LearningToQuestion at SemEval 2017 Task 3: Ranking Similar Questions by Learning to Rank Using Rich Features

Naman Goyal
Georgia Institute of Technology
naman.goyal21@gmail.com

Abstract

This paper describes our official entry LearningToQuestion for SemEval 2017 task 3 community question answer, subtask B. The objective is to rerank questions obtained in web forum as per their similarity to original question. Our system uses pairwise learning to rank methods on rich set of hand designed and representation learning features. We use various semantic features that help our system to achieve promising results on the task. The system achieved second highest results on official metrics MAP and good results on other search metrics.

1 Introduction

In online forums question answering is one of the most popular way for users to share information between each other. Due to the unstructured nature of these forums, it's a problem to find relevant information from the already existing information for users. One way to solve this problem is to design systems to automatically find similar content (question, answer, comment) to the user's posted question. SemEval-2017 task 3 (Nakov et al., 2017) focuses on solving this problem in community question answer by various subtasks of ranking relevant information in Qatar living forums data. The system presented in this paper focuses on **subtask B**, to re-rank given set of questions retrieved by search engine, in their similarity to original question.

The system is mainly designed by employing learning to rank methods on the rich feature set obtained by text processing of the question text.

2 Data

We primarily use the annotated training, development and testing dataset provided by the SemEval-2017 task 3 organizers. The dataset is collected by organizers from Qatar living forum. It's in the form of an original question and set of related questions. Each related question in training and development dataset is annotated with one of the 3 possible tags, *PerfectMatch*, *Relevant* or *Irrelevant*. A ranking task is required to rank both *PerfectMatch* and *Relevant* above *Irrelevant* questions without any distinction between the first two. The train dataset for subtask B consists of 317 original questions and 3169 retrieved questions by search engine roughly 10 related questions per original question. The organizers have also provided annotated test dataset from SemEval-2016 challenge.

Along with these we also used Glove embeddings (Pennington et al., 2014) which were pretrained using 6 billion tokens from Wikipedia-2014 and Gigaword dataset.

3 System

Since the task is a ranking task, our system uses learning to rank (Trotman, 2005) to model the ranking of questions. Learning to rank refers to various machine learning techniques used in ranking tasks. These have been studied in information retrieval literature and they power many of the industrial search engines. These systems mainly fall into 3 categories: pointwise, pairwise and listwise as described in (Liu et al., 2009). We use pairwise methods for our system with rich feature set. Our feature set is combination of various hand generated features and semantic features learned by neural network. In the following section we first describe these features and then the learning to rank method used.

Proceedings of the 11th International Workshop on Semantic Evaluations (SemEval-2017), pages 310–314,
Vancouver, Canada, August 3 - 4, 2017.

3.1 Features

3.1.1 Neural Embedding Similarity Feature

We use various neural network learned embeddings as similarity feature in our system. The system uses Siamese network to learn these similarity measures. Siamese nets were first introduced in the early 1990s by (Bromley et al., 1993) to solve signature verification as an image matching problem. A siamese neural network consists of twin networks which accept distinct inputs but are joined by an energy function at the top. This function computes some metric between the highest level feature representation on each side. The weights between both the networks are shared generally, so that they project the similar texts not far in the embedding dimension. We use contrastive loss described in (Chopra et al., 2005) as the loss function to the Siamese network. Glove pretrained vectors (300 dimension) are fed as input to the neural network. The final neural embeddings are generated by various architectures.

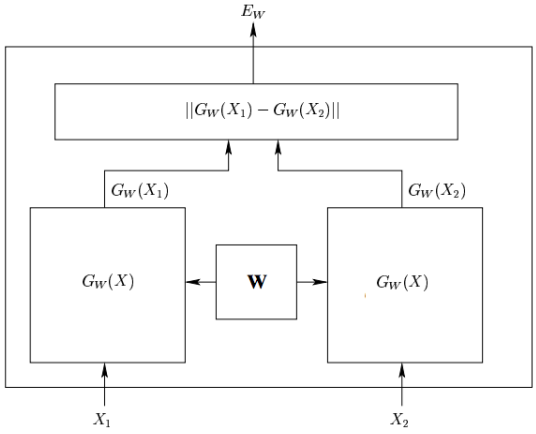

Figure 1: Simaese network

Figure 1 shows a siamese network, where X_1 represents the original question text and X_2 represents the candidate question text. G_W represents a complex nonlinear function which is represented by neural network having weights W. The euclidean distance of the vectors is used to compute the contrastive loss. The goal is to minimize the distance in the embedding space of the similar question text and maximize for non similar pairs. The contrastive loss can be given by following equation:

$$L = Y||G_W(X_1), G_W(X_2)||^2 + (1 - Y) \\ max(0, m - ||G_W(X_1), G_W(X_2)||^2)$$

where Y is annotated tag, 1 if X_1 and X_2 are similar, 0 otherwise. m is margin parameter for hinge loss, which is kept 1 for all our networks. We use following networks to generate text embedding:

Long Short Term Memory LSTM (Hochreiter and Schmidhuber, 1997) are popular variant of the the recurrent neural network architecture that captures the long term dependency in text and deals with vanishing gradient problem. Recently LSTMs have been very successful in various NLP tasks.

