A Classification-based Approach to Question Answering in Discussion Boards

Liangjie Hong and Brian D. Davison
Department of Computer Science and Engineering
Lehigh University
Bethlehem, PA 18015 USA
{lih307,davison}@cse.lehigh.edu

ABSTRACT

Discussion boards and online forums are important platforms for people to share information. Users post questions or problems onto discussion boards and rely on others to provide possible solutions and such question-related content sometimes even dominates the whole discussion board. However, to retrieve this kind of information automatically and effectively is still a non-trivial task. In addition, the existence of other types of information (e.g., announcements, plans, elaborations, etc.) makes it difficult to assume that every thread in a discussion board is about a question.

We consider the problems of identifying question-related threads and their potential answers as classification tasks. Experimental results across multiple datasets demonstrate that our method can significantly improve the performance in both question detection and answer finding subtasks. We also do a careful comparison of how different types of features contribute to the final result and show that non-content features play a key role in improving overall performance. Finally, we show that a ranking scheme based on our classification approach can yield much better performance than prior published methods.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: H3.3 Information Search and Retrieval; H.4 [**Information Systems Applications**]: H4.3 Communications Applications—*Bulletin boards*

General Terms

Algorithm, Experimentation

Keywords

question answering, discussion boards, online forums, classification

1. INTRODUCTION

Discussion boards, also known as online forums, are popular web applications widely used in different areas including customer support, community development, interactive reporting and online education. Online users share ideas, discuss issues and form communities within discussion boards, generating a large amount of content on a variety of topics. As a result, interest in knowledge discovery and information extraction from such sources has increased in the research community.

While the motivation for users to participate in discussion boards varies, in many cases, people would like to use discussion boards as problem-solving platforms. Users post questions, usually related to some specific problem, and rely on others to provide potential answers. Numerous commercial organizations such as Dell and IBM directly use discussion boards as problem-solving solutions for answering questions and discussing needs posed by customers. Cong et al. [8] found that 90% of 40 discussion boards they investigated contain question-answering knowledge. Using speech acts analysis on several sampled discussion boards, Kim et al. [22, 21] showed that question answering content is usually the largest type of content on discussion boards in terms of the number of user-generated posts. Therefore, mining such content becomes desirable and valuable.

Mining question answering content from discussion boards has several potential applications. First, search engines can enhance search quality for question or problem related queries by providing answers mined from discussion boards. Second, online Question Answering (QA) services such as *Yahoo! Answers*[1], *Answers.com*[2] and *AllExperts*[3] would benefit from using content extracted from discussion boards as potential solutions or suggestions when users ask questions similar to what people have discussed on forums. This would eliminate the time users wait for answers and enrich the knowledge base of those QA services as well since discussion boards have a longer history than that of QA services and also own a much larger amount of user generated content. Third, users who often provide questions in forums may have certain expert knowledge in particular areas. Researchers are trying to find experts in social media by utilizing question answering content; authorities are discovered in discussion boards by understanding question answering content and user interactions [4, 33, 20]. In addition, question answering content extracted from discussion boards can be further used to augment the knowledge base of automatic chat-bots [11, 15].

Although general content mining of discussion boards has gained significant attention in recent years, the retrieval of question and potential answers from forums automatically and effectively is still a non-trivial task. Users typically start a thread by creating an initial post with arbitrary content and others reply to it

[1]http://answers.yahoo.com/
[2]http://www.answers.com/
[3]http://www.allexperts.com/

SIGIR'09, July 19–23, 2009, Boston, Massachusetts, USA.

in accordance with the type of the first post. For example, if the first post is about a question, following posts may contain similar experiences and potential solutions. If the first post is an announcement, following posts may contain clarifications, elaborations and acknowledgments. Hence, due to the existence of different types of information, we cannot assume that every thread on a discussion board is about a question, which makes discussion boards fundamentally different from QA services like Yahoo! Answers that are designed specifically for question answering. Additionally, the asynchronous nature of discussion boards makes it possible or even common for multiple users to pursue different questions in parallel within one thread.

In this paper, we explore the problem of extracting question answering content from discussion boards and divide it into two subtasks: identifying question-related first posts and finding potential answers in subsequent responses within the corresponding threads. We address both subtasks as classification problems and focus on the following research questions:

1. Can we detect question-related threads in an efficient and effective manner? In addition to the content itself, what other features can be used to improve the performance? How much can the combinations of some simple heuristics improve performance?

2. Can we effectively discover potential answers without actually analyzing the content of replied posts? Who contributes those posts and where do those posts usually appear?

3. Can this task be treated as a traditional information retrieval problem suitable to a relevance-based approach to the retrieval of question-answering content?

