High-Performance Semantic Retrieval for Duplicate Question Detection on Quora via Fine-tuned Sentence-BERT

Yanzhi Ding
ID:21163438

Abstract

The rapid growth of community question-answering platforms like Quora has led to a high volume of duplicate questions, making efficient information retrieval a significant challenge. This paper presents a high-performance semantic retrieval system designed to identify duplicate questions within the Quora Question Pairs dataset. We employ a Bi-Encoder architecture based on the 'all-MiniLM-L6-v2' Sentence-BERT model, which is specifically optimized for generating semantically meaningful sentence embeddings. To further adapt the model to the target domain, we fine-tune it using a Triplet Loss function, which effectively learns to differentiate between similar and dissimilar questions. For efficient retrieval, we leverage the FAISS library to build a fast, scalable vector index. Our system is evaluated on a comprehensive set of information retrieval metrics. The experimental results demonstrate the effectiveness of our approach, achieving a strong Mean Reciprocal Rank (MRR) of 0.6401 on the test set, confirming that fine-tuned sentence transformers provide a robust and scalable solution for large-scale duplicate question detection.

1 Introduction

Community Question Answering (CQA) websites, such as Quora and Stack Overflow, have democratized knowledge sharing, accumulating vast repositories of information contributed by millions of users. However, the open and unconstrained nature of these platforms leads to significant content redundancy. Users frequently ask questions that have already been answered, often using different phrasing. This phenomenon of duplicate questions fragments knowledge, complicates information discovery, and degrades the overall user experience by forcing users to sift through multiple threads to find a definitive answer.

The primary challenge in detecting duplicate questions is the "lexical gap": questions can be semantically identical while having very little word overlap. For instance, "How can I start investing with a small amount of money?" and "What are the best ways for a beginner to get into the stock market on a budget?" share the same intent but use different vocabulary. Traditional information retrieval methods, such as those based on TF-IDF or BM25, are often insufficient as they rely heavily on keyword matching and struggle to capture the underlying meaning.

To address this, recent advancements in Natural Language Processing (NLP), particularly the advent of large-scale pre-trained language models like BERT (Devlin et al., 2019), have provided powerful tools for understanding sentence semantics. However, deploying these models for large-scale retrieval tasks presents a scalability challenge. Using a standard BERT model as a Cross-Encoder, which processes pairs of questions together, yields high accuracy but is computationally prohibitive for real-time search over millions of candidates.

A more practical approach is the Bi-Encoder architecture, popularized by Sentence-BERT (Reimers and Gurevych, 2019). This method encodes each question into a dense vector embedding independently. These embeddings can be precomputed and stored in an efficient search index. This transforms the retrieval task into a fast nearest neighbor search in a vector space, which is highly scalable.

In this work, we develop and evaluate an end-to-end semantic retrieval system for the Quora duplicate question task based on this principle. Our main contributions are:

- We design a complete system based on a Sentence-BERT Bi-Encoder architecture, demonstrating a practical solution to a real-

world problem.

- We fine-tune a compact, pre-trained model (‘all-MiniLM-L6-v2‘) on the Quora dataset using a Triplet Loss objective, significantly enhancing its domain-specific discriminative power.
- We integrate FAISS (Johnson et al., 2019) to create a highly efficient inference pipeline, ensuring the system is scalable to millions of questions.
- We conduct a thorough quantitative and qualitative analysis using a suite of standard information retrieval metrics, providing deep insights into the system's performance and limitations.

2 Related Work

The task of detecting semantically similar sentences has been a long-standing research area in NLP. The methodologies have evolved significantly over the years.

2.1 Lexical and Statistical Methods

Early approaches relied on lexical similarity features. Methods like TF-IDF (Term Frequency-Inverse Document Frequency) represent sentences as sparse vectors based on word counts, weighted by their importance. While effective for simple keyword search, they fail to capture semantics, synonymy, and word order.

2.2 Static Word Embeddings

The introduction of dense word embeddings, such as Word2Vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014), was a major breakthrough. These models learn vector representations of words from large text corpora. A common approach for sentence similarity was to average the embeddings of their constituent words. However, this "bag-of-words" approach disregards word order and syntax, and it cannot disambiguate context-dependent word meanings (polysemy).

2.3 Contextualized Embeddings with Transformers

The Transformer architecture (Vaswani et al., 2017) and subsequent pre-trained models like BERT (Devlin et al., 2019) revolutionized the field. By processing words in the context of the entire sentence, BERT generates contextualized embeddings that capture nuanced meanings.

