# DUPLICATE QUESTION DETECTION

**Ami Ladani**
Department of Computer Science & Engineering
IIT Kharagpur
*amiladani26@gmail.com*

**Archie Mittal**
Department of Computer Science & Engineering
IIT Kharagpur
*mittal.archie93@gmail.com*

**Jagriti Jalal**
Department of Computer Science & Engineering
IIT Kharagpur
*jjalal57@gmail.com*

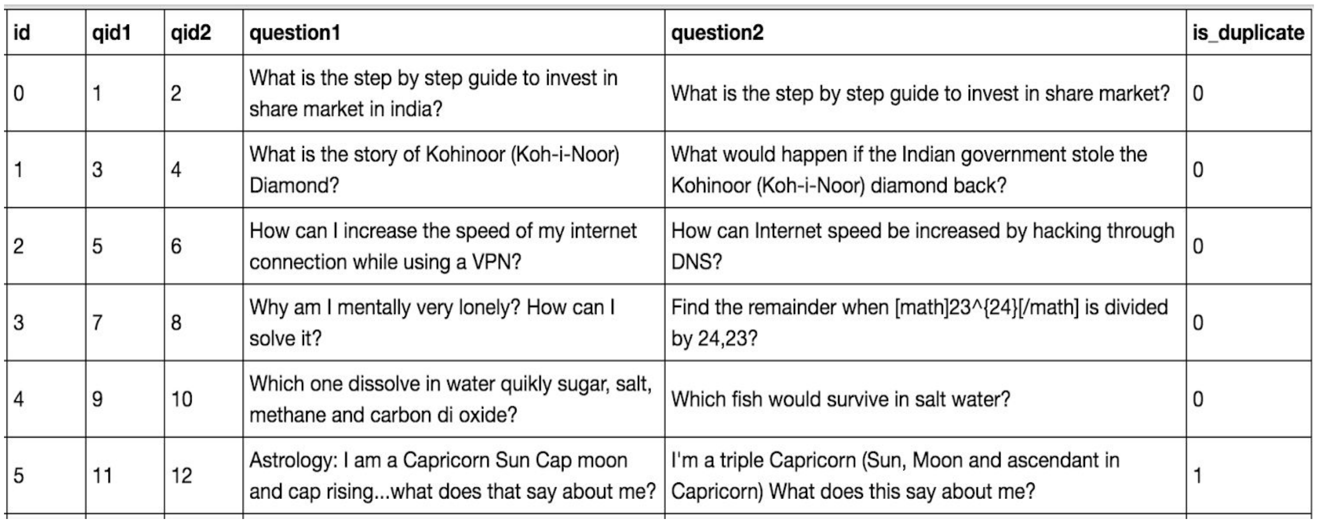
**Nidhi Mulay**
Department of Computer Science & Engineering
IIT Kharagpur
*nidhi27mulay@gmail.com*

## Abstract

In this project, we explore whether two questions are asking the same thing by analyzing the semantic equivalence between pairs of questions using a dataset of more than 400,000 labeled question pairs provided by Quora. Our deep learning approach to this problem uses a Long Short-Term Memory (LSTM) to encode each sentence. We compared the performance of this model with a baseline model which used Feedforward Neural Network. The results obtained shows that LSTMs outperform the baseline model in terms of accuracy. Moreover, both these deep learning techniques significantly outperform traditional NLP based methods and simple multilayer perceptron baselines.

## 1. Introduction

Websites such as Quora provide a platform to user where they can ask questions that are answered by the other users on that site. However, generally many questions that are being asked at any instant of time have already been asked by other users using different phrasing and wording. These redundancy in questions can be really frustrating for the highly active users whose feeds are frequently crowded with these redundant questions. So, our project focuses on detecting semantically equivalent questions to solve this problem in online forums such as Quora and Stack Overflow. Answers for duplicate questions can be combined to improve the efficiency and the quality of their service.