We choose several content-based and non-content based features and carefully compare them individually and also in combinations. We do not use any service- or dataset-specific heuristics or features (like the rank of users) in our classification model; therefore our approach should be usable in any discussion board. In order to test whether our method can improve performance in both subtasks, we mainly compare our approach with one recent similar work [8] (to our knowledge, the first to attack the same problem) and show significant improvements in experimental results.

The rest of this paper is organized as follows: We discuss related work below. Section 3 defines our tasks in more detail. Section 4 presents our features and gives a simple overview of other approaches from previous work. Experimental results are reported in section 5. Section 6 concludes the paper.

2. RELATED WORK

Although discussion boards are a popular destination for users looking for help, relatively little research directly addresses the problem of mining question answering content from discussion boards.

Cong et al. [8] was the first to address a problem similar to what we discuss in this paper. They developed a classification-based method for question detection by using sequential pattern features automatically extracted from both questions and non-questions in forums. They preprocessed each sentence from the first posts by applying a Part-Of-Speech (POS) tagger while keeping keywords including 5W1H (What, When, Why, Where, Which and How) words and modal words. The sequential pattern features are based on the results of the POS tagger. Though achieving reasonable performance, this approach suffers from the typically time-consuming POS analysis process. More importantly, the definition of "questions" in their work is slightly different from our work. They focused on question sentences or question paragraphs while we treat the first post as a whole if it is about a question. For the subtask of finding answers, they proposed an unsupervised graph-based approach for ranking candidate answers leveraging the relevance between replied posts, the similarity between the replied post and the first post, and author information as well. We will show that our method outperforms their approach both in effectiveness and efficiency.

A second related work is that of Ding et al. [9] who proposed a general framework based on Conditional Random Fields (CRFs) to detect the contexts and answers of questions from forum threads. They did not address the question detection subtask in the work and their approach is a complicated method that may not be applied to larger datasets. Some features they used within the framework are the same as what we will use in this paper. However, they did not provide a careful comparison of those features and show how different features contribute to the results.

In addition to these two directly related papers, there is some research on knowledge acquisition from discussion boards. Zhou and Hovy [34] presented a summarization system utilizing the input-reply pairs extracted from online chat archives. Their system is not specifically designed for question answering content. Feng et al. [11] proposed a system to automatically answer students' queries by matching the reply posts from an annotated corpus of archived threaded discussions with students' queries, which is a different problem from our work. Huang et al. [15] presented an approach for extracting high-quality <thread-title, reply> pairs as chat knowledge from online discussion boards so as to efficiently support the construction of a chat-bot for a certain domain. They also did not focus on question related threads in discussion boards.

Other previous work was trying to understand and mine discussion boards for more general purposes. Antonelli and Sapino [2] proposed a system to classify discussion threads based on rules derived by using both speech acts and graph analysis. Although their system can identify questions and answers as well as other types of threads, their dataset was small and they only provided precision measures in their experimental results. Kim et al. [22, 21] and Feng et al. [12] used speech acts analysis to mine and assess discussion boards for understanding students' activities and conversation focuses. They used only a small dataset and did not address question answering content in their work. Lin and Cho [23] introduced several techniques to preprocess questions extracted from discussion board including "garbage text" removal, question segmentation and merging questions. They did not discuss how to identify question content and their answers. Shrestha et al. [27] detected interrogative questions using a classification method and built a classifier to find answers using lexical features based on similarity measurement and email-specific features.

Compared to the problem we address, extensive research has been done on QA services like Yahoo! Answers or other Frequent Asked Questions (FAQ) services. Jeon et al. [17, 16], Duan et al. [10], and Cao et al. [5] tackled the problem of finding questions in the QA services that are semantically similar to a user's question. Song et al. [28] proposed a metric "question utility" for studying usefulness of questions and showed how question utility can be integrated into question search as static ranking. Jeon et al. [18] presented a framework for using non-textual features like click counts to predict the quality of answers, incorporated with language modeling-based retrieval model. Surdeanu et al. [29], Xue et al. [32], Berger et al. [3], Jijkoun et al. [19], and Riezler et al. [26] described various retrieval models or systems to extract

answers from QA or FAQ services. Liu et al. [24] proposed automatic summarization techniques to summarize answers for re-use purposes. Gyongyi et al. [13] performed an analysis of 10 months of Yahoo! Answers data that provided insights into user behavior and impact as well as into various aspects of the service and its possible evolution. Some of the above work is complementary to our approach, and therefore could be employed to enhance our methods but in general all work above does not need to detect questions.

Traditional Question Answering tasks in TREC style have been well studied; see for example Vorhees [31]. That work mainly focused on constructing short answers for a relatively limited types of questions, such as factoid questions, from a large corpus. This makes it possible to identify the answer type. In contrast, typical questions extracted in discussion boards are more complex and usually consist of multiple sentences or even several paragraphs, and it is also difficult to represent and identify answer types for those questions.

3. PROBLEM DEFINITION

In this section, we discuss the problem in detail and then present a definition of the problem.

