Redundant Questions Classification Based on Glove, XGBoost and Hyper Parameter Tuning

Suraj Kumar Jalapally[1] Subhasri Rallabandi[2] Sacheth Reddy Mamidi[3] Srikanth R[4]
[1,2,3,4]Chaitanya Bharathi Institute of Technology, Telangana, India

Abstract— A plethora of machine learning models have been put forth to model the sentence patterns accurately, which requires many features, parameters, and considerable computational ability. However, there was no extensive scrutiny in the number of relevant features, which compounds the model's ability for classification. This paper has explored four different models to achieve a robust system to classify the duplicate question pair from the raw dataset published by Quora. All the challenges posed by the dataset, like data imbalance, unprocessed data, are addressed in this system. From the four models, the XGBoost model advocating different feature sets and Hyper parametric tuning helped us achieve the following best results: Accuracy of 84.2%, Precision 83.35%, Recall 82.75%, F1 score 83%.

Keywords: Quora Question Pair, XGBoost, Hyper Parameter Tuning

I. INTRODUCTION

Comprehending the context (logical and lexical semantics) of queries posted on a forum provides essential insight. On a forum, questions linked to its cascading answers occupy a certain amount of space; a duplicate question would inherently consume just as many resources due to its own thread. Hence classifying a query as "original" or "duplicate" becomes extremely paramount and helps blot out redundant queries before being populated in the forum's repository.

Aiming to recreate the most probable usage instances, we used data that contains minute differences between the negative queries; the positive queries use completely different phrasing. Namely, "What do I need to learn to start my own advertising agency," which shares several similar characteristics with "How can I start my own advertising agency?" refers to a completely different context. While on the other hand, "How can I make my money make money?" has a different composition than "How can I earn more money?" but the original intent of the query is the same.

To address this conundrum of classifying queries as "duplicate" or "original," we have opted to use the XGBoost along with the Hyperparameter tuning approach. After the advanced feature extraction, we put them through the proposed model and have achieved an F1 score of 83% and an accuracy of 84.2%.

II. RELATED WORK

The Internet is a vast repository to gain and share knowledge. Some of the platforms where Internet users seek knowledge are Quora, Stack Overflow. An inquisitive user posts a question on these platforms while interested/concerned people contribute the answers. Nearly billions of users visit these platforms to contribute and seek knowledge by posting questions and answering questions. The common problem we face on these platforms is that many users ask similar questions. Multiple questions with the same objective will be ambiguous for users who seek knowledge, while at the same time, it is redundant work for writers to post the same answers multiple times.

In the work carried out by E Dadashov et.al (2017) [1], they have worked on a Quora duplicate questionnaire problem using Siamese with LSTM. Euclidean distance between the two sentences is considered as the feature for duplicate question identification. The second layer takes an input of the last output states of layer one; duplicates are identified with an F1 score of 79.5% for their proposed model.

Another work in this area on Quora dataset was carried on the application of the different Natural Language Processing models [2] on the given dataset and compared with different Machine learning models; as part of this work, for prediction of similarity they have worked on KNN, Random Forest, Decision Tree, Extra Trees, XGBoost and AdaBoost. However, the drawback in this system is that the features identified and given for the machine learning model are not that accurate and relevant.

Furthermore, the previous work from [1,2,3,4,7] has built neural network models like SWEM, LSTM, BERT, where they produced a good recall score. Their overall accuracy sums up to 82-83%, and F1 scores are below 80%. But here, we observed the Precision score is low and which is affecting the F1 score in a large magnitude.

The work proposed in this paper works in the identification of duplicate questions and matches the similar question already answered by extracting relevant features and training the model to get better accuracy and F1 score. Here we focus on building a robust model where precision and recall have the same impact over the F1 score.

III. DATASET

Quora hosts its dataset on S3; the dataset consists of over 400k question duplicate pairs. Here the challenge was to predict the pair of questions with the same meaning. Human experts give the ground-truth label; this interpretation can be subjective as the innate meaning of a sentence cannot be perfectly classified. The ground truth is not defined with complete accuracy; rather, they are estimates.

The Data fields here are "id" - which is the id of the question pair training set, "qid1", "qid2". Every question is given a Distinct id. question1 and question2 represent the complete sentence of the respective question. "is duplicate" represents the ground-truth value which is either 0 or 1. 1 represents that both the questions lead to the same meaning while 0 denotes that they are different.

