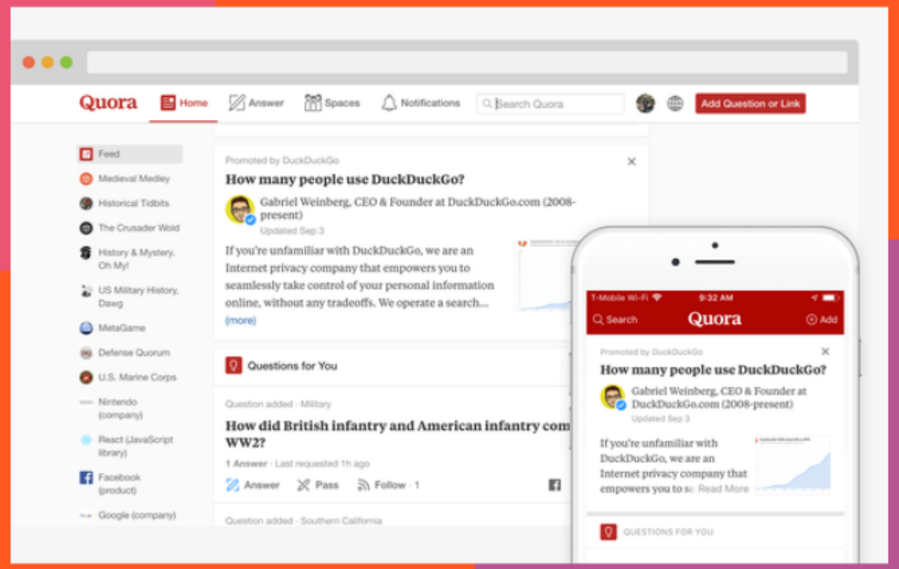
Akash H Rao




# Quora Question Pair Similarity Problem

Machine Learning in Addressing Quora Question Pair Similarity Problem
by Akash Rao

Problem Analysis

# Understanding the Problem Given

Enhancing Semantic Similarity Detection for Quora Questions

## What is Quora?

Quora is a social media platform where users ask questions and provide answers.

## What issue is it solving?

One issue in Quora is the existence of several questions with the same answers but phrased differently.

## Machine Learning and NLP Application

We will deploy machine learning and NLP techniques to train a model on the provided dataset.

## Flagging Question Matching

After creating the model, we will flag matching question pairs in the text dataset.

User Experience Enhancement

# Business Impact of Solving the Problem

Driving Success Through Enhanced User Experience

## Enhanced user experience

## Improved content management

## Increased engagement

## Efficient knowledge discovery

Language Complexity

# Challenges in Identifying Similar Questions

These are the initial challenges we might face as we proceed with the project after reviewing the data.

## 1 Unclear in language and typos

There were many questions that used ambiguous language, along with minor grammar and spelling mistakes.

## 2 Contextual differences

Pairs that look very different at a glance may be the same, whereas some pairs that look similar may be different.

## 3 Minor Variation

Stopwords and prepositions play a key role, with small details affecting the context

## 4 Large Dataset

Both the training and testing datasets are very large, so we must consider the time and complexity, which will make it challenging