| id | qid1 | qid2 | question1 | question2 | is_duplicate |
|---|---|---|---|---|---|
| 0 | 1 | 2 | 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? | 0 |
| 1 | 3 | 4 | What is the story of Kohinoor (Koh-i-Noor) Diamond? | What would happen if the Indian government stole the Kohinoor (Koh-i-Noor) diamond back? | 0 |
| 2 | 5 | 6 | How can I increase the speed of my internet connection while using a VPN? | How can Internet speed be increased by hacking through DNS? | 0 |
| 3 | 7 | 8 | Why am I mentally very lonely? How can I solve it? | Find the remainder when [math]23^{24}[/math] is divided by 24,23? | 0 |
| 4 | 9 | 10 | Which one dissolve in water quikly sugar, salt, methane and carbon di oxide? | Which fish would survive in salt water? | 0 |
| 5 | 11 | 12 | Astrology: I am a Capricorn Sun Cap moon and cap rising...what does that say about me? | I'm a triple Capricorn (Sun, Moon and ascendant in Capricorn) What does this say about me? | 1 |

*Figure 1: Sample of the Quora ”Question Pairs” dataset*

## 2. Approach

### 2.1 Data Preprocessing

The Quora dataset provides a completely labeled dataset of pairs of questions, as shown in Figure 1. First, we preprocess the data. We tokenize each sentence in the entire dataset using Tokenizer provided by Keras. All characters are converted to lowercase. We then replaced each token with its corresponding ID in the GloVe vocabulary. So the sentence is transformed into sequences of integer corresponding to tokenizer word index. The input questions vary significantly in length, from empty (0-length) up to 237 words. For computations in training and validation using matrix operations, we need all the input questions to have a fix length. To achieve this, we pad the shorter sentences with a designated zero-padding and truncate the longer sentences to the same standardized length, which is a hyperparameter of our model. We have set this maximum length to 25. At the end, we split our data into training, validation and test sets. For evaluating our models and tuning their hyperparameters, we have split the dataset into a training set and a test set, containing 85% and 15% of the question pairs, respectively.

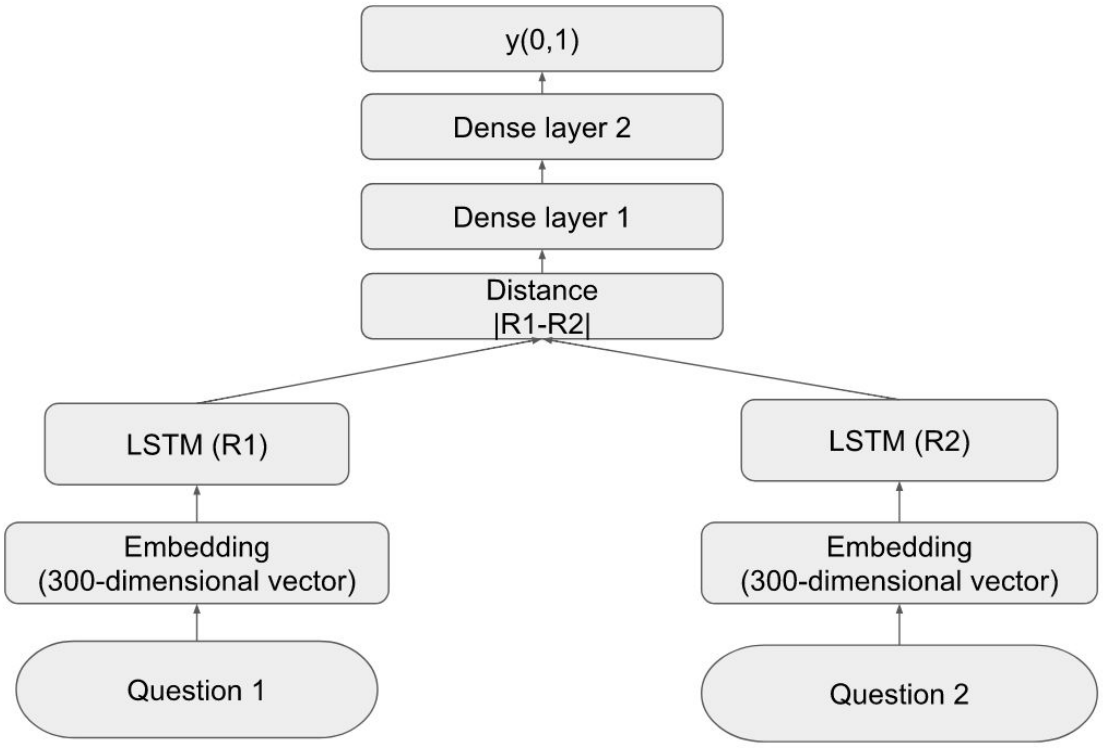


*Figure 2: LSTM architecture*

## 2.2 Sentence Encoding

In our project, we explore two LSTMs to encode each sentence of question-answer pair. During training, we update the word embeddings as well as the weights and biases within the LSTM cells. Each of these cells contains a single layer with the sigmoid activation function. The weights and the biases in the encoding layer are shared for both questions in the pair so that the network is smaller and easier to train. Each of these neural networks output a sentence vector of dimension H, the hidden state size in the LSTM cells. We initialize the word embeddings to the 300-dimensional GloVe vectors. The weights are initialized using word embeddings.

