Semantic Similarity Detection: From Traditional Models to Advanced SBERT
CS505 project: milestone4

Jingyi Zhang
jyz0328@bu.edu

Ruoxi Jin
jrx99@bu.edu

Abstract

This study compares traditional models (Bag-of-Words and TF-IDF) with a modern contextual model (Sentence-BERT, SBERT) in semantic similarity detection, addressing the limitations of traditional approaches in capturing deep semantics. By measuring semantic similarity between sentence pairs (e.g., "A plane is taking off." and "An airplane is taking off.") and outputting scores (e.g., 0.85), it supports applications such as information retrieval, question answering, machine translation evaluation, and plagiarism detection.

Using the STS Benchmark dataset, models were evaluated with metrics including F1-Score, Accuracy, Spearman Correlation, and Pearson Correlation. Results show SBERT outperforms baseline(BoW and TF-IDF) models across all metrics, with superior semantic understanding and prediction stability. Analysis of the predicted-to-true score ratio distribution further confirms SBERT's reliability in complex tasks.

This study provides strong support for semantic modeling in natural language processing tasks and offers practical insights for real-world applications in the field.

Background&Motivation

Semantic similarity plays a critical role in natural language processing (NLP), forming the backbone of tasks such as text comparison, machine translation, and information retrieval. Traditional methods, such as **Bag-of-Words (BoW)** and **TF-IDF (Term Frequency-Inverse Document Frequency)**, typically rely on word frequency statistics or keyword matching. These methods represent sentences as vectors and compute similarity using metrics such as cosine similarity. While effective for simpler tasks, they fail to capture deeper semantic relationships and contextual meaning.

As illustrated in Figure 1, traditional methods face two major limitations:

- They misjudge *lexically similar but semantically different* sentences, such as “How are you?” vs. “How old are you?”, which share similar words but differ in meaning.
- They fail to recognize *semantically similar yet lexically distinct* sentences. For example, “How old are you?” and “What is your age?” have no overlapping words, but they express the same semantic intent.

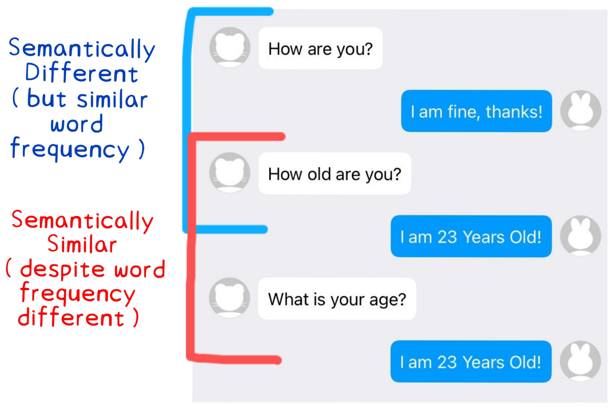

Semantically Different (but similar word frequency)
Semantically Similar (despite word frequency different)
How are you?
I am fine, thanks!
How old are you?
I am 23 Years Old!
What is your age?
I am 23 Years Old!

Figure 1: an example shows semantically different(but word frequency similar) sentences and semantically similar(despite word frequency different)sentences

These limitations highlight the inadequacy of traditional methods in capturing deep semantic understanding, especially for complex NLP tasks requiring contextual and holistic sentence interpretation. To address these challenges, we explore advanced deep learning models, such as **Sentence-BERT (SBERT)**, which leverage contextual embeddings and semantic modeling to achieve superior performance in identifying sentence relationships.

Research Goals

The primary goal of this research is to explore and improve methods for measuring sentence semantic similarity, aiming to bridge the gap between traditional approaches and modern contextual models. The specific objectives include:

- **Performance Evaluation:** Evaluate baseline models (BoW and TF-IDF) and the improved model (SBERT) on semantic similarity tasks using the STS Benchmark dataset. Analyze results with metrics like Pearson Correlation, Spearman Correlation, Accuracy, and F1-Score.

- **Limitations Analysis:** Identify the weaknesses of traditional models, focusing on their inability to capture deep semantics and context, and explain their struggles with complex tasks.

- **Model Improvement:** Implement and evaluate SBERT as an advanced contextual model. Use deep learning to enhance semantic representation and validate its performance over baseline models.

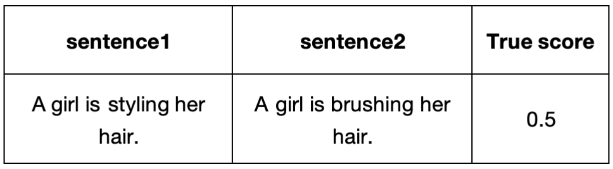
- **Prediction Reliability Assessment:** Compare predicted scores of all models with human annotations to assess accuracy and stability, supporting practical applications.

- **Exploration of Practical Applications:** Explore how semantic similarity models, especially SBERT, can improve tasks like information retrieval, machine translation, and question answering.

Data Source

For our Semantic Textual Similarity task, we utilize the **Semantic Textual Similarity Benchmark (STSb)** dataset from this link: https://huggingface.co/datasets/sentence-transformers/stsb/viewer/default/train.

