Leverage Web-based Answer Retrieval and Hierarchical Answer Selection to Improve the Performance of Live Question Answering

Guoshun Wu, Man Lan*
Shanghai Key Laboratory of Multidimensional Information Processing
Department of Computer Science and Technology,
East China Normal University, Shanghai 200241, P. R. China
`51141201064@ecnu.cn, mlan@cs.ecnu.edu.cn`*

Abstract

This paper describes the system we submitted to the Live QA Track in TREC 2015 which focused on *live* question answering for real-user questions. The real user questions extracted from Yahoo! Answers will be sent to the participant systems and the participants are required to provide an answer in real time (less than one minute). To address this task, we constructed a pipeline system which includes four components, i.e., question expansion, answer document retrieval, candidate answers selection and answer re-ranking. The purpose of the last three components aims at identifying accurate answers in a hierarchical manner. Then the answers with less than $1,000$ characters are returned to the server in less than one minute. Our system outperforms the averaged scores of all submitted runs on a 4-level Likert scale and ranks the $2th$ out of 14 teams.

1 Introduction

Open question answering (QA) has been a widely studied research problem in the past few years. The state-of-the-art QA systems are usually implemented in a pipeline architecture, consisting of several key components including question classification, question expansion, answer retrieval, answer reranking, etc. Most previous studies have focused on factoid questions (e.g., *Who*, *When*, *Where*). Recently, with the development of community-based Web sites such as Yahoo! Answer[1] (YA) and WikiAnswers[2], more researchers have concentrated on the domain of Community-based Question Answering (CQA). The content of CQA is usually generated by real users and there are more non-factoid questions in CQA than in QA, which make it quite difficult to get the right answer directly. One way to address this problem is to find the similar questions from a large CQA archives for each given question and then regard the answer for the most similar question as the answer for the original question.

The Live QA Track in TREC 2015 is a CQA-based task, focusing on *live* question answering for real user questions. These real user questions extracted from the stream of most recent questions submitted on the YA site that have not yet been answered by humans, will be sent to the participant systems. Then the participant systems are required to provide an answer with the support sources where it is extracted and synthesized in real time (less than one minute). The length of returned answer is limited to $1,000$ characters and will be manually judged by TREC editors on a 4-level Likert scale. The questions are restricted to the following eight YA categories[3], i.e., Arts & Humanities, Beauty & Style, Computers & Internet, Health, Home & Garden, Pets, Sports, and Travel. This Live QA Track has several challenges. Firstly, the questions are generated by real users, resulting in various qualities of questions. Secondly, the question body usually contains long texts, which requires to a deep linguistic analysis to extract the real purpose of each question. Finally, a time-consuming complicated system is not suitable due to the limitation of runtime.

Inspired by previous work on QA (Wang, 2006; Kolomiyets and Moens, 2011) and CQA (Bernhard and Gurevych, 2009; Surdeanu et al., 2011), in this

[1]https://answers.yahoo.com/
[2]http://wiki.answers.com/
[3]https://answers.yahoo.com/dir/index

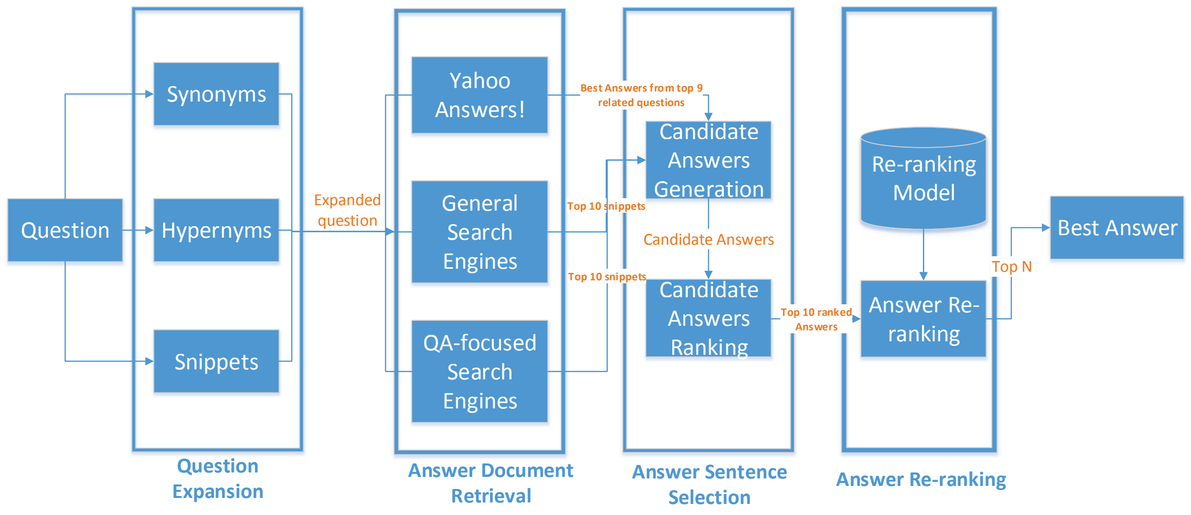

Figure 1: The architecture of our proposed system.

