Cluster-Based Selection of Statistical Answering Strategies

Lucian Vlad Lita and **Jaime Carbonell**
Carnegie Mellon University
{llita, jgc}@cs.cmu.edu

Abstract

Question answering (QA) is a highly complex task that brings together classification, clustering, retrieval, and extraction. Question answering systems include various statistical and rule-based components that combine and form multiple strategies for finding answers. However, in real-life scenarios efficiency constraints make it infeasible to simultaneously use all available strategies in a QA system. To address this issue, we present an approach for carefully selecting answering strategies that are likely to benefit individual questions, without significantly reducing performance. We evaluate the impact of strategy selection on question answering performance at several important QA stages: document retrieval, answer extraction, and answer merging. We present strategy selection experiments using a statistical question answering system, and we show significant efficiency improvements. By selecting 10% of the available answering strategies, we obtained similar performance when compared to using all of the strategies combined.

1 Introduction and Related Work

In the past few years, an increasing number of question answering systems have started employing multi-strategy approaches that attempt to complement one another when searching for answers to questions. These approaches often include multiple question classifications, several retrieval approaches, multiple answer extractors, and different data sources. Question answering performance is often presented within the context of official evaluations where systems are processing batches of questions with no time constraints. However, in real-life scenarios, only a limited number of these strategies (component combinations, parameter settings, etc) can be fully explored. In these cases, the trade-off between performance and problem complexity (and indirectly response time) require careful selection of answering strategies such that performance is optimized according to realistic constraints.

We present an *answering strategy* selection approach that directly addresses the performance-complexity trade-off and we apply it to a statistical, instance-based question answering system. In this paper we investigate the benefits of a principled strategy selection method when applied to the main components of a QA system: document retrieval, answer extraction, and answer merging (i.e. overall QA performance). We present experiments which show that by carefully selecting less than 10% of the available answering strategies, no significant performance degradation is observed. Moreover, we integrate a cluster-based confidence scoring method with an answer merging component and observe significant question answering performance improvements.

Experiments [Collins-Thompson *et al.*, 2004] using the CMU Javelin [Nyberg *et al.*, 2003] and Waterloo's MultiText [Clarke *et al.*, 2002] question answering systems corroborate the expected correlation between improved document retrieval performance and QA accuracy across systems. Results suggest that retrieval methods adapted for question answering which include question analysis performed better than ad-hoc IR methods, supporting previous findings [Monz, 2003].

Several practical approaches have been developed to deal with the complexity of the question answering process. The SMU system [Harabagiu *et al.*, 2000] and later the LCC system [Moldovan *et al.*, 2002] incorporate feedback loops between components of their question answering system. The CMU system treats the QA process as planning problem, formalizing the notion of feedback. Several other QA systems using statistical components [Chu-Carroll *et al.*, 2003; Nyberg *et al.*, 2003; Lita and Carbonell, 2004] introduced multiple answering strategies that can be used simultaneously and their results can be combined. Furthermore, when answering complex questions, [Harabagiu and Lacatusu, 2004] argue for a multi-strategy approach for question processing, extraction, and selection.

The strategy selection problem is closely related to active learning, which explores the trade-off between performance and cost. While active learning algorithms suggest data for labeling by minimizing the expected error [Roy and McCallum, 2001], in the problem of strategy selection, the goal is to reduce QA complexity by limiting the number of answering strategies while not increasing the error of the QA process.

2 Answering Strategies

Most question answering systems are implemented as a pipeline where different stages successively process data.

However, for each stage in the QA pipeline there is a variety of methods that can be employed. Each method typically has different parameters, needs different resources, and may produce answers with different confidences, which may not be comparable across methods. We will refer to a complete combination of components at each stage in the pipeline as an answering strategy. In most of today's QA systems, an **answering strategy** consists of the following components:

1. **question analysis** – produces an expected answer type, extracts question keywords, and analyzes the question. Part of speech tagging, parsing, semantic analysis and additional processing are often used in question analysis.
2. **retrieval** – specifies what query types and what query content yield high expected performance. Very often QA systems pre-specify the query type and additional content according to the question and answer types identified earlier in the strategy.
3. **answer extraction** – specifies how answers are identified from relevant documents. Answer extraction methods range from rule and pattern-based extractors to hidden markov models (HMM), maximum entropy, and support vector machine-based extractors.

