A Survey of Answer Extraction Techniques in Factoid Question Answering

Mengqiu Wang*
School of Computer Science
Carnegie Mellon University

Factoid question answering is the most widely studied task in question answering. In this paper, we survey several different techniques to answer extraction for factoid question answering, which aims at accurately pin-pointing the exact answer in retrieved documents. We compare these techniques under a unified view, from the perspective of the sources of information that each model uses, how they represent and extract these information, and the model they use for combining multiple sources of information. From our comparison and analysis, we draw conclusions of the successes and deficiencies in past approaches, and point out directions that may be interesting to future research.

1. Introduction

Question Answering (QA) is a fast-growing research area that brings together research from Information Retrieval, Information Extraction and Natural Language Processing. It is not only an interesting and challenging application, but also the techniques and methods developed from question answering inspire new ideas in many closely related areas such as document retrieval, time and named-entity expression recognition, etc. The first type of questions that research focused on was factoid questions. For example, "When was X born?", "In what year did Y take place?". The recent research trend is shifting toward more complex types of questions such as definitional questions (biographical questions such as "Who is Hilary Clinton?", and entity definition questions such as "What is DNA?"), list questions (e.g. "List the countries that have won the World Cup"), scenario-based QA (given a short description of a scenario, answer questions about relations between entities mentioned in the scenario) and why-type questions. Starting in 1999, an annual evaluation track of question answering systems has been held at the Text REtrieval Conference (TREC) (Voorhees 2001, 2003b). Following the success of TREC, in 2002 both CLEF and NTCIR workshops started multilingual and cross-lingual QA tracks, focusing on European languages and Asian languages respectively (Magnini et al. 2006; Yutaka Sasaki and Lin 2005).

The body of literature in the general field of QA has grown so large and diverse that it is infeasible to survey all areas in one paper. In this literature review we will focus on techniques developed for extracting answers of factoid questions. There are several motivations that drive us to choose this particular area.

First of all, there exist clearly defined and relatively uncontroversial evaluation standards for factoid QA. For some other types of questions such as definition questions, although there has been some emerging standards (Voorhees 2003a; Lin and Demner-Fushman 2005), evaluation

*Address: 3612A Newell Simon Hall, Carnegie Mellon University, 5000 Forbes Ave., Pittsburgh, PA, 15213 USA. E-mail: mengqiu@cs.cmu.edu

has remained somewhat controversial. In factoid QA, there is usually only one or at most a few correct answers to a given question, and the answer in most cases is a single word token or a short noun phrase. The system returns one or more ranked answer candidates for each question, and they are judged manually for correctness. In TREC and CLEF, each answer candidate is assessed as (Magnini et al. 2006):

1. correct: if neither more nor less than the information required by the question is given. The answer needed to be supported by the docid of the document(s) in which the exact answer was found, and the document has to be relevant.
2. unsupported: if either the docid was missing or wrong, or the supporting snippet did not contain the exact answer.
3. inexact: if contained less or more information than that required by the question (e.g. if question asks for year but answer contains both year and month).
4. incorrect: if the answer does not provide the required information.

The "supported" evaluation criterion is not as straight-forward as it appears to be. There is a grey area as to whether an answer should be assessed as "supported" when it matches the correct answer text and also appears in the correct document but the context in which it appears does not give enough information to support the answer. For example, suppose the answer to the question "Which US president visited Japan in 2004?" is "George Bush". Let us assume that the correct document contains two occurrences of "George Bush", as in "Shortly after George Bush won the 2004 election, he departed US for a South-East Asia tour ..." and in "On May 25th, George Bush arrived at Narita Airport and started his first visit to Japan ...". In this case, if we were shown only the first snippet, we cannot tell that George Bush is the answer to the question, and therefore the first occurrence of the string "George Bush" should not be assessed as an answer. But in current QA evaluations, as long as the answer string appeared in the relevant document, it is judged as correct.

At the end, top1 and top5 accuracies and Mean Reciprocal Rank (MRR) scores are reported for correct and correct+unsupported answers. TopN accuracy of correct answers is calculated as the number of questions in which at least one of the top N answer candidates is correct, divided by the total number of questions.
MRR is calculated as:

$$MRR = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{rank(Q_i)}$$

where N is the number of questions and $rank(Q_i)$ is the rank of the topmost correct answer of question i.

The second reason why we chose to survey factoid QA is because this task has been widely studied over the years and there exist a variety of different but interesting techniques. Last but not least, performance of the state-of-the-art factoid QA system for English is still in the low 70% range, it is our belief that there is plenty of room for further improvement. By reviewing and comparing existing techniques in answer extraction, we hope to summarize past experiences and shed light on new areas and directions for future exploration.

