Exploiting Salient Patterns for Question Detection and Question Retrieval in Community-based Question Answering

Kai Wang
Department of Computer Science
School of Computing
National University of Singapore
kwang@comp.nus.edu.sg

Tat-Seng Chua
Department of Computer Science
School of Computing
National University of Singapore
chuats@comp.nus.edu.sg

Abstract

Question detection serves great purposes in the cQA question retrieval task. While detecting questions in standard language data corpus is relatively easy, it becomes a great challenge for online content. Online questions are usually long and informal, and standard features such as question mark or 5W1H words are likely to be absent. In this paper, we explore question characteristics in cQA services, and propose an automated approach to detect question sentences based on lexical and syntactic features. Our model is capable of handling informal online languages. The empirical evaluation results further demonstrate that our model significantly outperforms traditional methods in detecting online question sentences, and it considerably boosts the question retrieval performance in cQA.

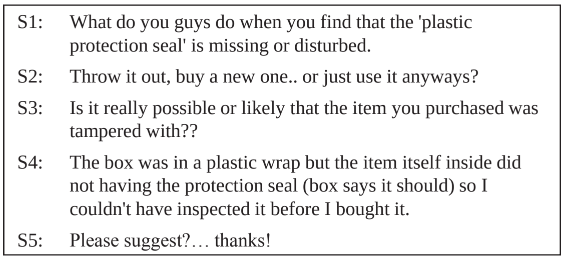
1 Introduction

Community-based Question Answering services (cQA) such as Yahoo! Answers have emerged as popular means of information exchange on the web. They not only connect a network of people to freely ask and answer questions, but also allow information seekers to search for relevant historical questions in the cQA archive (Agichtein et al., 2008; Xue et al., 2008; Wang et al., 2009).

Many research works have been proposed to find similar questions in cQA. The state-of-the-art retrieval models include the vector space model (Duan et al., 2008), language model (Duan et al., 2008; Jeon et al., 2005), Okapi model (Jeon et al., 2005), translation model (Jeon et al., 2005; Riezler et al., 2007; Xue et al., 2008), and syntactic tree matching model(Wang et al., 2009). Although experimental studies in these works show that the proposed models are capable of improving question retrieval, they did not give clear explanation on which portion of the question that the user query is actually matched against. A question thread from cQA usually comprises several sub-questions conveying different information needs, and it is highly desirable to identify individual sub-questions and match each of them to the user query. Getting sub-questions clearly identified not only helps the retrieval system to match user query to the most desirable content but also improves the retrieval efficiency.

However, the detection of sub-question is nontrivial. Question sentences in cQA are usually mixed with various description sentences, and they usually employ informal languages, where standard features such as question mark or utterance are likely to be absent. As such, simple heuristics using question mark or 5W1H words (*who, what, where, why, how*) may become inadequate. The demand of special techniques in detecting question sentences online arises due to three particular reasons. First, the question mark could be missing at the end of a question[1], or might be used in cases other than questions such as "*Really bad toothache?*". Second, some questions such as "*I'd like to know the expense of removing wisdom teeth*" are expressed in a declarative form, which neither contains 5W1H words nor is neccessarily ended with "?". Third, some question-like sentences do not carry any actual information need, such as "*Please help me?*". Figure 1 illustrates an example of a question thread

[1] It is reported (Cong et al., 2008) that 30% of online questions do not end with question marks.



S1: What do you guys do when you find that the 'plastic protection seal' is missing or disturbed.

S2: Throw it out, buy a new one.. or just use it anyways?

S3: Is it really possible or likely that the item you purchased was tampered with??

S4: The box was in a plastic wrap but the item itself inside did not having the protection seal (box says it should) so I couldn't have inspected it before I bought it.

S5: Please suggest?… thanks!

Figure 1: An example of a question thread extracted from Yahoo! Answers

from Yahoo! Answers, where sub-questions S1 and S2 are posted in non-standard forms, and S5 is merely a question-like simple sentence. To the best of our knowledge, none of the existing question retrieval systems are equipped with a comprehensive question detector module to handle various question forms online, and limited effort has been devoted to this direction.