This dataset is split into three parts:'stsb_train.csv','stsb_validation.csv',and 'stsb_test.csv' (training set, validation set and test set). The dataset contains three variables:**Sentence 1**,**Sentence 2** and **Score**.Sentence 1 and Sentence 2 are sentence pairs drawn from sources such as news headlines, video captions, and natural language inference data. Score is a human-annotated similarity score, normalized between 0 and 1. We treat this score as the *true score* in this project.



sentence1	sentence2	True score
A girl is styling her hair.	A girl is brushing her hair.	0.5

Figure 2: source dataset instances

Evaluation Standard

To comprehensively evaluate model performance, we adopt both **quantitative metrics-based evaluation**, including F1-Score, Accuracy, Spearman Correlation, and Pearson Correlation, and **predicted-to-true score ratio analysis**, which provides a distributional perspective on model reliability and accuracy.

1. **Metrics-based evaluation:** The following performance metrics are used to quantitatively assess the models:

   - **F1-Score:** F1-Score measures the balance between precision and recall using their harmonic mean. It accounts for both false positives and false negatives, making it ideal for evaluating models on imbalanced datasets. A higher F1-Score (close to 1) indicates better model performance, as it reflects a stronger balance between precision and recall.

     $$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

     where:

     – Precision $= \frac{\text{True Positives (TP)}}{\text{True Positives (TP)+False Positives (FP)}}$

     – Recall $= \frac{\text{True Positives (TP)}}{\text{True Positives (TP)+False Negatives (FN)}}$

   - **Accuracy:** Accuracy reflects the proportion of correctly predicted samples out of the total samples. It is a straightforward metric to assess a model's overall classification performance.A higher accuracy (close to 1) indicates that the model makes fewer overall errors.

     $$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}}$$

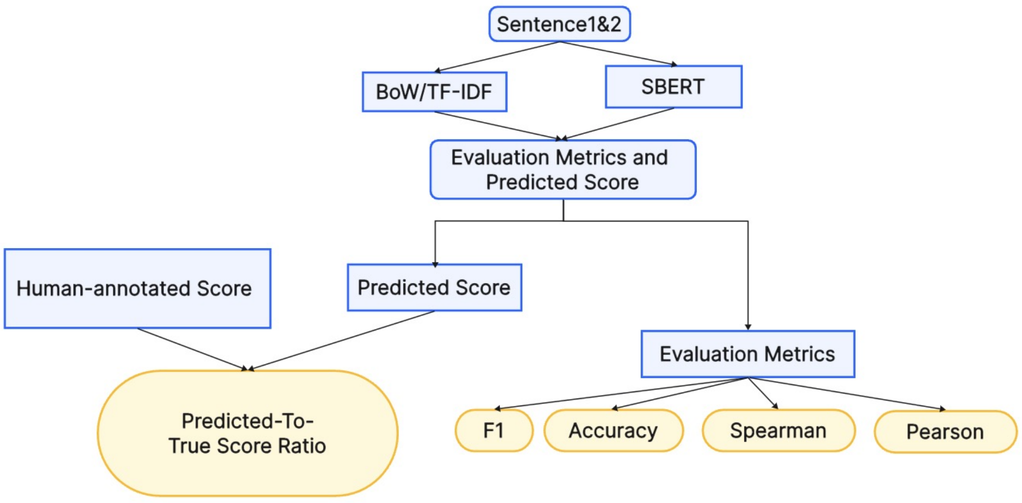


Figure 3: Evaluation Method Process. Include F1, Accuracy, Spearman & Pearson Correlation, predict/true ratios.

- **Spearman Correlation:** Spearman Correlation measures the rank correlation between predicted and true scores without assuming a linear relationship. A higher value (close to 1) indicates a strong monotonic relationship, while values near 0 suggest no correlation, and negative values indicate an inverse relationship. It is defined as:

$$\rho = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)}$$

  where:
  - d_i is the difference between the ranks of the i-th prediction and its corresponding true score.
  - n is the number of observations.
- **Pearson Correlation:** Pearson Correlation measures the linear relationship between predicted and true scores. A value close to 1 indicates a strong linear relationship, 0 suggests no correlation, and negative values indicate an inverse relationship. It is defined as:

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

  where:
  - x_i and y_i are the predicted and true scores for the i-th observation.
  - $\bar{x}$ and $\bar{y}$ are the mean values of the predicted and true scores, respectively.

2. **Predicted-to-True Score Ratio Analysis:** We compare the predicted scores of different models (e.g., BoW, TF-IDF, and SBERT) with human-annotated true scores and calculate the ratio between predicted and true scores to evaluate model reliability. Ratios are grouped into intervals (e.g., 0.1–0.2, 0.2–0.3, etc.) , and the number of sentence pairs within each interval is counted and visualized as histograms to assess performance. **A ratio close to 1 indicates high similarity between predicted and true scores.** The ratio is calculated using the following formula:

$$\text{Predicted-to-True Score Ratio} = \frac{\text{Predicted Score}}{\text{True Score}}$$

