arXiv:1603.07044v1 [cs.CL] 23 Mar 2016

Recurrent Neural Network Encoder with Attention for Community Question Answering

Wei-Ning Hsu and **Yu Zhang** and **James Glass**
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139, USA
{wnhsu,yzhang87,jrg}@csail.mit.edu

Abstract

We apply a general recurrent neural network (RNN) encoder framework to community question answering (cQA) tasks. Our approach does not rely on any linguistic processing, and can be applied to different languages or domains. Further improvements are observed when we extend the RNN encoders with a neural attention mechanism that encourages reasoning over entire sequences. To deal with practical issues such as data sparsity and imbalanced labels, we apply various techniques such as transfer learning and multitask learning. Our experiments on the SemEval-2016 cQA task show 10% improvement on a MAP score compared to an information retrieval-based approach, and achieve comparable performance to a strong handcrafted feature-based method.

1 Introduction

Community question answering (cQA) is a paradigm that provides forums for users to ask or answer questions on any topic with barely any restrictions. In the past decade, these websites have attracted a great number of users, and have accumulated a large collection of question-comment threads generated by these users. However, the low restriction results in a high variation in answer quality, which makes it time-consuming to search for useful information from the existing content. It would therefore be valuable to automate the procedure of ranking related questions and comments for users with a new question, or when looking for solutions from comments of an existing question.

Automation of cQA forums can be divided into three tasks: question-comment relevance (Task A), question-question relevance (Task B), and question-external comment relevance (Task C). One might think that classic retrieval models like language models for information retrieval (Zhai and Lafferty, 2004) could solve these tasks. However, a big challenge for cQA tasks is that users are used to expressing similar meanings with different words, which creates gaps when matching questions based on common words. Other challenges include informal usage of language, highly diverse content of comments, and variation in the length of both questions and comments.

To overcome these issues, most previous work (e.g. SemEval 2015 (Nakov et al., 2015)) relied heavily on additional features and reasoning capabilities. In (Rocktäschel et al., 2015), a neural attention-based model was proposed for automatically recognizing entailment relations between pairs of natural language sentences. In this study, we first modify this model for all three cQA tasks. We also extend this framework into a jointly trained model when the external resources are available, i.e. selecting an external comment when we know the question that the external comment answers (Task C).

Our ultimate objective is to classify relevant questions and comments without complicated handcrafted features. By applying RNN-based encoders, we avoid heavily engineered features and learn the representation automatically. In addition, an attention mechanism augments encoders with the ability to attend to past outputs directly. This becomes helpful when encoding longer sequences, since we no

longer need to compress all information into a fixed-length vector representation.

In our view, existing annotated cQA corpora are generally too small to properly train an end-to-end neural network. To address this, we investigate transfer learning by pretraining the recurrent systems on other corpora, and also generating additional instances from existing cQA corpus.

2 Related Work

Earlier work of community question answering relied heavily on feature engineering, linguistic tools, and external resource. (Jeon et al., 2006) and (Shah and Pomerantz, 2010) utilized rich non-textual features such as answer's profile. (Grundström and Nugues, 2014) syntactically analyzed the question and extracted name entity features. (Harabagiu and Hickl, 2006) demonstrated a textual entailment system can enhance cQA task by casting question answering to logical entailment.

More recent work incorporated word vector into their feature extraction system and based on it designed different distance metric for question and answer (Tran et al., 2015) (Belinkov et al., 2015). While these approaches showed effectiveness, it is difficult to generalize them to common cQA tasks since linguistic tools and external resource may be restrictive in other languages and features are highly customized for each cQA task.

Very recent work on answer selection also involved the use of neural networks. (Wang and Nyberg, 2015) used LSTM to construct a joint vector based on both the question and the answer and then converted it into a learning to rank problem. (Feng et al., 2015) proposed several convolutional neural network (CNN) architectures for cQA. Our method differs in that RNN encoder is applied here and by adding attention mechanism we jointly learn which words in question to focus and hence available to conduct qualitative analysis. During classification, we feed the extracted vector into a feed-forward neural network directly instead of using mean/max pooling on top of each time steps.

