Overfitting at SemEval-2016 Task 3: Detecting Semantically Similar Questions in Community Question Answering Forums with Word Embeddings

Hujie Wang
University of Waterloo
200 University Ave W
Waterloo, ON N2L 3G1, Canada
h3wang@uwaterloo.ca

Pascal Poupart
University of Waterloo
200 University Ave W
Waterloo, ON N2L 3G1, Canada
ppoupart@uwaterloo.ca

Abstract

This paper presents an approach for estimating the question-question similarity of an English dataset specified in Shared Task 3, subtask B of SemEval-2016. Given a new question and a set of the first 10 related questions retrieved by a search engine, participants are asked to produce a binary relevant/irrelevant judgement and rerank the related questions according to their similarity with respect to the original question. Our submitted system uses a 2-layer feed-forward neural network with the averages of word embedding vectors to predict the semantic similarity score of two questions. We also evaluate the results of Random Forests and Support Vector Machine in comparison to the Neural Network. Results on the test dataset show that the model achieves a Mean Average Precision 69.681.

1 Introduction

Finding semantically related questions is a difficult task due to two main factors: (1) paraphrasing, which can appear at different levels, e.g., lexical, phrasal, sentential (Madnani and Dorr, 2010); and (2) two questions could be asking different things but look for the same solution. Thus, traditional surface similarity measures such as Edit Distance, Jaccard, Dice and Overlap coefficient and their variations are not able to capture many cases of semantic relatedness. To preserve the semantic meaning of a question, we use weighted average word embedding vectors to model questions. We evaluate several algorithms that take the weighted average word embedding vectors of two questions as inputs and predict whether two questions are related or not.

To provide a benchmark so as to compare and develop question-question similarity measuring models in Community Question Answering forums, the Question-Question Similarity task in SemEval-2016 Task 3, Subtask B (Nakov et al., 2016) requires the participants to determine whether two questions are sementically related and given a new question, rerank all similar questions retrieved by a search engine. **Table 1** presents an example of a pair of semantically related questions.

We perform an extensive number of experiments using data from Shared Task 3 and compare three types of classifiers: Neural Network (NN), Support Vector Machine (SVM) and Random Forests (RF), as well as the impact of different weighting schemes for averaging word embedding vectors and different word embedding dimensions. The results show that 2-layer Feedforward Neural Network with Idf weighting scheme and in-domain word embeddings outperforms the rest of models.

Title: Best Bank
Body: Which is a good bank as per your experience in Doha
Title: Good Bank
Body: Hi Guys; I need to open a new bank accoount. Which is the best bank in Qatar ? I assume all of them will roughly be the same; but stll which has a slight edge (Money transfer; benifits etc) Thanks !!!

Table 1: An example of semantically related questions

Proceedings of SemEval-2016, pages 861–865,
San Diego, California, June 16-17, 2016.

2 Word Embedding Features

Recently (Mikolov et al., 2013) introduced word2vec, a novel word-embedding procedure. Their model is able to learn continuous vector representations of words from a very large dataset by using a feedforward Neural Net Language Model (NNLM). Specifically, The CBOW architecture predicts the current word based on the context, and the Skip-gram architecture predicts surrounding words given the current word. Learning the word embedding is entirely unsupervised and it can be computed on the text corpus of interest.

Each word is encoded by a column vector in an embedding matrix $W_e \in \mathbb{R}^{d \times |V|}$ learnt by using word2vec, where V is a fixed-sized vocabulary. Each column of the matrix represents the word embedding vector of the i-th word in V. Each word w is transformed into its vector representations v_w by using a matrix-vector product:

$$v_w = W_e b_w \tag{1}$$

where b_w is a one-hot encoding vector of size $|V|$ which has value 1 at the index of the word w and zero in all other positions. Let X_t and X_b be the set of word tokens of a question's title and body, then the vector representations of the question's title and body are the weighted vector average of the word embeddings associated with tokens in X_t and X_b:

$$Q_t = \frac{1}{|X_t|} \sum_{w \in X_t} v_w c_w \tag{2}$$

$$Q_b = \frac{1}{|X_b|} \sum_{w \in X_b} v_w c_w \tag{3}$$

where c_w denotes the weight of the word w. We also investigate the impact of different weighting schemes:

1. $c_w = 1$

2. $c_w = IDF(w)$ where IDF denotes inverse document frequency (Sparck Jones, 1972). Specifically,

$$IDF(w) = \log(1 + \frac{N}{n_w + 1}) \tag{4}$$

N is the total number of documents in the corpus and n_w is the number of documents where the term w appears. In our case, we define each question's body in the training dataset as a document.

