Updated Version – July 29, 2013

Question Answering Using Enhanced Lexical Semantic Models*

Wen-tau Yih Ming-Wei Chang Christopher Meek Andrzej Pastusiak
Microsoft Research
Redmond, WA 98052, USA
{scottyih,minchang,meek,andrzejp}@microsoft.com

Abstract

In this paper, we study the *answer sentence selection* problem for question answering. Unlike previous work, which primarily leverages syntactic analysis through dependency tree matching, we focus on improving the performance using models of lexical semantic resources. Experiments show that our systems can be consistently and significantly improved with rich lexical semantic information, regardless of the choice of learning algorithms. When evaluated on a benchmark dataset, the MAP and MRR scores are increased by 8 to 10 points, compared to one of our baseline systems using only surface-form matching. Moreover, our best system also outperforms pervious work that makes use of the dependency tree structure by a wide margin.

1 Introduction

Open-domain question answering (QA), which fulfills a user's information need by outputting direct answers to natural language queries, is a challenging but important problem (Etzioni, 2011). State-of-the-art QA systems often implement a complicated pipeline architecture, consisting of question analysis, document or passage retrieval, answer selection and verification (Ferrucci, 2012; Moldovan et al., 2003). In this paper, we focus on one of the key subtasks – *answer sentence selection*. Given a question and a set of candidate sentences, the task is to choose the *correct* sentence that contains the exact answer and can sufficiently support the answer choice. For instance, although both of the following sentences contain the answer "Jack Lemmon" to the question "Who won the best actor Oscar in 1973?" only the first sentence is correct.

A1: Jack Lemmon won the Academy Award for Best Actor for Save the Tiger (1973).

A2: Oscar winner Kevin Spacey said that Jack Lemmon is remembered as always making time for other people.

One of the benefits of answer sentence selection is that the output can be provided directly to the user. Instead of outputting only the answer, returning the whole sentence often adds more value as the user can easily verify the correctness without reading a lengthy document.

Answer sentence selection can be naturally reduced to a semantic text matching problem. Conceptually, we would like to measure how close the question and sentence can be *matched* semantically. Due to the variety of word choices and inherent ambiguities in natural languages, bag-of-words approaches with simple surface-form word matching tend to produce brittle results with poor prediction accuracy (Bilotti et al., 2007). As a result, researchers put more emphasis on exploiting both the syntactic and semantic structure in questions/sentences. Representative examples include methods based on deeper semantic analysis (Shen and Lapata, 2007; Moldovan et al., 2007) and on tree edit-distance (Punyakanok et al., 2004; Heilman and Smith, 2010) and quasi-synchronous grammar (Wang et al., 2007) that match the dependency parse trees of questions and sentences. However, such approaches often require more computational resources. In addition to applying a syntactic or semantic parser during run-time, finding the best matching between structured representations of sentences is not trivial. For example, the computational complexity of tree matching is $O(V^2L^4)$, where V is the number of nodes and L is the maximum depth (Tai, 1979).

*This is an updated version that is different from the one in the proceedings. The main differences are the experimental results in Tables 2 and 3, which are now made directly comparable to previous work. See Foonote 7 for detail.

Instead of focusing on the high-level semantic representation, we turn our attention in this work to improving the shallow semantic component, *lexical semantics*. We formulate answer selection as a semantic matching problem with a latent word-alignment structure as in (Chang et al., 2010) and conduct a series of experimental studies on leveraging recently proposed lexical semantic models. Our main contributions in this work are two key findings. First, by incorporating the abundant information from a variety of lexical semantic models, the answer selection system can be enhanced substantially, regardless of the choice of learning algorithms and settings. Second, while the latent alignment model performs better than unstructured models, the difference diminishes after adding the enhanced lexical semantics information. This may suggest that compared to introducing complex structured constraints, incorporating shallow semantic information is both more effective and computationally inexpensive in improving the performance, at least for the specific word alignment model tested in this work.

The rest of the paper is structured as follows. We first survey the related work in Sec. 2. Sec. 3 defines the problem of answer sentence selection, along with the high-level description of our solution. The enhanced lexical semantic models and the learning frameworks we explore are presented in Sec. 4 and Sec. 5, respectively. Our experimental results on a benchmark QA dataset is shown in Sec. 6. Finally, Sec. 7 concludes the paper.

