Learning Adaptable Patterns for Passage Reranking

Aliaksei Severyn(1) and **Massimo Nicosia**(1) and **Alessandro Moschitti**[1,2]
(1)DISI, University of Trento, 38123 Povo (TN), Italy
{severyn,m.nicosia,moschitti}@disi.unitn.it
(2)QCRI, Qatar Foundation, 5825 Doha, Qatar
amoschitti@qf.org.qa

Abstract

This paper proposes passage reranking models that (i) do not require manual feature engineering and (ii) greatly preserve accuracy, when changing application domain. Their main characteristic is the use of relational semantic structures representing questions and their answer passages. The relations are established using information from automatic classifiers, i.e., question category (QC) and focus classifiers (FC) and Named Entity Recognizers (NER). This way (i) effective structural relational patterns can be automatically learned with kernel machines; and (ii) structures are more invariant w.r.t. different domains, thus fostering adaptability.

1 Introduction

A critical issue for implementing Question Answering (QA) systems is the need of designing answer search and extraction modules specific to the target application domain. These modules encode handcrafted rules based on syntactic patterns that detect the relations between a question and its candidate answers in text fragments. Such rules are triggered when patterns in the question and the passage are found. For example, given a question[1]:

What is Mark Twain's real name?

and a relevant passage, e.g., retrieved by a search engine:

Samuel Langhorne Clemens, better known as Mark Twain.

the QA engineers typically apply a syntactic parser to obtain the parse trees of the above two sentences, from which, they extract rules like:

if *the pattern* "`What is` NP_2's ADJ `name`" *is in the question* **and** *the pattern* "NP_1 `better known as` NP_2" *is in the answer passage* **then** *associate the passage with a high score*[2].

Machine learning has made easier the task of QA engineering by enabling the automatic learning of answer extraction modules. However, new features and training data have to be typically developed when porting a QA system from a domain to another. This is even more critical considering that effective features tend to be as much complex and similar as traditional handcrafted rules.

To reduce the burden of manual feature engineering for QA, we proposed structural models based on kernel methods, (Moschitti et al., 2007; Moschitti and Quarteroni, 2008; Moschitti, 2008) with passages limited to one sentence. Their main idea is to: (i) generate question and passage pairs, where the text passages are retrieved by a search engine; (ii) assuming those containing the correct answer as positive instance pairs and all the others as negative ones; (iii) represent such pairs with two syntactic trees; and (ii) learn to rank answer passages by means of structural kernels applied to two trees. This enables the automatic engineering of structural/lexical semantic patterns.

More recently, we showed that such models can be learned for passages constituted by multiple sentences on very large-scale (Severyn and Moschitti, 2012). For this purpose, we designed a shallow syntactic representation of entire paragraphs by also improving the pair representation using relational tags.

In this paper, we firstly use our model in (Severyn and Moschitti, 2012) as the current baseline and compare it with more advanced structures derived from dependency trees.

[1]We use this question/answer pair from TREC QA as a running example in the rest of the paper.

[2]If the point-wise answer is needed rather than the entire passage, the rule could end with: **returns** NP_1

Proceedings of the Seventeenth Conference on Computational Natural Language Learning, pages 75–83,
Sofia, Bulgaria, August 8-9 2013.

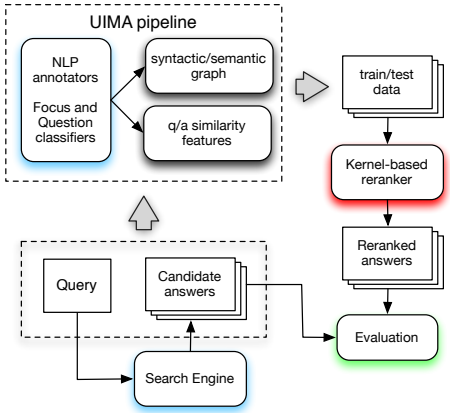

Figure 1: Kernel-based Answer Passage Reranking system

Secondly, we enrich the semantic representation of QA pairs with the categorical information provided by automatic classifiers, i.e., question category (QC) and focus classifiers (FC) and Named Entity Recognizers (NER). FC determines the constituent of the question to be linked to the named entities (NEs) of the answer passage. The target NEs are selected based on their compatibility with the category of the question, e.g., an NE of type PERSON is compatible with a category of a question asking for a human (HUM).