Techniques Overview

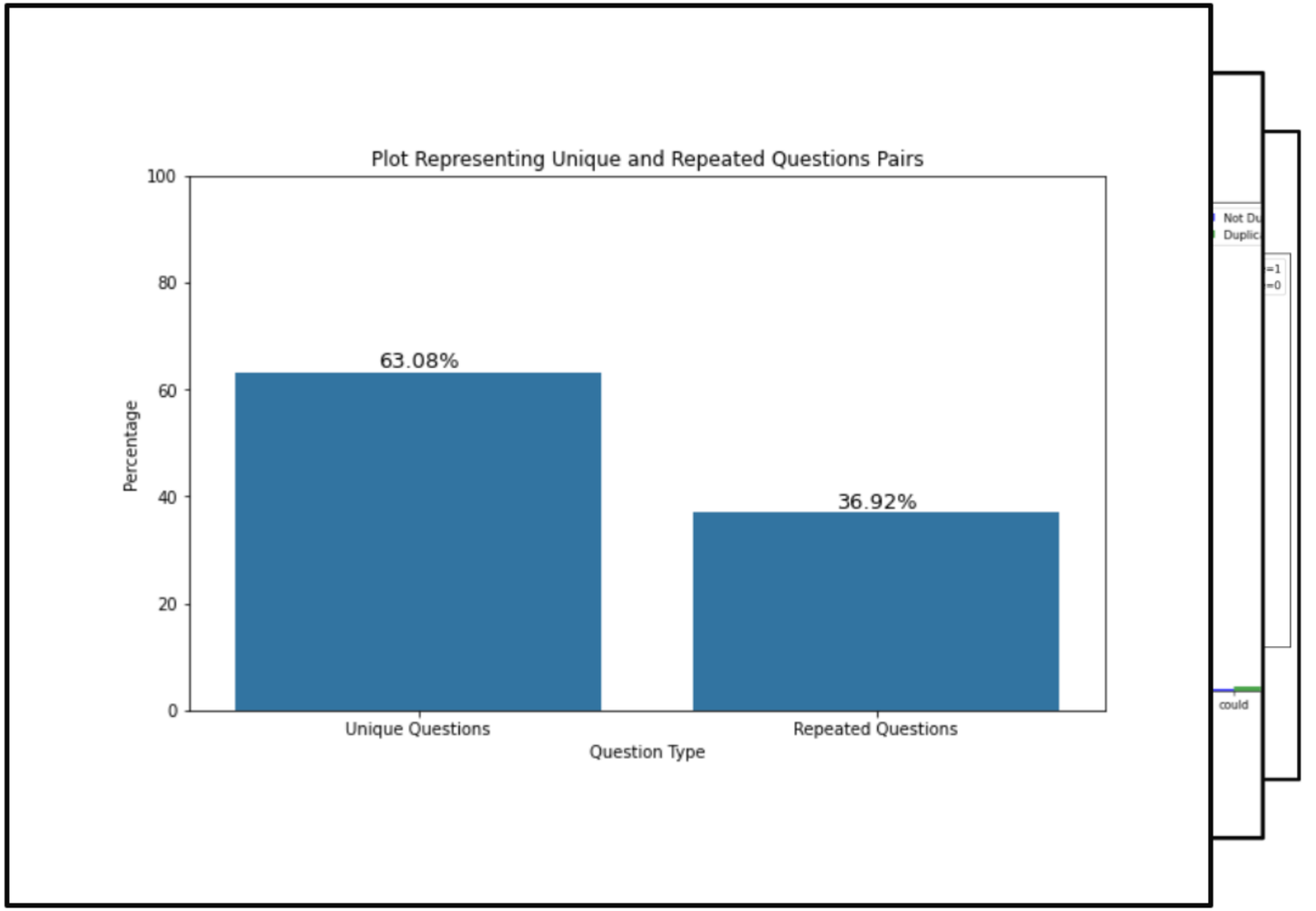
# Approaches to Solve the Problem

Exploring Various Techniques for Quora Question Pair Similarity

| | | |
|---|---|---|
| 1 | Understanding The Data | First went over the data and its textual information and perform EDA |
| 2 | Feature Engineering | Based on the knowledge gained, I extracted a few features and performed GloVe encoding using the SpaCy library. |
| 3 | Model Tranning | Train and evaluate the various models using different metrics such as loss and F1 score to decide the optimal model. |
| 4 | Ensemble Methods & Hypertunning | Perform hyperparameter tuning for the best models and combine multiple ML models to enhance prediction accuracy. |

