# Semantic Similarity-Based Duplicate Question Detection

Laura Roettges Cathy Cao Ellie Johnson
{roettges, ccao35, ekjohnson23}@wisc.edu
Team 8

## 1. Introduction

### 1.1. Motivation

Duplicate question detection is a fundamental task in natural language processing, with direct applications in search engines, customer support systems, and community-driven Q&A platforms such as Quora and Stack Overflow. These platforms often receive high volumes of user-submitted questions, many of which are semantically identical but differ in phrasing. For example:

$q1$: *How do I reverse a string in Python?*
$q2$: *What's the easiest way to reverse a Python string?*

Despite their different wording, both questions aim to retrieve the same information. Efficiently identifying and handling such duplicates can streamline answer delivery by reducing the need for redundant answer generation and reducing the time it takes to get a solution, improving user satisfaction.

### 1.2. Context & Literature Review

Several approaches have been proposed for tackling the problem of duplicate question detection, ranging from traditional information retrieval techniques such as lexical similarity and semantic overlap using resources such as WordNet [4], which is a large lexical database that groups words into 'cognitive synonyms,' to more advanced neural architectures [2]. A promising method is the use of Siamese neural networks, which learn to compare input pairs by projecting them into a shared embedding space and measuring their similarity [3, 5]. This architecture is well-suited for tasks involving pairwise comparison, as it consists of two identical subnetworks that share the same weights and parameters. Each subnetwork independently processes one of the two input items (in our case, encoded representations of $q1$ and $q2$), producing vector embeddings that are then compared using a similarity metric to inform the final classification. See Figure 1 for a visual overview, specific to our pipeline.

Imtiaz et al. [3] demonstrated that a Siamese neural network architecture, when trained on embeddings that span diverse knowledge domains, can improve generalization and performance, even outperforming some state-of-the-art methods. However, their experiments were limited to shorter input sequences (i.e., questions of 20 tokens or fewer), which may constrain the model's applicability to longer, more complex inputs.

Priyanka et al. [5] similarly reported strong performance from Siamese networks, specifically utilizing a BERT-based transformer model enhanced with self-attention mechanisms. Their approach incorporates both forward and backward attention to better capture contextual semantics within natural language input.

Recent advances in embedding models released by OpenAI alongside the GPT-4 suite (such as OpenAI's *text-embedding-3-large*) present an exciting opportunity to revisit this task. These newer models generate embeddings that are designed to capture rich semantic information across a broad range of domains, languages, and input types—including non-Latin scripts and special characters frequently found in naturalistic question data (e.g., mathematical symbols and code snippets), which raises questions about their potential to overcome challenges noted in previous studies with special character handeling [3]. However, to the best of our knowledge, there is no existing work evaluating their performance on the duplicate question detection task. This is especially true in the context of Siamese architectures or hybrid approaches that use these embeddings to augment traditional models like BERT through similarity-based features such as cosine similarity or Manhattan distance.

### 1.3. Goals

The primary goal of this project is to develop a system $P$ that accepts a pair of natural language questions $q_1$ and $q_2$ and outputs a binary classification indicating whether the questions are duplicates (1) or not (0). This task is framed as a binary classification problem

that utilizes similarity-based approaches.

We aim to evaluate the performance of OpenAI's *text-embedding-3-large model* on this task, both independently and compared to a custom BERT-based Siamese neural network. Specifically, we hope to determine whether this newer embedding model—designed with broader domain coverage and trained on diverse text, including non-Latin scripts and special characters (e.g., those used in mathematical expressions often found in Quora questions)—offers advantages over BERT-based models. Furthermore, we are interested in exploring whether incorporating similarity metrics derived from *text-embedding-3-large* (such as cosine similarity) can help overcome challenges with the BERT-based Siamese architectures.

## 2. Method

### 2.1. Data Set

Quora provides a data set consisting of 404351 question pairs available on Kaggle[1]. Each line in the set contains IDs for each question in the pair, the full text for each question, and a binary value that indicates whether the line truly contains a duplicate pair, i.e. the 'goal' or 'ground truth' we will use for training and evaluating our models.

The dataset is imbalanced, with approximately 37% of question pairs labeled as duplicates in the ground truth, and the remaining pairs as non-matches.

We do not modify any of the ground truth data even though error analysis reveals questionable cases. For example:

$q1$: *Can hamsters eat carrot? Why or why not?*

$q2$: *Can hamsters eat grapes?*

In the provided dataset, this pair was labeled as 1, i.e. matching questions. However, our team strongly felt that a value of 0 is more appropriate. For additional examples, see Appendix A. This ultimately could lead to some false positives and negatives in our validation that we actually feel are not true false positives or negatives, and these examples can negatively impact our training tentatively leading to lower precision rates.

We ultimately chose not to modify the ground truth labels to maintain consistency with the original dataset and to avoid accidentally introducing subjective bias. Editing the data could also harm the reproducibility and comparability with other work. There were also time restrictions that prevented us from doing this extensive review effort.

[1]Dataset available via https://www.kaggle.com/datasets/quora/question-pairs-dataset?resource=download

### 2.2. Data Pre-Processing

#### 2.2.1 Data Splitting

Initially, we split the data into three sets: train, test, and validation. To address the dataset imbalance, the training set, which is roughly 90% of the entire dataset, is comprised of a 50-50 split of matching and non-matching question pairs, which helps prevent models from learning to favor the majority class.

The test and validation sets, which are each roughly 5% of the entire dataset, retain the original class distribution and do not overlap with the training set.

Given that our dataset contains over 400,000 samples, allocating 90% of them for training still leaves over 20,000 rows each for validation and testing. This is sufficient data for evaluation while maximizing the training set size.

