SLS at SemEval-2016 Task 3: Neural-based Approaches for Ranking in Community Question Answering

Mitra Mohtarami, Yonatan Belinkov, Wei-Ning Hsu, Yu Zhang
Tao Lei, Kfir Bar†, Scott Cyphers, James Glass
MIT Computer Science and Artificial Intelligence Laboratory
†Tel Aviv University
{mitra, belinkov, wnhsu, yzhang87, taolei, cyphers, glass}@csail.mit.edu †kfirbar@post.tau.ac.il

Abstract

Community question answering platforms need to automatically rank answers and questions with respect to a given question. In this paper, we present the approaches for the Answer Selection and Question Retrieval tasks of SemEval-2016 (task 3). We develop a bag-of-vectors approach with various vector- and text-based features, and different neural network approaches including CNNs and LSTMs to capture the semantic similarity between questions and answers for ranking purpose. Our evaluation demonstrates that our approaches significantly outperform the baselines.

1 Introduction

Community Question Answering (cQA) forums are rapidly growing, resulting in an urgent need to automatically search for relevant answers among many responses provided for a given question (Answer Selection), and search for relevant questions to reuse their existing answers (Question Retrieval). In this paper, we aim to address the SemEval 2016 tasks (Nakov et al., 2016) that are designed for Answer Selection (AS) and Question Retrieval (QR). These tasks are briefly described as follows:

A Question-Comment Similarity: given a question and its first 10 comments in the question thread, rerank these 10 comments according to their relevance with respect to the question.

B Question-Question Similarity: given a new question (named original question) and the set of the first 10 related questions retrieved by a search engine, rerank the related questions according to their similarity regarding the original question.

C Question-External Comment Similarity: given a new question (original question) and the set of the first 10 related questions retrieved by a search engine, each associated with its first 10 comments appearing in its thread, rerank the 100 comments (10 questions x 10 comments) according to the new question.

D Question-External Question-Comment Pair Similarity: Given a new question and a set of 30 related questions retrieved by a search engine, each associated with one correct answer, rerank the 30 question-comment pairs according to their relevance with respect to the original question.

Task B is considered as QR and the others as AS problems. The first three tasks are evaluated on an English dataset and the fourth on an Arabic dataset. Several factors make all these tasks more challenging. First, cQA forums contain open-domain and non-factoid questions and answers, resulting in high variance Q&A quality (Márquez et al., 2015). A second factor is that the Q&A are long and their length may vary from several words to several hundred words. The third factor concerns the relatively close relation between some annotation labels; the comments in the tasks A and C are labeled as Relevant, Potential and Irrelevant, and the Relevant comments need to be ranked above the Potential and Irrelevant comments. From a natural language processing perspective, it is difficult to define a clear distinction between the relevant and potential labels.
To address these tasks, we first present a bag-of-vectors (BOV) approach in which various vector- and text-based features are designed and passed through a linear SVM classifier to compute the degree of relatedness between the Q&As. Then, we present different NN-based approaches including CNNs and LSTMs to compute the representations of the Q&As. We evaluate our models on the cQA corpus provided by SemEval. The results demonstrate that our approaches outperform the baselines.

2 Method

Given a question \( q \), a list of answers \( A \) for AS, and a list of questions \( Q \) for QR, we aim to rank the lists \( A \) and \( Q \) with respect to \( q \). To address these problems, we present a bag-of-vectors (BOV) approach to compute various vector- and text-based features for a classifier. Furthermore, we present NN-based approaches (LSTM with attention, CNN and RCNN) for learning the vector representations of the questions and answers to be used for capturing their semantic similarity. The degree of similarity between the Q&A is used for their ranking.

2.1 Bag-of-Vectors (BOV)

Previous work presented a BOV approach to address the classification tasks in cQA (Belinkov et al., 2015). In this paper, we aim to extend the previous approach for the ranking tasks by updating the feature sets and developing new models. The features are categorized into text, vector and meta-data based features that are briefly explained below (in the experiments section below we detail the features chosen for each task). Then, we explain our approach to shorten the length of Q&As in the Arabic data.