For readers who are interested in other types of questions and multilingual, cross-lingual QA research, the overviews of the TREC, CLEF and NTCIR QA tracks (Voorhees 2003b; Magnini et al. 2006; Yutaka Sasaki and Lin 2005) are good places to start.

Evaluation	# of Qs
TREC 8 (1999)	198
TREC 9 (2000)	692
TREC 10 (2001)	491
TREC 11 (2002)	499
TREC 12 (2003)	413
TREC 13 (2004)	231
NTCIR5 (2005)	200
CLEF 2006	200

Table 1
Summary of the number of test questions in recent TREC, CLEF and NTCIR tracks

The rest of this article is organized as the following: in Section 2, we will give a brief overview of QA systems. We will introduce the three main modules usually found in QA systems – question analysis, document retrieval and answer extraction, and explain their functionalities. Then in Section 3, we will focus on the answer extraction module, and closely examine several answer extraction techniques in the literature. In Section 4, we will draw connections between the answer extraction techniques that we discuss here and some of the recent advances in two other related areas – Textual Entailment and Zero-anaphora Resolution. In Section 5 we will point out some future directions that we think will be interesting to future research. Finally we will give a conclusion in 6.

2. A Brief Overview of QA systems

In a QA evaluation track, each system is given a document collection, a set of training questions, a gold-standard answer set, and a set of testing questions. The document collection consists of newswire articles collected from one or multiple news agencies over usually a couple of years. In most cases, these collections contain several million documents. The training set consists of questions taken from past years' test sets plus some additional ones. Both NTCIR and CLEF give a pre-defined list of question types. In CLEF, the types are: Person, Organization, Location, Time, Measure and Others; in NTCIR, the types are: Person, Organization, Location, Time, Date, Money, Percent, Numex (Measure), Artifact. TREC does not have a pre-defined list, but TREC questions include all of the above types and more fine-grained types (e.g. TREC has many questions on "how did X die?", and it is commonly categorized as MANNER_OF_DEATH). The distribution of these types varies from year to year, but there are usually more Person, Organization, Location and Date (Time) questions than other types. It is worth noting that to the best of our knowledge, all QA systems in the literature use supervised-learning methods to train their models, and therefore the amount of training data has a significant impact on the system's performance. In recent TREC QA tracks, the number of factoid training questions has increased to a couple of thousand. The number in NTCIR and CLEF has been significantly smaller, usually only a few hundred. It could be a potential reason why the best systems in NTCIR and CLEF do not have nearly as high accuracy as the systems in TREC. For testing, we have summarized the different test set sizes in recent TREC, NTCIR and CLEF tracks in Table 1.

A typical QA system usually employs a pipeline architecture that chains together three main modules:

- Question analysis module: this module processes the question, analyzes the question type, and produces a set of keywords for retrieval. Depending on the retrieval and answer extraction strategies, some question analysis module also perform syntactic and semantic analysis of the questions, such as dependency parsing and semantic role labeling.
- Document or passage retrieval module: this module takes the keywords produced by the question analysis module, and uses some search engine to perform document or passage retrieval. Two of the most popular search engine used by many QA systems are Indri (Metzler and Croft 2004) and Lucene [1].
- Answer extraction module: given the top N relevant documents or passages from the retrieval module, the answer extraction module performs detailed analysis and pin-points the answer to the question. Usually answer extraction module produces a list of answer candidates and ranks them according to some scoring functions.

In some systems, there are additional modules that provide extra functionalities. For example, query expansion using external resources (e.g. the Web) is often performed since questions can be quite short (e.g. When did Hitler die?), and thus only by taking keywords from the question may not yield enough contextual information for effective retrieval. Another commonly employed technique is called answer justification or answer projection (Sun et al. 2005b). The answer justification module either takes the answer produced by the system and tries to verify it using resources such as the Web, or it uses external databases or other knowledge sources to generate the answer, and "project" it back into the collection to find the right documents.

3. Answer Extraction by Structural Information Matching

In this section, we will closely examine several answer extraction techniques. At a high abstraction level, different answer extraction approaches can be described in a general way. They find answers by first recovering latent or hidden information on the question side and on the answer sentence side, and then they locate answers by some kind of structure matching.

Under this generic view, we will focus our discussion around the following questions:

- What sources of information are useful for finding answers?
- How do we obtain and represent useful information?
- How do we combine multiple information sources in making a unified decision?