Thirdly, we tested our models in a cross-domain setting since we believe that: (i) the enriched representation is supposed to increase the capability of learning effective structural relational patterns through kernel machines; and (ii) such structural features are more invariant with respect to different domains, fostering their adaptability.

Finally, the results show that our methods greatly improve on IR baseline, e.g., BM25, by 40%, and on previous reranking models, up to 10%. In particular, differently from our previous work such models can effectively use NERs and the output of different automatic modules.

The rest of the paper is organized as follows, Sec. 2 describes our kernel-based reranker, Sec. 3 illustrates our question/answer relational structures; Sec. 5 briefly describes the feature vectors, and finally Sec. 6 reports the experimental results on TREC and Answerbag data.

2 Learning to rank with kernels

2.1 QA system

Our QA system is based on a rather simple reranking framework as displayed in Figure 1: given a query question a search engine retrieves a list of candidate passages ranked by their relevancy. Various NLP components embedded in the pipeline as UIMA[3] annotators are then used to analyze each question together with its candidate answers, e.g., part-of-speech tagging, chunking, named entity recognition, constituency and dependency parsing, etc. These annotations are then used to produce structural models (described in Sec. 3), which are further used by a question focus detector and question type classifiers to establish relational links for a given question/answer pair. The resulting tree pairs are then used to train a kernel-based reranker, which outputs the model to refine the initial ordering of the retrieved answer passages.

2.2 Tree kernels

We use tree structures as our base representation since they provide sufficient flexibility in representation and allow for easier feature extraction than, for example, graph structures. We rely on the Partial Tree Kernel (PTK) (Moschitti, 2006) to handle feature engineering over the structural representations. The choice of PTK is motivated by its ability to generate rich feature spaces over both constituency and dependency parse trees. It generalizes a subset tree kernel (STK) (Collins and Duffy, 2002) that maps a tree into the space of all possible tree fragments constrained by the rule that the sibling nodes from their parents cannot be separated. Different from STK where the nodes in the generated tree fragments are constrained to include none or all of their direct children, PTK fragments can contain any subset of the features, i.e., PTK allows for breaking the production rules. Consequently, PTK generalizes STK, thus generating an extremely rich feature space, which results in higher generalization ability.

2.3 Preference reranking with kernels

To enable the use of kernels for learning to rank with SVMs, we use preference reranking (Joachims, 2002), which reduces the task to binary classification. More specifically, the problem of learning to pick the correct candidate h_i from a candidate set $\{h_1, \ldots, h_k\}$ is reduced to a binary classification problem by creating *pairs*: positive training instances $\langle h_1, h_2\rangle, \ldots, \langle h_1, h_k\rangle$ and negative instances $\langle h_2, h_1\rangle, \ldots, \langle h_k, h_1\rangle$. This set can then be used to train a binary classifier. At classification time the standard one-versus-all binarization method is applied to form all possible

[3] http://uima.apache.org/

pairs of hypotheses. These are ranked according to the number of *classifier votes* they receive: a positive classification of $\langle h_k, h_i \rangle$ gives a vote to h_k whereas a negative one votes for h_i.

A vectorial representation of such pairs is the difference between the vectors representing the hypotheses in a pair. However, this assumes that features are explicit and already available whereas we aim at automatically generating implicit patterns with kernel methods. Thus, for keeping implicit the difference between such vectors we use the following preference kernel:

$$P_K(\langle h_1, h_2 \rangle, \langle h'_1, h'_2 \rangle) = K(h_1, h'_1) + K(h_2, h'_2) - K(h_1, h'_2) - K(h_2, h'_1), \quad (1)$$

where h_i and h'_i refer to two sets of hypotheses associated with two rankings and K is a kernel applied to pairs of hypotheses. We represent the latter as pairs of question and answer passage trees. More formally, given two hypotheses, $h_i = \langle h_i(q), h_i(a) \rangle$ and $h_i = \langle h'_i(q), h'_i(a) \rangle$, whose members are the question and answer passage trees, we define $K(h_i, h'_i)$ as $TK(h_i(q), h'_i(q)) + TK(h_i(a), h'_i(a))$, where TK can be any tree kernel function, e.g., STK or PTK.

