Paraphrase Question Identification using Feature Fusion Network

Po-Cheng Pan and **Heng-Lu Chang**

1 Introduction

Quora is an on-line question-and-answer site which has grown significant popularities. One of the challenges is that more and more questions were posted with the same intent, causing people to spend more time finding the best answer to their question, and also making writers feel that they have to answer multiple versions of the same question. Our goal is to identify whether two different questions are having the same intent or not.

In this paper, we proposed a novel model coined as "Feature Fusion Network (FFN)" to improve the accuracy of identifying paraphrase questions. FFN fuses sentence embeddings and hand crafted features (HCFs) as input to the Deep Learning (DL) model to learn similarity between two questions. Our results showed 89% testing accuracy on Quora dataset, which is better than previous models that use purely DL methods. The proposed model takes advantage of learning rich features not just from sentence representations but also from HCFs. One of the reasons why FFN out-performs other DL models is because some HCFs are hard to learned through DL-only models. We also conducted an ablation study to show that our FFN without HCFs performs not as good as FFN.

2 Problem Definition

The problem falls within the category of paraphrase question identification (PQI), which can be defined as "the task of deciding whether two given questions have the same meaning". This can be seen as a binary classification problem where the input is a pair of questions and the output is either 0, meaning two questions are different from each other, or 1, where two questions have the same meaning. The dataset[1] we used were released by Quora which contains approximately 400 thousand training question pairs and 2 million testing question pairs. Table 1 showed samples of Quora dataset. PQI is an interesting problem in particular because nowadays there are too many similar questions being asked on Quora, causing similar question being answered at multiple places and making Quora users hard to find high quality answer. An algorithm which can identify paraphrase questions can redirect Quora users to the canonical question and save users lots of time.

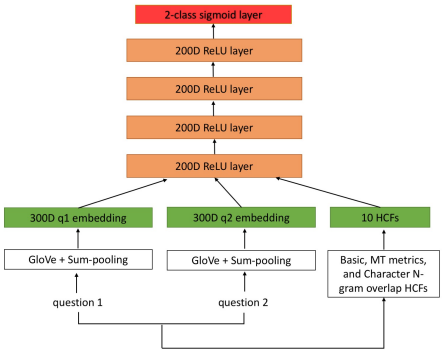

Figure 1: The Feature Fusion Network architecture: the input to this model are vector representations of both questions along with 10 hand crafted features (HCFs), four Regularized Linear Unit (ReLU) layer and the final softmax layer is our Neural Network model to identify paraphrase questions.

[1] https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs

question 1	question 2	is duplicate
How can I be a good geologist?	What should I do to be a great geologist?	1
Why do girls want to be friends with the guy they reject?	How do guys feel after rejecting a girl?	0
Why is creativity important?	Why creativity is important?	1
What are natural numbers?	What is a least natural number?	0

Table 1: Samples of Quora dataset

3 Feature Fusion Network (FFN)

Figure 1 shows the architecture of the FFN model proposed. The input to FFN is concatenation of two 300 dimensional question vector representations and 10 hand crafted features(HCFs). The output is a score where 1 indicates the input are paraphrase questions and 0 means they are not. Below we will discuss in details how we extract features and our Neural Network classifier model.

3.1 Question Sentence Embedding

We first use a pre-trained GloVe (Pennington et al., 2014) (trained on GloVe Common Crawl) to map each word into a 300 dimensional vector representation. We also limit the size of question to 25 words. The choice of using pre-train GloVe in stead of learning our own word embeddings is due to the consideration of generalization purpose. That is to say, we want to avoid overfitting the word vector representations in the training data.

For sentence vector representation, we exploit sum-pooling, which is to sum up all the word vectors in the sentence. Max and mean pooling for sentence embedding were also considered, but the results in terms of testing accuracy are not as good as sum-pooling. Therefore, each question are represented by a 300 dimensional vector after the sentence embedding algorithm (GloVe word embedding and sum-pooling). We then concatanate the vector representation of our question pair and ten HCFs to form a 610 dimensions of feature representation for every question pair.

3.2 Hand Crafted Features (HCF)

The raw Quora dataset as seen in Table 1 has only two features, which is the question pair. We engineered ten HCFs, which can be categorized into three sets: standard, machine translation evaluation metrics, and character n-gram overlap. In the following subsection we will discuss why we choose these three sets of HCF in details.

3.2.1 Standard Features

Standard hand crafted features include word count of question 1, word count of question 2, difference in word count, and number of word overlap between 2 questions. Standard features are well discussed in the Paraphrase identification tasks. The idea is that the smaller the word count difference and the more the common words between two sentences results in higher probability of being paraphrase sentence pair.