3.1 Identifying useful information for extracting answers

Before we examine specific methods and models for extracting answers, it is worth spending some time to think about what kind of information is useful in helping us finding answers. Early TREC systems focus on exploiting surface text information by either hand-crafting patterns or automatically acquiring surface text patterns (Soubbotin and Soubbotin 2001; Ravichandran and Hovy 2002). There are a few shortcomings of this approach. First of all, manually constructed surface patterns usually give good precision but poor recall. Although automatic learning of these patterns explicitly address this issue (Ravichandran and Hovy 2002), low recall is still identified as the major cause of bad performance in many pattern-based approaches (Xu, Licuanan, and

[1] http://lucene.apache.org/

Weischedel 2003). One solution to this problem is to combine patterns with other statistical methods. For example, Ravichandran, Ittycheriah and Roukos (2003) first extracted a set of 22,353 patterns using the approach described in (Ravichandran and Hovy 2002), then they used a maximum-entropy classifier to learn the appropriate weights of these patterns, on a training set that has 4,900 questions. Another problem as discussed by Ravichandran and Hovy (2002) is that surface patterns cannot capture long-distance dependencies. This problem can be addressed by recovering syntactic structures in the answer sentences, and enhance the patterns with such linguistic constructs (Peng et al. 2005).

Another source of information that is used by almost all question answering systems is named-entity (NE). The idea is that factoid questions fall into several distinctive types, such as "location", "date", "person", etc. Assuming that we can recognize the question type correctly, then the potential answer candidates can be limited down to a few NE types that correspond to the question type. Intuitively, if the question is asking for a date, then an answer string that is identified to be a location type named-entity is not likely to be the correct answer. However, it is important to bear in mind that neither question type classification nor NE recognition are perfect in the real world. Therefore, although systems can benefit from having fewer answer candidates to consider, using question type and named-entity to rule out answer candidates deterministically (Lee et al. 2005; Yang et al. 2003) can be harmful when classification and recognition errors occur. We will survey models that use NE information in combination with other sources of information in Section 3.3.

The aforementioned two types of information – sentence surface text patterns and answer candidate NE type – both come from the answer sentence side. The only information we have extracted from the question side is the question type, which is used for selecting patterns and NE types for matching. The structural information in the question sentence, which is not available in inputs in many other tasks (e.g. ad-hoc document retrieval), has not yet been fully-utilized. Recognizing this unique input source information, there have been many recent work on finding ways to better utilize structural information, and we will review them next.

3.2 Structural Information Extraction and Representation

Only recently have we started seeing work demonstrating significant performance gains using syntax in answer extraction (Shen and Klakow 2006; Sun et al. 2005a; Cui et al. 2005). As Katz and Lin (2003) pointed out, most early experiments that tried to bring in syntactic or semantic features to IR and QA showed little performance gain, and often resulted in performance degradation (Litkowski 1999; Attardi et al. 2001). (Katz and Lin 2003) suggested that one should use syntactic relations selectively only when it is helpful. In their paper, they identified two specific linguistic phenomena, namely "semantic symmetry" and "ambiguous modification", and derived ternary expressions such as "bird eats snake" and "largest adjmod planet" from dependency parse trees. On a small hand-selected test set that consists of 16 questions, they showed that using ternary expressions for semantic indexing and matching achieved a precision of 0.84 ± 0.11 while keyword based matching achieved only 0.29 ± 0.11 in precision. Another case of selectively using syntactic features is the work by Li (2003). In her work, six syntactically motivated heuristic factors were employed:

1. the size of the longest phrase in the question matched in the answer sentence.
2. the surface distance between answer candidate and the main verb in the answer sentence.

3. for "PERSON" type questions, a Boolean feature indicates whether the syntactic relationship (either "passive" or "active") between the answer candidate and the main verb is the same as the relationship in the question.
4. for "LOCATION" type questions, check the presence of possessive construct.
5. for "PERSON" type questions, check if both the answer candidate and all question keywords fall inside a adjective noun phrase.

Li (2003) tested these factors on the TREC-9 questions, and showed a 7.8% improvement over the baseline systems. A set of linguistically motivated syntactic patterns were also used in IBM's QA system for TREC-10 (Ittycheriah, Franz, and Roukos 2001). The three types of linguistic constructs listed in their paper are:

1. Is-relationship: a feature that activates when the answer candidate is the subject or object of a "be" verb.
2. Apposition: a feature that activates when the answer candidate is followed or preceded immediately by a comma
3. Subject-Verb/Verb-Object: a feature that activates when the answer candidate is the subject or object of a non-stopword verb.

A very similar set of linguistic features were also used in (Peng et al. 2005) for definitional QA.

To use syntactic and semantic structural and relational information, there is also a question of what granularity level we should represent these relations. If we were to perform exact matching of the syntactic relations between question sentence and answer sentence, we should have directly compared the syntactic sub-tree that contain keywords found in both sentence and also the question word and answer candidate word. But as often noticed, such a direct comparison results in severe data sparsity. Researchers have taken different approaches to reduce the data sparseness problem by transforming the syntactic or semantic relation.