3.1 Questions

If the first post of one thread is about a specific problem that needs to be solved, we would consider that post as a whole to be a question post. We do not focus on identifying "question sentences" or "question paragraphs" but instead to find whether the first post is a "question post". Since users often express their problems in an informal way and questions are stated in various formats, it is difficult to recognize questions at the sentence or even paragraph level.

For example, the following paragraph is a question post from UbuntuForums.org, the official discussion board of Ubuntu Linux.

> There are a number of threads on Firefox crashes, so it's nothing new. I upgraded from U8.04 to U8.10, but it's no better. Then I tried Seamonkey, and it worked fine for a couple of days. Now it too is crashing. I'm baffled. Anyone have any ideas what I can do?

Although the last sentence is a question sentence, it gives us little information about what the real problem is. The true problem is the scenario the author described with several sentences as a whole. This post has another paragraph providing machine configurations which we do not include here. Therefore, it is reasonable to treat the whole post as a question post.

If there are multiple questions discussed in the first post, the interaction in following replied posts might become complex (e.g., users may answer all those questions while others may only response to some of them). To simplify the task, we treat it as a single question post.

3.2 Answers

If one of the replied posts contains answers to the questions proposed in the first post, we regard that reply as an answer post. As we discussed above, we do not consider the number of answers should match the number of questions. Additionally, we only consider those replies that directly answer the questions from the first post. We ignore other questions (usually elaborated from the original ones) within replied posts and their corresponding answers. Although such answers may provide more information to the original questions and therefore could be potential better answers, in reality, users need to understand all replied posts above to get an overall idea and answers would become less meaningful if we only extract that single reply as the answer to the first post.

We also consider replied posts not containing the actual content of answers but providing links to other answers as answer posts. If multiple posts provide links to other potential answers, we treat the first one as the answer post.

3.3 Definition

A discussion board is a collection of threads. Each thread consists of the first post and following replied posts. Our task is:

1. To detect whether the first post is a "question post" containing at least one problem needed to be solved.
2. If the first post is a "question post", try to identify the best answer post either directly answering at least one question proposed in the first post or pointing to other potential answer sources.

Therefore, the result from our system is question-answer post pairs. Ideally, users do not need other information (e.g., the posts between them) to understand these pairs.

4. CLASSIFICATION METHODS

We consider both subtasks described in Section 3 as classification problems. In this section, we introduce the features we use and a brief review of previous approaches.

4.1 Question Detection

For this subtask, we describe and use several features other researchers have used previously (e.g., question mark, 5W1H words) as well as features that are borrowed from other fields (e.g., N-gram).

- Question mark: If users want to ask a question, they may express it in a question sentence and therefore the sentence may contain a question mark at the end.
- 5W1H Words: If there is a question sentence, users probably would use 5W1H words in it.
- Total number of posts within one thread: From our empirical study we found that if one thread has many posts, either the topic of the thread probably shifts or the original first post may not contain enough information and hence further clarifications or elaborations are needed. Both cases are not in our problem definition.
- Authorship: Who would usually ask questions? Recent work shows that high quality content is generated by highly authoritative authors in social media (e.g., Agichtein et al. [1] and Hu et al. [14]). In our context, we consider high quality contents to be answers and highly authoritative authors are users who usually answer others' questions. Therefore, by contrast, fresh users are more likely to post questions rather than answering questions and a large portion of total posts (including all replies) a fresh user makes are likely all questions.
- N-gram: Carvalho and Cohen [6] suggested that n-grams would improve speech acts analysis on E-mail. The task is similar to our work and therefore we would like to see whether this feature works for discussion boards.

In summary, we use the number of question marks, the number of each 5W1H words, total number of posts within one thread and authorship (the number of posts one user starts and the number of posts one user replies) as features.

4.2 Answer Detection

In this subtask, we focus on how to detect answer posts without analyzing the content of each post using natural language processing techniques. We are also interested in how non-content features can contribute to classification results.

- The position of the answer post: According to our definition of the problem, we notice that the answer post usually appears not very close to the bottom if the question receives a lot of replies.
- Authorship: Same as the last subtask.
- N-gram: Same as the last subtask.
- Stop words: Although "stop words" are usually regarded as "noise words", we want to see whether the author of answer posts would use more detailed and precise words rather than "stop words", in contrast to other types of posts such as elaborations, suggestions and acknowledgment.
- Query Likelihood Model Score (Language Model): We use this basic language model method to calculate the likelihood that a replied post is relevant to the original question post. We use this feature as an example to show how a relevance-based model performs in the task.

In summary, we use the position of the answer post, the authorship, N-gram, the count of each stop word and the score of Query Likelihood Model as features.

4.3 Other methods

We principally compare our method with the approaches introduced by Cong et al. [8], a recent work addressing a similar problem. We briefly review their method below.

