TakeLab-QA at SemEval-2017 Task 3: Classification Experiments for Answer Retrieval in Community QA

Filip Šaina, Toni Kukurin, Lukrecija Puljić,
Vanja Mladen Karan, Jan Šnajder
Text Analysis and Knowledge Engineering Lab
Faculty of Electrical Engineering and Computing, University of Zagreb
Unska 3, 10000 Zagreb, Croatia
{name.surname}@fer.hr

Abstract

In this paper we present the TakeLab-QA entry to SemEval 2017 task 3, which is a question-comment re-ranking problem. We present a classification based approach, including two supervised learning models – Support Vector Machines (SVM) and Convolutional Neural Networks (CNN). We use features based on different semantic similarity models (e.g., Latent Dirichlet Allocation), as well as features based on several types of pre-trained word embeddings. Moreover, we also use some hand-crafted task-specific features. For training, our system uses no external labeled data apart from that provided by the organizers. Our primary submission achieves a MAP-score of 81.14 and F1-score of 66.99 – ranking us 10th on the SemEval 2017 task 3, subtask A.

1 Introduction

The ever-growing *Community Question Answering* (CQA) on-line services are gaining popularity at an increasing rate. However, there are some problems inherent to question-answer collections created by on-line communities. A major issue is the sheer volume of CQA collections, which makes finding an answer to a user question infeasible without some kind of an automated retrieval system. Consequently, information retrieval on CQA collections has gained increased focus in the research community, giving rise to several shared tasks on SemEval (Nakov et al., 2017).

From a natural language processing perspective, this is a difficult task due to high variance in the quality of questions and answers in CQA collections (Màrquez et al., 2015). The cause of this is the self-moderated nature of CQA sites, which implies that there are few restrictions on who is allowed to answer a question.

In this paper we describe our entries for the SemEval 2017 Question-Comment Similarity subtask (Nakov et al., 2017). Given a question q and a comment list C, the task is to rank the comments in C according to their relevance with respect to q. Datasets for this task were extracted from Qatar Living, a web forum where people pose questions about various aspects of their daily life in Qatar.

Following Filice et al. (2016), we framed the task as a binary classification problem. We experimented with two classification approaches – Support Vector Machines (SVM) (Cortes and Vapnik, 1995) and Convolutional Neural Networks (CNN) (Kim, 2014). Most of the features we use follow the work of Barrón-Cedeno et al. (2015) and Nicosia et al. (2015). Moreover, we use embedding-based (Mihaylov and Nakov, 2016) features for both models.

The CNN model with the full feature set has proven to be the most successful, ranking 10th in the competition with a MAP-score of 81.14 and an F1-score of 66.99.

The rest of this paper is structured as follows. Section 2 gives a brief overview of the data set. A detailed description of the our models is given in Section 3. Section 4 outlines our experiments and results. Finally, we present our conclusions in Section 5.

2 Dataset

The dataset we used was provided by the shared-task organizers. Incoming user queries are denoted as *original questions*. For each original question, we are provided with 10 annotated threads. Each thread consists of a *related question*[1] and a set of

[1] Related questions are questions that have been asked in the past, and have a set of posted comments.

	relevant	*non-relevant*	*% relevant*
training	14,418	18,872	43.3
dev	2,198	3,152	41.1
overall	16,616	22,024	43.0

Table 1: Class distribution statistics.

10 comments posted as answers to the related question. Subtask A, the main focus of this work, is concerned with correctly ranking the 10 comments with respect to a related question (henceforth: question). Every *question-comment pair* contains a classification label *Good*, *Bad*, or *PotentiallyUseful*. In our experiments, labels *Bad* and *PotentiallyUseful* are both considered *non-relevant* and the label *Good* is considered *relevant*. Table 1 shows the class distribution of *question-comment* pairs in the shared task dataset. There is a slight bias towards the *non-relevant* class, but no mechanisms were implemented to address this. The official split allocates 86% of the data to the train set and 14% to the development set. For further information on the collection we refer to (Nakov et al., 2017).