2 Related Work

While the task of question answering has a long history dated back to the dawn of artificial intelligence, early systems like STUDENT (Winograd, 1977) and LUNAR (Woods, 1973) are typically designed to demonstrate natural language understanding for a small and specific domain. The Text REtrieval Conference (TREC) Question Answering Track was arguably the first large-scale evaluation of open-domain question answering (Voorhees and Tice, 2000). The task is designed in an information retrieval oriented setting. Given a factoid question along with a collection of documents, a system is required to return the exact answer, along with the document that supports the answer. In contrast, the Jeopardy! TV quiz show provides another open-domain question answering setting, in which IBM's Watson system famously beat the two highest ranked players (Ferrucci, 2012). Questions in this game are presented in a statement form and the system needs to identify the true question and to give the exact answer. A short sentence or paragraph to justify the answer is not required in either TREC-QA or Jeopardy!

As any QA system can virtually be decomposed into two major high-level components, *retrieval* and *selection* (Echihabi and Marcu, 2003), the answer selection problem is clearly critical. Limiting the scope of an answer to a sentence is first highlighted by Wang et al. (2007), who argued that it was more informative to present the whole sentence instead of a short answer to users.

Observing the limitations of the bag-of-words models, Wang et al. (2007) proposed a syntax-driven approach, where each pair of question and sentence are matched by their dependency trees. The mapping is learned by a generative probabilistic model based on a Quasi-synchronous Grammar formulation (Smith and Eisner, 2006). This approach was later improved by Wang and Manning (2010) with a tree-edit CRF model that learns the latent alignment structure. In contrast, general tree matching methods based on tree-edit distance have been first proposed by Punyakanok et al. (2004) for a similar answer selection task. Heilman and Smith (2010) proposed a discriminative approach that first computes a tree kernel function between the dependency trees of the question and candidate sentence, and then learns a classifier based on the tree-edit features extracted. Incorporating additional features along with the original tree-edit features, Yao et al. (2013) improved this method using a logistic regression classifier.

Although lexical semantic information derived from WordNet has been used in some of these approaches, the research has mainly focused on modeling the mapping between the syntactic structures of questions and sentences, produced from syntactic analysis. The potential improvement from enhanced lexical semantic models seems to have been deliberately overlooked.[1]

3 Problem Definition

We consider the answer selection problem in a supervised learning setting. For a set of questions $\{q_1, \cdots, q_m\}$, each question q_i is associated

[1] For example, Heilman and Smith (2010) emphasized that "The tree edit model, which does not use lexical semantics knowledge, produced the best result reported to date."

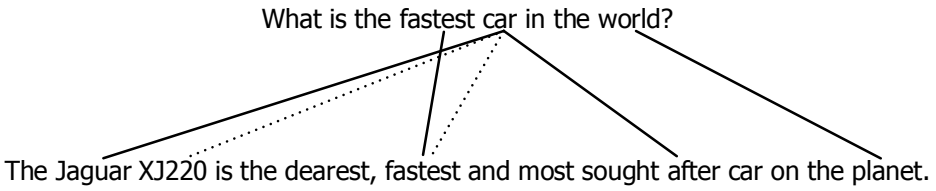

Figure 1: An example pair of question and answer sentence, adapted from (Harabagiu and Moldovan, 2001). Words connected by solid lines are clear synonyms or hyponym/hypernym; words with weaker semantic association are linked by dashed lines.

with a list of labeled candidate answer sentences $\{(y_{i_1}, s_{i_1}), (y_{i_1}, s_{i_2}), \cdots, (y_{i_n}, s_{i_n})\}$, where $y_{i_j} = 1$ indicates that sentence s_{i_j} is a correct answer to question q_i, and 0 otherwise. Using this labeled data, our goal is to learn a probabilistic classifier to predict the label of a new, unseen pair of question and sentence.

Fundamentally, what the classifier predicts is whether the sentence "matches" the question semantically. In other words, does s have the answer that satisfies the semantic constraints provided in the question? Without representing the question and sentence in logic or syntactic trees, we take a *word-alignment* view for solving this problem. We assume that there is an underlying structure h that describes how q and s can be associated through the relations of the words in them. Figure 1 illustrates this setting using a revised example from (Harabagiu and Moldovan, 2001). In this figure, words connected by solid lines are clear synonyms or hyponym/hypernym; words connected by dashed lines indicate that they are weakly related. With this alignment structure, features like the degree of mapping or whether all the content words in the question can be mapped to some words in the sentence can be extracted and help improve the classifier. Notice that the structure representation in terms of word-alignment is fairly general. For instance, if we assume a naive complete bipartite matching, then effectively it reduces to the simple bag-of-words model.

Typically, the "ideal" alignment structure is not available in the data, and previous work exploited mostly syntactic analysis (e.g., dependency trees) to reveal the latent mapping structure. In this work, we focus our study on leveraging the low-level semantic cues from recently proposed lexical semantic models. As will be shown in our experiments, such information not only improves a latent structure learning method, but also makes a simple bipartite matching approach extremely strong.[2]

4 Lexical Semantic Models

In this section, we introduce the lexical semantic models we adopt for solving the semantic matching problem in answer selection. To go beyond the simple, limited surface-form matching, we aim to pair words that are semantically related, specifically measured by models of word relations including *synonymy/antonymy*, *hypernymy/hyponymy* (the Is-A relation) and general *semantic word similarity*.

