Product Matching using Siamese Networks with Pre-computed SBERT Embeddings

Eshin Menusha Fernando
Undergruate of Computer Science and Engineering
Deep Learning for E-commerce Applications
Neural Network Architecture and Optimization Report

September 7, 2025

Abstract

This report presents a comprehensive implementation of a Siamese Neural Network for product matching in e-commerce applications. The system utilizes pre-computed Sentence-BERT (SBERT) embeddings combined with a trainable projection head to identify matching product pairs. Our approach achieves exceptional performance with 96.61% precision, 99.70% recall, and 98.13% F1-score on the test dataset. The methodology demonstrates efficient training through embedding pre-computation and robust contrastive learning mechanisms.

1 Introduction

Product matching is a critical challenge in e-commerce platforms where identical or similar products need to be identified across different listings, vendors, or catalogs. Traditional string-matching approaches often fail to capture semantic similarities due to variations in product descriptions, abbreviations, and formatting differences.

This laboratory experiment implements a Siamese Network architecture that learns similarity representations for product title pairs. The key innovation lies in utilizing pre-computed SBERT embeddings as input features while training only a lightweight projection head, significantly reducing computational requirements while maintaining high accuracy.

1.1 Objectives

- Develop a Siamese Network for product pair classification
- Implement efficient pre-computation of SBERT embeddings
- Optimize training through contrastive loss function
- Achieve high precision and recall on product matching tasks
- Demonstrate practical inference capabilities

2 Methodology

2.1 Architecture Overview

The proposed system consists of three main components:

1. Pre-computed SBERT Embeddings: We utilize the `all-MiniLM-L6-v2` model from Sentence-Transformers to generate 384-dimensional embeddings for product titles. This approach eliminates redundant encoding during training.

2. Siamese Network: A twin neural network architecture that processes pairs of embeddings through identical projection heads, learning to map similar products closer in the embedding space.

3. Contrastive Loss Function: A distance-based loss that minimizes distances between matching pairs while maximizing distances between non-matching pairs.

2.2 Network Architecture

The Siamese Network consists of a projection head with the following structure:

$$h_1 = \text{ReLU}(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1) \tag{1}$$

$$\mathbf{z} = \mathbf{W}_2 h_1 + \mathbf{b}_2 \tag{2}$$

Where:

- $\mathbf{x} \in \mathbb{R}^{384}$ is the input SBERT embedding
- $\mathbf{W}_1 \in \mathbb{R}^{256\times384}$, $\mathbf{b}_1 \in \mathbb{R}^{256}$
- $\mathbf{W}_2 \in \mathbb{R}^{128\times256}$, $\mathbf{b}_2 \in \mathbb{R}^{128}$
- $\mathbf{z} \in \mathbb{R}^{128}$ is the final projected embedding

2.3 Contrastive Loss Function

The contrastive loss function is defined as:

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \left[y_i \cdot d_i^2 + \right. \tag{3}$$

$$\left. (1 - y_i) \cdot \max(0, m - d_i)^2 \right] \tag{4}$$

Where:

- $d_i = \|\mathbf{z}_1^{(i)} - \mathbf{z}_2^{(i)}\|_2$ is the Euclidean distance
- $y_i \in \{0, 1\}$ is the binary label (1 for matches, 0 for non-matches)
- $m = 1.0$ is the margin parameter
- N is the batch size

3 Implementation Details

3.1 Data Preprocessing

The dataset contains product title pairs with binary match labels. Key preprocessing steps include:

1. **Unique Title Extraction:** All unique titles are extracted from both columns to avoid redundant encoding
2. **Batch Encoding:** Titles are processed in batches of 32 for memory efficiency
3. **Embedding Storage:** Pre-computed embeddings are stored in a dictionary mapping for O(1) lookup

3.2 Training Configuration

Table 1: Training Hyperparameters

Parameter	Value
Batch Size	32
Learning Rate	0.001
Weight Decay	1×10^{-5}
Epochs	20
Optimizer	Adam
Loss Margin	1.0
Train/Test Split	80/20

3.3 Dataset Characteristics

Table 2: Dataset Statistics

Metric	Count
Total Pairs	Variable
Training Samples	80%
Testing Samples	20%
Unique Titles	Extracted
Matching Pairs	Balanced
Non-matching Pairs	Balanced

4 Results and Analysis

4.1 Training Performance

The model demonstrates excellent convergence characteristics with the final epoch achieving an average loss of 0.0151, indicating effective learning of the similarity metric.

