Utilizing Co-Occurrence of Answers in Question Answering

Min Wu[1] and Tomek Strzalkowski[1,2]
[1] ILS Institute, University at Albany, State University of New York
1400 Washington Ave SS261, Albany NY, 12222
[2]Institute of Computer Science, Polish Academy of Sciences
minwu@cs.albany.edu, tomek@csc.albany.edu

Abstract

In this paper, we discuss how to utilize the co-occurrence of answers in building an automatic question answering system that answers a series of questions on a specific topic in a batch mode. Experiments show that the answers to the many of the questions in the series usually have a high degree of co-occurrence in relevant document passages. This feature sometimes can't be easily utilized in an automatic QA system which processes questions independently. However it can be utilized in a QA system that processes questions in a batch mode. We have used our pervious TREC QA system as baseline and augmented it with new answer clustering and co-occurrence maximization components to build the batch QA system. The experiment results show that the QA system running under the batch mode get significant performance improvement over our baseline TREC QA system.

1 Introduction

Question answering of a series of questions on one topic has gained more and more research interest in the recent years. The current TREC QA test set contains factoid and list questions grouped into different series, where each series has the target of a definition associated with it (Overview of the TREC 2004 Question Answering Track, Voorhees 2005). Usually, the target is also called "topic" by QA researchers. One of the restrictions of TREC QA is that "questions within a series must be processed in order, without looking ahead." That is, systems are allowed to use answers to earlier questions to help answer later questions in the same series, but can not use later questions to help answer earlier questions. This requirement models the dialogue discourse between the user and the QA system. However our experiments on interactive QA system show that some impatient QA users will throw a bunch of questions to the system and waiting for the answers returned in all. This prompted us to consider building a QA system which can accept as many questions as possible from users once in all and utilizing the relations between these questions to help find answers. We would also like to know the performance difference between the QA system processing the question series in an order and the QA system processing the question series as a whole. We call the second type of QA system as batch QA system to avoid the ambiguity in the following description in this paper.

What kind of relations between questions could be utilized is a key problem in building the batch QA system. By observing the test questions of TREC QA, we found that the questions given under the same topic are not independent at all. Figure-1 shows a series of three questions proposed under the topic "Russian submarine Kursk Sinks" and some relevant passages to this topic found in the TREC data set. These passages contain answers not to just one but to two or three of the questions. This indicates that the answers to these questions have high co-occurrence.

In an automatic QA system which processes the questions independently, the answers to the questions may or may not always be extracted due to algorithmic limitations or noisy information around the correct answer. However in building a batch QA system, the interdependence between the answers could be utilized to help to filter out the noisy information and pinpoint the correct answer for each question in the series.

Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL, pages 1169–1176,
Sydney, July 2006.

Topic Russian submarine Kursk sinks

1. When did the submarine sink? *August 12*
2. How many crewmen were lost in the disaster? *118*
3. In what sea did the submarine sink? *Barents Sea*

Some Related Passages

Russian officials have speculated that the Kursk collided with another vessel in the *Barents Sea*, and usually blame an unspecified foreign submarine. All *118* officers and sailors aboard were killed.

The Russian governmental commission on the accident of the submarine Kursk sinking in the *Barents Sea* on *August 12* has rejected 11 original explanations for the disaster.

.... as the same one carried aboard the nuclear submarine Kursk, which sank in the *Barents Sea* on *Aug. 12*, killing all *118* crewmen aboard.

The navy said Saturday that most of the *118*-man crew died *Aug. 12* when a huge explosion ....

Chief of Staff of the Russian Northern Fleet Mikhail Motsak Monday officially confirmed the deaths of *118* crewmen on board the Kursk nuclear submarine that went to the bottom of the *Barents Sea* on *August 12*.