3 Method

In this section, we first discuss long short-term memory (LSTM) units and an associated attention mechanism. Next, we explain how we can encode a pair of sentences into a dense vector for predicting relationships using an LSTM with an attention mechanism. Finally, we apply these models to predict question-question similarity, question-comment similarity, and question-external comment similarity.

3.1 LSTM Models

LSTMs have shown great success in many different fields. An LSTM unit contains a memory cell with self-connections, as well as three multiplicative gates to control information flow. Given input vector x_t, previous hidden outputs h_{t-1}, and previous cell state c_{t-1}, LSTM units operate as follows:

$$X = \begin{bmatrix} x_t \\ h_{t-1} \end{bmatrix} \tag{1}$$

$$i_t = \sigma(\mathbf{W_{iX}}X + \mathbf{W_{ic}}c_{t-1} + \mathbf{b_i}) \tag{2}$$

$$f_t = \sigma(\mathbf{W_{fX}}X + \mathbf{W_{fc}}c_{t-1} + \mathbf{b_f}) \tag{3}$$

$$o_t = \sigma(\mathbf{W_{oX}}X + \mathbf{W_{oc}}c_{t-1} + \mathbf{b_o}) \tag{4}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot tanh(\mathbf{W_{cX}}X + \mathbf{b_c}) \tag{5}$$

$$h_t = o_t \odot tanh(c_t) \tag{6}$$

where i_t, f_t, o_t are input, forget, and output gates, respectively. The sigmoid function $\sigma()$ is a soft gate function controlling the amount of information flow. Ws and bs are model parameters to learn.

3.2 Neural Attention

A traditional RNN encoder-decoder approach (Sutskever et al., 2014) first encodes an arbitrary length input sequence into a fixed-length dense vector that can be used as input to subsequent classification models, or to initialize the hidden state of a secondary decoder. However, the requirement to compress all necessary information into a single fixed length vector can be problematic. A neural attention model (Bahdanau et al., 2014) (Cho et al., 2014) has been recently proposed to alleviate this issue by enabling the network to attend to past outputs when decoding. Thus, the encoder no longer needs to represent an entire sequence with one vector; instead, it encodes information into a sequence of vectors, and adaptively chooses a subset of the vectors when decoding.

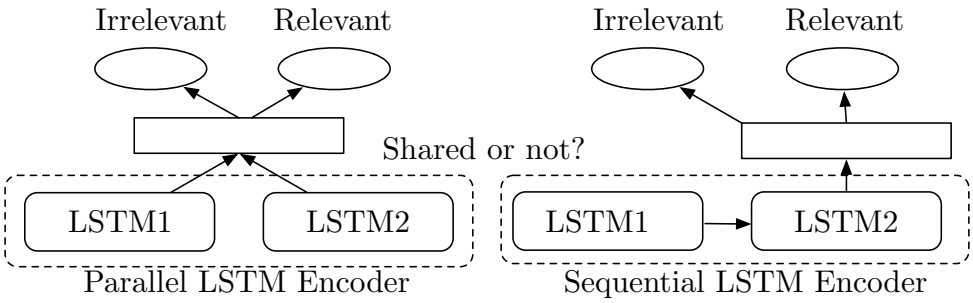


Figure 1: RNN encoder for related question/comment selection.