work we propose a pipeline system, consisting of four key components, i.e., question expansion (QE) (e.g., expanding questions with relevant information), answer document retrieval (ADR) (e.g., retrieving candidate documents which contain answers to the given question), answer sentence selection (ASS) (e.g., selecting answer sentences from the retrieved documents) and answer re-ranking (ARR) (e.g., extracting deep semantic features and training a supervised ranking model to rank answers). Obviously, the last three components aim at identifying accurate answers in a hierarchical manner. That is, we first retrieve answer documents from external Web resources, and then select out high relevant answer sentences from documents in a shallow analysis and finally re-rank answers by using supervised model based on deep semantic analysis.

The reminder of this paper is organized as follows. Section 2 overviews the whole architecture of our system. Section 3 presents the experimental results and discussion. Finally, Section 4 concludes this work.

2 System Overview

Figure 1 shows the architecture of our proposed system, which is composed of four key components, i.e., question expansion (QE), answer document retrieval (ADR), answer sentence selection (ASS), and answer re-ranking (ARR).

Given each input question, QE is to expand the content of question in three distinct ways, i.e., adding synonyms, or hypernyms of question words or the snippets returned by Search Engine (SE) into question. The expanded question is treated as a search query and submitted into a Web-based site in the ADR component. Then a set of documents which may contain the answers are returned from different external Web resources. To select out the high relevant sentences from large amount of retrieved answer documents, the ASS component is to rank these sentences using a shallow semantic analysis (the product of source weighting score and majority voting score). Finally, in ARR component, the selected top answer sentences are re-ranked by a supervised model trained on a subset of Yahoo! Web-

Scope Dataset L6[4] based on deep semantic analysis. We simply concatenated the top answers with a total length less than $1,000$ characters as the best answer and sent it back to the server in the final test stage.

Text preprocessing has been performed in all components. The *Stanford CoreNLP* (Manning et al., 2014) is used for sentence tokenization and part of speech (POS) tagging. The Natural Language Toolkit (*NLTK*)[5] is utilized for *WordNet* (WN) based stemming.

2.1 Question Expansion

Usually natural language questions can be expressed in various ways. In order to enrich the information of question, we present three methods to expand question. The first is to add the synonyms of head words in the question with the aid of WN, where the head words are all nouns and verbs present in the question. For convenience, we only used the synset of the first main sense of each noun and verb. Furthermore, in order to include the semantic abstractions of nouns in the question, the second method is to add the hypernyms of all nouns in the question. Different from the two methods which adopt the WN, the third method expands the question with the aid of Web page snippets returned by external Search Engines (SE). This is based on the observation that the returned snippets from SE are usually closely relevant to the search query. However, in order to alleviate the influence of noise introduced from snippets, we only chose the top *N* nouns from snippets ranked by *Jaccard* coefficient. Finally, we regarded the expanded question as a new query and used this new query to search relevant documents in the next component.

2.2 Answer Document Retrieval

Generally, if there is a large QA archive where each question is attached a best answer, we may retrieve the best answer from archive by finding the most similar question with the original question. However, no such large QA archive or corpus is provided by organizers. Neither is it possible for us to collect a large scale of documents and to build our own local corpus, which is labor intensive and time consuming. One possible solution is to make use of existing external resources such as search engines or QA web sites to retrieve answer documents.

In consideration of open domain of questions, we resort to external Web resources, i.e., online QA Web sites and Search Engines, to retrieve answer documents. In this work, we used two QA Web sites, i.e., Yahoo! Answer and Ask.Web and one search engine, i.e. Bing, as external resources. The YA Web site returns the list of most similar questions with the given question and does not directly provide correct answer. So we choose the top K ($K = 9$) similar questions and take the best answers as candidate answer documents. Note if there is no best answer provided to the related question, we select out the answer which is thumbed up by the most users as the best answer. Unlike YA only providing question list, the second QA site, i.e., Ask.web [6], is a question answering-focused Web site, which returns the answer snippets to the query question. The third source is general SE. The expanded questions are submitted into SE as a search query and the top *N* snippets returned by SE are regarded as candidate answer documents. In this work, we choose Bing to perform Web search. To keep a balance between the coverage and relevance of retrieved results, we choose the top *N* ($N = 10$) snippets as candidate documents both for Ask.web and Bing. Finally, these 29 ($K + 2N$) candidate answer documents retrieved from three sources are combined together and sent to the next component.

