On the Benefit of Incorporating External Features in a Neural Architecture for Answer Sentence Selection

Ruey-Cheng Chen, Evi Yulianti, Mark Sanderson, W. Bruce Croft†
RMIT University, Melbourne, Australia
†University of Massachusetts, Amherst, MA, USA
{ruey-cheng.chen,evi.yulianti,mark.sanderson}@rmit.edu.au,croft@cs.umass.edu

ABSTRACT

Incorporating conventional, unsupervised features into a neural architecture has the potential to improve modeling effectiveness, but this aspect is often overlooked in the research of deep learning models for information retrieval. We investigate this incorporation in the context of answer sentence selection, and show that combining a set of query matching, readability, and query focus features into a simple convolutional neural network can lead to markedly increased effectiveness. Our results on two standard question-answering datasets show the effectiveness of the combined model.

CCS CONCEPTS

•**Information systems** → **Information retrieval; Question answering;** •**Computing methodologies** → *Neural networks;*

KEYWORDS

Answer sentence selection, external features, convolutional neural networks

ACM Reference format:
Ruey-Cheng Chen, Evi Yulianti, Mark Sanderson, W. Bruce Croft†. 2017. On the Benefit of Incorporating External Features in a Neural Architecture for Answer Sentence Selection. In *Proceedings of SIGIR '17, Shinjuku, Tokyo, Japan, August 07-11, 2017,* 4 pages.
DOI: http://dx.doi.org/10.1145/3077136.3080705

1 INTRODUCTION

Deep learning approaches have recently become a central methodology in the research of question answering (QA). Many recent attempts in this area have focused on utilizing neural architectures, such as convolutional neural networks (CNN) [8, 19], long short-term memory networks [12], or attention mechanisms [15, 18], to explicitly model high-level question-answer structures. These advances outperform conventional approaches, which are based on engineered heuristics. However, whether deep learning will completely remove the need for such features remains an open question.

SIGIR '17, Shinjuku, Tokyo, Japan

DOI: http://dx.doi.org/10.1145/3077136.3080705

The lack of understanding as to whether the features are needed in a neural architecture is what we address in this paper. While the development of new models for question answering has been moving rapidly ahead in the past few years, a wealth of proven useful results from prior art [2, 9, 10, 16] were usually ignored. Recently, there has been some evidence pointing a value in using features [11, 19] in neural network models. Commonly used neural network substructures, such as multilayer perceptrons [3], have the capability to combine a large number of external features, so incorporating all available signals in a neural network could improve effectiveness and provide robust measurement of any effect [1].

In this paper, we expand on past work using an extensive set of experiments. We demonstrate that, by incorporating a list of 21 common text features into a state-of-the-art CNN model, one can achieve an effectiveness comparable to the currently best reported results on the TREC QA dataset and the WikiQA data.

2 EXTERNAL FEATURES

Our hypothesis is that one can assist relevance modeling with a set of external features to capture aspects of the data that are different to those captured in a neural network model. The chosen set of features should also cover basic signals that can be easily reproduced and implemented. In our experiments, we settle on a set of simple features known to be useful for question answering. We focus on retrieval and readability features. There are two motivations for this approach: 1) such features are better understood by information retrieval practitioners, and 2) they are relatively cheap to implement. Thus, we precluded the use of some sophisticated NLP features in our experiments, such as convolutional tree kernel [11] or syntactic similarity [10].

The full list of features is given in Table 1, they are divided into three categories.

Lexical and semantic matching. The first group of features address non-factoid question answering, described Yang et al. [16], were first selected, which cover topical relevance and semantic relatedness measures. The reference package released by Yang et al. was used in our implementation. For computing the language model and BM25 scores, we empirically set both μ in the language model and the average document length in BM25 to a fixed value of ten.

Readability. The second group of features focus on text readability [5], which is a set of seven common surface readability features, such as number of words/syllables per sentence or the complex-word ratio, plus one feature representing the notable Dale-Chall readability formula. We did not include other readability indices as

Table 1: List of unsupervised features used in this study

Lexical and semantic matching	
`Length`	Number of terms in the sentence
`ExactMatch`	Whether query is a substring
`Overlap`	Fraction of query terms covered
`OverlapSyn`	Fraction of query synonyms covered
`LM`	Language model score
`BM25`	BM25 score
`ESA`	Cosine similarity with the query ESA vector
`TAGME`	Overlap between query/sentence entities
`Word2Vec`	Cosine similarity with query word vectors
Readability	
`CPW`	Number of characters per word
`SPW`	Number of syllables per word
`WPS`	Number of words per sentence
`CWPS`	Number of complex words per sentence
`CWR`	Fraction of complex words
`LWPS`	Number of long words per sentence
`LWR`	Fraction of long words (> 7 chars)
`DaleChall`	The Dale-Chall readability index
Focus	
`MatchedNGram`	Maximum semantic relatedness between head question k-gram and any answer n-gram. See (1); 4 variants of (k, n) were used

most of these indices can be represented as a linear combination of the surface features.