Stage	**Strategy** S_A	**Strategy** S_B
1) *answer type*	temporal	year
2) *queries*	when mozart die	mozart die biography mozart died death
3) *extraction*	rule-based	HMM

Table 1: Answering strategies S_A and S_B use different answer types, different queries, and extraction methods.

When applied to a new question, an answering strategy processes the question text, retrieves documents and extracts a set of possible answers. In the case when multiple strategies are simultaneously applied to a new question, an **answer merging** component is employed to combine answers and confidences into a final answer set. Table 1 shows two simplistic strategies for the question "When did Mozart die?". In realistic scenarios the question analysis component produces more information than just an expected answer type, several queries are generated according to pre-specified types, and various processing is performed before answer extraction.

2.1 Cluster-Based Strategies

As the first stage in answering strategies, most question answering systems employ question ontologies. These ontologies combine expected answer types (date, location etc) and question types (*birthday(X)*, *nickname(X)*, *construction_date(X)* etc). Consider again the question "When did Mozart die?". Depending on the desired answer type granularity, this question can be classified as a *temporal* question, a *temporal::year* question, or more specifically as a *temporal::year::death_year* question. Each classification may lead to an entirely different answering strategy. Existing systems consider answer types ranging from simple answer type sets and QA specific ontologies to semantic networks such as WordNet which provide better coverage and more specificity. However, these ontologies are very restrictive and only take into account the answer type, disregarding question structure, or domain knowledge.

An approach that is similar to using ontologies is question clustering [Lita and Carbonell, 2004], in which training questions are clustered according to different similarity criteria such as shared number of n-grams (contiguous sequences of words), semantic similarity, and same answer type. Compared to fixed ontologies, this approach is adaptive to training data, is language and domain independent, and allows overlapping types (clusters) that do not have a hierarchical relationship. Figure 1 shows the relationship between an ontology and clustering as it is used in question analysis (stage 1) of a QA process. If clustering is performed at different granularities, each cluster corresponds to an ontology node. Thus, individual answering strategies are built for different clusters, rather than different ontology nodes.

This clustering approach[1] allows each component in an answering strategy to be learned only from i) training questions and ii) their known correct answers. Therefore strategies are learned for individual clusters, using corresponding questions as training data. The retrieval component learns which queries and query types have high performance when run on in-cluster training questions. The answer extraction component is trained on correct answers for all in-cluster questions. Finally, the answer merging component considers cluster statistics, retrieval performance, extraction performance and merges answer sets produced by answering strategies.

If there is sufficient data for learning (i.e. sufficient number of questions), the more clusters of training questions a QA system generates, the more answering strategies will be applied to new questions. However, while QA performance may increase with additional answering strategies, so will the noise (e.g. from irrelevant clusters) and the time it takes to actually run the strategies. Our goal is to allow the existence of multiple cluster-based strategies, but only select a set of clusters associated to the strategies most likely to lead to high performance. For document retrieval, high performance translates into high recall of relevant documents. For answer extraction high performance corresponds to a large number of correct answers being extracted.

Queries learned by different strategies often lead to some of the same relevant documents – e.g. the queries "*the first aria composed Mozart*" vs. "*aria Mozart*" may lead to an overlap in their retrieved document sets. If a strategy already leads to the retrieval of a document d_i, subsequent strategies will not benefit if they retrieve d_i again. Therefore, each strategy selection depends on the n-1 previously selected strategies

3 Experiments & Results

For our experiments we have chosen to use a statistical question answering system for several reasons. Statistical QA systems are usually faster to implement than rule-based systems, they require less knowledge resources and limited manual input, and their results are easier to replicate on standard datasets. In this paper we have implemented an instance-based question answering (IBQA) system [Lita and Car-

[1] for more details about this data-driven framework, refer to the [Lita and Carbonell, 2004] publication

