Selecting Sentences versus Selecting Tree Constituents for Automatic Question Ranking

Alberto Barrón-Cedeño and **Giovanni Da San Martino** and
Salvatore Romeo and **Alessandro Moschitti**

Qatar Computing Research Institute,
Hamad bin Khalifa University
Doha, Qatar
{albarron, gmartino, sromeo, amoschitti}@qf.org.qa

Abstract

Community question answering (cQA) websites are focused on users who query questions onto an online forum, expecting for other users to provide them answers or suggestions. Unlike other social media, the length of the posted queries has no limits and queries tend to be multi-sentence elaborations combining context, actual questions, and irrelevant information. We approach the problem of question ranking: given a user's new question, to retrieve those previously-posted questions which could be equivalent, or highly relevant. This could prevent the posting of nearly-duplicate questions and provide the user with instantaneous answers. For the first time in cQA, we address the selection of relevant text —both at sentence- and at constituent-level— for parse-tree-based representations. Our supervised models for text selection boost the performance of a tree kernel-based machine learning model, allowing it to overtake the current state of the art on a recently released cQA evaluation framework.

1 Introduction

Community-driven question-and-answering (cQA) sites are popular forums in which users ask and answer questions on diverse topics. The freedom in these websites, and the diversity of their users, promote massive participation, resulting in large amounts of texts in the form of both questions and answers.

All the steps in the assembly line of these highly-collaborative sites —from the question posting up to the seek for sensitive answers among those triggered by the question—, pose interesting natural language processing (NLP) challenges. When a user posts a query question, a model can search for previously-posted equivalent or relevant questions which might address the user's information need at once. Once a number of comments have been posted intending to answer a question, another model can select the most appropriate one, or at least rank them on the basis of their quality. The same technology can be applied to discard inappropriate or diverting answers. When a website has accumulated a significant amount of posts, a model can be implemented to look for near-duplicates or related questions and answers.

Different approaches have been proposed to address these tasks. Here we focus on the first of them: question re-ranking in cQA. That is, given a query question q and a pool of previously-posted questions D, rank the questions in D according to their relevance against q. This problem has attracted the attention of a manifold of research works which rely on the use of standard lexical similarity metrics, semantic representations, machine-translation models, tree kernels, or neural networks, to mention just some examples of representations and models. Nevertheless, those approaches neglect one of the inherent facets of cQA sites: opposed to other social media, users are free to post potentially ill-formed texts of anarchic lengths. They may include multiple sentences with courtesy chunks, potentially-redundant elaborations, or off-topic fragments. In this paper, we address the problem of selecting the most relevant text chunks in the questions in order to build a better representation of the texts to be fed into the machine learning machinery. We propose supervised and unsupervised models that operate both at sentence and at chunk level (using constituency parse trees) and apply them on top of a state-of-the-art tree-kernel-based classification model. To the best of our knowledge, this is the first time that this problem is addressed. Our

Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers,
pages 2515–2525, Osaka, Japan, December 11-17 2016.

results on a recently-released cQA corpus show that, by carefully selecting both sentences and words, the performance of the ranking model can be significantly improved (in the range of more than two MAP points), also improving the current best model on the evaluation framework we used.

The rest of our contribution is distributed as follows. Section 2 overviews the literature addressing different problems in cQA, paying special attention to question ranking. Section 3 describes the machine learning strategy to rank questions upon which we build our main contribution: the proposal of two models to select the most relevant —noise-less— text fragments to represent the questions, described in Section 4. Section 5 describes our experimental settings and discusses the obtained results. Finally, Section 6 draws conclusions and sketches some of our current efforts.

2 Related Work

Community question answering poses various challenges: answer and question ranking, and question de-duplication are three examples. We now review the related literature with focus on question ranking.

One of the first approaches to answer ranking relied completely on the website's metadata (Jeon et al., 2006), such as an author's reputation and click counts. Agichtein et al. (2008) explored a graph-based model of contributors relationships together with both content- and usage-based features. These approaches depend heavily on the forum's meta-data and social features. Still, as Surdeanu et al. (2008) stress, relying on this kind of data causes the model portability to be difficult; a drawback that disappears when focusing on the content of the questions and answers only. Therefore, our model is based on textual content only. Some of the most recent proposals aim at classifying whole threads of answers (Joty et al., 2015; Zhou et al., 2015) rather than each answer in isolation.

Question ranking can be approached from different fronts. Cao et al. (2008) approached it as a recommendation task: given a query question, recommend questions that could be interesting or relevant, regardless of whether they convey the same information request. They tackle this problem by comparing representations based on topic terms graphs; i.e., by judging topic similarity. In a follow up paper, Duan et al. (2008) searched for equivalent questions by considering the question's focus as well. Zhou et al. (2011) dodged the lexical gap between two questions by assessing their similarity on the basis of a (monolingual) phrase-based translation model (Koehn et al., 2003). They considered the (pre-filtered) contents of the question–answer pairs as their "parallel" corpus to learn the translation model from. Jeon et al. (2005b) had used monolingual translation as well. Given a large repository of question and answer threads, they looked for highly-similar threads (Jeon et al., 2005a). Similar answers are likely to address similar questions! The questions in the so-generated pairs compose their "parallel" corpus. Wang et al. (2009) computed their similarity function on the syntactic-tree representations of the questions. The more substructures the trees have in common, the more similar their associated questions are. A different approach using topic modeling for question retrieval was introduced by Ji et al. (2012) and Zhang et al. (2014). Here, the authors use LDA topic modeling to learn the latent semantic topics that generate question/answer pairs and use the learned topics distribution to retrieve similar historical questions.