Figure-1 Questions and Related Passages

We will discuss later in this paper how to utilize the co-occurrence of answers to a series of questions in building a batch QA system. The remainder of this paper is organized as follows. In the next section, we review the current techniques used in building an automatic QA system. Section 3 introduces the answers co-occurrence and how to cluster questions by the co-occurrence of their answers. Section 4.1 describes our TREC QA system and section 4.2 describes how to build a batch QA system by augmenting the TREC QA system with question clustering and answer co-occurrence maximization. Section 4.3 describes the experiments and explains the experimental results. Finally we conclude with the discussion of future work.

2 Related Work

During recent years, many automatic QA systems have been developed and the techniques used in these systems cover logic inference, syntactic relation analysis, information extraction and proximity search, some systems also utilize pre-compiled knowledge base and external online knowledge resource.

The LCC system (Moldovan & Rus, 2001; Harabagiu et al. 2004) uses a logic prover to select answer from related passages. With the aid of extended WordNet and knowledge base, the text terms are converted to logical forms that can be proved to match the question logical forms. The IBM's PIQUANT system (Chu-Carroll et al, 2003; Prager et al, 2004) adopts a QA-by-Dossier-with-Constraints approach, which utilizes the natural constraints between the answer to the main question and the answers to the auxiliary questions. Syntactic dependency matching has also been applied in many QA systems (Cui et al, 2005; Katz and Lin 2003). The syntactic dependency relations of a candidate sentence are matched against the syntactic dependency relations in the question in order to decide if the candidate sentence contains the answer. Although surface text pattern matching is a comparatively simple method, it is very efficient for simple factoid questions and is used by many QA systems (Hovy et al 2001; Soubbotin, M. and S. Soubbotin 2003). As a powerful web search engine and external online knowledge resource, Google has been widely adopted in QA systems (Hovy et al 2001; Cui 2005) as a tool to help passage retrieval and answer validation.

Current QA systems mentioned above and represented at TREC have been developed to answer one question at the time. This may partially be an artifact of the earlier TREC QA evaluations which used large sets of independent questions. It may also partially reflect the intention of the current TREC QA Track that the question series introduced in TREC QA 2004 (Voorhees 2005) simulate an interaction with a human, thus expected to arrive one at a time.

The co-occurrence of answers of a series of highly related questions has not yet been fully utilized in current automatic QA systems participating TREC. In this situation, we think it worthwhile to find out whether a series of highly related questions on a specific topic such as the TREC QA test questions can be answered together in a batch mode by utilizing the co-occurrences of the answers and how much it will help improve the QA system performance.

3 Answer Co-Occurrence and Question Clustering

Many QA systems utilize the co-occurrence of question terms in passage retrieval (Cui 2005).

Some QA systems utilize the co-occurrence of question terms and answer terms in answer validation. These methods are based on the assumption that the co-occurrences of question terms and answer terms are relatively higher than the co-occurrences of other terms. Usually the co-occurrence are measured by pointwise mutual information between terms.

During the development of our TREC QA system, we found the answers of some questions in a series have higher co-occurrence. For example, in a series of questions on a topic of disaster event, the answers to questions such as "when the event occurred", "where the event occurred" and "how many were injured in the event" have high co-occurrence in relatively short passages. Also, in a series of questions on a topic of some person, the answers to questions such as "when did he die", "where did he die" and "how did he die" have high co-occurrence. To utilize this answers co-occurrence effectively in a batch QA system, we need to know which questions are expected to have higher answers co-occurrence and cluster these questions to maximize the answers co-occurrence among the questions in the cluster.

Currently, the topics used in TREC QA test questions fall into four categories: "Person", "Organization", "Event" and "Things". The topic can be viewed as an object and the series of questions can be viewed as asking for the attributes of the object. In this point of view, to find out which questions have higher answers co-occurrence is to find out which attributes of the object (topic) have high co-occurrence.