Text-based features These features are mainly computed using text similarity metrics, word clustering and topic modeling. As the first set of text-based features, we use various text similarity metrics that measure string overlap between Q&As: Longest Common Substring (Gusfield, 1997), Longest Common Subsequence (Allison and Dix, 1986), Longest Common Subsequence Norm, Greedy String Tiling (Wise, 1996), Monge Elkan Second String (Monge and Elkan, 1997), Jaro Second String (Jaro, 1989), Jaccard coefficient (Lyon et al., 2004) and Containment measures (Broder, 1997). These metrics are explained in (Belinkov et al., 2015).

Another set of text-based features are computed using word clustering that has been useful in many supervised NLP approaches. We use Brown clustering (Brown et al., 1992; Liang, 2005) that creates word clusters such that they are hierarchical in a binary tree. In the tree, each word is assigned to a bitstring depending on its tree path, and the prefixes of the bitstring are the ancestor clusters used as additional features. We use an implementation of Brown clustering,\(^1\) that is designed as an HMM-based algorithm which partitions words into a base set of N (=500) clusters. Given a question or an answer as a document, its clusters are determined based on its word clusters. This captures the global clusters.

Topic modeling approaches can also be used to automatically identify topics of documents. We use Non-negative Matrix Factorization (NMF) for topic modeling. A document-term matrix is constructed with TF-IDF weights. This matrix is factored into a term-topic and a topic-document matrix. The N (=100) topics are derived from the contents of the documents, and the topic-document matrix describes topics of related documents. We use each column of the topic-document matrix as features for each individual document. The entire train, development and test datasets provided by SemEval 2015 and 2016 are employed to compute the word clustering and topic modeling features.

Vector-based features The concatenation of the normalized Q&A representations is used as vector-based features for a \((q, a)\) pair. The question or answer representation is obtained with the average of its word representations computed from Word2Vec vectors (Mikolov et al., 2013a; Mikolov et al., 2013b; Mikolov et al., 2013c). For English word vectors we use the GoogleNews vectors dataset.\(^2\) For Arabic word vectors we use Word2Vec to train 100-dimensional vectors on either the general domain Arabic Gigaword (Linguistic Data Consortium, 2011) or the domain-specific raw data provided with the task. We select the word vector set based on the performance of the development set.

\(^1\)https://github.com/percyliang/brown-cluster.
\(^2\)https://code.google.com/p/word2vec.
Furthermore, for the Arabic the pair of sentence vectors in the question and answer with the highest cosine similarity is used as features.

We use a zero vector if the question or answer contains only out-of-vocabulary words. To make it easier for the classifier to ignore the vectors in these cases, we design two boolean features to identify whether the question and answer are zero vectors.

**Metadata-based features** We use a metadata-based feature that identifies whether the user who posted the question is the same user who wrote the answer. This feature is useful to identify irrelevant dialogue answers, and used for the tasks A and C.

**Shrinking the sentence length** Some of the questions and answers in the community forum are very long. In fact, in the Arabic dataset questions and answers have an average length of 50 and 120 words, respectively. Therefore, we preprocess the texts using TextRank (Mihalcea and Tarau, 2004), a graph-based keyword extraction algorithm, for finding the most meaningful words within every thread. Once the meaningful words are found, we filter all other words from each thread instance, and build the subsequent feature representation based on the shortened texts.

Given a document, TextRank builds a graph representation, where nodes stand for word types, connected by undirected links representing co-occurrence within a window of size $N$. An importance weight is then calculated for each node, using an iterative formula introduced by PageRank (Brin and Page, 1998). We use our implementation of TextRank, in which we select a certain percentage of the words, defined as $P$, sorted top-down by importance weight, as the final keywords.

We treat each thread, including all its question-answer pairs, as an individual document for TextRank. We preprocess each document with MADA 3.1 (Habash et al., 2009), a context-sensitive lemmatizer, for finding word lemmas and part-of-speech tags. Finally, our TextRank graphs include only lemmas of content words, Latin-script words and words with no lemmas. Content words in this sense are defined as nouns, verbs, adjectives and adverbs. All other TextRank parameters are assigned with values according to (Mihalcea and Tarau, 2004).