Sun et al. (2005b) computed semantic similarities between the predicate-argument structures in the question and in the answer sentence. Instead of comparing each argument position separately, they ignored the difference between argument types, and used Jaccard coefficient to measure similarity between the sets of words that formed the arguments in the question and in the answer sentence.

In an earlier work, (Sun et al. 2005a; Cui et al. 2005) computed similarity between two dependency parse paths by first decomposing the paths into individual links, and then find all possible alignments between link fragments, and compute the mutual information between any pair of dependency link. The formula they gave was:

$$Sim(P_Q, P_A) = \frac{\epsilon}{1+len(P_Q)^{len(P_A)}} \, argmax_{\, all\_possible\_alignment} \sum_{i,j} MI(Rel_i^Q, Rel_j^A)$$

and the mutual information between a pair of dependency links was defined as:

$$MI(Rel_0, Rel_1) = \log \frac{\sum \alpha \times \delta(Rel_0, Rel_1)}{f_Q(Rel_0) \times f_A(Rel_1)}$$

where $f_Q(Rel)$ and $f_A(Rel)$ represent the count of Rel in the question and answer sentence, respectively. $\delta(Rel_0, Rel_1)$ is 1 when Rel_0 and Rel_1 appear in a question answer sentence pair, α is inversely proportional to the number of relation pairs for which δ is 1. Note that the claimed that this is a Mutual Information measure is a little far-stretched. The δ term is not a marginal joint probability of Rel_0 and Rel_1, and the f_Q and f_A terms are measured over different distributions (questions and answer sentences). This heuristic is closer in form

to Pointwise Mutual information (PMI), which is defined over values of random variables (if we think of a link as a random variable and the link types as values).

Instead of iterating over all possible alignment to compute the argmax, Shen and Klakow (2006) adapted the Dynamic Time Warping algorithm (Rabiner, Rosenberg, and Levinson 1978) for finding the best alignment with variable segment length but keeps the monotonicity of the aligned sequences. They used the same PMI-like heuristic between dependency links as a measure of similarity, but the final scoring of a pair of dependency path also takes into account phrase mapping scores of the words at the ends of the path. The ability to use stemming, synonym expansion and other form alternation techniques for approximate matching of words gained them more robustness and higher recall.

In an earlier work by Shen et al. (2005), a tree kernel function was used to compute the similarity between two dependency parse trees. They used a constituent parser to derive the syntactic parse tree, and then transform it into a dependency parse tree using hand-written rules. For each node in the dependency tree, they incorporated POS tag identity features, syntactic tag identity features, orthographic features (e.g. is the word capitalized?) and role features (is this word the answer candidate, or a question keyword?). Partial matching using tree kernels showed a good increase in matching coverage (from 24.32% coverage using exact matching to 49.94% coverage using tree kernels), but the accuracy increase was marginal (a 3% relative increase in top1 accuracy).

In decomposing a dependency path and sum over the similarities between individual links, we lose the order information encoded along the path. Bouma et al. (2005) presented a method that tries to find equivalence relations between dependency paths directly in a QA system for Dutch language. They implemented about 13 equivalence rules that captures dependency relations that expresses the same semantic meaning but with slight variation in forms.

If Bouma et al. (2005)'s approach represents the fine-grained end of the granularity spectrum in using dependency link structure, then the method proposed in (Punyakanok, Roth, and Yih 2004) is analogous to the coarse-grained end. They measured the similarity between a question and an answer sentence by computing the tree edit-distance of the two dependency parse trees. Compared with a baseline system using only bag-of-word features, their method showed a 39% relative increase in finding the correct document. Note that their system only finds the documents that contain answer, but does not pin-point the answer string. Tanev et al. (2005) borrowed the idea and used tree edit-distance as similarity measure for find answer strings in an Italian to English Cross-lingual QA system. However, their result was a somewhat disheartening. They only achieved an accuracy of 5.8% using the tree edit-distance approach.

3.3 Models for combining multiple sources of information in answer scoring

Nearly all QA systems use evidence from multiple sources to decide what the best answer is for a given question. Therefore inevitably, we need to define some model or some scoring function to combine different sources of evidence. Lee et al.(2005) first decided the NE type of answer candidates based on the question type, and then ranked answer candidates based on four types of evidence:

1. the count of NE types that occurred in the question occurring in the answer sentence, normalized by the number of NE types occurred in the question
2. the count of constraints other than NE type constraint that occurred in the question occurring in the answer sentence, normalized by the number of constraints occurred in the question

3. a pre-set score when the answer candidate string contains the identified question focus string
4. a pre-set score the answer candidate string appear in the answer sentence next to a term that contains the question focus string

Each listed condition will give a normalized score if satisfied, and the final scoring function is the sum of these scores.