The recent boom in neural network approaches has also impacted question retrieval. dos Santos et al. (2015) applied convolutional neural networks to retrieve semantically-equivalent questions' subjects. When dealing with whole questions —subject and (generally long) body—, they had to aggregate a bag-of-words neural network to boost the model's performance. They suggested that the performance of state-of-the-art models for semantic paraphrasing assessment on short texts (e.g., (Filice et al., 2015b)) cannot be applied straightforwardly to whole questions in cQA. For the first time, we address such problem in this paper.

The two editions of the SemEval Task 3 on cQA (Nakov et al., 2015; Nakov et al., 2016) have triggered a manifold of approaches. The datasets they released include manual crowd-sourced annotation rather than forum-inferred judgments. The 2015 edition focused on answer retrieval. The first performing system (Tran et al., 2015) applied machine translation in a similar fashion as Jeon et al. (2005b) and Zhou et al. (2011), together with topic models, embeddings, and similarities. Both the first and the second runners (Hou et al., 2015; Nicosia et al., 2015) applied supervised models with lexical, syntactic and meta-data features. The 2016 edition included a question retrieval challenge as well. We take ad-

vantage of the evaluation framework developed for this task. The top-three participants opted for SVMs as learning models. The top-ranked (Franco-Salvador et al., 2016) used SVM^{rank} (Joachims, 2006), the first (Barrón-Cedeño et al., 2016) and second (Filice et al., 2016) runners up used KeLP (Filice et al., 2015a) to combine various kernels. Another difference between these models is in the amount of knowledge they use. Franco-Salvador et al. (2016) rely heavily on distributed representations and semantic information sources, such as Babelnet and Framenet. Both Barrón-Cedeño et al. (2016) and Filice et al. (2016) use lexical similarities and tree kernels on parse trees. No statistically-significant differences were observed in the performance of these three systems.

To the best of our knowledge, so far the only other work exploring text selection to improve cQA systems is that of Romeo et al. (2016). In that paper, we do sentence selection and pruning on the basis of attention weights, computed with neural networks.

3 Base Model to Rank Questions

In this section we describe a state-of-the-art technique to assess question–question similarity in question ranking. This is the core of our ranking approach and the base on which we test our text selection models (cf. Section 4). We apply a learning-to-rank approach (Liu, 2009) to produce the ranking of forum questions. Related questions in cQA sites are usually spotted and labeled as such by the users themselves. This can be directly projected into a training dataset with binary annotations: *Relevant* vs. *Irrelevant* and a perfect ranking puts all the *Relevant* questions on top of all the *Irrelevant* ones, regardless of the order within both subsets (Cao et al., 2008; dos Santos et al., 2015; Jeon et al., 2005b). We adopt the same architecture as the recently-proposed, most successful, models on this kind of setting (Franco-Salvador et al., 2016; Barrón-Cedeño et al., 2016; Filice et al., 2016). We apply a kernel approach to solve a binary classification problem, $f : Q \times D \rightarrow \{Relevant, Irrelevant\}$, and sort the forum questions $d \in D$ according to their classification score against q: $f(q, d)$.

We opt for a tree kernel applied to parse-tree representations (Moschitti, 2006b; Sun et al., 2011), as it performs well in ranking both passages (Severyn and Moschitti, 2012) and questions (Barrón-Cedeño et al., 2016; Da San Martino et al., 2016; Filice et al., 2016). Additionally, the nodes of the parse trees of the pairs (q, d) are marked with a REL tag when there is at least a lexical match between the phrases of the questions (c.f. (Filice et al., 2015b) for details). The approach for dealing with a pair of trees, is to compose kernels on single trees $K^T(x_1, x_2)$:

$$K((q_i, d_i), (q_j, d_j)) = K^T(t(q_i, d_i)), t(q_j, d_j))) + K^T(t(d_i, q_i)), t(d_j, q_j))), \quad (1)$$

where d_i and d_j are the i^{th} and j^{th} retrieved questions and $t(x, y)$ extracts the syntactic tree from the text x, enriching it with REL tags computed with respect to y. Note that $t(x, y)$ is not symmetric: $t(y, x)$ would return the tree related to y enriched with REL tags. Specifically we apply a partial tree kernel (PTK) as the base kernel K^T, which counts the number of shared subtrees between x_1 and x_2 (Moschitti, 2006a).

4 Text Selection for Parse-Tree Representations

We complement our classifier with twenty-two similarities $sim(q, d)$ computed on lemmatized versions of the texts, which are described in Table 1. The inverse of the Google-generated position of d (included in the corpus) is included as well. We plug the similarities on an RBF kernel and combine it linearly with the tree kernel as they boost the performance of the tree kernels (Da San Martino et al., 2016).