2.3 Answers Sentence Selection

Typically, these *K+2N* candidate answer documents extracted from above ADR component still contain a number of non-relevant sentences. Therefore, the ASS component is to further prune out any non-relevant sentences from candidate answer documents in a fast and simple way. To do this, we implement a two-step method, i.e., candidate answers generation, and candidate answers ranking.

For candidate answer documents retrieved from different sources, we perform different processes to generate candidate answers. Specifically, the documents returned from YA usually contain long texts, so we first split the documents into sentences by using NLTK toolkit and then choose the maximal

[4]http://webscope.sandbox.yahoo.com/catalog.php?datatype=l
[5]http://www.nltk.org/

[6]http://www.ask.com/

continuous *n* sentences as candidate answers, where the length of continuous *n* sentences are less than 250 characters. As for the documents returned from Ask.web and Bing, we used these documents as candidate answers directly due to their short text length.

In the second step, we rank the candidate answers using the product of a source weighting score and a majority voting score. The two scores are proposed based on our observations. The source weighting score of candidate answer is to consider the importance of each source where it comes from. For example, the candidate answers from YA have higher quality than those from Ask.web and Bing because the candidate answers returned from YA are the best answers of related questions while the candidate answers from Ask.web and Bing are only snippets of related web pages. Besides, since Ask.web is a QA-focused Web site and Bing is a general SE, the candidate answers from Ask.web are supposed to have higher quality than those from Bing. Therefore, we assign different source weights to the candidate answers from YA, Ask.web, and Bing as of $1/2$, $1/3$ and $1/6$ respectively. The majority voting score of candidate answer is the average of word overlaps (wo) between current candidate answer and candidate answers from different documents. To calculate the word overlaps of candidate answer $A1$ and candidate answer $A2$, a series of preprocessing procedures are performed, for example, stop words, repeated words and punctations are removed, all words are converted to their lowercase and stemming are adopted. After that, the word overlap of $A1$ and $A2$ is measured as $|A1 \cap A2|/\max(|A1|, |A2|)$, where $|A1|$ and $|A2|$ denote the number of words of $A1$ and $A2$ respectively. Finally, we select the top 10 candidate answers with the ranking scores and sent them to the next component.

2.4 Answer Re-ranking

To further identify the accurate answer, the AR-R component is to re-rank the answers by evaluating how well they answer the given question. Unlike the above ASS component which only adopts a shallow analysis on answers, in this component we build a supervised machine learning model to re-rank the answers by deeply analyzing the relatedness between answers and given questions. Four type of features are employed, i.e., the word overlap between question and answer (Word Match Features), the probability of question-to-answer transformations using a translation model (Translation Based Features), the informativeness of answer (Answer Informativeness Features), lexical semantic similarity of question and answer (Lexical Semantic Similarity Features). We remove the stop words and repeated words from question and answer before extracting these features. The details of these four types of features are described as follows.

Word Match Feature (WM): This feature records the proportions of co-occurred words between a given QA pair, which is calculated using five measures: $|Q \cap A|$, $|Q \cup A|/|Q|$, $|Q \cap A|/|A|$, $|A - Q|/|A|$, $|Q - A|/|Q|$, where $|Q|$ and $|A|$ denote the number of the words of question Q and answer A.

Translation Based Feature (TB): The above WM feature only considers the overlapped surface words between Q and A and thus it may fail to "bridge the lexical gap" between question and answer. One possible solution is to regard this task as a statistic machine translation problem between question and answer by using the IBM Model 1(Brown et al., 1993) to learn the word-to-word probabilities. Following (Xue et al., 2008; Surdeanu et al., 2011), we regarded $P(Q|A)$, i.e., the translation probability of question Q when given answer A, as a translation based feature. The probabilities are calculated as:

$$P(Q|A) = \prod_{w \in Q} P(w|A)$$

$$P(w|A) = (1 - \lambda)P_{tr}(w|A) + \lambda P_{ml}(w|C)$$

$$P_{ml}(w|A) = \sum_{a \in A} P(w|a)P_{ml}(a|A)$$

where $P(w|A)$ is the probability that the question word w is generated from answer A, λ is smoothing parameter, C is a background collection. $P_{ml}(w|C)$ is computed by maximum likelihood estimator. $P(w|a)$ denotes the translation probability from answer word $a \in A$ to question word w. We used GIZA++ Toolkit[7] to compute the probability.