2.2 Long Short-Term Memory (LSTM) Networks with Attention

LSTMs have shown great success in many different fields, such as textual entailment (Rocktäschel et al., 2015), language modeling (Sundermeyer et al., 2012), and acoustic modeling (Graves et al., 2013). The recurrent structure as well as the ability to store long-term information make LSTMs suitable for encoding sequences of variable length into fixed-length representations.

However, for very long sequences, such as the comments in cQA tasks (~hundreds of words), an LSTM may still fail to compress all information into this representation. Recently, a neural attention model (Bahdanau et al., 2014) has been proposed to alleviate this issue by enabling the network to attend to all past outputs. The attention mechanism along with an LSTM is ideal for cQA tasks.

Following (Mohtarami et al., 2016), as illustrated in Figure 1, we apply two LSTMs to encode $(q, q')$ or $(q, a)$ respectively. The first LSTM reads one object, and passes information through hidden units to the second LSTM. The second LSTM then reads the other object and generates its representation biased by the first object after finishing reading.

By augmenting an attention mechanism to the encoder, we allow the second LSTM to attend to the sequence of output vectors from the first LSTM, and hence generate a weighted representation of the first object according to both objects. Let $h_N$ be the last output of the second LSTM and $M = [h_1, h_2, \ldots, h_L]$ be the sequence of output vectors of the first object. The weighted representation of the first object is

$$h' = \sum_{i=1}^{L} \alpha_i h_i$$

(1)
Table 1: Hyper-parameters of the LSTM model. The bold value is the selected parameter.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Embedding</td>
<td>init or random, fix or update</td>
</tr>
<tr>
<td>Two LSTM</td>
<td>shared or not</td>
</tr>
<tr>
<td># cells for LSTM</td>
<td>64, 128, 256</td>
</tr>
<tr>
<td># nodes for MLP</td>
<td>128, 256</td>
</tr>
<tr>
<td>Optimizer</td>
<td>AdaGrad, AdaDelta, SGD</td>
</tr>
<tr>
<td>learning rate</td>
<td>0.001, 0.01, 0.1</td>
</tr>
<tr>
<td>Regularizer</td>
<td>Dropout, L2 regularization</td>
</tr>
<tr>
<td>Dropout rate</td>
<td>0.0, 0.2, 0.3, 0.4, 0.5</td>
</tr>
<tr>
<td>L2</td>
<td>0, 0.001, 0.0001, 0.00001</td>
</tr>
</tbody>
</table>

The weight is computed by

$$\alpha_i = \frac{\exp(a(h_i, h_N))}{\sum_{j=1}^{L} \exp(a(h_j, h_N))}$$  \hspace{1cm} (2)

where $a()$ is the importance model that produces a higher score for $(h_i, h_N)$ if $h_i$ is useful for determining the object pair’s relationship. We parametrize this model using a feed-forward neural network.

To classify the relationship of this pair, another feed-forward neural network is built on top of the LSTMs that takes the representations of both objects, $h_N$ and $h'$, as input. Note that in our framework, we can use the augmented features $f$ to enhance the classifier. In this case, the final input to the classifier will become $h_N, h'$, and $f$. The details of this model are explained in (Mohtarami et al., 2016).

Our system is based on Theano (Bastien et al., 2012; Bergstra et al., 2010). Table 1 gives a list of hyperparameters we tried. As suggested by (Greff et al., 2015), the hyperparameters for an LSTM can be tuned independently. We tune each parameter separately on a dev set and pick the best one.

### 2.3 Convolutional Neural Network (CNN)

Convolutional Neural Networks (CNNs) are useful in many NLP tasks, such as language modeling (Kalchbrenner et al., 2014), semantic role labeling (Collobert and Weston, 2008) and semantic parsing (Yih et al., 2014). Our reason for using a CNN for cQA is that it can capture both features of n-grams and long-range dependencies (Yu et al., 2014), and can extract discriminative word sequences that are common in the training instances (Severyn and Moschitti, 2015). These traits make CNNs useful for dealing with long questions.