Questions in cQA websites tend to be composed of multiple sentences (c.f. Figure 1) and to include noisy or irrelevant fragments. We apply the same trick as Filice et al. (2016) to feed our tree kernel with pairs of single trees: we hang together the constituency parse trees of all the sentences in q (d) from an additional root node. Still, tree kernels are expensive and sensitive to noise, thus it is difficult for them to deal with multi-sentence noisy text. In order to tackle these issues in the questions' texts, we designed two general strategies which operate either at sentence or at word level. Our objective is twofold: we want to represent our questions with the shortest —most informative— text fragments and at the same time discard noise, such as acknowledgments or unnecessary elaborations.

Metric		Details
String similarity		
Greedy string tiling	(Wise, 1996)	Considering a minimum matching length of 3.
Longest common subsequence	(Allison and Dix, 1986)	Both standard and normalized by the first string.
Longest common substring	(Gusfield, 1997)	Based on generalized suffix trees.
Lexical similarity		
Jaccard coefficient	(Jaccard, 1901)	Over stopworded $[1, \ldots, 4]$-grams.
Word containment	(Lyon et al., 2001)	Over stopworded $[1, \ldots, 2]$-grams.
Cosine		Over stopworded $[1, \ldots, 4]$-grams.
		Over $[1, \ldots, 4]$-grams.
		Over $[1, \ldots, 3]$-grams of part of speech.
Syntactic similarity		
PTK	(Moschitti, 2006a)	Similarity between shallow syntactic trees.

Table 1: Overview of similarity metrics.

4.1 Learning to Select Sentences

This selection strategy occurs before the parse trees of the questions are built. The output of this process is a selected number of sentences from q and d, composed on the basis of sentence-pair rankings. Algorithm 1 sketches our strategy. Firstly, we combine all the possible pairs of sentences between questions q and d. Secondly, we compute a similarity function φ for each pair and rank the pairs accordingly. Thirdly, we identify the k pairs of sentences with the highest similarities. Finally, we discard those sentences from q and d which have not been included within the top-k pairs. Our rationale implies a number of constraints. The sentences that result from the process are in the same order as in the original questions; not in the order of the sentence ranking. This is because we want to preserve the discourse structure of the text. Sentences are promiscuous: they can be linked to different sentences from the other question. If a sentence is part of more than one of the top-k pairs, it is not duplicated in the output. Finally, the input question may go unaltered on either side. The similarity function φ is the key factor for this model and we experimented with two learning strategies:

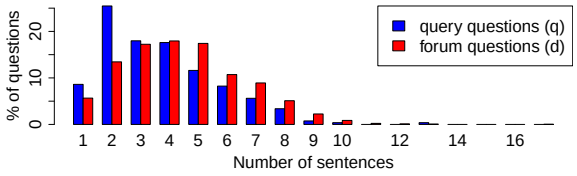


Figure 1: Distribution of question lengths in terms of sentences for query (q) and forum (d) questions.

Our unsupervised model is based on the cosine similarity over TF×IDF-weighted vector representations of the sentences in q and d.

Our supervised model is based on a binary SVM classifier, using the same similarity functions as in Section 3, over and RBF kernel. We add information in terms of features on the position of the sentence within the question: the inverse of the position, whether the sentence appears in position 1, between positions 2 and 4 (inclusive), or after position 4. We take 4 as threshold because it is the mode of the number of sentences in the questions of the corpus (mean=4.38). Our prediction function is $c(p_q, p_d) \in \{Relevant, Irrelevant\}$. The class labels are borrowed from those at question level.

We took good care of not misusing the development and test data. In the unsupervised model, we compute the document-frequency statistics on sentences from the training set only. In the supervised model, the scores for the training set are estimated by 5-fold cross validation. On dev and test, the used document frequencies are those from the training set and the scores are computed with the model trained on the training partition.

4.2 Learning to Select Constituents in Tree Kernel Spaces

Our strategy for selecting text at token level is to operate directly on the parse trees and prune those branches which are associated with less important or noisy text fragments. We use a supervised approach based on kernel methods to determine such fragments. A few facts on kernel methods need to be recalled to better understand our approach. After training a kernel method, using a kernel function $K()$, on the

Input :
q the query question
d the forum question
k the maximum number of pairs to return
Output:
q' q, after selecting the best sentences; $len(q') \leq len(q)$
d' d, after selecting the best sentences; $len(d') \leq len(d)$

$P \leftarrow sent(q) \times sent(d)$ // All the possible sentence pairs between q and d
for $p \in P$ **do**
  $score_p = \varphi(p_q, p_d)$ // Compute the similarity between the sentences
end
$P \leftarrow rank(P, score_P)$ // Rank the pairs in P in terms of the similarities
$P \leftarrow trim(P, k)$ // Select (at most) the top-k pairs in P
$q' \leftarrow q \cap P_q$ // Remove from q all the sentences not in P
$d' \leftarrow d \cap P_d$ // Remove from d all the sentences not in P
return q', d'

