Noise-Contrastive Estimation for Answer Selection with Deep Neural Networks

Jinfeng Rao[1], Hua He[1], and Jimmy Lin[2]

[1] Department of Computer Science, University of Maryland, College Park
[2] David R. Cheriton School of Computer Science, University of Waterloo

jinfeng@cs.umd.edu, huah@umd.edu, jimmylin@uwaterloo.ca

ABSTRACT

We study answer selection for question answering, in which given a question and a set of candidate answer sentences, the goal is to identify the subset that contains the answer. Unlike previous work which treats this task as a straightforward *pointwise* classification problem, we model this problem as a ranking task and propose a *pairwise* ranking approach that can directly exploit existing pointwise neural network models as base components. We extend the Noise-Contrastive Estimation approach with a triplet ranking loss function to exploit interactions in triplet inputs over the question paired with positive and negative examples. Experiments on TrecQA and WikiQA datasets show that our approach achieves state-of-the-art effectiveness without the need for external knowledge sources or feature engineering.

1. INTRODUCTION

Answer selection is an important component of an overall question answering system: given a question q and a candidate set of sentences $\{c_1, c_2, \dots c_n\}$, the task is to identify sentences that contain the answer. In a standard pipeline architecture [11], answer selection is applied to the output of a module that performs textual retrieval or lightweight term-based matching. Selected sentences can then be directly presented to users or serve as input to subsequent stages that identify “exact” answers [12].

Although answer selection is formally considered a *pointwise* classification problem, in reality candidate sentences are ranked in decreasing probability of containing the answer, and results are evaluated using similarity measurement metrics on ranked lists. The nature of the task inspired us to formalize it as a ranking problem.

In this work, we develop a novel *pairwise* ranking approach to learn the relative order of answer pairs. Given a question sentence, our approach takes a pair of candidate answer sentences as input and learns to predict which answer is more relevant to the question. We use Noise-Contrastive Estimation to learn joint representations of the triplet input (question, positive answer, negative answer) directly, then stack a triplet ranking loss function on top to learn nonlinear feature correlations from the joint representations. The objective is to minimize the total number of inversions in the rankings.

Our approach is flexible in that it can take advantage of existing pointwise models as base components. We use two recent pointwise neural networks that perform either sentence-level [3] or word-level modeling [4], both of which are competitive in various text processing tasks. We demonstrate the effectiveness of our approach against competitive pointwise baselines [4, 3, 20, 22, 10, 6, 16, 9] for the answer selection task. We show that joint representation learning from triplets comprising the question and both positive and negative answers is superior than learning a pointwise representation on (question, positive answer) pairs. Experiments on both TrecQA and WikiQA datasets show that our approach achieves state-of-the-art effectiveness without using sparse features, syntactic parsers, or external knowledge sources like WordNet.

2. RELATED WORK

There has been much recent work on applying neural networks to answer selection [3, 4, 5, 22, 14, 13, 10]. Previous work has been based on pointwise neural network models. For example, He et al. [3] developed a multi-perspective convolutional neural network model that incorporates fine-grained sentence representations. He and Lin [4] developed a 19-layer deep convolutional neural network model and a similarity focus layer to encourage comparisons between word contexts across sentences. Other researchers [5, 21] have also adopted attention mechanisms for convolutional neural networks to better model interactions between input sentences. Another research direction is the use of extra sparse features (e.g., BM25) [10, 14].

In contrast to previous work, we propose a pairwise ranking approach that takes advantage of Noise-Contrastive Estimation (NCE) for model training. The basic idea behind NCE is that a good model should be able to discriminate a good sample from its neighboring bad samples. Given pairs of positive and negative samples, the model should learn differentiable representations to distinguish positive and negative samples. This idea has been studied for word representation learning [7].

3. MODEL ARCHITECTURE

We show the overall architecture of our pairwise ranking model in Figure 1, consisting of two major components.

CIKM'16 , October 24 - 28, 2016, Indianapolis, IN, USA

ISBN 978-1-4503-4073-1/16/10. . . $15.00

DOI: http://dx.doi.org/10.1145/2983323.2983872

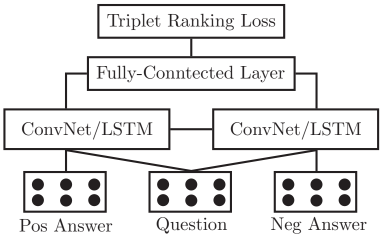


Figure 1: Architecture of our pairwise ranking model, which is trained on triplets comprised of (question, positive answer, negative answer).

