

Justification of Choices & Explanation of Approach

1. Choice of Models

To solve the Question Pair Similarity Classification task, I experimented with multiple model families, each capturing different levels of semantic information. The decision to use a progression of models—from simple baselines to advanced neural architectures—allows for a comprehensive comparison and ensures that the final approach is both justifiable and optimal.

A. Logistic Regression (TF-IDF Baseline)

This baseline model establishes a lower bound for performance.

Why this model?

- Simple and fast to train
- Provides interpretable results
- Good benchmark for text classification
- Helps determine how much more powerful deep learning models are in comparison

B. ANN with Embedding + Dense Layers

This model captures non-linear relationships between words.

Why this model?

- Learns dense embeddings over vocabulary
- Faster and lighter compared to recurrent networks
- Provides a mid-range benchmark between logistic regression and LSTM/Siamese models

C. Siamese LSTM Network

This model processes both questions through shared LSTM encoders and compares their latent vectors.

Why this model?

- Suitable for pairwise semantic similarity

- Shared weights enforce meaning-based comparison
- LSTM is effective for long-term dependencies in language
- The “distance + interaction” features (|diff|, mult) capture strong semantic signals

This model directly aligns with the requirements of the task (“use ANN / Siamese / LSTM”).

D. SBERT + MLP (Transfer Learning)

Sentence-BERT provides high-quality sentence embeddings optimized for semantic similarity tasks.

Why this model?

- Uses transformer knowledge without full fine-tuning
- Dramatically improves semantic understanding over TF-IDF
- Fast inference
- Allows classical ML (MLP) to be built on top of rich embeddings

Because SBERT outputs high-dimensional representations (1536 features after feature engineering), I applied PCA to reduce dimensionality to 256. This speeds up training, prevents memory issues, and improves generalization.

2. Justification of Evaluation Metrics

A. Accuracy

Useful for understanding general performance but alone can be misleading due to dataset imbalance (non-duplicate pairs are more common).

Therefore, accuracy was **not the only metric**.

B. F1-Score (Primary Metric)

The task involves identifying duplicate questions, which are fewer in number.

F1-score is preferred because:

- Balances precision and recall
- Rewards correctly predicting the minority class
- Reflects practical performance better than accuracy

C. Precision & Recall

- **Precision** ensures predicted duplicates are actually true duplicates
- **Recall** ensures the model does not miss duplicate questions

Duplicate detection is a real-world semantic search/retrieval problem, where recall is often important.

D. Confusion Matrix

Used to visualize where models make mistakes:

- False positives = predicting duplicates when they aren't
- False negatives = missing real duplicates

These errors have different real-world consequences, so matrix inspection is essential.

3. Justification of Preprocessing Pipeline

A. Text Cleaning

Lowercasing, removing punctuation, lemmatization, and stopword removal reduce noise.

B. Tokenization + Padding

Required for LSTM and ANN models that need fixed-length sequences.

C. TF-IDF

Captures discriminative word importance for classical ML baselines.

D. SBERT Embeddings

Provides contextualized, transformer-level semantic understanding.

This layered approach ensures that each model receives a form of input best suited to its architecture.

4. Justification of Tuning Steps

A. Baseline Models

Grid search and randomized search were used for:

- Logistic Regression (C, penalty)
- SVM (C, gamma)

These models benefit significantly from lightweight classical hyperparameter tuning.

B. Siamese LSTM

Tuning involved:

- Embedding dimension
- LSTM units
- Dropout
- Learning rate

I prioritized:

- Reducing overfitting
- Maintaining shared encoder balance
- Ensuring stable gradient behavior

C. SBERT + PCA + MLP

The biggest bottleneck was feature dimensionality.

To optimize:

1. PCA reduced 1536 \rightarrow 256 dimensions
2. Hidden layer sizes tuned for stability
3. Max_iter reduced to avoid slow convergence
4. Batch size & learning rate chosen based on training curve behavior

This significantly improved runtime and model stability.

5. Overall Approach Summary

1. **Start simple:** baseline with TF-IDF + LR
2. **Move to neural networks:** ANN and Siamese LSTM
3. **Use transformer embeddings:** SBERT
4. **Optimize for performance:** PCA, tuned hyperparameters
5. **Evaluate thoroughly:** accuracy, F1-score, confusion matrix

This tiered approach ensures:

- Strong justification
- Measurable progress
- Clear reasoning for final model choice

6. Final Notes

The combination of SBERT embeddings and a lightweight MLP (after PCA) resulted in:

- Best overall F1-score
- Fast inference
- Good generalization

Meanwhile, the Siamese LSTM provided a deep-learning architecture that learns pairwise semantic similarity directly from sequences, justifying its inclusion per task requirements.