VectorSLU: A Continuous Word Vector Approach to Answer Selection in Community Question Answering Systems

Yonatan Belinkov, Mitra Mohtarami, Scott Cyphers, James Glass
MIT Computer Science and Artificial Intelligence Laboratory, Cambridge, MA 02139, USA
{belinkov, mitra, cyphers, glass}@csail.mit.edu

Abstract

Continuous word and phrase vectors have proven useful in a number of NLP tasks. Here we describe our experience using them as a source of features for the SemEval-2015 task 3, consisting of two community question answering subtasks: *Answer Selection* for categorizing answers as *potential*, *good*, and *bad* with regards to their corresponding questions; and *YES/NO inference* for predicting a *yes*, *no*, or *unsure* response to a YES/NO question using all of its *good* answers. Our system ranked 6th and 1st in the English answer selection and YES/NO inference subtasks respectively, and 2nd in the Arabic answer selection subtask.

1 Introduction

Continuous word and phrase vectors, in which similar words and phrases are associated with similar vectors, have been useful in many NLP tasks (Al-Rfou et al., 2013; Bansal et al., 2014; Bowman et al., 2014; Boyd-Graber et al., 2012; Chen and Rudnicky, 2014; Guo et al., 2014; Iyyer et al., 2014; Levy and Goldberg, 2014; Mikolov et al., 2013c).

To evaluate the effectiveness of continuous vector representations for *Community question answering* (CQA), we focused on using simple features derived from vector similarity as input to a multi-class linear SVM classifier. Our approach is language independent and was evaluated on both English and Arabic. Most of the vectors we use are domain-independent.

CQA services provide forums for users to ask or answer questions on any topic, resulting in high variance answer quality (Màrquez et al., 2015). Searching for good answers among the many responses can be time-consuming for participants. This is illustrated by the following example of a question and subsequent answers.

Q: *Can I obtain Driving License my QID is written Employee?*

A1: *the word employee is a general term that refers to all the staff in your company ... you are all considered employees of your company*

A2: *your qid should specify what is the actual profession you have. I think for me, your chances to have a drivers license is low.*

A3: *his asking if he can obtain. means he have the driver license.*

Answer selection aims to automatically categorize answers as: *good* if they completely answer the question, *potential* if they contain useful information about the question but do not completely answer it, and *bad* if irrelevant to the question. In the example, answers **A1**, **A2**, and **A3** are respectively classified as *potential*, *good*, and *bad*. The Arabic answer selection task uses the labels *direct*, *related*, and *irrelevant*.

YES/NO inference infers a *yes*, *no*, or *unsure* response to a question through its *good* answers, which might not explicitly contain *yes* or *no* keywords. For example, the answer for **Q** is *no* with respect to **A2** that can be interpreted as a *no* answer to the question.

The remainder of this paper describes our features and our rationale for choosing them, followed by an analysis of the results, and a conclusion.

Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015), pages 282–287,
Denver, Colorado, June 4-5, 2015.

Text-based features
Text-based similarities
yes/no/probably-like words existing
Vector-based features
Q&A vectors
OOV Q&A
yes/no/probably-based cosine similarity
Metadata-based features
Q&A identical user
Rank-based features
Normalized ranking scores

Table 1: The different types of features.

2 Method

Continuous vector representations, described by Schütze (Schütze, 1992a; Schütze, 1992b), associate similar vectors with similar words and phrases. Most approaches to computing vector representations use the observation that similar words appear in similar contexts (Firth, 1957). The theses of Sahlgren (Sahlgren, 2006), Mikolov (Mikolov, 2012), and Socher (Socher, 2014) provide extensive information on vector representations.

Our system analyzes questions and answers with a `DkPro` (Eckart de Castilho and Gurevych, 2014) `uimaFIT` (Ogren and Bethard, 2009) pipeline. The `DkPro OpenNLP` (Apache Software Foundation, 2014) segmenter and chunker tokenize and find sentences and phrases in the English questions and answers, followed by lemmatization with the Stanford lemmatizer (Manning et al., 2014). In Arabic, we only apply lemmatization, with no chunking, using `MADAMIRA` (Pasha et al., 2014). Stop words are removed in both languages.