The base component is comprised of two pointwise neural network models, each of which takes a pair of (question, answer) sentences and produces a similarity score to represent the semantic distance of the pair. Our pairwise ranking model is expected to output a larger similarity score given the positive pair and a smaller score given the negative pair. On top, we evaluate a triplet ranking loss function (see below) to learn the joint representation of answer pairs.

Our model has a "Siamese" structure [2], which consists of two parallel pointwise models, each processing a pair of question and answer sentences. Parameters of the two base models are shared during training. In testing, we take one of the two base models to produce the feature representation of a (question, answer) pair, and then feed those features into the fully-connected layer for computing a relevance score. Although many previous pointwise models also use the "Siamese" structure, there are important differences: each base component of a typical pointwise model only takes one input sentence, whereas each base component of our pairwise model takes a (question, answer) pair as input and explores features interactions between the sentence pairs.

In this work we use existing published work [3, 4] as the base pointwise models; these have achieved state-of-the-art effectiveness on multiple tasks, i.e., paraphrase identification, semantic textual similarity measurements, and question answering. More importantly, these two models represent different approaches to model design. He et al. [3] focus on *sentence-level* modeling, which tries to capture semantic similarities from multiple perspectives by applying different types of similarity metrics, convolutional filters, poolings, and window sizes on input sentences. This model also incorporates a structured similarity layer over sentence representations for fine-grained comparisons. He and Lin [4] propose a *word-level* model and develop a pairwise word interaction layer to explicitly identify word pairs across input sentences, a similarity focus layer to guide model attention onto important word pairs, and a 19-layer deep ConvNet for classification. Due to limited space, we elide technical details of both and refer readers to the original papers.

Experimental results in Section 4 show that our pairwise ranking approach, when paired with each as the base component, is able to make further effectiveness improvements for both types of models.

3.1 Triplet Ranking Loss Function

Both word-level and sentence-level base components take input sentence pairs of (question, answer) and produce a score through a representation function $f(\cdot)$. This score captures how semantically similar the two input sentences are. Our goal is to learn a representation function $f(\cdot)$ such that given some question q, positive pairs (q, p^+) are assigned larger similarity scores than negative pairs (q, p^-):

$$f(q,p^+) > f(q,p^-), \forall q, p^+, p^-$$

where q, p^+, p^- denote the question, positive answer, and negative answer, respectively. We then use a triplet ranking loss, which minimizes the distance between the question q and a positive answer p^+, and maximizes the distance between the q and a negative answer p^-, summed over positive pairs (q, p^+) and the corresponding negative pool $p^- \in N(q, p^+)$:

$$\min_W \sum_{(q,p^+)} \sum_{p^- \in N} max(0, 1 - (f(q,p^+) - f(q,p^-)) + \lambda \|W\|^2$$

where λ is a regularization parameter, and W is the parameters of neural network model $f(\cdot)$.

3.2 Sampling Strategy

To counter overfitting, it is common practice to train a model with a large variety of samples. However, due to its pairwise nature, our pairwise ranking model requires $O(N^2)$ time complexity to enumerate all pairs, which is computationally impractical. Therefore, it is essential to balance the coverage of training samples and limit computational resources. We use three negative sampling strategies to select the most informative negative samples.

Random Sampling. We randomly select a number of negative samples for each positive answer.

Max Sampling. We select the most difficult negative samples. In each epoch, we compute the similarities between all (p^+, p^-) pairs using the trained model from the previous training epoch. Then we select the negative answers by maximizing their similarities to the positive answer:

$$maxNeg^i(p^-) = \arg\max_{p^-} sim(f^{i-1}(q,p^+), f^{i-1}(q,p^-))$$

where $maxNeg^i(p^-)$ is the selected negative sample in epoch i, $f^{i-1}(\cdot,\cdot)$ is the trained model in epoch $i-1$, and *sim* is the *cosine* distance. In the first epoch, we randomly select negative samples.

Mix Sampling. We take advantages of both random sampling and max sampling by selecting half of the samples from each strategy.

4. EXPERIMENTAL SETUP

We evaluated our ranking model on two question answering datasets. Relevant statistics are shown in Table 1.