In this paper, we extensively explore characteristics of questions in cQA, and propose a fully automated approach to detecting question sentences. In particular, we complement lexical patterns with syntactic patterns, and use them as features to train a classification model that is capable of handling informal online languages. To save human annotations, we further propose to employ one-class SVM algorithm for model learning, in which only positive examples are used as opposed to requiring both positive and negative examples.

The rest of the paper is organized as follows: Section 2 presents the lexical and syntactic patterns as used for question detection. Section 3 describes the learning algorithm for the classification model. Section 4 shows our experimental results. Section 5 reviews some related work and Section 6 concludes this paper.

2 Pattern Mining for Question Detection

As has been discussed, human generated content on the Web are usually not well formatted, and naive methods such as the use of question mark and 5W1H words are not adequate to correctly detect or capture all online questions. Methods based on hand-crafted rules also fail to cope with various question forms as randomly appeared on the Web. To overcome the shortcomings of these traditional methods, we propose to extract a set of salient patterns from online questions and use them as features to detect question sentences.

In this study, we mainly focus on two kinds of patterns – *sequential pattern* at the lexical level and *syntactic shallow pattern* at the syntactic level. Sequential patterns have been well discussed in many literature, including the identification of comparative sentences (Jindal and Liu, 2006), the detection of erroneous sentences (Sun et al., 2007) and question sentences (Cong et al., 2008) etc. However, works on syntactic patterns have only been partially explored (Zaki and Aggarwal, 2003; Sun et al., 2007; Wang et al., 2009). Grounded on these previous works, we next explain our mining approach of the sequential and syntactic shallow patterns.

2.1 Sequential Pattern Mining

Sequential Pattern is also referred to as *Labeled Sequential Pattern (LSP)* in the literature. It is in the form of $S \Rightarrow C$, where S is a sequence $\{t_1, \ldots, t_n\}$, and C is the class label that the sequence S is classified to. In the problem of question detection, a sequence is defined to be a series of tokens from questions, and the class labels are $\{Q, NQ\}$, which stand for question and non-question respectively.

The purpose of sequential pattern mining is to extract a set of frequent subsequence of words that are indicative of questions. For example, the word subsequence "*anyone know what* … *to*" could be a good indication to characterize the question sentence "*anyone know what I can do to make me less tired.*". Note that the mined sequential tokens need not to be contiguous as appeared in the original text.

There is a handful of algorithms available for frequent subsequence extraction. Pei et al. (2001) observed that all occurrences of a frequent pattern can be classified into groups (approximated pattern) and proposed a Prefixspan algorithm. The Prefixspan algorithm quickly finds out all relative frequent subsequences by a pattern growth method, and determines the approximated patterns from those subsequences. We adopt this algorithm in our work due to its high reported efficiency. We impose the following additional constraints for better control over the significance of the mined patterns:

1. Maximum Pattern Length: It limits the maximum number of tokens in a mined sequence.
2. Maximum Token Distance: The two adjacent tokens t_n and t_{n+1} in the pattern need to be within a threshold window in the original text.
3. Minimum Support: The minimum percentage of sentences in Q containing the pattern p.
4. Minimum Confidence: The probability of a pattern $p \Rightarrow Q$ being true in the whole database.

To overcome the word sparseness problem, we generalize each sentence by applying the Part-of-Speech (POS) tags to all tokens except some indicative keywords such as 5W1H words, modal words, stopwords etc. For instance, the question sentence *"How can I quickly tell if my wisdom teeth are coming"* is converted to *"How can I RB VBP if my NN NNS VBP VBG"*, on top of which the pattern mining is conducted. To further capture online language patterns, we mine a set of frequent tokens that are unique to cQA such as *"any1"*, *"im"* and *"whats"*, and keep them from being generalized. The reason to hold back this set of tokens is twofold. First, conventional POS taggers are trained from standard English corpus, and they could mis-tag these non-standard words. Second, the special online tokens are analogue to standard stopwords, and having them properly excluded could help reflect the online users' textual questioning patterns.

It is expected that the converted patterns preserve the most representative features of online questions. Each discovered pattern makes up a binary feature for the classification model that we will introduce in Section 3.

