Detecting Duplicate Question Pair

Author: Sanket Patel
Email: stp8954@berkeley.edu

Abstract

Online Q&A forums like Quora and Stack Overflow face a prevalent problem of duplicate questions and often rely on the users for tagging a question as duplicate or redirecting to a previously answered question. An automated system allows them to improve the user experience and the quality of their service by automatically tagging questions as duplicates. In this paper I explore some of the existing and proposed approaches to detecting similar questions on the Quora dataset released as part of the "Quora Question Pair" Kaggle Competition. I will explore couple of approaches 1. A combination of hand crafted features and Linear models and 2. Applying Neural Nets (CNN/RNN) to raw data.

1. Introduction

Detecting semantic similarity and equivalence in sentences has many application in industry, ranging from paraphrase detection for machine translation, detecting duplicate questions in Q&A forums to chatbots. Quora, one such Q&A forum in order to improve their duplicate qustion detection, released a dataset of labelled Question pairs.

Two questions are considered semantically equivalent if they can be adequately answered by the exact same answer. The dataset consists of 400k training samples, with each sample consisting of a pair of questions which are labelled Duplicate or Not Duplicate by human graders.

2. Background

Document similarity is a very old problem in natural language processing and there have been many novel approaches like K-Shingling (Broder [1]) to quantify the similarity between two text documents or using Support Vector Machine based learning using a set of hand crafted features.

Recent advancements in deep learning methods have made significant gains in tasks like semantic equivalence detection, surpassing traditional N-gram based techniques. Long Short Term Memory cells are a recent excitement in natural language processing since they are particularity good at dealing with sequential inputs. Attention mechanism provides an even more sophisticated approach such that it allows each item in the output sequence to condition on the selective items in the input sequence. It achieves this by keeping the intermediate outputs from the encoder LSTM from each step of the input sequence and training the model to learn to pay selective attention to these inputs and relate them to items in the output sequence. In [4] Rocktschel et al. uses Attention

mechanism to achieve a test accuracy of 83.5% on the Stanford Natural Language Inference corpus.

3. Approach

The Quora dataset provides a completely labeled dataset of pairs of questions. In the pre-processing step, I first lowercased the questions and removed all punctuations. I then replaced common word phrases like '*what's*' with '*what is*' and removed all single character words. Keras tokenizer was used to build word index of all the questions and to convert the question strings into word tokens. GloVe word embeddings downloaded from http://nlp.stanford.edu/data/glove.840B.300d.zip was used to create word embeddings. Each word in the sentence was replaced with the corresponding ID from GloVe and words not in GloVe were removed. The word embeddings were initialized to the 300-dimensional GloVe vector pre-trained by Pennington et al. [6] on the Wikipedia 2014 and Gigaword 5 datasets.

The input questions vary significantly in length from empty (0-length) up to 237 words. In order to batch our computations during training and evaluation using matrix operations, the inputs had to have a fixed length. For this the shorter sequences were padded with zero-padding and the longer sequences were truncated. The sequence length being the hyperparameter, to experiment with different length, training sets with multiple sequence length were created.

Lastly, the data was split into training, development (validation) and test sets. Test set was 10% and validation set was 10% of the remaining set.

3.1. Feature-Engineering and Logistic Regression

For the baseline model, engineered features were created. The features include, following types

Simple Features:

- Question character length
- Question word count
- Difference in lengths
- Number of common words

Distance Features:

- Cosine, jaccard, cityblock, Canberra, Euclidean, minkowski and braycurtis distance
- Skew and Kurtosis

Fuzzy Features:

A python library FuzzyWuzzy was used to create some fuzzy features. It provides an api to perform fuzzy string matching by using Levenshtein Distance to calculate the difference between sequences.