#### 2.2.2 Data Cleaning Informed by Preliminary Analysis

Manual inspection of the data revealed numerous questions too short to convey meaningful intent as well as a few empty set questions, such as those seen in Table 1.

| | | |
|---|---|---|
| . | ? | HH |
| What? | deleted | I'm |
| grammar | How long? | lol ? |
| Is? | hi | ok ? |
| i | What | o |
| A | Who is? | sss |
| 111 | Hh | spam |
| Which | ‘ | I |
| Can? | Aaas | Marriage |

Table 1. Sample of Questions Under 10 Characters

We removed rows that contained questions of less than 10 characters, as such short inputs are unlikely to yield meaningful matches or useful answers—especially in the context of Quora, where the goal is to provide users with relevant and pre-existing responses. Furthermore, these questions are better handled by simple rules or filters that detect common short strings that may indicate confusion or typos implemented outside a deep model pipeline.

We also removed excessively long questions with 100+ words (when splitting based on whitespace), as these are generally hyper-specific and presumed to be unlikely to have a match—see Appendix B for examples. Furthermore, certain transformer models have to-

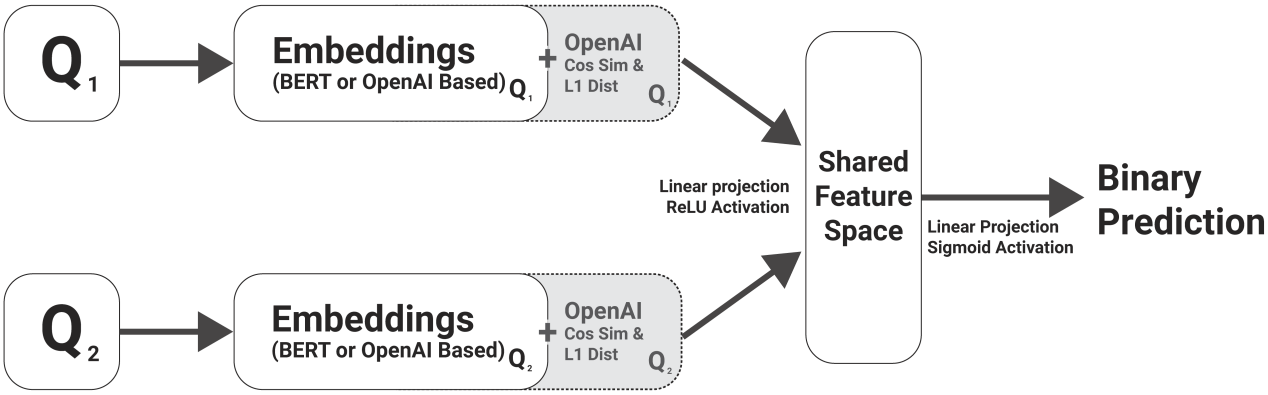



Figure 1. Siamese Neural Network architecture. $Q_1$ is the first set of questions, $Q_2$ is the comparison set. our BERT based embeddings are 384 dimensions while the OpenAI are 512 dimension. When adding in additional similarity metrics that adds 2 additional dimensions to the input feature space. These are condensed into a 256 hidden dimmension space, which is then condensed again to a 1 dimensional space to return a binary label [0, 1]

ken limits and removing lengthy questions has precedent in other work. Our data pruning is much less aggressive than that of the work of Imtiaz et al. [3] who only trained on data that could be split into 20 tokens or less, limiting their analysis to shorter questions in comparison to our experiments.

In order to further refine pre-processing, we examined false positives from a GPT-4o prompt-based experiment (see Appendix C) on an initial validation set. While the following insights are gathered from a different model and approach, we observed recurring patterns such as misinterpretations involving slashes (i.e. "w/" and "w/o" for "with" and "without"), suggesting potential ambiguity that could potentially affect embedding representations. Therefore, we decided to standardize such expressions in our pre-processing step. These false positive examples involving slash-based word concatenations can be seen in Table 2.

To address this, we introduced spacing around slashes to disambiguate compound terms—for instance, transforming "shepherd/collie" into "shepherd / collie"—and replaced shorthand notations with their full textual forms. Importantly, we ensured that any slashes occurring within `[math][/math]` tags remained untouched to avoid interfering with LaTeX-style mathematical expressions.

We also conducted a similar error analysis on predictions from the initial BERT-based Siamese neural network trained on unprocessed data (Baseline BERT SNN). Within the false negative instances, we noted challenges related to numeric reasoning and potential issues with inconsistent spelling. For instance, the

| **Question 1** | **Question 2** |
|---|---|
| What makes a **Husky/German Shepherd** puppy such a loyal companion? | What makes a **Labrador/Great Dane** mix such a loyal companion? |
| How can I get an intern opportunity at **Amazon/ThoughtWorks** in India? | How can a fresher enter ThoughtWorks India? |
| Which are some of the best web data scraping tools? | What is the best **tool/software** to scrape data from a website? |
| How much does it cost to build an **Android/iPhone** app from scratch? | How much does it cost to make an iOS or Android app? |

Table 2. False Positives with Slashes Concatenating Words

pair:

$q1$: *I have more than 6/10 and less than five ones. What number am I?*

$q2$: *I have more than 6 tens and less than 5 ones. What number am I?*

Although semantically similar, this pair was misclassified, likely due to differences in numeric formatting. To mitigate such issues, we converted numerical values outside of `[math][/math]` tags into their word equivalents using the *num2words* Python package[2]. This aims to improve model robustness by re-

[2] https://pypi.org/project/num2words/

ducing lexical variation in questions with numbers.