2.2 Syntactic Shallow Pattern Mining

The sequential patterns represent features at the lexical level, but we found that lexical patterns might not always be adequate to categorize questions. For example, the pattern {*when, do*} could presume the non-question "*Levator scapulae is used when you do the traps workout*" to be a question, whereas the question "*know someone with an eating disorder?*" could be overlooked due to the lack of indicative lexical patterns.

These limitations, however, could be alleviated by syntactic features. The syntactic pattern *(SBAR(WHADVP(WRB))(S(NP)(VP)))* extracted from the former example has the order of *NP* and *VP* being switched, which could indicate the sentence to be a non-question, whereas the pattern *(VP(VB)(NP(NP)(PP)))* may be evidence that the latter example is indeed a question, because this pattern is commonly witnessed in the archived questions. Figure 2 shows an example that two questions bear very different wordings but share the same questioning pattern *(S(NP(NN))(VP(VPB)(NP)))* at the syntactic level. In view of the above, we argue that patterns at the syntactic level could complement lexical patterns in identifying question sentences.

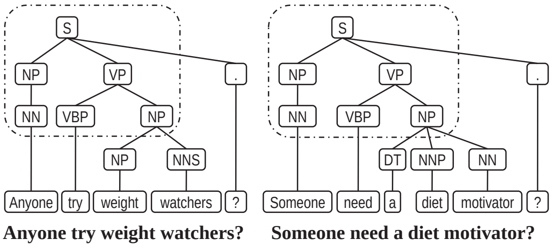


Figure 2: An example of common syntactic patterns observed in two different question sentences

To our knowledge, the mining of salient patterns at the syntactic level was limited to a few tasks. Zaki and Aggarwal (2003) employed tree patterns to classify XML data, Sun et al. (2007) extracted all frequent sub-tree structures for erroneous sentences detection, and Wang et al. (2009) decomposed the parsing tree into fragments and used them to match similar questions. Our work differs from these previous works in that: (1) we also utilize syntactic patterns for the question detection; and (2) we do not blindly extract all possible sub-tree structures, but focus only on certain portions of the parsing tree for better pattern representation and extraction efficiency.

Given a syntactic tree T, we define *syntactic pattern* as a part of sub-structures of T such that the production rule for each non-leaf node in the patterns is intact. For example, the pattern *(S(NP(NN))(VP(VPB)(NP)))* in Figure 2 is considered to be a valid syntactic pattern, whereas *(S(NP(NN))(VP(VPB)))* is not, since the production rule *VP→VPB·NP* is not strictly complied.

We take the following measures to mine salient syntactic patterns: First, we limit the depth of each syntactic pattern to be within a certain range.

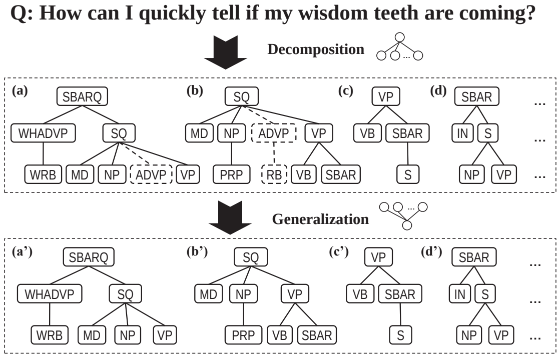


Figure 3: Illustration of syntactic pattern extraction and generalization process

It is believed that the syntax structure will become too specific if it is extended to a deeper level or too general if the depth is too shallow, neither of which produces good representative patterns. We therefore set the depth D of each syntactic pattern to be within a reasonable range ($2 \leq D \leq 4$). Second, we prune away all leaf nodes as well as the production rules at the POS tag level. We believe that nodes at the bottom levels do not carry much useful structural information favored by question detector. For example, the simple grammar rule *NP→DT·NN* does not give any insight to useful question structures. Third, we relax the definition of syntactic pattern by allowing the removal of some nodes denoting modifiers, preposition phrases, conjunctions etc. The reason is that these nodes are not essential in representing the syntactic patterns and are better excluded for generalization purpose. Figure 3 gives an illustration of the process for pattern extraction and generalization. In this example, several syntactic patterns are generated from the question sentence "*How can I quickly tell if my wisdom teeth are coming?*", and the tree patterns (a) and (b) are generalized into (a') and (b'), in which the redundant branch *(ADVP(RB))* that represents the adverb "*quickly*" is detached.