Algorithm 1: Sentence-level selection.

problem sketched in Section 3, the solution of the dual optimization problem is expressed as a linear combination of a subset of the training examples: $M = \{(\alpha_i, (q_i, d_i))\}$, where the (q_i, d_i) are training examples and α_i are the coefficients of the combination. The classification of a new example is obtained as the sign of the score function $f()$:

$$f\left(q, d\right) = \sum_{1 \leq i \leq |M|} \alpha_i K\left((q, d), (q_i, d_i)\right), \tag{2}$$

where $|M|$ is the number of support vectors, i.e., the number of elements of the set M. The higher the absolute value of the score of an example, the more confident the learning algorithm is in classifying it. We exploit such property of the kernel methods to devise a strategy to determine the importance $w(n)$ of a node. Let n be a node of a tree t, $\overset{n}{\triangle}$ is the proper sub-tree rooted at n, i.e., the tree composed of n and all its descendants in t. We use the score of $\overset{n}{\triangle}$ with respect to M to assess the importance of n:

$$w(n) = \begin{cases} \sum_{1 \leq i \leq |M|} \alpha_i K^T(\overset{n}{\triangle}, q_i) & \text{if } n \in q, q \in Q \\ \sum_{1 \leq i \leq |M|} \alpha_i K^T(\overset{n}{\triangle}, d_i) & \text{if } n \in d, d \in D. \end{cases} \tag{3}$$

In order to be consistent, only the parse trees of $q_i \in Q$ will be used to compute $w(n)$, if n belongs to a question in Q for each pair $(q_i, d_i) \in M$. Conversely if n belongs to a question in D only the parse trees of $d_i \in D$ will be used. Note that, by grouping and caching the scores computations as in (Aiolli et al., 2011, eq. (4)), the worst-case complexity of Eq. (3) for all the nodes is the same as the complexity of predicting the class of the pair (q, d).

Now we can proceed to prune a tree on the basis of the $w(n)$ importance estimated by model M for each of its nodes and a user-defined threshold. We prune a leaf node n if $-h < w(n) < h$. If n is not a leaf, then it is removed if all its children are going to be removed. Note that the threshold h determines the number of pruned nodes. Our algorithm has a constraint: REL-tagged nodes are never pruned, regardless of their estimated importance. This is because a REL tag indicates that q and d share a common leaf in $\overset{n}{\triangle}$, which conveys useful information for paraphrasing (Filice et al., 2015b).

5 Experiments

We perform three different experiments. Firstly, we evaluate the performance of our supervised sentence-level classifier and select the best of them for the next experiment. Secondly, we analyze the impact of the supervised and unsupervised sentence selectors in the performance of our overall question ranking model. Thirdly, we evaluate the impact of our tree-pruning strategy in the performance of our overall

Class	train	dev	test	overall
Relevant	1,083	214	233	1,530
Irrelevant	1,586	286	467	2,339
Total	2,669	500	700	3,869

Table 2: Class distribution in the training, development, and test partitions of the cQA-2016 corpus.

Model	Acc	P	R	F_1	MAP	AvgRec	MRR
sim	65.88	44.44	14.29	21.62	60.15	85.39	64.22
$sim + pos_{lin}$	68.24	53.85	25.00	34.15	61.13	85.79	64.22
$sim + pos_{RBF}$	71.76	59.09	46.43	52.00	62.84	86.95	66.67

Table 3: Performance of the sentence classifier on a subset of manually annotated sentences from the development set. Similarity features used on a linear kernel. Positional (pos) features are either used on a linear or an RBF kernel.

question ranking model. As discussed in Section 3, we use binary SVMs to generate the rankings, combining a tree kernel for the parse trees and an RBF kernel for the rest of features.

5.1 Setup

We run our experiments on the SemEval 2016 Task 3 on Community Question Answering evaluation framework (Nakov et al., 2016), which uses the cQA-2016 corpus in which each item includes a query question linked to ten potentially-relevant candidate questions. Table 2 shows the class distribution. We stick to the retrieval problem formulation of the task: a perfect model should rank relevant documents on top of the irrelevant ones. Our evaluation is based on mean average precision, average recall, and mean-reciprocal rank. This allows for a direct comparison against Task 3's official ones.

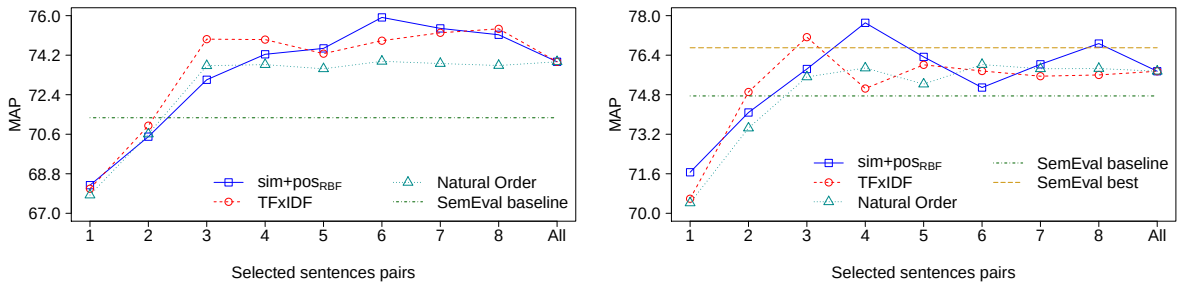
5.2 Impact of Sentence Selection in Question Ranking