TrecQA. The TrecQA dataset [15] is a widely-used benchmark for question answering, collected from the TREC Question Answering tracks and packaged by Yao et al. [19]. In the literature [20, 22, 10, 6, 16, 9], we observe two versions of TrecQA: both have the same training set but their development and test sets differ due to different pre-processing.

Previous work [22, 10, 6, 4] used the version that has 82 questions in the development set and 100 questions in the test set (what we call "Raw TrecQA"). However, there exists questions that have no answer sentences.[1] More recent work [16, 9] further cleaned the dataset by removing

[1] MAP and MRR scores computed with the official `trec_eval` scorer do not change if questions with empty answer sets are removed, since the scorer will ignore questions with empty answer set.

Dataset	Split	#Questions	#Pairs	%PosRate
Raw TrecQA	TRAIN	1229	53417	12.0
	DEV	82	1148	19.3
	TEST	100	1517	18.7
Clean TrecQA	DEV	65	1117	18.4
	TEST	68	1442	17.2
WikiQA	TRAIN	873	8672	12.0
	DEV	126	1130	12.4
	TEST	243	2351	12.5

Table 1: Statistics of TrecQA and WikiQA datasets

questions that have only positive/negative answers or no answers, resulting in only 65 questions in the development set and 68 questions in the test set (what we call "Clean TrecQA"). We evaluated our model on both versions for a fair comparison against previous work, and we show that the MAP/MRR scores reported on both TrecQA versions are *not* comparable.

WikiQA. The open domain question-answering WikiQA data was collected from Bing query logs [18]. For each question, the authors selected Wikipedia pages and used sentences in the summary paragraph as candidates, which were then annotated on a crowdsourcing platform. We follow the same pre-processing steps as Yang et al. [18], where questions with no correct candidate answers are excluded and answer sentences are truncated to 40 tokens.

We conducted experiments with two base components, each with its own settings. We used the GloVe word embeddings [8] for the sentence-level model [3] and PARAGRAM-PHRASE word embeddings [17] for the word-level model [4]. Both word embeddings have 300 dimensions. Words not found in the vocabulary were initialized randomly with values uniformly sampled from $[-.05, .05]$. We did not update word embeddings in all experiments. SGD with a learning rate of 10^{-3} for the sentence-level base component and RMS-PROP with a learning rate of 10^{-4} for the word-level base component were used for training. We chose these different word embeddings and optimizers in order to keep experimental settings the same as in previous work [3, 4]. We used the tanh function as the non-linear activation function and a dropout layer ($p = 0.5$) in the fully-connected layer, plus parameter regularization ($\lambda = 10^{-4}$). Our code and data are available online.[2]

Effectiveness is measured in terms of Mean Average Precision (MAP) and Mean Reciprocal Rank (MRR). In all experiments, we selected training models that obtain the best MRR scores on the development set for testing.

5. RESULTS

The results of our experiments on the three different test sets are shown in Table 2. SentLevel denotes the multi-perspective model [3] and WordLevel denotes the pairwise word interaction model [4]. The rows with PairwiseRank show our pairwise ranking approach with the use of the two different base components (either the WordLevel or SentLevel model). We compare our results to others reported in the literature on an ACL wiki site [1].

Reference	MAP	MRR
Wang et al. [14] s(2015)	0.713	0.791
Miao et al. [6] (2015)	0.734	0.812
Severyn et al. [10] (2015)	0.746	0.808
SentLevel [3] (2015)	0.762	0.830
WordLevel [4] (2016)	0.755	0.825
PairwiseRank+SentLevel	**0.780**	**0.834**
PairwiseRank+WordLevel	0.764	0.827

(a) Raw TrecQA

Reference	MAP	MRR
Santos et al. [9] (2016)	0.753	0.851
Wang et al. [16] (2016)	0.771	0.845
SentLevel [3] (2015)	0.777	0.836
WordLevel [4] (2016)	0.738	0.827
PairwiseRank+SentLevel	**0.801**	**0.877**
PairwiseRank+WordLevel	0.762	0.854

(b) Clean TrecQA

Reference	MAP	MRR
Miao et al. [6] (2015)	0.689	0.707
Santos et al. [9] (2016)	0.689	0.696
Wang et al. [16] (2016)	0.706	0.723
SentLevel [3] (2015)	0.693	0.709
WordLevel [4] (2016)	**0.709**	**0.723**
PairwiseRank+SentLevel	0.701	0.718
PairwiseRank+WordLevel	0.693	0.710

(c) WikiQA

Table 2: Results on TrecQA and WikiQA.

