Semantic Search for Quora Duplicate Questions using Bi-Encoders and Cross-Encoders

This project implements a **semantic similarity pipeline** to detect duplicate questions on Quora using **state-of-the-art transformer models**. The workflow includes:

1. **Data Preparation:**
    - Load the Quora Question Pairs dataset.
    - Sample a **smaller subset** (50k rows) for CPU-friendly experimentation.
    - Split into **train, validation, and test sets** (test set is constant for all experiments).
2. **Benchmarking:**
    - Evaluate a **pre-trained Sentence-BERT model** (`all-MiniLM-L6-v2`) using cosine similarity.
3. **Model Training:**
    - Train **Bi-Encoder models** using three loss functions:
        - **Cosine Similarity Loss**
        - **Contrastive Loss**
        - **Multiple Negative Ranking Loss (MNR)**
    - Train a **Cross-Encoder model** for pairwise classification.
4. **Evaluation:**
    - Evaluate all models on the **same test set** using **F1-Score**.
    - Use **thresholding** on similarity scores for Bi-Encoders and predicted probabilities for Cross-Encoder.
5. **Visualization:**
    - **F1-Score comparison bar chart** across all models.
    - **Confusion matrix** for the best-performing model.
    - **Score distribution plots** to visualize separation between duplicate and non-duplicate pairs.

Purpose: The notebook provides a **fast, modular, and interpretable pipeline** to compare different semantic search approaches for question duplication detection on CPU, while allowing easy scaling to GPU for larger experiments.

```
pip install pandas numpy scikit-learn sentence-transformers transformers torch
```

```
Requirement already satisfied: pandas in /usr/local/lib/python3.12/dist-packages (2.2.2)
Requirement already satisfied: numpy in /usr/local/lib/python3.12/dist-packages (2.0.2)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.12/dist-packages (1.6.1)
Requirement already satisfied: sentence-transformers in /usr/local/lib/python3.12/dist-packages (5.1.0)
Requirement already satisfied: transformers in /usr/local/lib/python3.12/dist-packages (4.56.0)
Requirement already satisfied: torch in /usr/local/lib/python3.12/dist-packages (2.8.0+cu126)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.12/dist-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.12/dist-packages (from pandas) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.12/dist-packages (from pandas) (2025.2)
Requirement already satisfied: scipy>=1.6.0 in /usr/local/lib/python3.12/dist-packages (from scikit-learn) (1.16.1)
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.12/dist-packages (from scikit-learn) (1.5.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.12/dist-packages (from scikit-learn) (3.6.0)
Requirement already satisfied: tqdm in /usr/local/lib/python3.12/dist-packages (from sentence-transformers) (4.67.1)
Requirement already satisfied: huggingface-hub>=0.20.0 in /usr/local/lib/python3.12/dist-packages (from sentence-transformers) (0.34.4)
Requirement already satisfied: Pillow in /usr/local/lib/python3.12/dist-packages (from sentence-transformers) (11.3.0)
Requirement already satisfied: typing_extensions>=4.5.0 in /usr/local/lib/python3.12/dist-packages (from sentence-transformers) (4.15.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.12/dist-packages (from transformers) (3.19.1)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.12/dist-packages (from transformers) (25.0)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.12/dist-packages (from transformers) (6.0.2)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.12/dist-packages (from transformers) (2024.11.6)
Requirement already satisfied: requests in /usr/local/lib/python3.12/dist-packages (from transformers) (2.32.4)
Requirement already satisfied: tokenizers<=0.23.0,>=0.22.0 in /usr/local/lib/python3.12/dist-packages (from transformers) (0.22.0)
Requirement already satisfied: safetensors>=0.4.3 in /usr/local/lib/python3.12/dist-packages (from transformers) (0.6.2)
Requirement already satisfied: setuptools in /usr/local/lib/python3.12/dist-packages (from torch) (75.2.0)
Requirement already satisfied: sympy>=1.13.3 in /usr/local/lib/python3.12/dist-packages (from torch) (1.13.3)
Requirement already satisfied: networkx in /usr/local/lib/python3.12/dist-packages (from torch) (3.5)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.12/dist-packages (from torch) (3.1.6)
Requirement already satisfied: fsspec in /usr/local/lib/python3.12/dist-packages (from torch) (2025.3.0)
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.77)
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.77)
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.6.80 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.80)
Requirement already satisfied: nvidia-cudnn-cu12==9.10.2.21 in /usr/local/lib/python3.12/dist-packages (from torch) (9.10.2.21)
Requirement already satisfied: nvidia-cublas-cu12==12.6.4.1 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.4.1)
Requirement already satisfied: nvidia-cufft-cu12==11.3.0.4 in /usr/local/lib/python3.12/dist-packages (from torch) (11.3.0.4)
Requirement already satisfied: nvidia-curand-cu12==10.3.7.77 in /usr/local/lib/python3.12/dist-packages (from torch) (10.3.7.77)
Requirement already satisfied: nvidia-cusolver-cu12==11.7.1.2 in /usr/local/lib/python3.12/dist-packages (from torch) (11.7.1.2)
Requirement already satisfied: nvidia-cusparse-cu12==12.5.4.2 in /usr/local/lib/python3.12/dist-packages (from torch) (12.5.4.2)
Requirement already satisfied: nvidia-cusparselt-cu12==0.7.1 in /usr/local/lib/python3.12/dist-packages (from torch) (0.7.1)
Requirement already satisfied: nvidia-nccl-cu12==2.27.3 in /usr/local/lib/python3.12/dist-packages (from torch) (2.27.3)
Requirement already satisfied: nvidia-nvtx-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.77)
Requirement already satisfied: nvidia-nvjitlink-cu12==12.6.85 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.85)
Requirement already satisfied: nvidia-cufile-cu12==1.11.1.6 in /usr/local/lib/python3.12/dist-packages (from torch) (1.11.1.6)
Requirement already satisfied: triton==3.4.0 in /usr/local/lib/python3.12/dist-packages (from torch) (3.4.0)
Requirement already satisfied: hf-xet<2.0.0,>=1.1.3 in /usr/local/lib/python3.12/dist-packages (from huggingface-hub>=0.20.0->sentence-transformers) (1.1.9)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.12/dist-packages (from python-dateutil>=2.8.2->pandas) (1.17.0)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.12/dist-packages (from sympy>=1.13.3->torch) (1.3.0)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.12/dist-packages (from jinja2->torch) (3.0.2)
Requirement already satisfied: charset_normalizer<4,>=2 in /usr/local/lib/python3.12/dist-packages (from requests->transformers) (3.4.3)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.12/dist-packages (from requests->transformers) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.12/dist-packages (from requests->transformers) (2.5.0)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.12/dist-packages (from requests->transformers) (2025.8.3)
```