Answer Informativeness Feature (AI): The AI feature measures the informativeness of a answer text. Intuitively, we assume that the answer which carries more information is more likely to be chosen

[7]http://www.statmt.org/moses/giza/GIZA++.html

as the best answer by users. We adopt the following answer information measures:

Answer Length: Number of answer words.

Answer Head Words: Number of nouns,verbs and adjectives in the answer.

Lexical Semantic Similarity Feature (LSS): Inspired by (Yih et al., 2013), we include the lexical semantic similarity features in our re-ranking model. We use the 300-dimensional version of word2vec (Mikolov et al., 2013) vectors, which is trained on part of Google News dataset (about 100 billion words). There are two ways to calculate the LSS features. One cosine similarity is calculated by summing up all word vectors in question and answer. Another is to adopt averaged pairwise cosine similarity between each word in question and answer.

To train a supervised ranking model for answer reraning, several algorithms have been experimented, for example, ranking Perceptron (Shen and Joshi, 2005), SVM-rank (Joachims, 2006), etc. In consideration of efficiency and timeliness of supervised ranking algorithm, we adopt SVM-rank in our system, which is an instance of structural SVM as a family of Support Vector Machine algorithms that model structured outputs-specifically tailored for ranking problems.

3 Experiment

3.1 Datasets

There are two datasets used in our system training experiments, i.e., Document Retrieval (DR) dataset and Answer Re-ranking (AR) dataset. The former is provided by the organizers and we use it to develop the question expansion and answer documen-t retrieval. The latter dataset is collected by ourselves from a sample of Yahoo! WebScope Dataset L6, which is used to build a supervised answer reranking model.

Document Retrieval Dataset (DR): The DR dataset is provided by the organizers including $1,000$ questions with given answers and we only choose the questions which have the best answer in YA site. Finally, we get 857 question-answer pairs used for system configuration for question expansion and answer document retrieval.

Answer Re-ranking Dataset (AR): The AR dataset is constructed by ourselves, which is extracted from a sample of Yahoo! WebScope Dataset L6 as a large collection of $4,483,032$ question-answer pairs. To construct our dataset, we implemented the following filtering steps.

Step 1**:** We only selected questions-answers pairs from eight categories mentioned in Section 1.

Step 2**:** To reduce the noise of questions and answers, we heuristically kept the questions and answers with at least five words.

Step 3**:** In order to meet the answer length requirement, we removed the answers which contain more than 250 characters.

Finally, we obtained $150,000$ question-answer pairs as our final AR dataset and 60% are used for training, 20% for development, and 20% for test.

3.2 Evaluation Metrics

To evaluate the performance of our system, several metrics are adopted.

The measure of performance of QE is embeded in ADR component. To evaluate ADR performance, we adopt the percentage of answered questions which have word overlap values larger than a certain value (e.g., 0.3, 0.4 and 0.5), where word overlap is calculated between retrieved answers from Web-based resources and the given best answers for each query question, as formulated in section 2.3. We treat ARR as a answer ranking problem and adopt *Mean Reciprocal Rank (MRR)* and *Re-ranking Precision@1* to evaluate it. The MRR is defined as the average of the reciprocal ranks of the correct answer. The Re-ranking Precision@1 is defined as the average Precision@1 over the questions set, where the Precision@1 of a query is defined as 1 if the correct answer is re-ranked into the first position, 0 otherwise. For the final test of Live QA Track, the results are judged by TREC editors firstly using 4-level s-cale as follows:

4:Excellent a significant amount of useful information and fully answers the question.

3:Good partially answers the question.

2:Fair marginally useful information.

Sources	YA			Ask.web			Bing		
Word Overlap	0.3	0.4	0.5	0.3	0.4	0.5	0.3	0.4	0.5
Raw	61.2	41.6	28.06	21.58	11.55	8.01	17.57	9.32	6.96
Synonyms	57.54	39.03	26.41	11.44	6.72	5.19	11.79	6.6	4.48
Hypernyms	57.42	38.8	26.06	18.16	9.67	7.08	12.45	7.07	4.83
Bing Snippets	50.47	34.78	23.82	15.8	8.25	5.78	9.08	4.72	2.83
Raw (YA+Ask+Bing)	68.04	45.99	31.13						

Table 1: The percentage of answered questions from different sources with different word overlap levels of ADR component on DR dataset, where *Raw*,*Synonyms*,*Hypernyms* and *Bing Snippets* are different question expansion methods.