4.1 Synonymy and Antonymy

Among all the word relations, *synonymy* is perhaps the most basic one and needs to be handled reliably. Although sets of synonyms can be easily found in thesauri or WordNet synsets, such resources typically cover only strict synonyms. When comparing two words, it is more useful to estimate the *degree* of synonymy as well. For instance, *ship* and *boat* are not strict synonyms because a ship is usually viewed as a large boat. Knowing that two words are somewhat synonymous could be valuable in determining whether they should be mapped.

In order to estimate the degree of synonymy, we leverage a recently proposed polarity-inducing latent semantic analysis (PILSA) model (Yih et al., 2012). Given a thesaurus, the model first constructs a *signed* d-by-n co-occurrence matrix W, where d is the number of word groups and n is

[2] Proposed by an anonymous reviewer, one justification of this word-alignment approach, where syntactic analysis plays a less important role, is that there are often few sensible combinations of words. For instance, knowing only the set of words {"car", "fastest", "world"}, one may still guess correctly the question "What is the fastest car in the world?"

the size of the vocabulary. Each row consists of a group of synonyms and antonyms of a particular sense and each column represents a unique word. Values of the elements in each row vector are the TFIDF values of the corresponding words in this group. The notion of *polarity* is then induced by making the values of words in the antonym groups negative, and the matrix is generalized by a low-rank approximation derived by singular-value decomposition (SVD) in the end. This design has an intriguing property – if the cosine score of two column vectors are positive, then the two corresponding words tend to be synonymous; if it's negative, then the two words are antonymous. The degree is measured by the absolute value.

Following the setting described in (Yih et al., 2012), we construct a PILSA model based on the Encarta thesaurus and enhance it with a discriminative projection matrix training method. The estimated degrees of both synonymy and antonymy are used our experiments.[3]

4.2 Hypernymy and Hyponymy

The *Class-Inclusion* or *Is-A* relation is commonly observed between words in questions and answer sentences. For example, to correctly answer the question "What color is Saturn?", it is crucial that the selected sentence mentions a specific kind of color, as in "Saturn is a giant gas planet with brown and beige clouds." Another example is "Who wrote Moonlight Sonata?", where *compose* in "Ludwig van Beethoven composed the Moonlight Sonata in 1801." is one kind of *write*.

Traditionally, WordNet taxonomy is the linguistic resource for identifying hypernyms and hyponyms, applied broadly to many NLP problems. However, WordNet has a number of well-known limitations including its rather limited or skewed concept distribution and the lack of the coverage of the Is-A relation (Song et al., 2011). For instance, when a word refers to a named entity, the particular sense and meaning is often not encoded. As a result, relations such as "Apple" *is-a* "company" and "Jaguar" *is-a* "car" cannot be found in WordNet. Similar to the case in synonymy, the Is-A relation defined in WordNet does not provide a native, real-valued degree of the relation, which can only be roughly approximated using the number of links on the taxonomy path connecting two concepts (Resnik, 1995).

In order to remedy these issues, we augment WordNet with the Is-A relations found in Probase (Wu et al., 2012). Probase is a knowledge base that establishes connections between 2.7 million concepts, discovered automatically by applying Hearst patterns (Hearst, 1992) to 1.68 billion Web pages. Its abundant concept coverage distinguishes it from other knowledge bases, such as Freebase (Bollacker et al., 2008) and WikiTaxonomy (Ponzetto and Strube, 2007). Based on the frequency of term co-occurrences, each Is-A relation from Probase is associated with a probability value, indicating the degree of the relation.

We verified the quality of Probase Is-A relations using a recently proposed SemEval task of relational similarity (Jurgens et al., 2012) in a companion paper (Zhila et al., 2013), where a subset of the data is to measure the degree of two words having a *class-inclusion* relation. Probase's prediction correlates well with the human annotations and achieves a high Spearman's rank correlation coefficient score, $\rho = 0.619$. In comparison, the previous best system (Rink and Harabagiu, 2012) in the task only reaches $\rho = 0.233$. These appealing qualities make Probase a robust lexical semantic model for hypernymy/hyponymy.

4.3 Semantic Word Similarity

The third lexical semantic model we introduce targets a general notion of *word similarity*. Unlike synonymy and hyponymy, word similarity is only loosely defined when two words can be associated by some implicit relation.[4] The general word similarity model can be viewed as a "back-off" solution when the exact lexical relation (e.g., *part-whole* and *attribute*) is not available or cannot be accurately detected.

