Enhance Deep Learning with Hand-Crafted Features for Quora Duplicate Question Detection

Dongguang You
dgynick@utexas.edu

Peng Su
pengsu@utexas.edu

Abstract

In this project, we tackled with the problem of duplicate question detection using the data provided by Quora[1]. We experimented with a variety of features, including Recurrent Neural Network(RNN) Sentence encoding, Convolutional Neural Network(CNN) sentence encoding, and a set of hand-crafted features. Our experiments show that the combination of Gated Recurrent Unit(GRU) encoding and our hand-crafted features achieves the best performance, with a test accuracy of 86.9%, and outperforms the 85.2% from the best baseline[2] on this dataset we have found.

1 Introduction

In the past decade, question-and-answer websites such as Quora have become a crucial part of our lives. Millions of people visit these sites every month, either to seek for answers for questions that require knowledge of a specific area, or to apply their expertise to provide answers. The huge number of users pool all sorts of knowledge together and makes it possible to share answers for various problems. Unfortunately, this has also led to difficulties in managing these sites. One difficulty, in particular, is that people ask the same question in different ways, and the answers for the same question can be scattered under different posts if the question database is not carefully managed. This will create tremendous trouble for answer-seekers, as they will have to look through many different posts about the same question in order to find a desirable answer. Given the sheer size of the question database, it is impractical for those duplicate questions to be merged manually. Hence it becomes crucial that the engineers in the backend are able to implement a duplicate detection mechanism that is both accurate and efficient.

In this project, we aimed to solve the problem of duplicate question detection. In other words, give a question pair, our model will be able to tell whether the two questions are semantically equivalent or not. Our major efforts consist in extracting the best features from a question pair, and we took three different approaches. First, inspired by "skip thoughts"(Kiros et al., 2015), we trained a Bidirectional Recurrent Neural Network with GRU cell to extract the sentence encodings and make predictions. Secondly, inspired by (Bogdanova et al., 2015), we trained a Convolutional Neural Network to do the same task. Thirdly, we extracted hand-crafted features that measure the question characteristics and differences. In the end, we evaluated our model based on the cross entropy loss on the test data, and the combination of our hand-crafted features and RNN yields the lowest loss and highest accuracy. Each approach will be explained in great details in the following section.

2 Implementation

2.1 Problem Definition

In this project, we aimed to solve the problem of duplicate question detection. More specifically, given a question pair, our model outputs a probability that

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

[2] https://web.stanford.edu/class/cs224n/reports/2748045.pdf

Table 1: Example duplicate pairs from training data

What should I do to be a great geologist?	How can I be a good geologist?
I can't remember my Gmail password or my recovery email. How can I recover my e-mail?	I was suddenly logged off Gmail. I can't remember my Gmail password and just realized the recovery email is no longer alive. What can I do?
How will a Trump presidency affect the students presently in US or planning to study in US?	What would a Trump presidency mean for current international master's students on an F1 visa?

Table 2: Example non-duplicate pairs from training data

When do you use シ instead of し?	When do you use "&" instead of "and"?
What is the step by step guide to invest in share market in india?	What is the step by step guide to invest in share market?
What universities does Rexnord recruit new grads from? What majors are they looking for?	What universities does B&G Foods recruit new grads from? What majors are they looking for?

the two questions are semantically equivalent. Table 1 and Table 2 present some examples from our labeled training data. The problem is much more challenging and interesting than it appears at first glance. For instance, the similarity/difference between two questions can be very fine-grained and sometimes hard to capture, as illustrated by the third row of Table 1 and the second row of Table 2. The second row of Table 1 indicates that solving this problem also requires understanding the intention of the question, which falls into the scope of Pragmatics. In addition, in some cases part of the question may be of a language other than English, such as the first row of Table 2. In this section, we explains all three approaches that we have implemented in this project.