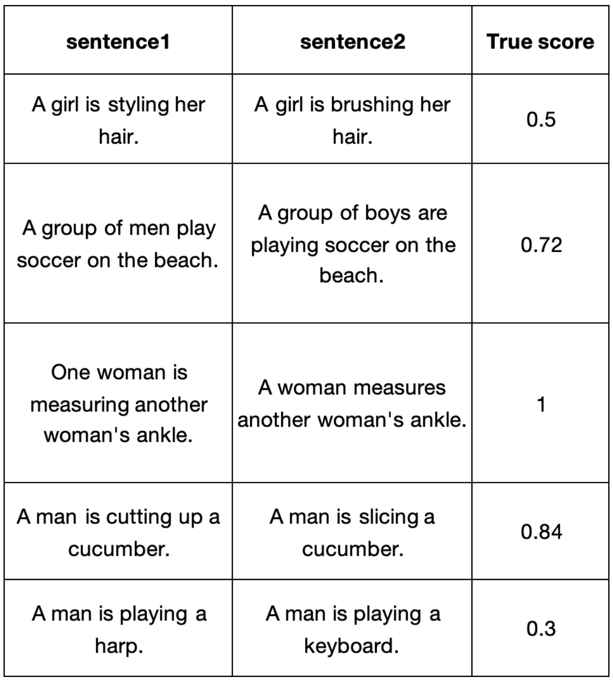
   This method evaluates the alignment between predicted scores and true scores from a distribution perspective, providing a clear and intuitive basis for performance analysis.

Figure 3 illustrates the evaluation process, where baseline models (BoW, TF-IDF) and SBERT generate predicted scores and evaluation metrics, including F1-Score, Accuracy, Spearman Correlation, and Pearson Correlation. Predicted scores are compared with human-annotated scores to access performance and reliability , while evaluation metrics directly assess performance and reliability.

Input and Output Example

During evaluation, the following is an example of input and output:

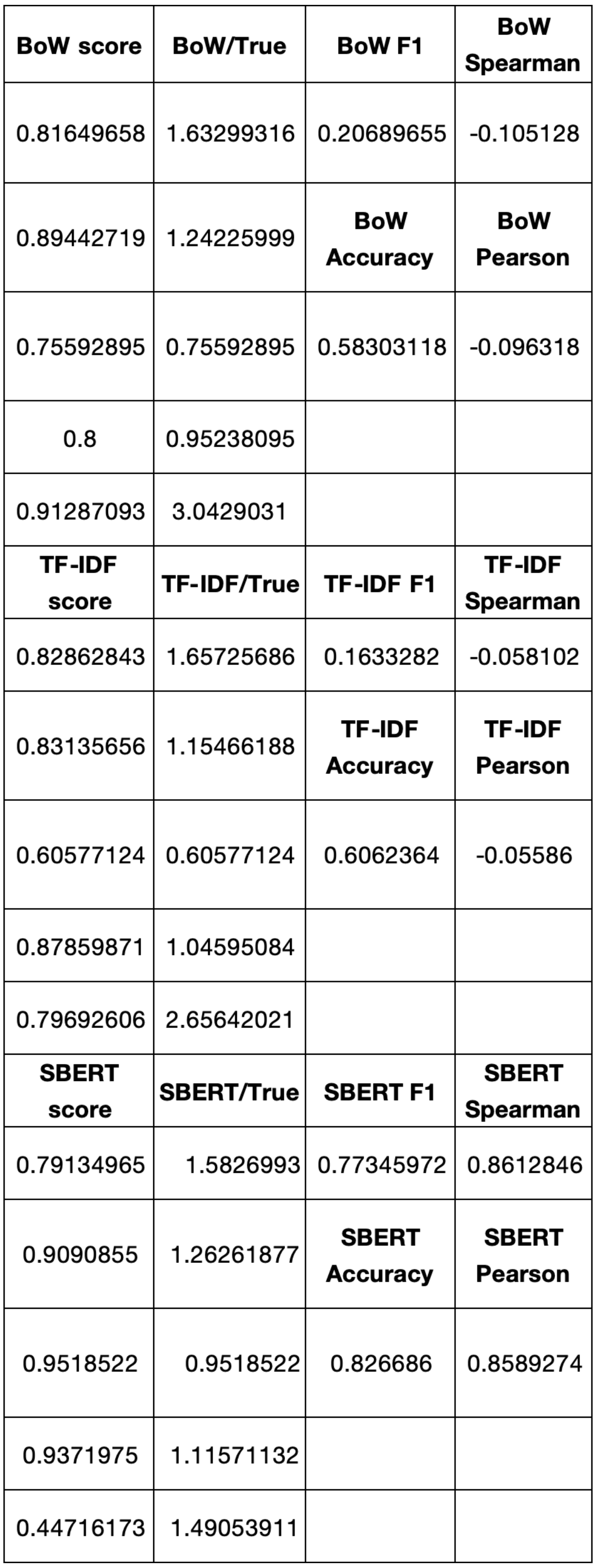
- **Input:** Input includes sentence 1 and sentence 2 as a sentence pair, and the human-annotated score as true score. Input are saved as '*stsb_train.csv*','*stsb_validation.csv*' and '*stsb_test.csv*' in *source_dataset* folder in our github.



sentence1	sentence2	True score
A girl is styling her hair.	A girl is brushing her hair.	0.5
A group of men play soccer on the beach.	A group of boys are playing soccer on the beach.	0.72
One woman is measuring another woman's ankle.	A woman measures another woman's ankle.	1
A man is cutting up a cucumber.	A man is slicing a cucumber.	0.84
A man is playing a harp.	A man is playing a keyboard.	0.3

Figure 4: input example

- **Output:** Output are saved as '*baseline_stsb_validation.csv*','*baseline_stsb_test.csv*' '*sbert_stsb_validation.csv*' and '*sbert_stsb_test.csv*' in *output_dataset* folder in our github. Output includes:
  - **Model Predicted Score**
  - **Predicted/True Ratio**
  - **Metrics-based evaluation : F1-Score, Accuracy, Spearman Correlation, Pearson Correlation**