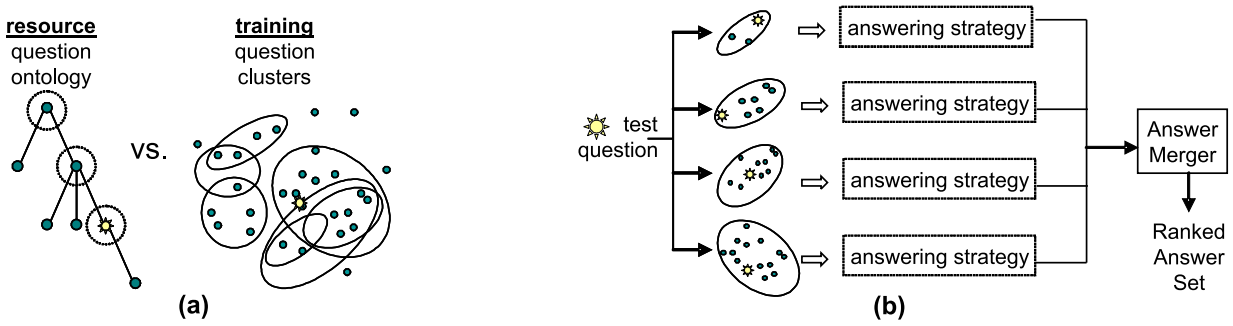

Figure 1: (a) Classification according to a question ontology versus classification according to a set of clusters in the training data (b) Answering a new question using multiple strategies learned from training questions in each cluster (or ontology node). The answer sets produced by individual cluster-based strategy are merged into a single ranked answer set.

bonell, 2004]. The question analysis component relies on clustering training questions at different levels of granularity and then classifying new questions into these clusters. For each cluster, an answering strategy is learned which equivalent to a set of retrieval, extraction, and answer type models. In the remainder of the paper we shall refer to these strategies as *cluster-based* since a single answering strategy is learned from the training questions of an individual cluster.

When new questions are presented to a QA system, they are classified according to the training clusters and the corresponding answering strategies are activated, each generating a set of answers. The strategy-specific answers are then merged into a final ranked answer list. Details on learning and implementation of the answer type, retrieval, and extraction models can be found in [Lita and Carbonell, 2004].

The question datasets used in the experiments presented in this paper consist of all temporal questions from previous trec evaluations TREC 8-13 [Voorhees, 1999 to 2004]. Temporal questions have the advantage of having a relatively high density necessary in training statistical QA components. They are also distributed such that they cover simple questions such as "*When did Beethoven die?*", but also structurally more complex questions such as "*What year did General Montgomery lead the Allies to a victory over the Axis troops in North Africa?*". Each question in this collection has a set of corresponding answer keys in the form of regular expressions.

The questions were processed with a part of speech tagger [Brill, 1994], a morphological analyzer [Minnen *et al.*, 2001], and a sentence splitter. Synonyms and hypernyms were extracted and used to enhance keyword-based queries in document retrieval. Several queries are generated for each answering strategy and at most one hundred documents were retrieved for each of these query through the Google API (*www.google.com/api*). Documents containing any reference to TREC or the actual question, and other problematic content were filtered out.

Two desired qualities in question answering systems are 1) *correctness* – correct answers should have higher rank than incorrect answers, and 2) *redundancy* – correct answers should occur more than once in documents. More specifically, the retrieval component should produce multiple relevant documents; the answer extraction component should be able to extract multiple instances of a correct answer and at the same time assign them higher confidence scores compared to incorrect answers. Towards this end, we evaluate question answering performance using several metrics:

- Mean reciprocal rank (**MRR**) is computed as the inverse of the rank of the first correct answer returned by a system. The higher the first correct answer can be found in the ranked answer list, the higher the MRR score.
- The second metric tests the existence of at least one correct answer within the top five answers (**Top5**) and is less strict compared to MRR.
- Towards the goal of redundancy, for document retrieval we evaluate the number of relevant documents obtained (i.e. documents containing at least a correct answer) and for answer extraction we evaluate the number of correct answers extracted.

	questions		*strategies*	
	MRR	*Top5*	*MRR*	*Top5*
all	0.431	0.656	0.256	0.461
extracted	0.576	0.879	0.267	0.480

Table 2: Statistical QA system results on temporal questions for i) *all* questions and ii) questions for which at least an answer was *extracted*. We show average score over all questions, but also the average performance over all strategies. However, for question performance, only a small number of strategies need to be successful.