Following (Mohtarami et al., 2016), as illustrated in Figure 2, we employ a CNN-based model to first compute a relatedness score for each pair, $(q, a)$ or $(q, q)$, and then rank the lists based on the resulting scores. In the model, for a given pair, the embedding vectors of $q$ and $a$ are considered as input. The CNN convolution and pooling layers then generate the convolutional vector representations. These vectors are concatenated with other additional feature vectors and used as input to a fully connected Multi-Layer Perceptron (MLP) whose softmax layer generates a probability score $P(y|q,a)$ over the labels $y \in \{0, 1\}$, where 1 means relevant, and 0 means irrelevant. The hyperparameter configuration of the CNN model is shown in Table 3, and the details of this model are explained in (Mohtarami et al., 2016).

### 2.4 Recurrent Convolutional Neural Network (RCNN)

For task B, we also apply the recurrent convolutional neural network model, which has been recently proposed and successfully applied to a similar question retrieval problem (Lei et al., 2015). Unlike traditional CNNs which only extract local n-gram features, RCNNs extract and aggregate all possible n-grams within the input sequence, including ones that are not consecutive. Similar to LSTMs and Gated Recurrent Units (GRUs), which have internal “memory” states, RCNNs maintain aggregated vectors to store the weighted average of n-gram features. These vectors are updated in a recurrent fashion when the input tokens are successively read into the network.

Following the set-up in (Lei et al., 2015), we take the last state vector as the final representation of the question. The parameters of RCNN encoder are trained in a max-margin fashion, maximizing the
3 Experimental Results

We evaluate our approaches on all the cQA tasks. We use the cQA datasets provided by SemEval 2016. The English data were collected from the Qatar Living forum and the Arabic data were collected from medical forums. Table 4 provides statistics for the datasets. As evaluation metrics, we use F1-score for a global assessment of the approaches in addition to the following ranking metrics: Mean Average Precision (MAP), Average Recall (AveRec) and Mean Reciprocal Rank (MRR). For the MAP, we use the average of MAP@1 to MAP@10.

Baselines For a baseline, we use the Information Retrieval (IR) ranking score that is computed as follows: given a $q$, the top 100 threads retrieved by Google from the Qatar Living forum are considered and the order of each thread is used as its IR ranking score. As another baseline, we use a system that randomly ranks a given list of Q or A.

Question-comment similarity The results for this task are shown in Table 5(a). The first two rows are the IR and random baseline results, and the next two rows are the best two performances among all SemEval submissions for this task. The other three rows are the results of our approaches and respectively submitted to SemEval as primary, contrastive1 and contrastive2 results. As shown in the table, our results are significantly higher than the baselines and comparable with the best results over all performance metrics, and there is no significant difference between the results of our approaches for this task. We use various combinations of our BOV, LSTM and CNN approaches, then select the best ones with respect to the development set. The combination of the approaches is computed by $1/(R_1 + R_2 + \ldots + R_i)$ where $R_i$ is the ranking of the $i^{th}$ approach.

In this task, the BOV includes all the features except for the NMF features, and we employ the order of the answers in their threads as augmented features for our NN-based approaches. The structure of the threads (e.g., answer order) can help to extract relevant answers (Barrón-Cedeño et al., 2015).

Question-question similarity Table 5(b) shows the results for this task. The first two rows are the results for IR and random baselines, and the next two are the best two performances of SemEval. The other three results are related to our approaches and respectively submitted to SemEval as primary, contrastive1 and contrastive2 results. The table shows that our results are significantly better than the baselines. While there is no significant difference between our contrastive1 and contrastive2 results with the best result, these results are higher than the second best SemEval result on MAP, and the highest result is obtained with our primary result on accuracy. With respect to our results, the combination of BOV, LSTM and RCNN achieves the highest result on MAP and the combination of BOV and RCNN is the best on F1.

In this task, the combination of the approaches is computed using a linear SVM with the feature vector $R_1, R_2, \ldots, R_i$ where $R$ is the ranking of the $i^{th}$ approach. Furthermore, in this experiment, the BOV includes all the features except the word clustering features, and we employ the ranking order of the IR as augmented features for our NN-based approaches.