Two main paradigms emerged for using BERT for sentence-pair tasks:

Cross-Encoder: In this setup, two sentences are concatenated with a special separator token and fed into the BERT model simultaneously. The model outputs a score indicating their similarity. While this approach achieves state-of-the-art accuracy because it allows for deep, token-level attention between the two sentences, it is extremely slow. For a database of n questions, finding the most similar one to a query requires n forward passes through the model, making it unsuitable for large-scale retrieval.

Bi-Encoder: To overcome the scalability issue, the Bi-Encoder (or Siamese) architecture was proposed. Two identical BERT models (with shared weights) process each sentence independently. This produces two fixed-size embeddings. The semantic similarity is then calculated using a distance metric like cosine similarity. As demonstrated by Sentence-BERT (Reimers and Gurevych, 2019), fine-tuning a BERT-based Siamese network on sentence-pair datasets creates embeddings that are highly effective for semantic search. This approach is much faster because all database embeddings can be computed offline. A query requires only one forward pass to generate its embedding, followed by a highly efficient vector similarity search. Our work is built upon this scalable Bi-Encoder paradigm.

2.4 Efficient Vector Search

The final piece of a large-scale retrieval system is the ability to search through millions or billions of vectors efficiently. A brute-force search is too slow. Approximate Nearest Neighbor (ANN) search algorithms provide a solution. Libraries like FAISS (Johnson et al., 2019), ScaNN (Guo et al., 2020), and Annoy offer data structures and algorithms (e.g., Inverted File Indexes, Product Quantization) that enable sub-linear time search with a minimal loss in accuracy. Our use of FAISS is a practical application of this line of work.

3 Methodology

Our system is designed as an end-to-end semantic retrieval pipeline, depicted in Figure 1. It consists of an offline indexing stage and an online retrieval stage.

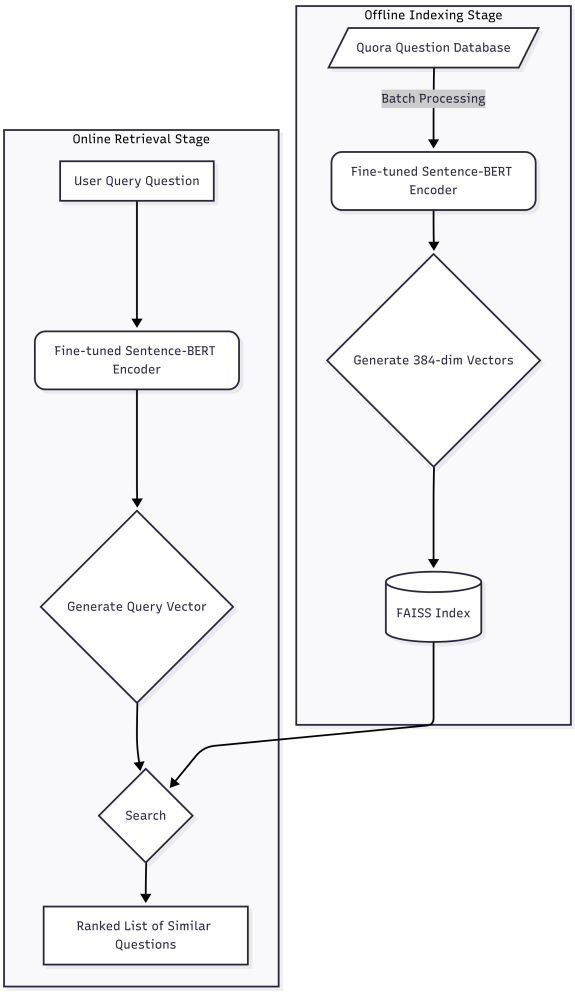

Figure 1: System Architecture. **Offline**: The Bi-Encoder processes all database questions into vectors, which are indexed by FAISS. **Online**: A new query is encoded by the same model, and FAISS retrieves the nearest neighbors from the index.

3.1 Bi-Encoder Model

We employ a Bi-Encoder network using the ‘sentence-transformers/all-MiniLM-L6-v2‘ model. This is a distilled version of BERT, having only 6 layers (‘L6‘), making it significantly faster than ‘BERT-base‘ while retaining strong performance. The model takes a variable-length sentence as input and outputs a fixed-size 384-dimensional embedding. This is achieved by feeding the contextualized output embeddings from the Transformer layers into a pooling layer (typically mean-pooling) to produce a single vector representing the entire sentence.

3.2 Fine-Tuning Strategy

While the base model is proficient at general semantic understanding, fine-tuning is crucial for adapting it to the specific linguistic style and nuances of the Quora dataset. We formulate this as a metric learning problem and use Triplet Loss for training.