POS Features:

- Number of unique POS tags
- Common POS tags
- Difference in number of distinct POS tags

The features were then normalized using Sklearn's StandardScalar and Logistic regression was used to train a model.

	precision	recall	f1-score
0	0.85	0.61	0.71
1	0.55	0.81	0.65
avg / total	0.74	0.68	0.69

Table 1: F1-Score for Logistic Regression model

Logistic Regression performed well for obvious duplicate and non-duplicate sentence. Table [2] lists some correct classification samples.

	question1	question2	is_duplicate
161303	How can I download Tor browser?	How do I download the Tor browser?	1
126718	What languages should a programmer learn?	What language should a programmer learn?	1
10670	Is it okay to work as an IT consultant in a pr...	What are some video games that despite very co...	0
268227	Any Java libraries for debugging and disassemb...	Is there any mobile application identified fak...	0

Table 2: Correct classification samples

Table [3] lists some samples of mis-classifications. As seen in the table LR model struggles where context is needed to make decisions. Eg. Sample 1 have only 2 words in common, but they mean the same. Another feature based on embedding distance could have helped with such scenarios.

	question1	question2	is_duplicate
120061	What might be the business plan in launching reliance jio?	What is the revenue model of Reliance Jio?	1
13217	What is dialogue writing?	What is 2+2 dialogue?	0
32097	Who painted this?	Who painted this painting? When was it painted?	0
340547	What is the father name of PR?	What is your father's name?	0
161731	Why are blue and red neon lights illegal or restricted for commercial uses in Mexico?	Why are blue and red neon lights illegal or restricted for commercial uses in Canada?	0

Table 3: Wrong classification samples

3.2. Bag of Words Model

For the BOW model, a dense vector representation of the questions was created and fed into a feed forward neural network. Each question was embedded into a 300-dimensional GloVe embedding. To feed into a neural network, a fixed sequence size n (25,40,50,75,100) was used which resulted in a [n,300] dimension matrix. The two representations were then merged using sum operation and fed to a feed forward network with relu activation, dropout and BatchNormalization. The dropout layer helps the model generalize better and BatchNormalization improves the performance and stability of the neural networks by normalizing the inputs of each layer in such a way that they have a mean output activation of zero and standard deviation of one. Binary Crossentropy was used as the loss function with adam optimizer.

	precision	recall	f1-score
0	0.85	0.85	0.85
1	0.74	0.74	0.74
avg / total	0.81	0.81	0.81

Table 4: F1-Score of BOW model

Figure [1] illustrates the BOW model architecture.

Figure 1: BOW model architecture

3.3. RNN with LSTM cell

Recurrent neural networks (RNNs) with Long Short-Term Memory units (LSTMs) have been used extensively in neural networks such as in Bowman et al. [5]. Because of their ability to utilize temporal information such as the order of the words in a sentence. In this approach LSTM was used to construct the question representation vector rather than using the sum of the words embedding matrix.

The GloVe embeddings were fed into the LSTM as inputs and the final output of the LSTM was taken as the question representation vector. Then sum of square distance was calculated for the two representations and was then sent through a three-layer neural network with relu activation function, dropout and batch normalization.

Figure 2 illustrates the model architecture for LSTM with distance. Experiments were done with replacing the Distance function with simple Dot, Sum and Maximum functions.

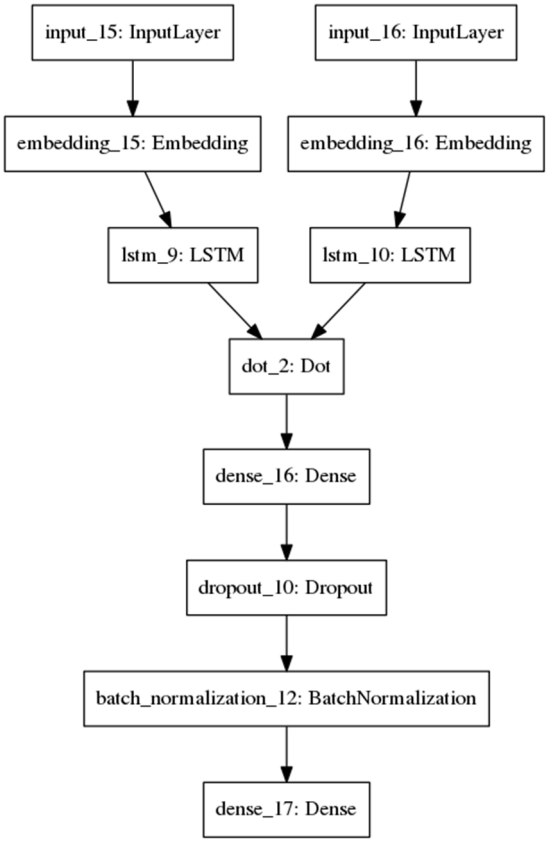

Figure 2: Architecture for LSTM with distance and Dot merge

	precision	recall	f1-score
0	0.86	0.81	0.83
1	0.70	0.77	0.73
avg / total	0.80	0.79	0.80