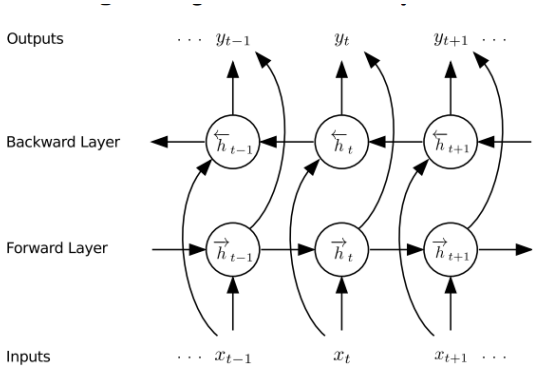

Figure 2: Bidirectional recurrent neural network

Figure 2 shows a bidirectional recurrent neural network architecture. Bi-directional RNN processes the text in both directions with separate hidden units and these hidden representations are concatenated together to create final hidden embedding. For bi-directional LSTM, the hidden unit is a LSTM cell combining of various gates. We use a bidirectional LSTM to generate a 256 dimensional vector for pair of text and train the model by back propagation using contrastive loss.

Gated Recurrent Unit Gated recurrent unit (GRU) (Chung et al., 2014) is another variant of RNN which were introduced recently as compared to LSTM. They also have seen similar success as LSTM in various NLP tasks. We use Bi-GRU as another network to generate the neural embeddings trained by siamese network similar to Bi-LSTM. The final hidden embedding size is 256 dimension for our Bi-GRU network also.

Convolution Net We also use convolution networks as another neural network architecture to generate embeddings inside the siamese network. We use 1D-convolution with 128 kernels, stride of 5 followed by 1D-max pool with pool-size of 5 and finally a dense layer to create a 128 dimension vector.

Implementation Details We use Keras [1] library with Theano (Theano Development Team, 2016) backend to train above 3 models. The batch size is set to 64 and dropout rate is 0.25. We run 25 epochs for each of these 3 networks training. It takes couple of hours to train on CPU. Instead of using the entire vectors into our final classifier, we compute cosine similarity of learned vectors of both the question text (for each of the 3 networks) and use that as a feature in our system.

3.1.2 Rank Features

We use rank given by the search engine as a feature in our system. This gives the system the baseline accuracy of the search engine.

3.1.3 Lexical Features

These set of features represent the lexical similarity between question texts. The lexical used are the common n-gram ($n = 1, 2, 3$) counts between the original question and a candidate question. Apart from these features, we compute a count vector and a tfidf vector for n-grams ($n = 1, 2, 3$) for both the question and candidate question and compute the cosine similarity between them.

3.1.4 Syntactical Features

These features represent the syntactical similarity between the texts of questions. This is represented by cosine similarity of POS count vector for n-gram ($n = 1, 2, 3$).

3.1.5 Semantic Features

Apart from the neural network learned semantic features, we also employ semantic similarity between question text generated by semantic net. We use the sentence similarity described in (Li et al., 2006) using WordNet as semantic net. The paper describes various heuristics used to generate this sentence similarity. First, word pair similarity is generated as a function of the shortest path between the words and height of their lowest common subsumer (LCS). This combines the word similarity with their specificity (abstract vs specific concept).

Then the sentence similarity is obtained as a linear combination of semantic similarity and the word order similarity. To generate semantic similarity, cosine between semantic vectors is obtained. The semantic vectors are generated by creating sentence vector of word presence and their similarity. Word order similarity is computed in the similar way as semantic similarity but the position of word in the sentence is used to generate the word order vector. Finally a linear combination of these two similarity features is used as the similarity measure between question texts. We use the same hyper-parameters as original paper that give the best results i.e. $\alpha = 0.2, \beta = 0.45, \eta = 0.4, \phi = 0.2, \delta = 0.85$.

The feature encodes semantic similarity and gives boost to system, shown in the results table.

3.1.6 Summarization Metrics Feature

There has been a lot of research in machine translation and summarization community to find metrics that correlate with human judgement on these tasks. We compute BLEU (Papineni et al., 2002) metrics for $1, 2, 3$ and 4 grams and compute a weighted addition (weights = $0.1, 0.1, 0.3, 0.5$). We also compute ROUGE-L (Lin, 2004), which is recall oriented similarity measure based on longest common subsequence (LCS), as a feature in our system.

3.1.7 Length feature

We compute the heuristic based on length in tokens of both the texts as $f(l_1, l_2) = \frac{abs(l_1 - l_2)}{l_1 + l_2}$.

3.1.8 Topic Similarity Feature

Topic modeling is used to generate the salient topics in the text. We use Latent Dirichlet allocation (LDA) (Blei et al., 2003) to compute topic similarity between texts. We train LDA topic model using the whole text (body and subject) as corpus. Then a topic distribution over the 50 topics is computed for both the text and cosine similarity is used as a feature in the system.