For comparison purposes, we have evaluated the instance-based QA system using MRR and Top5. Table 2 shows the MRR and the Top5 scores for the question dataset used in this paper. We present the QA system performance over all questions (*all*) in the dataset as well as the performance the questions with at least one proposed answer (*extracted*). Intuitively, the latter measure can be viewed as precision and the former measure as recall. At the question level, which is what TREC evaluations [Voorhees, 1999 to 2004] use, we show the performance by averaging over all questions – for each question we combine results from all strategies. At the strategy (cluster) level, we compute the performance by averaging over all strategies – for each strategy, we compute the performance over all questions it can be applied to. Note that not

all answering *strategies* are successful, hence the lower MRR average. At the same time, *questions* benefit from multiple strategies and therefore their average MRR is higher. Performance in the above experiments was computed through leave-one-out cross-validation.

In these experiments, each iteration corresponds to an answering strategy being selected. The newly selected answering strategy includes specific query types that are known to perform well on training questions in the their respective clusters. In addition, a cluster-specific SVM extractor finds and scores potential answers. In some of these experiments, the computation of a greedy-optimal (oracle) cluster selection method is tractable. This is not to be confused with a global optimal classifier that finds the absolute best strategy selection sequence – when tractable, we present its performance.

3.1 Selection for Document Retrieval

We assume a document to be relevant in the context of question answering if it contains a correct answer in a correct context. Since it is very difficult to automatically evaluate the correctness of context, the notion of relevance is sometimes relaxed to whether a document contains the correct answer, regardless of context. Through cluster-specific data, the retrieval component of the QA system learns n-grams and features that improve retrieval when added to queries. The improvement is measured when these queries are used to retrieve documents for the questions in the same cluster. The learned features become part of the cluster-based answering strategy which can be applied to new similar questions.

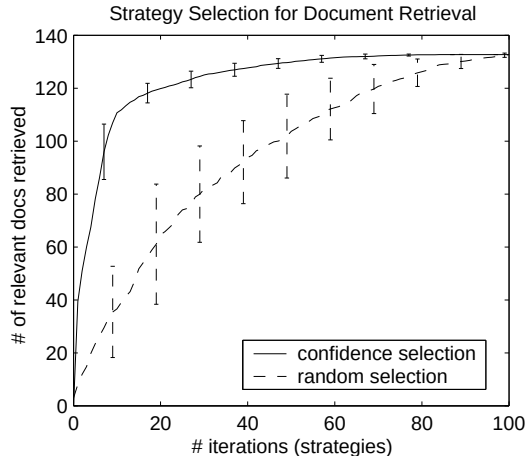

Figure 2: Smart selection based on strategy confidence allows the QA system to employ only 10% of its available strategies to retrieve 80% of the accessible relevant documents.

When trying to answer the question "*When did Mozart die?*" it may be beneficial to create queries that contain features such as "biography", "cemetery", "spent his life", "sacrificed himself", etc. In many QA systems the retrieval component contains rules for building better queries for specific types of questions – in our example: *time_of_death*. In the cluster-based approach, these features are learned from other similar questions in the training data, and get added to cluster-specific answering strategies. We measure the retrieval confidence $conf(A_{IR}(C_j)|q)$ of an answering strategy A derived from cluster C_j given a new test question q:

$$conf(A_{IR}(C_j)|q) = P(d^+|A_{IR}(C_j)) \cdot P(C_j|q) \cdot s(j) \quad (1)$$

where $P(d+|A_{IR}(C_j))$ is the probability of retrieving a relevant document d^+ using strategy $A_{IR}(C_j)$ and is measured by testing its effectiveness on a held-out set of questions from the cluster. $P(C_j|q)$ is the probability of a cluster containing questions similar to q and is given by the average similarity between q and q_j ($i \in C_j$) normalized over all clusters. $s(j)$ is a minimal cardinality condition for clusters.

Figure 2 shows the effect of using confidence-based selection in order to iteratively add appropriate answering strategies (i.e. beneficial query content). The more strategies are employed to create queries and retrieve new documents, the less time will be available for answer extraction and answer merging. The iterative process offers a good trade-off between performance and number of strategies used, as well as a good basis for user-defined utility functions. In our experiments, if the QA system selects only 10% of the available strategies, the retrieval performance is approximately 80% of the maximum achievable using the existing current strategies.

