

[2] Adamic, L. A., Zhang, J., Bakshy, E., and Ackerman, M. S. Knowledge sharing and yahoo answers: everyone knows something. In *Proceedings of the 17th international conference on World Wide Web*, WWW '08, pages 665–674, New York, NY, USA, 2008. ACM. doi: 10.1145/1367497.1367587.

[3] Amiri, H., Zarnani, A., Tavallaee, M., Abedinzadeh, Sadra andRahgozar, M., and Oroumchian, F. Investigation of the Lambda Parameter for Language Modeling Based Persian Retrieval. In *Proceedings of the Sixth International Conference on Informatics and Systems (INFOS'08)*, pages 39–44, March 2008.

[4] Blei, D. M., Ng, A. Y., and Jordan, M. I. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, Mar. 2003.

[5] Borodin, A., Roberts, G. O., Rosenthal, J. S., and Tsaparas, P. Link analysis ranking: algorithms, theory, and experiments. *ACM Trans. Internet Technol.*, 5(1):231–297, Feb. 2005. doi: 10.1145/1052934.1052942.

[6] Bouguessa, M., Dumoulin, B., and Wang, S. Identifying authoritative actors in question-answering forums: the case of Yahoo! answers. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '08, pages 866–874, New York, NY, USA, 2008. ACM. doi: 10.1145/1401890.1401994.

[7] Broder, A., Kumar, R., Maghoul, F., Raghavan, P., Rajagopalan, S., Stata, R., Tomkins, A., and Wiener, J. Graph structure in the Web. *Comput. Netw.*, 33(1-6):309–320, June 2000. doi: 10.1016/S1389-1286(00)00083-9.

[8] Chen, S. F., Chen, S. F., Goodman, J., and Goodman, J. An Empirical Study of Smoothing Techniques for Language Modeling. Technical report, 1998.

[9] Evans, B. M. and Chi, E. H. An elaborated model of social search. *Inf. Process. Manage.*, 46(6):656–678, Nov. 2010. doi: 10.1016/j.ipm.2009.10.012.

[10] Fagin, R., Kumar, R., and Sivakumar, D. Comparing top k lists. In *Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*, SODA '03, pages 28–36, Philadelphia, PA, USA, 2003. Society for Industrial and Applied Mathematics.

[11] Guo, J., Xu, S., Bao, S., and Yu, Y. 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, New York, NY, USA, 2008. ACM. doi: 10.1145/1458082.1458204.

[12] Gyongyi, Z., Koutrika, G., Pedersen, J., and Garcia-Molina, H. Questioning Yahoo! Answers. Technical Report 2007-35, Stanford InfoLab, 2007. URL `http://ilpubs.stanford.edu:8090/819/`.

[13] He, B. and Ounis, I. Query performance prediction. *Inf. Syst.*, 31(7):585–594, Nov. 2006. doi: 10.1016/j.is.2005.11.003.

[14] Herlocker, J. L., Konstan, J. A., Terveen, L. G., and Riedl, J. T. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, Jan. 2004. doi: 10.1145/963770.963772.

[15] Hofmann, T. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '99, pages 50–57, New York, NY, USA, 1999. ACM. doi: 10.1145/312624.312649.

[16] Horowitz, D. and Kamvar, S. D. The anatomy of a large-scale social search engine. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 431–440, New York, NY, USA, 2010. ACM. doi: 10.1145/1772690.1772735.

[17] Jelinek, F. and Mercer, R. L. Interpolated estimation of Markov source parameters from sparse data. In *In Proceedings of the Workshop on Pattern Recognition in Practice*, pages 381–397, Amsterdam, The Netherlands: North-Holland, May 1980.

[18] Jiao, J., Yan, J., Zhao, H., and Fan, W. ExpertRank: An Expert User Ranking Algorithm in Online Communities. In *Proceedings of the 2009 International Conference on New Trends in Information and Service Science*, NISS '09, pages 674–679, Washington, DC, USA, 2009. IEEE Computer Society. doi: 10.1109/NISS.2009.75.

[19] Jurczyk, P. and Agichtein, E. Discovering authorities in question answer communities by using link analysis. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, CIKM '07, pages 919–922, New York, NY, USA, 2007. ACM. doi: 10.1145/1321440.1321575.

[20] Kabutoya, Y., Iwata, T., Shiohara, H., and Fujimura, K. Effective Question Recommendation Based on Multiple Features for Question Answering Communities. In *Proceedings of the Fourth International Conference on Weblogs and Social Media*, Washington, DC, USA, 2010. Association for the Advancement of Artificial Intelligence, The AAAI Press.