Table 5: F1-Score for LSTM with distance function

	precision	recall	f1-score
0	0.83	0.85	0.84
1	0.73	0.70	0.71
avg / total	0.79	0.79	0.79

Table 6: F1-Score for LSTM with Dot merge function

3.4. Decomposable Attention Model

The decomposable attention model was developed by Parikh at el. [7] for sentence entailment. With only feed forward neural network and attention, this model is much simpler and computationally more efficient than other complex LSTM models that try to solve similar problems and yet perform better.

The idea of attention mechanism is to overcome the limitation that allows the network to learn where to pay attention in the input sequence for each item in the output sequence. Decomposable attention does so by building alignment (similarity) matrix by multiplying the projection of each question embedding, compute context vectors from the alignment matrix and mix with original signal. The model takes in the matrix of word embeddings as the input for each sentence and performs inter-sentence attention to predict the answer. The logic can be broken down into 3 steps.

Attend: Computes dot product of the projections of words in both questions, normalizes the dot product and uses them to align each word in question1 to a phrase in question2 and vice-versa. These alignments are then used to summarize the aligned phrase in the other sentence as a weighted sum. The initial word projections are computed using a feed forward Neural Network.

$$e_{ij} \coloneqq F'(\bar{a}_i, \bar{b}_j) \coloneqq F(\bar{a}_i)^T F(\bar{b}_j)$$

$$\beta_i \coloneqq \sum_{j=1}^{l_b} \frac{\exp(e_{ij})}{\Sigma_{k=1}^{l_b} \exp(e_{ik})} \bar{b}_j$$

$$\alpha_j \coloneqq \sum_{i:1}^{l_a} \frac{exp(e_{ij})}{\sum_{k=1}^{l_a} exp\,(e_{kj})} \bar{a}_i$$

Compare: For comparison each question representation is concatenated with the aligned phrase and then passed through a feed forward network G.

$$v_{1,i} \coloneqq G([\bar{a}_i, \beta_i]) \qquad \forall i \in [1 \dots \dots l_a]$$

$$V_{2,j} \coloneqq G([\bar{b}_j, \alpha_j]) \qquad \forall j \in [1 \dots i_b]$$

Aggregate: The comparison vectors are then summed and passed through a feed forward network to generate prediction.

