Duplicate Question Pairs Detection

Akshay Sharma#1, Anoosha Sagar#2, Nikhil Thakur#3, Rahul Dhavalikar#4

#*University of California, Los Angeles*

[1]akshaysharma23@g.ucla.edu
[2]asagar06@g.ucla.edu
[3]nikhilt44@g.ucla.edu
[4]rdhavalikar@g.ucla.edu

***Abstract* - Determining whether two questions are asking the same thing can be challenging, as word choice and sentence structure can vary significantly. Traditional natural language processing techniques been found to have limited success in separating related question from duplicate questions. In this report, we explore methods of determining semantic equivalence between pairs of questions using a dataset released by Quora. We explore different approaches involving using different classifiers with a rich feature set, a Siamese Neural Network which uses an LSTM, and an ensemble of the multiple approaches. Our ensemble model outperforms the classifier and Siamese models.**

***Keywords* - Quora, Similarity Detection, Siamese Neural Network, XGBoost, Light GBM, Logistic Regression, Random Forest, Ensemble models**

I. Introduction

One of the most crucial tasks which can be achieved by effective natural language processing is the task of evaluating sentence similarity. Determining sentence similarity between a pair of sentences finds many applications such as automated short-answer grading and machine translation [7] to name a few. However, it has been gaining traction in many question-answer and knowledge-sharing applications such as Quora [15] and Reddit [16] as they gain popularity. As the user base of these applications grow, the magnitude of questions and answers in their archives also grows to the possibility that a question asked by a user has already been asked before and answered. In such a scenario, it would be ideal for the application to suggest previously asked questions similar to the one which the user wishes to ask in order to improve user experience and enable efficient knowledge-sharing. This would save waiting time for the user and reduce frustration which can occur if the user sees different versions of the same question on his feed thus preventing display of other, more relevant content. Additionally, ensuring that duplicate or similar questions are not asked repeatedly reduces the burden on the storage and processing infrastructure of the application. However, it is difficult to find similar questions due to limitations of current search engines. Document similarity techniques used in text retrieval approaches primarily depend on the degree of overlap. Extending these approaches to measuring question similarity is ineffective and inefficient because question pairs are small and will have very low overlap. Furthermore, due to the generative nature of the language, a question can be expressed in different ways which can cause additional problems [1]. Hence, having knowledge of alternate phrasings of a question can boost search and discovery techniques.

In this report, we present different approaches which can be used to determine the similarity between question pairs obtained from Quora. We define a question pair to be duplicate if they express the same intent. Our first approach makes use of existing classifiers such as XGBoost, Light GBM, Logistic Regression and Random Forests to classify whether a question pair is similar. We devise a rich feature set containing a variety of similarity based, graph based, and language based features which are fed into these classifiers to achieve impressive results. Secondly, we explore the Siamese Neural Network Architecture which makes use of a pair of LSTMs sharing the same weights [2] to perform the same task. The results achieved are good, but not at par with some of the other classifier results. Finally, we create an ensemble model from the above two approaches and measure question pair similarity.

This report is organized as follows. Firstly, we review some of the previous work done for measuring sentence similarity and describe some of the techniques we will be using in our approaches. Secondly, we present in detail our models and approaches. Finally, we report our results and observations.

II. Related Work

Detecting semantically equivalent sentences or questions has been a long-standing problem in natural language processing and understanding. As Dey et al. [4] demonstrate, traditional machine learning algorithms such as Support Vector Machines (SVMs) using hand-picked and heterogeneous features such as word overlap/ similarity, negation modeling, sentence/phrase

* *The implementation of our project can be found at:* *https://github.com/asagar06/quora-duplicate-question-pairs-detection*
** *The complete set of results can be found at:* *https://goo.gl/yczTAM*

composition [2] and extensively preprocessed data perform well on the SemEval-2015 dataset. Deep learning techniques in natural language processing have made considerable progress in recent years. Most deep learning methods for detecting semantic equivalence rely on a “Siamese” neural network architecture [2] that takes to two input sentences and encodes them individually using the same neural network.

Estimating lexical and syntactic similarity using lemma n-gram overlaps, POS n-gram overlaps, character n-gram overlaps, and TF-IDF scores along with semantic similarity using semantic composition, Paragraph2vec, word alignment have shown great promise in tackling the sentence similarity tasks presented in various SemEval competitions [10]. Some successful systems in these competitions are the ASOBEK [11] system which uses a SVM classifier with lexical word overlap and character n-gram features and the MITRE [12] system which uses a recurrent neural network with string matching features. Additional deep learning models using Recursive Neural Networks and Recurrent Neural Networks have been used extensively as well [7].

III. Dataset Details

For this project we make use of the publically available Quora Question Pairs dataset available on Kaggle [8][17]. For our experiments, we make use of the training set provided by the website. The training set consists of 4,04,290 labeled question pairs. The fields in the dataset are shown in TABLE I. Of all the question pairs, 149302 are duplicates, or roughly 37% of the full dataset. We assume that questions marked as duplicates in the Quora dataset are semantically equivalent since Quora’s duplicate question policy concurs with our definition of semantic equivalence above. The dataset has been labeled manually by humans. Hence there is bound to be some noise in the labels. We have split our data into three sets of training, validation, and testing sets. The training set has 3,04,290 entries whereas the validation and test sets contain 50,000 entries each.

TABLE 1
Field Description For Our Dataset

Fields	Description
id	unique identifier for the question pair
qid1	unique identifier for the first question
qid2	unique identifier for the second question
question1	full unicode text of the first question
question2	full unicode text of the second question
is_duplicate	1 if questions are duplicates, 0 otherwise

IV. Our Approaches

A. *Using Existing Classifiers with Hand-Crafted Features*

In order to detect sentence similarity, we first explore our dataset and extract a variety of features. We first convert each question into its embedding by averaging the word embedding for every word in the question. The pre-trained embeddings from word2vec [3] have been used for this purpose. Some of our features are then extracted from these embeddings obtained and some from the direct question itself.