Contents on the Web are prone to noise, and most off-the-shelf parsers are not well-trained to parse online questions. For example, the parsing tree of the question "*whats the matter with it?*" will be very different from that of the question "*what is the matter with it?*". It would certainly be nice to know that "*whats*" is a widely used short form of the phrase "*what is*" on the Web, but we are lack of this kind of thesaurus. Nevertheless, we argue that the parsing errors would not hurt the question detector performance much as long as the mining database is large enough. The reason is that if certain irregular forms frequently occur on the Web, there will be statistical evidences that the syntactic patterns derived from it, though not desired, will commonly occur as well. In other words, we take the wrong patterns and utilize them to detect questions in the irregular forms. Our approach differs from other systems in that we do not intentionally try to rectify the grammatical errors, but leave the errors as they are and use the statistical based approach to capture those informal patterns.

The pattern extraction process is outlined in Algorithm 1. The overall mining strategy is analogous to the mining of sequential patterns, where *support* and *confidence* measures are taken into account to control the significance of the mined patterns. All mined syntactic patterns together with the lexical patterns will be used as features for learning the classification model.

Algorithm 1 *ExtractPattern(S, D)*

Input: A set of syntactic trees for sentences (S); the depth range (D)
Output: A set of sub-tree patterns extracted from S

```
$Patterns$ = $\{\}$
for all Syntactic tree $T \in S$ do
    Nodes ← Top-down level order traversal of $T$
    for all node $n \in Nodes$ do
        Extract subtree $p$ rooted under node $n$, with depth
        within the range $D$
        $p$ ← generalize($p$)
        $Patterns$.add($p$)
    end for
end for
return $Patterns$
```

3 Learning the Classification Model

Although Conditional Random Fields (CRF) is good sequential learning algorithm and has been used in other related work (Cong et al., 2008), here we select Support Vector Machines (SVM) as an alternative learner. The reason is that our task not only deals with sequential patterns but also involves syntactic patterns that possess no sequential criteria. Additionally, SVM has been widely shown to provide superior results compared to other classifiers.

The input to a SVM binary classifier normally consists of both positive and negative examples. While it is easy to discover certain patterns from questions, it is unnatural to identify characteristics for non-questions, as they usually do not share such common lexical and syntactic patterns. The lack of good negative examples leads traditional SVM to perform poorly. To adapt the imbalanced input data, we proposed to employ a one-class SVM method (Manevitz and Yousef, 2002) for learning. The basic idea of one-class SVM is to transform features from only positive examples via a kernel to a hyper-plane and treats the origin as the only member of the second class. It uses relaxation parameters to separate the positive examples from the origin, and finally applies the standard two-class SVM techniques to learn a decision boundary. As a result, anything outside the boundary are considered to be outliers (*i.e.* non-questions in this problem).

More formally, given n training samples $x_1, \ldots, x_n$ of one class, the hyperplane separating them from the origin is constructed by solving

$$\min \frac{1}{2}\|w\|^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i - \rho \qquad (1)$$

subject to: $w \cdot \Phi(x_i) \geq \rho - \xi_i$, where Φ is a kernel function, ξ_i is the slack variable, and ν is the parameter controlling the upper bound percentage of outliers. If w and ρ solve this problem, the decision function $f(x) = sign(w \cdot \Phi(x) - \rho)$ will be positive for most examples x_i in the training set.

Supervised learning methods usually require training data to be manually annotated. To save labeling efforts, we take a shortcut by treating all sentences ending with question marks as an initial positive examples. This assumption is acceptable, as Cong et al. (2008) reported that the rule-based method using only question mark achieves a very high precision of over 97% in detecting questions. It in turn indicates that questions ending with "?" are highly reliable to be real questions.