Focus. The third group of features used four parameterized variants of a newly proposed feature `MatchedNGram` to account for the matching between question head words and the potential answer n-gram. The feature takes the maximum of the semantic similarity between the first k question words and any n-grams in the answer, using the cosine similarity between question/answer word vectors as the similarity measure. Given k and n, the feature is defined as follows:

$$\texttt{MatchedNGram}(Q, A) = \max_{l} \cos\left[\sum_{i=1}^{k} \vec{q}_i, \sum_{j=l}^{l+n-1} \vec{a}_j\right]. \quad (1)$$

This simple feature explicitly looks for best matching answer n-grams with respect to question head phrases, such as "who invented", "how many", or "what year did". Word embeddings are leveraged in the computation of the similarity measure. Also, not all combinations of (k, n) are found effective. In our experiments, we empirically chose four best configurations of (k, n), which are $\{(k, n) \mid k \in \{2, 3\}, n \in \{2, 3\}\}$, based on their effectiveness within a learning-to-rank model.

3 EXPERIMENTS

Our evaluation was based on two widely used question answering benchmarks: the TREC QA and the WikiQA datasets. The first benchmark was originally developed for the task of identifying correct answer factoids in retrieved passages. We used the version prepared by Wang et al. [13], with 1,229 questions in the larger training set, 82 in the dev set, and 100 in the test set. No further filtering was performed on the data (the "raw" setting [7]).[1]

The second benchmark, WikiQA, was created by Yang et al. over the English Wikipedia summary passages and the Bing query logs [17], with crowdsourced annotations. This new benchmark is developed to counter biases introduced in the creation TREC QA: the reliance on using lexical overlap with the question as the sole indication of a candidate answer. Hence, this dataset is by design made more challenging for retrieval based methods. Some major follow-up works [7, 18] used a split that includes questions with all positive labels, a version slightly different from the split distributed in the original data. We used the same split as in Rao et al. and used 873 questions in the training set, 126 in the dev set, and 243 in the test set [7].

3.1 Neural Network Configuration

For the choice of a base system, which would serve as the experimental control, we chose to implement a bi-CNN architecture as proposed in Severyn and Moschitti [8]. This state-of-the-art model is preferred over other candidates, i.e., attention-based CNN [18], for the ease of implementation and parameter optimization. This architecture is fairly robust and in most cases overly excessive parameter tuning is not required.

Convolutional Neural Networks. Our implementation follows closely to the experimental setting in the original paper. Two sets of word embeddings were used: one with 50 dimensions, developed on top of English Wikipedia and the AQUAINT corpus [8], and the other a 300-dimension pre-trained model released by the word2vec project, using 100-billion words from Google News. The sparse word overlapping indicator features were also used in the convolutional layer [11]. The proposed 21 features were incorporated in the fully-connected layer which also combines pooled representations for the question and the answer sentences. The size of the kernel is set to 100 throughout the experiments. We used hyperbolic tangent tanh as the activation function and max pooling in the pooling layer. The network is trained by using stochastic gradient descent with mini batches. The batch size is set to 50 and AdaDelta update [20] was used with $\rho = 0.95$. Early stopping was deployed tracking the change of dev set effectiveness, and as a result the training almost always stopped in 5 to 10 epochs. We also experimented with dropout in two experimental runs by sweeping through a small set of dropout rates $\{0.1, 0.2, \ldots, 0.9\}$.

The attention mechanism can also affect the effectiveness of the neural network model. To control for this variable, we implemented a simple attention layer in the base CNN model to approximate the ABCNN-1 model, which is the simplest form of attention mechanism proposed in Yin et al [18]. In mathematical terms, an attention layer takes a question-side feature map $\mathbf{F}_q \in \mathcal{R}^{n_q \times d}$ and an answer-side feature map $F_a \in \mathcal{R}^{n_a \times d}$ as input. Here, n_q and n_a denote the maximum question/answer sentence length, respectively, and d denotes the dimension of the word embeddings.

[1] Some previous work chose to remove questions that contain no answers and led to two inconsistent data splits "raw" an "clean", so results on one split are not directly comparable to those on the other [7].