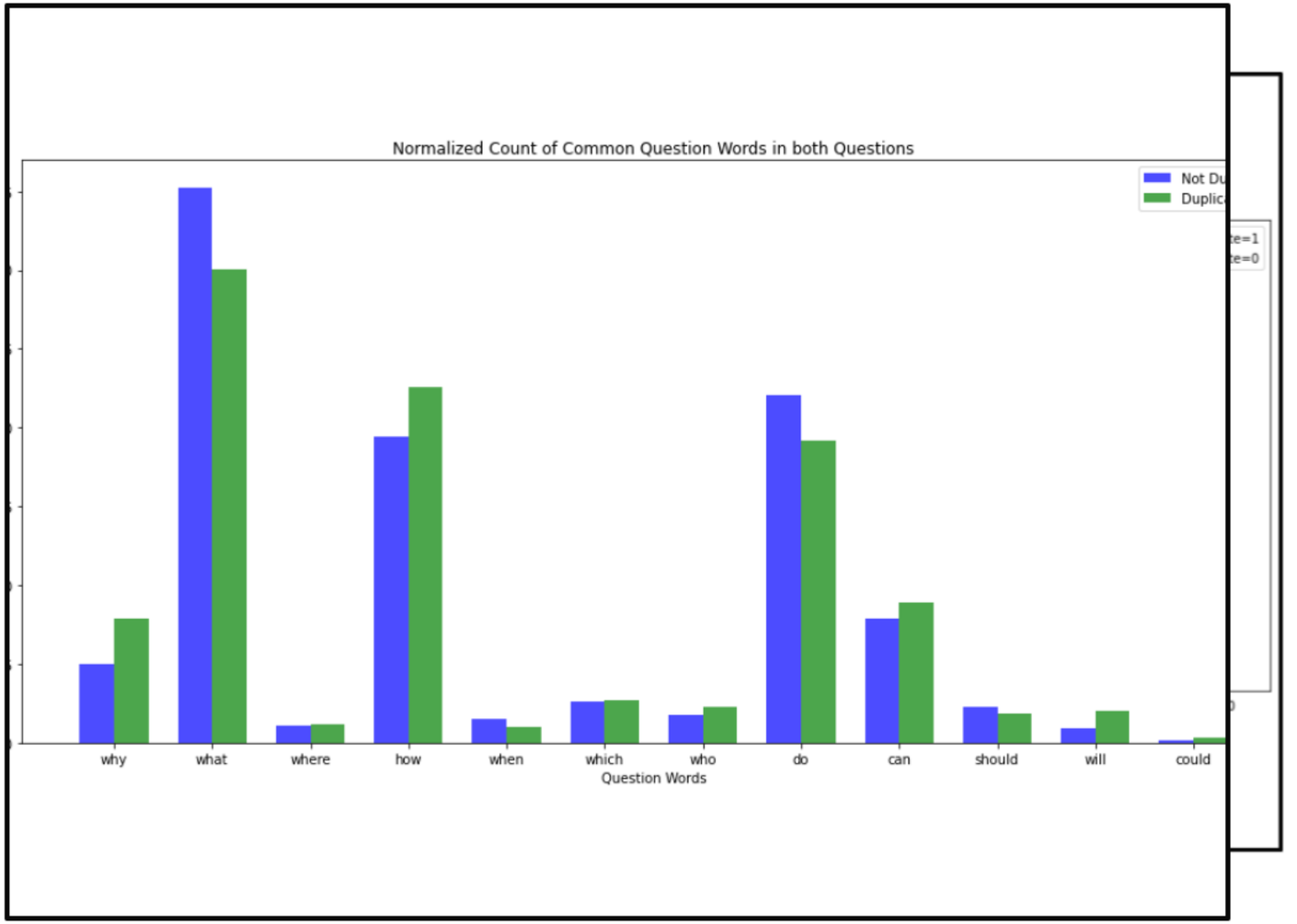
# Understanding The Data:

- The training set contains ~400,000 entries, and the test set contains ~3,500,000 entries
- Questions appear multiple times in the dataset, some more than others, with around 530,000 unique questions.
- There are more matches than non-matches, which may be an issue to consider.