## 2.3 Distance measure

Once the questions of each pair are individually encoded into sentence vectors, we need to combine them in some way to predict whether or not the pair is duplicate. The method we use involves calculating some distance between the sentence vectors and running logistic regression to make the prediction,

$$\widehat{y} = \sigma(a \cdot d(r1, r2) + b)$$

where $\sigma$ is the logistic or sigmoid function, $a,b \in R$ are learned parameters, and $d : R^H \times R^H \rightarrow R$ is a distance function between the two sentence vectors.
Inspired by previous work from Bogdanova et al. [5], we initially tried cosine distance. We also tested Euclidean (L2) distance and Manhattan distance.

$$d_{man}(r1,r2) = |r1 - r2|$$

We finally use Manhattan distance as it is giving more accuracy than any other distance in our model.

### 2.4 Loss

For the distance-logistic regression method, we use the binary-cross entropy loss as our loss function, adam optimizer and batch-normalization. To avoid overfitting, we use drop-out in dense layer 1 with fraction 0.3.

$$H(p,q) = -\sum_x p(x)\ \log q(x).$$

## 3. Experiment

The experiment was carried out by first building a simple feed-forward network to measure the distance between questions of a question pair in the feature space and classifying the pair as duplicate or not based on that.

### 3.1 Baseline model

The baseline model consists of :

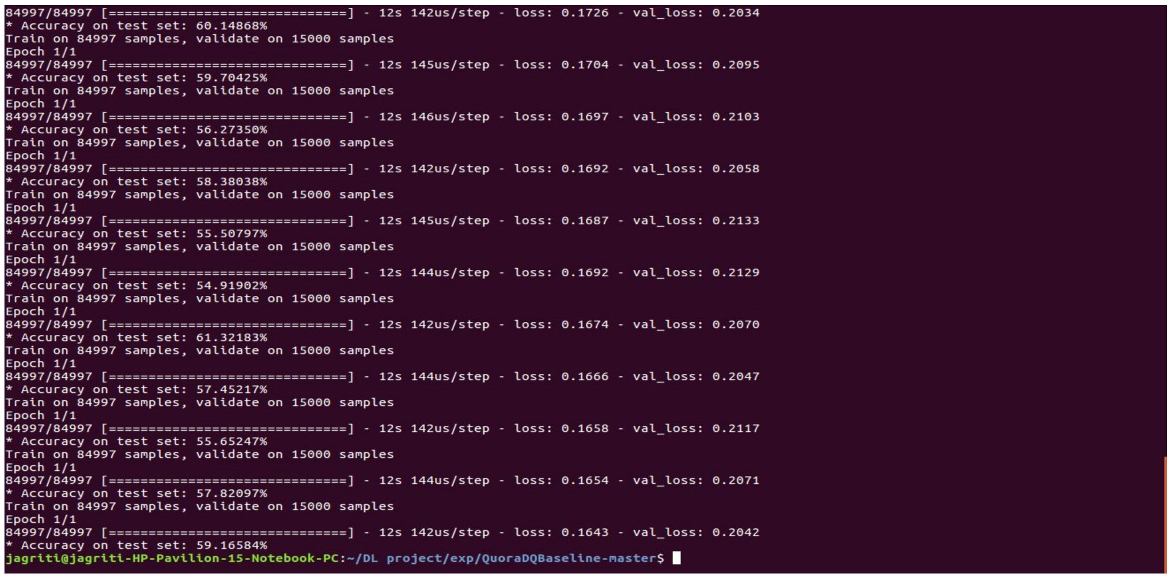
1. **Sentence encoding:** We used a word2vec model to convert the question words into word vectors to better capture the semantic relationship among the words in the corpus. The corpus here is the collection of all the words in all the questions of the dataset. Next, the questions were encoded as 300-dimensional feature-vectors using the built-in function tf-idf Embedding Vectorizer() from keras.