We started with three categories of TREC QA topics: "Event", "Person" and "Organization". For "Event" topic category, we divided it into two sub-categories: "Disaster Event" and "Sport Event". From the 2004 & 2005 TREC QA test questions, we manually collected frequently asked questions on each topic category and mapped these questions to the corresponding attributes of the topic. We focused on frequently asked questions because these questions are easier to be classified and thus served as a good starting point for our work. However for this technique to scale in the future, we are expecting to integrate automatic topic model detection into the system. For topic category "Person", the attributes and corresponding named entity (NE) tags list as follows.

Attribute	**Attribute's NE tag**
Birth Date	Date
Birth Place	Location
Death Date	Date
Death Place	Location
Death Reason	Disease, Accident
Death Age	Number
Nationality	Nationality
Occupation	Occupation
Father	Person
Mother	Person
Wife	Person
Children	Person
Number of Children	Number
Real Name	Person, Other
Nick Name	Person, Other
Affiliation	Organization
Education	Organization

For each topic category, we collected 20 sample topics as well as the corresponding attributes information about these topics. The sample topic "Rocky Marciano" and the attributes are listed as follows:

Attribute	**Attribute Value**
Birth Date	September 1, 1923
Birth Place	Brockton, MA
Death Date	August 31, 1969
Death Place	Iowa
Death Reason	airplane crash
Death Age	45
Buried Place	Fort Lauderdale, FL
Nationality	American
Occupation	heavyweight champion boxer
Father	Pierino Marchegiano
Mother	Pasqualena Marchegiano
Wife	Barbara Cousins
Children	Mary Ann, Rocco Kevin
No. of Children	two
Real Name	Rocco Francis Marchegiano
Nick Name	none
Affiliation	none
Education	none

From each attribute of the sample topic, an appropriate question can be formulated and relevant passages about this question were retrieved from TREC data (AQUAINT Data) and the web. A topic-related passages collection was formed by the relevant passages of questions on all attributes under the topic. Among the topic-related passages, the pointwise mutual information (PMI) of attribute values were calculated which consequently formed a symmetric mutual information matrix. The PMI of two attribute values x and y was calculated by the following equation.

$$PMI(x,y) = \log \frac{p(x,y)}{p(x)p(y)}$$

All the mutual information matrixes under the topic category were added up and averaged in order to get one mutual information matrix which reflects the general co-occurrence rela-

tions between attributes under the topic category. We clustered the attributes by their mutual information value. Our clustering strategy was to cluster attributes whose pointwise mutual information is greater than a threshold λ. We choose λ as equal to 60% of the maximum value in the matrix.

The operations described above were automatically carried out by our carefully designed training system. The clusters learned for each topic category is listed as follows.

"Person" Topic

Cluster1: Birth Date; Birth Place
Cluster2a: Death Date; Death Place;
Death Reason; Death Age
Cluster2b: Death Date; Birth Date
Cluster3: Father; Mother
Cluster4: Wife; Children; Number of Children
Cluster5: Nationality; Occupation

"Disaster Event" Topic

Cluster1: Event Date; Event Location; Event Casualty;
Cluster2: Organization Involved, Person Involved

"Sport Event" Topic

Cluster1: Winner; Winning Score
Cluster2: Location, Date

"Organization" Topic

Cluster1: Founded Date; Founded Location; Founder
Cluster2: Headquarters; Number of Members

The reason for the clustering of attributes of topic category is for the convenience of building a batch QA system. When a batch QA system is processing a series of questions under a topic, some of the questions in the series are mapped to the attributes of the topic and thus grouped together according to the attribute clusters. Then questions in the same group are processed together to obtain a maximum of answers co-occurrence. More details are given in section 4.2.

4 Experiment Setup and Evaluation

4.1 Baseline System

The baseline system is an automatic IE-driven (Information Extraction) QA system. We call it IE-driven because the main techniques used in the baseline system: surface pattern matching and N-gram proximity search need to be applied to NE-tagged (Named Entity) passages. The system architecture is illustrated in Figure-2. The components indicated by dash lines are not included in the baseline system and they are added to the baseline system to build a batch QA system. As shown in the figure with light color, the two components are question classification and co-occurrence maximization. Both our baseline system and batch QA system didn't utilize any pre-compiled knowledge base.