3 System Description

Considering we have class labels available for each question-comment pair, it is natural to frame this ranking task as a supervised classification problem. The input of the classifier is a vector of features that represents the question-comment pair (q, c). The output is a class – *relevant* or *non-relevant* – indicating whether c is relevant with respect to q. We have experimented with two supervised approaches for classifying the data, SVM and CNN, which were shown to be very successful in question-comment re-ranking tasks in previous work (Nakov et al., 2016). Note that both of these variants of our system fall into the *pointwise* category of the *learning-to-rank* paradigm (Cao et al., 2007). We next describe all our systems' components in detail.

3.1 Preprocessing

We have preprocessed all entries by tokenizing them, stemming the tokens, and removing stopwords using the NLTK toolkit.[2] These tokens were used as input to the feature extraction pipeline.

[2]http://www.nltk.org

3.2 Embedding-Based Features

As observed by Socher et al. (2011), in the case where a large training set is not available, using word embeddings obtained from an unsupervised language model can be an efficient method for improving performance. More specifically, in our scenario embeddings alleviate lexical gap that often arises when comparing a question to a relevant comment.

We use two types of embeddings. First, *word2vec* embeddings trained on Quatar Living questions and comments that was used by last year's participants (Mihaylov and Nakov, 2016). We used 800-dimensional word vectors.[3] Secondly, we use the *PARAGRAM*[4] model introduced by Wieting et al. (2016), which produces 300-dimensional word embeddings specifically tuned to work well when aggregated by a simple operation such as the average.

After some experimentation, we found the following embedding features offered the best performance boosts on the development set:

- **Question and Answer Embeddings** – using the *PARAGRAM* embeddings, we compute the vector representation of both the question and comment by averaging their corresponding content-word vectors. This yields two 300-dimensional vectors, which are fed into our models as 600 numerical features;

- **Word2vec average cosine similarity** – we compute vector representations of the question and comment in the same manner as for the previous feature, but using *word2vec* word vectors. We then introduce into our models a single numerical feature computed as the cosine similarity of the question and comment vector representations.

3.3 SEMILAR Features

Features listed under this group were all obtained from SEMILAR[5] (Rus et al., 2013). This is a library that implements a multitude of popular semantic similarity measures for text. We use it to calculate the similarity of the question to the candidate comment, and include each measure as a

[3] These vectors alone can produce a MAP-score of 78.45 on subtask A, and can be obtained from https://github.com/tbmihailov/semeval2016-task3-cqa

[4]We use the SL999 variant available at – http://ttic.uchicago.edu/~wieting/

[5]http://deeptutor2.memphis.edu/Semilar-Web/

numerical feature. In our experiments we include similarity measures based on:

- Lexical overlap – two variants, overlap before preprocessing and overlap after preprocessing;
- WordNet (Fellbaum, 1998);
- Latent Semantic Analysis (Landauer et al., 1998);
- Latent Dirichlet Allocation (Blei et al., 2003);
- BLEU (Papineni et al., 2002);
- Meteor (Denkowski and Lavie, 2011);
- Pointwise Mutual Information (Church and Hanks, 1990).

3.4 Other Features

In this group we list hand-crafted features, as well as features that do not fit into the previous two groups:

- **Words contained in comment** – we have tested each comment for some words that often seem to be useful in distinguishing good from bad comments in the previous editions of the task (Barrón-Cedeno et al., 2015). More specifically, we checked whether the comment contains words *yes, ok, sorry, no, sure*, or *can*, and encoded them as binary numerical features;
- **Answer contains question mark** – another binary numerical feature that has previously been proven useful is whether the comment body contains a question mark (Barrón-Cedeno et al., 2015);
- **Answer length** – longer comments tend to be better thought-out and, consequently, more useful with respect to the question at hand. This numerical feature represents the number of words in the comment after preprocessing;
- **Tf-Idf cosine similarity** – we determine *Tf-Idf* vectors for each question and comment, respectively. We then compute the cosine similarity of these vectors and include it as a numerical feature for our models.