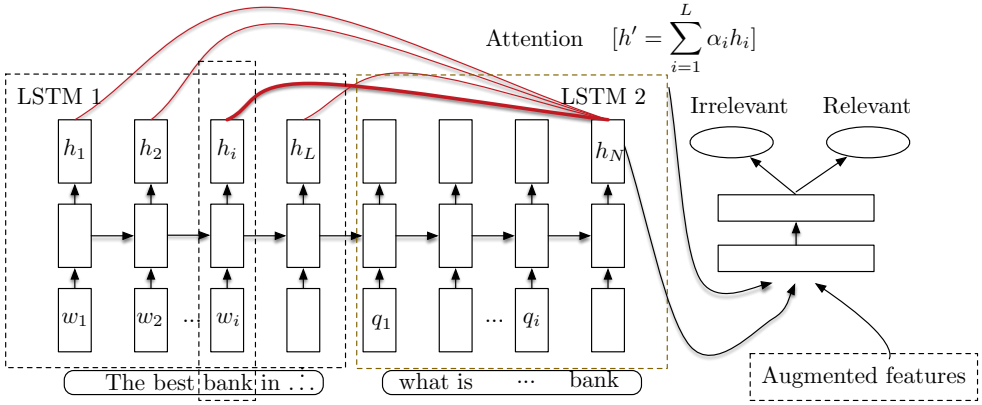


Figure 2: Neural attention model for related question/comment selection.

3.3 Predicting Relationships of Object Pairs with an Attention Model

In our cQA tasks, the pair of objects are (question, question) or (question, comment), and the relationship is relevant/irrelevant. The left side of Figure 1 shows one intuitive way to predict relationships using RNNs. Parallel LSTMs encode two objects independently, and then concatenate their outputs as an input to a feed-forward neural network (FNN) with a softmax output layer for classification.

The representations of the two objects are generated independently in this manner. However, we are more interested in the relationship instead of the object representations themselves. Therefore, we consider a serialized LSTM-encoder model in the right side of Figure 1 that is similar to that in (Rocktäschel et al., 2015), but also allows an augmented feature input to the FNN classifier.

Figure 2 illustrates our attention framework in more detail. The first LSTM reads one object, and passes information through hidden units to the second LSTM. The second LSTM then reads the other object and generates the representation of this pair after the entire sequence is processed. We build another FNN that takes this representation as input to classify the relationship of this pair.

By adding an attention mechanism to the encoder, we allow the second LSTM to attend to the sequence of output vectors from the first LSTM, and hence generate a weighted representation of first object according to both objects. Let h_N be the last output of second LSTM and $M = [h_1, h_2, \cdots, h_L]$ be the sequence of output vectors of the first object. The weighted representation of the first object is

$$h' = \sum_{i=1}^{L} \alpha_i h_i \tag{7}$$

The weight is computed by

$$\alpha_i = \frac{exp(a(h_i, h_N))}{\sum_{j=1}^{L} exp(a(h_j, h_N))} \tag{8}$$

where $a()$ is the importance model that produces a higher score for (h_i, h_N) if h_i is useful to determine

the object pair's relationship. We parametrize this model using another FNN. Note that in our framework, we also allow other augmented features (e.g., the ranking score from the IR system) to enhance the classifier. So the final input to the classifier will be h_N, h', as well as augmented features.

3.4 Modeling Question-External Comments

For task C, in addition to an original question (oriQ) and an external comment (relC), the question which relC commented on is also given (relQ). To incorporate this extra information, we consider a multitask learning framework which jointly learns to predict the relationships of the three pairs (oriQ/relQ, oriQ/relC, relQ/relC).

Figure 3 shows our framework: the three lower models are separate serialized LSTM-encoders for the three respective object pairs, whereas the upper model is an FNN that takes as input the concatenation of the outputs of three encoders, and predicts the relationships for all three pairs. More specifically, the output layer consists of three softmax layers where each one is intended to predict the relationship of one particular pair.

For the overall loss function, we combine three separate loss functions using a heuristic weight vector β that allocates a higher weight to the main task (oriQ-relC relationship prediction) as follows:

$$\mathcal{L} = \beta_1 \mathcal{L}_1 + \beta_2 \mathcal{L}_2 + \beta_3 \mathcal{L}_3 \quad (9)$$

By doing so, we hypothesize that the related tasks can improve the main task by leveraging commonality among all tasks.