However, the initial training data still contain many sentences ending with "?" but are not true questions. These possible outliers will shift the decision boundary away from the optimal one, and we need to remove them from the training dataset for better classification. Many preprocessing strategies are available for training data refinement, including bootstrapping, condensing, and editing etc. In this work, we employ a SVM-based data editing and classification method proposed by Song et al. (2008), which iteratively sets a small value to the parameter ν of the one-class SVM so as to continuously refine the decision boundary. The algorithm could be better visualized with Figure 4. In each iteration, a new decision boundary will be determined based on the existing set of data points, and a portion of possible outliers will be removed from the training set. It is expected that the learned hyperplane will eventually be very close to the optimal one.

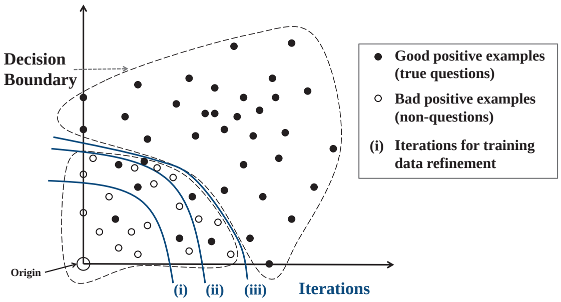

Decision Boundary
Origin
(i)
(ii)
(iii)
Iterations
● Good positive examples (true questions)
○ Bad positive examples (non-questions)
(i) Iterations for training data refinement

Figure 4: Illustration of one-class SVM classification with training data refinement (conceptual only). Three iterations (i) (ii) (iii) are presented.

We use the freely available software LIBSVM[2] to conduct the one-class SVM training and testing. A linear kernel is used, as it is shown to be superior in our experiments. In each refinement iteration, the parameter ν is conservatively set to 0.02. The number of iteration is dynamically determined according to the algorithm depicted in (Song et al., 2008). Other parameters are all set to default. The refined decision boundary from the training dataset will be applied to classify questions from non-questions. The question detector model learned will serve as a component for the cQA question retrieval system in our experiments.

4 Experiments

In this section, we present empirical evaluation results to assess the effectiveness of our question detection model. In particular, we first examine the effects of the number of patterns on question detection performance. We further conduct experiments to show that our question de-

[2] Available at: http://www.csie.ntu.edu.tw/˜cjlin/libsvm

# of Lexical Patterns		Confidence					# of Syntactic Patterns		Confidence				
		60%	65%	70%	75%	80%			60%	65%	70%	75%	80%
Support	0.40%	1685	1639	1609	1585	1545	Support	0.03%	916	758	638	530	453
	0.45%	1375	1338	1314	1294	1277		0.04%	707	580	488	402	341
	0.50%	1184	1151	1130	1113	1110		0.05%	546	450	375	308	261
	0.55%	1037	1007	989	975	964		0.06%	468	379	314	260	218

Table 1: Number of lexical and syntactic patterns mined over different *support* and *confidence* values

Lexical Patterns		Confidence									Syntactic Patterns		Confidence								
		65%			70%			75%					60%			65%			70%		
		P	R	F_1	P	R	F_1	P	R	F_1			P	R	F_1	P	R	F_1	P	R	F_1
Support	0.40%	85.7	90.7	88.1	86.9	88.6	87.7	87.8	86.6	87.2	Support	0.03%	80.4	83.3	81.9	85.1	77.5	81.1	90.7	70.2	79.1
	0.45%	86.6	90.2	88.4	88.9	88.5	88.7	89.6	86.7	88.2		0.04%	79.0	86.1	82.4	90.1	78.2	83.7	90.8	70.8	79.6
	0.50%	88.5	91.6	88.4	86.4	89.0	87.7	86.2	87.9	87.0		0.05%	80.3	82.5	81.4	88.8	78.4	83.3	89.9	69.0	78.1
	0.55%	86.5	89.9	88.1	88.1	87.5	87.8	88.0	89.2	88.6		0.06%	83.0	83.2	83.1	88.5	77.2	82.4	86.7	75.8	80.9

Table 2: Question detection performance over different sets of lexical patterns and syntactic patterns

tection model combining both lexical and syntactic features outperforms traditional rule-based or lexical-based methods. We finally demonstrate that our question detection model gives additional performance boosting to question matching.

4.1 Performance Variation over Different Pattern Sets

The performance of the question detection model can be sensitive to the number of features used for learning. To find the optimal number of features used for model training, we examine the performance variation over different amount of lexical and syntactic patterns undertaken for training.