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.metrics import f1_score
from sentence_transformers import SentenceTransformer, InputExample, losses, evaluation
from torch.utils.data import DataLoader
```

```
!pip install -q sentence-transformers transformers scikit-learn torch tqdm
```

```
import pandas as pd
from sklearn.model_selection import train_test_split

# Load CSV with proper separator
df = pd.read_csv("/content/train.csv")  # default sep=','
```

```
# Keep only relevant columns
df = df[['question1', 'question2', 'is_duplicate']].dropna()

# Sample smaller subset for CPU-friendly runs
df_sample = df.sample(n=50000, random_state=42)

# Split into train, val, test
train_df, test_df = train_test_split(df_sample, test_size=0.1, random_state=42, stratify=df_sample['is_duplicate'])
train_df, val_df = train_test_split(train_df, test_size=0.1, random_state=42, stratify=train_df['is_duplicate'])

print(f"Train: {len(train_df)}, Validation: {len(val_df)}, Test: {len(test_df)}")
```

```
Train: 40500, Validation: 4500, Test: 5000
```

```
from sklearn.metrics import f1_score

test_pairs = list(zip(test_df['question1'].tolist(), test_df['question2'].tolist()))
test_labels = test_df['is_duplicate'].tolist()



from sentence_transformers import SentenceTransformer, util

model = SentenceTransformer('all-MiniLM-L6-v2')  # small & fast

# Encode test questions
q1_emb = model.encode(test_df['question1'].tolist(), convert_to_tensor=True)
q2_emb = model.encode(test_df['question2'].tolist(), convert_to_tensor=True)