Despite these modifications, subsequent validation showed that these adjustments did not lead to a significant performance improvement, but did mildly improve our precision, see Table 3 to compare the Baseline BERT SNN against the Preprocessed BERT SNN for specific statistics. Our qualitative error analysis suggests that many of the remaining false positives are not merely due to surface-level text formatting, but rather stem from deeper semantic issues—such as missing domain-specific context (e.g., location, topic, or entity relevance) and emphasis on lexical overlap. Further discussion and analysis of these cases can be found in Section 3.

#### 2.2.3 Considered but Not Implemented

*Stop word removal:* We experimented with stop word removal using spaCy's English pipeline[3]. However, many of the default stop words are crucial for identifying question types (e.g., 'what', 'when', 'where', 'how', 'which', 'who') or for conveying contextual nuances (e.g., 'during', 'next', 'sometime', 'one', 'two'). Although we tested customizing the stop word list to preserve these terms, edge cases still posed challenges. For instance, removing stop words would reduce "How do I make friends?", "How to make friends?", and "How are my friends?" to a similar form like "how friends?", potentially causing incorrect matches. Additionally, since BERT and OpenAI embedding models are trained on natural language, we believe retaining stop words helps preserve important context and semantics. Thus, we opted not to implement stop-word removal in our reported tests.

*Spelling corrections:* we opted to not attempt to identify and fix spellings, as prior work showed it decreased the performance of their model on this dataset [2].

*Stemming and Lemmatization:* We chose not to apply additional stemming or lemmatization because our tokenization methods (discussed below) already breaks words down into subword units, which serves a similar purpose by reducing words to their core components. Additionally, given the extensive pretraining of models like OpenAI's *text-embedding-3-large*, we aimed to preserve the full contextual meaning of input text. Over-aggressive stemming could strip away meaningful distinctions (e.g., between 'universe' and 'university'), potentially harming model performance. Therefore, we relied on the pretrained models' internal handling of morphological variations rather than applying external pre-processing.

[3]See https://spacy.io/models/en; stop word list at https://github.com/explosion/spaCy/blob/master/spacy/lang/en/stop_words.py

#### 2.2.4 Tokenization and Embedding Generation

We experimented with multiple tokenization and embedding strategies, primarily relying on transformer-based language models. We categorize these methods into two major groups: BERT-based (pretrained and/or finetuned) and GPT-based embeddings. Each approach varies in its tokenization scheme, model architecture, and embedding dimensionality, as detailed below.

**BERT-Based Embeddings** *Pretrained (all-MiniLM-L6-v2)* For our Baseline & Preprocessed BERT Siamese Neural Network experiments, we used the *all-MiniLM-L6-v2* model from the SentenceTransformers library available in HuggingFace. This model adopts a BERT-style encoder architecture and is pretrained on over 1.17 billion sentence pairs using contrastive learning to produce semantically meaningful sentence embeddings.

Tokenization is handled by the WordPiece tokenizer, which splits text into subword units based on frequency statistics from its training corpus, which helps with rare or out-of-vocabulary words as they can be broken down into smaller, interpretable components. For example, 'unbelievably' might be split into: ['un', '##believable', '##ly']. The ## symbol denotes subword tokens that follow a root token. The vocabulary size for the *all-MiniLM-L6-v2* tokenizer is 30,522 tokens.

Once tokenized, input sequences are passed through the transformer encoder to produce dense vector representations. In the case of *all-MiniLM-L6-v2*, the final sentence embedding is derived by mean pooling over the token-level output representations (after applying attention masks). The output embedding dimension is 384.

More details on performance using this embedding strategy are available in Section 3.

*Finetuned SBERT* To capture domain-specific language and adjust the embeddings to emphasize semantic meaning rather than lexical overlap, we further finetuned the pretrained BERT encoder. Our finetuned enocder, referred to as *fine-*

*tuned_sbert_online_contrastive*, was trained using a supervised contrastive loss function on our training dataset. It retains the same tokenization and subword splitting strategy as the previous method, but adjusts the weights of the transformer encoder, impacting the vectorized embedding.

The encoder was finetuned exclusively on the training set to prevent data leakage with the validation and test sets. Finetuning occurred for four epochs with 500 warmup steps and a margin of 0.5 with OnlineContrastiveLoss, which has been shown to perform better than ContrastiveLoss [4]. This loss function was chosen to inject more semantic value into the embeddings by distancing semantically dissimilar embeddings despite high lexical overlap.

**OpenAI-Based Embeddings (text-embedding-3-large)** In addition to BERT-based methods, we evaluated embeddings generated via OpenAI's *text-embedding-3-large* model, accessed via OpenAI's API.

Input text is tokenized using OpenAI's proprietary Byte Pair Encoding (BPE) tokenizer. For typical English text, one token corresponds to approximately 4 characters.[5]

The original output embedding size is natively 3072 dimensions. However, we applied dimensionality reduction to 512 dimensions to strike a balance between semantic expressiveness and computational efficiency, ensuring the embeddings are compact enough for Siamese network training while limiting loss of semantic nuance. This was achieved using OpenAI's proprietary internal methodology for dimension reduction via a parameter specified in the API call.

### 2.3. BERT Siamese Neural Network Experiments

We conducted a series of experiments using the *all-MiniLM-L6-v2* model to generate sentence embeddings, which served as inputs to a Siamese Neural Network (SNN) for duplicate question detection. The SNN was designed to take as input a vector of the concatenated q1 embedding, q2 embedding, and element-wise difference between the two embeddings, making the input dimension $3 * 384$. This concatenated vector was then passed through a linear layer with a hidden dimension of 256 using a dropout of 0.3 and ReLU activation. The final layer produced a single-dimensional output with sigmoid activation to compute the similarity confidence. The SNN models were trained using binary cross-entropy loss and the Adam optimizer with a learning rate of $1e^{-4}$. During training, the F1 score of a small, separate early-stopping set was monitored to terminate training when no further significant improvements were observed ($\leq$ 0.005 F1 improvement). Each model was allowed to train for up to 50 epochs with a batch size of 64 and early stopping patience of five epochs, but the Baseline and Preprocessed BERT SNN trained for only six epochs before early stopping, both of the finetuned models (Finetuned BERT SNN, BERT + OpenAI Metrics SNN) only trained for one epoch, and the OpenAI Embedding SNN trained for 15 epochs.