3.5 SVM

We use the SVM classifier (Cortes and Vapnik, 1995) from ScikitLearn.[6] The relevance score of comment c with respect to question q is the confidence of the SVM classifier that the class is *relevant*, when presented (q, c) as input.

3.6 CNN

We follow the work of Severyn and Moschitti (2015). The overall architecture of the network includes two convolutional layers and the corresponding information flow:[7]

- The question q and comment c at the input are represented as matrices containing the *word2vec* embeddings of their words. They are the input of two separate convolutional layers that perform feature extraction;
- The max-pooling operation is applied to the resulting feature-maps;
- This results in task-specific representation vectors of the question and the comment. These vectors are concatenated and fed into a fully connected hidden layer, along with other features that we wish to include. Note that the extra features could be especially helpful in cases where many words in q and c that are not covered by the *word2vec* model, which may lead to meaningless features extracted by the convolutional layers;
- Finally, a fully connected softmax layer calculates the probability distribution over the output label.

The network is trained by mini-batch stochastic gradient descent, minimizing the cross-entropy loss. To combat overfitting we use both early stopping and dropout (Srivastava et al., 2014).

The score of comment c with respect to question q is the probability that the network assigns to the *relevant* class when presented with (q, c) as input.

4 Evaluation

4.1 Hyperparameters

We optimizeed the hyperparameters of the SVM classifier using cross-validation on the train set. We

[6] `http://scikit-learn.org/`

[7] The most notable difference is that we ommit the the similarity matrix present in their network.

varied the C and γ parameters [8], and experimented with both linear and non-linear kernels. We found that the RBF kernel with C set to 5 and γ set to 0.01 was the best performing combination.

For CNN hyperparameters we used as a starting point the values proposed by Kim (2014), and then tuned them empirically, to find the setting that optimizes the performance on the development set. More specifically, we used filter windows of size 3, 4, 5, with 64 feature maps for each window, a dropout rate of 0.7, and mini-batch size of 64.

4.2 Evaluation

Participants were allowed to make three submissions and mark them as *primary*, *contrastive1*, or *contrastive2*. All submissions were evaluated but only the *primary* was considered for the competition system ranking.

For our *contrastive1* run we submitted an SVM classifier trained on features from the *embedding-based* and *other* groups. We denote this model as *SVM-EO*.

In our *contrastive2* run we submitted a completely different combination: the CNN classifier with the *SEMILAR* features as additional input to the hidden layer. We refer to this variant of the model as *CNN-S*.

The last combination we considered was the CNN classifier with all the other features provided as additional input to the hidden layer. We refer to this submission as *CNN-EOS*. This model has access to both the feature representations generated by the convolutional layers, as well as to all other features. Thus, expectedly, it performed best on the development set,[9] and we decided to submit it as our *primary* run.

4.3 Results

Final evaluation results released by the shared-task organizers are shown in Table 2. [10] Our primary submission was ranked at 10th place, achieving a MAP-score of 81.14.

Our contrastive submissions, both SVM and CNN based, achieve comparable performances. The CNN seems to outperform SVM only when new features are added, suggesting that the features work best when used jointly.

	MAP	F1	Acc
Best system (KeLP)	88.43	69.87	73.89
CNN-EOS (primary)	**81.14**	66.99	**70.14**
SVM-EO	79.71	**67.87**	69.11
CNN-S	78.98	62.54	**70.14**
Baseline 1 (IR)	72.61	–	–
Baseline 2 (random)	62.30	66.36	52.70

Table 2: Submission results summary.

Furthermore, while the CNN model did prove to be better on the MAP metric, SVM outperforms it on the F1-score. This indicates that maximizing F1-score may not always maximize MAP-score.

5 Conclusion

In this paper we presented a re-ranking system developed to participate in SemEval 2017 Task 3 – Community Question Answering. The system consists of two supervised learning approaches – Support Vector Machines (SVM) and Convolutional Neural Networks (CNN), which outperformed the baseline model. While our models were trained only on the official dataset, better results may have been attained with additional training data.