We first evaluate our sentence ranking model in isolation, before assessing its impact on the question ranker. We trained a sentence level classifier, on the training partition of the corpus, as defined in Section 4.1 —considering the question-level annotations as gold standard. However, the latter choice introduces a lot of false positives as sentences in relevant questions may be unrelated. Thus, for correctly evaluating the sentence classifier, we used labels obtained by crowdsourcing. We selected sentence pairs to be annotated from the development set according to a few constraints. The (q, d) pairs we extract the sentences from must include at least five sentences in d. Each selected sentence must include at least 3 content words and 3 stop-words. This resulted in 125 sentences for this small control experiment. We submitted HIT's composed of 15 sentence pairs to CrowdFlower.[1] Annotators had to decide if two sentences "expressed the same information". Each item was annotated by 3 contributors, with an agreement of 0.83. 21% of the sentence pairs resulted as positive.

Table 3 shows the performance of different configurations of the sentence-level classifier, trained on the training partition and tested on the Crowdflower-annotated data. The task is the same as at question level: ranking, but we include accuracy, precision, recall, and F_1-measure to get a clear picture. Similarity measures constantly perform better on RBF kernels and, as observed, positional features do as well. The performance boost caused by the positional features is more evident in terms of F_1 and recall. These values might be perceived as relatively low, but let us keep in mind that many (unrelated) sentences in the training set inherit incorrect labels from the label of their question. Given these figures, we select the model with both similarities and position features on an RBF kernel to select sentences for the question ranker. Our unsupervised model does not require any evaluation nor tuning.

Now we look at the impact of our sentence selection model in the question ranker. Figure 2 shows the performance of the ranker as a function of the number of sentences. In our supervised model sim+pos$_{RBF}$, the x-axis refers to the number of pairs considered after the SVM-generated ranking. In our unsupervised model TF$\times$IDF, the x-axis stands for the number of sentences that represent q and d in the ranking model. We also display the behavior when the sentences are added in the natural order, which could be considered as a sentence-selection baseline. The constant line "SemEval baseline" corresponds to the competition baseline (Nakov et al., 2016). The upper constant line in Figure 2b corresponds to the winner of the SemEval competition (Franco-Salvador et al., 2016).

[1] https://www.crowdflower.com



(a) On the development set.

(b) On the test set.

Figure 2: Evolution of MAP for various sentence-selection strategies. The point marked as *All* on the x axis —where the curves converge— corresponds to the values obtained when using all the sentences.

It is worth observing that all our models depart relatively low performance values when considering one single sentence: MAP $\simeq$ 68.12 on development and 70.40 on test. Nevertheless, the natural order one stops improving at 3 sentences and it never goes over the base system, which considers the full texts. This is not the case for our selection models: already from 3 (and 4) sentences they improve over the full-text system. On the development set the supervised model reaches a MAP of 76.01 with five sentences and the unsupervised reaches 75.40 with eight sentences. The SVM supervised model is also the best on test, even when considering less sentences (Figure 2b). Again, the best performance is obtained by the supervised model. Although, TF$\times$IDF has a peak with 3 sentences, adding more pairs results in a quick performance decrease. Therefore, the supervised sentence classifier is more stable and thus preferable. In summary, our sentence pre-selection manages to identify those sentences which are more relevant to assess the similarity between two questions and produce a better ranking. Overall, the supervised model performs better than the unsupervised one and the best configurations improve over the SemEval best system. We select the best performing model on development —the SVM on sim+pos$_{RBF}$— and list its corresponding performance in Table 4, for comparison with the best pruning model and the rest of systems.

5.3 Impact of Constituency Tree Selection on Question Reranking

We experimented with two models for generating the weights $w()$ to prune the nodes: (*i*) an SVM with the kernels described in Section 3 (*model 1*); (*ii*) the same SVM trained on the training set after applying the sentence selection described in section 4.1 (*model 2*).