To enable traditional feature vectors it is enough to add the product $(\vec{x}_{h_1} - \vec{x}_{h_2}) \cdot (\vec{x}_{h'_1} - \vec{x}_{h'_2})$ to the structural kernel P_K, where $\vec{x}_h$ is the feature vector associated with the hypothesis h.

We opted for a simple kernel sum over a product, since the latter rarely works in practice. Although in (Moschitti, 2004) the kernel product has been shown to provide some improvement when applied to tree kernels over a subcategorization frame structure, in general, it seems to work well only when the tree structures are small and derived rather accurately (Giordani and Moschitti, 2009; Giordani and Moschitti, 2012).

3 Structural models of Q/A pairs

First, we briefly describe a shallow tree representation that we use as our baseline model and then propose a new dependency-based representation.

3.1 Shallow tree structures

In a shallow syntactic representation first explored for QA in (Severyn and Moschitti, 2012) each question and its candidate answer are encoded into a tree where part-of-speech tags are found at the pre-terminal level and word lemmas at the leaf level. To encode structural relationships for a given q/a pair a special **REL** tag is used to link the related structures. The authors adopt a simple strategy to establish such links: lemmas shared between a question and and answer get their parents (POS tags) and grandparents (chunk labels) marked by a **REL** tag.

3.2 Dependency-based structures

Given the ability of PTK to generate a rich set of structural features from a relatively flat shallow tree representation, we propose to use dependency relations between words to derive an alternative structural model. In particular, we use a variation of the dependency tree, where dependency relations are altered in such a way that the words are always at the leaf level. This reordering of the nodes in the dependency tree, s.t. words do not form long chains, which is typical in the standard dependency tree representation, is essential for PTK to extract meaningful fragments. We also add part-of-speech tags between the words and the nodes encoding their grammatical roles (provided by the original dependency parse tree). Again a special **REL** tag is used in the same manner as in the shallow representation to establish structural links between a question and an answer. Fig. 2 (top) gives an example of a dependency-based structure for our example q/a pair.

4 Relational Linking

The use of a special tag to mark the related fragments in the question and answer tree representations has been shown to yield more accurate relational models (Severyn and Moschitti, 2012). However, previous approach was based on a naïve hard matching between word lemmas.

Below we propose a novel strategy to establish relational links using named entities extracted from the answer along with question focus and category classifiers. In particular, we use a question category to link the focus word of a question with the named entities extracted from the candidate answer. For this purpose, we first introduce our tree kernel-based models for building a question focus and category classifiers.

4.1 Question focus detection

The question focus is typically a simple noun representing the entity or property being sought by the question (Prager, 2006). It can be used to search for semantically compatible candidate an-

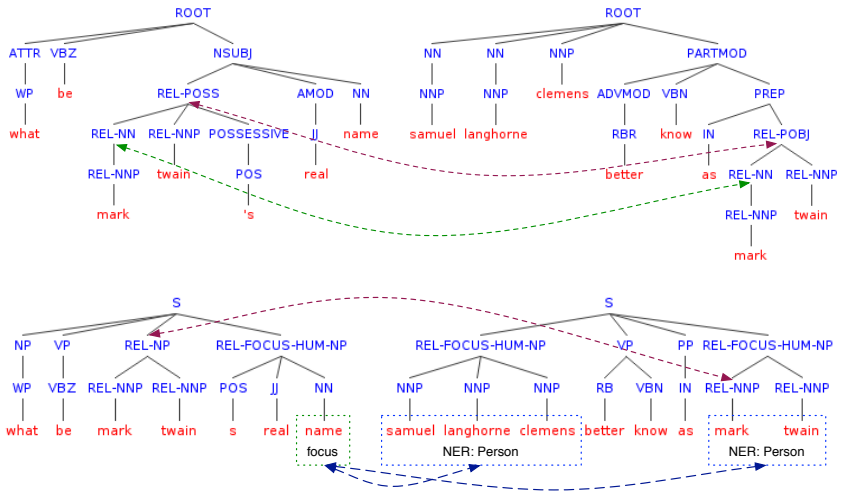

Figure 2: Dependency-based structure DEP (**top**) for the q/a pair. *Q: What is Mark Twain's real name? A: Samuel Langhorne Clemens, better known as Mark Twain*. Arrows indicate the tree fragments in the question and its answer passage linked by the relational **REL** tag. Shallow tree structure CH (**bottom**) with a typed relation tag **REL-FOCUS-HUM** to link a question focus word *name* with the named entities of type *Person* corresponding to the question category (HUM).

swers in document passages, thus greatly reducing the search space (Pinchak, 2006). While several machine learning approaches based on manual features and syntactic structures have been recently explored, e.g. (Quarteroni et al., 2012; Damljanovic et al., 2010; Bunescu and Huang, 2010), we opt for the latter approach where tree kernels handle automatic feature engineering.