2.2 RNN with GRU cell

As hinted by (Kiros et al., 2015), RNN trained to predict neighboring sentences is able to extract information that are very useful for the tasks of semantic relatedness classification and paraphrase detection. We finetune their skip-thoughts RNN encoder for duplicate question detection. Our network structure consists of a RNN sentence encoder, followed by two fully connected layers on the top.(See Figure 1).

Our RNN sentence encoder adopts the same structure as the encoder in (Kiros et al., 2015), which is a bidirectional RNN and each cell has hidden state size 1200(see Figure 2). In addition, our encoder's weights were initialized from (Kiros et al., 2015)'s pretrained model. The effect of such initialization was significant, which will be discussed later in this paper.

To add more non-linearity to our fully connected layers, we computed the element-wise product and square difference of the two encoding vectors, and concatenated them with the original encodings. The first fully connected layer is followed by a Rectified Linear Unit layer. Furthermore, we added L2 regularization to the weights in the fully connected layers to mitigate over-fitting.

2.3 CNN

(Kim, 2014) and (Bogdanova et al., 2015) have implemented CNN for sentence classification and question similarity detection from online user forums. Based on (Bogdanova et al., 2015), we modified the CNN architecture for our problem. As shown in 2.3, the CNN applies convolution on the word embeddings, followed by a max-pooling, and two fully-connected layer with Relu in the hidden layer and softmax activation function for classification. We adopted a few suggested configurations based on (Zhang and Wallace, 2015), which tested varies settings of CNN on sentence classification tasks. The word embeddings are initialized based on GloVe 300D embedding. Effect of doing such initializations is significant. As with our RNN architecture, we added element-wise product and squared difference of the two encoded vectors, generating surprisingly good results.

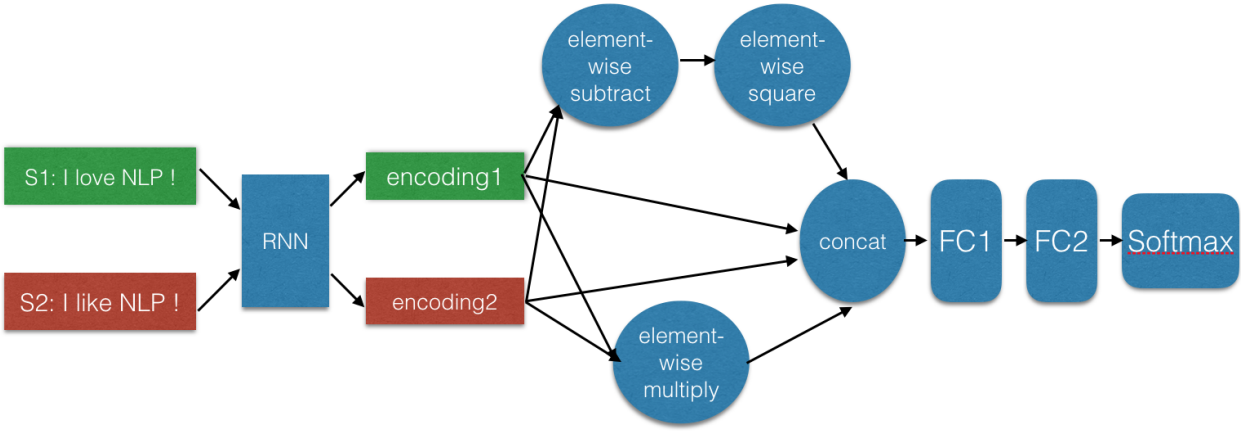

Figure 1: Our RNN network Architecture. A pair of sentences are encoded by the RNN separately. The encoding, concat, FC1 and FC2 have dimension 2400, 9600, 512 and 2 respectively.