[21] Kleinberg, J. M. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5): 604–632, Sept. 1999. doi: 10.1145/324133.324140.

[22] Kleinberg, J. M. Hubs, authorities, and communities. *ACM Comput. Surv.*, 31(4es), Dec. 1999. doi: 10.1145/345966.345982.

[23] Lavrenko, V. and Croft, W. B. Relevance based language models. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '01, pages 120–127, New York, NY, USA, 2001. ACM. doi: 10.1145/383952.383972.

[24] Li, B. and King, I. Routing questions to appropriate answerers in community question answering services. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, CIKM '10, pages 1585–1588, New York, NY, USA, 2010. ACM. doi: 10.1145/1871437.1871678.

[25] Li, B., King, I., and Lyu, M. R. Question routing in community question answering: putting category in its place. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, CIKM '11, pages 2041–2044, New York, NY, USA, 2011. ACM. doi: 10.1145/2063576.2063885.

[26] Liu, M., Liu, Y., and Yang, Q. Predicting best answerers for new questions in community question answering. In *Proceedings of the 11th international conference on Web-age information management*, WAIM'10, pages 127–138, Berlin, Heidelberg, 2010. Springer-Verlag.

[27] Liu, X. and Croft, W. B. Cluster-based retrieval using language models. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '04, pages 186–193, New York, NY, USA, 2004. ACM. doi: 10.1145/1008992.1009026.

[28] Liu, X., Croft, W. B., and Koll, M. Finding experts in community-based question-answering services. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, CIKM '05, pages 315–316, New York, NY, USA, 2005. ACM. doi: 10.1145/1099554.1099644.

[29] Liu, Y., Zhang, M., Cen, R., Ru, L., and Ma, S. Data cleansing for Web information retrieval using query independent features. *J. Am. Soc. Inf. Sci. Technol.*, 58(12):1884–1898, Oct. 2007. doi: 10.1002/asi.v58:12.

[30] Manning, C. D., Raghavan, P., and Schtze, H. Introduction to Information Retrieval. Cambridge University Press, Cambridge, England, 2009. Online edition (c) 2009 Cambridge UP.

[31] Miller, D. R. H., Leek, T., and Schwartz, R. M. A hidden Markov model information retrieval system. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '99, pages 214–221, New York, NY, USA, 1999. ACM. doi: 10.1145/312624.312680.

[32] Mooney, R. J. and Roy, L. Content-based book recommending using learning for text categorization. In *Proceedings of the fifth ACM conference on Digital libraries*, DL '00, pages 195–204, New York, NY, USA, 2000. ACM. doi: 10.1145/336597.336662.

[33] Morris, M. R., Teevan, J., and Panovich, K. What Do People Ask Their Social Networks, and Why? A Survey Study of Status Message Q&A Behavior. In *Proceedings of the 28th international conference on Human factors in computing systems*, CHI '10, pages 1739–1748, New York, NY, USA, 2010. ACM. doi: 10.1145/1753326.1753587.

[34] Page, L., Brin, S., Motwani, R., and Winograd, T. The PageRank Citation Ranking: Bringing Order to the Web. Technical Report 1999-66, Stanford InfoLab, November 1999. URL `http://ilpubs.stanford.edu:8090/422/`. Previous number = SIDL-WP-1999-0120.

[35] Pal, A. and Konstan, J. A. Expert identification in community question answering: exploring question selection bias. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, CIKM '10, pages 1505–1508, New York, NY, USA, 2010. ACM. doi: 10.1145/1871437.1871658.

[36] Ponte, J. M. and Croft, W. B. A language modeling approach to information retrieval. In *Proc. SIGIR*, 1998.

[37] Popescul, A., Ungar, L., Pennock, D., and Lawrence, S. Probabilistic Models for Unified Collaborative and Content-Based Recommendation in Sparse-Data Environments. In *Proc. of the Seventeenth Conference on Uncertainty in Artificial Intelligence (UAI-2001)*. San Francisco: Morgan Kaufmann, 2001.

[38] Qu, M., Qiu, G., He, X., Zhang, C., Wu, H., Bu, J., and Chen, C. Probabilistic question recommendation for question answering communities. In *Proceedings of the 18th international conference on World wide web*, WWW '09, pages 1229–1230, New York, NY, USA, 2009. ACM. doi: 10.1145/1526709.1526942.