3.2.2 Machine Translation Metrics

Machine translation (MT) metrics is used to identify semantic similarity between two sentences(Finch et al., 2005) if we consider single reference sentence as question 1 and candidate sentence as question 2. We engineered three features related to MT metrics: unigram-BLEU, bigram-BLEU, and BLEU2. BLEU stands for bilingual evaluation understudy(Papineni et al., 2002) which is one of the most popular automatic evaluation MT metrics. Unigram-BLEU are calculated using word unigram precision between two questions, and similarly, bigram-BLEU are calculated using word bigram precision between two questions. BLEU2 is the geometric average of unigram-BLEU and bigram-BLEU. Notice that we used add-1 smoothing(Lin and Och, 2004) for bigram-BLEU and BLEU2 to avoid harsh penalty if the word bigram precision equals to zero.

3.2.3 Character N-gram Features

We engineered overlap character bigram, trigram, and quadgram counts for both questions as features. These features are inspired by a paper(Eyecioglu

Model	Source of Word Embeddings	Accuracy
FFN	GloVe Common Crawl (840B tokens, 300D)	0.895
BiMPM model	GloVe Common Crawl (840B tokens, 300D)	0.881
LSTM with concatenation	Quora's text corpus	0.870
LSTM with distance and angle	Quora's text corpus	0.860
Decomposable attention	Quora's text corpus	0.860
L.D.C.	GloVe Common Crawl (840B tokens, 300D)	0.855
Multi-Perspective-LSTM	GloVe Common Crawl (840B tokens, 300D)	0.832
Siamese-LSTM	GloVe Common Crawl (840B tokens, 300D)	0.826
Random Forest	None	0.825

Table 2: Performance for paraphrase identification on Quara dataset

and Keller, 2015) where the authors achieved highest F-score when participating in the SemEval 2015-task1 for Twitter PI task. Overlap character N-gram can capture semantic information such as stemming. For example, "break" and "breaks" are lexically the same and they also have large overlapping character N-gram.

3.3 Neural Network Classifier

The concatenated question embeddings (600 dimensions) and HCFs (10 dimensions) are chosen to served as input (610 dimensions) to our Neural Network classifier. We referenced Github repository "keras-quora-question pairs[2]", which is a Keras implementation of the paper (Bowman et al., 2015) as our Neural Network classifier. The model consists of four fully-connected ReLU layer with 200 hidden units in each layer followed by a final sigmoid classification layer. We add a drop-out rate of 0.2 between layers to regularize our model. Drop-out has been shown to be useful in regularizing Neural Network models (Srivastava et al., 2014).

3.4 Training

Our FFN is trained by minimizing the cross-entropy loss. We use stochastic gradient descent to train our FFN model.

[2] https://github.com/bradleypallen/keras-quora-question-pairs

4 Experimental Evaluation

4.1 Dataset Preprocessing

4.1.1 Quora Dataset

The dataset we used was released by Quora which contains approximately 400 thousand training question pairs and 2 million testing question pairs. However, the labels of testing question pairs are hidden by Kaggle. To evaluate our model, we partition the Quora training question pairs into a $90/10$ train/test split. Then we run training with a further $90/10$ train/validation split.

4.1.2 Down Sampling Positive Label

By examining the training data carefully, the positive label take around 35% of the training data. However, we found out that there are only 17.5% of positive labels in Kaggle Public Leader board based on a kernel[3] on Kaggle. As a result, we did a down sampling on the positive label in the training data to ensure the balancing between training and testing set on Kaggle. Down sampling of minority labels is undesired because this losses lot of information. However, we still decided to do down sampling of positive labels because we assume that the Kaggle Private Leader board also has 17.5% of positive labels.

4.2 Evaluation Metrics

We used two evaluation metrics: One is accuracy and the other is cross-entropy loss. Accuracy is the most popular evaluation metric for PI tasks while cross-entropy loss is the evaluation metric on Kaggle

[3] https://www.kaggle.com/c/quora-question-pairs/leaderboard

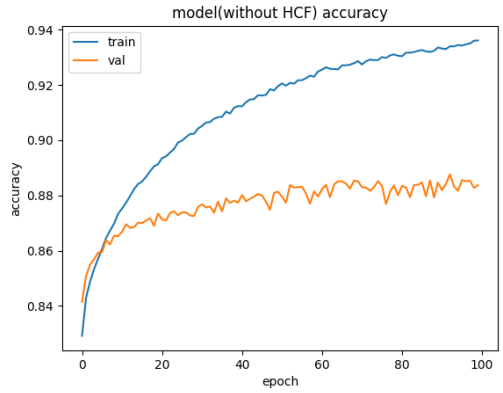

Figure 2: The accuracy of training and validation data of our FFN model without HCF.

