FINAL PROJECT PRESENTATION • NLP (DATS 6312_10)

QUORA QUESTION PAIR SIMILARITY

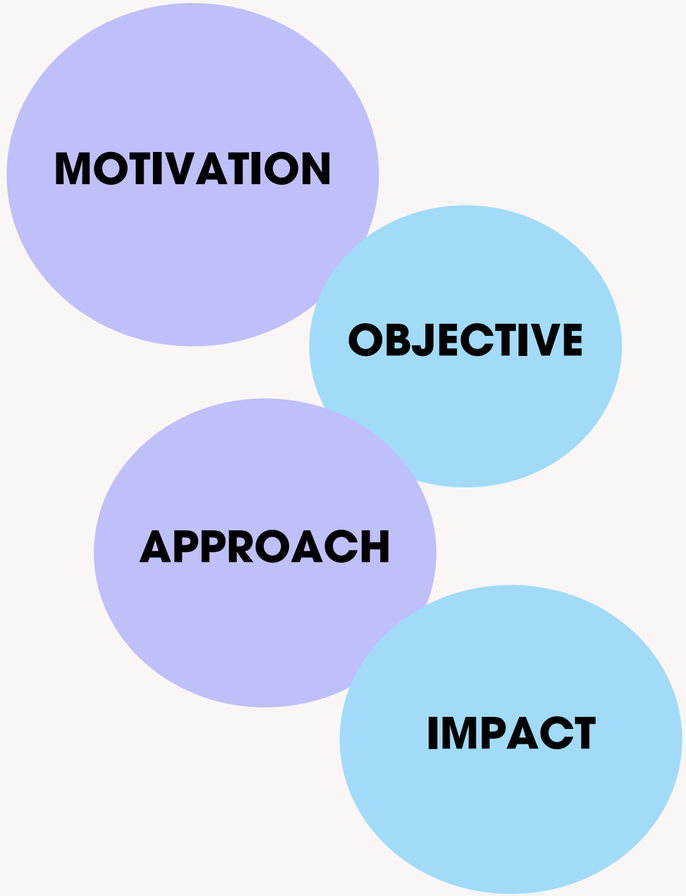
By: Group 5

Date: 12/09/2024

INTRODUCTION

This project focuses on solving the Quora Question Pair Similarity problem, a natural language processing (NLP) challenge. With over 100 million monthly visitors, Quora often encounters duplicate questions with the same intent, causing inefficiencies for users seeking answers and writers providing them.

By leveraging advanced NLP techniques, this project aims to classify whether two given questions are duplicates, improving the platform's ability to provide high-quality answers efficiently. The solution enhances user experience for both seekers and contributors by reducing redundancy and promoting better organization of knowledge.



DATA DESCRIPTION

The dataset is sourced from a Kaggle competition: https://www.kaggle.com/competitions/quora-question-pairs