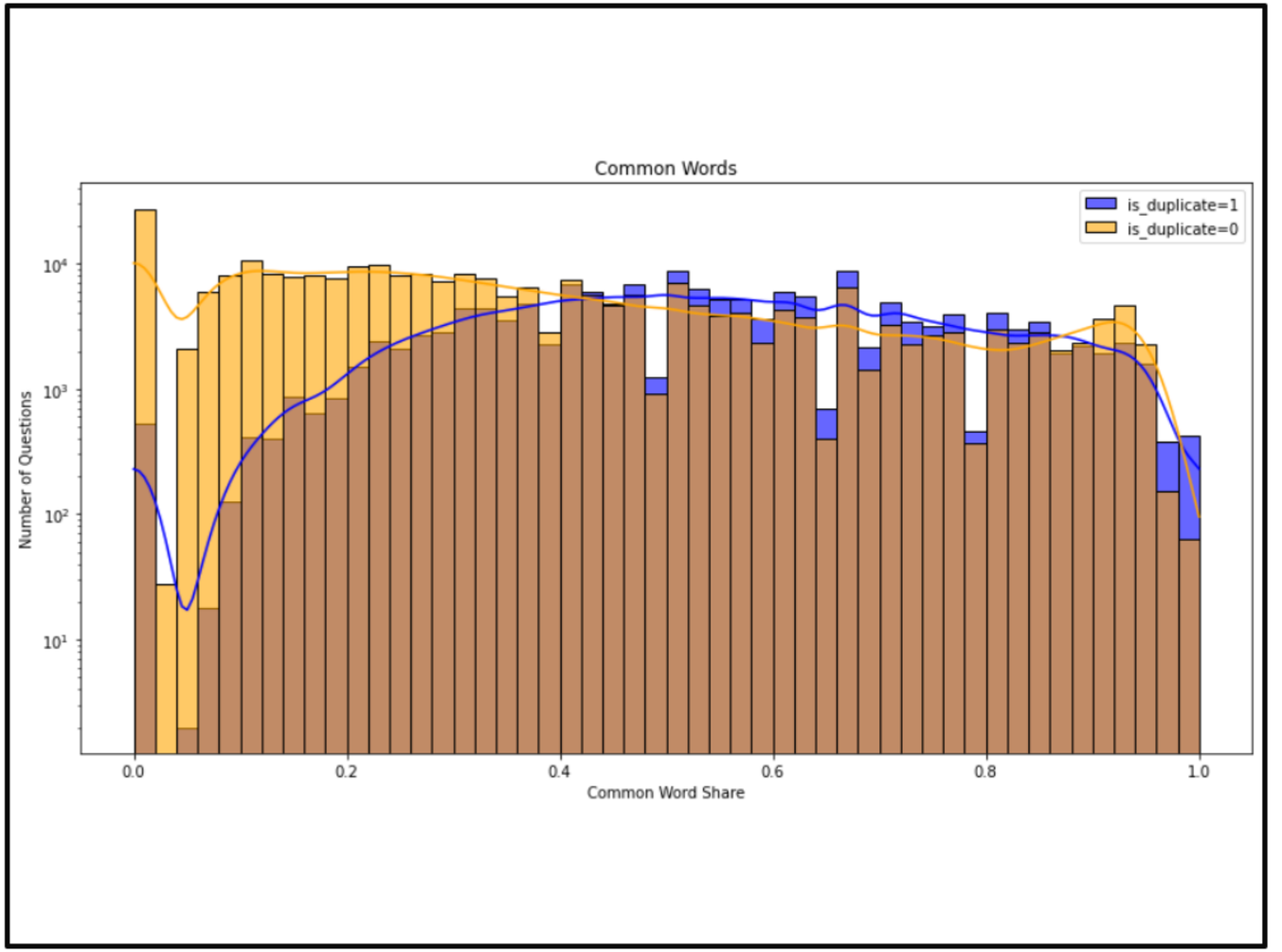
# Word Differences

- Under the assumption that common "question words" would likely indicate that the pair is a duplicate, we found that this was true for "how" and "why"
- in fact, the opposite was true.
- The data was normalized to account for the larger size of one question.