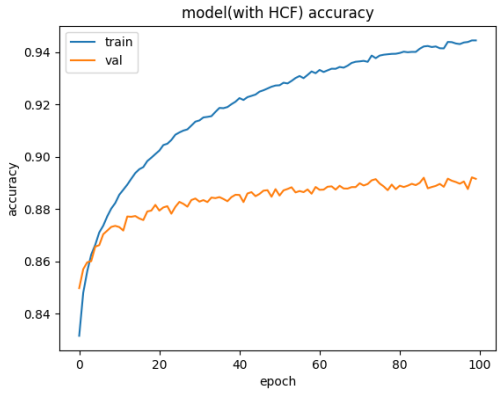

Figure 3: The accuracy of training and validation data of our FFN model.

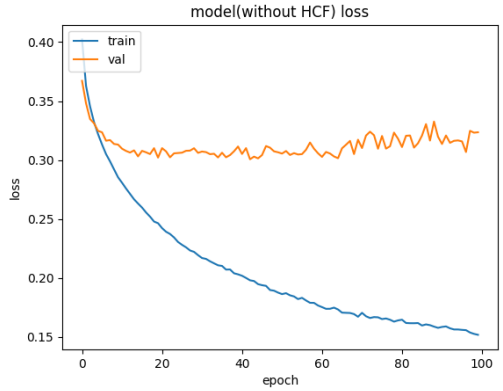

Figure 4: The cross-entropy loss of training and validation data of our FFN model without HCF.

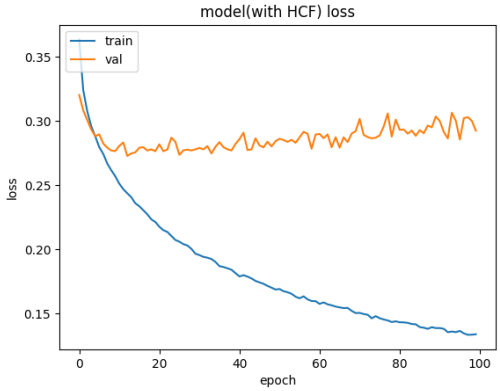

Figure 5: The cross-entropy loss of training and validation data of our FFN model

competition. Cross entropy can be calculated using the equation below, where y is the true label and $\hat{y}$ is the predicted label:

$$H(y, \hat{y}) = -y \log \hat{y} - (1 - y) \log(1 - \hat{y})$$

4.3 Experiment Results

Our results is based on testing on the 10% of the training data. We did not test FFN on the 2 million testing data because we do not have the true label of those testing data.

4.3.1 Comparison with Baselines

Table 2 presents the performances of all baseline models and the proposed "FFN" model. We can see that Multi-Perspective-LSTM and Siamese-LSTM achieve 83% accuracy (Wang et al., 2017). However, the three models: (1) LSTM with concatenation, (2) LSTM with distance and angle, and (3) Decomposable attention proposed by Quora (Lili Jiang and Dandekar, 2017) and the L.D.C method (Wang et al., 2016) can achieve $86 - 87\%$. Moreover, the best baseline is the BiMPM model (Wang et al., 2017) which outperforms other baselines by more than one percent. Finally, our FFN model can reach 89.5% accuracy, which is better than the all other baseline that use DL-only method. Therefore, our model is effective for the desired task.

4.3.2 Ablation Study

In our FFN model, we've built 10 HCF features to enhance the performance. To see the impact of HCF, we have run our FNN model without HCF for 100 epochs.

As shown in Figure 2, we can see that the training and validation accuracy of our FFN model without HCF increase as the number of epochs increases. As for FFN model, we can find similar behavior from Figure 3. However, the accuracy of the proposed

FFN model is 1% higher than FNN model without HCF in both training and validation set.

On the other hand, for both FFN and FFN without HCFs models we can see that the training cross-entropy loss decreases as the number of epochs increases. However, the validation loss decreases in the first 30 epochs and then slightly increases in the last 70 epochs. In addition, the loss of FFN model is $0.02 - 0.03$ lower than FFN model without HCF.

Finally, we use FFN on the testing data to evaluate the actual performance. From Figure 4 and 5, the loss starts to increase from around $30th$ epoch. Hence, we record the testing accuracy and loss of both models at $30th$ and $100th$ epoch in Table 3. FFN has better accuracy and loss than FFN without HCF. For $30th$ epoch, the accuracy of FFN is 1.1% higher and the loss of FNN is 0.031 lower than FFN without HCF. As for 100 epochs, the accuracy of FFN is 1% higher and the loss 0.02 lower than FFN without HCF. However, because the validation loss of FFN starts to decrease from $30th$ epoch, the loss derived from FFN trained for 30 epochs is 0.279, which is much lower than the loss derived from FFN trained for 100 epochs.

4.4 Discussion

From our experimental results, our model can achieve better accuracy than other baseline models. Even without HCF, our model still can reach almost 88% accuracy which is comparable to others.