The extracted features can be broadly categorized into Similarity Based Features, Graph Based Features, and Language Based Features. These extracted features are then fed into different classifiers and evaluated on the basis of various metrics such as accuracy, precision, recall, F1-score, ROC score, average precision-recall score, confusion matrix, and precision recall curve.

TABLE 2
Sample Data in Our Dataset

id	qid1	qid2	question1	question2	is_duplicate
447	895	896	What are natural numbers?	What is a least natural number?	0
1518	3037	3038	Which pizzas are the most popularly ordered pizzas on Domino’s menu?	How many calories does a Dominos pizza have?	0
3272	6542	6543	How do you start a bakery?	How can one start a bakery business?	1
3362	6722	6723	Should I learn python or Java first?	If I had to choose between learning Java and Python, what should I choose to learn first?	1

We use XGBoost [20], LightGBM [18], Random Forests [19], and Logistic Regression as our classifiers for performing the above task. Random forest classifiers are an ensemble learning method for classification that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes of the individual trees. XGBoost stands for eXtreme Gradient Boosting. Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. It builds the model in a stage-wise fashion like other boosting methods do, and it generalizes them by allowing optimization of an arbitrary differentiable loss function. LightGBM is a gradient boosting framework that uses tree based learning algorithms. It is designed to be distributed and efficient with the advantages of faster training speed, higher efficiency, lower memory usage, better accuracy, parallel and GPU learning support and capable of handling large-scale data.

1) *Similarity Based Features*

These features compute various similarity measures between the embedding vectors of the question pair. These measures help us measure how alike a question pair is. Some of the features extracted are Euclidean distance, Manhattan distance, Cosine similarity, Minkowski distance to name a few.

2) *Graph Based Features*

On analyzing our dataset, we observed that a question occurs many times across multiple question pairs in the data. Since the questions pairs were generated by a program based on pattern, we decided to generate a graph based on the dataset. We believe that these features can bring an improvement in the classification performance due to its ability to capture question co-occurrence. Each node of the graph represents a question and an edge between two nodes denotes that the question pair has occurred in the dataset. From this graph, we extract features such as degree of a node, common neighbors between nodes, and the PageRank [9] score of each node.

TABLE 3
COMPLETE LIST OF FEATURES EXTRACTED

Hand Crafted Features			
Similarity Based Features	**Graph Based Features**	**Language Based Features**	**Other Miscellaneous Features**
Jaccard Similarity	Common Neighbors	Common Bigram Count	Skew
Canberra Similarity	Degree	Common Named Entity Score	Kurtosis
City Block Similarity	PageRank Score	Common Noun Score	
Euclidean Similarity		Tf-IDF score	
Cosine Similarity		Sentiment Score	
Minkowski Similarity		Question Type	
Bray Curtis Similarity			
Word Mover's Distance			
Fuzzy Ratio			
Fuzzy Token Sort Ratio			
Fuzzy Partial Ratio		`	

3) *Language Based Features*

By using several popular NLP features, we wish to extract some syntactic and semantic information from our dataset which would be useful in classifying if a question pair is duplicate or not. These features have been computed using different NLP packages available. Features such as Tf-IDF scores, common bigram counts, question types, and common entities can collectively give a good indication of the syntactic and semantic information we seek.

The complete list of features used in our model is shown in TABLE 3.

B. *Siamese Neural Network Architecture*

We next implement a Siamese Neural Network Architecture. This model is inspired by Siamese twins who are twins which physically connected to each other from birth. A typical Siamese network consists of two LSTM layers with identical weights and parameters. Most deep learning methods for detecting semantic equivalence rely on a Siamese neural network architecture that takes to two input sentences and encodes them individually using the same neural network. A brief visualization is shown in Fig. 1.

Before we construct our neural network architecture and feed our input into it, we need to perform some preprocessing on our data. We first tokenize the sentences in the dataset using the nltk tokenizer[5]. The input questions vary significantly in length, from empty (0-length) up to 212 words. In order to batch our computations during training and evaluation using matrix operations, we needed the input questions to all have a fixed length. To do so, we padded the shorter sentences at the beginning with a designated zero-padding. We use the same split for training, validation, and testing sets as described in the Section III.

We begin by first creating embedding for each question using Word2Vec and then pass them through an LSTM layer one at a time. By this mechanism we obtain the encodings for each sentence. The error signal back propagated during training stems from the similarity between sentence encodings and how this predicted similarity deviates from the human annotated ground truth relatedness. The loss function to calculate the loss between the similarity function and the actual label, we used the Mean Squared Error (MSE) Loss. To compare the sentence representations, we used the same similarity function as suggested by [2] as below,

$$g(h_1, h_2) = exp\ (Manhattan\ Distance(h_1, h_2))$$

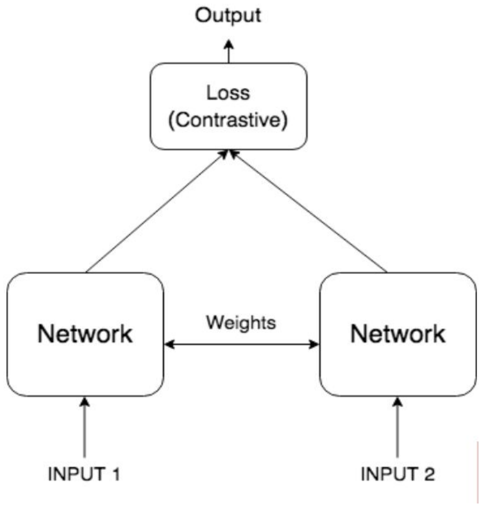


Fig. 1 Brief Visualization of Siamese Network.