3.2 Selection for Answer Extraction

For a particular cluster, the SVM answer extractor is trained on the documents obtained by running the retrieval component of the answering strategy (i.e. using the learned queries to retrieve documents). The basic features include proximity features, sentence statistics, and patterns (n-grams and paraphrases) that discriminate best between sentences that contain correct answers and sentences that do not. For the classifier, the value of these features is given by information gain.

The answer extractor used in these experiments consists of a support vector machine (SVM) classifier [Joachims, 2002] producing probabilities, and whose task is to decide whether or not sentences contain at least one correct answer. The SVM was trained on features extracted from one-sentence passages containing at least one keyword from the original question. The features consist of: distance between keywords and potential answers, keyword density in a passage, simple statistics such as document and sentence length, query type, lexical n-grams (up to six words in length), and paraphrases.

Under the cluster-based approach, it is not sufficient to train an answer extractor for each answering strategy. These strategies are trained on different number of questions (i.e. cluster size), they are sensitive to the notion of cluster relevance, and they are based on different query types and different relevant document distributions. Therefore, extractor confidence has to be taken within the context of its history – i.e. the rest of the answering strategy. We measure the answer extraction confidence $conf(A_{AE}(C_j)|q)$ of a strategy A derived from cluster C_j given a new test question q:

$$conf(A_{AE}(C_j)|q) = P(a^+|A_{AE}(C_j)) \cdot conf(A_{IR}(C_j)|q) \quad (2)$$

where $P(a^+|A_{AE}(C_j)$ is the probability of extracting a correct answer a^+ using the answering strategy $A_{AE}(C_j)$ –

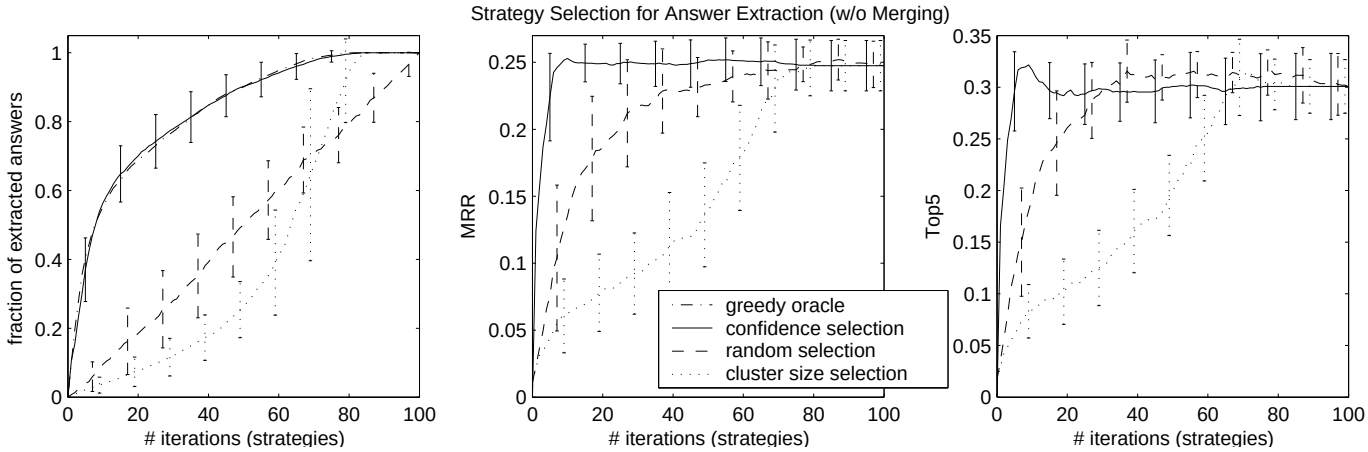

Figure 3: Selection based on confidence yields the best performance after carefully selecting a limited number of strategies (cluster, queries, and extractor) to answer a question. However, for better redundancy it is better use additional answering strategies if further correct answers are required. No answer merging method has been used here – answers preserve their individual strategy-generated scores.

more specifically, using the cluster-trained SVM extractor. $P(a^+|A_{AE}(C_j)$ is measured by testing its effectiveness on a held-out set of training questions from the cluster.