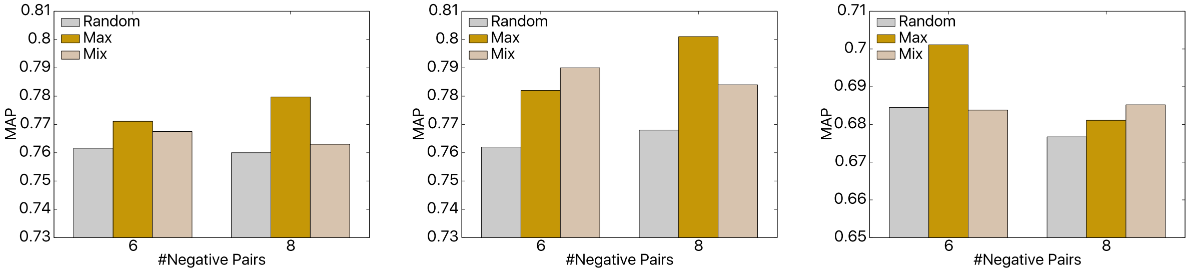
Our best results for all the three datasets in Table 2 were obtained from max sampling. On both the raw and clean versions of the TrecQA data, our PairwiseRank+SentLevel model achieves MAP and MRR scores that are among the best reported in the literature. The models achieve better effectiveness on clean TrecQA than on raw TrecQA. This is because the clean version removes questions with only negative answers: these questions will always have zero values when computing MAP and MRR scores, thereby degrading the overall effectiveness. Thus, numbers reported on raw and clean TrecQA data should not be compared.

In terms of the comparison between the original models and our pairwise ranking models, the two original base models (WordLevel and SentLevel) achieve competitive effectiveness, while our pairwise ranking models (PairwiseRank +SentLevel/WordLevel) can still improve on them in most cases. On the clean TrecQA data, this improvement is quite large. On the WikiQA data, the WordLevel model remains the best, but PairwiseRank+SentLevel improves over the base SentLevel model, bringing its effectiveness up to being on par with our best model.

Overall, these empirical results show that the joint representations learned from the triplet inputs are effective, and that our pairwise ranking approach is able to exploit such joint information.

We also studied the effects of the three different sampling strategies. Figures 2 and 3 show the effectiveness of different sampling strategies with the PairwiseRank+SentLevel and PairwiseRank+WordLevel models. We set the number of negative samples to six and eight. We only show the MAP figures here due to space limitations, but the patterns for MRR scores are generally similar. For the PairwiseRank+SentLevel and PairwiseRank+WordLevel mod-

[2] https://github.com/Jeffyrao/pairwise-neural-network

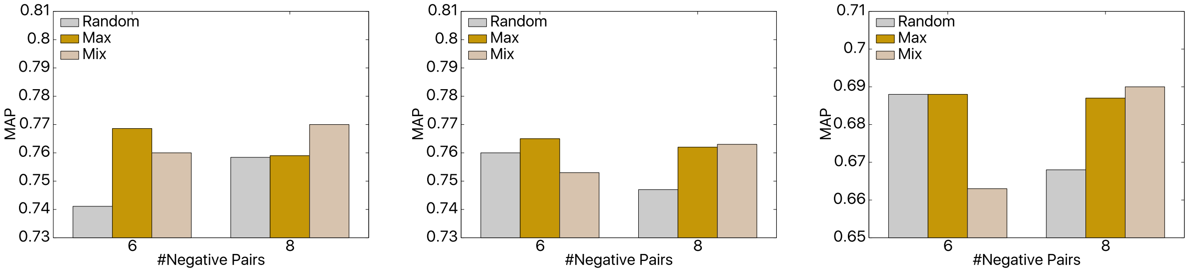


(a) Raw TrecQA

(b) Clean TrecQA

(c) WikiQA

Figure 2: Comparison of sampling strategies for PairwiseRank+SentLevel.



(a) Raw TrecQA

(b) Clean TrecQA

(c) WikiQA

Figure 3: Comparison of sampling strategies for PairwiseRank+WordLevel.

els, max and mix sampling consistently outperforms random sampling. In most settings, max sampling obtains the best performance. This shows that more “challenging” negative training samples are beneficial for training a better model in our pairwise ranking approach.