BoW score	BoW/True	BoW F1	BoW Spearman
0.81649658	1.63299316	0.20689655	-0.105128
0.89442719	1.24225999	**BoW Accuracy**	**BoW Pearson**
0.75592895	0.75592895	0.58303118	-0.096318
0.8	0.95238095		
0.91287093	3.0429031		
TF-IDF score	**TF-IDF/True**	**TF-IDF F1**	**TF-IDF Spearman**
0.82862843	1.65725686	0.1633282	-0.058102
0.83135656	1.15466188	**TF-IDF Accuracy**	**TF-IDF Pearson**
0.60577124	0.60577124	0.6062364	-0.05586
0.87859871	1.04595084		
0.79692606	2.65642021		
SBERT score	**SBERT/True**	**SBERT F1**	**SBERT Spearman**
0.79134965	1.5826993	0.77345972	0.8612846
0.9090855	1.26261877	**SBERT Accuracy**	**SBERT Pearson**
0.9518522	0.9518522	0.826686	0.8589274
0.9371975	1.11571132		
0.44716173	1.49053911		

Figure 5: BoW/TF-IDF/SBERT output example

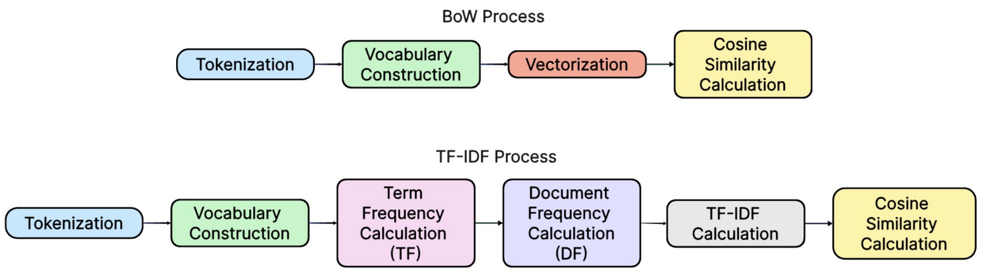


Figure 6: BoW/TF-IDF Process.

Model Details

We adopted **Bag-of-Words (BoW)** and **TF-IDF (Term Frequency-Inverse Document Frequency)** as traditional baselines and **Sentence-BERT (SBERT)** as the advanced semantic method. These baseline methods rely on word frequency and keyword matching, offering a simple yet effective approach for text representation. However, they lack the ability to capture deeper semantic relationships and contextual information. In contrast, SBERT leverages contextual embeddings and a Siamese network structure to generate semantically meaningful sentence representations, enabling more accurate similarity detection. This combination provides a comprehensive comparison between traditional and modern approaches to semantic similarity tasks.

- **Bag-of-Words (BoW):**
  - **Definition:** Represents sentences as word frequency vectors. The similarity between two sentence vectors v_1 and v_2 is computed using cosine similarity.
  - **Process:**
    1. **Tokenization:** Split sentences into individual words or tokens.
       ∗ **Example:**
       ```
       Sentence: "The cat sat"
       Tokens: ["The", "cat",
           "sat"]
       ```
    2. **Vocabulary Construction:** Build a vocabulary containing all unique words in the dataset.
       ∗ **Example:**
       ```
       Tokens: ["The", "cat",
           "sat", "on", "mat"]
       Vocabulary: {"The", "
           cat", "sat", "on",
           "mat"}
       ```
    3. **Vectorization:** Represent each sentence as a frequency vector based on the constructed vocabulary.
       ∗ **Example:**
       ```
       Sentence: "The cat sat"
       Vector: [1, 1, 1, 0, 0]
            # Corresponding to
            the vocabulary
         order
       ```
    4. **Cosine Similarity Calculation:** Compute the cosine similarity between sentence vectors to measure their similarity. For example, for two sentence vectors $v_1 = [1, 1, 1, 0, 0]$ and $v_2 = [0, 1, 1, 1, 0]$:

$$\text{Cosine Similarity} = \frac{v_1 \cdot v_2}{\|v_1\| \|v_2\|}$$

  - **Advantages:**
    ∗ **Simplicity:** Easy to implement with minimal computational cost, making it suitable for straightforward tasks and small datasets.
    ∗ **Interpretability:** The frequency-based representation directly maps to words in the vocabulary, facilitating debugging and analysis.
  - **Limitations:**
    ∗ **Ignores Context:** Treats sentences as unordered collections of words, losing semantic and syntactic information.
    ∗ **Semantic Ambiguity:** Struggles to distinguish sentences that are lexically similar but semantically different.
    ∗ **High Dimensionality:** Large vocabulary size can lead to computational challenges.