4 Experiments

We evaluate our approach on all three cQA tasks. We use the cQA datasets provided by the Semeval 2016 task [1]. The cQA data is organized as follows: there are 267 original questions, each question has 10 related question, and each related question has 10 comments. Therefore, for task A, there are a total number of 26,700 question-comment pairs. For task B, there are 2,670 question-question pairs. For task C, there are 26,700 question-comment pairs. The test dataset includes 50 questions, 500 related questions and 5,000 comments which do not overlap with the training set. To evaluate the performance, we use mean average precision (MAP) and F1 score.

Baseline System: Figure 4 illustrates our baseline systems. The IR-based system is scored by the Google search engine. For each question-comment pair, or question-question pair, we use Google's rank to calculate the MAP. While there is no training on the target data, we expect that Google used many external resources to produce these ranks. The feature-rich system is that proposed by (Belinkov et al., 2015) in SemEval-2015. In this approach, they compute *text-based*, *vector-based*, *metadata-based* and *rank-based features* from the pre-processed data. The features are used by a linear SVM for comment selection. This system includes traditional handcrafted features, and some RNN-based features (word vectors). It also includes the information from the IR system (ranked-based). So we believe it is a strong baseline to compare with our model.

RNN encoder: Our system is based on Theano (Bastien et al., 2012; Bergstra et al., 2010). Table 1 gives a list of hyper-parameters we considered. As suggested by (Greff et al., 2015), the hyper-parameters for LSTMs can be tuned independently. We tuned each parameter separately on a development set (split from the training set) and simply picked the best setting. Our experiments show that using word embeddings from Google-News provides modest improvements, but fixing the embedding degrades performance a lot. Also, using separate parameters for LSTMs is better than sharing. For the optimization method, AdaDelta converged faster, but AdaGrad gives better performance. Note that all the parameters were tuned on Task A, and we simply applied them to Task B and C. This is for saving computation, and also because Task A is more well-defined compared to B and C in terms of dataset size and label balance.

4.1 Preliminary Results

Table 2 shows the initial results using the RNN encoder for different tasks. We observe that the attention model always gets better results than the RNN without attention, especially for task C. However, the RNN model achieves a very low F1 score. For task B, it is even worse than the random baseline.

[1] http://alt.qcri.org/semeval2016/task3

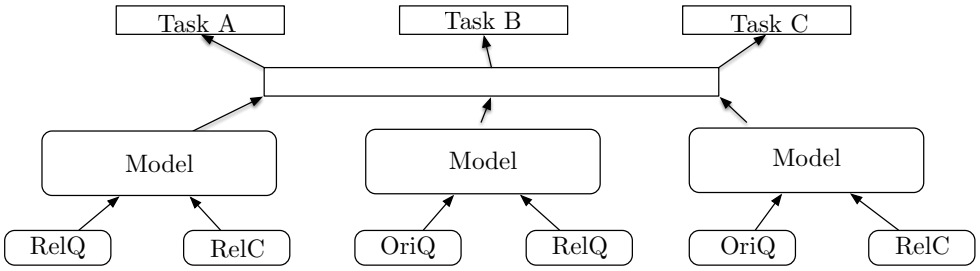


Figure 3: Joint learning for external comment selection.

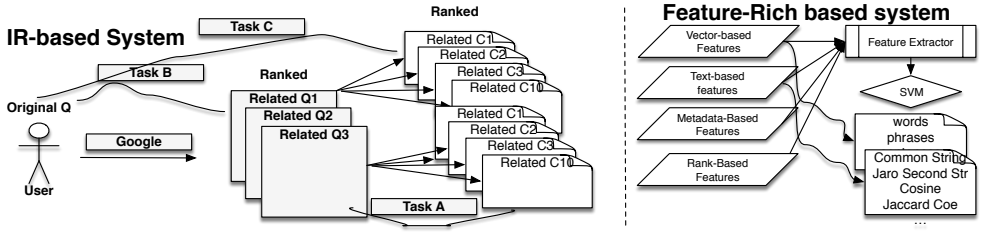


Figure 4: IR-based system and feature-rich based system.