In the question analysis component, questions are classified by their syntactic structure and answer target. The answer targets are classified as named entity types. The retrieved documents are segmented into passages and filtered by topic keywords, question keywords and answer target.

The answer selection methods we used are surface text pattern matching and n-gram proximity search. We build a pattern learning system to automatically extract answer patterns from the TREC data and the web. These answer patterns are scored by their frequency, sorted by question type and represented as regular expressions with terms of "NP", "VP", "VPN", "ADVP", "be", "in", "of", "on", "by", "at", "which", "when", "where", "who", ",", "-", "(". Some sample answer patterns of question type "when_be_np_vp" are listed as follows.

ADVP1 VP in <Date>([^<>]+?)<\/Date>
NP1.{1,15}VP.{1,30} in <Date>([^<>]+?)<\/Date>
NP1.{1,30} be VP in <Date>([^<>]+?)<\/Date>
NP1, which be VP in <Date>([^<>]+?)<\/Date>
VP NP1.{1,15} at .{1,15}<Date>([^<>]+?)<\/Date>
ADVP1.{1,80}NP1.{1,80}<Date>([^<>]+?)<\/Date>
NP1, VP in <Date>([^<>]+?)<\/Date>
NP1 of <Date>([^<>]+?)<\/Date>
NP1 be VP in <Date>([^<>]+?)<\/Date>

When applying these answer patterns to extract answer from candidate passages, the terms such as "NP", "VP", "VPN", "ADVP" and "be" are replaced with the corresponding question terms. The replaced patterns can be matched directly to the candidate passages and answer candidate be extracted.

Some similar proximity search methods have been applied in document and passage retrieval in the previous research. We applied n-gram proximity search to answer questions whose answers can't be extracted by surface text pattern matching. Around every named entity in the filtered candidate passages, question terms as well as topic terms are matched as n-grams. A question term is tokenized by word. We matched the longest possible sequence of tokenized word within the 100 word sliding window around the named entity. Once a sequence is matched, the corresponding word tokens are removed from the