This forces the LSTM to entirely capture the semantic differences during training. Our LSTM layers consist of 100 units. We used 100 units for the LSTM layer in our network as the computation cost increase by a huge amount since we have a lot of data as well. We used the Adam Optimizer for optimization. We implemented the Siamese architecture in PyTorch as well as Keras and found appreciable difference in performance which we will discuss in the following sections.

C. *Ensemble Model*

Finally, we create an ensemble model using the models obtained in the previous two approaches. Ensemble methods are meta-algorithms that combine several machine learning techniques into one predictive model in order to decrease variance(bagging), bias (boosting), or improve predictions (stacking). We use stacking in our problem. The base level models are trained based on a complete training set, then the meta-model is trained on the outputs of the base level model as features. The base level often consists of different learning algorithms and therefore stacking ensembles are often heterogeneous [14].

For our experiments, we created multiple ensemble models with different combinations of the previously obtained models. We took the predictions obtained from the base models and combined them using a fully connected neural network with Exponential Linear Unit (elu) activation, two hidden layers with 10 and 5 hidden units respectively. Our Ensemble model is shown in Fig. 2.

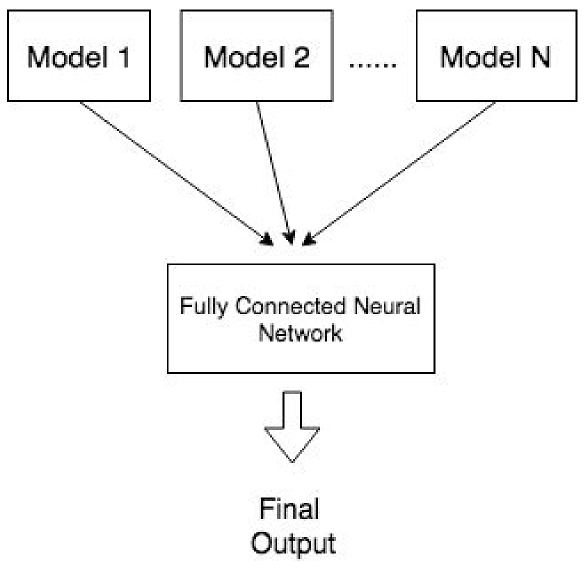


Fig. 2 Ensemble Model

V. Results

Our Light GBM model based on the handcrafted features achieved an accuracy of 87.78%. The XGBoost and Random Forest get slightly lower accuracies of 86.86% and 85.58% respectively. These models achieve a fairly good precision but lower recall values. Next, our implementation of Siamese classifier in PyTorch achieves an accuracy of maximum 75% but when we implemented the classifier in Keras, the maximum accuracy our model achieved was almost 83.05%. In our model ensembling approach, we experimented with multiple combinations of models. We experimented with different combinations of XGBoost (XGB), LightGBM (LGB), Random Forest (RF), Logistic Regression (LR) and Siamese LSTM network. The best accuracy of 88.93% was achieved for a combination of Siamese classifier and LightGBM. This combination model also achieves good performance on all metrics, especially an almost 5-10% increase in recall values. A combination of Siamese, LightGBM and XGBoost on the other hand gets a slightly lower accuracy of 88.75%. A combination with all 5 models gets a lower accuracy of 85.39% than the best model. A combination of 4 models without the Siamese model gives an accuracy of 84.49%.

The detailed results on different metrics for the various combinations of the models are listed in TABLE 4. The Figures 3-7 give a graphical comparison between the models. The confusion matrix and ROC curves are included in the APPENDIX section of our report.

TABLE 4
Results of Different Models and Ensembles

Model Type	**Model Details**	**Accuracy (%)**	**Recall (%)**	**Precision (%)**	**F1 Score**	**ROC AUC Score**	**Average Precision Score**
Classifier	XGBoost	86.86	73.59	87.088	0.7977	0.8383	0.92
	Light GBM	87.78	76.82	86.95	0.8157	0.8528	0.92
	Random Forest	85.58	74.68	83.46	0.788	0.833	0.9
Siamese Network	Siamese LSTM	83.05	70.65	78.98	0.7458	0.8022	0.83
Ensemble Methods	XGB+LGB	87.59	77.84	85.6	0.8154	0.8536	0.92
	XGB+ LGB+ RF+LR	84.49	79.81	76.98	0.7837	0.8342	0.87
	Siamese+XGB+LGB+LR+RF	85.398	84.97	76.26	0.804	0.853	0.89
	Siamese+XGB+LGB	88.752	83.148	84.638	0.8388	0.8747	0.93
	Siamese+RF	87.31	83.39	81.1	0.8223	0.864	0.92
	Siamese+LGB	88.93	81.83	86.05	0.8388	0.8731	0.92