As shown in Table 1, we compute *text-based*, *vector-based*, *metadata-based* and *rank-based* features from the pre-processed data. The features are used for a linear SVM classifier for answer selection and YES/NO answer inference tasks. YES/NO answer inference is only performed on *good* YES/NO question answers, using the YES/NO majority class, and *unsure* otherwise. SVM parameters are set by grid-search and cross-validation.

Text-based features These features are mainly computed using text similarity metrics that measure the string overlap between questions and answers: The *Longest Common Substring* measure (Gusfield, 1997) identifies uninterrupted common strings, while the *Longest Common Subsequence* measure (Allison and Dix, 1986) and the *Longest Common Subsequence Norm* identify common strings with interruptions and text replacements, while *Greedy String Tiling* measure (Wise, 1996) allows reordering of the subsequences. Other measures which treat text as sequences of characters and compute similarities include the *Monge Elkan Second String* (Monge and Elkan, 1997) and *Jaro Second String* (Jaro, 1989) measures. A *Cosine Similarity-type* measure based on term frequency within the text is also used. Sets of (1-4)-grams from the question and answer are compared with *Jaccard coefficient* (Lyon et al., 2004) and *Containment* measures (Broder, 1997).[1]

Another group of text-based features identifies answers that contain *yes*-like (e.g., "yes", "oh_yes", "yeah", "yep"), *no*-like (e.g., "no", "none", "nope", "never") and *unsure*-like (e.g., "possibly", "conceivably", "perhaps", "might") words. These word groups were determined by selecting the top 20 nearest neighbor words to the words *yes*, *no* and *probably* based on the cosine similarity of their `Word2Vec` vectors. These features are particularly useful for the YES/NO answer inference task.

Vector-based features Our *vector-based* features are computed from `Word2Vec` vectors (Mikolov et al., 2013a; Mikolov et al., 2013b; Mikolov et al., 2013d). For English word vectors we use the GoogleNews vectors dataset, available on the `Word2Vec` web site,[2] which has a 3,000,000 word vocabulary of 300-dimensional word vectors trained on about 100 billion words. For Arabic word vectors we use `Word2Vec` to train 100-dimensional vectors with default settings on a lemmatized version of the Arabic Gigaword (Linguistic Data Consortium, 2011), obtaining a vocabulary of 120,000 word lemmas.

We also use `Doc2Vec`,[3] an implementation of (Le and Mikolov, 2014) in the `gensim`

[1] These features are mostly taken from the QCRI baseline system: `http://alt.qcri.org/semeval2015/task3/index.php?id=data-and-tools`.

[2] `https://code.google.com/p/word2vec`.

[3] `http://radimrehurek.com/gensim/models/`

toolkit (Řehůřek and Sojka, 2010). `Doc2Vec` provides vectors for text of arbitrary length, so it allows us to directly model answers and questions. The `Doc2Vec` vectors were trained on the CQA English data, creating a single vector for each question or answer. These are the only vectors that were trained specifically for the CQA domain.

We implemented a UIMA annotator that associates a `Word2Vec` word vector with each in-vocabulary token (or lemma). No vectors are assigned for out of vocabulary tokens. Another annotator computes the *average* of the vectors for the entire question or answer, with no vector assigned if all tokens are out of vocabulary.

We initially used the cosine similarity of the question and answer vectors as a feature for the SVM classifier, but we found that we had better results using the normalized vectors themselves. We hypothesize that the SVM was able to tune the importance of the components of the vectors, whereas cosine similarity weights each component equally. If the question or answer has no vector, we use a 0 vector. To make it easier for the classifier to ignore the vectors in these cases, we add boolean features indicating out of vocabulary, *OOV Question* and *OOV Answer*.