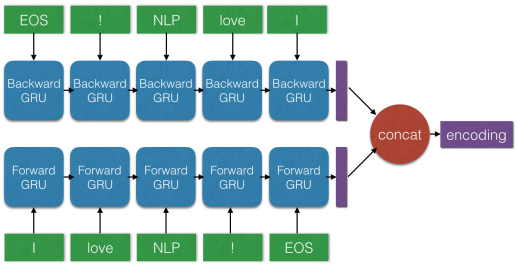

Figure 2: Our RNN encoder architecture. Each word is converted to an embedding of dimension 620. Each GRU cell has hidden state of dimension 1200, and the final output has dimension 2400.

2.4 Feature Engineering

Features that can capture the sentence characteristics and differences are extracted. These features alone are used to train a Gradient Boosting Decision Tree model, and also concatenated to the RNN encoder outputs to train a SVM model. RNN is based on NLTK tokenized data, so the hand-crafted features did not use NLTK tokenizer, but our own preprocessor. We have features denoting different kinds of W questions (What, Where, etc), H questions (How), and general questions (Is, Are, Can, etc). Some of our other features are related to question length, character length, number of words, number of common words, and question length difference. We also intended to add Term Frequency–Inverse Document Frequency(TFIDF) features. However, dimension reduction cannot be done very effectively in our situation because the questions are mostly short and more than 500 principal components are needed to capture 70% variance. Since questions that appear more often are more likely to have duplicates, we also included the number of occurrences of a question in the training and test dataset as two features. By GBDT feature importance (see Table 3), question frequencies are the two most important features. A Kaggle user's LinkedIn blog provided us some insights. [3] The GloVe 300d word embeddings are used as additional features, as well as multiple similarity/distance metrics based on the sentence vectors, including cosine, Manhattan, and Euclidean, etc.

3 Experiments

3.1 Dataset

In this project, we are provided with 404290 pair of labeled questions from Quora, with 63.08% labeled non-duplicate and 36.9% labeled duplicate. According to Quora's announcement[4], the negative examples, i.e. non-duplicate pairs, were augmented manually in certain ways, and some sanitization measures have been applied to the final dataset. Hence this dataset may not conform to the real world distribution of duplicate questions. As have