[39] Refaeilzadeh, P., Tang, L., and Liu, H. Cross-Validation. In LIU, L. and ÖZSU, M. T., editors, *Encyclopedia of Database Systems*, pages 532–538. Springer US, 2009.

[40] Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. GroupLens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, CSCW '94, pages 175–186, New York, NY, USA, 1994. ACM. doi: 10.1145/192844.192905.

[41] Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, WWW '01, pages 285–295, New York, NY, USA, 2001. ACM. doi: 10.1145/371920.372071.

[42] Singhal, A. Modern Information Retrieval: A Brief Overview. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 24(4):35–42, 2001.

[43] Smucker, M. D. and Allan, J. An investigation of dirichlet prior smoothing's performance advantage. Technical report, 2005.

[44] Song, F. and Croft, W. B. A general language model for information retrieval. In *Proceedings of the eighth international conference on Information and knowledge management*, CIKM '99, pages 316–321, New York, NY, USA, 1999. ACM. doi: 10.1145/319950.320022.

[45] Turner, T. C., Smith, M. A., Fisher, D., and Welser, H. T. Picturing Usenet: Mapping Computer-Mediated Collective Action. *Journal of Computer-Mediated Communication*, 10(4), 2005.

[46] Wang, J., de Vries, A. P., and Reinders, M. J. T. Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '06, pages 501–508, New York, NY, USA, 2006. ACM. doi: 10.1145/1148170.1148257.

[47] Wasserman, S. and Faust, K. Social Network Analysis: Methods and Applications (Structural Analysis in the Social Sciences). Structural analysis in the social sciences, 8. Cambridge University Press, 1 edition, Nov. 1994.

[48] Witten, I. and Bell, T. The zero-frequency problem: estimating the probabilities of novel events in adaptive text compression. *Information Theory, IEEE Transactions on*, 37(4): 1085 –1094, jul 1991. doi: 10.1109/18.87000.

[49] Zhai, C. and Lafferty, J. A study of smoothing methods for language models applied to Ad Hoc information retrieval. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '01, pages 334–342, New York, NY, USA, 2001. ACM. doi: 10.1145/383952.384019.

[50] Zhai, C. and Lafferty, J. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22(2):179–214, Apr. 2004. doi: 10.1145/984321.984322.

[51] Zhang, J., Ackerman, M. S., and Adamic, L. Expertise networks in online communities: structure and algorithms. In *Proceedings of the 16th international conference on World Wide Web*, WWW '07, pages 221–230, New York, NY, USA, 2007. ACM. doi: 10.1145/1242572.1242603.

[52] Zhou, Y., Cong, G., Cui, B., Jensen, C. S., and Yao, J. Routing Questions to the Right Users in Online Communities. In *Proceedings of the 2009 IEEE International Conference on Data Engineering*, ICDE '09, pages 700–711, Washington, DC, USA, 2009. IEEE Computer Society. doi: 10.1109/ICDE.2009.44.

[53] Answerbag. http://www.answerbag.com. last accessed, April 2012.

[54] Answerbag, About. http://www.answerbag.com/about-us. last accessed, April 2012.

[55] Answerbag, Guidelines. http://www.answerbag.com/guidelines. last accessed, April 2012.

[56] Answers.com. http://www.answers.com. last accessed, April 2012.

[57] Answers.com, Help. http://wiki.answers.com/help/help_center. last accessed, April 2012.

[58] AOL Answers. http://aolanswers.com. last accessed, April 2012.

[59] The Apache Software Foundation. http://www.apache.org. last accessed, April 2012.

[60] AskMetaFilter. http://ask.metafilter.com. last accessed, April 2012.

[61] Askville. http://askville.amazon.com. last accessed, April 2012.

[62] Askville, Help. http://askville.amazon.com/faq.html. last accessed, April 2012.

[63] Baidu. http://www.baidu.com. last accessed, April 2012.

[64] Bing. http://www.bing.com. last accessed, April 2012.

[65] ChaCha. http://www.chacha.com. last accessed, April 2012.

[66] DOM Parser. http://www.w3schools.com/dom/dom_parser.asp. last accessed, April 2012.

[67] Eclipse. http://www.eclipse.org. last accessed, April 2012.

[68] Facebook. http://www.facebook.com. last accessed, April 2012.

[69] Facebook Questions. http://www.facebook.com/questions. last accessed, April 2012.

[70] Fluther. http://www.fluther.com. last accessed, April 2012.

[71] Google. http://www.google.com. last accessed, April 2012.

[72] Google+. https://plus.google.com. last accessed, April 2012.