It is an imbalanced dataset where duplicate data points are vastly greater than non-duplicate data points. The distribution of questions in the dataset does not directly imply the questions which were asked on Quora. All the provided Data source is structured labelled data containing some noise which is handled in the Data pre-processing phase.

IV. DATA PREPARATION

Pre-processing of the data is imperative as it brings out valuable, analysable, and predictable information from the raw text. The dataset contains HTML tags, noise, insignificant characters, and even characters not in ASCII code. So we have cleaned up the data by removing HTML tags, stop words, clearing the punctuations, replacing extended words with its contracted form of words, for instance, replacing "I would" with "I'd," "they will" with "they'll" and performing Stemming (removing suffix and converting to its original stem form, i.e., base or root form, this method is an integral part of information retrieval). All these pre-processing steps do extract productive information from the data. The total number of words will reduce after pre-processing the data. Even with reduced text size, we could get the same performance for both linear and non-linear models. It shows that these pre-processing methods are effective.

V. FEATURE EXTRACTION

The extracted features from the data are organized into three feature sets, Rudimentary feature set, a Complex feature set which includes Fuzzy-Wuzzy feature set and word2vec features by taking TF-IDF weights. We have extracted a total 794 features from above feature sets.

In the Rudimentary feature set we have considered, Frequencies of qid1 and qid2 it shows how often the question are been asked, addition and absolute difference between qid1 and qid2, Length of question, Unique common words, Total number of words, Similar words. Probability Distribution Function (PDF) is applied for every feature presented here and has some significance in classifying the dataset if it is duplicate or not. After applying the PDF for Similar word feature it showed significant separability that denotes it has better classifying potential in this Rudimentary feature set. This feature set contains 11 unique features.

In the Complex feature set, the features we considered are maximum and minimum number of common stop tokens, common words, token length. Additionally, we have considered the first and last words of the qid1 and qid2, absolute difference in the length between the two question pairs, mean token length. Further we extended, applying fuzzy features [5] such as highest substring ratio, fuzzy ratio and partial fuzzy ratio, token set and token sort ratio. This feature set contains 15 different features which shows separability and are effective features which are cardinal in classification. For analysing these multidimensional feature-space we used t-distribution stochastic neighbourhood embedding as shown in fig [1] which gives understanding of how these features advocate in classification.

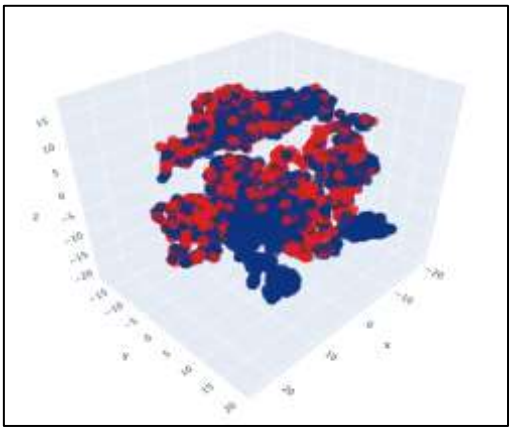

Fig. 1: TSNE Distribution

In the word2vec feature set, we calculated scores of Term Frequency Inverse Document Frequency (TFIDF) then converted those words into vectors by using TFIDF weights which gives better results than using only average word2vec vectors. Here we used the GLOVE model [6] which is present in the Spacy library, and it is a pre-trained sophisticated model. This model uses Wikipedia for training as it is sturdy in its extensive word semantics.

For qid1 and qid 2, extracted 384 features for each qid. Hence, combining all the feature sets we get 794 features. Now we added these features to the loaded dataset.

VI. PARAMETERS

A. Log Loss

Log loss is a metric where a classification is done using Likelihood, where we get a probabilistic value. Here we considered log loss over mean square error as in non-linear models we might have many convex or concave curves where we have many minima's so the gradient descent will find it difficult to find the global minima. Hence, we have considered Log loss as the metric which is good for non-linear models as well. We use this metric to compare the performance between different models. To compare the performance among different models we used this metric.

$$H_p(x) = -\frac{1}{N}\sum_{i=1}^{N} yi.log\ log\ \big(p(yi)\big) + (1-yi).\ log\ log\ \big(1-p(yi)\big) \quad [I]$$

here $H_p(x)$ - Log Loss

B. Precision, Recall, F-Score

The ratio of the accurately predicted positive occurrences to the total positive observations predicted is known as Precision. High Precision signifies a low misclassification rate. Recall is used when the cost of false negatives is high. F_1 score denotes the Harmonic Mean of Recall and Precision, where both measures are given equal weights.