Another example of ad-hoc feature weighting is the TREC-9 system described in (Ittycheriah et al. 2001). In the answer selection module, the following distance metrics are computed for each 3 sentences window:

1. Matching words: the sum of TFIDF scores of the question words appear in the 3 sentence window
2. Thesaurus match: the sum of TFIDF scores of the question words whose thesaurus matching by WordNet appear in the 3 sentence window
3. Mis-matching words: the sum of TFIDF scores of the question words that do not appear in the 3 sentence window
4. Dispersion: the number of words in the answer sentence that occur between matched question words
5. Cluster words: the number of words that occur next to each other in both the answer sentence window and the question

Having computed these distance metrics, they used an ad-hoc scoring function which was not detailed in the paper to combine these metrics.

Linear sum of feature values with manually selected weights can be found in many other systems (Li 2003; Bouma et al. 2005). Beyond the ad-hoc nature of these approaches that rely on manually assigned weights, there are some more severe drawbacks. First of all, having to manually assign weights means that the number of features the system can consider is very limited. One can hardly hope that weights manually assigned to a handful of features would remain meaningful and optimal, not to mention hundreds of features as were employed in some systems. Secondly, a system tuned in this way is not likely to give robust performance across different collections and test sets.

In the TREC-10 IBM system (Ittycheriah, Franz, and Roukos 2001) a maximum-entropy model was used for automatically learning features weights and combining multiple features. In the maximum entropy model, the probability of the event that an answer candidate A is correct given question Q is defined as:

$$p(x = C|A, Q) = \frac{\exp \vec{f}(x = C, A, Q) \cdot \vec{\theta}}{\exp \vec{f}(x = C, A, Q) \cdot \vec{\theta} + \exp \vec{f}(x = W, A, Q) \cdot \vec{\theta}}$$

Here x is the judgment of A. It is a binary random variable that can take on values of either C(correct) or W(wrong). $\vec{f}$ is a feature vector and $\vec{\theta}$ are the feature weights. Inputs to their model were the question and the answer string. The model only has one binary random variable and therefore inference is very straight-forward. A total of 31 features were used in the system. They were mainly identity features over named-entity types and certain pre-defined syntactic patterns. They obtained a 34.5% improvement over the TREC-9 system.

The same MaxEnt model was subsequently adopted by many other systems and showed consistent improvement over ad-hoc scoring functions (Shen and Klakow 2006; Shen, Kruijff, and Klakow 2005). When Shen and Klakow (Shen and Klakow 2006) decomposed dependency paths into link segments and computed correlation of each aligned link pair using PMI-like heuristic, they did not sum over the link pairs as Cui et al. (2005) did. Instead they used each link type pair as a feature in the MaxEnt model, and learned different weights for different link pairs. (Echihabi et al. 2005) used a MaxEnt model to combine the predictions of three different answer extraction modules: a knowledge base extraction module, a pattern-based module and a statistical based module that models answer extraction using a noisy-channel paradigm. They used 48 different types of features in the MaxEnt model, and these features can be classified into four main categories:

- Component-specific: the score and rank output produced by each individual answer extraction module
- Redundancy-specific: the count of answer candidate in the collection, and also the log and square root of the count
- Qtarget-specific: the combination of some classes of question type with the score of individual extraction module
- Blatant-error-specific: a set of features that aim at eliminating obvious errors (such as answers do not contain pronouns)

The final top1 score after MaxEnt re-ranking is 47.21%, while the highest individual answer extraction module only achieved 35.83% (the knowledge-based extraction module). Those results suggest that automatically learning feature weights is a very effective technique in combining multiple sources of evidence.

Shen and Klakow (2006) replicated results of some of the key earlier work in answer extraction. They compared the performance of different approaches under the same experimental set up, which serves as a nice summary of some of the techniques we have discussed. The comparison is shown in Table 2.

4. Connections with related areas

All the systems and techniques we have surveyed so far reside within the QA research domain. However, we noticed that advances in some other research areas bear close proximity to techniques we surveyed here.

One of the most evident examples is the work in Textual Entailment. The task of textual entailment is to recognize semantic entailment for any given pair of sentences. For example, “no cat talks” entails “my cat doesn’t talk”, while “my cat talks” does not entail “all cats talk”. (Haghighi, Ng, and Manning 2005) proposed an approach by representing each sentence in the pair as a directed acyclic graph, where the nodes are lexical words and edges are dependency relations between words. They perform graph matching via node and link alignment, and judge entailment based on the amount of matched content. At a higher abstraction level, this is very similar to Shen and Klakow (2006)’s work. Given the question sentence and potential answer sentence as the pair, Shen and Klakow also created two graphs represented by dependency parse structure. Then instead of taking the full graph, they took the subgraphs that only contain links that have the answer candidate node or question word node as one end, and perform graph matching.