3.1.9 Other Features

We use count of 10 selected common question words also as a feature in the system.

All of the above features are calculated for both the question subject and body separately.

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

Features	Dev 2017 as test set MAP	MRR	Test 2017 as test set MAP	MRR
Baseline	71.35	76.67	41.85	46.42
Above + length features	70.92	76.73	41.60	46.35
Above + common ngram counts	71.36	77.79	43.06	46.08
Above + cosine count ngram	71.23	78.33	43.45	45.41
Above + cosine tfidf ngram	73.10	80.73	43.34	49.29
Above + cosine pos ngram	72.98	80.73	44.96	49.10
Above + semantic similarity	73.29	81.17	45.52	50.05
Above + GRU embeddings	74.21	81.17	45.61	50.05
Above + LSTM embeddings	74.43	81.17	45.91	51.15
Above + Convnet embeddings	74.60	81.17	45.91	51.15
Above + BLEU, ROUGE L	74.88	81.17	46.17	52.02
Above + common question word	74.88	81.17	46.45	52.88
Above + topic similarity	75.10	81.33	46.93	53.01
primary	75.10	81.33	46.93	**53.01**
contrastive-1	74.88	81.17	47.03	52.47
contrastive-2	74.60	81.17	47.23	53.22
All features (Pointwise)	74.43	81.17	45.89	51.07
Best Primary	-	-	**47.22**	50.07
Best Overall	-	-	49.00	52.41

Table 1: Results on dev and test set with various features

3.2 Learning to rank

We use pairwise learning to rank for ranking task which poses the ranking problem as classification problem to minimize the average number of inversions in ranking. This formulation is more closer to ranking task than predicting relevance as regression and also has theoretical guarantees of maximizing the MAP in ranking (Chen et al., 2009).

First, we create these pairs by taking original question Q_o and two candidate questions of which one was relevant and other one not, Q_{c1} and Q_{c2}. Then we generate above mentioned feature vectors $f(Q_o, Q_{c1})$, $f(Q_o, Q_{c2})$ and use feature difference $f(Q_o, Q_{c1}) - f(Q_o, Q_{c2})$ to the classifier. In total, 5949 such pairs are used for training. Logistic regression is used for our *primary* submission and linear kernel *SVM* with regularization parameter as 1 for our both contrastive submissions. The submitted systems *primary* and *contrastive-1* use train, development as training and *contrastive-2* uses test-2016 in concatenation for training.

4 Results

The results generated by the system on test data were submitted as an entry to SemEval-2017 task 3 subtask B. Our primary entry achieved second place on the MAP which was official metric for ranking. Also it achieved highest MRR amongst all the primary submissions.

Table 1 shows the dev and test set accuracy for our system with each feature applied incrementally. Our both *contrastive* submissions trained on SVM achieved better test accuracy than training on Logistic Regression. Thus the Ranking-SVM is able to generalize better.

We also experimented with pointwise learning to rank method and got inferior results thus corroborating the fact that pairwise methods are helping our system in achieving better accuracy.

5 Conclusion and Future Work

This paper presented a system which uses sophisticated learning to rank method with semantic features to obtain promising results on ranking similar questions. The paper shows that semantic features and pairwise learning are essential components to the system by ablation tests.

In future, we would like to extend our neural architecture to attention based models which have shown success in recent times. We also plan to use Triplet loss (Hoffer and Ailon, 2015) which captures ranking task in better way. Another direction is to use state-of-art listwise learning to rank methods that can directly optimize MAP.

References

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

Jane Bromley, James W. Bentz, Léon Bottou, Isabelle Guyon, Yann LeCun, Cliff Moore, Eduard Säckinger, and Roopak Shah. 1993. Signature verification using a "siamese" time delay neural network. *IJPRAI* 7(4):669–688.

Wei Chen, Tie-Yan Liu, Yanyan Lan, Zhi-Ming Ma, and Hang Li. 2009. Ranking measures and loss functions in learning to rank. In *Advances in Neural Information Processing Systems*. pages 315–323.

Sumit Chopra, Raia Hadsell, and Yann LeCun. 2005. Learning a similarity metric discriminatively, with application to face verification. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. IEEE, volume 1, pages 539–546.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* .

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Elad Hoffer and Nir Ailon. 2015. Deep metric learning using triplet network. In *International Workshop on Similarity-Based Pattern Recognition*. Springer, pages 84–92.

Yuhua Li, David McLean, Zuhair A Bandar, James D O'shea, and Keeley Crockett. 2006. Sentence similarity based on semantic nets and corpus statistics. *IEEE transactions on knowledge and data engineering* 18(8):1138–1150.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out: Proceedings of the ACL-04 workshop*. Barcelona, Spain, volume 8.

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

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.

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.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*. volume 14, pages 1532–1543.

Theano Development Team. 2016. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints* abs/1605.02688. http://arxiv.org/abs/1605.02688.

Andrew Trotman. 2005. Learning to rank. *Information Retrieval* 8(3):359–381.