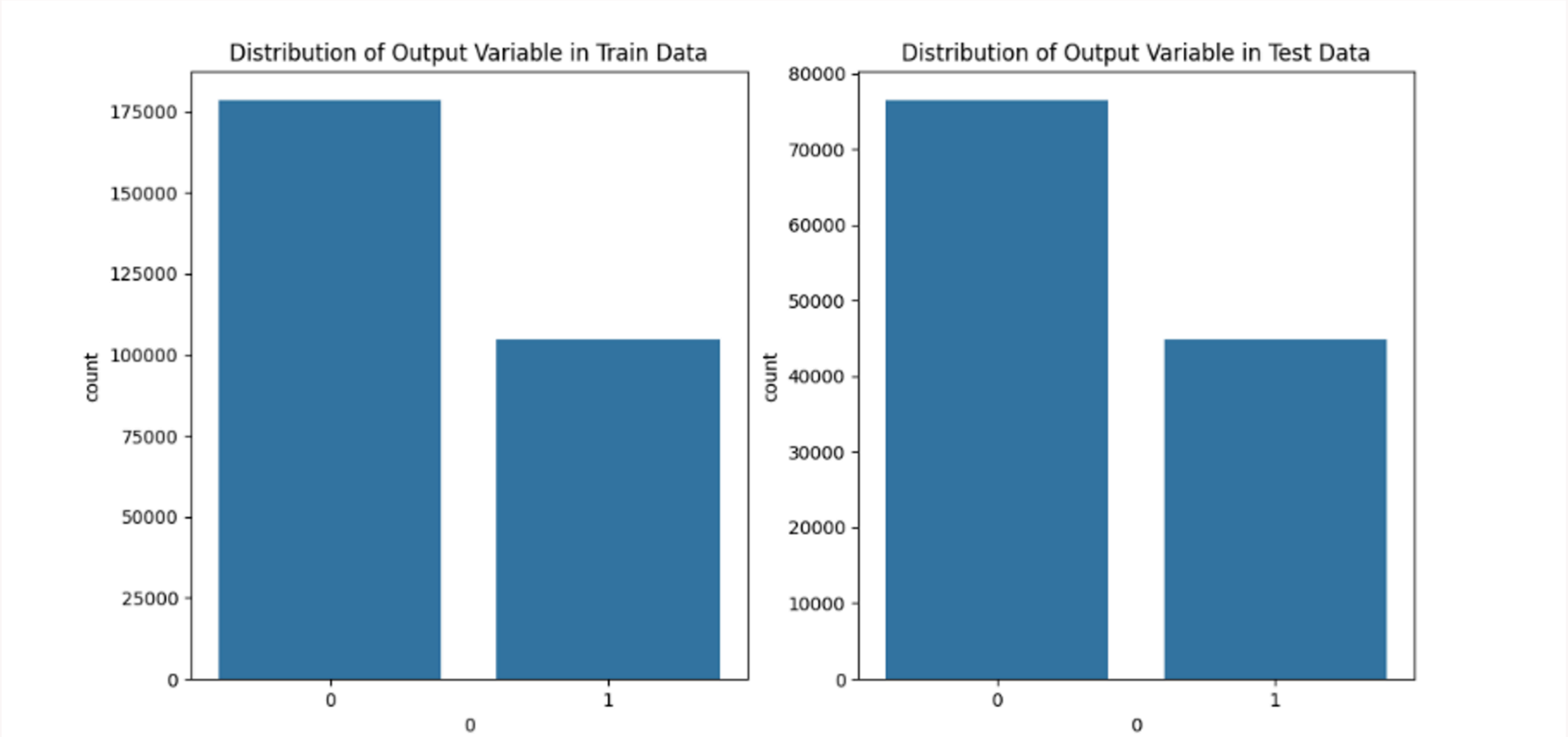
- There are an estimated **404k rows** of question pairs
- There are **6 features:**
  - **id -** the id of a training set question pair
  - **qid1, qid2 -** unique ids of each question
  - **question1, question2 -** the full text of each question
  - **is_duplicate -** the target variable, set to 1 if question1 and question2 have essentially the same meaning, and 0 otherwise.
  -

DATA PREPROCESSING

Steps taken as part of data preprocessing include:

- **Text Normalization:** Converts text to lowercase, replaces contractions, and standardizes symbols (e.g., currency, percentages).
- **Text Cleaning:** Removes non-word characters, HTML tags, and processes large numbers into readable formats
- **Stemming and Final Processing:** Applies stemming using PorterStemmer, removes extra spaces, and ensures a clean, whitespace-separated sequence.

VISUALIZING THE IMBALANCE IN DATA



FEATURE ENGINEERING

We extracted several features, all of which can be assigned to the following **three** broad categories:

Fuzzy Features

The word-to-word fuzzy similarity score is measured.

Eg: fuzz_ratio, fuzz_partial_ratio etc.

Common Subsequence features

We analysed the similarity between the different parts of the same sentence.

Eg: largest_common_subsequence, jaccard_similarity etc.

Token Features

We analysed the logistics of stopwords and non stopwords.

Eg: common non-stopwords, common stopwords, word size difference

FEATURE EXTRACTION

We explored two techniques for generating embeddings: one involved using weighted TF-IDF, and the other utilized CLS token embeddings extracted from the BERT base encoder. We created weighted TF-IDF sentence embeddings for two sets of text data (question1 and question2) using the combination of TF-IDF weights and pre-trained word embeddings provided by spaCy.