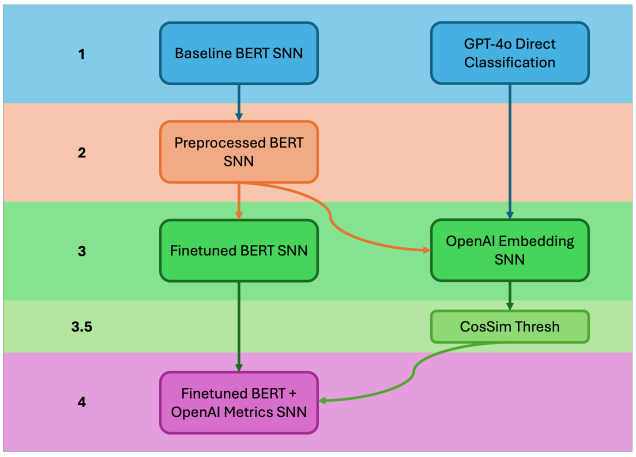



Figure 2. Experiment Progression: the diagram illustrates our progression across our experiments

Our initial experiment (*Baseline BERT SNN*) used raw question pairs without any pre-processing. Based on the errors we observed, we pre-proccessed the dataset as detailed in Section 2.2.2. We then repeated the experiment using a pre-processed version of the dataset (*Pre-processed BERT SNN*), resulting in only a very slight performance improvement.

Upon analyzing the false positives (which dominated the errors), we observed frequent errors in which the two questions shared most of their wording but differed by a critical word—such as a specific location or entity. This led us to hypothesize that finetuning the BERT encoder could improve its sensitivity to such subtle distinctions and improve areas of high lexical overlap. Consequently, we trained a domain-specific version of Sentence-BERT on our pre-processed dataset using contrastive loss (*Finetuned BERT SNN*). This model adjusted the encoder weights to better capture domain-relevant semantic cues.

[4]See https://www.sbert.net/docs/sentence_transformer/training_overview.html

[5]Examples of OpenAI's tokenization can be found at https://platform.openai.com/tokenizer

The results of these experiments are detailed in Section 3, with additional analysis for the finetuned method provided in Appendix D, for example Table 7 has samples of question pairs that were successfully classified after finetuning that previously were not.

### 2.4. OpenAI-Based Experiments

To explore the potential of large language models in this task, we experimented with two approaches using OpenAI's models.

***GPT-4o Direct Classification*** We initially used GPT-4o as a direct binary classifier. Each question pair was submitted via API along with a tailored prompt (Appendix C), and the model returned a predicted label (Yes/No) along with a confidence score. While this method yielded interpretable outputs, it proved inefficient due to high latency and cost, making it unsuitable for full-scale dataset processing. Additionally, it performed poorly in practice—achieving the lowest F1 score (48.92%) and precision (33.74%) of all evaluated methods, despite its high recall (88.89%), indicating a tendency to over predict duplicates.

***OpenAI Embedding-Based Methods*** With the goal of utilizing embeddings from a larger and more diverse training set to improve scalability and better capture semantic meaning, we transitioned to using OpenAI's *text-embedding-3-large* model to obtain embeddings for each preprocessed question. Each question is represented solely using its embedding vector (512-dimensional after dimensionality reduction).

Using these embeddings, we conducted two follow-up experiments:

1. *OpenAI Cosine Similarity Thresholding:* We computed cosine similarity between question embeddings in the validation set and classified question pairs based on a fixed similarity threshold. This simple method served as a fast baseline and enabled direct comparison with the BERT-based models in terms of false positive and false negative distributions.
2. *OpenAI Embedding SNN:* Here, we used OpenAI-derived embeddings as inputs to a Siamese Neural Network trained to classify duplicate question pairs. This approach allowed us to assess how well OpenAI embeddings performed within the same neural architecture as our BERT-based experiments.

Across both approaches, we observed notable differences in failure patterns when compared with the BERT-based models. This prompted the development of a hybrid method, discussed next.

### 2.5. Combined BERT and OpenAI-Based Experiments

Given potential differences in strengths of the BERT and OpenAI embeddings, we designed a hybrid model that combines features from both.

*Method:* We used finetuned BERT embeddings alongside two metrics derived from OpenAI embeddings—cosine similarity and Manhattan distance. These three components formed the input feature set for a final Siamese Neural Network classifier.

*Justification:* Upon comparing the error patterns of BERT and OpenAI-based classifiers, we found that over 66% of the false positives were unique when comparing the false positives with the thresholding method and 52% were unique when comparing against the OpenAI embedding SNN. That is, many errors made by one model were correctly classified by the other and vice versa. This finding suggested that a combined model might benefit from the complementary representations each embedding space offers. Additional qualitative examples are presented in Appendix E. Furthermore, other studies such as Homma et al. [2] suggested that using Manhattan distance resulted in better performance of their Recurrent Neural Network. We hypothesized that using Manhattan distance in conjunction with cosine similarity as an additional feature input to the SNN may improve our model's performance. Note that it would have also been reasonable to test using only one metric – for example, cosine similarity alone. However, we included both metrics in our initial experiment because cosine similarity captures angular similarity, while Manhattan distance reflects magnitude-based differences. In theory, combining them could provide a more comprehensive view of the relationships between OpenAI embeddings.