In Figure 3 we evaluate the effectiveness of our selection method (*confidence selection*) according to MRR, Top5, and the fraction of correct answers extracted out of the total number of correct answers that would be extracted if all strategies were used. The *random selection* consists of random sampling from the available strategies and using them to extract more answers. The *cluster size selection* is an intuitive baseline which gives priority to more specific, focused strategies that correspond to clusters with higher similarity to the test question: $P(C_j|q)$. However, it does not perform well, since cluster similarity is a necessary property, but it is not sufficient in the selection process. Finally, the *greedy oracle* optimizes at each iteration the strategy that yields the most additional correct answers. In many cases, our *confidence selection* method performs virtually indistinguishable from the oracle sequence.

While there is a benefit in using this selection method to quickly obtain a larger number of correct answers, if high answer redundancy is required, then further strategies must be used. However, in terms of MRR and Top5 scores, a very small number of carefully selected strategies can be as effective as utilizing all of the available answering strategies. A very important observation is that performance does not degrade with subsequent iterations and increased number of strategies. This can be explained by the fact that the best strategies provide the highest confidence values and corresponding answer scores, while unsuccessful strategies do not introduce additional noise. In these experiments no answer merging method was used. Each instance of an answer was treated separately, with its original confidence score given by a specific strategy. This approach does not provide any boost in confidence if the same answer has been seen more than once. However, it provides a measure of relative answering strategy noise and is informative to the performance of the answer extraction stage in a QA system.

3.3 Selection for Answer Merging

At this stage in the QA pipeline, all answering strategies that were activated produce a strategy-specific answer set. The task of the answer merging component is to make use of redundancy and re-score the answers such that the correct answers are ranked higher than incorrect answers. The answering merging method we implemented consists of a weighted sum of the individual answer confidences for all answer instances with the same surface form. The answer confidence $conf(a_k|q)$ of an answer a_k at the end of the question answering process is aggregated across all clusters C_j and is given by:

$$conf(a_k|q) = \sum_{a_k} \sum_{C_j} P(a_k|A_{AE}(C_j)) \cdot conf(A_{AE}(C_j)|q) \quad (3)$$

where $P(a_k|A_{AE}(C_j)$ is the probability of extracting a correct answer a_k using the answering strategy $A_{AE}(C_j)$.

Both in terms of MRR and Top5 scores, as seen from Figure 4 and Figure 3, the weighted answer merging method gains approximately 0.15 MRR points (60%) and also 0.15 Top5 points (43%) in performance. The gap trade-off between using the *confidence selection* scores and using all strategies also improved. As in the case of answer extraction, it is encouraging that the *confidence selection* approach closely follows the *greedy optimal* selection.

4 Conclusions & Future Work

An increasing number of question answering systems are relying on multi-strategy approaches in order to find answers to questions. They rely on multiple question classifications, answer extractors, multiple retrieval methods using several data sources, and different web-based services. While QA performance is often presented on batch processing of questions with no time constraints, in real-life scenarios, only a

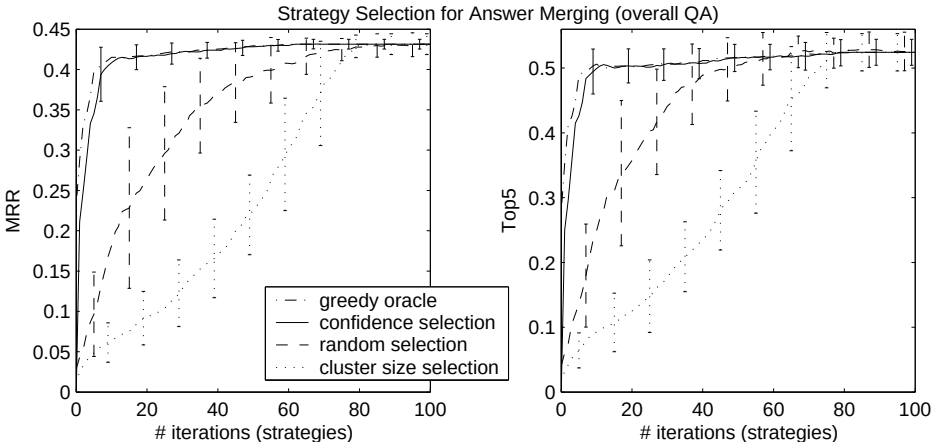

Figure 4: Final QA performance – for answer merging, confidence based selection performance allows the QA system to select less than 10% of the strategies with nearly maximum performance. The trade-off is marginally better for the MRR metric, since it requires better relative scoring from answer merging.

limited number of these strategies can be fully explored. Under these scenarios response time and performance trade-offs require careful selection of answering strategies such that performance is optimized subject to constraints.