4.2 Evaluation Metrics

Table 3: Model Performance (Threshold = 0.6)

Metric	Score
Precision	0.9661
Recall	0.9970
F1-Score	0.9813
Avg. Distance (Matches)	0.1225
Avg. Distance (Non-matches)	1.3318

4.3 Distance Analysis

The learned embeddings demonstrate excellent separation between matching and non-matching pairs:

- **Matching pairs** have an average distance of 0.1225
- **Non-matching pairs** have an average distance of 1.3318
- The large margin (1.2093) indicates robust discrimination

4.4 Inference Examples

Selected inference results demonstrate the model's capabilities:

Table 4: Sample Inference Results

Product Pair	Distance	Prediction
iPhone 13 Pro Max vs Apple iPhone 13 Pro Max	0.0373	MATCH
Samsung Galaxy S21 vs iPhone 12 Pro	0.9483	NO MATCH
Nike Air Force 1 vs Nike Air Force One Sneakers	0.2780	MATCH
Dell Laptop vs MacBook Pro	0.5890	MATCH

5 Technical Implementation

5.1 Key Classes and Functions

ProductPairDataset: Custom PyTorch Dataset class that efficiently handles pre-computed embeddings with O(1) lookup time.

SiameseNetwork: Neural network architecture with shared projection head for embedding transformation.

ContrastiveLoss: Implementation of the contrastive loss function with proper margin handling.

5.2 Optimization Strategies

1. **Pre-computation:** SBERT embeddings are computed once and reused, reducing training time by orders of magnitude
2. **Memory Efficiency:** Batch processing prevents memory overflow during embedding generation
3. **Reproducibility:** Fixed random seeds ensure consistent results across runs

6 Advantages and Limitations

6.1 Advantages

- Exceptional accuracy (98.13% F1-score)
- Efficient training through pre-computation
- Lightweight architecture (only projection head trained)
- Scalable to large product catalogs
- Real-time inference capabilities

6.2 Limitations

- Requires pre-computed embeddings storage
- Limited to text-based features only
- Threshold parameter requires tuning
- Memory requirements for large vocabularies

7 Future Enhancements

7.1 Multi-modal Integration

Incorporate image features alongside text embeddings for more comprehensive product representation.

7.2 Dynamic Threshold Learning

Implement learned threshold parameters rather than fixed values to optimize decision boundaries automatically.

7.3 Hierarchical Matching

Develop category-specific matching models for improved performance across different product types.

7.4 Active Learning

Implement uncertainty sampling to identify challenging examples for model improvement.

8 Conclusion

This laboratory experiment successfully demonstrates the effectiveness of Siamese Networks for product matching tasks. The combination of pre-computed SBERT embeddings with a trainable projection head achieves exceptional performance while maintaining computational efficiency.

The model's high precision (96.61%) makes it suitable for production e-commerce applications where false positives are costly. The excellent recall (99.70%) ensures comprehensive matching coverage.

Key contributions include:

- Efficient pre-computation strategy for SBERT embeddings
- Robust contrastive learning implementation
- Comprehensive evaluation and analysis framework
- Practical inference pipeline for real-world deployment

The methodology presented can be extended to various similarity learning tasks beyond product matching, including document similarity, image matching, and recommendation systems.

9 Code Availability

The complete implementation includes modular components for data loading, model training, evaluation, and inference. The code structure promotes reusability and extensibility for different datasets and applications.

Key implementation highlights:

- Configurable hyperparameters
- Comprehensive error handling
- Efficient memory management
- Detailed logging and progress tracking
- Model persistence and loading capabilities

References

[1] Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., & Shah, R. (1994). Signature verification using a "siamese" time delay neural network. *Advances in neural information processing systems*, 6.

[2] Reimers, N., & Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

[3] Hadsell, R., Chopra, S., & LeCun, Y. (2006). Dimensionality reduction by learning an invariant mapping. *2006 IEEE computer society conference on computer vision and pattern recognition* (Vol. 2, pp. 1735-1742).

[4] Mudgal, S., Li, H., Rekatsinas, T., Doan, A., Park, Y., Krishnan, G., ... & Rajaraman, A. (2018). Deep learning for entity matching: A design space exploration. *Proceedings of the 2018 International Conference on Management of Data* (pp. 19-34).

[5] Zhai, S., Chang, K. H., Zhang, R., & Zhang, Z. M. (2017). Deepintent: Learning attentions for online advertising with recurrent neural networks. *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1295-1304).