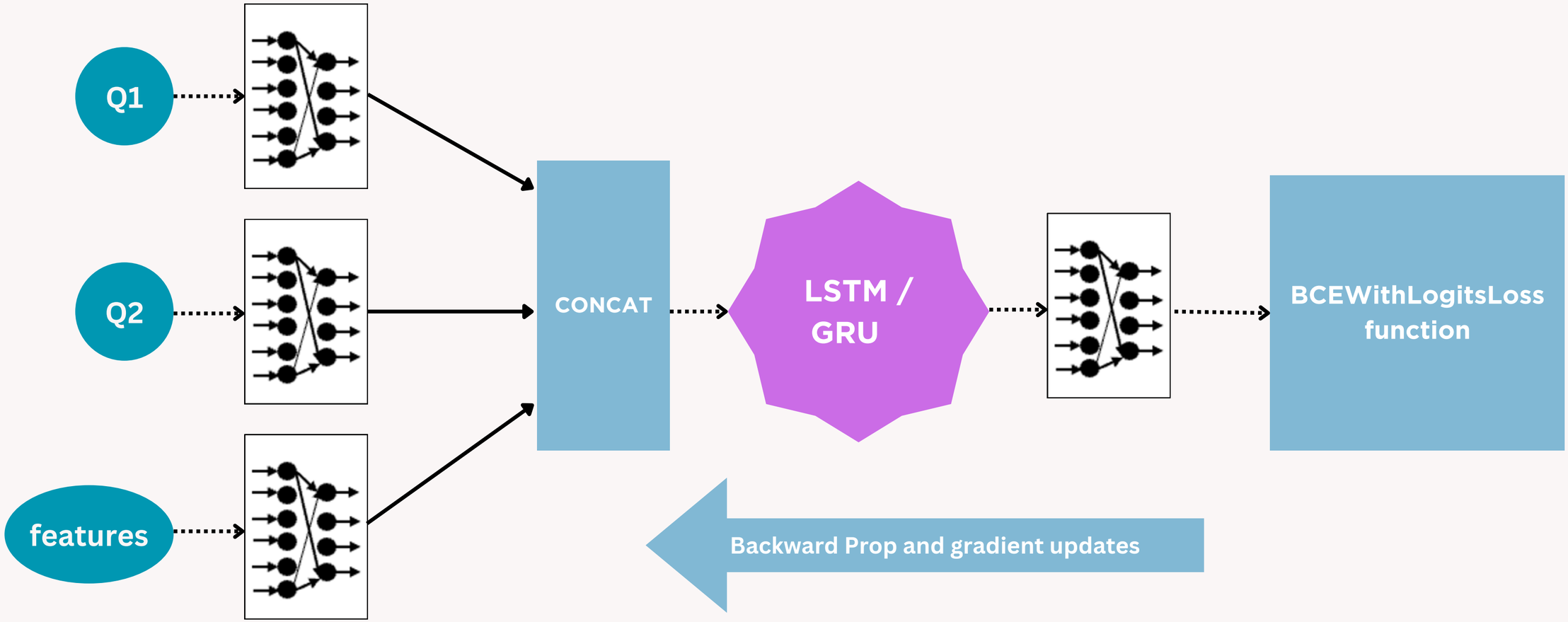
Classical ML Algorithms

By integrating both the embeddings and the engineered features, we utilized several **classical machine learning algorithms** to perform the binary classification task of identifying duplicate question pairs.

Algorithms	Weighted TF-IDF		BERT Embeddings	
	F1 macro	F1 micro	F1 macro	F1 micro
Naive Bayes	61%	61.3%	61.82%	62.17%
Logistic Regression	70%	74%	79.31%	81.23%
Random Forest	79%	81.7%	82.04%	83.97%
Xgboost	82%	83%	84.10%	85.38%