To detect the questions, they used the supervised learning approach Sequential Pattern Mining. First, each sentence is preprocessed by a POS tagger only leaving 5W1H words and modal words. Then the sequential patterns are generated by a modified version of the PrefixSpan algorithm [25] to incorporate both minimum support and minimum confidence, which are assigned empirically. They treat all generated patterns as features. They considered "finding answers" as a retrieval problem. The retrieval model they introduced is a graph-based model incorporated with inter-posts relevance, authorship and the similarity between replied posts and the first post. They showed two variations of the graph-based model that one is combined with the Query Likelihood language model and another is combined with the KL-divergence language model.

We implement all these methods and compare them in our experiments. Note that they did not explicitly define "question" and "answer". Therefore, our task may be slightly different from theirs.

5. EXPERIMENTS

5.1 Data and Experiment Settings

We selected two discussion boards as our data source. We crawled 721,442 threads from Photography On The Net[4], a digital camera forum (DC dataset), and 555,954 threads from UbuntuForums[5], an Ubuntu Linux community forum (Ubuntu dataset).

Table 1: The Features and Their Abbreviations

Features	Abbrev.
Question Mark	QM
5W1H Words	5W
Total # Posts	LEN
Sequential Patterns	SPM
N-grams	NG
Authorship	AUTH
Position	POSI
Query Likelihood Model	LM
Stop Words	SW
Graph+Query Likelihood Model	GQL
Graph+KL-divergence Model	GKL

For the question detection subtask, we randomly sampled 572 threads from the Ubuntu dataset and 500 threads from the DC dataset. We manually labeled all first posts in these threads into question posts and non-question posts using our criteria introduced in Section 3. For answer detection subtask, we selected 500 additional question-related threads from both data sources. Therefore, we have 2,580 posts in total (including the first posts) from the Ubuntu dataset and 3,962 posts in total (including the first posts) from the DC dataset. We manually labeled all posts into answers and non-answers. We note that in accordance with our problem definition, only one answer post per thread is labeled as such (the remainder are labeled as non-answers).

We preprocessed all posts by modifying possible abbreviations into their full form (e.g., "we're" into "we are", "it's" into "it is") and stemming all words. For Sequential Pattern Mining, the Stanford Log-linear Part-Of-Speech Tagger [30] was used and minimum support and minimum confidence were set to 1.5% and 80% respectively. For N-gram, we generated 3,114 N-grams (1-5 grams) from the Ubuntu dataset and 1,604 N-grams from DC dataset for question detection while 2,600 N-grams from Ubuntu dataset and 1,503 N-grams from DC dataset for answer detection. For stopwords, we used 571 normal stop words.[6] We use LIBSVM 2.88 [7] as our classifier and all classification results are obtained through 10-fold cross validation. In order to avoid classification bias and get better results, we balanced our data into around 50% positive samples versus 50% negative samples in all experiments. For example, we have 500 positive instances and 2080 negative instances for answer detection on Ubuntu dataset. Therefore, we replicated the positive training instances four times to give 2,000 examples (but left the test set unchanged). Since in any real settings, the data is inherently skewed, a better learning approach such as cost-sensitive learning may be more realistic. Table 1 shows all the features we used and their abbreviations.

5.2 Question Detection

We first evaluate the performance of features introduced in Section 4.1 individually. Table 2 gives the results of precision, recall, F-measure and accuracy (sorted by accuracy) of the Ubuntu dataset and Table 3 shows the results from the DC dataset. It is easily to notice that *Length*, *5W1H* and *Question Mark*, three simple heuristics, generally cannot give good performance while *Sequential Pattern Mining* always outperforms these simple methods on

[4]http://photography-on-the.net/

[5]http://ubuntuforums.org/

[6]http://www.lextek.com/manuals/onix/stopwords2.html

Table 2: Single Feature Ubuntu Question

Features	Prec.	Recall	F1	Accu.
LEN	0.568	0.936	0.707	0.623
5W	0.613	0.759	0.679	0.651
QM	0.649	0.634	0.641	0.656
AUTH	0.700	0.725	0.712	0.716
SPM	0.692	0.829	0.754	0.738
NG	0.770	0.906	0.833	0.823

Table 3: Single Feature DC Question

Features	Prec.	Recall	F1	Accu.
5W	0.601	0.429	0.500	0.579
LEN	0.564	0.730	0.636	0.590
QM	0.578	0.779	0.664	0.612
SPM	0.642	0.702	0.671	0.661
AUTH	0.723	0.791	0.755	0.748
NG	0.752	0.799	0.775	0.772

both datasets, which validates the experiments performed by Cong et al. [8]. Additionally, the results show that *Authorship* is a much better heuristic and can achieve reasonable performance compared with *Sequential Pattern Mining* although it seems that performance may be dataset dependent. On both dataset, *N-grams* achieves the best performance in all metrics in terms of a single feature. This suggests that users do use certain language patterns to express problems and questions in discussion boards. Table 4 shows 10 sample N-grams extracted from DC dataset that were used for question detection. Note that the results are stemmed words.