Even though the bag of words approach showed encouraging results, we found it to be too coarse, so we also compute average vectors for each sentence. For English, we also compute average vectors for each chunk. Then we look for the best matches between sentences (and chunks) in the question and answer in terms of cosine similarity, and use the pairs of (unnormalized) vectors as features.[4] More formally, given a question with sentence vectors $\{q_i\}$ and an answer with sentence vectors $\{a_j\}$, we take as features the values of the vector pair $(\hat{q}, \hat{a})$ defined as:

$$(\hat{q}, \hat{a}) = \arg\max_{(q_i, a_j)} \frac{q_i \cdot a_j}{\|q_i\| \, \|a_j\|}$$

We also have six features corresponding to the greatest cosine similarity between the comment word vectors and the vectors for the words *yes, Yes, no, No, probably* and *Probably*. These features are more effective for the YES/NO classification task.

`doc2vec.html`.

[4]Post-evaluation testing showed no significant difference between using normalized or unnormalized vectors.

Metadata-based features As a *metadata-based* indicator, the *Q&A identical user* identifies if the user who posted the question is the same user who wrote the answer. This indicator is useful for detecting irrelevant *dialogue* answers.

Rank-based features We employ SVM Rank[5] to compute ranking scores of answers with respect to their corresponding questions. After generating all other features, SVM Rank is run to produce ranking scores for each possible answer. For training SVM Rank, we convert answer labels to ranks according to the following heuristic: *good* answers are ranked first, *potential* ones second, and *bad* ones third. Ranking scores are then used as features for the classifier. The normalization of these scores can be used as *rank-based* features to provide more information to the classifier, although these scores are also used without any other features as explained in Section 3.

3 Evaluation and Results

We evaluate our approach on the answer selection and YES/NO answer inference tasks. We use the CQA datasets provided by the Semeval 2015 task that contain 2600 training and 300 development questions and their corresponding answers (a total number of 16,541 training and 1,645 development answers). About 10% of these questions are of the YES/NO type. We combined the training and development datasets for training purposes. The test dataset includes 329 questions and 1976 answers. About 9% of the test questions are bipolar.

We also evaluate our performance on the Arabic answer selection task. The dataset contains 1300 training questions, 200 development questions, and 200 test questions. This dataset does not include YES/NO questions.

English answer selection Our approach for the answer selection task in *English* ranked 6th out of 12 submissions and its results are shown in Table 2. *VectorSLU-Primary* shows the results when we include all the features listed in Table 1 except the rank-based features. *VectorSLU-Contrastive* shows the results when we include all the features except

[5]`http://www.cs.cornell.edu/people/tj/svm_light/svm_rank.html`.

Method	Macro-F1	Accuracy
VectorSLU-Primary	49.10	66.45
VectorSLU-Contrastive	49.54	70.45
JAIST (best)	**57.19**	**72.52**
Baseline	22.36	50.46

Table 2: Results for the English answer selection task.

Method	Macro-F1	Accuracy
VectorSLU-Primary	70.99	76.32
VectorSLU-Contrastive	73.18	78.12
QCRI (best)	**78.55**	**83.02**
Baseline	24.03	56.34

Table 3: Results for the Arabic answer selection task.

the rank-based and text-based features. Interestingly, *VectorSLU-Contrastive* leads to a better performance than *VectorSLU-Primary*. The lower performance of *VectorSLU-Primary* could be due to the high overlap between text-based features in different classes that can clearly mislead classifiers. For example, **A1**, **A2** and **A3** (see Section 1) all have a considerable word overlap with their question, while only **A2** is a *good* answer. The last two rows of the table are respectively related to the best performance among all submissions and the majority class baseline that always predicts *good*.

Arabic answer selection Our approach for answer selection in *Arabic* ranked 2nd out of 4 submissions. Table 3 shows the results. In these experiments, we employ all features listed in Table 1 except for yes/no/probably-based features, since the Arabic task does not include YES/NO answer inference. Vectors were trained from the Arabic Gigaword (Linguistic Data Consortium, 2011). We found lemma vectors to work better than token vectors.