In particular, to detect the question focus word we train a binary SVM classifier with tree kernels applied to the constituency tree representation. For each given question we generate a set of candidate trees where the parent (node with the POS tag) of each candidate focus word is annotated with a special **FOCUS** tag. Trees with the correctly tagged focus word constitute a positive example, while the others are negative examples. To detect the focus for an unseen question we classify the trees obtained after tagging each candidate focus word. The tree yielding the highest classification score reveals the target focus word.

4.2 Question classification

Question classification is the task of assigning a question to one of the pre-specified categories. We use the coarse-grain classes described in (Li and Roth, 2002): six non-overlapping classes: Abbreviations (ABBR), Descriptions (DESC, e.g. definitions or explanations), Entity (ENTY, e.g. animal, body or color), Human (HUM, e.g. group or individual), Location (LOC, e.g. cities or countries) and Numeric (NUM, e.g. amounts or dates). These categories can be used to determine the Expected Answer Type for a given question and find the appropriate entities found in the candidate answers. Imposing such constraints on the potential answer keys greatly reduces the search space.

Previous work in Question Classification reveals the power of syntactic/semantic tree representations coupled with tree kernels to train the state-of-the-art models (Bloehdorn and Moschitti, 2007). Hence, we opt for an SVM multi-classifier using tree kernels to automatically extract the question class. To build a multi-class classifier we train a binary SVM for each of the classes and apply a one-vs-all strategy to obtain the predicted class. We use constituency trees as our input representation.

4.3 Linking focus word with named entities using question class

Question focus captures the target information need posed by a question, but to make this piece of information effective, the focus word needs to be linked to the target candidate answer. The focus word can be lexically matched with words present in an answer, or the match can be established using semantic information. Clearly, the latter approach is more appealing since it helps to alleviate the lexical gap problem which makes the näive string matching of words between a question and its answer less reliable.

Hence, we propose to exploit a question category (automatically identified by a question type classifier) along with named entities found in the answer to establish relational links between the tree structures of a given q/a pair. In particular, once the question focus and question category

Table 1: Question classes → named entity types.

Question Category	Named Entity types
HUM	Person
LOC	Location
NUM	Date, Time, Money, Percentage
ENTY	Organization, Person

are determined, we link the focus word w_{focus} in the question, with all the named entities whose type matches the question class. Table 1 provides the correspondence between question classes and named entity types. We perform tagging at the chunk level and use two types of relational tags: plain **REL-FOCUS** and a tag typed with a question class, e.g., **REL-FOCUS-HUM**. Fig. 2 (bottom) shows an example q/a pair where the typed relational tag is used in the shallow syntactic tree representation to link the chunk containing the question focus *name* with the named entities of the corresponding type *Person* (according to the mapping defined in Table 1), i.e. *samuel langhorne clemens* and *mark twain*.

5 Feature vector representation

While the primary focus of our study is on the structural representations and relations between q/a pairs we also include basic features widely used in QA:

Term-overlap features. A cosine similarity between a question and an answer: $sim_{COS}(q, a)$, where the input vectors are composed of: (i) n-grams (up to tri-grams) of word lemmas and part-of-speech tags, and (ii) dependency triplets. For the latter, we simply hash the string value of the predicate defining the triple together with its argument, e.g. *poss(name, twain)*.

PTK score. For the structural representations we also define a similarity based on the PTK score: $sim_{PTK}(q, a) = PTK(q, a)$, where the input trees can be both dependency trees and shallow chunk trees. Note that this similarity is computed between the members of a q/a pair, thus, it is very different from the one defined in Eq. 1.

NER relatedness represents a match between a question category and the related named entity types extracted from the candidate answer. It counts the proportion of named entities in the answer that correspond to the question type returned by the question classifier.

In our study feature vectors serve a complementary purpose, while the main focus is to study the virtue of structural representations for reranking. The effect of a more extensive number of pairwise similarity features in QA has been studied elsewhere, e.g., (Surdeanu et al., 2008).