$$v_1 = \sum_{i=1}^{l_a} v_{1,i}$$

$$v_2 = \sum_{j=1}^{l_b} v_{2,j}$$

$$\hat{y} = H\big((v_1, v_2]\big)$$

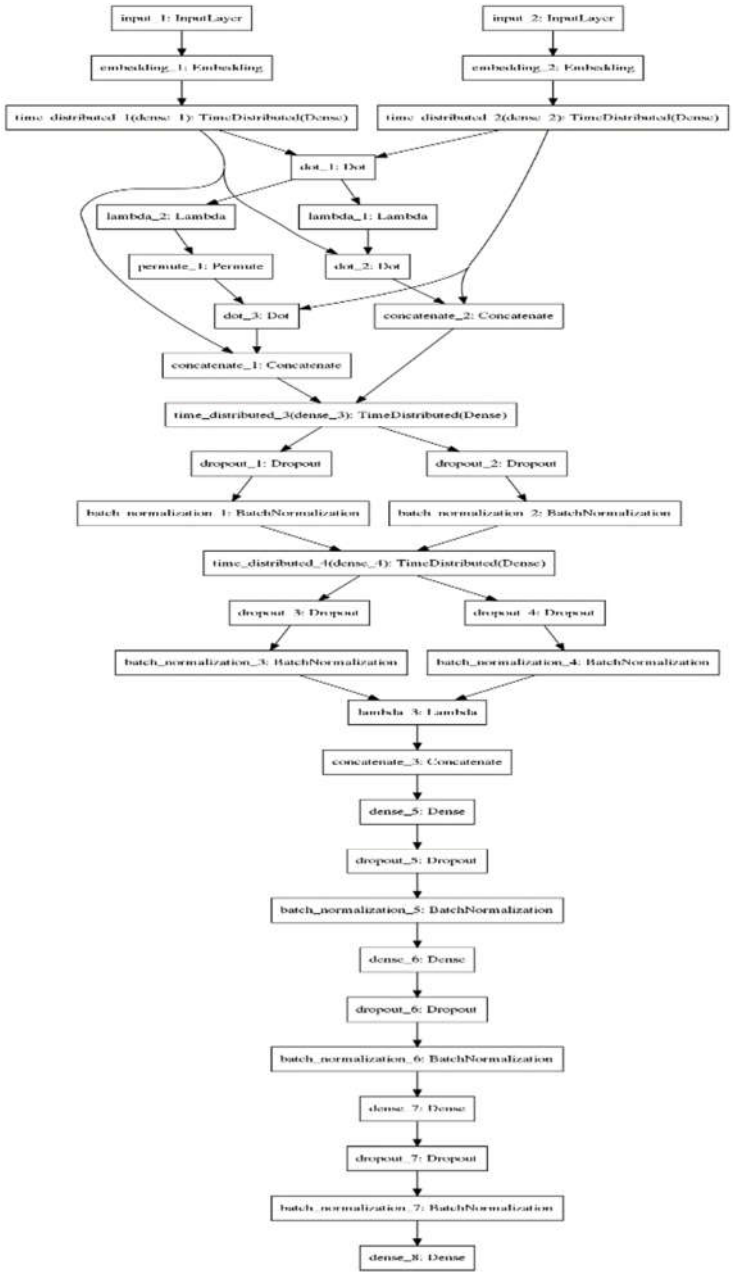

Figure 3: Decomposable Attention Network Architecture

	precision	recall	f1-score
0	0.86	0.85	0.86
1	0.75	0.77	0.76
avg / total	0.82	0.82	0.82

Table 7: F1-Score for Decomposable Attention network

4. Experiments

4.1. Data

The dataset was downloaded from the competition website https://www.kaggle.com/c/quora-question-pairs/data . Each line in the dataset is in the format

[ID] [Question 1 ID] [Question 2 ID] [Question 1] [Question 2] [isduplicate]

Where "is_duplicate" is either 0 or 1 indicating whether the two questions are duplicates. The dataset has a total of 404290 questions in all. Figure 1 shows a sample of the data.

	id	qid1	qid2	question1	question2	is_duplicate
0	0	1	2	What is the step by step guide to invest in sh...	What is the step by step guide to invest in sh...	0
1	1	3	4	What is the story of Kohinoor (Koh-i-Noor) Dia...	What would happen if the Indian government sto...	0
2	2	5	6	How can I increase the speed of my internet co...	How can Internet speed be increased by hacking...	0
3	3	7	8	Why am I mentally very lonely? How can I solve...	Find the remainder when [math]23^{24}[/math] i...	0
4	4	9	10	Which one dissolve in water quikly sugar, salt...	Which fish would survive in salt water?	0

Table 8: Sample lines of the data set

4.2. Setup and Training

All experiments were run on a Desktop PC with 8 Cores, 32GB RAM and a NVIDIA 1060 GPU. Keras with Tensorflow backend was steup using Docker container and all other required dependencies were added to the container.

During training, all weights were initialized with the Xavier initializer and the biases were initialized to zero vector. For all the neural network model cross entropy loss with softmax was used as the loss function. L2-regularization was applied to weights to avoid overfitting.

All models were run for various combinations of sequence length (25,40,50,75,100) , L2-regularization, dense layer size, LSTM hidden layer size , dropout rates and batch size. Accuracy was measured as number of labels predicted correctly divided by the total number of labels predicted.

4.3. Error Analysis

For error analysis results of Decomposable Attention network was analyzed. Both True Positive and False Positive results were analyzed. Based on the analysis some updates were made to pre-processing steps.