- **TF-IDF (Term Frequency-Inverse Document Frequency):**
  - **Definition:** Enhances BoW by assigning weights to words based on their importance in the corpus. The weight for term t in document d is calculated as:

    $$\text{TF-IDF}(t,d) = \text{TF}(t,d) \cdot \log\left(\frac{N}{\text{DF}(t)}\right)$$

    where:
    * $\text{TF}(t,d)$ is the term frequency of t in d.
    * N is the total number of documents.
    * $\text{DF}(t)$ is the number of documents containing t.
  - **Process:**
    1. **Tokenization:** Split sentences into individual words or tokens.
       * **Example:**
       ```
       Sentence: "The cat sat"
       Tokens: ["The", "cat",
           "sat"]
       ```
    2. **Vocabulary Construction:** Build a vocabulary containing all unique words in the dataset.
       * **Example:**
       ```
       Tokens: ["The", "cat",
           "sat", "on", "mat"]
       Vocabulary: {"The", "
           cat", "sat", "on",
           "mat"}
       ```
    3. **Term Frequency Calculation (TF):** Compute the frequency of each word in the document.

       $$\text{TF}(t,d) = \frac{\text{count of } t \text{ in } d}{\text{total words in } d}$$

       * **Example:**
       ```
       Sentence: "The cat sat"
       TF("cat") = 1 / 3 =
           0.33
       ```
    4. **Document Frequency Calculation (DF):** Determine how many documents contain each word. DF(t) is number of documents containing t.
       * **Example:**
       ```
       Word: "cat"
       DF("cat") = 10 (found
           in 10 documents)
       ```
    5. **TF-IDF Calculation:** Compute the TF-IDF value for each word in the sentence.

       $$\text{TF-IDF}(t,d) = \text{TF}(t,d) \cdot \log\left(\frac{N}{\text{DF}(t)}\right)$$

       * **Example:**
       ```
       If N = 100 and DF("cat
           ") = 10:
       TF-IDF("cat") = TF("cat
           ") * log(100 / 10)
       ```
    6. **Cosine Similarity Calculation:** Compute the cosine similarity between sentence vectors to measure their similarity. For sentence vectors v1 and v2:

       $$\text{Cosine Similarity} = \frac{v_1 \cdot v_2}{\|v_1\|\|v_2\|}$$
  - **Advantages:**
    * **Weighted Representation:** Highlights meaningful words by reducing the influence of common terms.
    * **Efficiency:** Produces sparse vectors that are easier to process and store compared to dense representations.
  - **Limitations:**
    * **Context Ignorance:** Similar to BoW, fails to capture word dependencies.
    * **Vocabulary Sensitivity:** Requires a fixed vocabulary, which can limit flexibility.
    * **Limited Semantic Understanding:** Cannot handle semantically equivalent but lexically different sentences.
- **Reason for Selecting BoW and TF-IDF:**
  - **Foundational Methods:** Serve as traditional and widely-used text representation techniques.
  - **Comparative Analysis:** Their simplicity enables a clear performance contrast with advanced models like SBERT.
  - **Reliable Baselines:** Provide consistent initial results for straightforward tasks, even with their limitations.

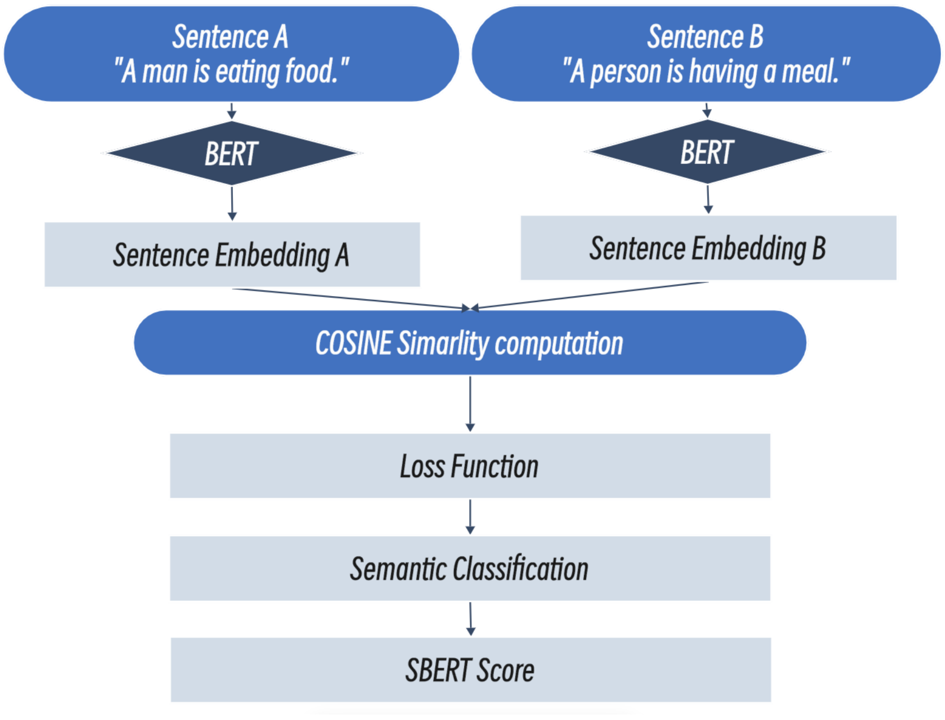


Figure 7: SBERT Process.

- **Sentence-BERT (SBERT):**
  - **Definition:** Sentence-BERT (SBERT) is an advanced model designed for sentence-level semantic similarity tasks. It extends the BERT architecture by incorporating a Siamese network structure to efficiently compute embeddings for input sentence pairs. These embeddings are then used to compute similarity scores.
  - **Process:** Sentence-BERT (SBERT) performs sentence embedding and similarity computation through the following steps:
    1. **Embedding generation:** SBERT uses a BERT model to create embeddings for input sentences S_1 and S_2:

       $$e_1 = \text{BERT}(S_1), \quad e_2 = \text{BERT}(S_2)$$

       Here, S_1 and S_2 are the input sentences to be computed. For example: S_1= "The cat sits on the mat", S_2 = "A cat is sitting on a mat".