Method	Highlights	MRR	Top1	Top5
Density (similar to Li 2003)	answer candidate is chosen according to surface distance to question phrases	0.45	0.36	0.56
SynPattern(Shen, Kruijff & Klakow 2005)	automatically extracted syntactic relation patterns; partial dep-parse tree matching using tree kernel	0.56	0.53	0.60
StrictMatch (similar to Tanev et al. 2005)	strict dependency relation matching; correlation score is 1 if the two relations match exactly, 0 otherwise	0.57	0.49	0.67
ApprMatch (Cui et al. 2005)	approximate dependency relation matching by decomposing dep-path into link segments; using PMI-like heuristics to estimate link pair correlation; sum over link pairs to get candidate scores	0.60	0.53	0.70
CorME (Shen & Klakow 2006)	approximate dependency relation matching as in ApprMatch; use MaxEnt model to assign different weights to different link pairs; incorporate word/phrase mapping score into path correlation measure	0.67	0.62	0.74

Table 2
Comparison of different answer extraction techniques using TREC 99-03 questions for training and TREC 04 questions for testing. The experiments assume gold-standard answer typing and only documents that contain correct answers were used as input.

Perhaps even more similar to Shen and Klakow (2006) is the work done by MacCartney et al. (2006). Instead of judging entailment direcly based on graph matching, they treated graph matching as a first step. Using graph matching results as features, and together with other global features such as polarity and antonym features that would have not easily fit into the first step model. They designed a logistic regression classifier as the second step to make the final entailment judgment. Shen and Klakow (2006) also took the graph matching results as features, and used a maximum entropy classifier to produce the final ranking of answer candidates.

Another example is (Iida, Inui, and Matsumoto 2006) on using syntactic patterns in Zero-anaphora Resolution. The definition of zero-anaphora as given by Iida et al. is “a gap in a sentence that has an anaphoric function similar to a pro-form (e.g. pronoun)”. Examples of zero-anaphora can be found in their paper and thus are omitted here. There are lots of details in their paper specific to the zero-anaphora resolution task, but what is of interest to us is their model for intra-sentential zero-anaphora resolution (intra-sentential means the antecedent occurs within the same sentence as the zero-pronoun), called “tournament model”. The tournament model assigns a probability to each word in the sentence (except the zero-pronoun word) for being the antecedent and ranks them. This is similar to the way Shen and Klakow (2006) and Cui et al. (2005) ranks each word in candidate answer sentence for being the actual answer. To rank each candidate antecedent word, Iida et al. produced the dependency parse tree of the sentence, and extracted dependency path between the zero-pronoun word and the candidate antecedent word. This dependency path is then fed to a boosting classifier, in which each decision stump is

associated with a subtree, much similar to syntactic relation patterns with individually learned scores in (Shen and Klakow 2006). Some other hand-selected heuristics were also used as features in the boosting classifier in (Iida, Inui, and Matsumoto 2006), again similar to the way orthographic features and NE features were used in (Shen and Klakow 2006; Shen, Kruijff, and Klakow 2005).

5. Future directions

From the systems and techniques we reviewed in Section 3, we observe a trend that more syntax is coming into this line of research. But there are a few prominent deficiencies in the current approaches. The real challenge in answer extraction is to recognize syntactic and semantic variations when a question is expressed in the answer text. For example, in the question "how did X die" and answer sentence "Y killed X", both syntactic structure and semantic structure changed when we used word "kill" (a killing event) to express "die" (a dying event). The way Shen and Klakow (2006) and Sun et al. (2005a) recognized syntactic equivalence is by first decomposing dependency path into individual segments, then used alignment algorithms (DTW in (Shen and Klakow 2006) and permutation (Sun et al. 2005a)) that do not preserve the linguistic information encoded in the sequence, and finally sum up the similarity measure of each segment based on PMI-like heuristic. One may argue that this is an over-simplistic and too coarse-grained model. Also, although (Shen and Klakow 2006) used a MaxEnt model to learn different weights for different link types, it only combines different types of dependency features without any structural constraint. We think that there is definitely interesting work to be done in learning the structural equivalence and transformation using more expressive models. But for more powerful models, we will inevitably run into data sparsity problems. It is not clear at this point whether the data we have now is sufficient of not, and there is also a question that whether we are making full use of the data we have. Another related direction is to go beyond syntax and explore more semantic level information. We hope to see more interesting work along these lines in the near future.