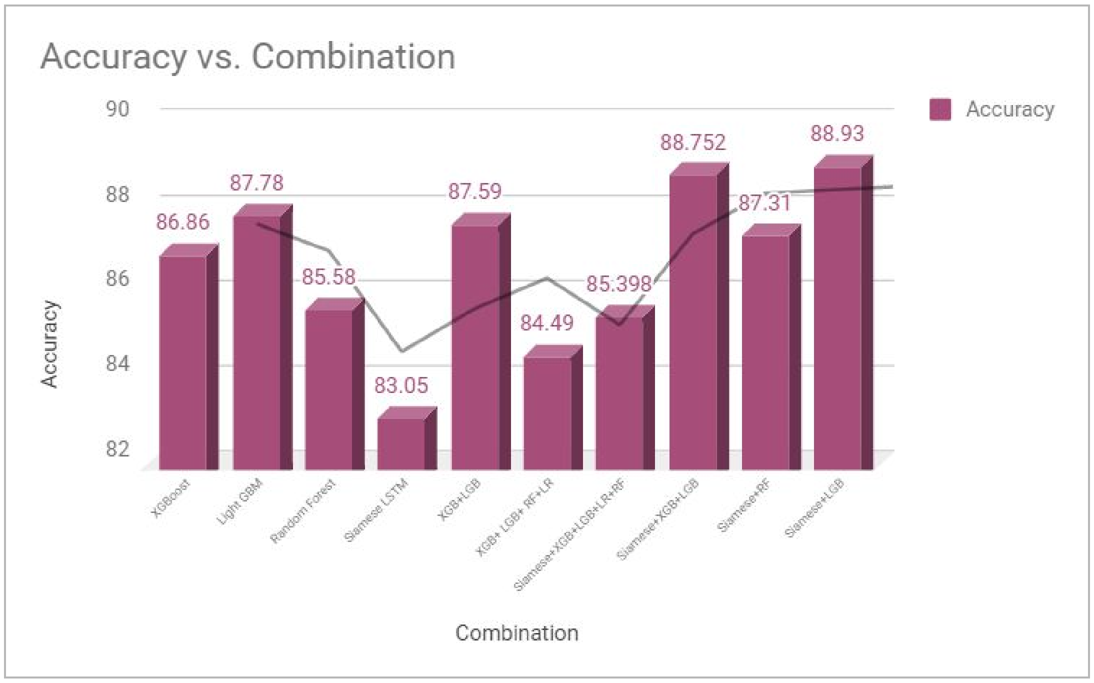


Fig. 3 Accuracy of Our Models

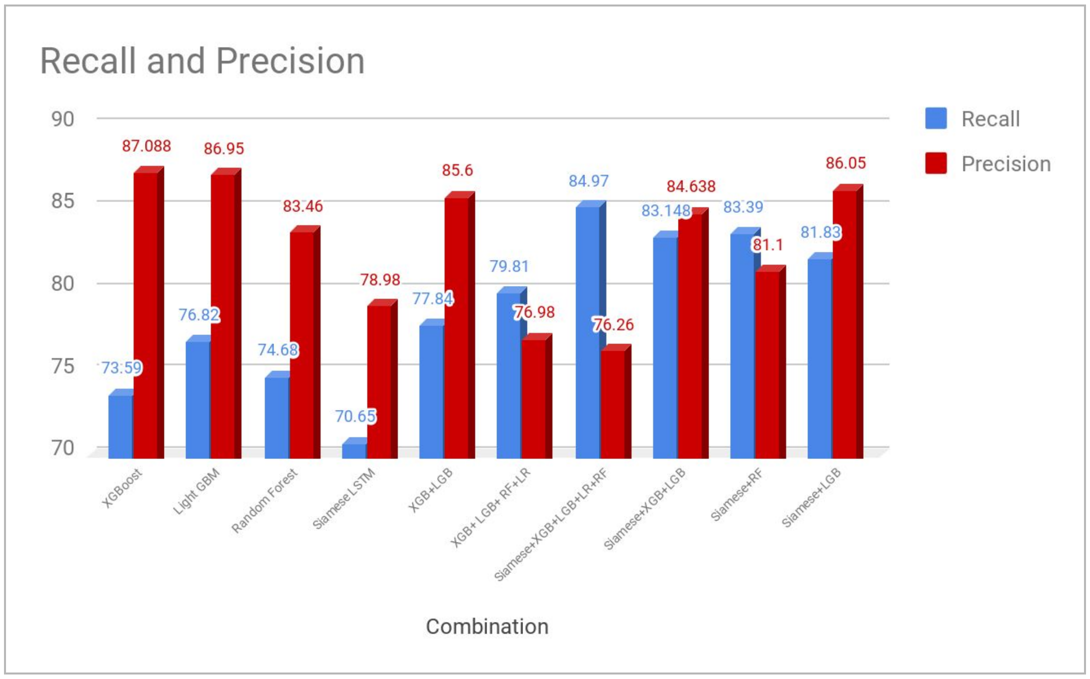


Fig. 4 Recall and Precision of our Models

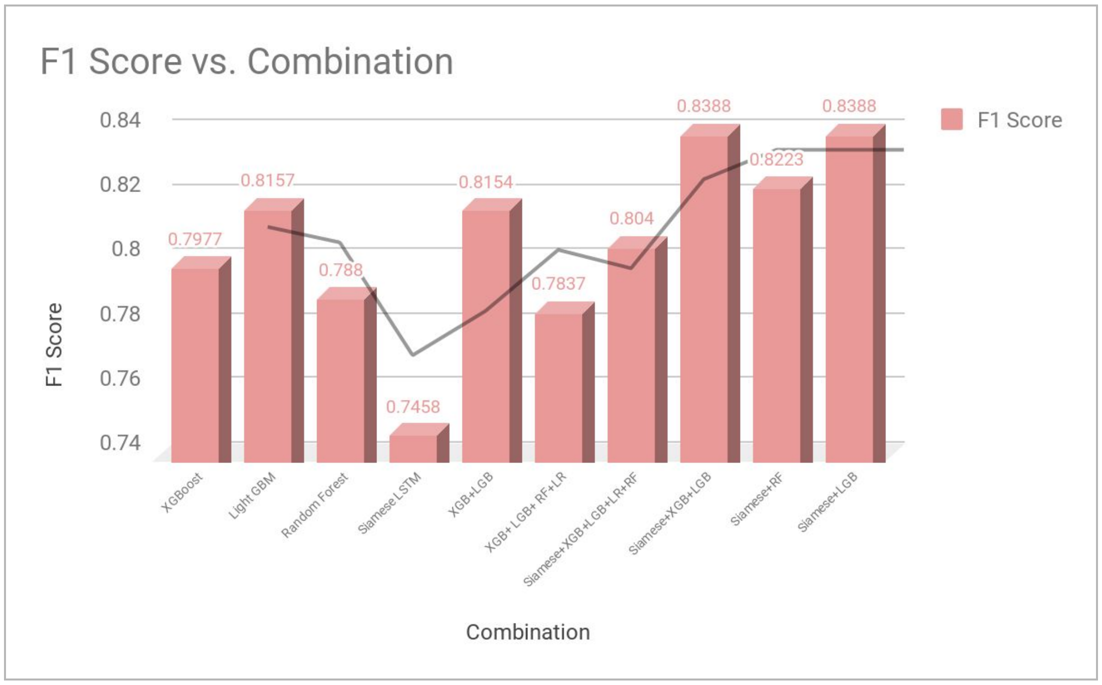


Fig. 5 F1 Score of our Models

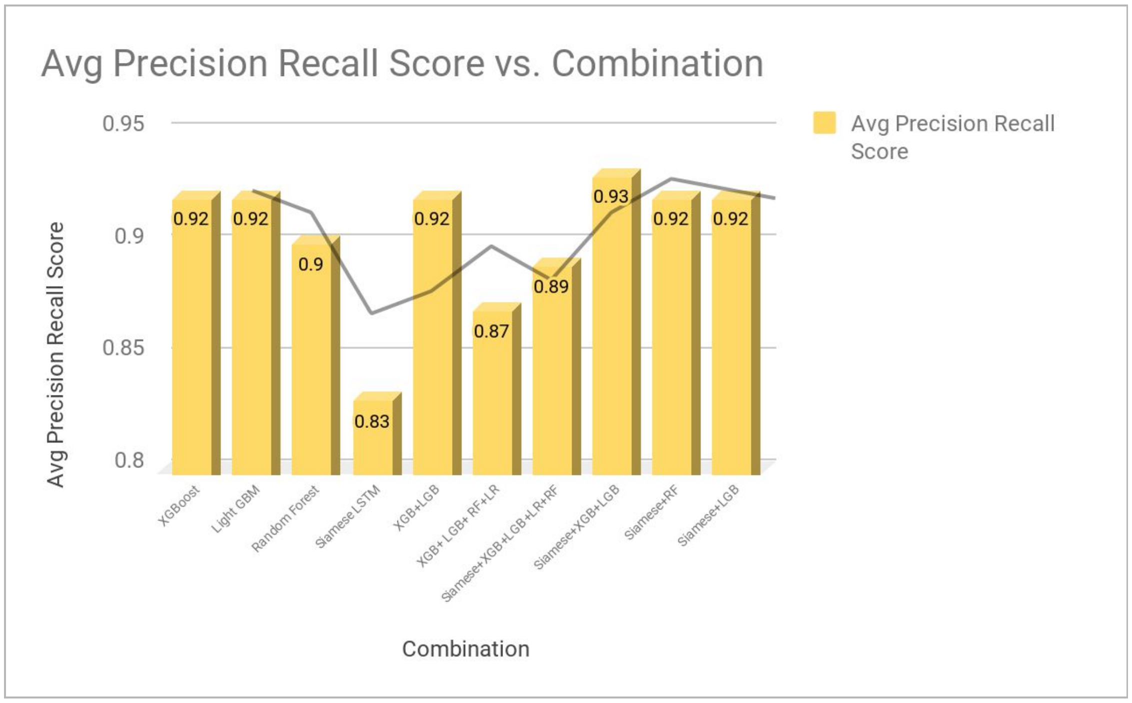


Fig. 6 Average Precision Recall Score of Our Models

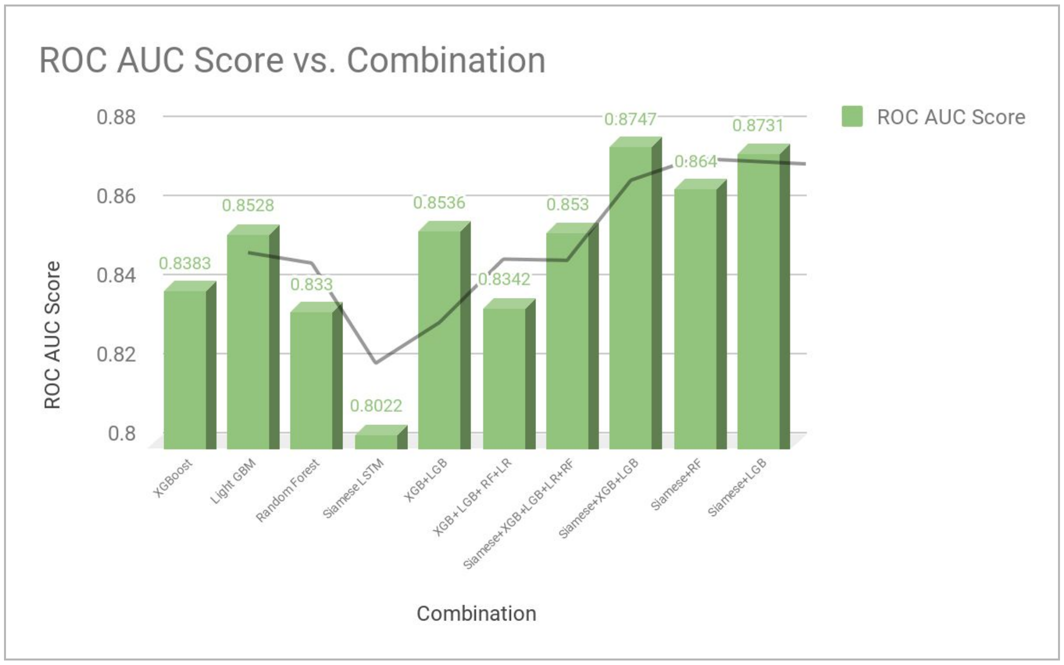


Fig. 7 ROC AUC Scores of Our Models

VI. Observations

On using multiple classifiers with our rich feature set which includes similarity, graph and language based features, we observed that advanced classifiers like XGBoost and LightGBM outperform traditional classifiers like Random Forest and Logistic. Some of these hand-crafted features go a long way in improving the accuracy of our models, particularly graph based features. Our Siamese model gives a good accuracy but is outperformed by classifiers using hand-crafted features. We attribute this to the fact that there are a considerable number of questions which greater than 50 words and LSTM's are not able to capture the meaning of longer sentences, as seen in Fig. 10.