Among various word similarity models (Agirre et al., 2009; Reisinger and Mooney, 2010; Gabrilovich and Markovitch, 2007; Radinsky et al., 2011), the vector space models (VSMs) based on the idea of *distributional similarity* (Turney and Pantel, 2010) are often used as the core component. Inspired by (Yih and Qazvinian, 2012), which argues the importance of incorporating heterogeneous vector space models for measuring word similarity, we leverage three different VSMs

[3] Mapping two antonyms may be desired if one of them is in the scope of negation (Morante and Blanco, 2012; Blanco and Moldovan, 2011). However, we do not attempt to resolve the negation scope in this work.

[4] Instead of making the distinction, *word similarity* here refers to the larger set of relations commonly covered by *word relatedness* (Budanitsky and Hirst, 2006).

in this work: Wiki term-vectors, recurrent neural network language model (RNNLM) and a concept vector space model learned from click-through data. Semantic word similarity is estimated using the cosine score of the corresponding word vectors in these VSMs.

Contextual term-vectors created using the Wikipedia corpus have shown to perform well on measuring word similarity (Reisinger and Mooney, 2010). Following the setting suggested by Yih and Qazvinian (2012), we create term-vectors representing about 1 million words by aggregating terms within a window of $[-10, 10]$ of each occurrence of the target word. The vectors are further refined by applying the same vocabulary and feature pruning techniques.

A recurrent neural network language model (Mikolov et al., 2010) aims to estimate the probability of observing a word given its preceding context. However, one by-product of this model is the word embedding learned in its hidden-layer, which can be viewed as capturing the word meaning in some latent, conceptual space. As a result, vectors of related words tend to be close to each other. For this word similarity model, we take a 640-dimensional version of RNNLM vectors, which is trained using the Broadcast News corpus of 320M words.[5]

The final word relatedness model is a projection model learned from the click-through data of a commercial search engine (Gao et al., 2011). Unlike the previous two models, which are created or trained using a text corpus, the input for this model is pairs of aggregated queries and titles of pages users click. This parallel data is used to train a projection matrix for creating the mapping between words in queries and documents based on user feedback, using a Siamese neural network (Yih et al., 2011). Each row vector of this matrix is the dense vector representation of the corresponding word in the vocabulary. Perhaps due to its unique information source, we found this particular word embedding seems to complement the other two VSMs and tends to improve the word similarity measure in general.

5 Learning QA Matching Models

In this section, we investigate the effectiveness of various learning models for matching questions and sentences, including the *bag-of-words* setting and the framework of learning latent structures.

5.1 Bag-of-Words Model

The bag-of-words model treats each question and sentence as an unstructured bag of words. When comparing a question with a sentence, the model first matches each word in the question to each word in the sentence. It then aggregates features extracted from each of these word pairs to represent the whole question/sentence pair. A binary classifier can be trained easily using any machine learning algorithm in this standard supervised learning setting.

Formally, let $x = (q, s)$ be a pair of question q and sentence s. Let $V_q = \{w_{q_1}, w_{q_2}, \cdots, w_{q_m}\}$ and $V_s = \{w_{s_1}, w_{s_2}, \cdots, w_{s_n}\}$ be the sets of words in q and s, respectively. Given a word pair (w_q, w_s), where $w_q \in V_q$ and $w_s \in V_s$, feature functions $\phi_1, \cdots, \phi_d$ map it to a d-dimensional real-valued feature vector.

We consider two aggregate functions for defining the feature vectors of the whole question/answer pair: *average* and *max*.

$$\Phi_{avg_j}(q, s) = \frac{1}{mn} \sum_{\substack{w_q \in V_q \\ w_s \in V_s}} \phi_j(w_q, w_s) \quad (1)$$

$$\Phi_{max_j}(q, s) = \max_{\substack{w_q \in V_q \\ w_s \in V_s}} \phi_j(w_q, w_s) \quad (2)$$

Together, each question/sentence pair is represented by a $2d$-dimensional feature vector.

We tested two learning algorithms in this setting: *logistic regression* and *boosted decision trees* (Friedman, 2001). The former is the log-linear model widely used in the NLP community and the latter is a robust non-linear learning algorithm that has shown great empirical performance.

The bag-of-words model does not require an additional inference stage as in structured learning, which may be computationally expensive. Nevertheless, its lack of structure information could limit the expressiveness of the model and make it difficult to capture more sophisticated semantics in the sentences. To address this concern, we investigate models of learning latent structures next.

5.2 Learning Latent Structures

One obvious issue of the bag-of-words model is that words in the unrelated part of the sentence may still be paired with words in the question, which introduces noise to the final feature vector.