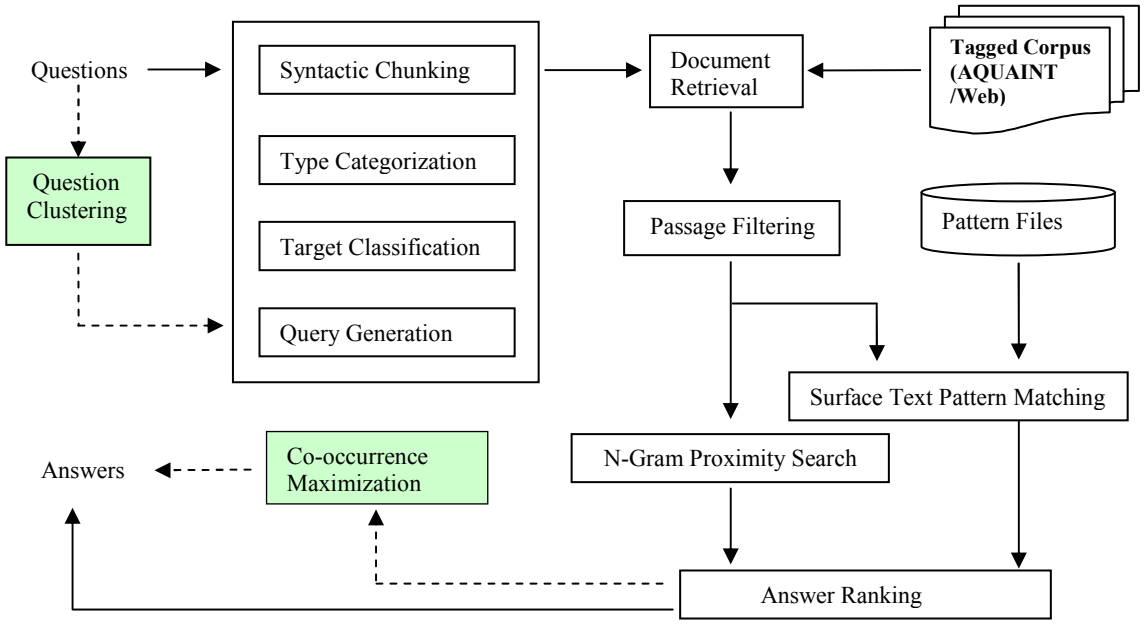


Figure-2 Baseline QA System & Batch QA System (dashed lines and light colored component)

token list and the same searching and matching is repeated until the token list is empty or no sequence of tokenized word can be matched. The named entity is scored by the average weighted distance score of question terms and topic terms.

Let $Num(t_i...t_j)$ denotes the number of all matched n-grams, $d(E, t_i...t_j)$ denotes the word distance between the named entity and the matched n-gram, $W1(t_i...t_j)$ denotes the topic weight of the matched n-gram, $W2(t_i...t_j)$ denotes the length weight of the matched n-gram. If $t_i...t_j$ contains topic terms or question verb phrase, 0.5 is assigned to $W1$, otherwise 1.0 is assigned. The value assigned to length weight $W2$ is determined by λ, the ratio value of matched n-gram length to question term length. How to assign $W2$ is illustrated as follows.

$W2(t_i...t_j)=0.4$ if $\lambda<0.4$;
$W2(t_i...t_j)=0.6$ if $0.4\leq\lambda\leq 0.6$;
$W2(t_i...t_j)=0.8$ if $\lambda>0.6$;
$W2(t_i...t_j)=0.9$ if $\lambda>0.75$.

The weighted distance score $D(E,QTerm)$ of the question term and the final score $S(E)$ of the named entity are calculated by the following equations.

$$D(E,QTerm)=\frac{\sum_{t_i...t_j}\frac{d(E,t_i..t_j)\times W1(t_i..t_j)}{W2(t_i..t_j)}}{Num(t_i..t_j)}$$

$$S(E)=\frac{\sum_{i}^{N}D(E,QTerm_i)}{N}$$

4.2 Batch QA System

The batch QA system is built from the baseline system and two added components: question classification and co-occurrence maximization. In a batch QA system, questions are classified before they are syntactically and semantically analyzed. The classification process consists of two steps: topic categorization and question mapping. Firstly the topic of the series questions is classified into appropriate topic category and then the questions can be mapped to the corresponding attribute and clustered according to the mapped attributes. Since the attributes of topic category is collected from frequently asked questions, there are some questions in the question series which can't be mapped to any attribute. These unmapped questions are processed individually.

The topic categorization is done by a Naïve Bayes classifier which employs features such as stemmed question terms and named entities in the question. The training data is a collection of 85 question series labeled as one of four topic categories: "Person", "Disaster Event", "Sport Event" and "Organization". The mapping of question to topic attribute is an example-based syntactic pattern matching and keywords matching.

The questions grouped together are processed as a question cluster. After the processing of answer selection and ranking, each question in the cluster gets top 10 scored candidate answers which forms an answer vector $A(a_1, ..., a_{10})$.

Suppose there are n questions in the cluster, the task of answer co-occurrence maximization is to retrieve a combination of n answers which has maximum pointwise mutual information (PMI). This combination is assumed to be the answers to the questions in the cluster.