	Task A		Task B		Task C	
Model	MAP	F1	MAP	F1	MAP	F1
Random	0.4860	0.5004	0.5595	**0.4691**	0.1383	0.1277
Parallel LSTM	0.6123	0.6091	0.5553	0.4087	0.2413	0.0057
Seq LSTM	0.6175	0.6063	0.5620	0.4299	0.2356	0.0115
w/ Attention	**0.6239**	**0.6323**	**0.5723**	0.4334	**0.2837**	**0.1449**

Table 2: The RNN encoder results for cQA tasks (bold is best).

Embedding	**init** or random, fix or **update**
Two LSTM	shared or **not**
#cells for LSTM	64, **128**, 256
# nodes for MLP	128, **256**
Optimizer	**AdaGrad**, AdaDelta, SGD
learning rate	0.001,**0.01**,0.1
Regularizer	**Dropout**, L2 regularization
Dropout rate	0.0, 0.2, 0.3, **0.4**, 0.5
L2	0, 0.001, **0.0001**, 0.00001

Table 1: The hyper-parameters we tuned. Terms in bold represent the selected final parameters.

We believe the reason is because for task B, there are only 2,670 pairs for training which is very limited training for a reasonable neural network. For task C, we believe the problem is highly imbalanced data. Since the related comments did not directly comment on the original question, more than 90% of the comments are labeled as irrelevant to the original question. The low F1 (with high precision and low recall) means our system tends to label most comments as irrelevant. In the following section, we investigate methods to address these issues.

4.2 Robust Parameter Initialization

One way to improve models trained on limited data is to use external data to pretrain the neural network. We therefore considered two different datasets for this task.

- Cross-domain: The Stanford natural language inference (SNLI) corpus (Bowman et al., 2015) has a huge amount of cleaned premise and hypothesis pairs. Unfortunately the pairs are for a different task. The relationship between the premise and hypothesis may be similar to the relation between questions and comments, but may also be different.
- In-domain: since task A seems has reason-

able performance, and the network is also well-trained, we could use it directly to initialize task B.

To utilize the data, we first trained the model on each auxiliary data (SNLI or Task A) and then removed the softmax layer. After that, we retrain the network using the target data with a softmax layer that was randomly initialized.

For task A, the SNLI cannot improve MAP or F1 scores. Actually it slightly hurts the performance. We surmise that it is probably because the domain is different. Further investigation is needed: for example, we could only use the parameter for embedding layers etc. For task B, the SNLI yields a slight improvement on MAP (0.2%), and Task A could give (1.2%) on top of that. No improvement was observed on F1. For task C, pretraining by task A is also better than using SNLI (task A is 1% better than the baseline, while SNLI is almost the same).

In summary, the in-domain pretraining seems better, but overall, the improvement is less than we expected, especially for task B, which only has very limited target data. We will not make a conclusion here since more investigation is needed.

4.3 Multitask Learning

As mentioned in Section 3.4, we also explored a multitask learning framework that jointly learns to predict the relationships of all three tasks. We set 0.8 for the main task (task C) and 0.1 for the other auxiliary tasks. The MAP score did not improve, but F1 increases to 0.1617. We believe this is because other tasks have more balanced labels, which improves the shared parameters for task C.

4.4 Augmented data

There are many sources of external question-answer pairs that could be used in our tasks. For example: WebQuestion (was introduced by the authors of SEMPRE system (Berant et al., 2013)) and The SimpleQuestions dataset [2]. All of them are positive examples for our task and we can easily create negative examples from it. Initial experiments indicate that it is very easy to overfit these obvious negative examples. We believe this is because our negative examples are non-informative for our task and just introduce noise.

Since the external data seems to hurt the performance, we try to use the in-domain pairs to enhance task B and task C. For task B, if relative question 1 (rel1) and relative question 2 (rel2) are both relevant to the original question, then we add a positive sample (rel1, rel2, 1). If either rel1 and rel2 is irrelevant and the other is relevant, we add a negative sample (rel1, rel2, 0). After doing this, the samples of task B increase from $2,670$ to $11,810$. By applying this method, the MAP score increased slightly from 0.5723 to 0.5789 but the F1 score improved from 0.4334 to 0.5860.