Deep Learning Methods

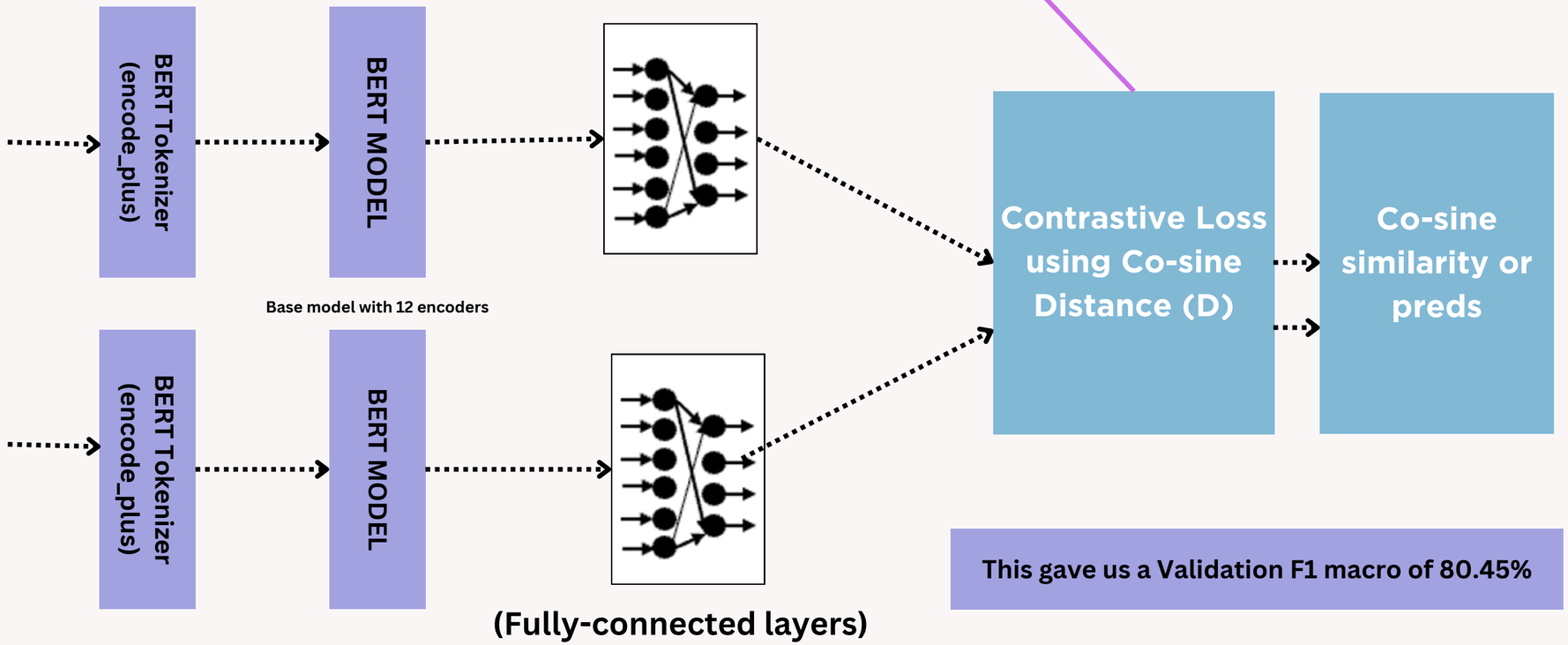
After checking with ML models we moved to applying Deep Learning methods:



BERT

We have selectively finetuned two encoder layers of our bert model. We aligned the embedddings using this model and then used cosine similarity to compute predictions.

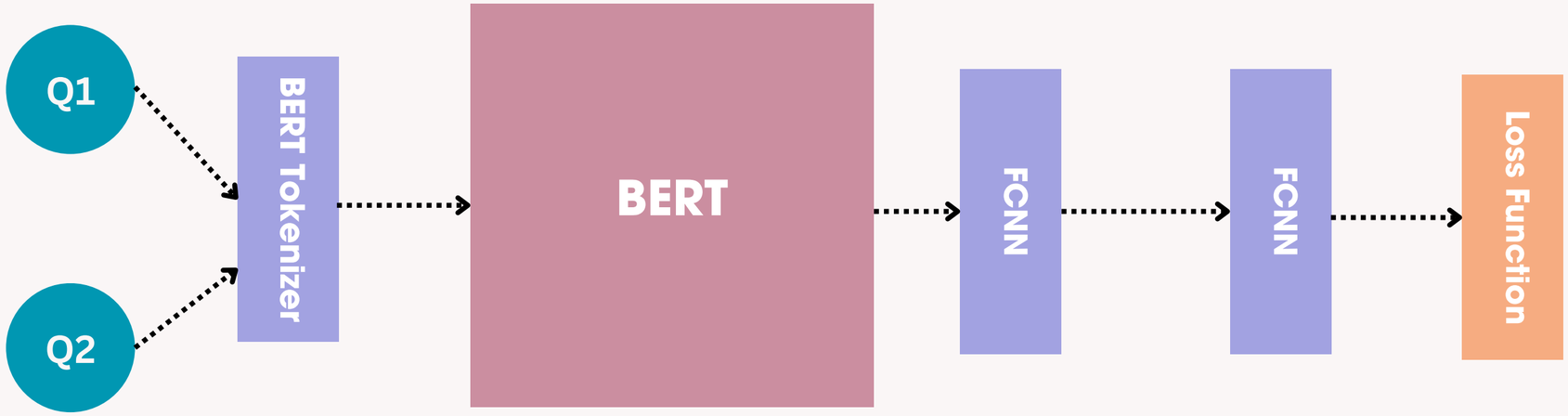
$$L = Y \cdot 0.5 \cdot D^2 + (1 - Y) \cdot 0.5 \cdot \max(0, \text{margin} - D)^2$$



This gave us a Validation F1 macro of 80.45%

BERT

We have selectively finetuned two encoder layers of our bert model. Along with this we have added two fcnns to capture additional non - linear relationship



This gave us a Validation F1 macro of 93.28%

DEMO