[3] https://www.linkedin.com/pulse/duplicate-quora-question-abhishek-thakur

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

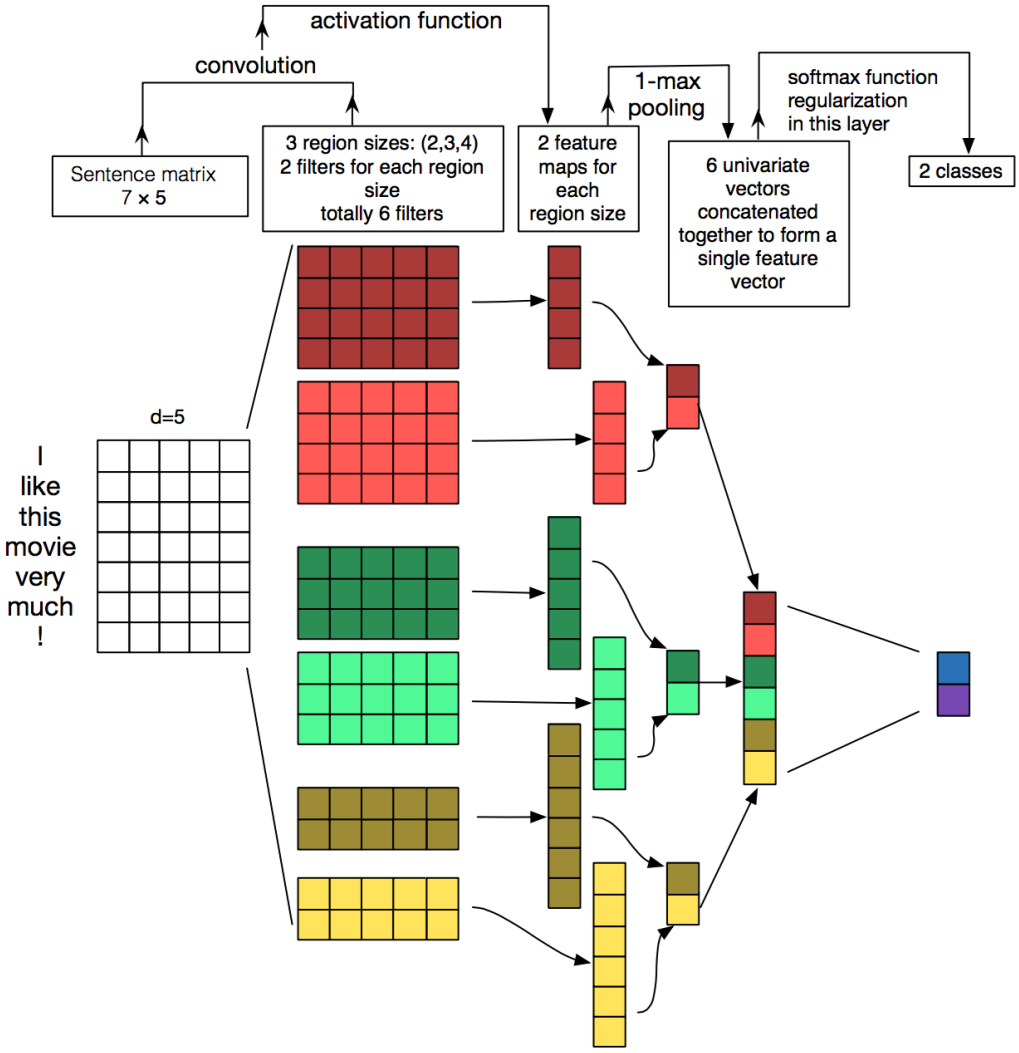

Figure 3: CNN Architecture

been shown before, the division between duplicates and non-duplicates is very fine-grained here.

The total token count is around 10 Million, and the training data vocabulary size is between 80K and 100K, depending on the tokenzier being used. We preprocessed the training and testing data using the Python NLTK package. The effect of this preprocessing is significant. With a crude version of preprocessing, a vocabulary of size 80000 cannot recognize 10% of all the tokens. With the NLTK preprocessing, a vocabulary of the same size only misses 0.9% of all the tokens. Figure 4 shows that the non-duplicate question pairs may also have a lot of overlapping words, making this problem very challenging.

3.2 Methodology

We evaluate our methods mainly based on test cross entropy loss, as opposed to based on accuracy. Accuracy only reflects the performance of the model when 0.5 is chosen as the decision threshold. However, the costs of false positives and false negatives are usually different in practice, and the cross en-

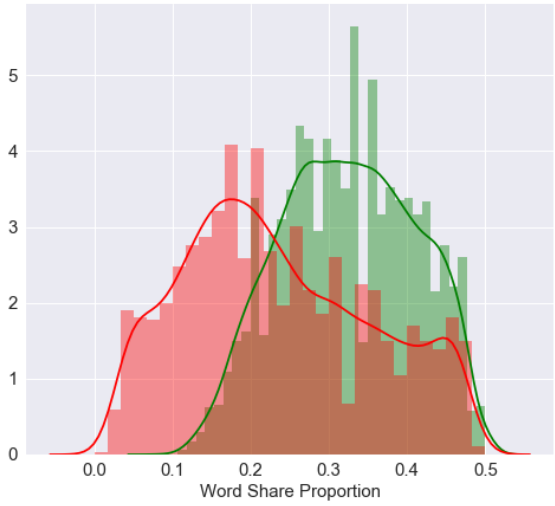

Figure 4: Distribution of word-share proportion of duplicate (green) and non-duplicate (red) question pairs. There are a lot of question pairs with high shared words that are both duplicate and non-duplicate question pairs.

tropy can better reflect the overall precision of the model across different values of recall. In the end, we still present the accuracy and Precision over Recall Curve of our models. Also, we will show qualitative examples to illustrate what information the RNN has learned to extract.