# Word Difference

- We continued the same analysis for all words, including stopwords.
- The Y-axis represents the number of pairs, and the X-axis represents the number of common words divided by the total words. Again, the data is normalized.
- As we can see, the increase in common words generally decreases the chances of a pair being a duplicate.
- However, in the 0-5% range, the chances of a pair being a duplicate shoot up
- This goes against the initial assumption made



## ***Extract Features***

Going over we will extract features that might usefull

### first_word_match and last_word_match

Match of the first words like "why, what etc" may be a factor, endding such as "best in india" and "best in USA" can also be very importent

---

### common_stopwords and stopword_ratio

Stopwords formed a very important context, with many pairs differing by only a few stopwords.

---

### common_words and len_diff

The number of common non-stopwords between the pair and the difference in the number of words.

---

### fuzz_ratio

Using the Python library that calculates the distance between two strings.

---

### char_overlap

Percentage of characters (including special characters) that are common.

Before training, we will analyze which features are relevant. As we can see, most have a very low correlation, so we will only consider those above 0.3, specifically last_word_match and fuzz_ratio. Total_words may also be used, but our dataset is already large, and including it could add unnecessary complexity.

| Factor | Correlation |
|---|---|
| first_word_match | 0.205171 |
| last_word_match | 0.312503 |
| common_stopwords | 0.113999 |
| len_diff | 0.201628 |
| common_words | 0.259125 |
| char_overlap | 0.229945 |
| fuzz_ratio | 0.383966 |

# Leveraging GloVe Pre-trained Embeddings

1. Given the initial assumptions were wrong, and considering the complexity of the textual data with the semantics, I used GloVe 300d from SpaCy.

2. GloVe generates dense word vectors that capture the semantic relationships between words.

3. By using GloVe, we can maintain the semantic integrity of sentences, ensuring that subtle differences are recognized and preserved.

4. Given its power, pre-processing steps like stemming, lemmatization, and removing stopwords are generally unnecessary.

Performance Metrics Comparison of Various ML Models

# Comparison of Different Machine Learning Models ( with ~25%)

Give the size as well as time and computation limitations I started out with 25% with no hypertunning, with the best ones I performed hypertnning. Finally tranied and saved the model on the best model and used it in test.csv

| Model | Accuracy | Loss | F1 Score |
|---|---|---|---|
| Logistic Regression | 0.73 | 0.52 | 0.73 |
| Random Forest | 0.77 | 0.47 | 0.76 |
| Gradient Boosting | 0.74 | 0.51 | 0.74 |
| XGBoost | 0.77 | 0.46 | 0.77 |
| SVM | 0.74 | 0.53 | 0.73 |
| neural network | 0.77 | 0.45 | 0.70 |

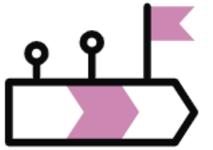
# Final Model on Complete Data

| | Best Parameters | Loss | F1 |
|---|---|---|---|
| XGBoost | • colsample_bytree: 0.7<br>• learning_rate = 0.2<br>• max_depth = 5<br>• n_estimators = 200,<br>• subsample = 1.0 | 0.43 | 0.79 |
| Basic NN | | 0.45 | 0.78 |

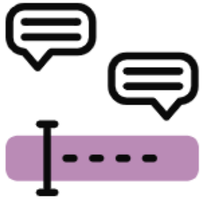
# Improvements:

Utilize deep learning tools and complex models like BERT and GPTs using HuggingFace, and build neural networks with TensorFlow or PyTorch.

Incorporate and experiment with pre-trained models, such as a spelling corrector, or other AI tools , named entity recognition, and part-of-speech tagging.



Conduct a more comprehensive model hyperparameter tuning and model selection, including ensemble methods.



Utilizing different encoding methods and feature extraction techniques like TF-IDF, word2vec, and topic modeling.

# Pratical implemention: Things to consider if deploying

- Assuming the task was to deploy on the Quora platform, the method would be completely different as this was optimized to fill the datasheet.
- Depending on the specifications, several modifications should be considered, such as an efficient search algorithm..
- We would want there to be a controllable threshold of similarity.
- Therefore, an alternative is a pop-up that shows similar questions and asks the user if they provide the same answer they asked for.
- This can also be used to train a new model.
- A slight bias towards flagging as non-pair may be necessary.