NLP Similarity Assessment:
Identifying Duplicate Questions on Quora

By Akash H Rao

June 23, 2024

Abstract

This comprehensive report describes the methodologies applied to develop a predictive model capable of discerning semantically similar questions on the Quora platform. This initiative aims to streamline user experience by minimizing content redundancy, thus enhancing the informativeness and efficiency of user interactions.

1 Introduction

The Quora platform serves as a hub for knowledge sharing, where users post queries and contribute responses. A prevalent issue of duplicate questions can undermine the quality of user experience and content discovery. This study details a approach to developing a robust model capable of detecting semantically similar questions, thereby aiding in maintaining the platform's integrity and efficiency.

2 Data Acquisition and Preliminary Analysis

The dataset consists of approximately 400,000 question pairs, tagged as either duplicates or non-duplicates. An initial analysis was performed to assess the quality of the data, which included identifying missing values and understanding the structural properties of the dataset. The test set contains approximately 3,500,000 entries. Questions appear multiple times in the dataset; some appear more frequently than others, with around 530,000 unique questions. Notably, there are more non-matches than matches, which may present an issue that needs consideration.

3 Exploratory Data Analysis (EDA)

Our EDA aimed to uncover underlying patterns and potential anomalies within the data:

- **Distribution Analysis:** We analyzed the frequency distributions of words and question lengths, which helped in understanding common question formulations.
- **Missing Values:** We developed strategies for handling missing data,
- **Common words between the pair** Analysis of common words in question pairs shows that an increase in common words generally decreases the likelihood of pairs being duplicates. However, in the 0-0.5 common word range, the chances of a pair being a duplicate increase significantly dataset.

4 Data Preprocessing

The preprocessing stage was meticulously designed to optimize textual data for sophisticated machine learning analysis:

- **Text Cleaning:** Punctuation marks were retained as they play a crucial role in understanding the nuances of questions, providing key contextual indicators.
- **GloVe Embeddings:** We utilized 300-dimensional GloVe embeddings provided by SpaCy to enhance our semantic analysis capabilities. This approach allows us to capture the underlying meanings of words within the text, facilitating a deeper understanding of question context and subtleties.
- **Tokenization and Handling Missing Data:** Systematic tokenization was applied to structure the text data suitably, while careful management of missing data ensured the integrity and usability of our dataset for subsequent feature engineering.

4.1 Leveraging GloVe Pre-trained Embeddings

GloVe (Global Vectors for Word Representation) embeddings are a form of word representation that capture the essence of words through their co-occurrence probabilities in large corpora. These embeddings are pretrained on vast datasets, enabling them to discern and encode the contextual relationships between words. For instance of a pair in the data:

- The query "What is One Coin?" refers to a specific entity known as "OneCoin," a term that could be misunderstood without proper contextual embedding.
- Its pair, "What is this one coin?" about a coin, mostly posted with an image, and requires understanding the contextual difference that the numeric adjective "one" introduces.

By incorporating 300D GloVe embeddings, our model can maintain the semantic integrity of sentences, distinguishing subtle semantic differences effectively. Preprocessing techniques such as stemming, lemmatization, and the removal of stopwords generally unnecessary, as the embeddings account for variations in word forms and meanings.

5 Feature Engineering

Crafted features specifically tailored to capture the essence of question similarity:

- **Keyword Matching:** The features `first_word_match` and `last_word_match` identify structural similarities at critical points in the questions.
- **Stopword Analysis:** Features such as `common_stopwords`, `stopword_ratio`, and `common_words` quantify the syntactic and semantic parallels.
- **Text Length and Character Analysis:** We assessed text length discrepancies and character overlap, providing a quantitative basis for similarity estimation.

Correlation Assessment: A correlation matrix was constructed to evaluate the predictive strength of each feature, ensuring only the most impactful features were retained.

6 Model Preparation and CSV Export

Post feature-engineering, our dataset was enriched with advanced contextual embeddings and selected critical features. This enhanced dataset was subsequently exported in CSV format to be trained on Google Colab.