For task C, we used task A's data directly. The results are very similar with a slight improvement on MAP, but large improvement on F1 score from 0.1449 to 0.2064.

4.5 Augmented features

To further enhance the system, we incorporate a one hot vector of the original IR ranking as an additional feature into the FNN classifier. Table 3 shows the results. In comparing the models with and without augmented features, we can see large improvement for task B and C. The F1 score for task A degrades slightly but MAP improves. This might be because task A already had a substantial amount of training data.

4.6 Comparison with Other Systems

Table 4 gives the final comparison between different models (we only list the MAP score because it is the official score for the challenge). Since the two baseline models did not use any additional data, in this table our system was also restricted to the provided training data. For task A, we can see that if there is enough training data our single system already performs better than a very strong feature-rich based system. For task B, since only limited training data is given, both feature-rich based system and our system are worse than the IR system. For task C, our system also got comparable results with the feature-rich based system. If we do a simple system combination (average the rank score) between our system and the IR system, the combined system will

[2] http://fb.ai/babi.

	Task A		Task B		Task C	
Model	MAP	F1	MAP	F1	MAP	F1
w/ Attention	0.6239	**0.6323**	0.5723	0.4334	0.2837	0.1449
w/ Attention + aug features	**0.6385**	0.6218	**0.6585**	**0.5382**	**0.3236**	**0.1963**

Table 3: cQA task results with augmented features (bold is best).

give large gains on tasks B and C[3]. This implies that our system is complimentary with the IR system.

	Task A	Task B	Task C
Model	MAP	MAP	MAP
IR	0.538	0.714	0.307
Attention	0.639	0.659	0.324
Feature-Rich & IR	0.632	0.685	0.339
Attention & IR	**0.639**	**0.717**	**0.394**

Table 4: Compared with other systems (bold is best).

5 Analysis of Attention Mechanism

In addition to quantitative analysis, it is natural to qualitatively evaluate the performance of the attention mechanism by visualizing the weight distribution of each instance. We randomly picked several instances from the test set in task A, for which the sentence lengths are more moderate for demonstration. These examples are shown in Figure 5, and categorized into short, long, and noisy sentences for discussion. A darker blue patch refers to a larger weight relative to other words in the same sentence.

5.1 Short Sentences

Figure 5a illustrates two cQA examples whose questions are relatively short. The comments corresponding to these questions are *"...snorkeling two days ago off the coast of dukhan..."* and *"the doha international airport..."*. We can observe that our model successfully learns to focus on the most representative part of the question pertaining to classifying the relationship, which is *"place for snorkeling"* for the first example and *"place can ... visited in qatar"* for the second example.

[3]The feature-rich based system was already combined with the IR system)

5.2 Long Sentences

In Figure 5b, we investigate two examples with longer questions, which both contain 63 words. Interestingly, the distribution of weights does not become more uniform; the model still focuses attention on a small number of hot words, for example, *"puppy dog for ... mall"* and *"hectic driving in doha ... car insurance ... quite costly"*. Additionally, some words that appear frequently but carry little information for classification are assigned very small weights, such as *I/we/my*, *is/am*, *like*, and *to*.