Table 2: Effectiveness results on TREC QA and WikiQA datasets. Best-performing runs in each word-embedding group are underlined and the overall best result on individual benchmarks printed in boldface. Relative improvements (+/-%) are measured against the group control (base system). Significant differences with respect to bagged LambdaMART and the group control are indicated by †/‡and */**, respectively, for $p < 0.05$/$p < 0.01$ using the paired t-test.

			TREC QA			WikiQA		
System	Attn?	Drop?	MAP	MRR	S@1	MAP	MRR	S@1
Runs (AQUAINT/Wikipedia)								
CNN	×	×	76.2	80.9	73.7	66.0	67.4	52.3
Combined Model	×	×	77.9 (+2.2%)	82.2 (+1.6%)	74.7 (+1.4%)	67.2 (+1.8%)‡	68.5 (+1.6%)‡	53.9 (+3.1%)‡
Combined Model	×	✓	**78.2 (+2.6%)**	**83.7 (+3.5%)**	76.8 (+4.2%)	64.7 (-2.0%)	65.7 (-2.5%)	48.6 (-7.1%)
CNN	✓	×	75.4	79.9	71.6	65.3	66.8	52.7
Combined Model	✓	×	77.2 (+2.4%)	81.1 (+1.5%)	72.6 (+1.4%)	70.0 (+7.2%)$^{\ddagger *}$	71.4 (+6.9%)$^{\ddagger *}$	58.4 (+10.8%)$^{\ddagger *}$
Combined Model	✓	✓	77.3 (+2.5%)	82.0 (+2.6%)	74.7 (+4.3%)	69.0 (+5.7%)‡	70.9 (+6.1%)$^{\ddagger *}$	58.4 (+10.8%)‡
Runs (Google News)								
CNN	×	×	76.1	82.3	75.8	67.3	69.1†	57.2‡
Combined Model	×	×	73.8 (-3.0%)	79.2 (-3.8%)	70.5 (-7.0%)	69.2 (+2.8%)‡	70.2 (+1.6%)‡	56.0 (-2.1%)‡
Combined Model	×	✓	74.8 (-1.7%)	80.1 (-2.7%)	71.6 (-5.5%)	69.2 (+2.8%)‡	70.7 (+2.3%)‡	56.4 (-1.4%)‡
CNN	✓	×	75.0	81.1	73.7	66.3	68.3	54.7‡
Combined Model	✓	×	76.5 (+2.0%)	82.5 (+1.7%)	74.7 (+1.4%)	69.4 (+4.7%)‡	71.2 (+4.2%)‡	57.6 (+5.3%)‡
Combined Model	✓	✓	76.3 (+1.7%)	82.5 (+1.7%)	74.7 (+1.4%)	67.9 (+2.4%)‡	69.7 (+2.0%)‡	56.0 (+2.4%)‡
Reference methods								
Bagged LambdaMART			75.7	81.3	72.6	63.0	63.8	46.5
LSTM [12]			71.3	79.1		—	—	
CNN [8]			74.6	80.8		—	—	
aNMM [15]			75.0	81.1		—	—	
ABCNN-3 [18]			—	—		69.2	71.1	
PairwiseRank + SentLevel [7]			78.0	83.4		**70.1**	**71.8**	

The matrix $\mathbf{A} \in \mathcal{R}^{n_q \times n_a}$ representing the "attention" is computed internally to the layer as follows:

$$\mathbf{A}_{i,j} = \frac{1}{1 + \|\mathbf{F}_q[i,:] - \mathbf{F}_a[j,:]\|}, \tag{2}$$

with $\|\cdot\|$ being the euclidean distance function. Then, the layer generates two new attention-based feature maps, $\mathbf{F}'_q$ and $\mathbf{F}'_a$, which are to be combined in the follow-up convolutional layers:

$$\mathbf{F}'_q = A\,W_q \qquad \mathbf{F}'_a = A^T\,W_a, \tag{3}$$

where $W_q \in \mathcal{R}^{n_a \times d}$ and $W_a \in \mathcal{R}^{n_q \times d}$ denote model weights which are to be learned from the data.

3.2 Baselines

A number of published results were included as reference runs in the experiments [8, 12, 15, 18], including a recently proposed neural model PairwiseRank [7]. For comparing with learning-to-rank systems, a Bagged LambdaMART model trained using RankLib[2] over the 21 sentence features is included. The bagged LambdaMART was trained by optimizing NDCG@20 with subsampling rate 0.7, feature sampling rate 0.3, using 300 bags.