On the other hand, we can find that although the validation accuracy decreases as the training epoch increases, the loss starts to decrease from the 30th epoch. It's reasonable because when we train our model for many epochs, it starts to be overfitted.

Moreover, we can see that adding HCF would improve the performance of our model because the HCFs contain some information which can't be known by using the GloVe as word embeddings to generate the question embeddings like machine translation metrics and character N-gram features as mentioned in Section 3.

5 Related Work

There are several tasks related to identifying semantically equivalent questions. In the following paragraph, we outline the difference between these tasks and their methods.

model	accuracy	loss
FFN (30 epochs)	0.879	0.279
FFN without HCF (30 epochs)	0.868	0.310
FFN (100 epochs)	0.894	0.291
FFN without HCF (100 epochs)	0.884	0.311

Table 3: Accuracy and loss of testing data

One of the well-known model for solving this problem is the random forest model with tens of handcrafted features, including the cosine similarity of the average of the $word2vec$ embeddings of tokens, the number of common words, the number of common topics labeled on the questions and the part-of-speech tags of the words.

Inspired by recent advances in the deep learning search community, there are also many end-to-end learning solutions to the duplicate detection problem. The following models are quoted from an article of Quora(Lili Jiang and Dandekar, 2017).

First of all, a deep architecture that used the Long Short Term Memory network(LSTM), variant of Recurrent Neural Networks(RNNs), which is better in capturing long-term dependencies. It trained its own word embeddings using Quora's text corpus to generate question embeddings for the two questions. Then fed those question embeddings into a representation layer.

Second, a similar model to the LSTM one but uses two empirically motivated handcrafted features (Tai et al., 2015): (1) the distance, calculated as the sum of squares of the difference of the two vectors and (2) the angle, calculated as an element-wise multiplication of the two vector representations.

Third, an attention-based approach from Google (Parikh et al., 2016) combines neural network attention with token alignment, commonly used in machine translation. Similar to the previous two approaches, this model represents each token from the question with a word embedding.

These three approaches achieve around 88% accuracy on Quora's test data. On the other hand, our model achieves almost 89% accuracy, which is slightly better than these approaches.

Additionally, HCFs fused with sentence embeddings was also exploited in a paper (Suggu et al., 2016) which dealt with Answer-Question Prediction

(AQP) tasks.

6 Future Work

Although our FFN model can achieve high accuracy and low loss on the testing data we built, the performance of our proposed model behaves mediocre in the Kaggle competition. Two reasons are suspected. First, in the dataset given by Quora, there are 400 thousand question pairs in training data and two million question pairs in testing data. The amount of the testing data is significantly greater than the training data. Second, the training and testing data may be sampling from different populations. In order words, the training data may not possess the same property as randomly sampled ones from the original data set.

To deal with this, because the testing data is unlabeled, one promising method is known as the semi-supervised learning which is expect to make use of unlabeled testing data for training.

Moreover, there are many different models related with the task. On way to take advantage of the fact is to ensemble the methods together to obtain a better predictive performance which could not be obtained from any of the constituent learning algorithm alone.

Our final goal is building a system for retrieval of semantically equivalent questions. To be specific, given a corpus and a question, the task is to find all questions in the corpus that are semantically equivalent to the given question.

7 Conclusion

In this paper, we propose a method for identifying semantically equivalent questions, coined as "Feature Fusion Network (FFN)". Our model combines sentences embedding and the HCFs as input to a Deep Learning model to judge whether the question pair is similar or not. Our results showed 89% testing accuracy which is better than all other Deep Learning models.

References

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*.

Asli Eyecioglu and Bill Keller. 2015. Asobek: Twitter paraphrase identification with simple overlap features and svms. *Proceedings of SemEval*.

Andrew Finch, Young-Sook Hwang, and Eiichiro Sumita. 2005. Using machine translation evaluation techniques to determine sentence-level semantic equivalence. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, pages 17–24.

Shuo Chang Lili Jiang and Nikhil Dandekar. 2017. Engineering at quora- semantic question matching with deep learning. In *Quora*.

Chin-Yew Lin and Franz Josef Och. 2004. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 605. Association for Computational Linguistics.

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*, pages 311–318. Association for Computational Linguistics.

Ankur P Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. *arXiv preprint arXiv:1606.01933*.

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

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

Sai Praneeth Suggu, Kushwanth N Goutham, Manoj K Chinnakotla, and Manish Shrivastava. 2016. Deep feature fusion network for answer quality prediction in community question answering. *arXiv preprint arXiv:1606.07103*.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075*.

Zhiguo Wang, Haitao Mi, and Abraham Ittycheriah. 2016. Semi-supervised clustering for short text via deep representation learning. *arXiv preprint arXiv:1602.06797*.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral multi-perspective matching for natural language sentences. *arXiv preprint arXiv:1702.03814*.