6 Experiments

We report the results on two QA collections: factoid open-domain QA corpus from TREC and a community QA corpus Answerbag. Since we focus on passage reranking we do not carry out answer extraction. The goal is to rank the passage containing the right answer in the top position.

6.1 Corpora

TREC QA. In the TREC QA tasks, answer passages containing correct information nuggets, i.e. answer keys, have to be extracted from a given text corpus, typically a large corpus from newswire. In our experiments, we opted for questions from 2002 and 2003 years, which totals to 824 questions. AQUAINT newswire corpus[4] is used for searching the supporting answers.

Answerbag is a community-driven QA collection that contains a large portion of questions that have "professionally researched" answers. Such answers are provided by the website moderators and allow for training high quality models. From the original corpus containing 180k question/answer pairs, we use 1k randomly sampled questions for testing and 10k for training.

Question Focus. We use 3 datasets for training and evaluating the performance of our focus detector: SeCo-600 (Quarteroni et al., 2012), Mooney GeoQuery (Damljanovic et al., 2010) and the dataset from (Bunescu and Huang, 2010). The SeCo dataset contains 600 questions from which we discarded a subset of multi-focus questions and non-interrogative queries. The Mooney GeoQuery contains 250 question targeted at geographical information in the U.S. The first two datasets are very domain specific, so we also carried out experiments with the dataset from (Bunescu and Huang, 2010), which contains the first 2000 questions from the answer type dataset from Li and Roth annotated with focus words. We removed questions with implicit and multiple focuses.

Question Classification. We used the UIUC dataset (Li and Roth, 2002)[5] which contains 5952

[4] http://www.ldc.upenn.edu/Catalog/docs/LDC2002T31/

[5] although the QC dataset from (Li and Roth, 2002) includes additional 50 fine grain classes we opted for using only 6 coarse classes that are sufficient to capture the coarse semantic answer type of the candidate answer. This choice also results in a more accurate multi-class classifier.

factoid questions from different sources (USC, TREC 8, TREC 9, TREC 10). For training the classifiers we excluded questions from TREC 8 to ensure there is no overlap with the data used for testing models trained on TREC QA.

6.2 Models and Metrics

Our models are built applying a kernel-based reranker to the output of a search engine.

6.2.1 BM25

We use Terrier[6] search engine, which provides BM25 scoring model for indexing and retrieval. For the TREC QA 2002 and 2003 task we index AQUAINT corpus treating paragraphs as documents. The resulting index contains about 12 million items. For the Answerbag we index the entire collection of 180k answers. We retrieve a list of top 50 candidate answers for each question.

6.2.2 Reranking models

To train our reranking models we used SVM-light-TK[7], which encodes structural kernels in SVM-light (Joachims, 2002) solver. In particular, we use PTK on the relational tree structures combined with the polynomial kernel of degree 3 applied to the feature vectors. Therefore, different representations lead to different models described below.

CH - our basic shallow chunk tree (Severyn and Moschitti, 2012) used as a **baseline** structural reranking model.

DEP - dependency tree augmented with POS tags and reorganized relations suitable for PTK.

V - reranker model using similarity features defined in Sec. 5.

DEP+V, CH+V - a combination of tree structures and similarity feature vectors.

+FC+QC - relational linking of the question focus word and named entities of the corresponding type using Focus and Question classifiers.

+TFC+QC - a typed relational link refined a question category.[8]

6.2.3 Metrics

We report the following metrics, most commonly used in QA: Precision at rank 1 (P@1), i.e., the percentage of questions with a correct answer at rank 1, Mean Reciprocal Rank (MRR), and Mean Average Precision (MAP). The reported metrics are averages after performing a 5-fold cross-validation. We used a paired t-test at 95% confidence to compare the performance of our models to a baseline.

Table 2: Structural representations on TREC QA.