[5] http://www.fit.vutbr.cz/~imikolov/rnnlm/

This is observed in many question/sentence pairs, such as the one below.

Q: Which was the first movie that James Dean was in?

A: James Dean, who began as an actor on TV dramas, didn't make his screen debut until 1951's "Fixed Bayonet."

While this sentence correctly answers the question, the fact that James Dean began as a TV actor is unrelated to the question. As a result, an "ideal" word alignment structure should not link words in this clause to those in the question. In order to leverage the latent structured information, we adapt a recently proposed framework of *learning constrained latent representations* (LCLR) (Chang et al., 2010). LCLR can be viewed as a variant of Latent-SVM (Felzenszwalb et al., 2009) with different learning formulations and a general inference framework. The idea of LCLR is to replace the decision function of a standard linear model $\theta^T \phi(x)$ with

$$\arg\max_h \theta^T \phi(x, h), \tag{3}$$

where θ represents the weight vector and h represents the latent variables.

In this answer selection task, $x = (q, s)$ represents a pair of question q and candidate sentence s. As described in Sec. 3, h refers to the latent alignment between q and s. $\phi_j(x, h)$ is the sum of $\phi_j(w_q, w_s)$ for each aligned pair of words (w_q,w_s) according to h, divided by the number of words in the question q. The intuition behinds Eq. (3) is: candidate sentence s correctly answers question q if and only if the decision can be supported by the *best* alignment h.

The objective function of LCLR is defined as:

$$\begin{aligned} \min_\theta \quad & \frac{1}{2}||\theta||^2 + C \sum_i \xi_i^2 \\ \text{s.t.} \quad & \xi_i \geq 1 - y_i \max_h \theta^T \phi(x, h) \end{aligned}$$

Note that the alignment is latent, so LCLR uses the binary labels in the training data as feedback to find the alignment for each example.

The computational difficulty of the inference problem (Eq. (3)) largely depends on the constraints we enforce in the alignment. Complicated constraints may result in a difficult inference problem, which can be solved by integer linear programming (Roth and Yih, 2007). In this work, we considered several sets of constraints for the alignment task, including a two-layer phrase/word alignment structure, but found that they generally performed the same. Therefore, we chose the many-to-one alignment[6], where inference can be solved exactly using a simple greedy algorithm.

6 Experiments

We present our experimental results in this section by first introducing the data and evaluation metrics, followed by the results of existing systems and some baseline methods. We then show the positive impact of adding information of word relations from various lexical semantics models, with some discussion on the limitation of the word-matching approach.

6.1 Data & Evaluation Metrics

The answer selection dataset we used was originally created by Wang et al. (2007) based on the QA track of past Text REtrieval Conferences (TREC-QA). Questions in this dataset are short factoid questions, such as "What is Crips' gang color?" In average, each question is associated with approximately 33 answer candidate sentences. A pair of question and sentence is judged positive if the sentence contains the exact answer key *and* can provide sufficient context as supporting evidence.

The training set of the data contains manually labeled 5,919 question/sentence pairs from TREC 8-12. The development and testing sets are both from TREC 13, which contain 1,374 and 1,866 pairs, respectively. The task is treated as a sentence ranking problem for each question and thus evaluated in Mean Average Precision (MAP) and Mean Reciprocal Rank (MRR), using the official TREC evaluation program. Following (Wang et al., 2007), candidate sentences with more than 40 words are removed from evaluation, as well as questions with only positive or negative candidate sentences.[7]

[6] Each word in the question needs to be linked to a word in the sentence. Each word in the sentence can be linked to zero or multiple words in the question.

[7] Among the 72 questions in the test set, 4 of them would always be treated answered incorrectly by the evaluation script used by previous work. This makes the upper bound of both MAP and MRR become 0.9444 instead of 1. In order to make our results directly comparable to theirs, we use this setting in our experiments.

System	MAP	MRR
Wang et al. (2007)	0.6029	0.6852
Heilman and Smith (2010)	0.6091	0.6917
Wang and Manning (2010)	0.5951	0.6951
Yao et al. (2013)	0.6307	0.7477

Table 1: Test set results of existing methods, taken from Table 3 of (Yao et al., 2013).

	Dev		Test	
Baseline	MAP	MRR	MAP	MRR
Random	0.4567	0.5318	0.3965	0.4929
Word Cnt	0.5897	0.6597	0.5707	0.6266
Wgt Word Cnt	0.6493	0.7261	0.5975	0.6515

Table 2: Results of three baseline methods.