We first performed a 5-fold cross validation on the training set to select a pruning threshold. For each cross validation split into train/test, we first perform learning on train to get weigths using *model 1*, then we prune both train and test partitions and finally we apply the "Base Model" described in section 3 to assess its performance. In order to have more clues on the behavior of our technique, once learning has been performed on a training partition, we predict on the development and test sets as well. Figure 3 reports the results. Considering the curve related to the prediction on the split of the training set, pruning more than 35% of the nodes also increases MAP, with a peak of 69.41 when 75% of the nodes are

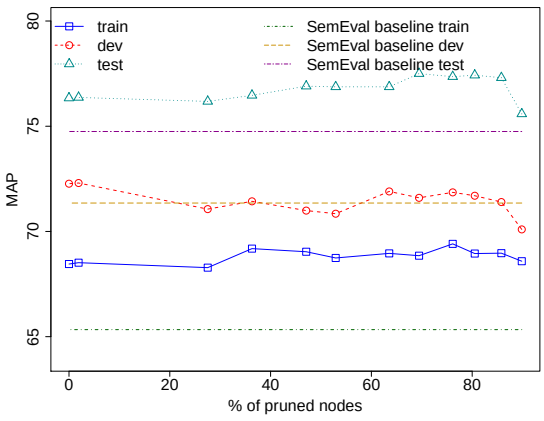
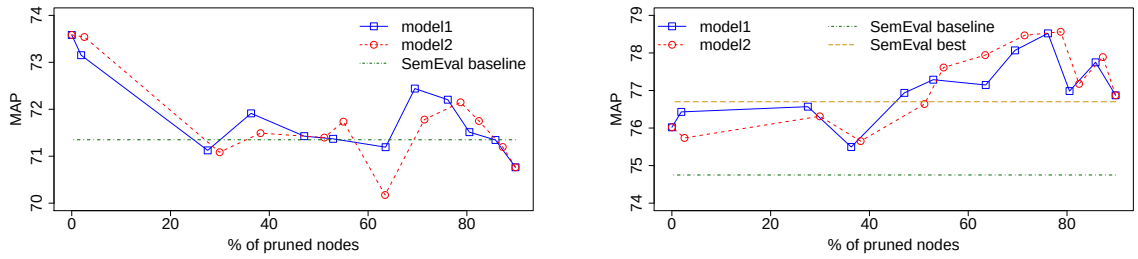


Figure 3: Cross validation for the pruning *model 1*. Each training fold is tested on the remaining part of the training set, the development and test sets.

Model	selection	Acc	P	R	F_1	MAP	AvgRec	MRR
Sentence *sim+pos*	6 sent	78.43	67.98	66.52	67.25	75.09	90.35	82.7
Pruning *model 2*	75% nodes pruned	80.57	69.96	72.96	71.43	**78.56**	91.39	85.12
Full text	-	78.71	68.58	66.52	67.54	76.02	90.70	84.64
SemEval Baseline	-	-	-	-	-	74.75	88.30	83.79
SemEval Best	-	76.57	63.53	69.53	66.39	76.70	90.31	83.02

Table 4: Performance of selected models on test.



(a) On the development set.

(b) On the test set.

Figure 4: MAP of base model with respect to the percentage of nodes pruned. Pruning performed with *model 1* and *model 2*.

pruned ($+0.958$ with respect to not applying pruning). The results on the development set do not show any increase in MAP. This may be due to annotation differences between the training/test and development partitions. At the same time the decrease is not significant. The behavior on the test set is similar to the one on the training set, although the peak is around 70% of nodes pruned this time (MAP=77.5, $+1.16$ w.r.t not applying pruning).

We performed further experiments by using all the training set for computing the weights $w()$ and, after pruning, for learning the "Base Model". This time we used the two models, *model 1* and *model 2*. Figure 4 shows the results on the development and test sets. Note that both on development and test sets there is not much difference between performing the pruning with *model 1* or *model 2*. The two plots show a similar pattern to the corresponding curves in Figure 3: on development there is no improvement, although there is a peak around 70% of nodes pruned, while on the test there is a notable improvement, the peak being MAP=78.56 ($+2.5$). Once again, our models manage to improve over SemEval's best (Franco-Salvador et al., 2016), even if it does not rely on any external knowledge. Note that the peak coincides with the threshold that we would select on cross validation on the training set, i.e. 75% of nodes pruned. We performed a student paired t-test (significance level 0.05) to check whether the MAP differences are statistically significant. It turns out that the best value on test allows us to have a statistically significant improvement with respect to the baseline.

The reduction of the size of the trees due to pruning, reduces significantly the running time, both in the learning and prediction phases. For example, the best model prunes 75% of the nodes, which translates into a reduction of the learning time of 12.7 times, from 134.51 to 10.53 minutes; the time for computing the predictions decreases from 15.14 to 9.22 minutes (1.64 times less).

Table 4 summarizes the results obtained with the different models. The tree kernel model itself, denoted as "Full text" in the table, improves over the SemEval baseline. Our best sentence selection model on development results in a worst performance on test. As aforementioned, it is not robust enough. Still it allows the pruning *model 2* system to improve by 1.86 MAP points the SemEval's best presented system.

6 Conclusions

Establishing question–question similarity for question ranking is of great importance in real-world community question answering tasks. While tree kernels are a key component of many state-of-the-art systems for question–question similarity, they are negatively affected by noisy and uninformative texts,

which are common for forum questions posted by web users. In this paper we studied the problem of selecting the most significant sentences and parse tree fragments from the question's text. For this purpose, we used unsupervised and supervised methods, exploiting standard cosine similarity models and we modeled supervised classifiers for learning effective sentence selection and tree fragments.