We hypothesize that RNN would produce the best results because of the advantage of pretraining, and CNN would yield competitive results when the word embedding are initialized from GloVe [5]. We suspect that hand-crafted features may not perform so well as the neural networks models, but can complement their performance if combined properly.

3.3 Results

3.3.1 RNN

Our basic workflow is as follows: In the first stage, we trained our aforementioned RNN architecture with 75% of the training data(shuffled randomly), among which 5000 are used as validation data. In the second stage, we used the rest training data to fit different classifiers on top of the fully connected layers from the neural network, among which 5000 are used as validation data and another 5000 are used as test data.(This test data is used for all the test evaluations in our project)

Specifically, for the first stage, we experimented with multiple configurations of the RNN sentence encoder during training because of a few technical concerns. First of all, is the effect of pretraining beneficial? Secondly, is it necessary to also finetune the word embedding weights in addition to the parameters inside GRU? Thirdly, how does the extra non-linearity affect the performance? Lastly, would it improve the performance if more fully connected layers are added to make the network deeper?

Hence we experimented with five slightly different models. During training, we applied the Adam gradient descent algorithm(Kingma and Ba, 2014). Our batch size is 128 and our vocabulary size is 20000. We started with a learning rate of 0.01 and multiply it with a factor of 0.25 after every epoch. The learning rate for GRU parameters is one-tenth of that for the fully connected layers. If the embedding layer is trained, its learning rate is one-tenth of that for the GRU parameters. The training results for the four models are shown in figure 5.

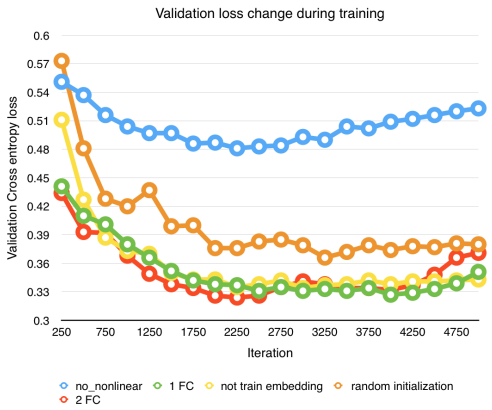

Figure 5: The validation loss of our five RNN variants during training. The 2FC architecture is the one presented in Figure 1. The 1FC baseline has FC1 layer removed from the network. Additional changes are made to the 1FC baseline to form the other three baselines.

As can be observed, initializing from skip thoughts' pretrained weights does help reduce the loss, and whether or not the embedding layer is fine-tuned makes little difference. Removing the non-linearity when there is only one fully connected layer leads to extremely bad performance, possibly

[5] https://nlp.stanford.edu/projects/glove/

because one fully connected layer alone can hardly simulate a non-linear relationship. Although making the network deeper makes minor difference in terms of the performance, we still prefer it because it allows us to reduce the dimension of the feature to be extracted from 9600 to 512. Therefore for the rest of this paper we only consider the 2FC architecture.

To qualitatively evaluate the feature our RNN model has learned, in Figure 6 we show the four nearest neightbors in the feature space of three seed questions, before and after our finetuning. In the first example, fine-tuning enhanced the model's capabilities of capturing key phrases such as "age limit" and sentence structures such as "32 or 64 bit". In the second example, the finetuned model appears to have a better understanding of the seed sentence, as it is able to emphasize on the semantic relation between "high pay packages" and "promoted". In the third example, both failed to find a semantically similar question, although the model before fine-tuning is able to find questions starting with "how".