Dataset: We collected a total of around 800k question threads from Yahoo! Answers Healthcare domain. From the collected data, we generated the following three datasets:

- Pattern Mining Set: Comprising around 350k sentences from 60k question threads, where those ending with "?" are treated as questions and others as non-questions.
- Training Set: Positive examples comprising around 130k sentences ending with "?" from another 60k question threads for the one-class SVM learning algorithm.
- Testing Set: Two annotators are asked to tag randomly picked sentences from the remaining set. A total of 2,004 question sentences and 2,039 non-question sentences are annotated.

Methods & Results: We use different combinations of *support* and *confidence* values to generate different set of patterns. The *support* value ranges from 0.40% to 0.55% for lexical patterns with a step size of 0.05%, and ranges from 0.03% to 0.06% for syntactic patterns with a step size of 0.01%. The *confidence* value for both patterns ranges from 60% to 80% with a step size of 5%. These value ranges are empirically determined. Table 1 presents the number of lexical and syntactic patterns mined against different *support* and *confidence* value combinations.

For each set of lexical or syntactic patterns mined, we use them as features for model training. We convert the training sentences into a set of feature vectors and employ the one-class SVM algorithm to train a classifier. The classifier will then be applied to predict the question sentences in the testing set. To evaluate each question detection model, we employ Precision (P), Recall (R), and F_1 as performance metrics, and Table 2 presents the results[3].

We observe from Table 2 that given a fixed support level, the precision generally increases with the confidence level for both lexical and syntactic patterns, but the recall drops. The lexical feature set comprising 1,314 sequential patterns as generated with {*sup=0.45%*, *conf=70%*} gives the best F_1 score of 88.7%, and the syntactic feature set comprising 580 syntactic patterns generated from {*sup=0.04%*, *conf=65%*} gives the best F_1 score of 83.7%. It is noted that the sequential patterns give relatively high recall while the syntactic patterns give relatively high precision. Our reading is that the sequential patterns are capable of capturing most questions, but it may also give wrong predictions to non-questions such as "*Lev-*

[3] The results for certain *confidence* levels are not very promising and are not shown in the table due to lack of space.

ator scapulae is used when you do the traps workout” that bears the sequential pattern $\{$*when, do*$\}$. On the other hand, the syntactic patterns could give reliable predictions, but its coverage could suffer due to the limited number of syntactic patterns. We conjecture that a combination of both features could further improve the performance.

4.2 Performance Comparison with Traditional Question Detection Methods

We next conduct experiments to compare the performance of our question detection model to traditional rule-based or lexical-based methods.

Methods & Results: We set up five different systems for meaningful comparisons:

1. 5W1H (baseline1): a rule-based method using 5W1H to determine a question sentence.
2. Question Mark (baseline2): a method using the question mark “?” to judge a question.
3. SeqPattern: Using only the set of 1,314 sequential patterns as features.
4. SynPattern: Using only the set of 580 syntactic patterns as features.
5. SeqPattern+SynPattern: Merging both lexical and syntactic patterns and use them as a set of features for question detection.

We again employ Precision (P), Recall (R), and F_1 as performance metrics to evaluate each question detection system, and tabulate the comparison results in Table 3. From the Table, we observe that 5W1H performs poorly in both precision and recall, and question mark based method gives relatively low recall although the precision is the highest amongst all the methods evaluated. This is in line with the results as observed in (Cong et al., 2008). SeqPattern outperforms the two baseline systems in both R and F_1 scores, and its combination with SynPattern augments the performance in both precision and recall by a lot. It also achieves statistically significant improved results (t-test, p-value$<$0.05) as compared to other four systems. These results are consistent with our intuition that syntactic patterns can leverage sequential patterns in improving the question detection performance.

It is noted that SeqPattern+SynPattern exhibits the highest recall (R) amongst all the systems. The significance test further suggests that many question sentences miss-detected by 5W1H or Question Mark method could be properly captured by our model. This improvement is meaningful, as the question coverage is also an important factor in the cQA question retrieval task, where high recall implies that more similar questions could be matched and returned, hence improving the question retrieval performance.