MODELS	MAP	MRR	P@1
BM25	0.22	28.02	18.17
V	0.22	28.40	18.54
STRUCTURAL REPRESENTATIONS			
CH (S&M, 2012)	0.28	35.63	24.88
CH+V	$0.30^{\dagger}$	$37.45^{\dagger}$	$27.91^{\dagger}$
DEP	$0.30^{\dagger}$	$37.87^{\dagger}$	$28.05^{\dagger}$
DEP+V	$0.30^{\dagger}$	$37.64^{\dagger}$	$28.05^{\dagger}$
REFINED RELATIONAL TAG			
CH+V+QC+FC	$0.32^{\ddagger}$	$39.48^{\ddagger}$	$29.63^{\ddagger}$
CH+V+QC+TFC	$0.32^{\ddagger}$	$39.49^{\ddagger}$	$30.00^{\ddagger}$
DEP+V+QC+FC	$0.31^{\ddagger}$	37.49	28.56
DEP+V+QC+TFC	$0.31^{\ddagger}$	$38.05^{\ddagger}$	$28.93^{\ddagger}$

6.3 Passage Reranking Results

We first evaluate the impact of two different syntactic representations using shallow and dependency trees. Then, we evaluate the accuracy boost when such structures are enriched with automatically derived tags, e.g., question focus and question category and NEs found in the answer passage.

6.3.1 Structural representations

Table 2 reveals that using V model results in a small improvement over BM25 baseline. Indeed, similarity scores that are most often based on word-overlap measures even when computed over various q/a representations are fairly redundant to the search engine similarity score. Instead, using the structural representations, CH and DEP, gives a bigger boost in the performance. Interestingly, having more features in the CH+V model results in further improvement while DEP+V seems to remain insensitive to additional features provided by the V model.

6.3.2 Semantically Enriched Structures

In the following set of experiments we explore another strategy for linking structures for a given q/a pair. We automatically detect the question focus word and link it to the related named entities in the answer, selected accordingly to the question category identified by the question classifier (QC+FC). Further refining the relational link

[6] http://terrier.org/

[7] http://disi.unitn.it/moschitti/Tree-Kernel.htm

[8] † is used for showing the results of DEP, DEP+V and CH+V structural representations that are significantly better than the baseline model CH, while ‡ indicates improvement of +QC+FC and +QC+TFC tagging applied to basic structural representations, e.g. CH+V and DEP+V.

Table 3: Accuracy (%) of focus classifiers.

DATASET	ST	STK	STK+BOW	PTK
MOONEY	73.0	81.9	81.5	80.5
SECO-600	90.0	94.5	94.5	90.0
BUNESCU	89.7	98.3	98.2	96.9

Table 4: Accuracy (%) of question classifiers.

DATASET	STK+BOW	PTK
LI & ROTH	86.1	82.2
TREC TEST	79.3	78.1

with the question category yields QC+TFC model. First, we report the results of training our question focus detector and question category classifier.

Focus classifier results. Table 3 displays the accuracies obtained by the question focus detector on 3 datasets using different kernels: the ST (subtree kernel where fragments contain full subtrees including leaves), STK, STK+bow (bag-of-words feature vector is added) and PTK. As we can see, using STK model yields the best accuracy and we use it in our pipeline to automatically detect the focus.

Question classifier results. Table 4 contains the accuracies of the question classifier on the UIUIC dataset and the TREC questions that we also use for testing our reranker models. STK+bow performs better than PTK, since here the input representation is a plain constituency tree, for which STK is particularly suited. Hence, we use this model to predict the question category.

Ranking results. Table 2 (bottom) summarizes the performance of the CH+V and DEP+V models when coupled with QC+FC and QC+TFC strategies to establish the links between the structures in a given q/a pair. CH structural representation with QC+FC yields an interesting improvement, while further refining the relational tag by adding a question category (QC+TFC) gives slightly better results.

Integrating the refined relational tag into the DEP based structures results more problematic, since the dependency tree is less suitable for representing multi-word expressions, named entities in our case. Hence, using the relational tag to mark the nodes spanning such multi-word entities in the dependency structure may result in less meaningful features than in CH model, where words in a phrase are naturally grouped under a chunk node. A more detailed discussion on the merits of each model is provided in the Sec. 6.5.

Table 5: Cross-domain experiment: training on Answerbag and testing on TREC QA.