[73] Internet world stats. http://www.internetworldstats.com/stats.htm. last accessed, April 2012.

[74] Java 1.6. http://www.oracle.com/technetwork/java/javase/overview/index.html. last accessed, April 2012.

[75] LinkedIn. http://www.linkedin.com. last accessed, April 2012.

[76] LinkedIn Answers. http://www.linkedin.com/answers. last accessed, April 2012.

[77] LinkedIn Learning Center. http://learn.linkedin.com/answers. last accessed, April 2012.

[78] Apache LOG4J. http://logging.apache.org/log4j/1.2. last accessed, April 2012.

[79] OODesign, Abstract Factory Pattern. http://www.oodesign.com/abstract-factory-pattern.html. last accessed, April 2012.

[80] Oshiete Goo. http://oshiete.goo.ne.jp. last accessed, April 2012.

[81] ReferenceAnswers. http://reference.answers.com. last accessed, April 2012.

[82] Simple API for XML. http://en.wikipedia.org/wiki/Simple_API_for_XML. last accessed, April 2012.

[83] SnowballStemmer. http://snowball.tartarus.org/index.php. last accessed, April 2012.

[84] Stanford Log-linear Part-Of-Speech Tagger. http://nlp.stanford.edu/software/tagger.shtml#About. last accessed, April 2012.

[85] StAX Parser. http://en.wikipedia.org/wiki/StAX. last accessed, April 2012.

[86] Tripadvisor. http://www.tripadvisor.com. last accessed, April 2012.

[87] Information Retrieval. http://www1.informatik.uni-mainz.de/lehre/ir/skript-sose-10/IR-SoSe10.pdf. last accessed, April 2012.

[88] Vector Normalization. http://pyevolve.sourceforge.net/wordpress/?tag=term-frequency. last accessed, April 2012.

[89] WikiAnswers. http://wiki.answers.com. last accessed, April 2012.

[90] Wikipedia, Bag of Words Model. http://en.wikipedia.org/wiki/Bag_of_words_model. last accessed, April 2012.

[91] Wikipedia, Information Retrieval. http://en.wikipedia.org/wiki/Information_retrieval. last accessed, April 2012.

[92] Wikipedia, Language Model. http://en.wikipedia.org/wiki/Language_model. last accessed, April 2012.

[93] Wikipedia, Mean Absolute Error. http://en.wikipedia.org/wiki/Mean_absolute_error. last accessed, April 2012.

[94] Wikipedia, Mean Reciprocal Rank. http://en.wikipedia.org/wiki/Mean_reciprocal_rank. last accessed, April 2012.

[95] Wikipedia, Mean Squared Error. http://en.wikipedia.org/wiki/Mean_squared_error. last accessed, April 2012.

[96] Wikipedia, PageRank. http://en.wikipedia.org/wiki/Pagerank. last accessed, April 2012.

[97] Wikipedia, Precision and Recall. http://en.wikipedia.org/wiki/Precision_and_recall. last accessed, April 2012.

[98] Wikipedia, QLLM. http://en.wikipedia.org/wiki/Query_likelihood_model. last accessed, April 2012.

[99] Wikipedia, Social Search. http://en.wikipedia.org/wiki/Social_search. last accessed, April 2012.

[100] Wikipedia, TF-IDF. http://en.wikipedia.org/wiki/Tf-idf. last accessed, April 2012.

[101] Wikipedia, Vector Space Model. http://en.wikipedia.org/wiki/Vector_space_model. last accessed, April 2012.

[102] Wikipedia, Data Cleansing. http://en.wikipedia.org/wiki/Data_cleansing. last accessed, April 2012.

[103] Wikipedia, LOG4J. http://en.wikipedia.org/wiki/Log4j. last accessed, April 2012.

[104] Wikipedia, World Population. http://en.wikipedia.org/wiki/World_population. last accessed, April 2012.

[105] Extensible Markup Language. http://www.w3.org/XML. last accessed, April 2012.

[106] Yahoo! http://www.yahoo.com. last accessed, April 2012.

[107] Yahoo! Answers. http://answers.yahoo.com. last accessed, April 2012.

[108] Yahoo! Answers Help. http://help.yahoo.com/l/us/yahoo/answers. last accessed, April 2012.

[109] Yahoo! Answers Statistics. http://help.yahoo.com/l/us/yahoo/answers/overview/overview-55778.html. last accessed, April 2012.

[110] Yahoo! Webscope dataset ydata-yanswers-all-questions-v1_0 . http://research.yahoo.com/Academic_Relations. last accessed, April 2012.