What Can Be Learned from Previously Answered Questions? A Corpus-Based Approach to Question Answering

Marcin Skowron and Kenji Araki

Graduate School of Engineering, Hokkaido University,
Kita-ku Kita 13-jo Nishi 8-chome, 060-8628 Sapporo, Japan

Abstract. We present corpus-based approach to question answering, which enables our system to classify a question category, generate a query and verify answer candidates. The system uses the Internet to find information required to provide an answer. The role of the corpus-based modules is to support the system with knowledge that provides the means to effectively use previously answered questions as the system experience base. We present the Query Pattern Generation method, which demonstrates that the system can automatically acquire knowledge on how to optimally generate a query for a given question category and question syntax. The corpus-based answer candidate verification is an effective tool to exclude answer candidates that do not belong to a question category, and is able to provide a proper length for the answer.

1 Introduction

In recent years the amount of text available online has been rapidly increasing; in 2003, there were 3 billion web pages and their numbers continue to grow. While the amount of coverage and information increases, it has also become more difficult for the average Internet user to access specific and reliable information. We consider that for many users the most natural approach to such a task would be to ask a question in natural language form such as "Who discovered Pluto?" or "What company is the largest Japanese ship builder?". The result ought to be an exact answer, resembling as close as possible, those given by human beings. However, the current open domain Internet based Question Answering (QA) Systems[1] often fail to meet this requirement.

The analysis of the working process of the current QA Systems reveals the problem areas inherent in them, among which we find a query generation and answer candidates extraction processes to be the most significant ones. Sheer size of the textual resources accessible on the Internet makes it unfeasible to apply on a larger scale the complicated and time-consuming Natural Language Processing (NLP) techniques. As a result, Internet based QA systems

[1] From now on we refer to open domain, Internet based Question Answering Systems simply as a QA Systems

are bound to use fast and relatively simple methods to deliver an answer in a timely manner. At the same time, these systems need to guaranty high performance, by providing users with the correct information. In this paper we will introduce methods that address the flaws of the current QA Systems, and provide higher precision in a query generation and answer candidate extraction process. Such approaches contribute also to a reduction in the overall processing time.

Our system differs from most QA Systems in its extensive use of the modules built based on the corpus, which consist of questions with correct answers and question category classification. These modules are used to assign a question category, generate a query and verify answer candidates. Our system uses the Internet to find information required to provide an answer. The role of the corpus-based modules is to support the system with knowledge that provides the means to effectively use previously answered questions as the system experience base. For question classification, we use Support Vector Machine (SVM) [3], which is known to work well for sparse, high dimensional problems, and the Category-Unique Pattern pre-classification method. Using this approach we achieved better results than those reported in previous literature. We present a Query Pattern Generation method, which for a given category and question syntax, forms a set of queries that return answers in a relatively undistorted set of snippets. The preliminary tests have demonstrated a significant improvement in the results when using this method of query generation, over the currently used one in the QA Systems. Answer candidate verification is performed by applying a set of regular expressions that are automatically generated using the surface, syntactic and semantic features of the answers, from the same question category. The length of the answer returned by our system is question-dependent, aiming only to include those words that are required.

2 Corpus-Based Methods for Question Answering

2.1 Question Classification

In order to provide an answer to a user's question from a large collection of texts, the system is required to impose some constraints on the scope of possible answers, such as question classification. In addition, in our system question category information is used in the query generation process. The system performs question classification using the Category-Unique Patterns (CUP), and models generated by Support Vector Machine (SVM) [6] and trained on the corpus consisting of 5,500 questions labeled manually by UIUC [8], with 6 coarse-grained and 50 fine-grained categories. Table 1 shows these question categories.

Out tests with the SVM demonstrated that when using only text surface features (a bag-of-words approach), the classification generates errors when words from a question does not appear in the feature space, and other

Table 1. The coarse and fine-grained question categories

Coarse	**Fine**
ABBR	abbreviation, expansion
DESC	definition, description, manner, reason
ENTY	animal, body, color, creation, currency, disease, event, food, instrument, language, letter, other, plant, product, religion, sport, substance, symbol, technique, term, vehicle, word
HUM	description, group, individual, title
LOC	city, country, mountain, other, state
NUM	code, count, date, distance, money, order, other, percent, period, speed, temperature, size, weight

words are not sufficient to assign a question to a correct category. This is a case in the question like: "What is BBC?" - where the word "BBC" is not found in the feature space, and the remaining words "What" and "is" appear in several categories, preventing the assignment of a correct category (ABBR:expansion). To address such problems, we add a surface feature of the words (letter capitalization) as a new entry to the feature space. For example, it assigns the same feature for nouns consist of only capitalized letters like words BBC or CPR, which is different from a feature assigned for word Tokyo (the first capitalized letter followed by the small letters).