Performance comparisons are presented in Section 3.

### 2.6. Post Processing

All of our models experienced overall low precision even though we did see iterative improvements, so as our final experiment we added an additional post-processing step to the *BERT + OpenAI Metrics SNN* model to take a more cautious approach to predicting duplicates. Our model output both the binary predictions and their weight prior to sigmoid activation of the

final prediction, so we generated a modified prediction based on an additional threshold such that if the weight was not over the threshold value we would by default predict 0 (not a duplicate) rather than 1 (duplicate). We tested with numerous thresholds - this approach achieved our best results but may overfit the data in some cases. Results are indicated in 3 under the *Combined SNN + PostProcessing Thresh* Method, which includes both results for the validation set and our test set. Since we finetuned the threshold on the validation set data, we wanted to evaluate its performance on the test set to understand how it generalizes.

### 2.7. Evaluation

We evaluate all models using the following metrics:

- **Accuracy**: Overall proportion of correctly classified question pairs.

$$\text{Accuracy} = \frac{TP + TN}{T} \tag{1}$$

  where $TP$ is the number of true positives, $TN$ the number of true negatives, and $T$ the total number of predictions.
- **Precision**: Correctly identified duplicates as a proportion of all predicted duplicates.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2}$$

  where $FP$ is the number of false positives.
- **Recall**: Measures how well the model identifies true duplicates:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{3}$$

  where $FN$ is the number of false negatives.
- **F1 Score**: Harmonic mean of precision and recall:

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

  We further used the F1 score in our loss functions during model training.
- **Manual Error Analysis**: We also manually inspect a subset of false positives and false negatives to identify failure modes such as lexical overlap with semantic drift or ambiguous phrasing.

## 3. Results

### 3.1. Best Performing Method

We demonstrated iterative improvements on our models, with the highest performing model (without postprocessing) being the final combined approach *BERT + OpenAI Metrics SNN*, which achieved an accuracy of 88.12% and an F1 score of 62.75% on our validation set. Adding a postprocessing thresholding strategy to more cautiously predict matches allowed us to achieve an accuracy of 92.81% and an F1 score of 68.24% demonstrating improved performance. The thresholding strategy sacrificed recall in an attempt to meet our ideal goal of achieving over 70% scores for both recall and precision – unfortunately we were only able to achieve 66.11% for precision using this method. To determine our best threshold we finetuned our threshold value based on iterative threshold testing on the validation set landing on a threshold value of 0.88465 which maximized our observed F1 value. This precision result is 33% higher than the GPT4o prompt baseline and 26% higher than our Bert Baseline, demonstrating significant improvement from our initial test cases. We conducted further testing of our final approach on our test set (from our data split) to solidify our findings, we achieved an accuracy of 92.58% and an F1 score of 68.04% with the *Combined SNN + PostProcessing Thresh* $> 0.88465$ method which demonstrates that the threshold we chose still performed well when used outside the validation set that it was optimized for. Our results from an 'accuracy' perspective places within the top 3 papers reported on the *PapersWithCode* leaderboard for the Quora Question Pairs task[6], however those models do exceed our F1 scores in significant ways. **Table 3 summarizes the performance of all our evaluated methods.**

### 3.2. Notable Results from Our Previous Methods

We observed that finetuning the BERT model demonstrated a large improvement in terms of accuracy and harmonic mean with an over 7 point improvement for the F1 score and about a 3 point accuracy improvement over the non-finetuned *Preprocessed BERT SNN*.

Interestingly, using OpenAI's embeddings in an SNN setup also performed competitively, achieving an F1 score of 58.71% and 85.80% accuracy, slightly outperforming the non-finetuned BERT embeddings. This suggests that OpenAI's embeddings capture broader semantic relationships out of the box than the pretrained *all-MiniLM-L6-v2* model. However, the finetuned BERT model ultimately outperformed OpenAI's embeddings, highlighting BERT's greater adaptability to task-specific nuances – an advantage reinforced by the fact that OpenAI's current

[6]See https://paperswithcode.com/sota/question-answering-on-quora-question-pairs for leaderboard accuracy comparisons.

| Method | F1 | Accuracy | Recall | Precision |
|---|---|---|---|---|
| **Validation Set** | | | | |
| GPT-4o Direct Classification | 48.92 | 77.23 | 88.89 | 33.74 |
| Baseline BERT SNN | 55.06 | 83.63 | 91.24 | 39.43 |
| Preprocessed BERT SNN | 55.87 | 84.45 | 89.80 | 40.55 |
| Finetuned BERT SNN | 62.16 | 87.70 | 92.19 | 46.88 |
| OpenAI Embedding SNN | 58.71 | 85.80 | 92.13 | 43.08 |
| OpenAI Cosine Sim. Threshold $\geq 0.83$ | 48.27 | 85.46 | 61.88 | 39.56 |
| BERT + OpenAI Metrics SNN | 62.75 | 88.12 | 91.34 | 47.79 |
| Combined SNN + PostProcessing Thresh $> 0.88465$ | 68.24 | 92.81 | 70.51 | 66.11 |
| **Final Test Set** | | | | |
| BERT + OpenAI Metrics SNN | 61.57 | 87.27 | 92.01 | 46.26 |
| Combined SNN + PostProcessing Thresh $> 0.88465$ | 68.04 | 92.58 | 71.22 | 65.13 |

Table 3. Results (in Percentages) across all Classification Methods. **Top block:** validation-set results. **Bottom block:** final test-set results. We used the harmonic mean (F1 score) as our primary evaluation metric. For the cosine similarity thresholding approach, we report on the maximum F1 score achieved across multiple threshold tests. Across all methods the combined Bert + OpenAI Metric SNN approach combined with a post processing threshold achieved the highest F1 score.