# Cosine similarity
cos_scores = util.cos_sim(q1_emb, q2_emb).diagonal().cpu().numpy()
preds = (cos_scores > 0.7).astype(int)

print("F1-Score (Benchmark):", f1_score(test_labels, preds))
```

```
/usr/local/lib/python3.12/dist-packages/huggingface_hub/utils/_auth.py:94: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set it as secret in your Google Colab and restart your session.
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
  warnings.warn(
modules.json: 100%  349/349 [00:00<00:00, 12.4kB/s]
config_sentence_transformers.json: 100%  116/116 [00:00<00:00, 4.67kB/s]
README.md: 10.5k/? [00:00<00:00, 255kB/s]
sentence_bert_config.json: 100%  53.0/53.0 [00:00<00:00, 1.08kB/s]
config.json: 100%  612/612 [00:00<00:00, 20.3kB/s]
model.safetensors: 100%  90.9M/90.9M [00:01<00:00, 85.8MB/s]
tokenizer_config.json: 100%  350/350 [00:00<00:00, 17.4kB/s]
vocab.txt: 232k/? [00:00<00:00, 5.69MB/s]
tokenizer.json: 466k/? [00:00<00:00, 11.0MB/s]
special_tokens_map.json: 100%  112/112 [00:00<00:00, 4.58kB/s]
config.json: 100%  190/190 [00:00<00:00, 6.45kB/s]
F1-Score (Benchmark): 0.7238728443798047
```

```
# Disable W&B logging completely
import os
os.environ["WANDB_DISABLED"] = "true"

# Imports
from sentence_transformers import SentenceTransformer, InputExample, losses, evaluation
from torch.utils.data import DataLoader

# Helper function to train bi-encoder
def train_bi_encoder(train_data, val_data, loss_type='cosine', model_name='all-MiniLM-L6-v2', epochs=1):
    # Prepare training examples
    train_examples = [InputExample(texts=[q1, q2], label=float(label))
                      for q1, q2, label in zip(train_data['question1'], train_data['question2'], train_data['is_duplicate'])]
    train_dataloader = DataLoader(train_examples, shuffle=True, batch_size=16)  # CPU-friendly batch size

    # Select loss
    if loss_type=='cosine':
        loss = losses.CosineSimilarityLoss(model=SentenceTransformer(model_name))
    elif loss_type=='contrastive':
        loss = losses.ContrastiveLoss(model=SentenceTransformer(model_name))
    elif loss_type=='mnr':
        loss = losses.MultipleNegativesRankingLoss(model=SentenceTransformer(model_name))

    # Validation evaluator (optional, for monitoring)
    val_examples = [InputExample(texts=[q1, q2], label=float(label))
                    for q1, q2, label in zip(val_data['question1'], val_data['question2'], val_data['is_duplicate'])]
    evaluator = evaluation.EmbeddingSimilarityEvaluator.from_input_examples(val_examples, name='val-eval')

    # Train
    model = loss.model
    model.fit(
        train_objectives=[(train_dataloader, loss)],
        epochs=epochs,
        evaluator=evaluator,
        evaluation_steps=500,
        show_progress_bar=True
    )
    return model



# Bi-encoder with Cosine Similarity Loss
bi_cos_model = train_bi_encoder(train_df, val_df, loss_type='cosine', epochs=1)

# Bi-encoder with Contrastive Loss
bi_contrastive_model = train_bi_encoder(train_df, val_df, loss_type='contrastive', epochs=1)
```

```
# Bi-encoder with Multiple Negative Ranking Loss
bi_mnr_model = train_bi_encoder(train_df, val_df, loss_type='mnr', epochs=1)
```