Another method is required to correctly discriminate questions like "What is the Ohio state bird?" (ENTY:animal) and "What is Australia's national flower?" (ENTY:plant). Here, several words like: Ohio, state, bird, Australia, national, flower are highly undistinguishable, existing at the same time in a few categories (DESC:definition, description; ENTY:animal, plant; LOC:country, city, state; NUM:count, date). To enable correct assignment of a category for these kind of questions, additional semantic features were introduced, using 22 selected WordNet [9] hyponym categories like: length, body part, plant, substance, animal, time period, quantitative relation, etc. If found, these hyponyms are assigned for all common nouns found in a question and add as a new entry in the feature space. This approach has lead to the improvement of classification results, as presented in Table 2.

Further analysis of the training data revealed that some sentence patterns that frequently exist in one category, do not appear in others. These patterns were constructed using the fivegrams, fourgrams and trigrams of the initial question words, with common nouns substituted with the selected hyponyms category, where words formed only with capitalized letters were represented in the one symbolic form. If such a pattern is found to frequently exist in one category and do not appear in others, it is stored and used to discriminate a question category for the entries from the test data. These Category-Unique Patterns are used to pre-classify questions before the SVM classifier. Using this combined approach, our Question Classification module generated a better result than those reported by others [8,14], for the same training and test

data collection. Table 2 shows the accuracy of question classification for the fine-grained categories.

Table 2. The question classification accuracy for the fine-grained categories

	SVM [14]	**SVM(feat.)**	SNOW [8]	**CUP+SVM(feat.)**
First classification	80.2%	**82.7%**	84.2%	**86%**

2.2 Query Generation

Limitations of the Current Approach of Query Generation
One of the most fundamental problem of question answering is that of finding spaces where the distance between questions and sentences containing a correct answer is small and where the distance between questions and sentences containing incorrect answer is large [4]. In our opinion, in the frequently used query generation method, potentially useful information is often lost, thus reducing the likelihood of generating a reliable query and making the problem of finding a small distance between question and sentences that contain a correct answer, even more difficult. Current QA Systems often generate a query by removing the functional words (prepositions, determiners, pronouns, conjunctions and discourse particles) and other frequently used ones [1,5,13,15]. For example, for the question "What does CPR stand for?", the query (CPR stand) would be generated. Once submitted to a search engine, this query retrieves a distorted set of snippets (search result, a title and some textual string segments of the related web document) where the correct answer - "cardiopulmonary resuscitation" occurs relatively infrequently. However, for the same question a more reliable query ("CPR stands for"), can be formed to retrieve a less distorted set of snippets, where answer candidate occurs much more frequently. To generate such a query, the preposition (excluded in the current method), and knowledge on how to optimally combine question words to form an "exact phrase", is required. In the current approach, these means are not available. Moreover, the queries generated by the current QA systems do not provide any information on where to expect an answer candidate to appear, thus complicating the candidates extraction process. Such information can be associated with the latter query ("CPR stands for" <answer candidate>). Additional improvement of query reliability can be achieved by extending it with a word or other non-letter characters, which frequently connect a given query with a correct answer - Surface Text Pattern (STP) [12]. For the question "When was Queen Victoria born?", the query ("Queen Victoria was born on") can be generated by the addition of the preposition "on", which is frequently found with the answer, like in the phrase "[..]Queen Victoria was born on 24 May 1819 [..]".
Below we present a method that for a given question category and question

syntax, automatically generates a set of such queries, including their optional extension with an STP. Such queries additionally provide information on where to expect answer candidates to appear.

Query Pattern Generation Method

To reliably obtain an answer, the system needs to construct a query that returns it in a relatively undistorted set of snippets. A query that is too precise does not generate a sufficient number of answer candidates to allow a frequency-based selection of an answer or does not return any snippets at all. On the other hand, too general a query is likely to produce a distorted set of snippets, where a correct answer is rarely provided.