As an interesting side-note, (Kiros et al., 2015) mentioned a trick named vocabulary expansion. During training, a small vocabulary size is used, which speeds up the process but causes the network's failure to recognize many words. After training, they find the overlap of their small vocabulary and word2vec's big vocabulary, and fits a linear regression from word2vec's embedding to their embedding. In the end, their small vocabulary is augmented with word2vec's vocabulary. Surprisingly, in our experiments, the performance consistently gets worse after vocabulary expansion, across all three types of classifiers. Hence we remain doubtful about the effect of this simple method.

3.3.2 CNN

The same 5000 samples are used as test dataset, and 90% of the remaining is used as training and 10% used as validation. (Zhang and Wallace, 2015) proposed a few configuration variations, and we experimented with two of them. The first is the effect of using pre-trained word embeddings VS training from scratch. For the pre-trained embeddings, we experimented the Google News word2vec and GloVe embeddings (100d and 300d). The second is whether to add an extra hidden fully connected layer with RELU activation. Our results show that using GloVe 300d pre-trained word embeddings and adding a Relu hidden layer achieve the best performance. The filter sizes we used are $(3, 5, 7)$ and each with 128 filters. The lowest validation loss we achieved with CNN is 0.342. On the test data, the loss is 0.336, and accuracy is 85.6. The precision-recall curve is shown in Figure 7.

Table 3: Feature Importance

Feature	Importance
Question 2 frequency	100
Question 1 frequency	94
Fuzzy Q Ratio	80
Fuzz token set ratio	53
Number of common words	46
Fuzz partial ratio	45
Fuzz token sort ratio	37
Minkowski distance	17
Braycurtis distance	14
Euclidean distance	11

3.3.3 Hand-crafted features

In total we have 643 hand-crafted features. We experimented with Random Forest and Gradient Boosting Decision Tree (GBDT). GBDT performed the best, reaching prediction accuracy 0.83 and loss 0.357 on the test set. The optimal max_depth of GBDT is 5. 10 of the most important features are listed in Table 3. The importance values are normalized so that the highest is 100. Frequencies of the two questions are two most important features. This makes sense as questions asked more often is more likely to be duplicates. There are a few fuzzy ratio related features, which use Levenshtein Distance to calculate the differences between sequences. For more information, please refer to this link [6]. None of the 600 features of the two embedded vectors have importance higher than 6. But the sum of these 600 features' importance is 360. The precision-recall curve of the test dataset with 40,429 question pairs is shown in Figure 7.

3.3.4 Feature combination

In this stage, we extracted the deep representation(features in the last hidden layer) of the rest

[6] https://github.com/seatgeek/fuzzywuzzy

	seed	1st neighbor	2nd neighbor	3rd neighbor	4th neighbor
before fine-tuning	what will be the max age limit for the ias exam in 2015 , 32 or 29?	what will be the effect on share market after the banning of 500 and 1000 notes ?	what will be the long term effect of removing 500 and 1000 currency notes ?	what will be the long term effect of removing 500 and 1000 currency notes ?	what will be the best isp in usa for hosting web server
after fine-tuning		what will be age limit for upsc in 2017 ?	what is the age limit to take the sats and acts ? is there an age limit to it or not ?	what is the processor of my pc , 32 or 64 bit ?	what would be age limit for upsc 2017 onwards ?
before fine-tuning	career advice : why are scientists not paid such high pay packages like engineers and it sector people ?	career advice : what is the reason many engineering students ca n't get jobs in india ?	career advice : what are the real reasons some people get promoted and others do n't ?	career advice : what are the success tricks for preparing gate in 3 months ?	career advice : how can i get a job at facebook or google in 6 months ?
after fine-tuning		career advice : what are the real reasons some people get promoted and others do n't ?	career advice : what is the reason many engineering students ca n't get jobs in india ?"	career advice : i am currently working as a automation tester in india mnc what new tool or technology is suitable for my career growth in long run ?	career advice : i have studied law in india . can i get a job in the usa ?
before fine-tuning	how can you get pregnant during miscarriage ?	how can you get pregnant when you are not in your period ?	how can you get rid of infected pimples ?	how can you get rid of infected pimples ?	how can you legally make spongebob gifs ?
after fine-tuning		can you only get pregnant during ovulation ?	can you only get pregnant during ovulation ?	can you only get pregnant during ovulation ?	can you get pregnant from pre-cum ?