2. **Similarity analysis:** A simple feed-forward network with three hidden layers was created for this. Size of the hidden layers is 128 hidden units each. The objective is to reduce the

vector distance between the two question of a question pair in the dataset. For this, the objective function used is simple Euclidean distance between the two 300-dimensional tf-idf vectors corresponding to the two questions obtained from step 1. The network is fully-connected and the loss function used to optimize the objective function is contrastive loss given as:

$$L_{contrastive} = y\_true * (y\_pred)^2 + (1 - y\_true) * (\max(1 - y\_pred, 0))^2$$

Also, the optimizer used to optimize the model was Adam optimizer, also a built-in facility of keras. The model was trained for 50 epochs and then tested with the test-set. The accuracy achieved was approximately 60% as shown in figure 3.



```
84997/84997 [==============================] - 12s 142us/step - loss: 0.1726 - val_loss: 0.2034
* Accuracy on test set: 60.14868%
Train on 84997 samples, validate on 15000 samples
Epoch 1/1
84997/84997 [==============================] - 12s 145us/step - loss: 0.1704 - val_loss: 0.2095
* Accuracy on test set: 59.70425%
Train on 84997 samples, validate on 15000 samples
Epoch 1/1
84997/84997 [==============================] - 12s 146us/step - loss: 0.1697 - val_loss: 0.2103
* Accuracy on test set: 56.27350%
Train on 84997 samples, validate on 15000 samples
Epoch 1/1
84997/84997 [==============================] - 12s 142us/step - loss: 0.1692 - val_loss: 0.2058
* Accuracy on test set: 58.38038%
Train on 84997 samples, validate on 15000 samples
Epoch 1/1
84997/84997 [==============================] - 12s 145us/step - loss: 0.1687 - val_loss: 0.2133
* Accuracy on test set: 55.50797%
Train on 84997 samples, validate on 15000 samples
Epoch 1/1
84997/84997 [==============================] - 12s 144us/step - loss: 0.1692 - val_loss: 0.2129
* Accuracy on test set: 54.91902%
Train on 84997 samples, validate on 15000 samples
Epoch 1/1
84997/84997 [==============================] - 12s 142us/step - loss: 0.1674 - val_loss: 0.2070
* Accuracy on test set: 61.32183%
Train on 84997 samples, validate on 15000 samples
Epoch 1/1
84997/84997 [==============================] - 12s 144us/step - loss: 0.1666 - val_loss: 0.2047
* Accuracy on test set: 57.45217%
Train on 84997 samples, validate on 15000 samples
Epoch 1/1
84997/84997 [==============================] - 12s 142us/step - loss: 0.1658 - val_loss: 0.2117
* Accuracy on test set: 55.65247%
Train on 84997 samples, validate on 15000 samples
Epoch 1/1
84997/84997 [==============================] - 12s 144us/step - loss: 0.1654 - val_loss: 0.2071
* Accuracy on test set: 57.82097%
Train on 84997 samples, validate on 15000 samples
Epoch 1/1
84997/84997 [==============================] - 12s 142us/step - loss: 0.1643 - val_loss: 0.2042
* Accuracy on test set: 59.16584%
jagriti@jagriti-HP-Pavilion-15-Notebook-PC:~/DL project/exp/QuoraDQBaseline-master$
```

*Figure 3: Results for the baseline model*

### 3.2 Results

For evaluating our models and tuning their hyperparameters, we randomly split the remaining non-test data into a training set and a validation set, containing 85% and 15% of the question pairs, respectively. In assessing the performance of our model, we looked primarily at training and validation accuracy and compared it for different hidden layer sizes, the results for which are shown in Figure 4 and Figure 5 respectively.