6.2 Baseline Methods

Several systems have been proposed and tested using this dataset. Wang et al. (2007) presented a generative probabilistic model based on a Quasi-synchronous Grammar formulation and was later improved by Wang and Manning (2010) with a tree-edit CRF model that learns the latent alignment structure. In contrast, Heilman and Smith (2010) proposed a discriminative approach that first computes a tree kernel function between the dependency trees of the question and candidate sentence, and then learns a classifier based on the tree-edit features extracted. This method was later enhanced by Yao et al. (2013) with additional features. Table 1 summarizes their results on the test set. All these systems incorporated lexical semantics features derived from WordNet and named entity features.

In order to further estimate the difficulty of this task and dataset, we tested three simple baselines. The first is *random scoring*, which simply assigns a random score to each candidate sentence. The second one, *word count*, is to count how many words in the question that also occur in the answer sentence, after removing stopwords[8], and lowering the case. Finally, the last baseline method, *weighted word count*, is basically the same as identical word matching, but the count is re-weighted using the IDF value of the question word. This is similar to the BM25 ranking function (Robertson et al., 1995). The results of these three methods are shown in Table 1.

[8]We used a list of 101 stopwords, including articles, pronouns and punctuation.

6.3 Incorporating Rich Lexical Semantics

We test the effectiveness of adding rich lexical semantics information by creating examples of different feature sets. As described in Sec. 5, all the features are based on the properties of the pair of a word from the question and a word from the candidate sentence. Stopwords are first removed from both questions and sentences and all words are lower-cased. Features used in the experiments can be categorized into six types: identical word matching (I), lemma matching (L), WordNet (WN), enhanced Lexical Semantics (LS), Named Entity matching (NE) and Answer type checking (Ans). Inspired by the *weighted word count* baseline, all features except (Ans) are weighted by the IDF value of the question word. In other words, the IDF values help decide the importance of word pairs to the model.

Staring from the our baseline model, *weighted word count*, the identical word matching (I) feature checks whether the pair of words are the same. Instead of checking the surface form of the word, lemma matching (L) verifies whether the two words have the same lemma form. Arguably the most common source of word relations, WordNet (WN) provides the primitive features of whether two words could belong to the same synset in WordNet, could be antonyms and whether one is a hypernym of the other. Alternatively, the enhanced lexical semantics (LS) features apply the models described in Sec. 4 to the word pair and use their estimated degree of synonymy, antonymy, hyponymy and semantic relatedness as features. Named entity matching (NE) checks whether two words are individually part of some named entities with the same type. Finally, when the question word is the WH-word, we check if the paired word belongs to some phrase that has the correct answer type using simple rules, such as "*Who* should link to a word that is part of a named entity of type *Person*." We created examples in each round of experiments by augmenting these features in the same order, and observed how adding different information helped improve the model performance.

Three models are included in our study. For the unstructured, bag-of-words setting, we tested logistic regression (LR) and boosted decision trees (BDT). As mentioned in Sec. 5, the features for the whole question/sentence pair are the *average* and *max* of features of all the word pairs. For

	LR		BDT		LCLR	
Feature set	MAP	MRR	MAP	MRR	MAP	MRR
1: I	0.5975	0.6515	0.5768	0.6343	0.6073	0.6724
2: I+L	0.6189	0.6668	0.5940	0.6368	0.6259	0.6714
3: I+L+WN	0.6483	0.7150	0.6243	0.6895	0.6761	0.7365
4: I+L+WN+LS	0.6784	0.7552	0.6967	0.7899	0.7071	0.7675
5: All	0.6818	0.7616	0.6940	0.7894	0.7092	0.7700

Table 3: Test results of various models and feature groups. Logistic regression (LR) and boosted decision trees (BDT) are the two unstructured models. LCLR is the algorithm for learning latent structures. Feature groups are identical word matching (I), lemma matching (L), WordNet (WN) and enhanced Lexical Semantics (LS). *All* includes these four plus Named Entity matching (NE) and Answer type checking (Ans).

the structured-output setting, we used the framework of learning constrained latent representation (LCLR) and required that each question word needed to be mapped to a word in the sentence. Hyper-parameters are selected using the ones that achieve the best MAP score on the development set. Results of these models and feature sets are presented in Table 3.

We make two observations from the results. First, while incorporating more information of the word pairs in general helps, it is clear that mapping words beyond surface-form matching with the help of WordNet (Line #3 vs. #2) is important. Moreover, when richer information from other lexical semantic models is available, the performance can be further improved (Line #4 vs. #3). Overall, by simply incorporating more information on word relations, we gain approximately 10 points in both MAP and MRR compared to surface-form matching (Line #4 vs. #2), consistently across all three models. However, adding more information like named entity matching and answer type verification does not seem to help much (Line #5 vs. #4). Second, while the structured-output model usually performs better than both unstructured models (LCLR vs. LR & BDT), the performance gain diminishes after more information of word pairs is available (e.g., Lines #4 and #5).