System Combination	$P(\%)$	$R(\%)$	$F_1(\%)$
(1) 5W1H	75.37	49.50	59.76
(2) Question Mark	**94.12**	77.50	85.00
(3) SeqPattern	88.92	88.47	88.69
(4) SynPattern	90.06	78.19	83.71
(5) SeqPattern+SynPattern	92.11	**89.67**	**90.87**

Table 3: Performance comparisons for question detection on different system combinations

4.3 Performance Evaluation on Question Retrieval with Question Detection Model

To further demonstrate that our question detection model can improve question retrieval, we incorporate it into different question retrieval systems.

Methods: We select a simple bag-of-word (BoW) system retrieving questions at the lexical level, and a syntactic tree matching (STM) model matching questions at the syntactic level (Wang et al., 2009) as two baselines. For each baseline, we further set up two different combinations:

- Baseline+QM: Using question mark to detect question sentences, and perform question retrieval on top of the detected questions.
- Baseline+QD: Using our proposed model to detect question sentences, and perform question retrieval on top of the detected questions.

This gives rise to additional 4 different system combinations for comparison.

Dataset: We divide the dataset from Yahoo! Answers into a question repository set (750k) and a test set (50k). For the baseline systems, all the repository sentences containing both questions and non-questions are indexed, whereas for systems equipped with QM or QD, only the detected question sentences are indexed for retrieval. We randomly select 250 single-sentence questions from the test set as queries, and for each query, the retrieval system will return a list of top 10 question matches. We combine the retrieved results from different systems and ask two annotators to label each result to be either “relevant” or “irrel-

System Combination	BoW	BoW +QM	BoW +QD	STM	STM +QM	STM +QD
MAP (%)	58.07	59.89	60.68	66.53	68.41	69.85
% improvement of MAP over:						
Baseline	N.A.	+3.13	+4.49	N.A.	+2.83	+4.99
Baseline+QM	N.A.	N.A.	+1.32	N.A.	N.A.	+2.10
P@1 (%)	59.81	61.21	63.55	63.08	64.02	65.42

Table 4: Question retrieval performance on different system combinations measured by MAP and P@1 (Baseline is either BoW or STM)

evant” without telling them which system the result is generated from. By eliminating some query questions that have no relevant matches, the final testing set contains 214 query questions.

Metrics & Results: We evaluate the question retrieval performance using two metrics: Mean Average Precision (MAP) and Top One Precision (P@1). The results are presented in Table 4.

We can see from Table 4 that STM outperforms BoW. Applying QM or QD over BoW and STM boosts the system performance in terms of both MAP and P@1. They also achieve statistical significance as judged by paired t-test (p-value$<$0.05). More specifically, the MAP on QM coupled systems improves by 3.13% and 2.83% respectively over BoW and STM. This is evidence that having question sentences clearly identified could help to retrieve relevant questions more precisely, as without question detection, the user query is likely to be matched to irrelevant description sentences. Our question detection model (QD) further improves the MAP by 1.32% and 2.1% respectively over BoW+QM and STM+QM, and it also yields better top one precision by correctly retrieving questions at the first position on 136 and 140 questions respectively, out of a total of 214 questions. These improvements are in line with our expectation that our model incorporating salient features at both the lexical and syntactic levels is comprehensive enough to capture various forms of questions online, and hence improve the performance of question matching.

5 Related Work

Research on detecting question sentences can generally be classified into two categories. The first category simply employs rule-based methods such as question mark, 5W1H words, or hand-crafted regular expressions to detect questions. As discussed, these conventional methods are not adequate to cope with online questions.

The second category uses machine learning approaches to detect question sentences. Shrestha and McKeown (2004) proposed a supervised rule induction method to detect interrogative questions in email conversations based on part-of-speech features. Yeh and Yuan (2003) used a statistical approach to extract a set of question-related words and derived some syntax and semantic rules to detect mandarin question sentences. Cong et al. (2008) extracted labeled sequential patterns and used them as features to learn a classifier for question detection in online forums.

Question pattern mining is also closely related to the learning of answer patterns. Work on answer patterns includes the web based pattern mining (Zhang and Lee, 2002; Du et al., 2005) and a combination of syntactic and semantic elements (Soubbotin and Soubbotin, 2002) etc.