Our results on the recently-released SemEval 2016 cQA corpus show that supervised models can greatly improve the quality of text selection, thus reducing the size of the parse trees. An immediate consequence of this fact is that the prediction is faster (in our experiments prediction up to 50% and training more than 12 times), without any significant loss in MAP performance. In fact, it turns out that in most cases the structures removed might contain misleading information for the learning algorithm, therefore the MAP increased on the test set. Our proposed model outperforms the top systems submitted to the SemEval 2016 task on community Question Answering.

In the future, we would like to experiment with more advanced selection techniques, which have been shown successful in traditional text summarization; for instance, by using discourse structure.

Acknowledgements

This research was performed by the Arabic Language Technologies (ALT) group at HBKU's Qatar Computing Research Institute (QCRI), part of Qatar Foundation, within the Interactive sYstems for Answer Search project (Iyas).

References

Eugene Agichtein, Aristides Gionis, Carlos Castillo, Gilad Mishne, and Debora Donato. 2008. Finding high-quality content in social media with an application to community-based question answering. In *In Proceedings of WSDM*.

Fabio Aiolli, Giovanni Da San Martino, and Alessandro Sperduti. 2011. Extending Tree Kernels with Topological Information. volume 6791 of *Lecture Notes in Computer Science*, pages 142–149. Springer.

Lloyd Allison and Trevor Dix. 1986. A bit-string longest-common-subsequence algorithm. *Inf. Process. Lett.*, 23(6):305–310, December.

Alberto Barrón-Cedeño, Giovanni Da San Martino, Shafiq Joty, Alessandro Moschitti, Fahad Al-Obaidli, Salvatore Romeo, Kateryna Tymoshenko, and Antonio Uva. 2016. Convkn at semeval-2016 task 3: Answer and question selection for question answering on arabic and english fora. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 896–903, San Diego, California, June. Association for Computational Linguistics.

Yunbo Cao, Huizhong Duan, Chin-Yew Lin, Yong Yu, and Hsiao-Wuen Hon. 2008. Recommending questions using the mdl-based tree cut model. In *Proceedings of the 17th International Conference on World Wide Web*, WWW '08, pages 81–90, New York, NY, USA. ACM.

Giovanni Da San Martino, Alberto Barrón-Cedeño, Salvatore Romeo, and Alessandro Moschitti. 2016. Learning to re-rank questions in community question answering using advanced features. In *Proceedings of the 25th ACM International Conference on Information and Knowledge Management*, CIKM '16, Indianapolis, IN, October. Association for Computational Linguistics.

Cicero dos Santos, Luciano Barbosa, Dasha Bogdanova, and Bianca Zadrozny. 2015. Learning hybrid representations to retrieve semantically equivalent questions. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 694–699, Beijing, China, July. Association for Computational Linguistics.

Huizhong Duan, Yunbo Cao, Chin-Yew Lin, and Yong Yu. 2008. Searching questions by identifying question topic and question focus. In *ACL*, pages 156–164.

Simone Filice, Giuseppe Castellucci, Danilo Croce, Giovanni Da San Martino, Alessandro Moschitti, and Roberto Basili. 2015a. KeLP: a Kernel-based Learning Platform in java. In *The workshop on Machine Learning Open Source Software (MLOSS): Open Ecosystems*, Lille, France, July. International Conference of Machine Learning.

Simone Filice, Giovanni Da San Martino, and Alessandro Moschitti. 2015b. Structural representations for learning relations between pairs of texts. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1003–1013, Beijing, China, July. Association for Computational Linguistics.

Simone Filice, Danilo Croce, Alessandro Moschitti, and Roberto Basili. 2016. Kelp at semeval-2016 task 3: Learning semantic relations between questions and answers. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1116–1123, San Diego, California, June. Association for Computational Linguistics.

Marc Franco-Salvador, Sudipta Kar, Thamar Solorio, and Paolo Rosso. 2016. Uh-prhlt at semeval-2016 task 3: Combining lexical and semantic-based features for community question answering. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 814–821, San Diego, California, June. Association for Computational Linguistics.

Dan Gusfield. 1997. *Algorithms on Strings, Trees, and Sequences Computer Science and Computational Biology*. Cambridge University Press.

Yongshuai Hou, Cong Tan, Xiaolong Wang, Yaoyun Zhang, Jun Xu, and Qingcai Chen. 2015. Hitsz-icrc: Exploiting classification approach for answer selection in community question answering. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 196–202, Denver, Colorado, June. Association for Computational Linguistics.

Paul Jaccard. 1901. Étude comparative de la distribution florale dans une portion des Alpes et des Jura. *Bulletin del la Société Vaudoise des Sciences Naturelles*, 37(142):547–579.

Jiwoon Jeon, W. Bruce Croft, and Joon Ho Lee. 2005a. Finding semantically similar questions based on their answers. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '05*, page 617, New York, New York, USA, aug. ACM Press.

Jiwoon Jeon, W. Bruce Croft, and Joon Ho Lee. 2005b. Finding similar questions in large question and answer archives. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, CIKM '05, pages 84–90, Bremen, Germany.