6.4 Limitation of Word Matching Models

Although we have demonstrated the benefits of leveraging various lexical semantic models to help find the association between words, the problem of question answering is nevertheless far from solved using the word-based approach. Examining the output of the LCLR model with all features on the development set, we found that there were three main sources of errors, including uncovered or inaccurate entity relations, the lack of robust question analysis and the need of high-level semantic representation and inference. While the first two can be improved by, say, using a better named entity tagger, incorporating other knowledge bases and building a question classifier, how to solve the third problem is tricky. Below is an example:

Q: In what film is Gordon Gekko the main character?

A: He received a best actor Oscar in 1987 for his role as Gordon Gekko in "Wall Street".

This is a correct answer sentence because "winning a best actor Oscar" implies that the role Gordon Gekko is the main character. It is hard to believe that a pure word-matching model would be able to solve this type of "inferential question answering" problem.

7 Conclusions

In this paper, we present an experimental study on solving the answer selection problem using enhanced lexical semantic models. Following the word-alignment paradigm, we find that the rich lexical semantic information improves the models consistently in the unstructured bag-of-words setting and also in the framework of learning latent structures. Another interesting finding we have is that while the latent structured model, LCLR, performs better than the other two unstructured models, the difference diminishes after more information, including the enhanced lexical semantic knowledge and answer type verification, has been incorporated. This may suggest that adding

shallow semantic information is more effective than introducing complex structured constraints, at least for the specific word alignment model we experimented with in this work.

In the future, we plan to explore several directions. First, although we focus on improving TREC-style open-domain question answering in this work, we would like to apply the proposed technology to other QA scenarios, such as community-based QA (CQA). For instance, the sentence matching technique can help map a given question to some questions in an existing CQA database (e.g., Yahoo! Answers). Moreover, the answer sentence selection scheme could also be useful in extracting the most related sentences from the answer text to form a summary answer. Second, because the task of answer sentence selection is very similar to paraphrase detection (Dolan et al., 2004) and recognizing textual entailment (Dagan et al., 2006), we would like to investigate whether systems for these tasks can be improved by incorporating enhanced lexical semantic knowledge as well. Finally, we would like to improve our system for the answer sentence selection task and for question answering in general. In addition to following the directions suggested by the error analysis presented in Sec. 6.4, we plan to use logic-like semantic representations of questions and sentences, and explore the role of lexical semantics for handling questions that require inference.

Acknowledgments

We are grateful to Mengqiu Wang for providing the dataset and helping clarify some issues in the experiments. We also thank Chris Burges and Hoifung Poon for valuable discussion and the anonymous reviewers for their useful comments.

References

E. Agirre, E. Alfonseca, K. Hall, J. Kravalova, M. Paşca and A. Soroa. 2009. A study on similarity and relatedness using distributional and WordNet-based approaches. In *Proceedings of NAACL*, pages 19–27.

M. Bilotti, P. Ogilvie, J. Callan, and E. Nyberg. 2007. Structured retrieval for question answering. In *Proceedings of SIGIR*, pages 351–358.

E. Blanco and D. Moldovan. 2011. Semantic representation of negation using focus detection. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2011)*.

K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *ACM Conference on Management of Data (SIGMOD)*, pages 1247–1250.

A. Budanitsky and G. Hirst. 2006. Evaluating WordNet-based measures of lexical semantic relatedness. *Computational Linguistics*, 32:13–47, March.

M. Chang, D. Goldwasser, D. Roth, and V. Srikumar. 2010. Discriminative learning over constrained latent representations. In *Proceedings of NAACL*.

I. Dagan, O. Glickman, and B. Magnini, editors. 2006. *The PASCAL Recognising Textual Entailment Challenge*, volume 3944. Springer-Verlag, Berlin.

W. Dolan, C. Quirk, and C. Brockett. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of COLING*.

A. Echihabi and D. Marcu. 2003. A noisy-channel approach to question answering. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 16–23.

Oren Etzioni. 2011. Search needs a shake-up. *Nature*, 476(7358):25–26.

P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. 2009. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 99(1).

D. Ferrucci. 2012. Introduction to "This is Watson". *IBM Journal of Research and Development*, 56(3.4):1–1.

J. Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, 29(5):1189–1232.

E. Gabrilovich and S. Markovitch. 2007. Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In *AAAI Conference on Artificial Intelligence (AAAI)*.

J. Gao, K. Toutanova, and W. Yih. 2011. Clickthrough-based latent semantic models for web search. In *Proceedings of SIGIR*, pages 675–684.

S. Harabagiu and D. Moldovan. 2001. Open-domain textual question answering. Tutorial of NAACL-2001.

M. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of COLING*, pages 539–545.