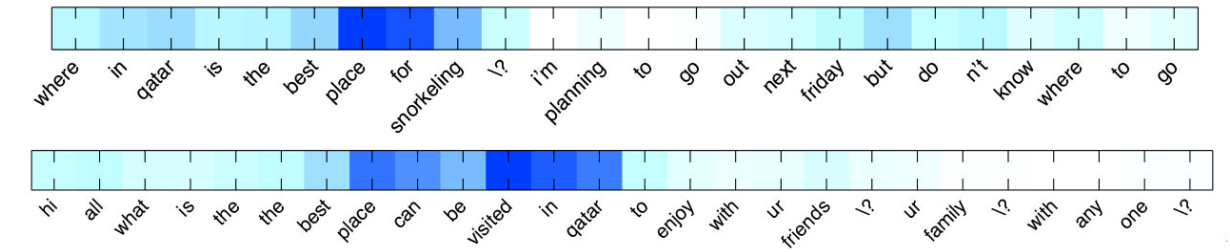
5.3 Noisy Sentence

Due to the open nature of cQA forums, some content is noisy. Figure 5c is an example with excessive usage of question marks. Again, our model exhibits its robustness by allocating very low weights to the noise symbols and therefore excludes the noninformative content.

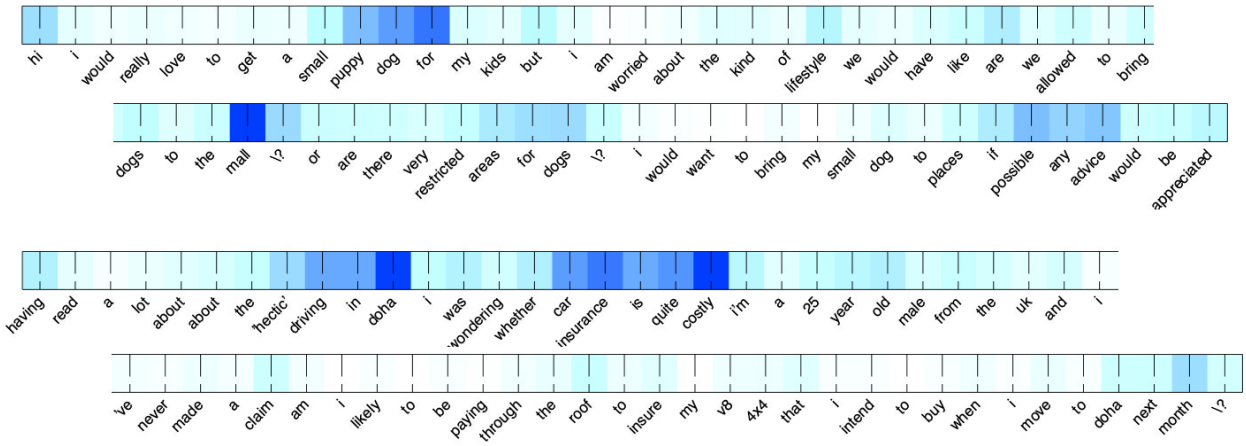
6 Conclusion

In this paper, we demonstrate that a general RNN encoder framework can be applied to community question answering tasks. By adding a neural attention mechanism, we showed quantitatively and qualitatively that attention can improve the RNN encoder framework. To deal with a more realistic scenario, we expanded the framework to incorporate metadata as augmented inputs to a FNN classifier, and pretrained models on larger datasets, increasing both stability and performance. Our model is consistently better than or comparable to a strong feature-rich baseline system, and is superior to an IR-based system when there is a reasonable amount of training data.

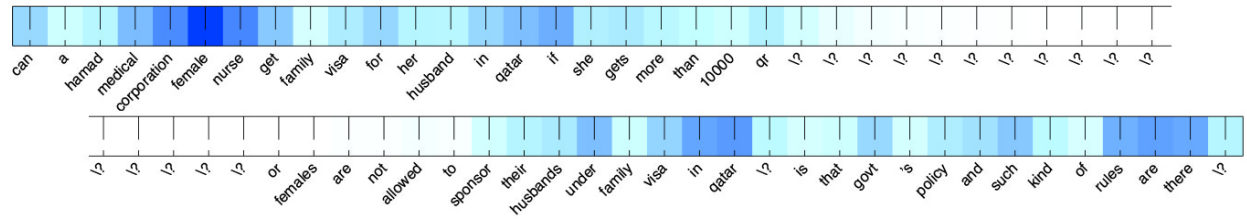
Our model is complimentary with an IR-based system that uses vast amounts of external resources but trained for general purposes. By combining the two systems, it exceeds the feature-rich and IR-based system in all three tasks.