Figure 6: This figure shows the nearest neighbors for three seed questions in the feature space of our RNN model, before and after fine-tuning. Note that the same question may appear more than once in our dataset, so some neighbors are exact duplicates.

training data using the previous RNN model, and trained three types of classifiers including Support Vector Machine(SVM), Random Forest(RF), and Gradient Boosting method(GB). GB generated the best performance. And unsurprisingly, this has produced our best results, a test loss of 0.292 and test accuracy of 86.9%.

3.4 Summary of results

A complete performance comparison of all four methods are shown in Figure 7. It's obvious that RNN+FE consistently outperforms the other three across different recalls. Our RNN model outperforms our CNN model for low recalls, while our CNN model achieves a better test accuracy.

3.5 Discussion

Our results support our previous hypothesis. Pre-training on skip-thoughts does give our RNN model a significant advantage. This is possibly because training neural network is essentially finding a good local minimum in a highly non-convex objective space, and a good initialization will make it much easier to find a local minimum that's over-fitting less.

Combining our RNN feature with our handcrafted features further improves the performance. This is expected because we used a relatively small vocabulary size when training our RNN model, and it may fail to able to recognize some keywords in a question. The word embedding information in our handcrafted features can mitigate this problem. In addition, the question frequency feature provides useful

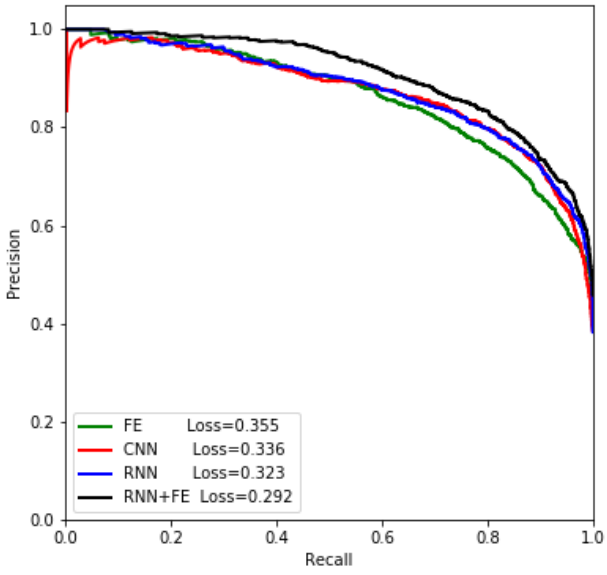

Figure 7: This figure shows Precision-Recall of all the four models we tested, with losses.

Table 4: Test Accuracy Comparison

Ours(RNN + FE)	**86.9%**
Dadashov[7]	83.8%
Homma[8] &Sy	85.2%

prior information for our task.

To demonstrate our model's performance more clearly, we found two recent works on this dataset and compare our accuracies(see Table 4). Note that our model is not tuned to optimize accuracy, and we only report it here for the convenience of comparison.

4 Related Work

One most straight forward approach to duplicate question detection is to apply semantic parsing to both questions and compare their meaning representation. One instance of semantic parsing method is (Kate, 2007), which trains SVM with string similarity kernel to find the most likely semantic parse. The method it proposed had the advantage of being able to learn under a wide range of supervision. However, in our situation, the only supervision available is a binary label indicating whether the two questions are semantically equivalent of not. Therefore it is extremely challenging to construct an accurate semantic parser from our data.