API does not support custom finetuning of its embedding models.

Our simple cosine similarity thresholding on OpenAI embeddings method can achieve competitive accuracy results but demonstrates a trade-off between ease-of-use and performance. At a threshold of 0.83, it achieved 85.46% accuracy and a 48.27% F1 score, representing the highest F1 we observed across multiple tests of thresholding-based methods. While even higher accuracies were observed at stricter thresholds (e.g., 86.94% at threshold=0.85 and 89.43% at 0.90), they came at the cost of worse F1 scores. See Figure 3 for more thresholds.

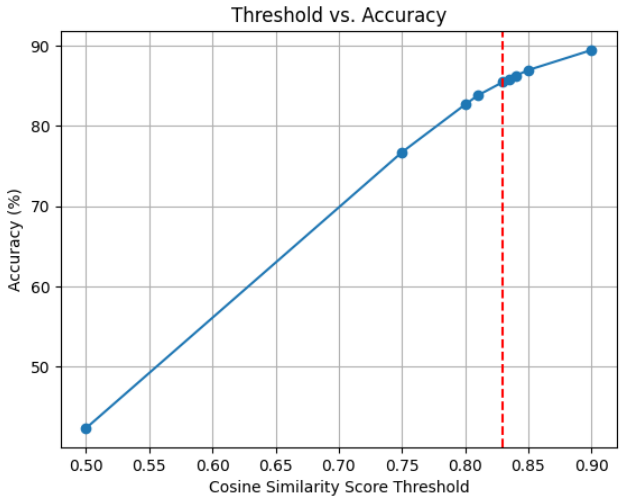
Finally, GPT-4o performs poorly for direct classification. Despite its advanced reasoning capabilities, *GPT-4o Direct Classification* achieved the lowest F1 (48.92%), likely due to prompt sensitivity and overfitting to lexical overlaps. This suggests that LLMs may require careful prompt engineering or fine-tuning for this task. Beyond performance, this method is also impractical for large-scale use due to high latency and API costs. Each inference required a separate API call, making it slow and expensive to apply to a dataset of over 400,000 pairs. These limitations underscore that, despite their power, large language models like GPT-4o may require careful prompt engineering, task-specific adaptation, or finetuning to perform reliably in structured classification tasks.

As such, these results show that while finetuned BERT and hybrid models require more training, they deliver superior performance. Augmenting a BERT-based SNN with metrics like cosine similarity from OpenAI embeddings improves precision while minimally impacting recall, leading to stronger overall results. However, this improvement does come with a financial cost of generating these embeddings, so our results do not sufficiently indicate that using these embeddings and metrics based off of them are worthwhile at this time.

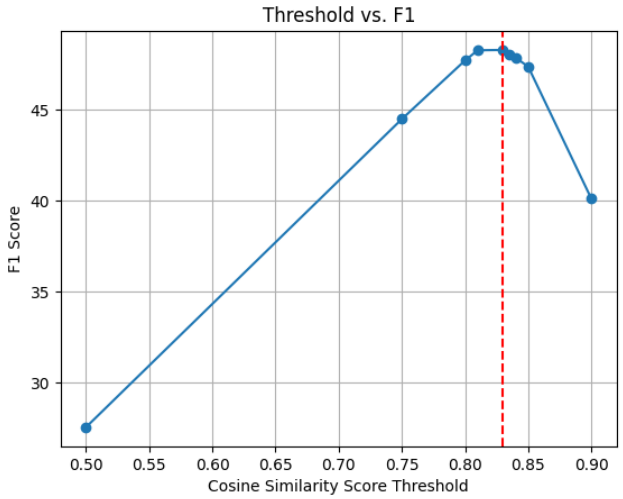
## 4. Conclusion

Our experiments, conducted iteratively, show that while a hybrid architecture, like *BERT + OpenAI Metrics SNN*, can achieve competitive performance on Quora's duplicate question detection task, there is still significant room for improvement. Our final F1 score of 68.04%, that we observed for our best performing model when assessed on our test set, falls short of human-level performance as well as the upper bounds of reported state-of-the-art models like that of DeBERTa variants [1] with $F1 > 90\%$, underscoring the inherent challenges of semantic matching in noisy, real-world data.

Much of our manual review showed that there is room for refinement of the goal data provided by the Quora Question Pairs dataset. This is because some of the ground truth predictions did not seem accurate (see




(a) Trade-off between threshold value and model performance (Accuracy).




(b) Trade-off between threshold value and F1 score.

Figure 3. Comparison of threshold effects on performance metrics using OpenAI embeddings. When classifying duplicates using cosine similarity on OpenAI embeddings. While accuracy increases with stricter thresholds, F1 score peaks at threshold=0.83 due to diminishing recall at higher thresholds.

Appendix A), impacting both the training and validation of models on this dataset. However, a comprehensive relabeling of the 400K+ question pairs was beyond the scope of this project, and we chose to preserve the original labels for consistency with prior work.

Our main challenge was with improving precision scores. We gathered some common themes around areas that continue to contribute to low precision values based on consistent patterns within question pairs yielding false positive predictions.