Since *N-grams* and *Sequential Pattern Mining* (which requires a POS tagger) are relatively complicated methods (vs. simple heuristics such as finding question marks and 5W1H words), the computational effort may be impractical for large datasets. In order to avoid high computation methods, we do further experiments on the combinations of those simple methods and see whether the performance can improve and therefore make simple combinations viable alternatives.

Tables 5 and 6 show the combinations of simple features compared to *N-grams* and *Sequential Pattern Mining*. We observe that the performance can be improved by combining features. Specifically, *Authorship+Question Mark+5W1H Words+Length* achieved similar or even better results than *Sequential Pattern Mining* on both datasets. Notice that the computation of these features is much simpler than *Sequential Pattern Mining*. In addition, *Question Mark+5W1H Words+Length*, which only requires local information, also achieved reasonable performance compared to those features individually since *Authorship* needs global information. From these results, we found that although these features individually cannot give much evidence reflecting whether a post concerns a question, the combination of them is able to characterize the first post and interestingly, none of these simple features attempts to understand the real semantics of the question posts.

Table 4: Example N-grams from DC Question Dataset

i do not know if	i wa wonder if anyon
what is the best way	i do not have
i am not sure	do not know what
i am look for	i can not
do not know	would like to

Table 5: Combined Features Ubuntu Question

Method	Prec.	Recall	F1	Accu
QM+LEN	0.657	0.655	0.656	0.666
AUTH+LEN	0.679	0.757	0.716	0.708
5W+LEN	0.673	0.821	0.740	0.719
QM+5W	0.756	0.636	0.691	0.723
QM+5W+LEN	0.744	0.701	0.722	0.738
SPM	0.692	0.829	0.754	0.738
AUTH+QM+5W+LEN	0.731	0.762	0.746	0.748
NG	0.770	0.906	0.833	0.823

Table 6: Combined Features DC Question

Method	Prec.	Recall	F1	Accu.
QM+5W	0.614	0.764	0.681	0.648
5W+LEN	0.627	0.709	0.666	0.650
SPM	0.642	0.702	0.671	0.661
QM+LEN	0.656	0.764	0.706	0.687
QM+5W+LEN	0.672	0.755	0.711	0.698
NG	0.752	0.799	0.775	0.772
AUTH+LEN	0.813	0.874	0.843	0.839
AUTH+QM+5W+LEN	0.863	0.889	0.876	0.876

5.3 Answer Detection

For this subtask, we first did the experiments using individual features, as we did in Question Detection. In order to compare with the methods introduced by Cong et al. [8], we used the ranking score from their retrieval models as a feature to train our classifier. Since *Graph-based model+Query Likelihood Model* and *Graph-based model+KL-divergence Model* perform similarly on both datasets (shown later in Section 5.4), we only use *Graph-based model+Query Likelihood Model* in this subtask as an example.

Tables 7 and 8 show the experimental results. In general, *Language Model* and *Graph+Query Likelihood Model* did not perform well using the ranking score as features. A possible reason is that these methods are mainly based on relevance retrieval models, which aim to find the information most relevant to the query (in our case, the question posts). Since all posts within a question thread may be more or less relevant to the question, it is difficult to rank them and distinguish the best answers from others based on content relevance or similarity measurement. In addition, relevance-based models may unable to handle big lexical gaps between questions and answers. We show one example from UbuntuForums below.

The first post:

```
can any one help me load ubuntu
8.10 on to my pc?  i have a asus AS
V3-P5V900 but when i load from cd it
keeps crashing , i think i dose not
reconise the graphics card.  when i
boot from cd it asks me what lauguge
ENGLISH then when try to load it
crash again i have tryed help and
put in via=771 any help please ?
```

The answer post:

```
You might try using the
"Alternate" install CD:
http://www.ubuntu.com/getubuntu/
downloadmirrors#alternate
```

Table 7: Single Feature Ubuntu Answer

Method	Prec.	Recall	F1	Accu.
GQL	0.673	0.575	0.620	0.650
Stopword	0.665	0.617	0.640	0.655
NG	0.690	0.638	0.663	0.678
LM	0.717	0.650	0.682	0.699
POSI	0.743	0.730	0.737	0.712
AUTH	0.715	0.823	0.765	0.721

Table 8: Single Feature DC Answer

Method	Prec.	Recall	F1	Accu.
GQL	0.661	0.535	0.591	0.628
LM	0.726	0.603	0.659	0.685
AUTH	0.680	0.800	0.735	0.710
NG	0.735	0.680	0.706	0.716
Stopword	0.730	0.696	0.712	0.717
POSI	0.780	0.88 0	0.827	0.815

Notice that this answer post contains a web link while all "keywords" (e.g., ubuntu 8.10, asus AS V3-P5V900, crash and etc.) in the first post do not appear in the answer post. If we calculate Query Likelihood Model score for the answer post, nearly all words in the question post can only receive "background" smoothing score and hence the model would rank this post "irrelevant". Essentially the same situation happens when using similarity measurement (e.g., cosine similarity).