[2]https://www.lemurproject.org/ranklib.php

3.3 Results

Table 2 gives the effectiveness results for the TREC QA and the WikiQA datasets. The effectiveness of answer selection is measured by Mean Average Precision (MAP), Mean Reciprocal Rank (MRR), and Success at 1 (S@1), using the `trec_eval` package following Severyn and Moschitti [8]. The experimental runs are divided into four groups, according to the word embeddings in use (AQUAINT/Wikipedia or Google News) and whether the attention mechanism is enabled. In each group, the original CNN model is the experimental control, and runs with external features combined (denoted as Combined Model) are the treatments.

On both sets of data, the PairwiseRank model gives the best results. The baseline bagged LambdaMART model appears to be strong on the TREC QA data, beating a number of neural network models, but it does not appear particularly effective on the more challenging WikiQA dataset.

Our base CNN model is found to be superior to the learning-to-rank model, and it gave better results than the original implementation [8] on the TREC QA benchmark. In general, the Combined Model reliably improves the base CNN model. All experimental configurations appear to benefit from the inclusion of the 21 extra features, with two exceptions: runs using GoogleNews embeddings on the TREC QA benchmark, and one of the dropout runs on the

WikiQA data. On the TREC QA benchmark, we saw an increase of 1.3%–4.3% in the three evaluation metrics. On WikiQA, the increase is around 1.6%–7.2% in MAP/MRR and 2.4%–10.8% in S@1. These increases are in most cases consistent with each other, except that in one particular configuration a decreased S@1 is observed alongside improved MAP and MRR scores.

The AQUAINT/Wikipedia embeddings appear to have a slight advantage over the Google News embeddings. The best performing runs on both benchmarks using this embedding achieved a marked increased in effectiveness compared to the best known results. On the TREC QA benchmark, however, the Combined Model with dropout surpassed the PairwiseRank model [7] in both MAP and MRR. On WikiQA, the Combined Model with the attention mechanism outperformed ABCNN-3 (i.e., a stronger variant of ABCNN-1), with the achieved effectiveness only marginally below the effectiveness of the PairwiseRank model. In most cases, we found that the Combined Model works the best when the attention mechanism is used together without dropout. We conjecture that in this case the attentional CNN model works differently to the data, as external features on their own tend to fit certain aspects well enough.

Based on these results, we conclude that, for answer sentence selection, combining the proposed external features into a convolutional neural architecture has a benefit of improving overall modeling effectiveness. This improvement is evident even when the neural architecture is slightly altered to perform advanced neural functions such as attention mechanism or dropout. The evidence is that the highly tuned convolutional neural architecture failed to model certain aspects in the data, which can be captured with a set of simple features. This points to a limitation of neural network methodology previously not mentioned in research on question answering.

4 RELATED WORK

There is a rich body of work in question answering focused on answer sentence selection [7, 8, 11, 12, 14, 15, 17–19]. Most of these efforts address the architectural issues in neural network models. Yu et al. [19] utilize a CNN architecture to model question-answer pairs, and this approach was taken by Yang et al [17] and Severyn and Moschitti [8], who later expanded the network architecture into a bi-CNN model. Wang et al. [14] decomposed vectors into similar/dissimilar components and used a two-channel CNN to capture the signals. The attention mechanism was investigated in Yin et al. [18] and Yang et al. [15]. He and Lin [4] used a Bidirectional LSTM to model the context of input sentences. Rao et al. [7] proposed a pairwise ranking method that uses two bi-CNN architectures to perform sentence-level pairwise ranking.

Feature engineering has been a popular methodology for modeling question-answer structure, and is still actively used in non-factoid question answering or answer re-ranking [10, 16]. One commonality between these specialized tasks is a focus on retrieving or ranking passage-level answers [6]. Surdeanu et al. [10] proposed using a translation model to capture the mapping between the high-level linguistic representations of the question and the answer. Yang et al. [16] proposed using query matching, semantic, and context features to select answer sentences for non-factoid questions.

5 CONCLUSIONS

We provide empirical evidence to support the use of conventional features in deep learning models on the task of answer sentence selection. We show that a convolutional neural network model benefits from a group of commonly used text features and outperforms the best published result on a commonly used question answering benchmark. The fact that neural networks can still benefit from these conventional features may point to new possibilities in the evolution of new neural architectures. In future work, we will seek to expand this analysis to other neural architectures, such as LSTM-CNN or recurrent neural networks, and other question answering benchmarks that are more recent and larger.