Triplet Construction: From the Quora Question Pairs dataset, we generate training triplets (q_a, q_p, q_n), where:

- q_a (anchor): A question from the dataset.
- q_p (positive): A different question from the dataset that is a known duplicate of q_a.
- q_n (negative): A random question from the dataset that is not a duplicate of q_a.

This process creates a large set of training examples that explicitly teach the model what constitutes a similar and a dissimilar pair in this domain.

Triplet Loss Function: The goal of Triplet Loss is to structure the embedding space $\mathbb{R}^d$ such that the distance between an anchor and a positive is smaller than the distance between the anchor and a negative, by at least a certain margin α. The loss is defined as:

$$L(q_a, q_p, q_n) = \max\left(0, ||E_a - E_p||_2^2 - ||E_a - E_n||_2^2 + \alpha\right) \tag{1}$$

Here, E_a, E_p, E_n are the embeddings of the anchor, positive, and negative questions, respectively. $||\cdot||_2^2$ denotes the squared Euclidean distance. The margin α is a hyperparameter that enforces a separation between positive and negative pairs. Minimizing this loss function "pulls" positive pairs closer together and "pushes" negative pairs further apart in the embedding space.

3.3 Inference and Scalable Retrieval with FAISS

Once the model is fine-tuned, we create a retrieval index.

1. **Offline Indexing**: We perform a single forward pass for every question in our knowledge base (the Quora dataset) to generate its embedding. These 297,750 vectors are then stored in a FAISS index. For this project, we use an ‘IndexFlatIP‘, which performs an exact, exhaustive search based on Inner Product (which is equivalent to cosine similarity for normalized vectors). While brute-force, this is feasible for the dataset size and provides a perfect accuracy baseline. For larger datasets, an

approximate index like ‘IndexIVFPQ‘ would be used.

2. **Online Querying**: When a new query question arrives, we encode it using the fine-tuned model to get its vector. This query vector is then used to search the FAISS index, which rapidly returns the IDs and similarity scores of the top-k most similar questions from the database.

4 Experiments

4.1 Dataset

We use the Quora Question Pairs dataset, which is a standard benchmark for this task. It contains over 400,000 question pairs.

- **Total Questions**: Over 537,000 unique questions.
- **Total Pairs**: Over 404,000 labeled pairs.
- **Duplicate Ratio**: Approximately 37% of the pairs in the training set are labeled as duplicates.

We split the data to create a training set for fine-tuning, a knowledge base for indexing (297,750 questions), and a held-out test set for evaluation (44,663 queries).

4.2 Implementation Details

- **Frameworks**: PyTorch 1.12, Sentence-Transformers 2.2.2, Faiss-gpu 1.7.2.
- **Model**: ‘sentence-transformers/all-MiniLM-L6-v2‘.
- **Optimizer**: AdamW with a learning rate of 2×10^{-5}.
- **Training**: Fine-tuned for 1 full epoch on 72,810 triplets. The average loss for the epoch was 3.7671.
- **Hardware**: The experiment was run on a machine equipped with a CUDA-enabled GPU.

4.3 Evaluation Metrics

We assess our system’s performance using standard top-k information retrieval metrics: Recall@k, Precision@k, Mean Reciprocal Rank (MRR), Mean Average Precision (MAP@k), and Normalized Discounted Cumulative Gain (NDCG@k). These metrics evaluate the quality of the ranked list of retrieved questions.

4.4 Results and Analysis

4.4.1 Quantitative Analysis

The overall performance of our fine-tuned system is presented in Table 1. The system achieves a strong **MRR of 0.6401**. This is a key metric, indicating that on average, the first correct duplicate is found very close to the top of the ranked list (between rank 1 and 2).

k	Recall	Precision	MAP	NDCG
1	0.5065	0.5065	0.5065	0.5065
3	0.7358	0.3298	0.6039	0.6391
5	0.8226	0.2295	0.6120	0.6697
10	0.9115	0.1331	0.6045	0.6895

Table 1: Top-k retrieval performance on the Quora test set. The overall MRR for the system is **0.6401**.

The Recall@k values are particularly encouraging. With Recall@1 at 0.5065, our system returns a correct duplicate as the very top result for over 50

The MAP and NDCG scores, which evaluate the entire ranking quality, remain consistently high, further validating the model’s ability to rank relevant items above irrelevant ones. Figure 2 visualizes this performance.