1:Bad no useful information for the question.

Then several performance measures are built on above 4-level scale as follows:

avg-score(0-3) averaged score over all queries (transferring above 1-4 level scores to 0-3, comparing 1-level score with no-answer score).

succ@i+ number of questions with *i*+ score (*i*=1..4) divided by number of all questions.

prec@i+ number of questions with *i*+ score (*i*=2..4) divided by number of answered only questions.

3.3 Preliminary Experiments

In order to determine the optimum system configuration, we constructed preliminary experiments on the datasets of DR and AR. We did not construct experiments on ASS component individually because of the simpleness of the component.

3.3.1 Answer Document Retrieval

To compare the impacts of question expansion methods and the qualities of sources where answer documents come from, we constructed this experiment on DR dataset. Table 1 shows the percentage of answered questions from different sources with different word overlap levels (e.g., 0.3, 0.4 and 0.5) in ADR component, where *Raw* represents only using the original question words as query, *Synonyms*, *Hypernyms* and *Bing Snippets* represent the methods of using synonyms, hypernyms and bing snippets to expand question. We also perform to search answer documents by combining three sources without question expansion (i.e., *Raw (YA+Ask+Bing)*). From the results, we make the following three observations:

Feature sets	MRR(%)	P@1(%)
Random Baseline	36.85	15.08
WM Baseline	42.40	20.38
WM + TB	42.67	20.48
WM + TB + AI	44.41	23.02
WM + TB + AI + LSS	45.17	23.57

Table 2: Results of ARR component on test set of AR.

Components		Runtime
QE	Synonyms	0.06
	Hypernyms	0.05
	Bing Snippets	3.22
ADR	YA	13.45
	Ask.web	6.73
	Bing	4.24
ASS		1.83
ARR		3.69
Total		33.39

Table 3: The runtime(s) per question of four components on DR dataset.

(1) The answers from YA have higher quality than those from Ask.web and Bing on all word overlap levels.

(2) Although question expansion by synonyms, hypernyms and Bing snippets has been shown a promotion on information retrieval (Carpineto and Romano, 2012), it reduces the performance in our experiments. The word ambiguity of synonyms and hypernyms in questions and the noises introduced by snippets may cause this phenomenon.

(3) The combination of three sources increase the performance, but brings more unrelated answers.

3.3.2 Answer Re-ranking

In this experiment, we adopted *SVM-rank* with $c = 300, kernel = linear$ and the smoothing parameter $\lambda = 0.3$ for translation feature, where these parameters are tuned by the experiment on develop-

No	Run	#Answered questions	avg score (0-3)	succ@2+	succ@3+	succ@4+
1	CMUOAQA	1064	1.081	0.532	0.359	0.190
2	***ecnucs***	994	0.677	0.367	0.224	0.086
3	NUDTMDP1	1041	0.670	0.353	0.210	0.107
	avg	1007	0.467	0.262	0.146	0.060

Table 4: Results on TREC2015 Live QA test set.

ment set of AR dataset. Table 2 depicts the performance of two baselines and our proposed re-ranking model on test set of AR. The first baseline (Random) sorts the answers by random scores. The second baseline (WM) trains the model using just word match features. From Table 2, we clearly found several observations. Firstly, both AI and LSS features significantly increase both P@1 and MRR performance over two baselines. Since AI is a measure of the informativeness of answer text, this indicates that users trend to choose the answer with more information. Unlike the surface word match features which only consider the surface form, the LSS features integrate the context information and therefore the word embeddings complement the surface word match information. Secondly, the TB features do not make as much contribution on performance improvement as we expected. The possible reason may be that the question-answer alignment data contains much noise and reduces the quality of word to word translation probability.

3.3.3 Time Cost

Since there is a time limit required to return answer, i.e., less than one minute for each question, we also compared the runtime of four components under our experimental environment (e.g., the computer with Intel i7 CPU, 8GB Memory and on the Science and Technology Network of our university). Table 3 shows the runtime (second) per question of four components on DR dataset. From the table, it is interesting to find following observations. Firstly, it takes about average 33.39 seconds per question to return the answer in our system, which is acceptable. Secondly, because of no need of any online source, QE (Synonyms, Hypernyms) component and ASS component take a short time to run. Inversely, for ADR component, it takes the max proportion of time to search related documents from Web sources. Finally, the ARR component does not take much time to rank candidate answers and this owes to the practical efficiencies of pre-trained models.

