Identifying Duplicate Questions with Siamese Neural Networks

Arnav Ghosh, Zhao Shen, and Brandon Kates
Cornell Data Science, Cornell University

The task of identifying questions that have the same intent has a variety of applications, including information centralization and question redirection. We believe that deep learning techniques are capable of performing well in this task, and use a Siamese neural network architecture to achieve results that are comparable to state-of-the-art. As this research is still in progress, we also articulate modifications to our model architecture that could potentially result in even better performance.

I. INTRODUCTION

Our goal is to identify duplicate questions, where such questions are defined as having the same intent and maintaining the property that any answer to one question could be an acceptable answer to the other.

The most apparent applications of a high-performance model for this task include information centralization and question redirection in question-answer forums, where the former results in the aggregation of existing question threads and the latter suggests threads to users wishing to submit a question that has already been answered before.

Our model should be able to take in two questions and determine whether or not they are duplicates. We identified a Siamese neural network architecture as the most intuitive manner to achieve this goal, since it processes distinct inputs with identical networks whose outputs are then accumulated and processed with some function to produce a final output. In our task, the idea is to use these twin networks on each question to produce two question embeddings, and to then use some similarity function to decide whether to classify them as duplicates.

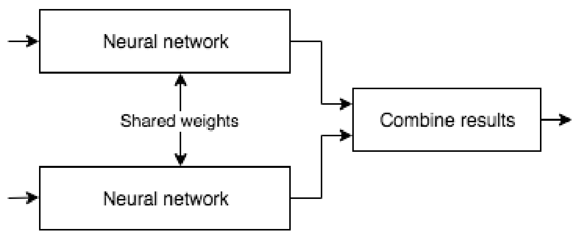

FIG. 1. A Siamese neural network uses two identical networks to process two inputs, and uses some function to unify those outputs.

II. DATA

The dataset we use is Quora Question Pairs, a public Kaggle dataset pulled from the popular question-answer forum Quora. This dataset contains 404,351 pairs of questions, each with the following attributes.

id: A unique identification tag for the question pair (integer).

qid1: A unique identification tag for the first question (integer).

qid2: A unique identification tag for the second question (integer).

question1: The text representation of the first question (string).

question2: The text representation of the second question (string).

is_duplicate: Whether or not the two questions are duplicates (0 or 1).

III. APPROACH

A. Data Preprocessing

In addition to replacing some dataset-specific characters so that we could convert the strings to the UTF-8 format, we lemmatized the data, turning words into their root forms so that the model might be able to generalize better and not overfit to specific tenses or word forms.

We also removed question pairs from the dataset that contained questions that were too short or long. Over 99% of our questions had word lengths in the range of 3 to 50 (see Figure 2). Shorter sentences are nonsensical statements such as "aaaa", whereas longer sentences ramble on about highly specific circumstances. We removed these extreme-length question pairs to improve the generalizability of our model and stop it from fixating on meaningless or unhelpful cases.

In addition, we augmented our data by synthesizing new question pairs from the existing entries in the dataset. We note that questions that are identical are naturally duplicates, so we create new data that matches unique questions to themselves. We also note that our network is not fully symmetric, so we also swap the order of questions in each pair. With these two steps we more than double the size of our dataset, and give our model a significant boost in both performance metrics and overfit prevention.

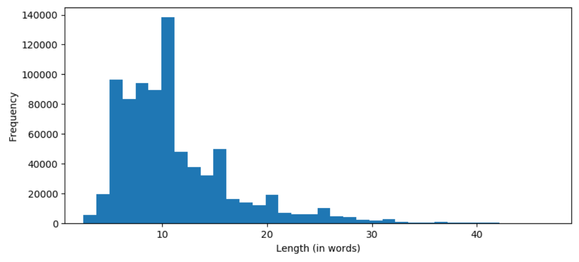

FIG. 2. A histogram of sentences based on sentence length.

1. Did Ronald\NNP Reagan\NNP have\VB a mannerism\PRP in his speech\NN?
2. How\WRB did Ronald\NNP Reagan\NNP react\VB to 9/11\CD?

$$\Downarrow$$
$$\{WRB, CD, PRP, NN\}$$

FIG. 3. Obtaining a bag of POS tags from a question pair.