We computed ranking scores with SVM Rank for both *VectorSLU-contrastive* and *VectorSLU-Primary*. In the case of *VectorSLU-contrastive*, we used these scores to predict labels according to the following heuristic: the top scoring answer is labeled as *direct*, the second scoring answer as *related*, and all other answers as *irrelevant*. This decision mechanism is based on the distribution in the training and development data, and proved to work well on the test data. However, for our primary submission we were interested in a more principled mechanism. Thus, in the *VectorSLU-primary* system we computed 10 extra classification features from the ranking scores. These features are used to provide prior knowledge about relative ranking of answers with respect to their corresponding questions. To compute these features, we first rank answers with respect to questions and then scale the resultant scores into the [0,1] range. We then consider 10 binary features that indicate whether the score of each input answer is the range of [0,0.1), [0.1,0.2), ..., [0.9,1), respectively. Note that each feature vector contains exactly one 1 and nine 0s.

Method	Macro-F1	Accuracy
VectorSLU-Primary (best)	**63.70**	**72.00**
VectorSLU-Contrastive	61.90	68.00
Baseline	25.00	60.00

Table 4: Results for the English YES/NO inference task.

The last two rows of the table are related to the best performance and the majority class baseline that always predicts *irrelevant*.

English YES/NO inference For the indirect YES/NO answer inference task, we achieve the best performance and ranked 1st out of 8 submissions. Table 4 shows the results. *VectorSLU-Primary* and *VectorSLU-Contrastive* have the same definition as in Table 2. Both approaches with or without the text-based features outperform the baseline that always predicts *yes* as the majority class and other submissions. This indicates the effectiveness of the vector-based features.

4 Related Work

We are not aware of any previous CQA work using continuous word vectors. Our vector features were somewhat motivated by existing text-based features, taken from the QCRI baseline system, replacing text-similarity heuristics with cosine similarity. Some of the approaches to classifying answers can be found in the general CQA literature, such as (Toba et al., 2014; Bian et al., 2008; Liu et al., 2008).

5 Conclusion

In summary, we represented words, phrases, sentences and whole questions and answers in vector space, and computed various features from them for a classifier, for both English and Arabic. We showed the utility of these vector-based features for addressing the answer selection and the YES/NO answer inference tasks in community question answering.

Acknowledgments

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

References

Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. 2013. Polyglot: Distributed Word Representations for Multilingual NLP. *CoRR*, abs/1307.1662.

Lloyd Allison and Trevor I. Dix. 1986. A bit-string longest-common-subsequence algorithm. *Information Processing Letters*, 23(5):305 – 310.

Apache Software Foundation. 2014. OpenNLP.

Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2014. Tailoring Continuous Word Representations for Dependency Parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 2: Short Papers*, pages 809–815.

Jiang Bian, Yandong Liu, Eugene Agichtein, and Hongyuan Zha. 2008. Finding the Right Facts in the Crowd: Factoid Question Answering over Social Media. In *Proceedings of the 17th International Conference on World Wide Web*, WWW '08, pages 467–476, New York, NY, USA.

Samuel R. Bowman, Christopher Potts, and Christopher D. Manning. 2014. Recursive Neural Networks for Learning Logical Semantics. *CoRR*, abs/1406.1827.

Jordan L. Boyd-Graber, Brianna Satinoff, He He, and Hal Daumé III. 2012. Besting the Quiz Master: Crowdsourcing Incremental Classification Games. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, July 12-14, 2012, Jeju Island, Korea*, pages 1290–1301.

Andrei Z. Broder. 1997. On the Resemblance and Containment of Documents. In *Proceedings of the Compression and Complexity of Sequences 1997*, SEQUENCES '97, pages 21–, Washington, DC, USA.

Yun-Nung Chen and Alexander I. Rudnicky. 2014. Dynamically Supporting Unexplored Domains in Conversational Interactions by Enriching Semantics with Neural Word Embeddings. In *Proceedings of the 2014 Spoken Language Technology Workshop, December 7-10, 2014, South Lake Tahoe, Nevada, USA*, pages 590–595.

Richard Eckart de Castilho and Iryna Gurevych. 2014. A broad-coverage collection of portable NLP components for building shareable analysis pipelines. In Nancy Ide and Jens Grivolla, editors, *Proceedings of the Workshop on Open Infrastructures and Analysis Frameworks for HLT (OIAF4HLT) at COLING 2014*, pages 1–11, Dublin, Ireland, August.