       The resulting embeddings e_1 and e_2 are high-dimensional vectors capturing the semantic meaning of the sentences. For instance, e_1 = [0.1, 0.5, -0.3, $\dots$], e_2 = [0.2, 0.6, -0.1, $\dots$].
    2. **COSINE Similarity Computation:** Compute the cosine similarity between the sentence embeddings to measure semantic similarity.

       $$\text{Cosine Similarity} = \frac{e_1 \cdot e_2}{\|e_1\| \|e_2\|}$$
    3. **Model Training:** Fine-tune the SBERT model on a dataset to produce embeddings aligned with similarity scores. During this step, the model learns to minimize the discrepancy between predicted and human-annotated similarity scores using a carefully designed loss function.
       - **Loss Function:** The Cosine Similarity Loss aligns the model's predictions with the human-annotated similarity scores by minimizing the squared error:

         $$L = \frac{1}{N} \sum_{i=1}^{N} (\text{Cosine Similarity}(e_{1i}, e_{2i}) - y_i)^2$$

         Here, e_{1i} and e_{2i} are the embeddings for sentence pair i, and y_i

is the human-labeled similarity score.

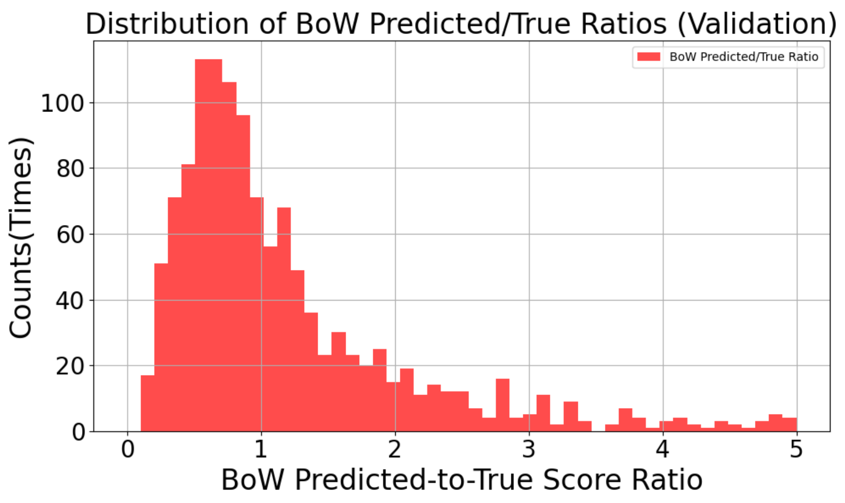
- **Parameter Tuning:** The model's performance is enhanced by adjusting critical hyperparameters, includes:
  - **Model Name:** `MODEL_NAME = 'paraphrase-MiniLM-L6-v2'` : A pre-trained lightweight SBERT model that balances efficiency and performance.
  - **Number of Epochs:** `EPOCHS = 6` : Determines how many times the model iterates over the training dataset.
  - **Batch Size:** `BATCH_SIZE = 32` : Specifies the number of samples processed simultaneously during training to optimize computational efficiency.
  - **Learning Rate:** `LEARNING RATE = 1e-5` : Controls the step size for updating model weights during optimization.
  - **Similarity Threshold:** `SIMILARITY THRESHOLD = 0.7` : Defines the boundary for classifying sentence pairs as similar or dissimilar.

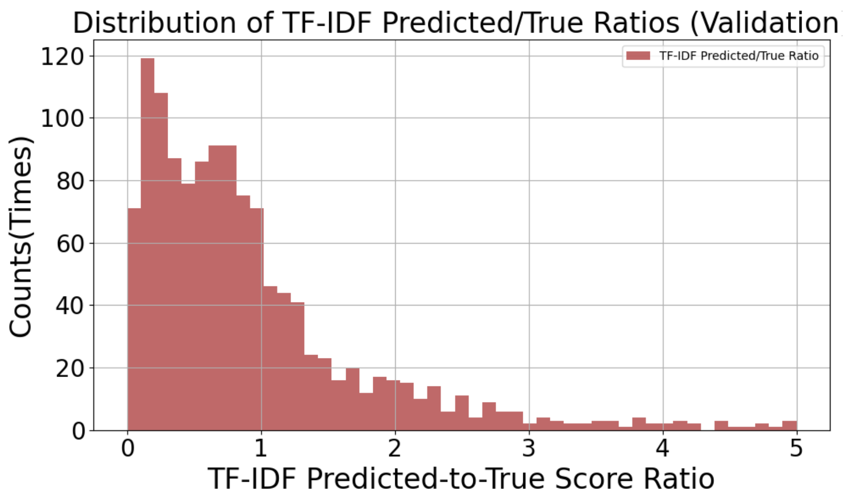
4. **Semantic Classification :** For specific tasks, sentence embeddings are fed into a fully connected layer to perform classification or regression tasks:

$$\hat{y} = \sigma(W \cdot e + b)$$

where σ is the activation function, and W and b are learnable parameters.