1. **Weak understanding of role reversal or changes in directional flow within similar syntax.** For example, the question pairs "What is the most profitable product to import from Canada to India?" and "What is the most profitable product to import from India to Canada?" are predicted as matches. This type of error was improved when we added thresholding but false positive error review still shows some evidence of this trend.
2. **Devaluation of 'you' vs. 'I' vs. 'we' or simply who is the object of the question.** For example, question pairs like "How do you transfer from a csu to a uc?" and "How do i transfer from a csu to a uc?" were classified as matches often. We believe that in most practical use cases it may be acceptable to identify these questions as duplicates because answer responses would be very similar, but for use cases where these distinctions are important, our model struggles.
3. **Lapses in placing enough importance on words that indicate a key distinction between the sentences.** For example the word 'comedy' in the following pair failed to successfully distinguish these questions: q1: 'which are some of the best romantic movies?', q2:'what are the best romantic comedy movies?' Similarly, questions with date information are often not distinguished correctly. For example, q1: "does college street, kolkata remain open on saturdays?" and q2: "does college street, kolkata remain open on sundays?" were classified as matches. For date differences specifically, oversampeling strategies when defining the training set could help improve model performances. However, for other scenarios this is a challenging thing to overcome. It is possible that training our model longer or using a BERT model with a larger word dictionary can help to address some of these.

Trends around weak understanding of role reversal or changes in directional flow within similar syntax, as well as failing to distinguish common differences like differences of dates, could potentially be addressed with oversampling methods for the training set. While this was beyond the scope for this project, we recommend it for future tests.

Exploring alternative activation functions beyond sigmoid activation could yield possible improvements as well, but this was also out of scope of our study given time limitations.

Our experiments highlight that while off-the-shelf LLM embeddings offer strong semantic baselines, task-specific finetuning and hybrid modeling can sig-

nificantly boost both performance and interpretability. Future work can explore label quality, richer input features, and targeted finetuning to close the remaining performance gap to state-of-the-art benchmarks.

## 5. Contribution & Code Base

All authors contributed to the efforts of this project in a equitable manner.

- **Cathy** managed the OpenAI code calls to gather insights for both our prompt based approach and for gathering embeddings from OpenAI's API. She also set up data splitting.
- **Ellie** conceptualized, coded, and trained the Baseline BERT SNN model; coded and trained the Preprocessed BERT SNN model; analyzed errors to conceptualize, code, and train the Finetuned BERT SNN model; and trained the BERT + OpenAI Metrics SNN model.
- **Laura** performed preprocessing steps including exploration of stopword removal, modified the SNN code for the Combined BERT + OpenAI Metrics SNN method, performed postprocessing thresholding, wrote helper functions which support reviewing false positives and negatives, generating evaluating results, and cleaning the data.
- **All** contributed to analysis of false positive and negatives to determine next steps and collectively reviewed this paper for alignment.

Our code base can be reviewed at https://github.com/roettges/774, please review the README for further guidance. Note that not all data files are included based on file-transfer and storage considerations, but the full code base for conducting our experiments is available there. If you would like access to certain data files not included in GitHub such as the OpenAI encoding files please contact Laura Roettges at roettges@wisc.edu

## References


[1] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. {DEBERTA}: {DECODING}-{enhanced} {bert} {with} {disentangled} {attention}. In *International Conference on Learning Representations*, 2021. 8

[2] Yukiko Homma and Christopher Yeh. Detecting duplicate questions with deep learning. 2017. 1, 4, 6

[3] Zainab Imtiaz, Muhammad Umer, Muhammad Ahmad, Saleem Ullah, Gyu Sang Choi, and Arif Mehmood. Duplicate questions pair detection using siamese malstm. *IEEE Access*, 8:21932–21942, 2020. 1, 3

[4] George A. Miller. WordNet: A lexical database for English. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*, 1994. 1

[5] Gutti Venkata Ranga Priyanka, Anuradha T, and Niktha Malladi. Duplicate quora questions pair detection using siamese bert and ma-lstm. In *2023 3rd International Conference on Advances in Computing, Communication, Embedded and Secure Systems (ACCESS)*, pages 192–196, 2023. 1

## A. Poorly Labeled Ground Truth Examples

| Question 1 | Question 2 | Goal | Our Interpretation |
|---|---|---|---|
| Where did the Civil War begin? | What year did the Civil War begin? | Match (1) | Non-Match (0): Different focus – location vs. time |
| Can someone translate into latin "QVIS ERVDIET WOTHOVT DOCVMENTA"? | Can someone translate this to English from latin I guess? | Match (1) | Non-Match (0): Missing Context - no latin included to translate in 2nd question |
| Why did Mint split from Ubuntu? | Which one is better: Linux Mint or Ubuntu? | Match (1) | Non-Match (0): Mismatched Question Types - why vs comparison type questions |

Table 4. Question Match Evaluation Table

## B. Sample Questions over 100 words

| Question Text |
|---|
| Heartbreak? Heartbreak? She's my girlfriend for two months, I chose her over my girlfriend for 2 years. I like her so much to the point that I can't let her go even if she wants to end our relationship because of the other people around us most especially her family. I do the things for her that I'm not used to for a girl and I am willing to sacrifice everything just to have a little time with her. A little and limited time that I'm asking from her but she don't wanna give it to me. She's scared that someone might see us, that she's still having an affair with me. I love her and I want to be with her at least once a week even if just for a limited time. I'm not sure if I'm doing the right thing, all of my friends told me to stop it and just let it go 2 months is just 2 months not a deep relationship. But they don't feel what I feel, in this span of time I learned a lot, I learned how to love, to be loved, to sacrifice a good life, and to sacrifice a better clear future. My mind tells me to stop, but my heart tells me to hold, don't give up, stay with her and give her the unconditional love. Should I follow my mind or follow my heart? |
| I'm 33 and diagnosed with depression. I don't like my work and I have realised that I made the wrong choices concerning my studies. I too realised (under professional psychologist surveillance) that I am a High Potential Adult with a IQ of 147 and a high emotional level. I'm married, I have two wonderful childrens and a house to pay for. I feel that I'm living a wrong life in the sense that the situation I am in, does not fit me at all other than the fact that I deeply love my family. Did someone ever succeed at changing their life with a similar kind of situation? |
| Wifi problem... My Mtnl wifi router is frustrating me from the last one week. I use it to connect my galaxy ace plus to the internet. The porblem is that router is responding slowly while i surf the internet though i'm getting full speed of 2mbps. I also use my phone's wifi in my college and there the phone works fine and responds quickly. I tried to connect some other phone to connect with my router and the router is giving quick reply to that phone. I'm unable to find where the problem is. I tried to contact mtnl for the same but they are not entertaining me. Please help me out... |
| Brain Teasers: Two women set out to the market to sell some oranges. Each had 30 oranges. The first lady sold oranges at 2 for a rupee and second lady sold oranges at 3 for rupee. At last, the first and second lady made Rs 15 and Rs 10 respectively. So the total amount is Rs 25. The next day, when they sold their 30 oranges together to make business profitable, they pooled their sixty oranges and sold at rate of 5 for Rs 2. After they sold all oranges, they found they had only Rs 24. They could not understand where the one rupee went. Where did it go? |