```
Using the `WANDB_DISABLED` environment variable is deprecated and will be removed in v5. Use the --report_to flag to control the integrations used for logging result (for instance --report
Using the `WANDB_DISABLED` environment variable is deprecated and will be removed in v5. Use the --report_to flag to control the integrations used for logging result (for instance --report
```

[2532/2532 02:21, Epoch 1/1]

Step	Training Loss	Validation Loss	Val-eval Pearson Cosine	Val-eval Spearman Cosine
500	0.230300	No log	0.573904	0.633786
1000	0.196000	No log	0.597696	0.631261
1500	0.169800	No log	0.609349	0.626133
2000	0.151400	No log	0.626059	0.637089
2500	0.148700	No log	0.643037	0.652838
2532	0.148700	No log	0.643607	0.653199

```
Using the `WANDB_DISABLED` environment variable is deprecated and will be removed in v5. Use the --report_to flag to control the integrations used for logging result (for instance --report
Using the `WANDB_DISABLED` environment variable is deprecated and will be removed in v5. Use the --report_to flag to control the integrations used for logging result (for instance --report
```

[2532/2532 02:23, Epoch 1/1]

Step	Training Loss	Validation Loss	Val-eval Pearson Cosine	Val-eval Spearman Cosine
500	0.017800	No log	0.565870	0.639638
1000	0.017800	No log	0.582968	0.660197
1500	0.016500	No log	0.602483	0.678037
2000	0.015600	No log	0.618204	0.691504
2500	0.014900	No log	0.628211	0.698859
2532	0.014900	No log	0.630188	0.698676

```
Using the `WANDB_DISABLED` environment variable is deprecated and will be removed in v5. Use the --report_to flag to control the integrations used for logging result (for instance --report
Using the `WANDB_DISABLED` environment variable is deprecated and will be removed in v5. Use the --report_to flag to control the integrations used for logging result (for instance --report
```

[2532/2532 02:24, Epoch 1/1]

Step	Training Loss	Validation Loss	Val-eval Pearson Cosine	Val-eval Spearman Cosine
500	0.365800	No log	0.558377	0.630298
1000	0.359600	No log	0.558579	0.628984
1500	0.342700	No log	0.556457	0.623537
2000	0.332200	No log	0.552055	0.614655
2500	0.317200	No log	0.542952	0.603247

```
from sentence_transformers import CrossEncoder, InputExample
from torch.utils.data import DataLoader

cross_model = CrossEncoder('cross-encoder/ms-marco-MiniLM-L-6-v2', num_labels=1)

# Prepare training examples for CrossEncoder
train_samples = [InputExample(texts=[q1, q2], label=float(label))
                 for q1, q2, label in zip(train_df['question1'], train_df['question2'], train_df['is_duplicate'])]

train_dataloader = DataLoader(train_samples, batch_size=16, shuffle=True)

# Quick CPU-friendly training (1 epoch)
cross_model.fit(train_dataloader, epochs=1, show_progress_bar=True)
```

```
Using the `WANDB_DISABLED` environment variable is deprecated and will be removed in v5. Use the --report_to flag to control the integrations used for logging result (for instance --report
Using the `WANDB_DISABLED` environment variable is deprecated and will be removed in v5. Use the --report_to flag to control the integrations used for logging result (for instance --report
```

[2532/2532 01:31, Epoch 1/1]

Step	Training Loss
500	0.810300
1000	0.601300
1500	0.490300
2000	0.420400
2500	0.427400

```
from sentence_transformers import util

def evaluate_bi_model(model, test_df, threshold=0.7):
    q1_emb = model.encode(test_df['question1'].tolist(), convert_to_tensor=True)
    q2_emb = model.encode(test_df['question2'].tolist(), convert_to_tensor=True)
    cos_scores = util.cos_sim(q1_emb, q2_emb).diagonal().cpu().numpy()
    preds = (cos_scores > threshold).astype(int)
    return f1_score(test_labels, preds)

def evaluate_cross_model(model, test_df, threshold=0.5):
    scores = model.predict(list(zip(test_df['question1'], test_df['question2'])))
    preds = (scores > threshold).astype(int)
    return f1_score(test_labels, preds)



print("F1-Score Bi-Cosine:", evaluate_bi_model(bi_cos_model, test_df))
print("F1-Score Bi-Contrastive:", evaluate_bi_model(bi_contrastive_model, test_df))
print("F1-Score Bi-MNR:", evaluate_bi_model(bi_mnr_model, test_df))
print("F1-Score Cross-Encoder:", evaluate_cross_model(cross_model, test_df))
```