N-gram did not outperform other features in this subtask, which suffers from various expressions in answer posts. Interestingly, the *Stopword* approach has performance similar to *N-gram* in both datasets. *N-gram* usually requires more computational effort than *Stopword* since *Stopword* has a fixed number of features for all datasets while *N-gram* needs to be generated separately and usually contains thousands of features. Therefore, in our later experiments, we use *Stopword* instead of *N-gram*. We also note that *Authorship* and *Position*, two simple heuristics, perform reasonably well and achieve comparatively high F1-Score on both datasets.

Inspired by question detection subtask, we conducted experiments using combinations of features on the two datasets. Tables 9 and 10 provide the corresponding results. In this subtask, we not only combine simple heuristics but also combine non-content features and content-based features. The first interesting finding is that *Position+Authorship* outperforms all other feature combinations and greatly improves the performance. This would explain that senior members usually answer questions in certain positions (e.g., near to the top post). This combination is easy to compute and there are no other parameters to tune.

In order to better understand how these two features contribute to the final results, we plot them in Figure 1 and Figure 2 for both datasets. The X-axis shows the ratio of the number of starting posts versus follow-up posts for users who answered questions in our datasets. The Y-axis shows the ratio of the position of answer posts from the top of the thread versus to the bottom. Both figures demonstrate the obvious signal that most answer posts are close to the top when the author of these posts are senior users who usually write replies rather than starting posts.

We also notice that the combination of content-based features (e.g., *Language Model*, *Stop words*) and non-content features (e.g., *Position*, *Authorship*) may also get better results compared to Table 7 and Table 8. The *Position+Stopword* combination performed reasonably well on both datasets, only requires local information, and

Table 9: Combined Features Ubuntu Answer

Method	Prec.	Recall	F1	Accu.
LM+GQL	0.726	0.718	0.722	0.695
Stopword+NG	0.735	0.786	0.760	0.726
LM+POSI	0.733	0.812	0.770	0.733
LM+Stopword	0.758	0.764	0.761	0.735
LM+AUTH	0.739	0.840	0.786	0.748
POS+Stopword	0.785	0.811	0.798	0.773
LM+POSI+Stopword	0.785	0.814	0.799	0.774
LM+POSI+AUTH	0.929	0.964	0.946	0.940
POSI+AUTH	0.935	0.969	0.952	0.946

Table 10: Combined Features DC Answer

Method	Prec.	Recall	F1	Accu.
LM+GQL	0.735	0.594	0.657	0.688
LM+AUTH	0.700	0.771	0.734	0.719
Stopword+NG	0.737	0.688	0.712	0.720
LM+Stopword	0.765	0.717	0.740	0.747
LM+POSI	0.780	0.879	0.827	0.815
LM+POSI+Stopword	0.846	0.899	0.872	0.867
POSI+Stopword	0.846	0.901	0.873	0.868
LM+POSI+AUTH	0.951	0.991	0.970	0.970
POSI+AUTH	0.958	0.993	0.975	0.975

is simpler than any kind of relevance-based features. In general, we can see that performance benefits from a combination of features, especially those simple features. Additionally, the combination of non-content and content features also improves performance significantly.

5.4 Other Experiments

We also propose a simple ranking scheme based on the classification method. The ranking score is simply computed by linearly combining position and authorship information:

$$s = \alpha * V_1 + (1 - \alpha) * \beta * V_2 + (1 - \alpha) * (1 - \beta) * V_3$$

where V_1,V_2 and V_3 are scores from classifiers of combination of position and authorship, position only and authorship only respectively. α and β are empirical parameters and we set 0.6 to both of them.

Table 11 shows the results compared to basic Query Likelihood Language Model, Graph-based+KL-divergence model proposed by [8] in terms of Precision@1 and Mean Reciprocal Rank (MRR) where MRR is the mean of the reciprocal ranks of the answers over a set of questions. Our ranking scheme outperforms other previous relevance-based approaches.

6. CONCLUSION

In this paper we defined the problem of selecting Question and Answer post pairs from discussion boards and addressed it as a classification problem. The contributions of this paper include:

1. We show that the use of N-grams and the combination of several non-content features can improve the performance of detecting question-related threads in discussion boards.

2. We show that the number of posts a user starts and the number of replies produced and their positions are two crucial factors in determining potential answers.