In contrast to previous work, we do not only focus on standard language corpus, but extensively explore characteristics of online questions. Our approach exploits salient question patterns at both the lexical and syntactic levels for question detection. In particular, we employ the one-class SVM algorithm such that the learning process is weakly supervised and no human annotation is involved.

6 Conclusion

This paper proposed a new approach to detecting question sentences in cQA. We mined both lexical and syntactic question patterns, and used them as features to build classification models. The mining and leaning process is fully automated and requires no human intervention. Empirical evaluation on the cQA archive demonstrated the effectiveness of our model as well as its usefulness in improving question retrieval performance.

We are still investigating other features that are helpful to detect questions. One promising direction for future work is to also employ lexical and syntactic patterns to other related areas such as question type classification etc. It is also interesting to employ a hybrid of CRF and SVM learning methods to boost the accuracy and scalability of the classifier.

References

Agichtein, Eugene, Carlos Castillo, Debora Donato, Aristides Gionis, and Gilad Mishne. 2008. Finding high-quality content in social media. In *WSDM*, pages 183–194.

Cong, Gao, Long Wang, Chin-Yew Lin, Young-In Song, and Yueheng Sun. 2008. Finding question-answer pairs from online forums. In *SIGIR*, pages 467–474.

Du, Yongping, Helen Meng, Xuanjing Huang, and Lide Wu. 2005. The use of metadata, web-derived answer patterns and passage context to improve reading comprehension performance. In *HLT*, pages 604–611.

Duan, Huizhong, Yunbo Cao, Chin-Yew Lin, and Yong Yu. 2008. Searching questions by identifying question topic and question focus. In *HLT-ACL*, pages 156–164.

Jeon, Jiwoon, W. Bruce Croft, and Joon Ho Lee. 2005. Finding similar questions in large question and answer archives. In *CIKM*, pages 84–90.

Jindal, Nitin and Bing Liu. 2006. Identifying comparative sentences in text documents. In *SIGIR*, pages 244–251.

Manevitz, Larry M. and Malik Yousef. 2002. One-class svms for document classification. *J. Mach. Learn. Res.*, 2:139–154.

Pei, Jian, Jiawei Han, Behzad Mortazavi-asl, Helen Pinto, Qiming Chen, Umeshwar Dayal, and Mei chun Hsu. 2001. Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth. In *ICDE*, pages 215–224.

Riezler, Stefan, Alexander Vasserman, Ioannis Tsochantaridis, Vibhu Mittal, and Yi Liu. 2007. Statistical machine translation for query expansion in answer retrieval. In *ACL*, pages 464–471.

Shrestha, Lokesh and Kathleen McKeown. 2004. Detection of question-answer pairs in email conversations. In *COLING*, page 889.

Song, Xiaomu, Guoliang Fan, and M. Rao. 2008. Svm-based data editing for enhanced one-class classification of remotely sensed imagery. *Geoscience and Remote Sensing Letters, IEEE*, 5(2):189–193.

Soubbotin, Martin M. and Sergei M. Soubbotin. 2002. Use of patterns for detection of likely answer strings: A systematic approach. In *TREC*.

Sun, Guihua, Gao Cong, Xiaohua Liu, Chin-Yew Lin, and Ming Zhou. 2007. Mining sequential patterns and tree patterns to detect erroneous sentences. In *AAAI*, pages 925–930.

Wang, Kai, Zhaoyan Ming, and Tat-Seng Chua. 2009. A syntactic tree matching approach to finding similar questions in community-based qa services. In *SIGIR*, pages 187–194.

Xue, Xiaobing, Jiwoon Jeon, and W. Bruce Croft. 2008. Retrieval models for question and answer archives. In *SIGIR*, pages 475–482.

Yeh, Ping-Jer and Shyan-Ming Yuan. 2003. Mandarin question sentence detection: A preliminary study. In *EPIA*, pages 466–478.

Zaki, Mohammed J. and Charu C. Aggarwal. 2003. Xrules: an effective structural classifier for xml data. In *KDD*, pages 316–325.

Zhang, Dell and Wee Sun Lee. 2002. Web based pattern mining and matching approach to question answering. In *TREC*.