- **Advantages:**
  - **Efficient Similarity Computation:** Pre-computes embeddings for sentences, enabling fast similarity calculation during inference.
  - **High Semantic Accuracy:** Rather than relying on word frequency, SBERT captures the contextual meaning of sentences, making it robust for semantic similarity tasks.
  - **Flexibility:** Easily fine-tuned for a wide range of downstream tasks such as sentence classification, clustering, and information retrieval.
  - **Pre-trained Models:** Offers a variety of pre-trained models optimized for different tasks, reducing the need for extensive data preparation.
- **Limitations:**
  - **Resource Intensive:** Requires more computational resources compared to traditional models like BoW and TF-IDF.
  - **Dependency on Quality Data:** Relies on high-quality and sufficiently large datasets for fine-tuning.
- **Why We Choose SBERT as the Advanced Model:**
  - **Semantic Understanding:** SBERT captures the semantic relationships between sentences rather than relying solely on word frequency or surface-level word matching. This capability makes it particularly effective for tasks that require understanding the nuanced meaning of different sentence structures.
  - **State-of-the-Art Performance:** SBERT demonstrates superior performance in semantic similarity tasks, significantly outperforming traditional methods like BoW and TF-IDF on evaluation metrics such as Pearson and Spearman correlations. This exceptional accuracy makes it a preferred choice for semantic similarity problems.
  - **Adaptability:** SBERT's architecture is highly flexible, allowing for easy fine-tuning on specific datasets and tasks. It can be applied to various real-world applications, including sentence classification, text clustering, and information retrieval, with minimal adjustments.
  - **Efficiency:** By precomputing sentence embeddings during the inference phase, SBERT greatly reduces the time required for similarity calculations. This efficiency makes it ideal for scenarios requiring real-time or large-scale semantic similarity computations.





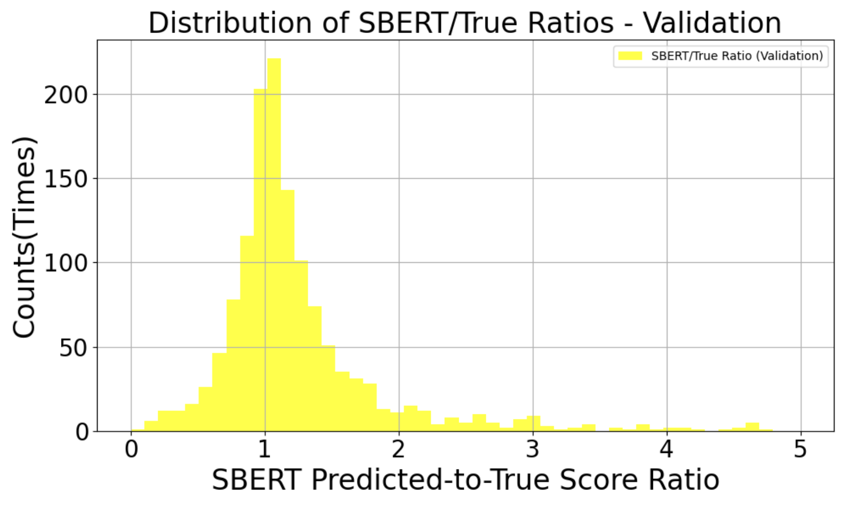
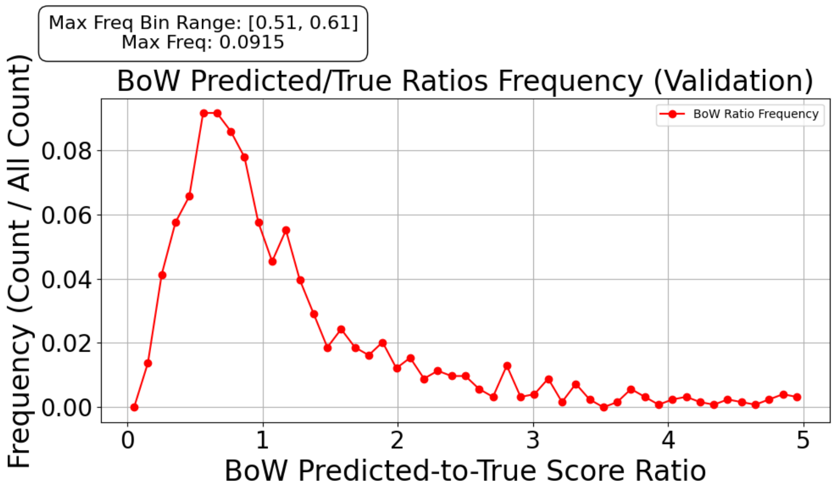
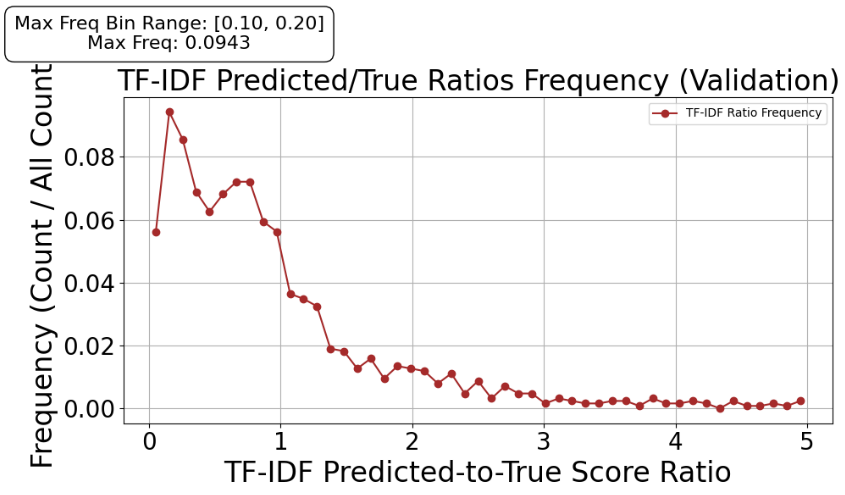


Figure 8: BoW/TF-IDF/SBERT Distribution of Predicted to True Ratios for Validation Dataset. The X-axis represents the ratio of the Bow predicted score to the original human-generated (true) score, while the Y-axis shows the counts of each ratio (eg 100 means this ratio appears 100 times).