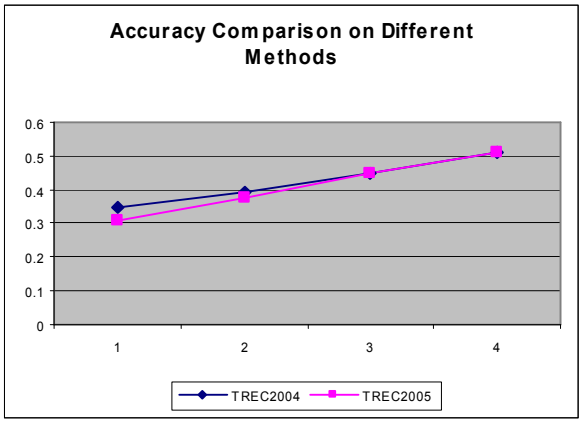
There are a total of 10^n possible combinations among all the candidate answers. If the PMI of every combination should be calculated, it is computationally inefficient. Also, some combinations containing noisy information may have higher co-occurrence than the correct answer combination. For example, the correct answers combination to questions showed in figure-1 is "August 12; 118; Barents Sea". However, there is also a combination of "Aug. 12, two; U.S." which has higher pointwise mutual information due to the frequently occurred noisy information of "two U.S. submarines" and "two explosions in the area Aug. 12 at the time".

To reduce this negative effect brought by the noisy information, we started from the highest scored answer and put it in the final answer list. Then we added the answers one by one to the final answer list. The added answer has the highest PMI with the answers in the final answer list. It is important here to choose the first answer added to the final answer list correctly. Otherwise, the following added answers will be negatively affected. So in our batch QA system, a correct answer should be scored highest among all the answer candidates of the questions in the cluster. Although this can't be always achieved, it can be approximated by setting higher threshold both in passage scoring and answer ranking. However, in the baseline system, passages are not scored. They are equally processed because we wanted to retrieve as many answer candidates as possible and answer candidates are ranked by their matching score and redundancy score.

4.3 Performance Evaluation

The data corpus we used is TREC QA data (AQUAINT Corpus). The test questions are TREC QA 2004 and TREC QA 2005 questions. Each topic is followed with a series of factoid questions. The number of questions selected from TREC 2004 collection is 230 and the number of question series is 65. The number of questions selected from TREC 2005 collection is 362 and the number of question series is 75.

We performed 4 different experiments: (1). Baseline system. (2). Batch QA system (Baseline system with co-occurrence maximization). (3). Baseline system with web supporting. (4). Batch QA with web supporting. We introduced web supporting into the experiments because usually the information on the web tends to share more co-occurrence and redundancy which is also proved by our results.



Compared between the baseline system and batch system, the experiment results show that the overall accuracy score has been improved from 0.34 to 0.39 on TREC 2004 test questions and from 0.31 to 0.37 on TREC 2005 test questions. Compared between the baseline system and batch system with web supporting, the accuracy score can be improved up to 0.498. We also noticed that the average number of questions under each topic in TREC 2004 test questions is 3.538, which is significantly lower than the 4.8267 average in TREC 2005 questions series. This may explain why the improvement we obtained on TREC2004 data is not as significant as the improvement obtained on TREC 2005 questions.

The accuracy score of each TREC2005 question series is also calculated. Figure3-4 shows the comparisons between 4 different experiment methods. We also calculate the number of question series with accuracy increased, unchanged and decreased. It is also shown in the following table. ("+" means number of question series with accuracy increased, "=" unchanged and "-" decreased.)

TREC2005 Question Series (75 question series)	+	-	=
Baseline + Co-occurrence	25	5	45
Baseline + Web	40	2	33
Baseline + Co-occurrence + Web	49	2	24