7 Model Training

Our model training protocol was designed to efficiently manage computational resources, accommodating the extensive nature of the dataset. This process was structured into three distinct phases:

1. **Initial Training (100,000 Samples):**
   - **Objective:** Establish a performance baseline and identify models with high potential.
   - **Models Trained:**
     - *Linear Model:* Establishes a straightforward performance benchmark.
     - *Random Forest:* Evaluates robustness to overfitting, ideal for preliminary feature assessment.
     - *Gradient Boosting and XGBoost:* Known for their efficiency with large datasets; provide early insights into feature importance.
     - *Neural Network (TensorFlow):* Probes complex nonlinear relationships, showing initial promise due to effective modeling of nuanced word interactions.
     - *Support Vector Machine (SVM):* Effective in scenarios with high-dimensional data, crucial for text classification.
2. **Intermediate Training (200,000 Samples):**
   - **Objective:** Enhance and hypertune promising models.
   - **Models Trained:**
     - *XGBoost:* Adjustments and tuning based on initial phase insights.
     - *Optimized Neural Network:* Architecture and parameter adjustments based on performance metrics to better handle textual nuances.
     - *Random Forest:* Phased out due to excessive training durations.
3. **Final Training (Complete Dataset - 404,290 Samples):**
   - **Objective:** Maximize accuracy and generalization capability by leveraging the full dataset.
   - **Models Trained:**
     - *XGBoost (Optimized):* Further enhancements to maximize robustness and efficiency for large-scale deployment.
     - *Neural Network (Hypertuned with Keras Tuner):* Performance issues noted with increased data volume, suggesting potential overfitting. Future adjustments will focus on revising network architecture and increasing regularization to prevent overfitting and ensure scalability.

8 Model Performance Evaluation: XGBoost Classifier

The XGBoost Classifier was elected as the final model following a thorough assessment of its robust performance throughout the structured training phases. This section delves into the performance metrics of the model, providing a detailed evaluation and insights into its efficacy.

8.1 XGBoost Classifier Results:

The XGBoost Classifier achieved notable metrics with an accuracy of 79% and an F1 Score of 78%, complemented by a Log Loss of 0.43. These metrics suggest that the model not only predicts accurately but also maintains a balanced trade-off between precision and recall, which is crucial for the nuanced task of text classification.

Table 1: Classification Report for XGBoost Classifier

Class	Precision	Recall	F1-Score	Support
0	0.82	0.85	0.83	50,803
1	0.72	0.68	0.70	30,055

8.2 Overall Performance:

The classifier's performance is underscored by its consistency across various metrics:

- **Macro Average:**
  - Precision: 0.77
  - Recall: 0.76
  - F1-Score: 0.77
- **Weighted Average:**
  - Precision: 0.78
  - Recall: 0.79
  - F1-Score: 0.78

8.3 Interpretation and Insights:

The XGBoost classifier's performance can be attributed to its ability to handle large-scale data efficiently, its robustness to overfitting, and its flexibility in modeling complex non-linear relationships. The log loss value of 0.43 indicates that the model is not only making accurate classifications but also provides strong probabilistic assessments, which enhance its reliability in practical applications.

Furthermore, the disparity in performance between the two classes (0 and 1) provides critical insights. The higher performance on class 0 suggests that the model is more effective at identifying non-duplicate questions, possibly due to more distinct patterns or a larger quantity of training examples.

9 Challenges and Deployment

Despite the performance of our model in controlled test environment, the method would be completely different as this was optimized to fill the datasheet:

- **Reduce Latency:** To maintain a seamless user experience on Quora, it is critical that the model operates with minimal latency. Optimizing model inference and ensuring efficient integration with Quora's backend systems are essential to achieve rapid response times.
- **Handling Ambiguities and Typos:** User-generated content often includes ambiguities or typographical errors, which can mislead the model. Implementing advanced natural language processing techniques to enhance contextual understanding will be crucial for accurately handling these challenges.

- **Deployment Specifics:** Adapting the model to perform effectively in a live environment involves improving real-time data handling and seamless integration with Quora's existing search algorithms. This adaptation is necessary to ensure the model can promptly identify and suggest relevant similar questions.

Strategic initiatives designed to refine the system and enhance its readiness for real-world applications:

- **Efficient Search Algorithms:** Algorithms capable of dynamically searching and suggesting similar questions swiftly. This enhancement will improve real-time interactions with users, making the platform more responsive and intuitive.
- **Controllable Similarity Thresholds:** By implementing adjustable similarity thresholds, we will fine-tune the system's sensitivity to align with ongoing user feedback and the evolving dynamics of content on the platform. This flexibility will allow for more precise and relevant question matching.
- **Interactive Feedback Loop:** An interactive feature will be introduced, prompting users to verify the similarity of the suggested questions to their original query. This feedback mechanism will serve dual purposes: it will assist in the continuous training of the model and while helping the user find their solution easier.
- **Bias Adjustment:** develop strategies to adjust and mitigate biases inherent in the model, particularly those that might lead to over-flagging or under-representing certain types of content. This approach will ensure fair and unbiased model performance across a diverse range of user queries.