Table 4: Statistics of the dataset through the tasks.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original questions</td>
<td>-</td>
<td>317</td>
<td>317</td>
<td>1,281</td>
</tr>
<tr>
<td>Related questions</td>
<td>6,398</td>
<td>3,169</td>
<td>3,169</td>
<td>37,795</td>
</tr>
<tr>
<td>Comments</td>
<td>40,288</td>
<td>-</td>
<td>31,690</td>
<td>37,795</td>
</tr>
</tbody>
</table>

Table 2: The hyper-parameters of RCNN model.

Table 3: The hyper-parameters of CNN model.

<table>
<thead>
<tr>
<th>Embedding</th>
<th>word2vec, fixed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hidden dimension</td>
<td>300</td>
</tr>
<tr>
<td>Filter width</td>
<td>5</td>
</tr>
<tr>
<td>Optimizer</td>
<td>AdaDelta</td>
</tr>
<tr>
<td>Learning rate</td>
<td>0.95</td>
</tr>
<tr>
<td>Dropout rate</td>
<td>0.5</td>
</tr>
<tr>
<td>L2</td>
<td>0.000001</td>
</tr>
</tbody>
</table>

(cosine) similarity difference between positive question pairs and negative pairs. The hyperparameter configuration of the model is shown in Table 2.

---

### Table 5: Results on test data for answer selection and question retrieval tasks

#### Question-external comment similarity

The results for this task are shown in Table 5(c). The first two rows are the IR and random baseline results and the next two rows are the best two SemEval results. The other three rows are our results that respectively are primary, contrastive1 and contrastive2 results. As shown in the table, our results are significantly higher than the baselines but lower than the best SemEval results. We use a similar combination approach to task A for our contrastive results and the primary is computed using the BOV and IR features as the augmented features for LSTM. In this task, the BOV includes all the features except for the word clustering and NMF features, and we employ both the ranking order of the IR and answer order as augmented features for our NN-based models.

#### Question-external question-comment pair similarity

This task is only available for Arabic. Our feature set for this task is somewhat simplified compared to the English tasks: we only use our BOV approach with simple text- and vector-based features. similarities are computed only on word- and sentence-level, and not on chunk-level as in (Belinkov et al., 2015). We do not use word clustering or topic modeling features and we note that the Arabic dataset has no associated metadata. Furthermore, in this dataset every original question has a number of related question-answer pairs. To fully exploit this information we compute two sets of features: one between the original and related questions, and one between the original question and the related answer. We then concatenate the two sets as
the final feature representation to the classifier.

Our primary submission is a uniform combination of scores from four different settings of shrinking the length: (i) no shrinking (all words are kept as is); (ii) only keeping content lemmas (iii) only content lemmas and TextRank with $N = 3$, $P = 5$; (iv) only content lemmas and TextRank with $N = 4$, $P = 1$. We also submit (i) as contrastive1 and (iii) as contrastive2. These settings were chosen based on the performance on the development set.

As Table 5(d) shows, our primary submission ranks first on all ranking metrics and on F1. Our contrastive submissions are also very competitive.

Finally, we experimented with two sets of word vectors, either trained from the general domain Gigaword corpus ($\sim 1B$ words) or the domain-specific unsupervised data provided with the task ($\sim 26M$ words). Despite the very different sizes of the raw corpora, we found mixed results: the general domain vectors were useful with no shrinking (i) while the domain-specific ones were more beneficial with shrinking (ii-iv); we used these settings for the submission. Using word vectors trained on a combined corpus from both raw datasets did not result in additional improvement.

4 Conclusion

We developed bag-of-vectors and neural network approaches, and demonstrated their effectiveness on the cQA tasks for ranking a list of questions or answers for a given question. We evaluated our approaches on the SemEval-2016 corpus and our results significantly outperform the baselines. In addition, our results are comparable to the result of the best submission to SemEval-2016 for English and achieved the first place for Arabic.

Acknowledgments

This research was supported by the Qatar Computing Research Institute (QCRI). We would like to thank Alessandro Moschitti, Preslav Nakov, Lluís Márquez, and other members of the QCRI Arabic Language Technologies group for their collaboration on this project.

References


Mita Mohtarami, Wei-Ning Hsu, Yu Zhang, and James Glass. 2016. Answer Selection and Question Retrieval with Neural Networks. Under review.