If we view it from a different perspective, duplicate Question detection is a typical sentence classification problem, which requires the ability to capture the semantic meaning of the questions. Traditional methods usually involve coming up with some carefully designed feature and train an effective classifier that takes the feature and produces a prediction. We have shown that these features do work, but not perform as well as the neural network based methods. This is because the hand-crafted features capture mostly superficial characteristics of the sentences, such as length, common words, and min number of deletions, insertions and substitutions to transform one to another [9], but could not capture any real semantics. In this project, we also calculated some distance/similarity measures (Cosine, Euclidean, Manhattan, etc.) based on the word embeddings as well as the embedding themselves as features. These features are not purely "hand-crafted", as they benefit from the recent deep learning techniques.

Recently in the Natural Language Processing community, there is a trend to shift from feature engineering to architecture engineering with deep learning. People have devised various types of Neural Network architectures that are able to learn desirable features from labeled data.

(Kim, 2014) is one of the first study applying CNN to sentence classification problems, and achieved state-of-the-art performance in 4 of the 7 tasks. They proposed using both static and re-trained word embedding vectors and achieved slightly better results than not doing so. (Bogdanova et al., 2015) applied CNN to a very similar task with ours using online user forum data. They applied cosine similarity on the convoluted and max-pooled vectors to evaluate question similarity, and applied mean-squared-error as loss. To be consistent with Quora's evaluations, we used cross-entropy loss. (Kim, 2014) used a single fully connected layer. We added a Relu layer in between which works well.

(Kiros et al., 2015) proposes an encoder-decoder architecture using Recurrent Neural Networks with Long Short Term Memory(LSTM) cells. The net-

[9] https://github.com/seatgeek/fuzzywuzzy

work was trained to take a sentence in a paragraph and predict the previous and next sentence. The learned sentence encoding can then be used as features to train for other tasks such as semantic relatedness and paraphrase detection. Their method achieves performance that's very close to the State of Art in both tasks. In our problem, the task can be considered a union of semantic relatedness, paraphrase detection and a few other tasks. The sentences are restricted to question types, and because of this shift of input domain, we make the network parameters tunable during training, as opposed to them.

5 Future work

In the future, we will continue to enhance our approach in several ways. First of all, even after our preprocessing, there's still a significant number of words that cannot be recognized using our vocabulary, and we are going to explore reliable spelling correction. Secondly, for our RNN approach, we are going to implement wider context and try implementing the attention mechanism. Thirdly, we are going to incorporate better handling strategy for languages other than English.

6 Conclusion

In this project, we explored different types of features for the task of duplicate question detection. We found that combining our fine-tuned RNN feature with our hand-crafted features yields the best performance, with a test loss of 0.292 and a test accuracy of 86.9%. Although CNN is not considered as the "to-go" tool for language processing, we found CNN generating competitive results for this task. Our results reveal that traditional feature-engineering techniques are not out-dated in the deep learning era. In designing real world application systems, different kinds of models should be carefully experimented, compared, and ensembled to exploit their own advantages and optimize the performance.

Acknowledgments

We acknowledge the contribution of the Question Pair Dataset by Quora, and we would like to thank Shallue[10] for providing their code and tuned parameters for the skip-thoughts architecture. Lastly, we would like to thank Prof. Raymond J. Mooney[11] and his Teaching Assistant Abhishek Sinha for their technical help throughout this project.

References

Dasha Bogdanova, Cícero Nogueira dos Santos, Luciano Barbosa, and Bianca Zadrozny. 2015. Detecting semantically equivalent questions in online user forums. In *CoNLL*.

Rohit J. Kate. 2007. *Learning for Semantic Parsing with Kernels under Various Forms of Supervision*. Ph.D. thesis, Department of Computer Sciences, University of Texas at Austin, Austin, TX, August.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *CoRR*, abs/1408.5882.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. 2015. Skip-thought vectors. *CoRR*, abs/1506.06726.

Ye Zhang and Byron C. Wallace. 2015. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *CoRR*, abs/1510.03820.

[10] https://github.com/tensorflow/models/tree/master/skip_thoughts [11] https://www.cs.utexas.edu/users/mooney/