Given a pair of questions ($Q^{(1)}$, $Q^{(2)}$), we compute $Q_t^{(1)}$, $Q_b^{(1)}$, $Q_t^{(2)}$ and $Q_b^{(2)}$ by using the above transformations. Given vectors u and v, we denote the similarity features $sim_{(u,v)}$ as the concatenation of $u \cdot v$ (component-wise product of u and v) and $|u - v|$. These two features were also used by (Tai et al., 2015). The final vector representations or features for the input pair of questions ($Q^{(1)}$, $Q^{(2)}$) is defined as the concatenation of $sim_{(Q_t^{(1)}, Q_t^{(2)})}$ and $sim_{(Q_b^{(1)}, Q_b^{(2)})}$. Those features are then fed into a classifier to predict a binary True/False label.

3 Experiments and Results

3.1 Datasets

In our experiments we use data from the community-created Qatar Living Forums[1] collected for SemEval-2016 task 3, subtask B. There are 317 original questions and 3,169 related questions. The dataset is pre-splitted into 264 original questions and 2669 related question for training, as well as 50 original questions and 500 related question for validation. Each data point is a pair of questions (an original question and a related question) and a similarity label, which is either "PerfectMatch", "Relevant" or "Irrelevant". According to the task description, we need to predict a binary label where "True" covers "PerfectMatch" and "Relevant", and "False" covers "Irrelevant", and rerank a set of related questions according to their similarity with respect to the original question.

3.2 Preprocessing

Several text preprocessing operations are performed before we extract features. We first transform all words into lowercase, filter out stop words, remove all punctuations and replace numbers with the token "NUMBER". Then the Phrase Detection Toolkit implemented in Gensim is used to group tokens together if they form a phrase. After that, all unique words and phrases are selected from the training set as our vocabulary which has a size of 11990. We deal with unseen words in the test set by marking them as "UNK".

[1] www.qatarliving.com/forum

Model	Classifier	Weighting Scheme	Word Embedding Dimension
NN-Idf-100	NN	$c_w = Idf(w)$	100
RF-Idf-30	RF	$c_w = Idf(w)$	30
NN-Avg-30	NN	$c_w = 1$	30
RF-Avg-30	RF	$c_w = 1$	30
SVM-Avg-80	SVM	$c_w = 1$	80
SVM-Avg-30	SVM	$c_w = 1$	30

Table 2: Description of models

3.3 Word Embedding Features

We perform pre-training using the skip-gram NN architecture (Mikolov et al., 2013) available in the Gensim Word2vec tool[2]. We use in-domain word embeddings vectors trained on the Community Question Answering dataset provided by SemEval-2016 Task 3 with the following parameter settings: (1) 30, 80 and 100 dimensional embedding vectors; (2) the maximum distance between the current and predicted word within a sentence is set to 5; (3) ignoring all words with total frequency lower than 5; (4) for each positive sample, 5 negative samples are drawn for negative sampling; (5) the number of iterations is set to 30.

3.4 Models

Several classification algorithms are explored on development dataset including Feedforward Neural Network (NN), Support Vector Machine (SVM) and Random Forests (RF). Due to the lack of the time, we only submitted the predictions from one of our NN classifiers. Configuration details of each model are presented in **Table 2**.

3.4.1 Feedforward Neural Network

We consider a two-layer Feedforward Neural Network with 64 sigmoid activation hidden units as our classifier. Weights are initialized with random orthogonal conditions (Saxe et al., 2013). We apply Dropout (Srivastava et al., 2014) with a fixed dropping probability of 0.9 for the input layer and 0.5 for the hidden layer. Our network is trained by minimizing the categorical cross entropy error over the training set using Adam (Kingma and Ba, 2014), a first-order gradient-based optimization method. We use the backpropagation algorithm (Rumelhart et al., 1988) to compute gradients of the network. In our experiments, we implement the NN and the backpropagation algorithm using Keras[3].