For future work we would like to include additional features, such as tree kernels (Filice et al., 2016) and sentiment-specific word embeddings proposed by Tang et al. (2014). We would also like to experiment with different classification models. An intriguing venue would be obtaining more training data from other popular CQA sites, and explore if some cross-domain information transfer that could benefit ranking models is possible. Finally, as indicated by our experiments, we would like to consider approaches that optimize the ranking model explicitly for the MAP-score.

References

Alberto Barrón-Cedeno, Simone Filice, Giovanni Da San Martino, Shafiq R Joty, Lluís Màrquez, Preslav Nakov, and Alessandro Moschitti. 2015. Thread-level information for comment classification in community question answering. In *Proceedings of the meeting of the Association for Computational Linguistics*. pages 687–693.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3(Jan):993–1022.

[8] γ only for the RBF kernel.

[9] Prediction accuracy – number of correct predictions over all pairs, is used as a overall training and development performance metric.

[10] http://alt.qcri.org/semeval2017/task3/data/uploads /semeval2017_task3_results.pdf

Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*. ACM, pages 129–136.

Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational linguistics* 16(1):22–29.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning* 20(3):273–297.

Michael Denkowski and Alon Lavie. 2011. Meteor 1.3: Automatic metric for reliable optimization and evaluation of machine translation systems. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*. Association for Computational Linguistics, pages 85–91.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database.*. Bradford Books.

Simone Filice, Danilo Croce, Alessandro Moschitti, and Roberto Basili. 2016. Kelp at SemEval-2016 task 3: Learning semantic relations between questions and answers. *Proceedings of SemEval* 16:1116–1123.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Thomas K Landauer, Peter W Foltz, and Darrell Laham. 1998. An introduction to latent semantic analysis. *Discourse processes* 25(2-3):259–284.

Lluís Màrquez, James Glass, Walid Magdy, Alessandro Moschitti, Preslav Nakov, 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 2015)*.

Todor Mihaylov and Preslav Nakov. 2016. Semanticz at SemEval-2016 task 3: Ranking relevant answers in community question answering using semantic similarity based on fine-tuned word embeddings. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval 2016)*. Association for Computational Linguistics, San Diego, California, pages 804 – 811. http://www.aclweb.org/anthology/S16.

Preslav Nakov, Doris Hoogeveen, Lluís Màrquez, Alessandro Moschitti, Hamdy Mubarak, Timothy Baldwin, and Karin Verspoor. 2017. SemEval-2017 task 3: Community question answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation*. Association for Computational Linguistics, Vancouver, Canada, SemEval '17.

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*. Association for Computational Linguistics, San Diego, California, SemEval '16.

Massimo Nicosia, Simone Filice, Alberto Barrón-Cedeno, Iman Saleh, Hamdy Mubarak, Wei Gao, Preslav Nakov, Giovanni Da San Martino, Alessandro Moschitti, Kareem Darwish, et al. 2015. Qcri: Answer selection for community question answering-experiments for arabic and english. In *Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval*. volume 15, pages 203–209.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, pages 311–318.

Vasile Rus, Mihai C Lintean, Rajendra Banjade, Nobal B Niraula, and Dan Stefanescu. 2013. Semilar: The semantic similarity toolkit. In *Association for Computational Linguistics (Conference System Demonstrations)*. pages 163–168.

Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, pages 373–382.

Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Stroudsburg, PA, USA, EMNLP '11, pages 151–161. http://dl.acm.org/citation.cfm?id=2145432.2145450.

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15(1):1929–1958.

Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. 2014. Learning sentiment-specific word embedding for twitter sentiment classification. In *Proceedings of the meeting of the Association for Computational Linguistics*. pages 1555–1565.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016. Towards universal paraphrastic sentence embeddings. In *Proceedings of the International Conference on Learning Representations*.