John Firth. 1957. A Synopsis of Linguistic Theory, 1930-1955. *Studies in Linguistic Analysis*, pages 1–32.

Daniel (Zhaohan) Guo, Gokhan Tur, Wen-tau Yih, and Geoffrey Zweig. 2014. Joint Semantic Utterance Classification and Slot Filling with Recursive Neural Networks. pages 554–559.

Dan Gusfield. 1997. *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*. Cambridge University Press, New York, NY, USA.

Mohit Iyyer, Jordan L. Boyd-Graber, Leonardo Max Batista Claudino, Richard Socher, and Hal Daumé III. 2014. A Neural Network for Factoid Question Answering over Paragraphs. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 633–644.

Matthew A. Jaro. 1989. Advances in Record-Linkage Methodology as Applied to Matching the 1985 Census of Tampa, Florida. *Journal of the American Statistical Association*, 84(406):414–420.

Quoc V. Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. *CoRR*, abs/1405.4053.

Omer Levy and Yoav Goldberg. 2014. Dependency-Based Word Embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 2: Short Papers*, pages 302–308.

Linguistic Data Consortium. 2011. Arabic Gigaword Fifth Edition. `https://catalog.ldc.upenn.edu/LDC2011T11`.

Yandong Liu, Jiang Bian, and Eugene Agichtein. 2008. Predicting Information Seeker Satisfaction in Community Question Answering. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, pages 483–490, New York, NY, USA.

Caroline Lyon, Ruth Barrett, and James Malcolm. 2004. A theoretical basis to the automated detection of copying between texts, and its practical implementation in the ferret plagiarism and collusion detector. In *Plagiarism: Prevention, Practice and Policies 2004 Conference*.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60.

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)*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient Estimation of Word Representations in Vector Space. *CoRR*, abs/1301.3781.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed Representations of Words and Phrases and their Compositionality. *CoRR*, abs/1310.4546.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013c. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 3111–3119.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013d. Linguistic Regularities in Continuous Space Word Representations. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 9-14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA*, pages 746–751.

Tomàš Mikolov. 2012. *Statistical Language Models Based on Neural Networks*. Ph.D. thesis, Brno University of Technology.

Alvaro Monge and Charles Elkan. 1997. An Efficient Domain-Independent Algorithm for Detecting Approximately Duplicate Database Records.

Philip Ogren and Steven Bethard. 2009. Building Test Suites for UIMA Components. In *Proceedings of the Workshop on Software Engineering, Testing, and Quality Assurance for Natural Language Processing (SETQA-NLP 2009)*, pages 1–4, Boulder, Colorado, June.

Arfath Pasha, Mohamed Al-Badrashiny, Mona Diab, Ahmed El Kholy, Ramy Eskander, Nizar Habash, Manoj Pooleery, Owen Rambow, and Ryan Roth. 2014. MADAMIRA: A Fast, Comprehensive Tool for Morphological Analysis and Disambiguation of Arabic. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 1094–1101, Reykjavik, Iceland.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May.

Magnus Sahlgren. 2006. *The Word-Space Model*. Ph.D. thesis, Stockholm University.

Hinrich Schütze. 1992a. Dimensions of Meaning. In *Proceedings of Supercomputing '92*, pages 787–796.

Hinrich Schütze. 1992b. Word Space. In *NIPS*, pages 895–902.

Richard Socher. 2014. *Recursive Deep Learning for Natural Language Processing and Computer Vision*. Ph.D. thesis, Stanford University.

Hapnes Toba, Zhao-Yan Ming, Mirna Adriani, and Tat-Seng Chua. 2014. Discovering high quality answers in community question answering archives using a hierarchy of classifiers. *Information Sciences*, 261(0):101 – 115.

Michael J. Wise. 1996. YAP3: Improved Detection Of Similarities In Computer Program And Other Texts. In *SIGCSEB: SIGCSE Bulletin (ACM Special Interest Group on Computer Science Education)*, pages 130–134.