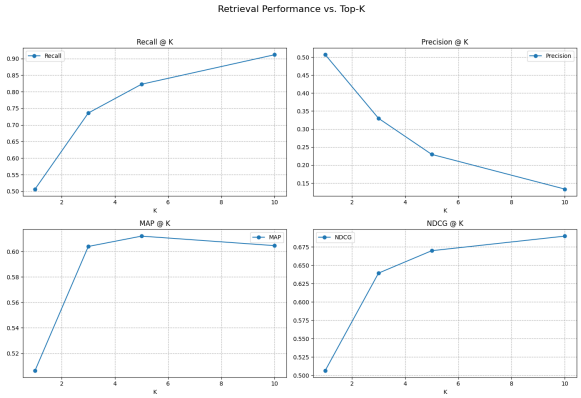

Figure 2: Visualization of key retrieval metrics (Recall@k, Precision@k, NDCG@k) as a function of k.

4.4.2 Qualitative Analysis (Error Analysis)

To better understand the model’s limitations, we manually inspected cases where the system failed. The errors can be categorized as follows:

- **Subtle Semantic Differences**: The model sometimes struggles with pairs that are lexically very similar but semantically different.

For example, "What are the best travel destinations in June?" vs. "What were the best travel destinations last June?". The model may incorrectly rank these as highly similar due to high word overlap.

- **World Knowledge and Commonsense**: Questions requiring external knowledge can be challenging. For example, "Who was the US president before Donald Trump?" and "Who was the US president after George W. Bush?". The model might not recognize these as referring to the same person (Barack Obama) unless such specific facts were present in its training data.

- **Negation and Complex Syntax**: Sentences with complex negation or conditional clauses can confuse the model. For instance, "What should I not do if I want to lose weight?" vs. "What should I do to lose weight?". While related, they are not duplicates, and the model might struggle with the negation.

4.5 Baseline Comparison

To contextualize the performance of our fine-tuned Sentence-BERT model, we compared it against a strong traditional baseline: TF-IDF with Cosine Similarity.

TF-IDF Baseline: We represented each question in the database and each test query as a vector using Term Frequency-Inverse Document Frequency (TF-IDF). TF-IDF is a classical information retrieval method that weighs words based on their frequency in a document and their rarity across the entire corpus. It excels at lexical matching but lacks true semantic understanding. Similarity scores were computed using the cosine similarity between the resulting sparse vectors.

Results: The performance comparison is presented in Table 2. The results clearly demonstrate the superiority of the fine-tuned semantic model. While the TF-IDF baseline achieves a respectable MRR of 0.4482, primarily by matching keywords in questions, our Sentence-BERT model outperforms it by a significant margin across all metrics. This highlights the critical advantage of semantic understanding over lexical matching for a task rich in paraphrasing like duplicate question detection.

Metric	TF-IDF Baseline	Our Model
MRR	0.4482	**0.6401**
Recall@1	0.3156	**0.5065**
Recall@5	0.6319	**0.8226**
Recall@10	0.7428	**0.9115**

Table 2: Performance comparison between the TF-IDF baseline and our fine-tuned Sentence-BERT model.

5 Conclusion

In this project, we successfully built and evaluated an effective, scalable semantic retrieval system for duplicate question detection. By fine-tuning a Sentence-BERT Bi-Encoder on the Quora dataset with Triplet Loss and integrating the FAISS library for fast retrieval, we achieved strong performance, exemplified by an MRR of 0.6401. This work demonstrates a practical and robust solution to a common problem in CQA platforms.

For future work, several avenues could be explored. First, employing more advanced triplet mining strategies, such as hard negative mining, could further improve the model's discriminative ability. Second, using larger and more powerful base models might yield better performance, at the cost of computational resources. Finally, implementing a two-stage re-ranking system, where our fast Bi-Encoder retrieves a set of candidates that are then re-ranked by a more accurate but slower Cross-Encoder, could potentially combine the best of both worlds in terms of speed and accuracy.

Limitations

The performance of our model is inherently tied to the quality and scope of the Quora dataset. It may not generalize well to questions from highly specialized domains (e.g., legal or medical texts) without further fine-tuning. The model does not possess true commonsense reasoning and can be misled by adversarially crafted questions or complex linguistic phenomena. Lastly, the system's effectiveness relies on a pre-existing knowledge base; it is designed to find existing duplicates, not to generate answers to novel questions.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the*

North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186.

Ruiqi Guo, Philip Sun, Erik Lindgren, Quan Geng, David Simcha, Felix Chern, and Sanjiv Kumar. 2020. Accelerating large-scale inference with anisotropic vector quantization. In *International Conference on Machine Learning*, pages 3887–3897. PMLR.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.