3.4 System Configuration

Based on above experimental analysis, we built system configuration as the following list:

(1) Except for the QE component, we used all the other three components to build our whole system.

(2) For the ADR component, we used all three Web sources because their combination achieved the best.

(3) For ARR component, we adopted a SVM-rank algorithm ($kernel = linear, c = 300$) and used WM, TB, AI and LSS features to train the model.

3.5 Results on the test set of Live QA Track

In the final test stage, a test dataset with $1,087$ questions is provided. In above system configuration, we perform the live QA procedure on given test dataset.

Table 4 shows the results of top 3 teams (our run is *ecnucs*) and the averaged results (avg) officially released by Live QA Track organizers in TREC 2015. From Table 4, we find the following observation. Firstly, our pipeline system outperforms the averaged scores of all submitted runs with respect to above four evaluation metrics. Secondly, in our run the number of answered questions is much less than the averaged value and other teams. Although our system answered the given questions in time, 93 answers from our run has not been received by the organizer due to the out of time of network. Thirdly, the leading run, CMUOAQA from Carnegie Mellon University, performed very well compared to our run, according to all measures. However, the score of all results are still by far less than the maximum possible *avgScore* of 3.0. This is caused by the complexity of the task and it is also a challenge for human beings.

4 Conclusion

In this paper, we proposed a pipeline system to address Live QA Track of TREC 2015, i.e., QE, ADR, ASS and ARR. Although we presented three methods including expansion of synonyms, hyperonyms and Bing snippets to expand the question, they did not perform as well in the experiment as in information retrieval task. Thus our final system consists of the last three components. In ADR component, we retrieved answer documents from three different sources (i.e., YA, Ask.web, and Bing). Then in ASS component we selected candidate answers from the documents. After that we constructed a answer re-ranker which extracts four types of features (i.e., WM, TB, AI, and LSS) and adopts *SVM-rank* algorithm to train a supervised model. Finally, the top answers returned by ARR component are concatenated as the best answer on the condition that the total length of answers is less than $1,000$ characters.

For feature work, we focus on the using of question context information to improve the performance of ADR component and ARR component. Also we will explore more features to improve re-ranking performance.

Acknowledgments

This research is supported by grants from Science and Technology Commission of Shanghai Municipality under research grant no. (14DZ2260800 and 15ZR1410700) and Shanghai Collaborative Innovation Center of Trustworthy Software for Internet of Things (ZF1213).

References

Delphine Bernhard and Iryna Gurevych. 2009. Combining lexical semantic resources with question & answer archives for translation-based answer finding. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 728–736, Suntec, Singapore, August. Association for Computational Linguistics.

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Comput. Linguist.*, 19(2):263–311, June.

Claudio Carpineto and Giovanni Romano. 2012. A survey of automatic query expansion in information retrieval. *ACM Comput. Surv.*, 44(1):1:1–1:50, January.

Thorsten Joachims. 2006. Training linear svms in linear time. In Tina Eliassi-Rad, Lyle H. Ungar, Mark Craven, and Dimitrios Gunopulos, editors, *KDD*, pages 217–226. ACM.

Oleksandr Kolomiyets and Marie-Francine Moens. 2011. A survey on question answering technology from an information retrieval perspective. *Inf. Sci.*, 181(24):5412–5434, December.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

Libin Shen and Aravind K. Joshi. 2005. Ranking and reranking with perceptron. *Machine Learning*, 60(1-3):73–96.

Mihai Surdeanu, Massimiliano Ciaramita, and Hugo Zaragoza. 2011. Learning to rank answers to non-factoid questions from web collections. *Computational Linguistics*, 37(2).

Mengqiu Wang. 2006. A survey of answer extraction techniques in factoid question answering. In *Proceedings of the Human Language Technology Conference and North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*.

Xiaobing Xue, Jiwoon Jeon, and W. Bruce Croft. 2008. Retrieval models for question and answer archives. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, pages 475–482, New York, NY, USA. ACM.

Wen-tau Yih, Ming-Wei Chang, Christopher Meek, and Andrzej Pastusiak. 2013. Question answering using enhanced lexical semantic models. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1744–1753, Sofia, Bulgaria, August. Association for Computational Linguistics.