6. CONCLUSION

In this work, we proposed and evaluated a novel contrastive learning approach to answer selection that can use existing deep neural network models as plug-in components. Experiments show that our contrastive learning approach can improve upon the component models under competitive settings on standard QA datasets. We present three strategies for selecting negative samples and demonstrate the effectiveness of selecting the most “difficult” training examples to distinguish between good and bad answers.

Acknowledgments. This work was supported by the U.S. National Science Foundation (NSF) under IIS-1218043 and CNS-1405688 and the Natural Sciences and Engineering Research Council of Canada (NSERC). Any opinions, findings, conclusions, or recommendations expressed are solely those of the authors.

7. REFERENCES

[1] ACL. Question answering (state of the art), http://aclweb.org/aclwiki/index.php?title=Question_Answering_(State_of_the_art), accessed Aug., 18, 2016.

[2] J. Bromley, J. W. Bentz, L. Bottou, I. Guyon, Y. LeCun, C. Moore, E. Säckinger, and R. Shah. Signature verification using a “siamese” time delay neural network. *IJPRAI*, 1993.

[3] H. He, K. Gimpel, and J. Lin. Multi-perspective sentence similarity modeling with convolutional neural networks. *EMNLP*, 2015.

[4] H. He and J. Lin. Pairwise word interaction modeling with deep neural networks for semantic similarity measurement. *NAACL*, 2016.

[5] H. He, J. Wieting, K. Gimpel, J. Rao, and J. Lin. UMD-TTIC-UW at SemEval-2016 task 1: Attention-based multi-perspective convolutional neural networks for textual similarity measurement. *SemEval*, 2016.

[6] Y. Miao, L. Yu, and P. Blunsom. Neural variational inference for text processing. *arXiv:1511.06038*, 2015.

[7] T. Mikolov and J. Dean. Distributed representations of words and phrases and their compositionality. *NIPS*, 2013.

[8] J. Pennington, R. Socher, and C. D. Manning. GloVe: Global vectors for word representation. *EMNLP*, 2014.

[9] C. d. Santos, M. Tan, B. Xiang, and B. Zhou. Attentive pooling networks. *arXiv:1602.03609*, 2016.

[10] A. Severyn and A. Moschitti. Learning to rank short text pairs with convolutional deep neural networks. *SIGIR*, 2015.

[11] S. Tellex, B. Katz, J. Lin, G. Marton, and A. Fernandes. Quantitative evaluation of passage retrieval algorithms for question answering. *SIGIR*, 2003.

[12] E. M. Voorhees. Overview of the TREC 2002 question answering track. *TREC*, 2002.

[13] D. Wang and E. Nyberg. CMU OAQA at TREC 2015 LiveQA: Discovering the right answer with clues. *TREC*, 2015.

[14] D. Wang and E. Nyberg. A long short-term memory model for answer sentence selection in question answering. *ACL*, 2015.

[15] M. Wang, N. A. Smith, and T. Mitamura. What is the Jeopardy model? A quasi-synchronous grammar for QA. *EMNLP-CoNLL*, 2007.

[16] Z. Wang, H. Mi, and A. Ittycheriah. Sentence similarity learning by lexical decomposition and composition. *arXiv:1602.07019*, 2016.

[17] J. Wieting, M. Bansal, K. Gimpel, and K. Livescu. Towards universal paraphrastic sentence embeddings. *ICLR*, 2016.

[18] Y. Yang, W.-t. Yih, and C. Meek. WikiQA: A challenge dataset for open-domain question answering. *EMNLP*, 2015.

[19] X. Yao, B. Van Durme, C. Callison-Burch, and P. Clark. Answer extraction as sequence tagging with tree edit distance. *HLT-NAACL*, 2013.

[20] W.-t. Yih, M.-W. Chang, C. Meek, and A. Pastusiak. Question answering using enhanced lexical semantic models. *ACL*, 2013.

[21] W. Yin, H. Schütze, B. Xiang, and B. Zhou. ABCNN: Attention-based convolutional neural network for modeling sentence pairs. *arXiv:1512.05193*, 2015.

[22] L. Yu, K. M. Hermann, P. Blunsom, and S. Pulman. Deep learning for answer sentence selection. *NIPS deep learning workshop*, 2014.