We tried to learn our own embeddings from random initialization and tried to fit word2vec embeddings to our dataset, but both of these experiments resulted in overfitting as our training set is small. Our final Siamese implementation performed well on our dataset and did not overfit the training examples as evident in the Fig. 8 and Fig. 9. We also observed that we got a better result with Keras implementation of Siamese network as opposed to Pytorch implementation which may be attributed to superior optimization and hyperparameters used in the Keras library.

Finally, we observe that some of our ensemble models tend to outperform both traditional and advanced classifiers as well as the Siamese model. We get the best accuracy from our best stacked ensemble models which outperform each of the individual models due its smoothing nature and ability to highlight each base model where it performs best and discredit each base model where it performs poorly. For this reason, stacking is most effective when the base models are significantly different.

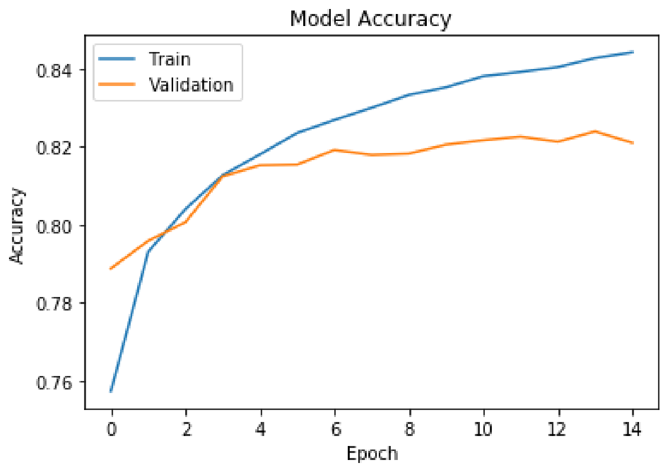


Fig. 8 Plot of Accuracy vs Epoch for the Siamese Network

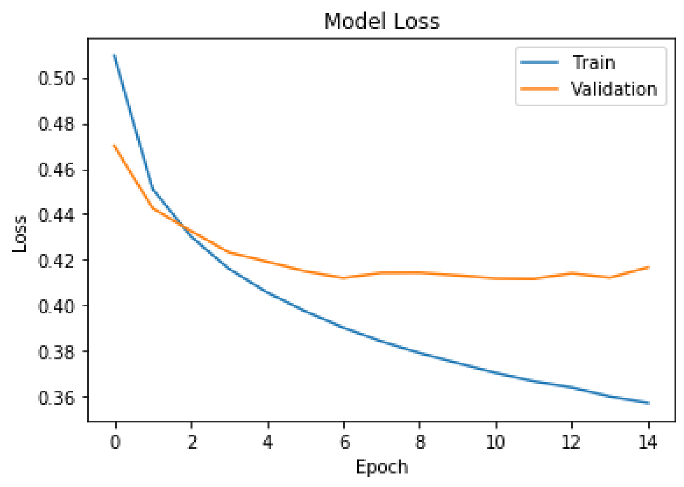


Fig. 9 Plot of Loss vs Epoch for the Siamese Network

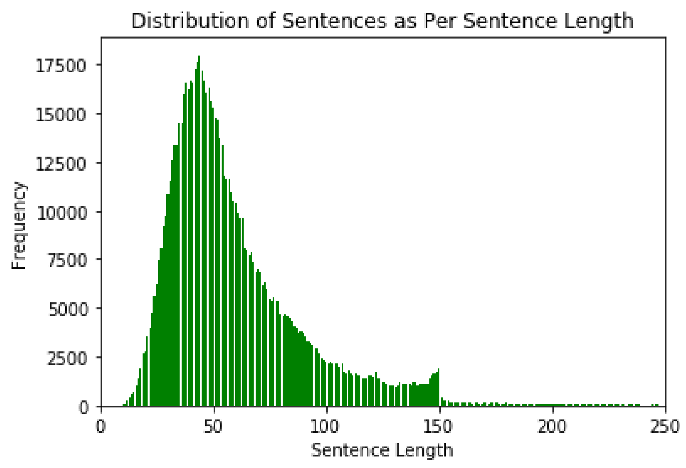


Fig. 10 Plot of Distribution of Sentences as Sentence Length

VII. Future Work

Our best ensemble models outperform all traditional and advanced classifiers such as XGBoost, Random Forest, Logistic and LightGBM, and even the Siamese architecture. This work can be extended to include data augmentation to further improve the balance of data. The Siamese architecture can also be tried out with Gated Recurrent Units (GRU) instead of LSTM. Some of the recent papers in NLP have shown promising results using ELMo (Embeddings from Language Models), but it comes with a very high computation cost. We think that using ELMo [13] for this task will give a higher accuracy as ELMo builds word embeddings using the context from the dataset. Another possible extension to our work may be to use padding on each batch instead of doing it on entire dataset or maybe try out some different method of dealing with variable length sequences.