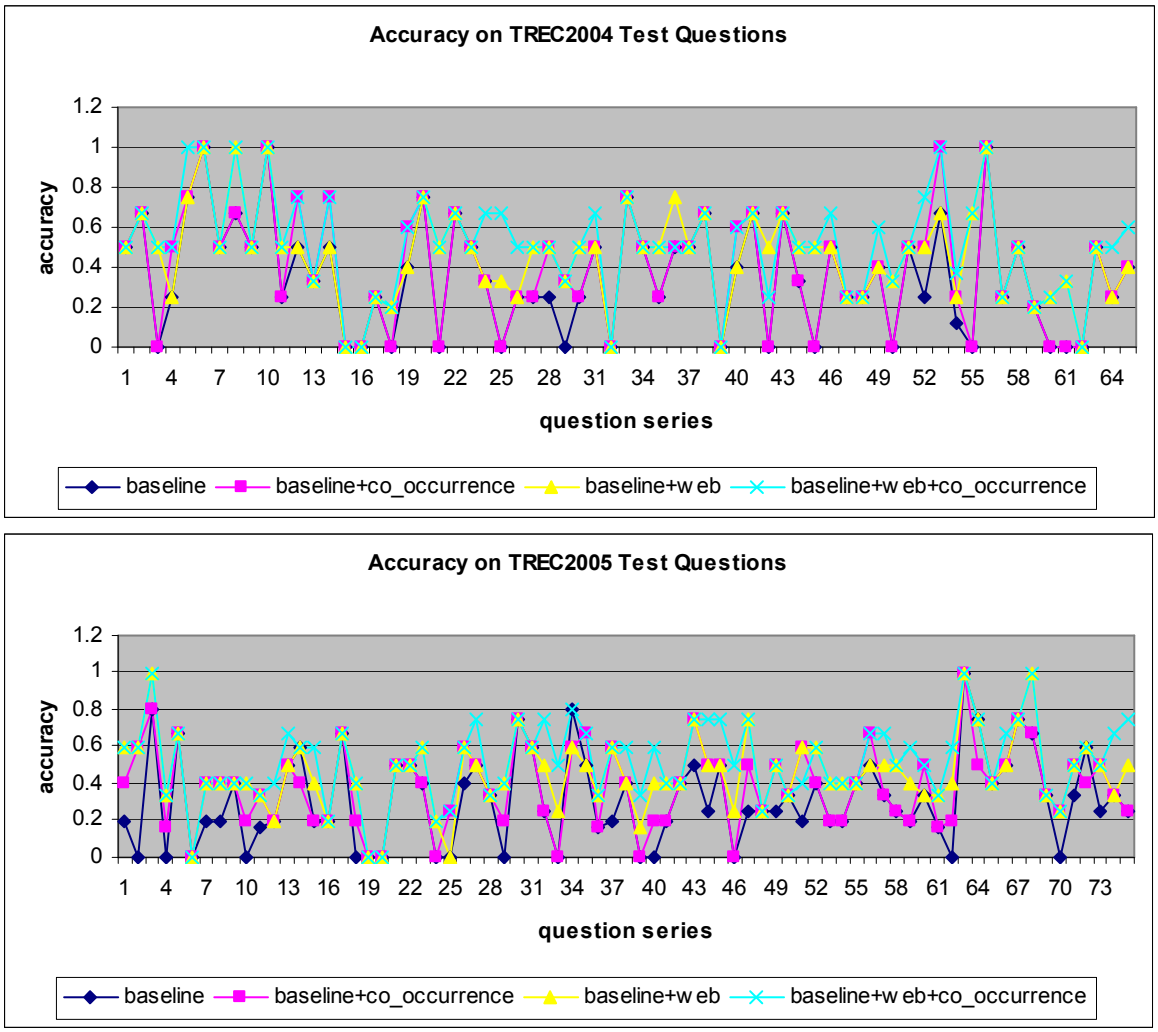


Figure 3-4 Comparison of TREC2004/2005 Question Series Accuracy

Some question series get unchanged accuracy because the questions can't be clustered according to our clustering template so that it can't utilize the co-occurrence of answers in the cluster. Some question series get decreased accuracy because the questions because the noisy information had even higher co-occurrence, the error occurred during the question clustering and the answers didn't show any co-relations in the retrieved passages at all. A deep and further error analysis is necessary for this answer co-occurrence maximization technique to be applied topic independently.

5 Discussion and Future Work

We have demonstrated that in a QA system, answering a series of inter-related questions can be improved by grouping the questions by expected co-occurrence of answers in text. The improvement can be made without exploiting the pre-compiled knowledge base.