6. Conclusion

In this paper, we surveyed a variety of techniques in answer extraction for factoid question answering. We have seen a movement from earlier simple models that do not explore much of the syntactic or semantic structural information to more recent models to more recent models that measures question and answer similarities based on dependency structures and uses maximum-entropy models to automatically learn weights to combine multiple sources of information. We analyzed the similarities and connections with related work in Textual Entailment and Zero-anaphora Resolution domains. Lastly, as a result of this survey, we have found that the full potential of using dependency structures to recognize answer and question equivalence has not yet been fully explored. The models that current QA systems employ for finding structural matching have remained somewhat ad-hoc. Clever models and ways of using syntactic and semantic information in answer extraction are still open to future research.

References

Attardi, Giuseppe, Antonio Cisternino, Francesco Formica, Maria Simi, Alessandro Tommasi, Ellen M. Voorhees, and D. K. Harman. 2001. Selectively using relations to improve precision in question answering. In *Proceedings of the 10th Text REtrieval Conference (TREC-10)*, Gaithersburgh, MD, USA, November.

Bouma, Gosse, Jori Mur, Gertjan van Noord, Lonneke van der Plas, and Jõrg Tiedemann. 2005. Question answering for dutch using dependency relations. In *Proceedings of the Cross-Language Evaluation*

Forum 2005 workshop (CLEF), Vienna, Austria, September.
Cui, Hang, Renxu Sun, Keya Li, Min-Yen Kan, and Tat-Seng Chua. 2005. Question answering passage retrieval using dependency relations. In *Proceedings of the 28th ACM-SIGIR International Conference on Research and Development in Information Retrieval*, Salvador, Brazil, August.
Echihabi, Abdessamad, Ulf Hermjakob, Eduard Hovy, Daniel Marcu, Eric Melz, and Deepak Ravichandran, 2005. *Advances in Textual Question Answering*, chapter How to Select an Answer String? Kluwer.
Haghighi, Aria, Andrew Y. Ng, and Chris Manning. 2005. Robust textual inference via graph matching. In *Proceedings of the Human Language Technology Conference/Empirical Methods in Natural Language Processing (HLT-EMNLP)*.
Iida, Ryu, Kentaro Inui, and Yuji Matsumoto. 2006. Exploiting syntactic patterns as clues in zero-anaphora resolution. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING-ACL)*, Sydney, Australia, July.
Ittycheriah, Abraham, Martin Franz, and Salim Roukos. 2001. Ibm's statistical question answering system – trec-10. In *Proceedings of the 10th Text REtrieval Conference (TREC-10)*, Gaithersburgh, MD, USA, November.
Ittycheriah, Abraham, Martin Franz, Wei-Jing Zhu, Adwait Ratnaparkhi, and Richard J. Mammone. 2001. Question answering using maximum entropy components. In *Proceedings of the Second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*.
Katz, Boris and Jimmy Lin. 2003. Selectively using relations to improve precision in question answering. In *Proceedings of the EACL-2003 Workshop on Natural Language Processing for Question Answering*, April.
Lee, Cheng-Wei, Cheng-Wei Shih, Min-Yuh Day, Tzong-Han Tsai, Tian-Jian Jiang, Chia-Wei Wu, Cheng-Lung Sung, Yu-Ren Chen, Shih-Hung Wu, and Wen-Lian Hsu. 2005. Asqa: Academia sinica question answering system for ntcir-5 clqa. In *Proceedings of the 5th NTCIR Workshop Meeting (NTCIR-5)*, Tokyo, Japan, December.
Li, Xiaoyan. 2003. Syntactic features in question answering. In *Proceedings of the 26th ACM-SIGIR International Conference on Research and Development in Information Retrieval*, Toronto, Canada, August.
Lin, Jimmy and Dina Demner-Fushman. 2005. Automatically evaluating answers to definition questions. Technical report, MIT CSAIL.
Litkowski, Kenneth C. 1999. Question-answering using semantic relation triples. In *Proceedings of the 8th Text REtrieval Conference (TREC-8)*, Gaithersburgh, MD, USA, November.
MacCartney, Bill, Trond Grenager, Marie-Catherine de Marneffe, Daniel Cer, and Christopher D. Manning. 2006. Learning to recognize features of valid textual entailments. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*.
Magnini, Bernardo, Danilo Giampiccolo, Pamela Forner, Christelle Ayache, Petya Osenova, Anselmo Pe nas, Valentin Jijkoun, Bogdan Sacaleanu, Paulo Rocha, and Richard Sutcliffe. 2006. Overview of the clef 2006 multilingual question answering track. In *Proceedings of the Cross-Language Evaluation Forum 2006 workshop (CLEF)*, Alicante, Spain, September.
Metzler, Donald and Bruce W. Croft. 2004. Combining the language model and inference network approaches to retrieval. *Information Processing and Management Special Issue on Bayesian Networks and Information Retrieval*, 40(5).
Peng, Fuchun, Ralph Weischedel, Ana Licuanan, and Jinxi Xu. 2005. Combining deep linguistics analysis and surface pattern learning: A hybrid approach to chinese definitional question answering. In *Proceedings of the Human Language Technology Conference and North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*, Vancouver, Canada, October.
Punyakanok, Vasin, Dan Roth, and Wen-Tau Yih. 2004. Mapping dependencies trees: An application to question answering. In *Proceedings of the 8th International Symposium on Artificial Intelligence and Mathematics*, Fort Lauderdale, FL, USA.
Rabiner, Lawrence R., Aaron E. Rosenberg, and Stephen E. Levinson. 1978. Considerations in dynamic time warping algorithms for discrete word recognition. *The Journal of the Acoustical Society of America*, 63(S1):S79.
Ravichandran, Deepak and Eduard Hovy. 2002. Learning surface text patterns for a question answering system. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, Philadelphia, PA, USA, July.