The idea of a Query Pattern Generation method [11] is to automatically find an optimal combination of question and non-question words, which when submitted to a search engine, retrieves as undistorted set of snippets as possible. To avoid the necessity to build and verify large numbers of permutations of words from a question, we use Question Syntactic Representation (QSR), which reduces computational and extensive Internet usage requirements, while ensuring that several potentially important question word combinations are being verified. The QSR is created using POS tags [2], which are later processed using the connection rules that allow linking specific combinations of tags such as: Determiner+Noun (/DT /NN = /NP) or Proper Noun+Proper Noun (/NNP /NNP = /NP). A QSR of "When was Queen Victoria born?" would thus be (/WBR/VBD1/NP1/VBN2/?). In the next step a set of transformation and reduction rules is applied. Transformation rules enable for example to change a verb tense. A question "When did French revolutionaries storm the Bastille?" would be transformed to "When/WRB French_revolutionaries/NP1 stormed/VBD1 the_Bastille/NP2 ?/?". The reduction rules exclude for example Wh-words and the question mark, which were found to be redundant in generating a reliable query. To be able to use an "exact match" in a query, a pair of quotation marks is added to the QP elements. For the above question, there are 5 QP elements formed (was/VBD1 Queen_Victoria/NP1 born/VBN2 " "). In the next step permutations of QP elements are generated, and the ones that do not form a valid query (e.g. does not have any noun phrase or the quotation marks do not embody any word), are excluded. For our example question the script generated 38 valid QP combinations like ("Queen Victoria was" born), ("Queen Victoria") or (Queen Victoria was born). These queries are then sent one by one to the search engine. If available, the first set of 100 snippets is collected.

The reliability of a given query is calculated as a number of answers (the multiple occurrences of an answer in one snippet is counted only once) to the number of accessed snippets (a number between 1-100). The QPs with the highest reliability level are selected and stored. For the example question "When was Queen Victoria born?" and the answer "1819", 24 QP that returned at least one correct answer were found. Table 3 presents the results

Table 3. Query Pattern Generation results

Position	Query Pattern	Reliability level for 100 snippets
1	"/NP1/VBD1/VBN2" "Queen Victoria was born"	50
2	"/NP1/VBD1"/VBN2 "Queen Victoria was" born	30
...		
8	/NP1/VBD1/VBN2 Queen Victoria was born	27

Table 4. Query Pattern Generation results for STP extended QP

Method	Query	Reliability level	No. of snippets
QPG	"Queen Victoria was born in"	80	61
Current	Queen Victoria was born	27	100
QPG	"BBC stands for"	11	100
Current	BBC is	2	100

for some of these QPs. We have found that the most reliable query found for this question (position 1) is approximately 85% better than one generated by a current QA System (position 8). Analysis of the results for the test set of 50 questions has revealed that all the queries generated using our method achieved a higher reliability level, compared to the current method. For the test set, the improvement rate varied depending on the question, between 12%-89%, compared to the best query generated using current method.

In the same process, using all snippets containing a correct answer, the Surface Text Patterns (STP) are extracted. The most frequent STP is joined to QP. Such an extended QP is verified using the method described above. If it is found to form a reliable query, it is added as an additional QP to be stored along with a particular question category and QSR, in the Query Pattern Repository. Table 4 presents the results of QPs extended with the discovered STP, for two questions - "When was Queen Victoria born?" (1819) and "What is BBC?" (British Broadcasting Corporation), compared to the queries generated using a current method.

Next, we explain the Query Pattern Generation process using the example question, "Who was the first man on the Moon?". First, the script generates a set of valid QP combinations and after verification, a QP "/NP1/IN1/NP2 /VBD1+" is found to form a reliable query, like in the phrase "[..]the first man on the Moon was Neil Armstrong[..]" (note: the plus sign in the QP shows the expected place of answer candidate occurrence). In this case there are no additional characters between the QP and an answer, so an STP is not created. However, for another QP "+/NP1/IN1/NP2", the script finds in the phrase "[..]Neil Armstrong, the first man on the Moon[..]" the STP ",". Thus, after a positive verification of the extended QP, the new record in the QP Repository is formed in the following manner: QCat (HUM:individual),

QSR (/WP/VBD1/NP1/IN1/NP2/?), QP (+/NP1/IN1/NP2), (+,/NP1/IN1 /NP2).

2.3 Answer Candidate Extraction and Verification

We find the answer candidate extraction process, after query generation, to be the second most significant problem area that exists in the QA Systems. In the current method, judgment of whether a sentence/paragraph is a valid answer candidate is based on the process of word matching between a question and the sentence/paragraph [1,15]. This may result in returning answers that do not have any semantic relation to a question. Furthermore, the set length of an answer candidate sentence/paragraph/snippet [1,5,7,10,15] often makes it impossible to provide the user with an answer in a human-like form. It also limits usability of such information for other systems that could use it in their working processes. Our system uses information on where an answer candidate is likely to appear (information included with every QP) to extract phrases in the one sentence window (we found that such a scope is suitable for the majority of the factual questions' answers, which usually consist of a word or a few words). However, it is not sufficient to reliably verify answer candidates and set a proper answer length. Our answer candidate verification module uses the information acquired from the corpus of previously answered questions. Using this resource, the system is equipped with knowledge about common elements, possible variations and other characteristic features of answers from the same question category. Here we present some examples of answers for a few question categories: NUM:date (1789; 5th November 1976; May 17-20, 2004), ENTY:animal (white tiger, giraffe, mice), ABBR:abbreviation (CNN, LASER, ASEAN), HUM:title (actor, US President, the former leader of the Trade Unions). Our method applied to extract this knowledge includes examining the surface, syntactic and semantic features of answers from a particular question category. Surface features include: number of words, letter capitalization, non-letter characters, etc. A syntactic feature is a POS tag assigned to a word, and a semantic feature is a hyponym category assigned for nouns [9].