(a) short sentences



(b) long sentences



(c) noisy sentence

Figure 5: Visualization of attention mechanism on short, long, and noisy sentences.

Moreover, our approach is also language independent. We have also performed preliminary experiments on the Arabic portion of the SemEval-2016 cQA task. The results are competitive with a hand-tuned strong baseline from SemEval-2015.

Future work could proceed in two directions: first, we can enrich the existing system by incorporating available metadata and preprocessing data with morphological normalization and out-of-vocabulary mappings; second, we can reinforce our model by carrying out word-by-word and history-aware attention mechanisms instead of attending only when reading the last word.

References

[Bahdanau et al.2014] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

[Bastien et al.2012] Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, James Bergstra, Ian J. Goodfellow, Arnaud Bergeron, Nicolas Bouchard, and Yoshua Ben-

gio. 2012. Theano: new features and speed improvements. Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop.

[Belinkov et al.2015] Yonatan Belinkov, Mitra Mohtarami, Scott Cyphers, and James Glass. 2015. Vectorslu: A continuous word vector approach to answer selection in community question answering systems. In *Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval*, volume 15.

[Berant et al.2013] J. Berant, A. Chou, R. Frostig, and P. Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *EMNLP*.

[Bergstra et al.2010] James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. 2010. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, June. Oral Presentation.

[Bowman et al.2015] S. Bowman, G. Angeli, C. Potts, and C. Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*.

[Cho et al.2014] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

[Feng et al.2015] Minwei Feng, Bing Xiang, Michael R Glass, Lidan Wang, and Bowen Zhou. 2015. Applying deep learning to answer selection: A study and an open task. *arXiv preprint arXiv:1508.01585*.

[Greff et al.2015] Klaus Greff, Rupesh Kumar Srivastava, Jan Koutník, Bas R. Steunebrink, and Jürgen Schmidhuber. 2015. LSTM: A search space odyssey. *CoRR*, abs/1503.04069.

[Grundström and Nugues2014] Jakob Grundström and Pierre Nugues. 2014. Using syntactic features in answer reranking. In *AAAI 2014 Workshop on Cognitive Computing for Augmented Human Intelligence*, pages 13–19.

[Harabagiu and Hickl2006] Sanda Harabagiu and Andrew Hickl. 2006. Methods for using textual entailment in open-domain question answering. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 905–912. Association for Computational Linguistics.

[Jeon et al.2006] Jiwoon Jeon, W Bruce Croft, Joon Ho Lee, and Soyeon Park. 2006. A framework to predict the quality of answers with non-textual features. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 228–235. ACM.

[Nakov et al.2015] R. Nakov, L. Marquez, W. Magdy, A. Moschitti, and J. Glass. 2015. Semeval-2015 task 3: Answer selection in community question answering. In *Proc. SamEval*, pages 282–287.

[Rocktäschel et al.2015] Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, and Phil Blunsom. 2015. Reasoning about entailment with neural attention. *arXiv preprint arXiv:1509.06664*.

[Shah and Pomerantz2010] Chirag Shah and Jefferey Pomerantz. 2010. Evaluating and predicting answer quality in community qa. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 411–418. ACM.

[Sutskever et al.2014] Ilya Sutskever, Oriol Vinyals, and Quoc VV Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

[Tran et al.2015] Quan Hung Tran, Vu Tran, Tu Vu, Minh Le Nguyen, and Son Bao Pham. 2015. Jaist: Combining multiple features for answer selection in community question answering. In *Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval*, volume 15.

[Wang and Nyberg2015] Di Wang and Eric Nyberg. 2015. A long short-term memory model for answer sentence selection in question answering. In *ACL*.

[Zhai and Lafferty2004] C. Zhai and J. Lafferty. 2004. A study of smoothing methods for language models applied to information retrieval. In *ACM Trans. Inf. Syst*.