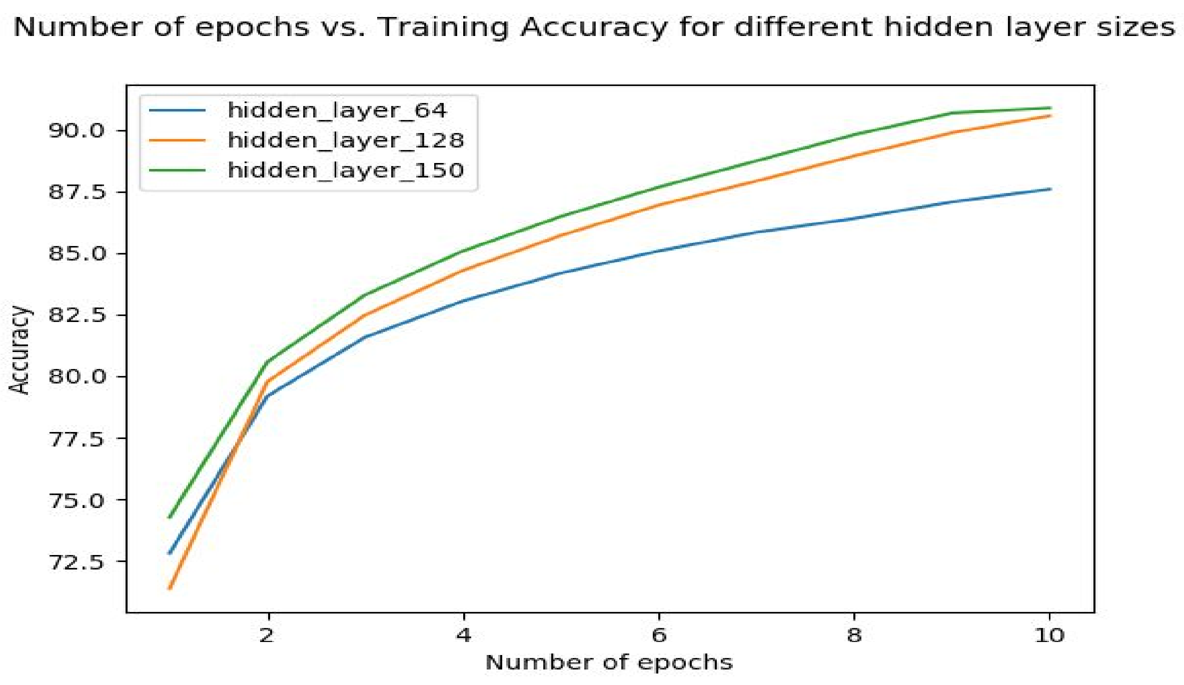


*Figure 4: Comparison of training accuracy for different hidden layer sizes*

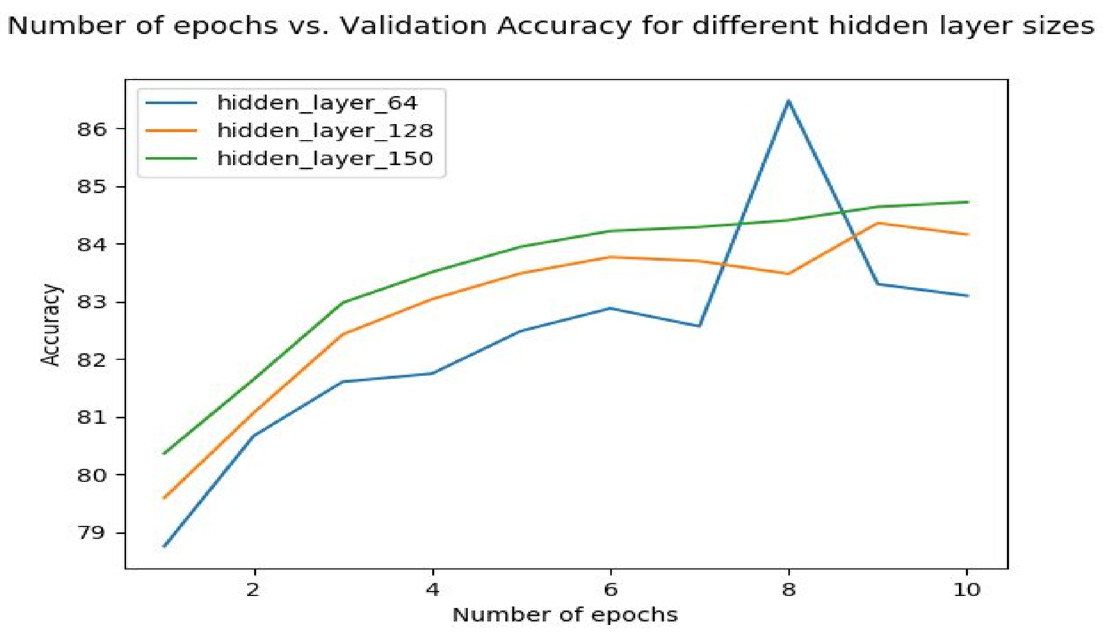


*Figure 5: Comparison of validation accuracy for different hidden layer sizes*

After tuning the hyperparameters (standardized length and hidden layer size) on the validation set, we ran our models on the test sets. The results are shown in Figure 6. Our LSTM model achieved 82.9% accuracy comparable to the 60% accuracy achieved by our baseline model.