M. Heilman and N. Smith. 2010. Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1011–1019.

D. Jurgens, S. Mohammad, P. Turney, and K. Holyoak. 2012. SemEval-2012 Task 2: Measuring degrees of relational similarity. In *Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 356–364.

T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur. 2010. Recurrent neural network based language model. In *Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 1045–1048.

D. Moldovan, M. Paşca, S. Harabagiu, and M. Surdeanu. 2003. Performance issues and error analysis in an open-domain question answering system. *ACM Transactions on Information Systems (TOIS)*, 21(2):133–154.

D. Moldovan, C. Clark, S. Harabagiu, and D. Hodges. 2007. COGEX: A semantically and contextually enriched logic prover for question answering. *Journal of Applied Logic*, 5(1):49–69.

R. Morante and E. Blanco. 2012. *SEM 2012 shared task: Resolving the scope and focus of negation. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics*, pages 265–274.

S. Ponzetto and M. Strube. 2007. Deriving a large scale taxonomy from wikipedia. In *AAAI Conference on Artificial Intelligence (AAAI)*.

V. Punyakanok, D. Roth, and W. Yih. 2004. Mapping dependencies trees: An application to question answering. In *International Symposium on Artificial Intelligence and Mathematics (AI & Math)*.

K. Radinsky, E. Agichtein, E. Gabrilovich, and S. Markovitch. 2011. A word at a time: computing word relatedness using temporal semantic analysis. In *WWW '11*, pages 337–346.

J. Reisinger and R. Mooney. 2010. Multi-prototype vector-space models of word meaning. In *Proceedings of NAACL*.

P. Resnik. 1995. Using information content to evaluate semantic similarity in a taxonomy. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

B. Rink and S. Harabagiu. 2012. UTD: Determining relational similarity using lexical patterns. In *Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 413–418.

S. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford. 1995. Okapi at TREC-3. In *Text REtrieval Conference (TREC)*, pages 109–109.

D. Roth and W. Yih. 2007. Global inference for entity and relation identification via a linear programming formulation. In Lise Getoor and Ben Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press.

D. Shen and M. Lapata. 2007. Using semantic roles to improve question answering. In *Proceedings of EMNLP-CoNLL*, pages 12–21.

D. Smith and J. Eisner. 2006. Quasi-synchronous grammars: Alignment by soft projection of syntactic dependencies. In *Proceedings of the HLT-NAACL Workshop on Statistical Machine Translation*, pages 23–30.

Y. Song, H. Wang, Z. Wang, H. Li, and W. Chen. 2011. Short text conceptualization using a probabilistic knowledgebase. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2330–2336.

K. Tai. 1979. The tree-to-tree correction problem. *J. ACM*, 26(3):422–433, July.

P. Turney and P. Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37(1):141–188.

E. Voorhees and D. Tice. 2000. Building a question answering test collection. In *Proceedings of SIGIR*, pages 200–207.

M. Wang and C. Manning. 2010. Probabilistic tree-edit models with structured latent variables for textual entailment and question answering. In *Proceedings of COLING*.

M. Wang, N. Smith, and T. Mitamura. 2007. What is the Jeopardy model? A quasi-synchronous grammar for QA. In *Proceedings of EMNLP-CoNLL*.

T. Winograd. 1977. Five lectures on artificial intelligence. In A. Zampolli, editor, *Linguistic Structures Processing*, pages 399–520. North Holland.

W. Woods. 1973. Progress in natural language understanding: An application to lunar geology. In *Proceedings of the National Computer Conference and Exposition (AFIPS)*, pages 441–450.

W. Wu, H. Li, H. Wang, and K. Zhu. 2012. Probase: a probabilistic taxonomy for text understanding. In *ACM Conference on Management of Data (SIGMOD)*, pages 481–492.

X. Yao, B. Van Durme, C. Callison-Burch, and P. Clark. 2013. Answer extraction as sequence tagging with tree edit distance. In *Proceedings of HLT-NAACL*.

W. Yih and V. Qazvinian. 2012. Measuring word relatedness using heterogeneous vector space models. In *Proceedings of NAACL-HLT 2012*, pages 616–620.

W. Yih, K. Toutanova, J. Platt, and C. Meek. 2011. Learning discriminative projections for text similarity measures. In *ACL Conference on Natural Language Learning (CoNLL)*, pages 247–256.

W. Yih, G. Zweig, and J. Platt. 2012. Polarity inducing latent semantic analysis. In *Proceedings of EMNLP-CoNLL*, pages 1212–1222.

A. Zhila, W. Yih, C. Meek, G. Zweig, and T. Mikolov. 2013. Combining heterogeneous models for measuring relational similarity. In *Proceedings of HLT-NAACL*.