Following are some observed issues with classification

- Numbers sometimes carry context
  - "*How can I get taller after 25?","How do I get taller at 18?*"
- Incorrect grammar
  - "*Why does Chechnya not stand on the worldmap?","Why is Chechnya not on the worldmap?*"
- Too much data cleaning?
  - "*What is Bitcoin currency?","Is Bitcoin a currency?*"
- More background knowledge
  - "*How many synovial joints are there in the spine?","How many joints are in the vertebral column?*

4.4. Results

Model	Precision (Test)	Recall (Test)	F1-Score (Test)
Logistic Regression	0.74	0.68	0.69
BOW	0.81	0.81	0.81
LSTM + Distance	0.80	0.79	0.80
LSTM + Dot merge	0.79	0.79	0.79
Decomposable Attention	**0.82**	**0.82**	**0.82**

Table 9: Results of all models on the Quora dataset

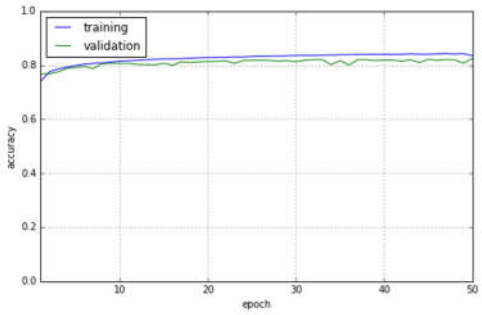

Figure 4: Train-Validation loss With Regularization

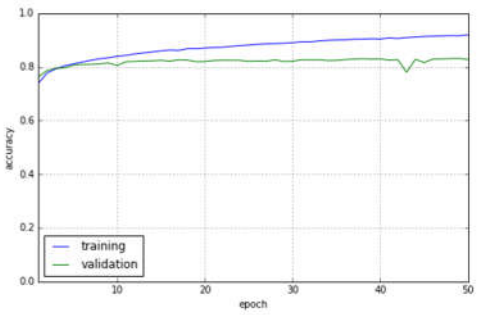

Figure 5: Train-Validation loss without Regularization

For each model multiple experiments were run with different parameters for maximum sequence length, hidden layer size and depth of the feed forward layers. From all the experiments the models with highest accuracy were selected.

From Table 9 it can be concluded that Decomposable attention network had the best accuracy. Considering the simplicity of the BOW model, it did pretty well too. It can be hypothesized that a Word Embedding trained on Quora dataset could help improve performance of both models.

It should be noted that in the paper Parikh at el. Recommended using tanh as activation function, but relu gave lower loss and higher accuracy for this particular dataset. Also applying L2 regularization to hidden layer weights helped the model from overfitting.

5. Conclusion

Surprisingly the simple Bag of Words model, performs better than the LSTM models. And as expected the Decomposable Attention model gets the best result, but unlike the model described in the paper, *relu* activation gave better result than *tanh*.

LSTMs are lot more expensive than non LSTM model. Even thou Decomposible Attention had 3x more trainable features, it was 2x faster than the LSTM models. Even thou both have the same computational complexity, due to the sequential nature of LSTMS they cannot be parallelized.

Also most errors can be attributed to data pre-processing. Some errors are due to over processing and some are due to under processing. Words not in GloVe embedding were removed from the sequence. Replacing them with a constant word token or a random gaussian embedding with mean 0 and standard deviation 0.01 as described in the paper could help improve the model. Spell checking and grammar corrections are other ways to improve the accuracy.

Source Code: https://github.com/stp8954/w266_project

REFERENCES

[1] Broder, A. (1997) On the resemblance and containment of documents. Proceedings of the Compression and Complexity of Sequences 1997, SEQUENCES'97, Washington, DC, USA. IEEE Computer Society.
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.24.779&rep=rep1&type=pdf

[2] Andrai Z. Broder (2013) Discriminative Improvements to Distributional Sentence Similarity
https://www.cc.gatech.edu/~jeisenst/papers/ji-emnlp-2013.pdf

[3] Detecting Semantically Equivalent Questions in Online User Forums
https://www.cc.gatech.edu/~jeisenst/papers/ji-emnlp-2013.pdf

[4] Reasoning About Entailment using Neural Attention https://arxiv.org/pdf/1509.06664.pdf

[5] A Decomposable Attention Model for Natural Language Inference.
https://arxiv.org/pdf/1606.01933.pdf

[6] GloVe: Global Vectors for Word Representation https://nlp.stanford.edu/pubs/glove.pdf

[7] Code Reference https://github.com/bradleypallen/keras-quora-question-pairs

[8] Code Reference https://github.com/explosion/spaCy