Using these features, the script automatically generates a set of answer candidate verification rules in the form of regular expressions, used to exclude candidates that are not related to a particular question category, or more specifically, do not resemble any answer from the same category found in the corpus so far. Using these regular expressions, a proper answer length can also be discovered. For example, for the question category HUM:title, the proper scope of the answer can be determined from the following phrase: "[..]Bill Clinton, the former US president has visited[..]", using the syntactic features and knowing that there were no verbs found in the answers in this category. For the question category ENTY:animal in the phrase: "[..]the cheetah is the fastest animal on land[..]", the usage of the semantic feature,

a hyponym category "animal" for the noun cheetah, leads to the extraction of a legitimate answer candidate.

3 Conclusion and Future Work

Efficient acquisition and usage of knowledge from a large collection of text documents, like that of the Internet, requires the appropriate tools. We believe that our approach contributes to the improvement of the current QA Systems by addressing some of the flaws inherent in them. The Query Pattern Generation method demonstrates that the system can automatically acquire knowledge on how to optimally form a query for a given question category and question syntax. Using this method, a system also obtains information on where an answer candidate is likely to occur and what words or non-letter characters frequently connect it to a given query, even if these elements, were not present in the question. The results of the preliminary tests are promising, showing a significant improvement over the currently used method. The corpus-based answer candidate verification provides an effective tool to exclude answer candidates that do not belong to a given question category. Furthermore, by analyzing answers from the same question category, the system is able to some extent, to mimic human-like answers, thus allowing easier access to information for users, as well as computer systems, which can effectively use it in their working processes.
The usage of a corpus for question answering reveals potential that should be further explored and applied to improve the performance of QA Systems. The methods presented above provide such improvement while fulfilling the short time processing requirement. Our future work includes extensive system testing and evaluation of the proposed methods to determine their individual contribution to the overall performance of the system.

References

1. Abney S., Collins M., Singhal A. (2000) Answer Extraction. In Proceedings of the 6th ANLP Conference, 296-301
2. Brill E. (1995) Transformation-Based Error Driven Learning and Natural Language Processing: A Case Study in Part of Speech Tagging. Computational Linguistics 21(4), 543-566
3. Cortes C., Vapnik V. (1995) Support-Vector Network, Machine Learning 20, 1-25
4. Echihabi A., Marcu D. (2003) A Noisy-Channel Approach to Question Answering, Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics, 16-23
5. Hovy E., Gerber L., Hermajakob U., Junk M., Lin C. (2000) Question Answering in Webclopedia. Ninth Text REtrieval Conference (TREC-9), 655-664
6. Joachims T. (1999) Advances in Kernel Methods: Support Vector Learning, chapter Making large-Scale SVM Learning Practical. Chap. 11. MIT-Press

7. Kwok C., Etzioni O., Weld D.S. (2001) Scaling Question Answering to the Web. In the Proceedings of the 10th World Wide Web Conference (WWW2001), 150-161
8. Li X., Roth D. (2002) Learning Question Classifiers. In Proceedings of the 19th International Conference on Computational Linguistics (COLING'02), 556-562
9. Miller G. (1995) WordNet: a lexical database for English. Communications of the ACM 38(11), 39-41
10. Radev D., Fan W., Qi H., Wu H., Grewal A. (2002) Probabilistic Question Answering on the Web. Proceedings of the 11th World Wide Web Conference
11. Skowron M., Araki K. (2003) Basic Idea of Corpus-Supported Approach to Question Answering. Convention Record of the Hokkaido Chapters of the IEEE, 298-299
12. Soubbotin M.M., Soubbotin S.M. (2002) Patterns of Potential Answer Expressions as Clues to the Right Answer. In Proceedings of the 10th Text retrieval Conference (TREC10), 175-182
13. Zhang D., Lee W.S. (2003) A Web-based Question Answering System. Proceedings of the SMA Annual Symposium
14. Zhang D., Lee W.S. (2003) Question Classification using Support Vector Machines. Proceedings of the 26th ACM SIGIR
15. Zheng Z. (2002) AnswerBus Question Answering System. Proceeding of HLT Conference 2002, 24-27