3.4.2 Support Vector Machine

We use the (Gaussian) radial basis function kernel

$$K(x, x') = \exp(-\gamma|x - x'|^2) \quad (5)$$

with $\gamma = \frac{1}{4d}$, where d is the dimension of word embedding vectors. We set the regularization parameter C of soft margin cost function to 1.

3.4.3 Random Forests

We explored a large set of parameter values by doing randomized search and selecting only the best set of parameters for our RF models. Specifically, we use the following fine-tuned parameters:

- The number of trees: 382
- Measurement of the quality of a split: entropy
- The maximum depth of the tree: 33
- The minimum number of samples required to split an internal node: 5
- The minimum number of samples in newly created leaves: 2
- The number of features to consider when looking for the best split: 116

3.5 Results and Discussion

Table 3 summarizes the performance of our systems, along with the baseline systems provided by the organizers and the top three systems (UH-PRHLT-primary, ConvKN-primary and Kelp-primary). The results show that models with Idf weighting scheme outperform models using naive word embedding averages. The performance differences between different models using the same weighting scheme and the same word embedding

[2]www.radimrehurek.com/gensim

[3]https://github.com/fchollet/keras

Model	Rank	MAP	AvgRec	MRR	Precision	Recall	F1	Accuracy
NN-Idf-100	—	**71.55**	**86.21**	**81.76**	56.84	57.08	56.96	71.29
RF-Idf-30	—	70.67	85.43	79.25	62.69	54.08	58.06	74.00
NN-Avg-30*	10	69.68	85.10	80.18	**63.20**	**67.81**	**65.42**	**76.14**
RF-Avg-30	—	68.65	84.43	78.37	60.71	51.07	55.48	72.71
SVM-Avg-80	—	67.06	82.55	78.80	55.91	60.94	58.32	71.00
SVM-Avg-30	—	61.39	78.20	71.91	61.59	43.35	50.88	72.14
UH-PRHLT-primary	1	76.70	90.31	83.02	63.53	69.53	66.39	76.57
ConvKN-primary	2	76.02	90.70	84.64	68.58	66.52	67.54	78.71
Kelp-primary	3	75.83	91.02	82.71	66.79	75.97	71.08	79.43
Baseline 1 (IR)	—	74.75	88.30	83.79	—	—	—	—
Baseline 2 (random)	—	46.98	67.92	50.96	32.58	73.82	45.20	40.43
Baseline 3 (all true)	—	—	—	—	33.29	100.00	49.95	33.29
Baseline 4 (all false)	—	—	—	—	—	—	—	66.71

Table 3: Performance of out top 6 systems and baseline systems on test dataset, as well as top ranking systems. The second column shows the rank of the primary runs with respect to the official MAP score. NN-Avg-30 is the model we used for the official submission. MAP and MRR stand for Mean Average Precision and Mean Reciprocal Rank, respectively.

dimension are not significant, which shows that the type of the classifier is not the bottleneck of overall performance. Also, we find that RF and NN perform slightly better than SVM. We also observe that using a large word embedding dimension (≥ 80) results in severe overfitting on the development dataset since we have a small dataset. Applying strong regularization such as Dropout with high dropping probability reduces overfitting and helps our NN models achieve high performance. Our best classifier (NN-Idf-100) achieves the highest Mean Average Precision among our classifiers.

4 Conclusion

In this paper, we address the Question-Question Similarity task by building three types of supervised classifiers. The results show that classifiers without heavy feature engineering are able to outperform the random baseline by a large margin. It also demonstrates that using the proposed features based on word embeddings is effective and able to capture semantic meanings of a question.

References

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Nitin Madnani and Bonnie J Dorr. 2010. Generating phrasal and sentential paraphrases: A survey of data-driven methods. *Computational Linguistics*, 36(3):341–387.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

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 '16, San Diego, California, June. Association for Computational Linguistics.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1988. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1.

Andrew M Saxe, James L McClelland, and Surya Ganguli. 2013. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*.

Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, January.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representa-

tions from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075*.