MODELS	MAP	MRR	P@1
BM25	0.22	27.91	18.08
V	0.23	28.86	18.90
BASIC STRUCTURAL REPRESENTATIONS			
CH (S&M, 2012)	0.24	30.25	20.42
CH+V	$0.25^{\dagger}$	$31.31^{\dagger}$	$21.28^{\dagger}$
DEP+V	$0.26^{\dagger}$	$33.26^{\dagger}$	$22.21^{\dagger}$
REFINED RELATIONAL TAG			
CH+V+QC+TFC	$0.27^{\ddagger}$	$33.53^{\ddagger}$	$22.81^{\ddagger}$
DEP+V+QC+TFC	$0.29^{\ddagger}$	$34.25^{\ddagger}$	$23.45^{\ddagger}$

6.4 Learning cross-domain pairwise structural relationships

To test the robustness of the syntactic patterns automatically learned by our structural models, we conduct a cross-domain experiment, i.e. we train a model on Answerbag data and test it on TREC. It should be noted that unlike TREC data, where the answers are simply passages containing the correct answer phrase, answers in Answerbag specifically address a given question and are generated by humans. Additionally, TREC QA contains only factoid questions, while Answerbag is a community QA corpus with a large portion of non-factoid questions. Interestingly, the results demonstrate the robustness of our syntactic relational model which captures patterns shared across different domains, e.g. TREC and Answerbag data.

Table 5 shows that: (i) models based on dependency structures result in a better generalization ability extracting more robust syntactic patterns; and (ii) the strategy to link the question focus with the related named entities in the answer provides an interesting improvement over the basic structural representations.

6.5 Error Analysis

Consider our running example q/a pair from Sec. 1. As the first candidate answer, the search engine retrieves the following incorrect passage: *"The autobiography of Mark Twain", Mark Twain*. It is relatively short and mentions the keywords {*Mark, Twain*} twice, which apparently results in a high score for the BM25 model. Instead, the search engine ranks the correct answer at position 34. After reranking using the basic CH+V model the correct answer is promoted by 20 positions. While using the CH+V+QC+FC model the correct answer advances to position 6. Below, we provide the intuition behind the merits of QC+FC and QC+TFC encoding question focus and ques-

tion category into the basic models.

The model learned by the reranker represents a collection of q/a pairs from the training set (support vectors) which are matched against each candidate q/a pair. We isolated the following pair from the model that has a high structural similarity with our running example:

Q: What is Barbie's full name?

A: The toy is called after Barbie Millicent Roberts from Willows.

Despite differences in the surface forms of the words, PTK extracts matching patterns, e.g. `[S NP [VP VBN] [PP IN] REL-NP]`, which yields a high similarity score boosting the rank of the correct candidate. However, we note that at the same time an incorrect candidate answer, e.g. *Mark Twain was accused of racist language.*, exhibits similar patterns and also gets a high rank. The basic structural representation is not able to encode essential differences from the correct answer candidate. This poses a certain limitation on the discriminative power of CH and DEP representations. Introducing a focus tag changes the structural representation of both q/a pairs, s.t. the correct q/a pair preserves the pattern (after identifying word *name* as focus and question category as HUM, it is transformed to `[S REL-FOCUS-NP [VP VBN] [PP IN] REL-FOCUS-NP]`), while it is absent in the incorrect candidate. Thus, linking the focus word with the related NEs in the answer helps to discriminate between structurally similar yet semantically different candidates.

Another step towards a more fine-grained structural representation is to specialize the relational focus tag (QC+TFC model). We propose to augment the focus tag with the question category to avoid matches with other structurally similar but semantically different candidates. For example, a q/a pair found in the list of support vectors:

Q: What is Mark Twain's place of birth?

A: Mark Twain was raised in Hannibal Missouri.

would exhibit high structural similarity even when relational focus is used (since the relational tag does not incorporate the question class LOC), but refining the focus tag with the question class eliminates such cases.

7 Related Work

Previous studies similar to ours carry out passage reranking by exploiting structural information, e.g. using subject-verb-object relations (Attardi et al., 2001; Katz and Lin, 2003). Unfortunately, the large variability of natural language makes such triples rather sparse thus different methods explore soft matching (i.e., lexical similarity) based on answer types and named entity types (Aktolga et al., 2011). Passage reranking using classifiers of question and answer pairs were proposed in (Radlinski and Joachims, 2006; Jeon et al., 2005).

Regarding kernel methods, our work in (Moschitti et al., 2007; Severyn and Moschitti, 2012) was the first to exploit tree kernels for modeling answer reranking. However, such method lacks the use of important relational information between a question and a candidate answer, which is essential to learn accurate relational patterns. In contrast, this paper relies on shallow and dependency trees encoding the output of question and focus classifiers to connect focus word and NEs of the answer passage. This provides more effective relational information, which allows our model to significantly improve on previous rerankers.