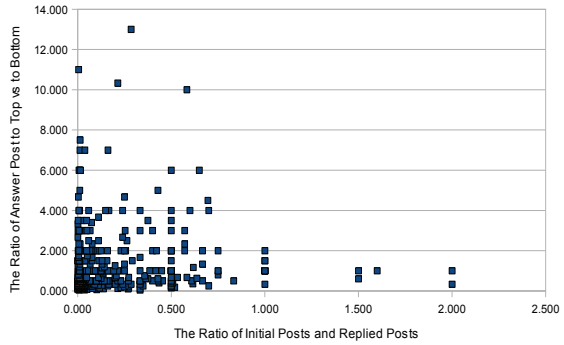

Figure 1: Authorship and Position on Ubuntu

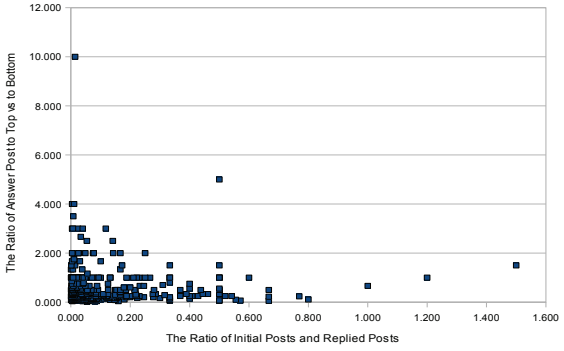

Figure 2: Authorship and Position on DC

3. We show that relevance-based retrieval methods would not be effective in tackling the problem of finding possible answers but the performance can be improved by combining with non-content features while we treat retrieval scores as features.

4. Using classification results, we are able to design a simple ranking scheme that outperforms previous approaches when retrieving potential answers from discussion boards.

Future work might consider the following problems.

1. This work only addresses answer posts that directly answered the question posts. The more realistic problem is how we can model the questions expanded in later posts and the answers to those expanded questions. Can we extract useful sentences from the elaborative posts that clarify the original question or expand the question and "feed back" into the original question post to help understand the question? Can we combine several potential answer posts together to make a better answer post?

2. This work does not consider the number of questions in the question posts. Can we separate multiple questions within the question posts? If so, can we find corresponding answers and represent them in a reasonable way?

This work explicitly defines the problem of selecting question answering post pairs from discussion boards and shows better performance compared to previous approaches. We believe that this is a first step toward a better understanding of the interaction of question answering in such media.

Table 11: Ranking Scheme

	Ubuntu		DC	
Method	P@1	MRR	P@1	MRR
LM	0.352	0.559	0.274	0.468
GQL[8]	0.360	0.570	0.220	0.414
GKL[8]	0.358	0.556	0.223	0.415
POSI+AUTH	0.902	0.949	0.928	0.964

Acknowledgments

This work was supported in part by a grant from the National Science Foundation under award IIS-0545875. We appreciate the thoughtful discussions with XiaoGuang Qi, Na Dai and Jian Wang.

7. REFERENCES

[1] E. Agichtein, C. Castillo, D. Donato, A. Gionis, and G. Mishne. Finding high-quality content in social media. In *Proceedings of the 1st International ACM Conference on Web Search and web Data Mining (WSDM)*, pages 183–194, New York, NY, 2008. ACM.

[2] F. Antonelli and M. Sapino. A rule based approach to message board topics classification. In *Advances in Multimedia Information Systems*, pages 33–48, 2005.

[3] A. Berger, R. Caruana, D. Cohn, D. Freitag, and V. Mittal. Bridging the lexical chasm: Statistical approaches to answer-finding. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 192–199, New York, NY, 2000. ACM.

[4] M. Bouguessa, B. Dumoulin, and S. Wang. Identifying authoritative actors in question-answering forums: The case of Yahoo! answers. In *Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 866–874, New York, NY, 2008. ACM.

[5] Y. Cao, H. Duan, C.-Y. Lin, Y. Yu, and H.-W. Hon. Recommending questions using the MDL-based tree cut model. In *Proceeding of the 17th international conference on World Wide Web (WWW)*, pages 81–90, New York, NY, USA, 2008. ACM.

[6] V. R. Carvalho and W. W. Cohen. Improving email speech acts analysis via n-gram selection. In *Proceedings of the HLT/NAACL 2006 Analyzing Conversations in Text and Speech Workshop (ACTS)*, pages 35–41, New York City, NY, June 2006. Association for Computational Linguistics.

[7] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available from `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[8] G. Cong, L. Wang, C.-Y. Lin, Y.-I. Song, and Y. Sun. Finding question-answer pairs from online forums. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 467–474, New York, NY, 2008. ACM.

[9] S. Ding, G. Cong, C. Lin, and X. Zhu. Using conditional random fields to extract contexts and answers of questions from online forums. In *Proceedings of 46th Annual Meeting of the Association for Computational Linguistics: Human Language Tchnologies (ACL:HLT)*, pages 710–718, Columbus, OH, June 2008.