Although our system can cluster frequently asked questions on topics of "Events", "Persons" and "Organizations", there are still some highly related questions which can't be clustered by our method. Here are some examples.

Topic Carlos the Jackal
1. When was he captured?
2. Where was he captured?

Topic boxer Floyd Patterson
1. When did he win the title?
2. How old was he when he won the title?
3. Who did he beat to win the title?

To cluster these questions, we plan to utilize event detection techniques and set up an event topic "Carlos the Jackal captured" during the answering process, which will make it easier to cluster "When was the Carlos the Jackal captured?" and "Where was the Carlos the Jackal captured?"

Can this answers co-occurrence maximization approach be applied to improve QA performance

on single questions (i.e. 1-series)? As suggested in the reference paper (Chu-Carrol and Prager), we may be able to add related (unasked) questions to form a cluster around the single question. Another open issue is what kind of effect will this technique bring to answering series of "list" questions, i.e., where each question expects a list of items as answer. As we know that the answers of some "list" questions have pretty high co-occurrence while others don't have co-occurrence at all. Future work involves experiments conducted on these aspects.

Acknowledgement

The Authors wish to thank BBN for the use of NE tagging software IdentiFinder, CIIR at University of Massachusetts for the use of Inquery search engine, Stanford University NLP group for the use of Stanford parser. Thanks also to the anonymous reviewers for their helpful comments.

References

Chu-Carrol, J., J. Prager, C. Welty, K. Czuba and D. Ferrucci. "A Multi-Strategy and Multi-Source Approach to Question Answering", In Proceedings of the 11th TREC, 2003.

Cui, H., K. Li, R. Sun, T.-S. Chua and M.-Y. Kan. "National University of Singapore at the TREC 13 Question Answering Main Task". In Proceedings of the 13th TREC, 2005.

Han, K.-S., H. Chung, S.-B. Kim, Y.-I. Song, J.-Y. Lee, and H.-C. Rim. "Korea University Question Answering System at TREC 2004". In Proceedings of the 13th TREC, 2005.

Harabagiu, S., D. Moldovan, C. Clark, M. Bowden, J. Williams and J. Bensley. "Answer Mining by Combining Extraction Techniques with Abductive Reasoning". In Proceedings of 12th TREC, 2004.

Hovy, E. L. Gerber, U. Hermjakob, M. Junk and C.-Y. Lin. "Question Answering in Webclopedia". In Proceedings of the 9th TREC, 2001.

Lin, J., D. Quan, V. Sinha, K. Bakshi, D. Huynh, B. Katz and D. R. Karger. "The Role of Context in Question Answering Systems". In CHI 2003.

Katz, B. and J. Lin. "Selectively Using Relations to Improve Precision in Question Answering". In Proceedings of the EACL-2003 Workshop on Natural Language Processing for Question Answering. 2003.

Moldovan, D. and V. Rus. "Logical Form Transformation of WordNet and its Applicability to Question Answering". In Proceedings of the ACL, 2001.

Monz. C. "Minimal Span Weighting Retrieval for Question Answering" In Proceedings of the SIGIR Workshop on Information Retrieval for Question Answering. 2004.

Prager, J., E. Brown, A. Coden and D. Radev. "Question-Answering by Predictive Annotation". In Proceedings of SIGIR 2000, pp. 184-191. 2000.

Prager, J., J. Chu-Carroll and K. Czuba. "Question Answering Using Constraint Satisfaction: QA-By-Dossier-With-Constraints". In Proceedings of the 42nd ACL. 2004.

Ravichandran, D. and E. Hovy. "Learning Surface Text Patterns for a Question Answering System". In Proceedings of 40th ACL. 2002.

Soubbotin, M. and S. Soubbotin. "Patterns of Potential Answer Expressions as Clues to the Right Answers". In Proceedings of 11th TREC. 2003.

Voorhees, E. "Using Question Series to Evaluate Question Answering System Effectiveness". In Proceedings of HLT 2005. 2005.