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} \quad [II]$$

$$Recall = \frac{True\ Postives}{True\ Positives + False\ Negatives} \quad [III]$$

$$F_1\ Score = \frac{2*Precision*Recall}{Precision+Recall} \quad [IV]$$

VII. MODEL

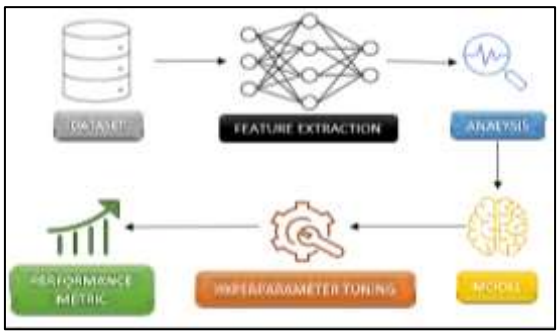

Fig. 2: Model

We have divided the whole modelling system into six phases, where we process the data and filter out unwanted information. Then we have extracted features into three different sets: Rudimentary, Complex, TFIDF feature sets. Then an exploratory analysis is done through Pair plots, Probability Distribution Function, and plotting TSNE, where we could find the relevancy of features. We compared different non-linear machine learning models. Incremental appending of feature set to model is done to report the performance at each phase. Hyperparameter tuning is applied to tune the parameters for better performance and measured with relevant metrics.

A. Linear SVM

As the data showed some linearly separable characteristics, we have applied Linear Support Vector Machines. In the first phase we applied with rudimentary, and complex features the log loss obtained is 0.4032. Which is comparatively better than Logistic Regression. Further we went on applying Grid and Random search where the log loss came out to be 0.3619.

Alpha	Log Loss
0.00000661	0.370714685
0.0000082	0.366990679
0.00000832	0.374065859
0.00001667	0.367526936
0.00001683	0.372955536
0.00001745	0.370591123
0.00001769	0.3619
0.00001943	0.36547605
0.00002237	0.375131304
0.00002323	0.376091065
0.00002555	0.378360112
0.00002707	0.368486648
0.00002739	0.372634488
0.00002745	0.37240519

Table 1: Linear SVM

For alpha(α) at $1.768e^{-05}$, we got the best log loss of 0.3619; the Accuracy and F1 score obtained are 82.6% and 81.06%, respectively.

B. Logistic Regression

As the number of dimensions, we extracted is moderate, we applied Logistic Regression as in the analysis phase the PDF's and pair plots have shown a significant separability. After applying the rudimentary feature set and Complex feature set we got a log loss of 0.4004. Further, added Tf-Idf features and then tuned the parameter by hyper parameter tuning with Grid and Random search the log loss has improved to 0.3582.

Alpha	Log Loss
0.00000413	0.361228668
0.00000547	0.358879025
0.00000773	0.3582
0.00000884	0.358886412
0.00000938	0.35901257
0.00001143	0.359949314
0.00001583	0.36249079
0.00001628	0.362684456
0.00001782	0.363574546
0.00001914	0.364198067
0.00002192	0.365549212
0.00002659	0.367434468
0.0000269	0.3675203
0.00002832	0.368070798

Table 2: Logistic Regression

We got the best hyper-parameter when alpha (α) is $7.73e^{-06}$; this model has a high recall for class 0 but low for class 1. The prediction ability of class 0 is greater than class 1. The accuracy achieved in this model is 82.6%, Precision 81.8%, Recall 80.5%, F1 score 81.14%.

C. XGBoost

As we have seen in the TSNE visualization that the separability of data points is significant. So, for non-linear modelling, we have used Random Forest and XGBoost. For Random Forest, we have used a Rudimentary and complex feature set; it gave 0.4203, which was higher than Logistic Regression and Linear SVM. As it was underperforming, we did not proceed further. As the dimensions are not very high, we considered Gradient Boosting Decision trees. Furthermore, we have considered XGBoost over AdaBoost because, in our feature set, we might have some irrelevant features; in that case, AdaBoost becomes slow as it is not optimized for its agility and underperforms comparatively to XGBoost.