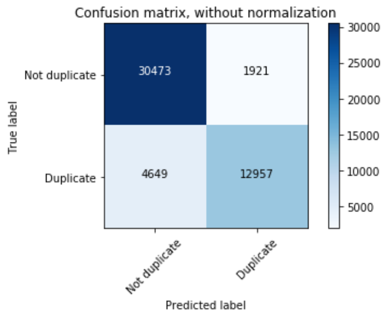
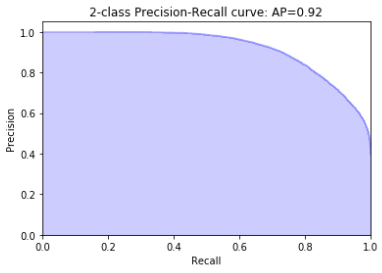
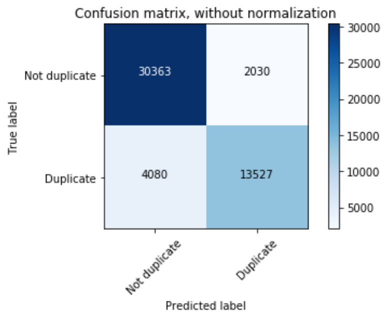
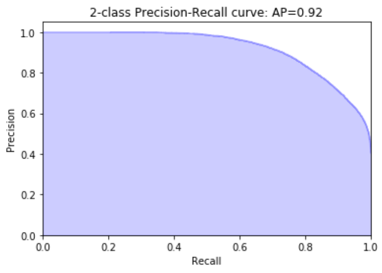
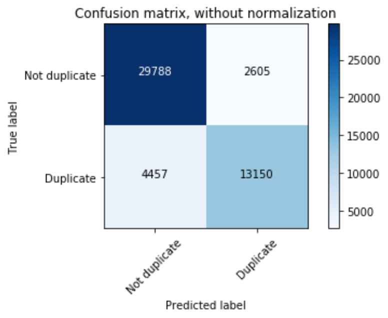
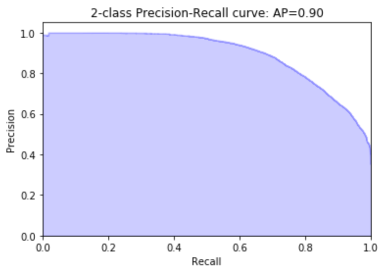
VIII. Conclusion

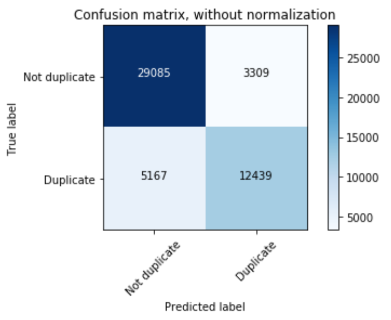
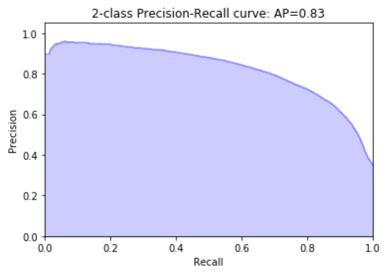
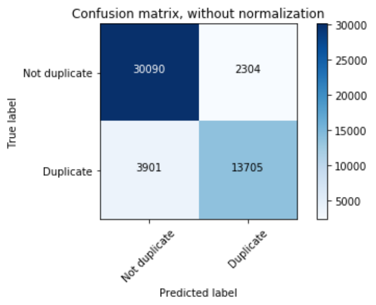
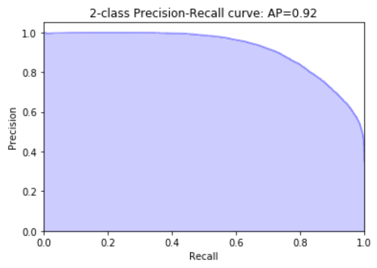
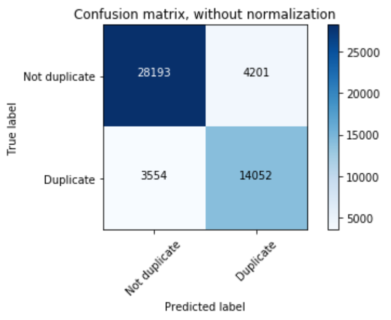
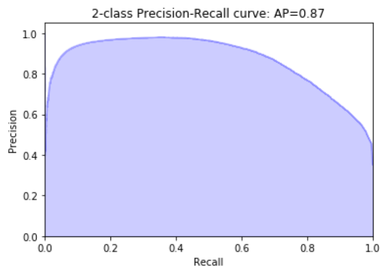
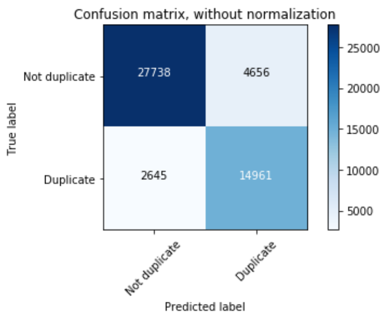
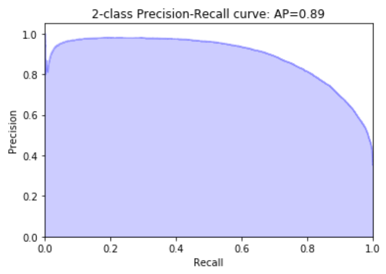
In this project, we have attempted to provide solutions for measuring semantic textual similarity and applied it to the question pair dataset provided by Quora and Kaggle. We explore three approaches for determining text similarity, namely classifiers using rich hand-crafted features, Siamese neural network architecture, and ensemble models. Our results are promising, with the ensemble model consisting of the Siamese network and Light GBM classifier being the top performer. Interestingly, all our classifiers outperform the Siamese network demonstrating the strength of our features chosen and the negative impact of long sentence lengths on LSTMs which are at the heart of the Siamese network.