B. Baselines

We explore two baselines: a multinomial naive bayes classifier and a linear support vector machine classifier with L1 regularization. Both these baselines employ the following features:

Jaccard Similarity: Each question was represented as set of shingles of lengths one to four. We used the Jaccard Coefficient[1], as shown below, to obtain a Jaccard Similarity Score for the two questions.

$$J(S(d_1), S(d_2)) = \frac{S(d_1) \cap S(d_2)}{S(d_1) \cup S(d_2)}$$

POS Tags of Unshared Words: Each question was represented as a bag of part-of-speech tags of the words contained in the question. We used the difference of these two sets as features. POS tags were chosen instead of the words themselves because the questions cover a wide variety of domains and consequently, the same content words do not feature frequently in the dataset.

C. Deep Learning

At a high level, our deep learning model is a Siamese neural network architecture that takes in two inputs (the questions) and runs each of them through a GRU recurrent neural network, then uses the two outputs in a similarity function to determine whether to classify the questions as duplicates. We use L2 regularization with a lambda value of 0.0001 on the weights and biases of our network.

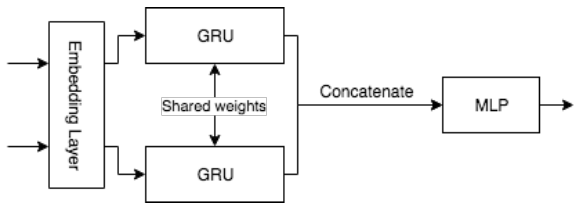

FIG. 4. Our Siamese network architecture. GRU outputs are concatenated and fed into a multi-layer perceptron.

After preprocessing our data, we split the questions into individual words. We then convert each of these words into a vector using a pretrained GloVe word embedding model [3]. We use the 300-dimensional Wikipedia 2014/Gigaword 5 embeddings, so each word becomes a 300-dimensional vector. At test time, we handle unknown words by assigning them the average embedding over all words contained in the GloVe model.

We set the number of cells in our GRU model to exactly the maximum sentence length that we cut out earlier in preprocessing (50 in our case). We pad the shorter questions with zero embeddings and mask those cells so that they do not affect the state of the network. At test time, we would truncate the beginnings of longer questions, since we assume that in those longer rambling sentences, the last words are the ones most relevant to the question.

For each question, we feed its word embeddings into the GRU and use the state of the network at the end of processing each of these embeddings as our output. We process each question with identical networks, which we ensure by sharing weights during training.

Our similarity function of choice is a multi-layer perceptron with a single hidden layer, ending in a softmax that outputs the probability of the questions being duplicates. We do this to take advantage of the ability of neural networks to discern indirect relationships (as opposed to direct relationships such as in Euclidean similarity), but only use one hidden layer to keep both the number of parameters and the complexity of the discerned similarities relatively low, to reduce the probability of overfitting.

$$[q1 \quad q2 \quad |q1 - q2| \quad q1 \odot q2]$$

To input our two questions into the fully connected network we concatenate the two sentence embeddings as well as their element-wise product and the absolute value of their difference. We add the latter two features in order to help the network learn valuable features immediately. This concatenation step is the source of asymmetry in the network, and one of the reasons that flipping question pairs in preprocessing is so effective. The fully connected network outputs the probability that our questions are duplicates.