Extreme gradient boosting (XGBoost) is an ensemble technique where each tree boosts the attributes that led to misclassification of the previous tree. Here we applied Rudimentary and Complex Feature sets; the log-loss obtained is 0.363. It gave better results than the Linear models. This shows that the data is complex, and a non-linear model shows better separability than Logistic Regression and Linear SVM. Further, we have added word2vec TF-IDF features the number of estimators was 500, learning rate 0.1 by these parameters for XGBoost over test data the log loss obtained is 0.312, which is a significant improvement. To make the model more robust, we have used Distance Features. From pre-trained glove word vectors, we obtained Word Movers Distance (WMD). Using the model, we obtained the average word vector for the qid1 and qid2. From that average word vector model, we obtained the following distances:

- Hamming distance (shows in how many bit positions they differ).
- Levenshtein (shows the measure of the distance between two sequences).
- Cosine distance (denotes higher the similarity lesser the distance)

– Canberra distance (gives an intuition on the distance between data points in the vector space).

Further, we have added Distance feature sets along with Rudimentary, Complex, and Word2Vec feature sets, and we have done hyperparameter tuning by Grid and Random search. The results are as follows:

Estimators	Log Loss
100	0.354347334
150	0.342696034
200	0.335940615
300	0.327201854
400	0.322281726
600	0.316831569
800	0.312

Table 3: XGBoost

XGBoost model achieved 0.312 log loss, 84.20 accuracy%, 83.35% precision, 82.75% recall, and 83% F1 Score where the learning rate parameter 0.12 number of estimators is 800, minimum child weight parameter is 5, maximum depth is 3, alpha 150 and lambda 350. These parameters helped in achieving the best result.

Model	Work	Accuracy	Precision	Recall	F1 Score
Logistic Regression	Explored model	82.60	81.80	80.50	81.14
Linear SVM	Explored model	82.60	81.90	80.25	81.06
XGBoost	Proposed	84.20	83.35	82.75	83.00
Siamese with LSTM	E Dadashov et.al[1]	83.20	73.00	86.80	79.30
Ensemble	E Dadashov et.al[1]	83.80	70.20	83.70	76.40
CBOW	S Lakshay et.al[2]	83.40	-	-	77.80
BiLSTM + Attention	S Lakshay et.al[2]	82.30	-	-	76.40
SWEMConcat	S Dinghan et.al[3]	83.03	-	-	-
BERTlarge	I Peter et.al[4]	72.00	-	-	-

Table 4: Comparison Chart with Previous Work

VIII. Results and Discussion

Although it is an imbalanced dataset, we built a robust model consisting of XGBoost and different feature sets with hyperparameter tuning. Since the dimensions (features) are not more than 1000, extreme gradient boosting trees did not show any overfitting but rather resulted in a robust model. As shown in Table – 4, this model outperformed other machine learning models, giving us a benchmarking F1 score.

IX. Conclusion

Upon working on the Quora Question pair problem we have concluded that our results which use XGBoost with feature extraction and hyper parameter tuning have a competitive advantage over models used previously to address similar problems. Considering time and hardware constraints, we reached 84.2% accuracy, 83.35% Precision, 82.75% Recall and 83% F1-score. Our model has outperformed other models like CBOW, Siamese with LSTM, $BERT_{large}$, $SWEM_{concat}$. By eliminating hardware and time constraints, compounding the existing model with sophisticated deep learning models, we can achieve better results.

References

[1] Dadashov, Elkhan, Sukolsak Saksuwong, and Katherine Yu. "Quora question duplication." (2017).
[2] Lakshay Sharma, Laura Graesser, Nikita Nangia, and Utku Evci, "Natural language understanding with the quora question pairs dataset." arXiv preprint arXiv:1907.01041 (2019).
[3] Dinghan Shen, Guoyin Wang, Wenlin Wang, Martin Renqiang Min, Qinliang Su, Yizhe Zhang, Chunyuan Li, Ricardo Henao, and Lawrence Carin, "Baseline needs more love: On simple word-embedding-based models and associated pooling mechanisms." arXiv preprint arXiv:1805.09843 (2018).
[4] Peter Izsak, Moshe Berchansky, and Omer Levy. "How to Train BERT with an Academic Budget." arXiv preprint arXiv:2104.07705 (2021).
[5] Bosker, H.R. Using fuzzy string matching for automated assessment of listener transcripts in speech intelligibility studies Behav Res (2021). https://doi.org/10.3758/s13428-021-01542-4
[6] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. "Glove: Global vectors for word representation." Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014.
[7] Shashank Pathak, Ayush Sharma, Shashank Shekhar Shukla, (2018) "Semantic String Similarity for Quora Question Pairs", International Journal of Advances in Science, Engineering and Technology (IJASEAT), pp. 77-80, Volume-6, Issue-4