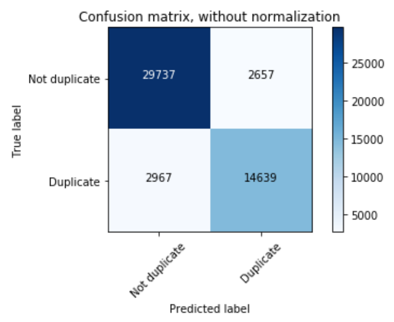
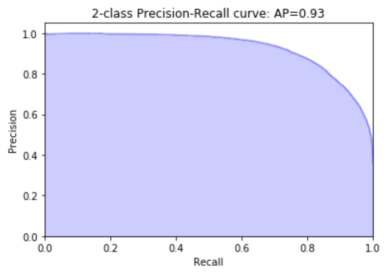
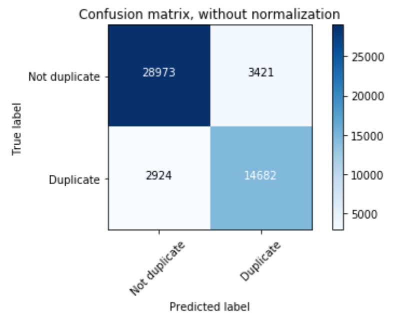
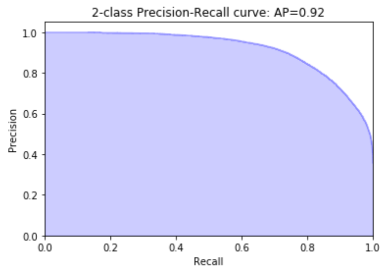
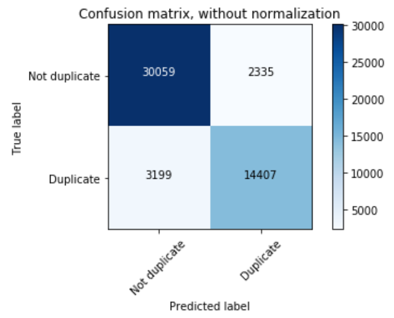
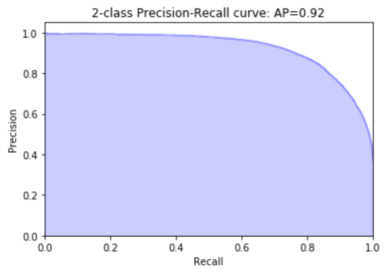
Appendix

In this section, we report the confusion matrices and precision recall curves for our models.

TABLE 5
Confusion Matrices and Precision Recall Curves for our Models

Model Type	Model Details	Confusion Matrix	Precision Recall Curve
Classifier	XGBoost		
	Light GBM		
	Random Forest		

Siamese Network	Siamese LSTM		
Ensemble Methods	XGB +LGB		
	XGB + LGB + RF + LR		
	Siamese + XGB + LGB + LR + RF		

	Siamese + XGB + LGB		
	Siamese + RF		
	Siamese + LGB		

References

[1] Achananuparp, Palakorn, et al. "Utilizing sentence similarity and question type similarity to response to similar questions in knowledge-sharing community." *Proceedings of QAWeb 2008 Workshop*. Vol. 214. 2008.

[2] Mueller, Jonas, and Aditya Thyagarajan. "Siamese Recurrent Architectures for Learning Sentence Similarity." *AAAI*. 2016.

[3] Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." *arXiv preprint arXiv:1301.3781* (2013)..

[4] Dey, Kuntal, Ritvik Shrivastava, and Saroj Kaushik. "A paraphrase and semantic similarity detection system for user generated short-text content on microblogs." *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. 2016.

[5] Bird, Steven, and Edward Loper. "NLTK: the natural language toolkit." *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*. Association for Computational Linguistics, 2004.

[6] Achananuparp, Palakorn, Xiaohua Hu, and Xiajiong Shen. "The evaluation of sentence similarity measures." *International Conference on data warehousing and knowledge discovery*. Springer, Berlin, Heidelberg, 2008.

[7] Sanborn, Adrian, and Jacek Skryzalin. "Deep learning for semantic similarity." *CS224d: Deep Learning for Natural Language Processing. Stanford, CA, USA: Stanford University*(2015)

[8] *Quora Question Pairs | Kaggle* [Online] Available: https://www.kaggle.com/c/quora-question-pairs

[9] Langville, Amy N., and Carl D. Meyer. *Google's PageRank and beyond: The science of search engine rankings*. Princeton University Press, 2011.

[10] Brychcín, Tomáš, and Lukáš Svoboda. "UWB at SemEval-2016 Task 1: Semantic textual similarity using lexical, syntactic, and semantic information." *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*. 2016.

[11] Eyecioglu, A. and Keller, B. (2015). ”*ASOBEK: Twitter paraphrase identification with simple overlap features and SVMs*.” In Proceedings of SemEval.

[12] Zarrella, G., Henderson, J., Merkhofer, E. M., and Strickhart, L. (2015). ”*MITRE: Seven systems for semantic similarity in tweets.*” In Proceedings of SemEval.

[13] Peters, Matthew E., et al. "*Deep contextualized word representations.*" arXiv preprint arXiv:1802.05365 (2018).

[14] *Ensemble learning to improve machine learning results* [Online] Available: https://blog.statsbot.co/ensemble-learning-d1dcd548e936

[15] *Home - Quora* [Online] Available: https://www.quora.com/

[16] *reddit: the front page of the internet* [Online] Available: https://www.reddit.com/

[17] *Kaggle: Your Home for Data Science* [Online] Available: https://www.kaggle.com/

[18] Ke, Guolin, et al. "LightGBM: A highly efficient gradient boosting decision tree." Advances in Neural Information Processing Systems. 2017.

[19] Ho, Tin Kam. "Random decision forests." Document analysis and recognition, 1995., proceedings of the third international conference on. Vol. 1. IEEE, 1995.

[20] Chen, Tianqi, and Carlos Guestrin. "Xgboost: A scalable tree boosting system." Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. ACM, 2016.