Table 5. Questions with Over 100 Words

## C. GPT-4o Prompt

```
prompt_content = f"Are the
↪ following two questions
↪ semantically
    similar?\n1. {q1}\n2. {q2}"

completion =
↪ openai.chat.completions.create(
    model="gpt-4o",
    messages=[
        {
            "role": "developer",
            "content": "Respond
            ↪ only in the format
            ↪ '[Y|N]
                [confidence
                ↪ score]'. The
                ↪ confidence
                ↪ score should
                be a number from
                ↪ 0-100,
                ↪ inclusive."
        },
        {
            "role": "user",
            "content":
            ↪ prompt_content
        }
    ]
)
```

## D. Deeper Analysis on the Finetuned BERT model's Performance

Table 7 includes a review of sample question pairs which were correctly classified after finetuning the BERT model that previously failed. These examples suggest that finetuning improved tolerance for detail mismatches (more or less specific) suggesting better semantic understanding relying somewhat less on lexical overlap.

However, there certainly are examples that continue to fail inappropriately exhibiting still too much reliance on lexical overlap overall. These do tend to have *lower* confidence or prediction scores, which could suggesting that adding a post processing step in our pipeline to favor even higher predictions may be worth exploring. A couple of examples are in table, 6.

Note that we continue to see examples that are predicted as duplicates when the ground truth data suggests that they are not, but we disagree with the underlying ground truth data. There are also scenarios that frankly a human may also have a hard time deciphering such as ID: 169189, q1: "how long would it take to get six packs?", q2: "how long would it take to get a six pack?". Does the first question relate to six packs of beer or is it related to the fitness goal of getting six packs for example yourself and another person getting in shape?

| ID | Question 1 | Question 2 | Weight |
|---|---|---|---|
| 55548 | Which one should i choose 50 mm or 40mm prime lens? | what advantages do a 40mm prime lens have over a 35 or 50? | 0.678 |
| 320331 | how can i start learning robotics? | where can i learn to build advanced robotic equipment from scratch? | 0.525 |

Table 6. False Positive Sample Question Pairs with their prediction scores. The weight column indicates the prediction before Sigmoid Activation for the binary predictions

| ID | Question 1 | Question 2 | Previous FP/FN |
|---|---|---|---|
| 145215 | What is the police subculture **in the U.S.?** | What is police subculture? | FP |
| 243771 | Which is the best place in mumbai for a candle light dinner? | Which is the best place in mumbai, **near borivali**, for a candle light dinner? | FP |
| 174442 | How do i improve english **in a non-english-speaking environment**? | How can i become fluent in english? | FP |
| 398749 | What do you think lifelong learning is? | What is the lifelong learning **program?** | FP |
| 149846 | Which is the best book for preparing clat **2018**? | What are the best books to prepare for clat **2017**? | FP |
| 16940 | How do i use my windows 10 pc as bluetooth speakers for my android phone? | How do i use my windows 8 pc as bluetooth speakers for my android phone? | FP |
| 149846 | Who has a **higher iq**, hillary clinton or donald trump? | Who is more **honest**, hillary clinton or donald trump? who lies more? | FP |
| 396536 | what are the benefits of doing an mba after getting an undergraduate degree in **ministry?** | what are the benefits of doing an mba after getting an undergraduate degree in **sociology?** | FP |
| 166640 | What are **some causes** for a lump on the side of my head? | How can i **prevent** getting lumps on the side of my head? | FP |

Table 7. Sample question pairs that were correctly classified after finetuning embeddings. The fourth column indicates if it was previously a false positive or a false negative that had been corrected.

## E. Comparing Results of the FineTuned Bert Model with the GPT SNN

We compared false positives classified by the Finetuned BERT SNN that were not classified as false positives for the OpenAI Embedding SNN to false positives classified by the OpenAI Embedding SNN that were not classified as false positives for the Finetuned BERT SNN and made a few observations.

The Finetuned BERT model seemed to perform better than the OpenAI Embedding model on 'time' based distinctions. For example the OpenAI embedding model classified the following as duplicates where the BERT finetuned model did not: q1: "How do you study english?" and q2: "How do i study english **in a day?**" as well as q1: "How do i lose weight **in a month**? and "What are the best ways to lose weight?"

The OpenAI embedding model placed too much importance on topic similarity without identifying distinguishing features of the questions that make them unique, for example classifying q1: "What makes a bat a mammal?" and q2: "Are bats mammals?" as duplicates. However this could be a reason to combine it with the Bert finetuned model, as the BERT finetuned model failed to recognize scenarios that have high lexical overlap but different topic semantic interpretations, for example the BERT Finetuned model indicated these questions as duplicates when the OpenAI model did not: q1: "How do i get into harvard as an undergrad?" q2: "How do i get to harvard campus?"