In this paper we presented a strategy selection approach that directly addresses these issues and we apply it to a statistical instance-based question answering system. Through experiments we have shown the significant benefits of a principled strategy selection method on document retrieval, answer extraction, and answer merging (i.e. overall QA performance) using several metrics. By carefully selecting 10% of the available answering strategies, we obtained similar performance to the scenario in which we employ all strategies. Moreover, the cluster-based confidence scoring method was also incorporated into answer merging which improved performance in terms of MRR and Top5 significantly.

We are currently experimenting with multiple answer extractors that are inherently different from SVM-based extractors in order to increase the potential extraction coverage. We also intend to investigate additional selection methods and explore active learning approaches. In particular, we are interested in the effect of incremental training data on the selection of answering strategies.

References

[Brill, 1994] E. Brill. Some advances in rule-based part of speech tagging. *AAAI*, 1994.

[Chu-Carroll *et al.*, 2003] J. Chu-Carroll, K. Czuba, J. Prager, and A. Ittycheriah. In question answering, two heads are better than one. In *HLT-NAACL*, 2003.

[Clarke *et al.*, 2002] C. Clarke, G. Cormack, G. Kemkes, M. Laszlo, T. Lynam, E. Terra, and P. Tilker. Statistical selection of exact answers. In *TREC*, 2002.

[Collins-Thompson *et al.*, 2004] K. Collins-Thompson, E. Terra, J. Callan, and C. Clarke. The effect of document retrieval quality on factoid question-answering performance. In *SIGIR*, 2004.

[Harabagiu and Lacatusu, 2004] S. Harabagiu and F. Lacatusu. Strategies for advanced question answering. In *HLT-NAACL Workshop on Pragmatics of QA*, 2004.

[Harabagiu *et al.*, 2000] S. Harabagiu, D. Moldovan, M. Pasca, R. Mihalcea, M. Surdeanu, R. Bunescu, R. Girju, V. Rus, and P. Morarescu. Falcon: Boosting knowledge for answer engines. In *TREC*, 2000.

[Joachims, 2002] T. Joachims. Learning to classify text using support vector machines. *Dissertation*, 2002.

[Lita and Carbonell, 2004] L. V. Lita and J. Carbonell. Instance-based question answering: A data-driven approach. *EMNLP*, 2004.

[Minnen *et al.*, 2001] G. Minnen, J. Carroll, and D. Pearce. Applied morphological processing of english. *Natural Language Engineering*, 2001.

[Moldovan *et al.*, 2002] D. Moldovan, S. Harabagiu, R. Girju, P. Morarescu, F. Lacatusu, A. Novischi, A. Badulescu, and O Bolohan. Lcc tools for question answering. In *TREC*, 2002.

[Monz, 2003] C. Monz. From document retrieval to question answering. In *Ph. D. Dissertation, Universiteit Van Amsterdam*, 2003.

[Nyberg *et al.*, 2003] E. Nyberg, T. Mitamura, J. Callan, J. Carbonell, R. Frederking, K. Collins-Thompson, L. Hiyakumoto, Y. Huang, C. Huttenhower, S. Judy, J. Ko, A. Kupsc, L. V. Lita, V. Pedro, D. Svoboda, and B. Vand Durme. The javelin question-answering system at trec 2003: A multi strategy approach with dynamic planning. In *TREC*, 2003.

[Roy and McCallum, 2001] N. Roy and A. McCallum. Toward optimal active learning through sampling estimation of error reduction. In *ICML*, 2001.

[Voorhees, 1999 to 2004] E. M. Voorhees. Overview of the trec 2003 question answering track. In *TREC*, 1999 to 2004.