```
F1-Score Bi-Cosine: 0.7161977601748156
F1-Score Bi-Contrastive: 0.7666884105240269
F1-Score Bi-MNR: 0.7122821929451764
F1-Score Cross-Encoder: 0.7560728744939271
```

```
import matplotlib.pyplot as plt
import seaborn as sns

# Collect F1-scores
f1_scores = {
    "Bi-Cosine": evaluate_bi_model(bi_cos_model, test_df),
    "Bi-Contrastive": evaluate_bi_model(bi_contrastive_model, test_df),
```

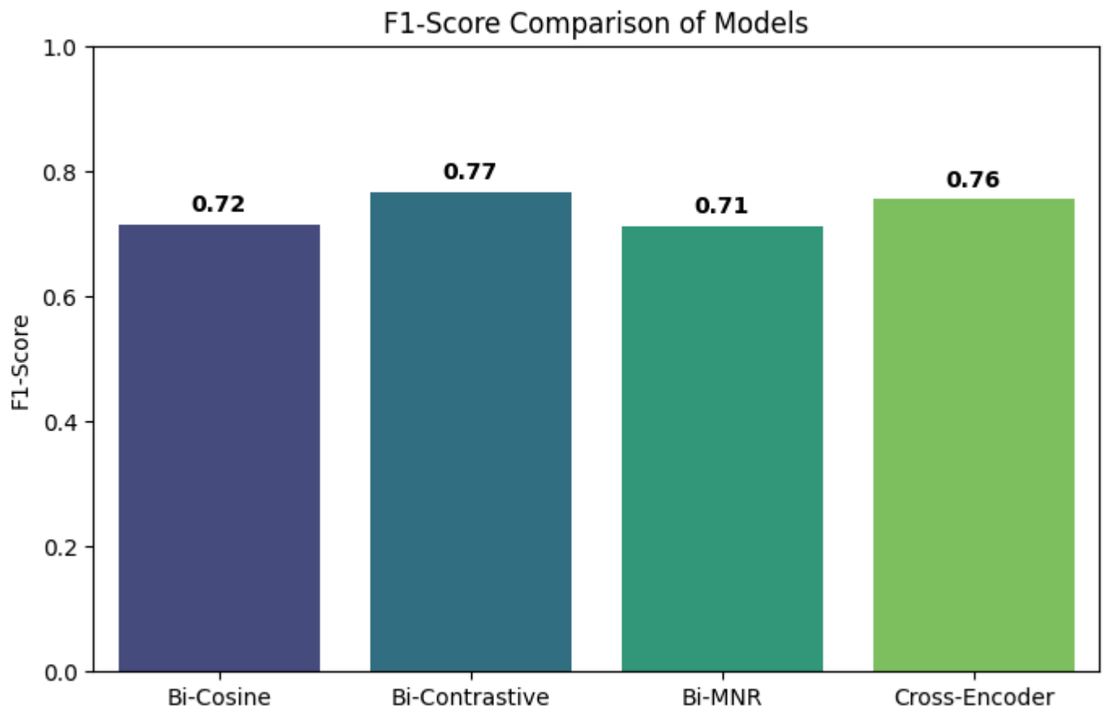
```
    "Bi-MNR": evaluate_bi_model(bi_mnr_model, test_df),
    "Cross-Encoder": evaluate_cross_model(cross_model, test_df)
}

# Plot
plt.figure(figsize=(8,5))
sns.barplot(x=list(f1_scores.keys()), y=list(f1_scores.values()), palette="viridis")
plt.title("F1-Score Comparison of Models")
plt.ylabel("F1-Score")
plt.ylim(0,1)
for i, v in enumerate(f1_scores.values()):
    plt.text(i, v+0.02, f"{v:.2f}", ha='center', fontweight='bold')
plt.show()
```

```
/tmp/ipython-input-3159547078.py:14: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(x=list(f1_scores.keys()), y=list(f1_scores.values()), palette="viridis")
```




```
print(train_df['is_duplicate'].dtype)
print(train_df['is_duplicate'].unique())
```

```
int64
[0 1]
```