Ravichandran, Deepak, Abharam Ittycheriah, and Salim Roukos. 2003. Automatic derivation of surface text patterns for a maximum entropy based question answering system. In *Proceedings of the Human Language Technology Conference and North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*, Edmonton, Canada, May.

Shen, Dan and Dietrich Klakow. 2006. Exploring correlation of dependency relation paths for answer extraction. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING-ACL)*, Sydney, Australia, July.

Shen, Dan, Geert-Jan M. Kruijff, and Dietrich Klakow. 2005. Exploring syntactic relation patterns for question answering. In *Proceedings of the Second International Joint Conference on Natural Language Processing (IJCNLP)*, Jeju Island, Republic of Korea, October.

Soubbotin, Martin M. and Sergei M. Soubbotin. 2001. Patterns for potential answer expressions as clues to the right answers. In *Proceedings of the 10th Text REtrieval Conference (TREC-10)*, Gaithersburgh, MD, USA, November.

Sun, Renxu, Hang Cui, Keya Li, Min-Yen Kan, and Tat-Seng Chua. 2005a. Dependency relation matching for answer selection. In *Proceedings of the 28th ACM-SIGIR International Conference on Research and Development in Information Retrieval*, Salvador, Brazil, August.

Sun, Renxu, Jing Jiang, Yee Fan Tan, Hang Cui, Tat-Seng Chua, and Min-Yen Kan. 2005b. Using syntactic and semantic relation analysis in question answering. In *Proceedings of the 14th Text Retrieval Conference (TREC-14)*, Gaithersburg, MD, USA, November.

Tanev, Hristo, Milen Kouylekov, Bernardo Magnini, Matteo Negri, and Kiril Simov. 2005. Exploiting linguistic indices and syntactic structures for multilingual question answering: Itc-irst at clef 2005. In *Proceedings of the Cross-Language Evaluation Forum 2005 workshop (CLEF)*, Vienna, Austria, September.

Voorhees, Ellen M. 2001. Overview of the trec 2001 question answering track. In *Proceedings of the 10th Text REtrieval Conference (TREC-10)*, Gaithersburgh, MD, USA, November.

Voorhees, Ellen M. 2003a. Evaluating answers to definition questions. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (HLT-NAACL)*, Edmonton, Canada, May.

Voorhees, Ellen M. 2003b. Overview of the trec 2003 question answering track. In *Proceedings of the 12th Text REtrieval Conference (TREC-12)*, Gaithersburgh, MD, USA, November.

Xu, Jinxi, Ana Licuanan, and Ralph Weischedel. 2003. Trec2003 qa at bbn: Answering definitional questions. In *Proceedings of the 12th Text REtrieval Conference (TREC-12)*, Gaithersburgh, MD, USA, November.

Yang, Hui, Hang Cui, Min-Yen Kan, Mstislav Maslennikov, Long Qiu, and Tat-Seng Chua. 2003. Qualifier in trec-12 qa main task. In *Proceedings of the 12th Text REtrieval Conference (TREC-12)*, Gaithersburgh, MD, USA, November.

Yutaka Sasaki, Hsin-Hsi Chen, Kuang-Hua Chen and Chuan-Jie Lin. 2005. Overview of the ntcir-5 cross-lingual question answering task (clqa1). In *Proceedings of the 5th NTCIR Workshop Meeting (NTCIR-5)*, Tokyo, Japan, December.