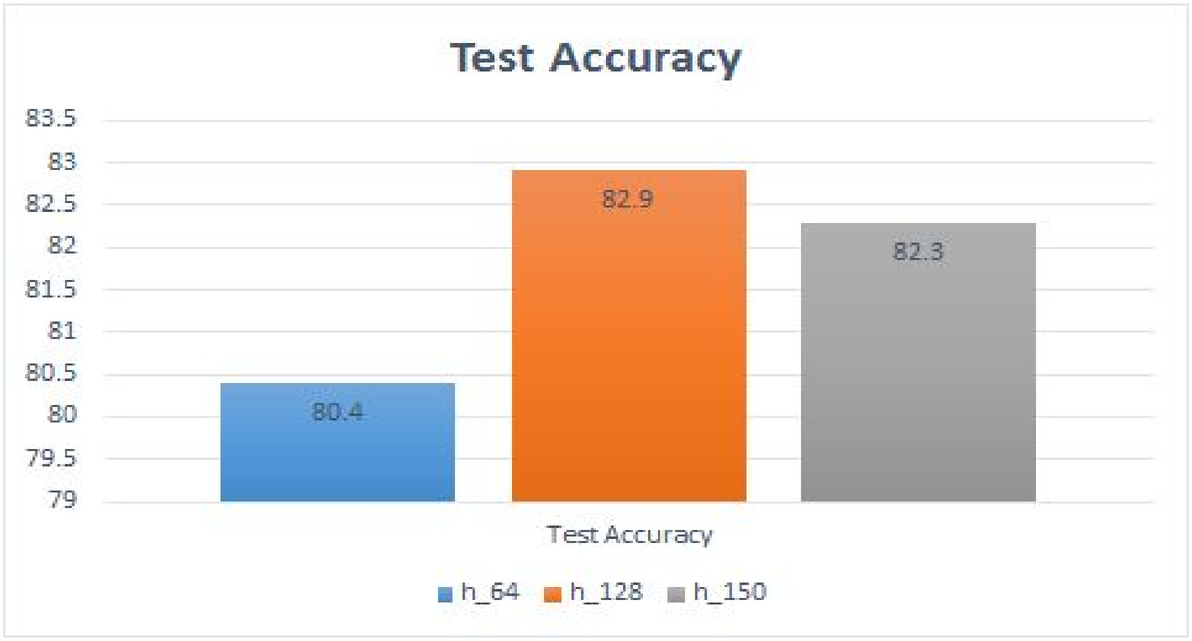


*Figure 6: Test accuracy for different hidden layer sizes*

## 4. Conclusion

Deep learning methods have outperform the baseline for the task of duplicate question pair detection. The LSTM architecture achieved a 20% performance gain over a simpler Feedforward architecture. The LSTM method had comparable and much better performance than the baseline and was much faster to train. Also, we saw that accuracy increases on increasing the hidden layer size.

## 5. References

[1] Richard Socher Jeffrey Pennington and Christopher D. Manning. Glove: Global vectors for word representation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, pages 1532–1543, 2014.
[2] Adrian Sanborn and Jacek Skryzalin. Deep learning for semantic similarity. 2015.

[3] 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.

[4] Samuel R. Bowman, Jon Gauthier, Abhinav Rastogi, Raghav Gupta, Christopher D. Manning, Christopher Potts. (2016). A Fast Unified Model for Parsing and Sentence Understanding. https:// arxiv.org/pdf/1603.06021.pdf.

[5] Dasha Bogdanova, Cicero dos Santos, Luciano Barbosa, and Bianca Zadrozny. Detecting semantically equivalent questions in online user forums. Proceedings of the 19th Conference on Computational Natural Language Learning, 1:123–131, 2015.