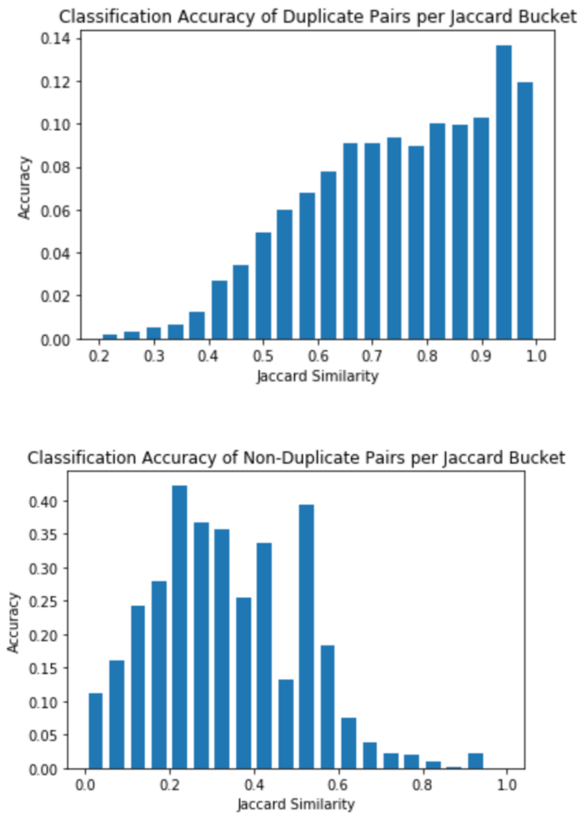

FIG. 5. Validation accuracy of the SVM on the two types of question pairs when partitioned by Jaccard Similarity.

IV. EXPERIMENTAL RESULTS

A. Baseline Results

Model	Val Acc.	Val F1.	Optimal Parameters
Naive Bayes	0.597	0.545	--
Linear SVM	0.659	0.575	C=1.0

TABLE I. Results of the two baselines on the validation set.

We performed a grid-search to find the optimal value of C for the SVM classifier. Across all values of C, the weights placed on Jaccard Similarity were consistently an order of magnitude higher than every other feature, suggesting that the classification was primarily being performed on word overlap between the two questions.

Furthermore, we observed that the classifier placed non-zero weights on only four features: Jaccard Similarity, Nouns, Verbs and Prepositions. Interestingly, neither adjectives or adverbs were assigned non-zero weights, contradicting previous studies that point out their usefulness in identifying duplicate questions, though this would require validation on other datasets.

Fig.5 shows the distributions of questions that have been correctly classified by the SVM classifier with respect to the pairs Jaccard Similarity. In Fig.5.a, we observe that duplicate questions that have a high Jaccard Similarity are more likely to be correctly classified than those with lower Jaccard Similarity. In contrast, the sharp fall at a Jaccard Similarity of 0.6 in Fig.5.b indicates that the classifier is less likely to correctly classify non-duplicate questions that have a high Jaccard Similarity.

Consequently, this dependence on the Jaccard Similarity and the distribution of question pairs in the dataset allows us to explain the contrasting performance of the models on the two types of question pairs. Both types of question pairs in the dataset have primarily low-to-moderate Jaccard Similarity. Thus, the model is able to classify non-duplicate questions fairly well because many of them have a low enough Jaccard Similarity to be able to be classified as non-duplicates. For duplicate questions, the model is looking for high word overlap, which is not the case in this dataset and thus, the classification accuracy takes a hit.

These results are consistent with the idea that our approximation of the semantic relation between two questions via content word overlap is not sufficiently nuanced for the problem.

B. GRU Results

Model	Train Acc	Train F1	Val Acc	Val F1
Euclidean similarity	0.938	0.950	0.748	0.759
MLP similarity	0.963	0.951	0.838	0.790

TABLE II. Training/validation set accuracy/F1 score on different Siamese models.

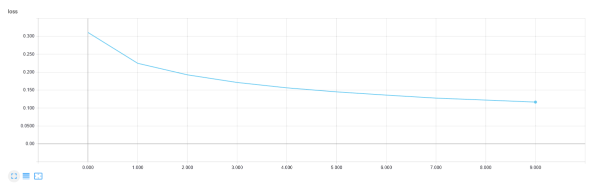

FIG. 6. Training accuracy of MLP model.

We averaged our 5-fold cross-validation results to obtain our model statistics. As expected, our Euclidean similarity architecture performs worse than our multilayer perceptron similarity architecture.

However, our MLP similarity architecture is still very flawed, as illustrated by the erratic validation loss graph (see Figure 7). We believe that this is a combination of overfitting as well as having the potential to use loss

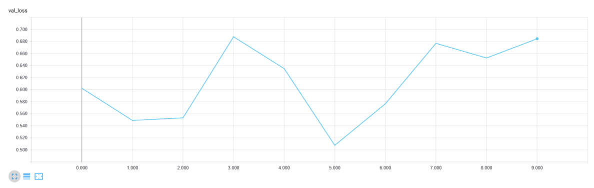

FIG. 7. Validation accuracy of MLP model.

functions that are more suited to our task such as triplet loss, and will focus on methods to correct these issues in the future (see section V: Future Directions).