Jiwoon Jeon, W. Bruce Croft, Joon Ho Lee, and Soyeon Park. 2006. A framework to predict the quality of answers with non-textual features. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '06*, page 228, New York, New York, USA, aug. ACM Press.

Zongcheng Ji, Fei Xu, Bin Wang, and Ben He. 2012. Question-answer topic model for question retrieval in community question answering. In *Proceedings of the 21st ACM international conference on Information and Knowledge Management*, pages 2471–2474. ACM.

Thorsten Joachims. 2006. Training linear svms in linear time. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, pages 217–226, New York, NY, USA. ACM.

Shafiq Joty, Alberto Barrón-Cedeño, Giovanni Da San Martino, Simone Filice, Lluís Màrquez, Alessandro Moschitti, and Preslav Nakov. 2015. Global thread-level inference for comment classification in community question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 573–578, Lisbon, Portugal, September. Association for Computational Linguistics.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 48–54, Stroudsburg, PA, USA. Association for Computational Linguistics.

Tie-Yan Liu. 2009. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331.

Caroline Lyon, James Malcolm, and Bob Dickerson. 2001. Detecting short passages of similar text in large document collections. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, EMNLP '01, pages 118–125, Pittsburgh, PA, USA.

Alessandro Moschitti. 2006a. Efficient Convolution Kernels for Dependency and Constituent Syntactic Trees. In Johannes Fürnkranz, Tobias Scheffer, and Myra Spiliopoulou, editors, *Machine Learning: ECML 2006*, volume 4212 of *Lecture Notes in Computer Science*, pages 318–329. Springer Berlin Heidelberg.

Alessandro Moschitti. 2006b. Making tree kernels practical for natural language learning. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 113–120.

Preslav Nakov, Lluís Màrquez, Walid Magdy, Alessandro Moschitti, James Glass, and Bilal Randeree. 2015. SemEval-2015 Task 3: Answer Selection in Community Question Answering. In *Proceedings of the 9th International Workshop on Semantic Evaluation*, SemEval '15, Denver, CO, USA.

Preslav Nakov, Lluís Màrquez, Alessandro Moschitti, Walid Magdy, Hamdy Mubarak, abed Alhakim Freihat, Jim Glass, and Bilal Randeree. 2016. Semeval-2016 task 3: Community question answering. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 525–545, San Diego, California, June. Association for Computational Linguistics.

Massimo Nicosia, Simone Filice, Alberto Barrón-Cedeño, Iman Saleh, Hamdy Mubarak, Wei Gao, Preslav Nakov, Giovanni Da San Martino, Alessandro Moschitti, Kareem Darwish, Lluís Màrquez, Shafiq Joty, and Walid Magdy. 2015. QCRI: Answer selection for community question answering - experiments for Arabic and English. In *Proceedings of the 9th International Workshop on Semantic Evaluation*, SemEval '15, Denver, Colorado, USA.

Salvatore Romeo, Giovanni Da San Martino, Alberto Barrón-Cedeño, Alessandro Moschitti, Yonatan Belinkov, Wei-Ning Hsu, Yu Zhang, Mitra Mohtarami, and James Glass. 2016. Neural attention for learning to rank questions in community question answering. In *Proceedings of the 26th International Conference on Computational Linguistics*, Osaka, Japan.

Aliaksei Severyn and Alessandro Moschitti. 2012. Structural relationships for large-scale learning of answer reranking. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '12, pages 741–750, Portland, Oregon, USA.

Jun Sun, Min Zhang, and Chew Lim Tan. 2011. Tree sequence kernel for natural language. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, AAAI'11, pages 921–926. AAAI Press.

Mihai Surdeanu, Massimiliano Ciaramita, and Hugo Zaragoza. 2008. Learning to rank answers on large online QA collections. In *Proceedings of ACL-08: HLT*, pages 719–727, Columbus, Ohio, June. Association for Computational Linguistics.

Quan Hung Tran, Vu Tran, Tu Vu, Minh Nguyen, and Son Bao Pham. 2015. Jaist: Combining multiple features for answer selection in community question answering. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 215–219, Denver, Colorado, June. Association for Computational Linguistics.

Kai Wang, Zhaoyan Ming, and Tat-Seng Chua. 2009. A syntactic tree matching approach to finding similar questions in community-based qa services. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 187–194. ACM.

Michael Wise. 1996. Yap3: Improved detection of similarities in computer program and other texts. In *Proceedings of the Twenty-seventh SIGCSE Technical Symposium on Computer Science Education*, SIGCSE '96, pages 130–134, New York, NY, USA.

Kai Zhang, Wei Wu, Haocheng Wu, Zhoujun Li, and Ming Zhou. 2014. Question retrieval with high quality answers in community question answering. In *CIKM*.

Guangyou Zhou, Li Cai, Jun Zhao, and Kang Liu. 2011. Phrase-based translation model for question retrieval in community question answer archives. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 653–662. Association for Computational Linguistics.

Guangyou Zhou, Tingting He, Jun Zhao, and Po Hu. 2015. Learning continuous word embedding with metadata for question retrieval in community question answering. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 250–259, Beijing, China, July. Association for Computational Linguistics.