REFERENCES

[1] Timothy G. Armstrong, Alistair Moffat, William Webber, and Justin Zobel. 2009. Improvements That Don'T Add Up: Ad-hoc Retrieval Results Since 1998. In *Proceedings of CIKM '09*. ACM, New York, NY, USA, 601–610.

[2] Matthew W. Bilotti, Jonathan Elsas, Jaime Carbonell, and Eric Nyberg. 2010. Rank Learning for Factoid Question Answering with Linguistic and Semantic Constraints. In *Proceedings of CIKM '10*. ACM, New York, NY, USA, 459–468.

[3] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to Rank Using Gradient Descent. In *Proceedings of ICML '05*. ACM, New York, NY, USA, 89–96.

[4] Hua He and Jimmy Lin. 2016. Pairwise Word Interaction Modeling with Deep Neural Networks for Semantic Similarity Measurement. In *Proceedings of NAACL '16*. 937–948.

[5] Tapas Kanungo and David Orr. 2009. Predicting the Readability of Short Web Summaries. In *Proceedings of WSDM '09*. ACM, New York, NY, USA, 202–211.

[6] Mostafa Keikha, Jae Hyun Park, W. Bruce Croft, and Mark Sanderson. 2014. Retrieving Passages and Finding Answers. In *Proceedings of ADCS '14*. ACM, New York, NY, USA, Article 81, 4 pages.

[7] Jinfeng Rao, Hua He, and Jimmy Lin. 2016. Noise-Contrastive Estimation for Answer Selection with Deep Neural Networks. In *Proceedings of CIKM '16*. ACM, 1913–1916.

[8] Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to Rank Short Text Pairs with Convolutional Deep Neural Networks. In *Proceedings of SIGIR '15*. ACM, 373–382.

[9] Huan Sun, Hao Ma, Wen-tau Yih, Chen-Tse Tsai, Jingjing Liu, and Ming-Wei Chang. 2015. Open Domain Question Answering via Semantic Enrichment. In *Proceedings of WWW '15*. 1045–1055.

[10] Mihai Surdeanu, Massimiliano Ciaramita, and Hugo Zaragoza. 2011. Learning to Rank Answers to Non-Factoid Questions from Web Collections. *Computational Linguistics* 37, 2 (April 2011), 351–383.

[11] Kateryna Tymoshenko, Daniele Bonadiman, and Alessandro Moschitti. 2016. Convolutional Neural Networks vs. Convolution Kernels: Feature Engineering for Answer Sentence Reranking. In *Proceedings of NAACL '16*. 1268–1278.

[12] Di Wang and Eric Nyberg. 2015. A Long Short-Term Memory Model for Answer Sentence Selection in Question Answering. In *Proceedings of ACL '15*. 707–712.

[13] Mengqiu Wang, Noah A. Smith, and Teruko Mitamura. 2007. What is the Jeopardy Model? A Quasi-Synchronous Grammar for QA. In *Proceedings of EMNLP '07*. 22–32.

[14] Zhiguo Wang, Haitao Mi, and Abraham Ittycheriah. 2016. Sentence Similarity Learning by Lexical Decomposition and Composition. In *Proceedings of COLING 2016*. Osaka, Japan, 1340–1349.

[15] Liu Yang, Qingyao Ai, Jiafeng Guo, and W. Bruce Croft. 2016. aNMM: Ranking Short Answer Texts with Attention-Based Neural Matching Model. In *Proceedings of CIKM '16*. ACM, 287–296.

[16] Liu Yang, Qingyao Ai, Damiano Spina, Ruey-Cheng Chen, Liang Pang, W. Bruce Croft, Jiafeng Guo, and Falk Scholer. 2016. Beyond Factoid QA: Effective Methods for Non-factoid Answer Sentence Retrieval. In *Proceedings of ECIR '16*. Springer International Publishing, 115–128.

[17] Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. WikiQA: A Challenge Dataset for Open-Domain Question Answering. In *Proceedings of EMNLP '15*. Lisbon, Portugal, 2013–2018.

[18] Wenpeng Yin, Hinrich Schtze, Bing Xiang, and Bowen Zhou. 2015. ABCNN: Attention-Based Convolutional Neural Network for Modeling Sentence Pairs. *arXiv:1512.05193 [cs]* (Dec. 2015).

[19] Lei Yu, Karl Moritz Hermann, Phil Blunsom, and Stephen Pulman. 2014. Deep Learning for Answer Sentence Selection. *arXiv:1412.1632 [cs]* (Dec. 2014).

[20] Matthew D Zeiler. 2012. ADADELTA: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701* (2012).