As noted earlier, the dataset obtained doesnt categorize questions by any useful metric, either by the topic, type or the amount of paraphrasing between questions. Thus, in order to understand the strengths and limitations of our models, we evaluate their performance on a specific partition of the validation set.

A question pair is classified as an **elaboration** if one question contains the other question in its entirety with word order preserved, and has additional words appended and/or prepended.

$$\overbrace{\underbrace{\text{Why is poverty considered a social problem}}_{\text{Question 1}}\text{ in Cuba}}^{\text{Question 2}}$$

As indicated in Table.III, the models accuracy on this set of questions pairs is lower than the overall accuracy. Furthermore, we observe that 81% of these questions were classified as duplicates, whereas only 51% of the elaborations are duplicates. Because, the question pairs only differ in a small amount of words, this suggests that the model is not able to sufficiently capture the semantic content of a given question and we would like to run further experiments to validate this claim.

Accuracy	F1-Score	Precision
0.720	0.754	0.891

TABLE III. Results for the GRU Model on the 1,471 elaboration question pairs in the validation set.

V. FUTURE DIRECTIONS

A. Embeddings Based on Triplet Loss

We would like to explore the triplet loss function as described in the FaceNet paper (see Figure 8). We'd use an architecture of three neural networks with shared weights and combine them to generate the embeddings. The paper uses CNN's to generate embeddings on images, but we would use GRU's to generate embeddings of our sentence pairs.[2]

The main challenge with the triplet loss method is triplet selection. The Quora dataset gives us pairs of questions identified as duplicates, and pairs not identified as duplicates. The triplet loss requires that we have two vectors with the same label, and one with a different label. Therefore, the only useful data are pairs of duplicate questions. We can then augment those pairs with a random question from the rest of the dataset. The FaceNet algorithm discusses different kinds of triplets, including hard positive and hard negative which are the best for training the network. They would be computed automatically from our randomly generation sets of triplets.

Additionally, we have a different dataset than the FaceNet paper. We want to identify if a pair of questions are duplicates, whereas FaceNet is a multi-class classification for images. Therefore, we require significant changes to the algorithm in order for triplet loss to work for sentences.

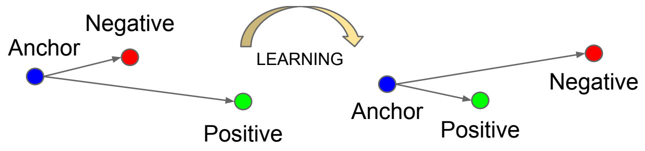

FIG. 8. The **Triplet Loss** minimizes the distance between an *anchor* and a *positive*, both of which have the same identity, and maximizes the distance between the *anchor* and a *negative* of a different identity.[2]

B. Data Augmentation

The dataset is imbalanced, with far more duplicate questions than non-duplicates. Thus, it would be prudent to add more non-duplicate pairs by picking two questions from two different pairs and labelling them as non-duplicates.

VI. RELATED WORKS

Duplicate question detection has been the topic of previous papers, including that of Addair [3], who used convolutional neural networks and LSTMs to retrieve embeddings for each question, and that of Homma et al. [4], who used GRUs and vanilla RNNs for the embedding step and performed much experimentation with different similarity functions. The proposed models in these papers were also constructed with the Siamese neural network architecture.

[1] Bogdanova D. et. al. Detecting semantically equivalent questions in online user forums. *Association for Computational Linguistics*, 19:123–131, 2015.

[2] Kalenichenko D. Philbin J. Schroff, F. Facenet: A unified embedding for face recognition and clustering. *Google*, 2015.

[3] T. Addair. Duplicate question pair detection with deep learning. 2016.

[4] Sy S. Homma, Y. and C. Yeh. Detecting duplicate questions with deep learning. *Neural Information Processing Systems*, 30, 2016.

[5] Socher R. Manning C.D. Pennington, J. Glove: Global vectors for word representation.