8 Conclusions

This paper shows a viable research direction in the automatic QA engineering. One of its main characteristics is the use of structural kernel technology to induce features from structural semantic representations of question and answer passage pairs. The same technology is also used to construct question and focus classifiers, which are used to derive relational structures.

An interesting result of this paper is that to design an answer passage reranker for a new domain, we can use off-the-shelf syntactic parsers and NERs along with little training data for the QC and FC classifiers. This is due to the fact that: (i) the kernel technology is able to automatically extract effective structural patterns; and (ii) the extracted patterns are rather robust, e.g., models learned on Answerbag improve accuracy on TREC test data.

Acknowledgements

This research is partially supported by the EU's 7th Framework Program (FP7/2007-2013) (#288024 LIMOSINE project) and an Open Collaborative Research (OCR) award from IBM Research.

References

Elif Aktolga, James Allan, and David A. Smith. 2011. Passage reranking for question answering using syntactic structures and answer types. In *ECIR*.

Giuseppe Attardi, Antonio Cisternino, Francesco Formica, Maria Simi, and Ro Tommasi. 2001. Piqasso: Pisa question answering system. In *TREC*, pages 599–607.

Stephan Bloehdorn and Alessandro Moschitti. 2007. Combined syntactic and semantic kernels for text classification. In *ECIR*.

Razvan Bunescu and Yunfeng Huang. 2010. Towards a general model of answer typing: Question focus identification. In *CICLing*.

Michael Collins and Nigel Duffy. 2002. New Ranking Algorithms for Parsing and Tagging: Kernels over Discrete Structures, and the Voted Perceptron. In *ACL*.

Danica Damljanovic, Milan Agatonovic, and Hamish Cunningham. 2010. Identification of the question focus: Combining syntactic analysis and ontology-based lookup through the user interaction. In *LREC*.

Alessandra Giordani and Alessandro Moschitti. 2009. Syntactic structural kernels for natural language interfaces to databases. In *Proceedings of ECML PKDD*, ECML PKDD '09. Springer-Verlag.

Alessandra Giordani and Alessandro Moschitti. 2012. Translating questions to sql queries with generative parsers discriminatively reranked. In *Proceedings of The 24rd International Conference on Computational Linguistics*, India. Coling 2012.

Jiwoon Jeon, W. Bruce Croft, and Joon Ho Lee. 2005. Finding similar questions in large question and answer archives. In *CIKM*.

T. Joachims. 2002. Optimizing search engines using clickthrough data. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 133–142.

Boris Katz and Jimmy Lin. 2003. Selectively using relations to improve precision in question answering.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING*.

Alessandro Moschitti and Silvia Quarteroni. 2008. Kernels on linguistic structures for answer extraction. In *ACL*.

Alessandro Moschitti, Silvia Quarteroni, Roberto Basili, and Suresh Manandhar. 2007. Exploiting syntactic and shallow semantic kernels for question/answer classification. In *ACL*.

Alessandro Moschitti. 2004. A study on convolution kernels for shallow statistic parsing. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 335–342, Barcelona, Spain, July.

Alessandro Moschitti. 2006. Efficient convolution kernels for dependency and constituent syntactic trees. In *ECML*.

Alessandro Moschitti. 2008. Kernel methods, syntax and semantics for relational text categorization. In *CIKM*.

Christopher Pinchak. 2006. A probabilistic answer type model. In *In EACL*.

John M. Prager. 2006. Open-domain question-answering. *Foundations and Trends in Information Retrieval*, 1(2):91–231.

Silvia Quarteroni, Vincenzo Guerrisi, and Pietro La Torre. 2012. Evaluating multi-focus natural language queries over data services. In *LREC*.

Filip Radlinski and Thorsten Joachims. 2006. Query chains: Learning to rank from implicit feedback. *CoRR*.

Aliaksei Severyn and Alessandro Moschitti. 2012. Structural relationships for large-scale learning of answer re-ranking. In *SIGIR*.

M. Surdeanu, M. Ciaramita, and H. Zaragoza. 2008. Learning to rank answers on large online QA collections. In *Proceedings of ACL-HLT*.