[10] H. Duan, Y. Cao, C.-Y. Lin, and Y. Yu. Searching questions by identifying question topic and question focus. In *Proceedings of 46th Annual Meeting of the Association for*

Computational Linguistics: Human Language Tchnologies (ACL:HLT), Columbus, OH, June 2008.

[11] D. Feng, E. Shaw, J. Kim, and E. Hovy. An intelligent discussion-bot for answering student queries in threaded discussions. In *Proceedings of the 11th International Conference on Intelligent User Interfaces (IUI)*, pages 171–177, New York, NY, 2006. ACM.

[12] D. Feng, E. Shaw, J. Kim, and E. Hovy. Learning to detect conversation focus of threaded discussions. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 208–215, Morristown, NJ, 2006. Association for Computational Linguistics.

[13] Z. Gyöngyi, G. Koutrika, J. Pedersen, and H. Garcia-Molina. Questioning Yahoo! Answers. In *Proceedings of the First Workshop on Question Answering on the Web*, 2008.

[14] M. Hu, E.-P. Lim, A. Sun, H. W. Lauw, and B.-Q. Vuong. On improving Wikipedia search using article quality. In *Proceedings of the 9th Annual ACM International Workshop on Web Information and Data Management (WIDM)*, pages 145–152, New York, NY, 2007. ACM.

[15] J. Huang, M. Zhou, and D. Yang. Extracting chatbot knowledge from online discussion forums. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 423–428, Jan. 2007.

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

[17] J. Jeon, W. B. Croft, and J. H. Lee. Finding similar questions in large question and answer archives. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management (CIKM)*, pages 84–90, New York, NY, 2005. ACM.

[18] J. Jeon, W. B. Croft, J. H. Lee, and S. Park. A framework to predict the quality of answers with non-textual features. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 228–235, New York, NY, 2006. ACM.

[19] V. Jijkoun and M. de Rijke. Retrieving answers from frequently asked questions pages on the web. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management (CIKM)*, pages 76–83, New York, NY, 2005. ACM.

[20] P. Jurczyk and E. Agichtein. Discovering authorities in question answer communities by using link analysis. In *Proceedings of the sixteenth ACM Conference on Information and Knowledge Management (CIKM)*, pages 919–922, New York, NY, 2007. ACM.

[21] J. Kim, G. Chern, D. Feng, E. Shaw, and E. Hovy. Mining and assessing discussions on the web through speech act analysis. In *Proceedings of the Workshop on Web Content Mining with Human Language Technologies at the 5th International Semantic Web Conference*, 2006.

[22] J. Kim, E. Shaw, D. Feng, C. Beal, and E. Hovy. Modeling and assessing student activities in on-line discussions. In *Proceedings of the Workshop on Educational Data Mining at AAAI*, 2006.

[23] C.-J. Lin and C.-H. Cho. Question pre-processing in a QA system on internet discussion groups. In *Proceedings of the Workshop on Task-Focused Summarization and Question Answering*, 2006.

[24] Y. Liu, S. Li, Y. Cao, C.-Y. Lin, D. Han, and Y. Yu. Understanding and summarizing answers in community-based question answering services. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008)*, pages 497–504, Manchester, UK, August 2008.

[25] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M.-C. Hsu. PrefixSpan: Mining sequential patterns efficiently by prefix-projected pattern growth. In *Proceedings of the 17th International Conference on Data Engineering (ICDE)*, pages 215–224, Los Alamitos, CA, 2001. IEEE Computer Society.

[26] S. Riezler, A. Vasserman, I. Tsochantaridis, V. Mittal, and Y. Liu. Statistical machine translation for query expansion in answer retrieval. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, 2007.

[27] L. Shrestha and K. McKeown. Detection of question-answer pairs in email conversations. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING)*, page 889, Morristown, NJ, 2004. Association for Computational Linguistics.

[28] Y.-I. Song, C.-Y. Lin, Y. Cao, and H.-C. Rim. Question utility: A novel static ranking of question search. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*, July 2008.

[29] M. Surdeanu, M. Ciaramita, and H. Zaragoza. Learning to rank answers on large online qa collections. In *46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, 2008.

[30] K. Toutanova and C. D. Manning. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 63–70. Association for Computational Linguistics, 2000.

[31] E. M. Voorhees. The TREC question answering track. *Nat. Lang. Eng.*, 7(4):361–378, 2001.

[32] X. Xue, J. Jeon, and W. B. Croft. Retrieval models for question and answer archives. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 475–482, 2008.

[33] J. Zhang, M. S. Ackerman, and L. Adamic. Expertise networks in online communities: structure and algorithms. In *Proceedings of the 16th International Conference on World Wide Web (WWW)*, pages 221–230, 2007.

[34] L. Zhou and E. Hovy. Digesting virtual "geek" culture: the summarization of technical internet relay chats. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 298–305, 2005.