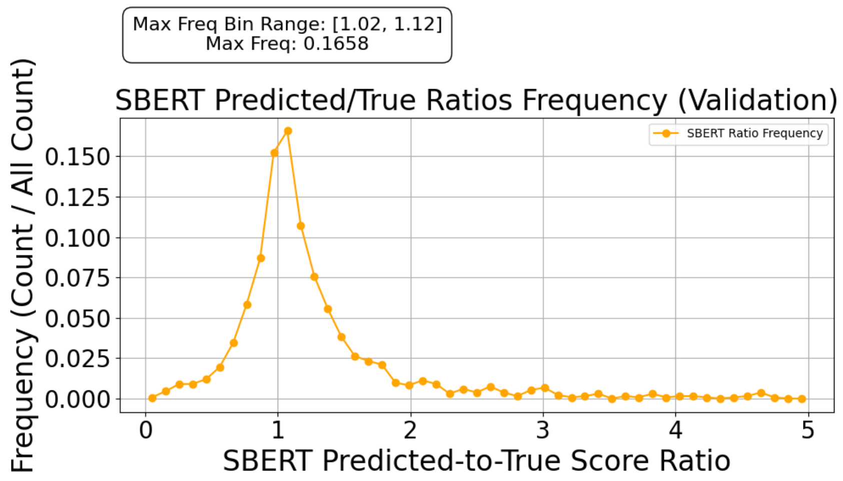
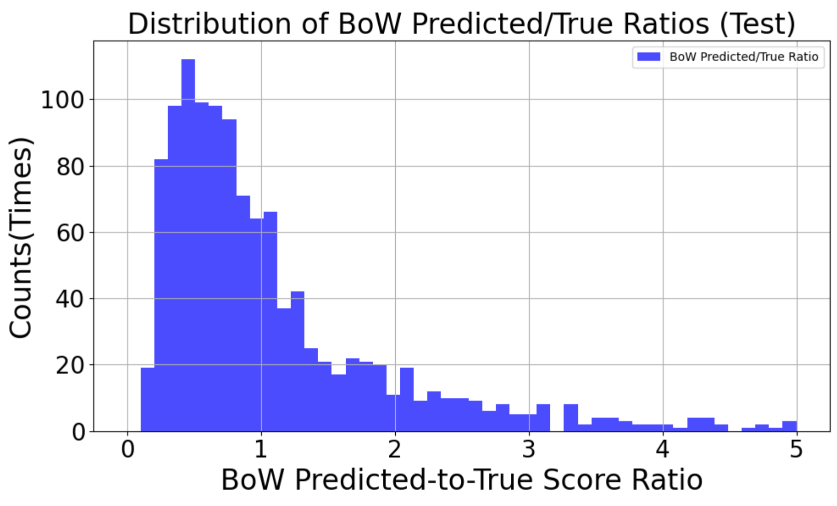
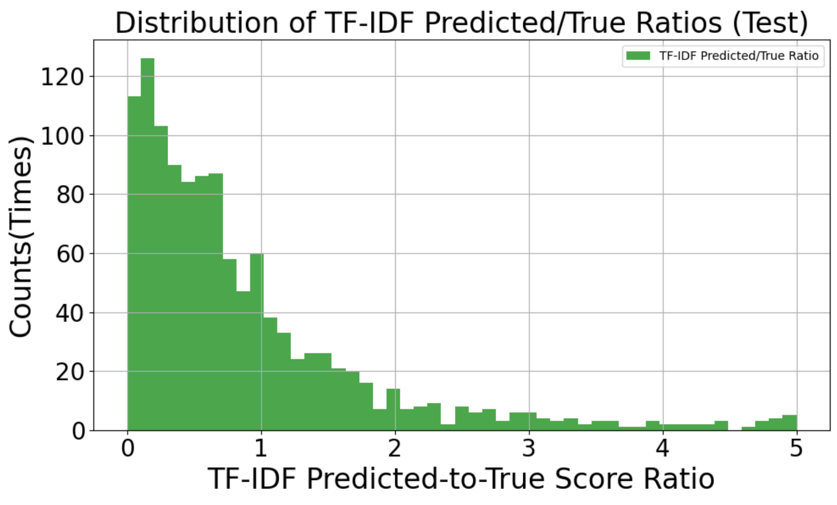


Figure 9: BoW/TF-IDF/SBERT Frequency of Predicted to True Ratios for Validation Dataset. The X-axis represents the ratio of the Bow predicted score to the original human-generated (true) score. The Y-axis shows frequency of each ratio,where the frequency is defined as the count of a specific Predicted-to-True Ratio divided by the total count (e.g., 0.1 means that 10 percent of the total ratios fall within this bin)





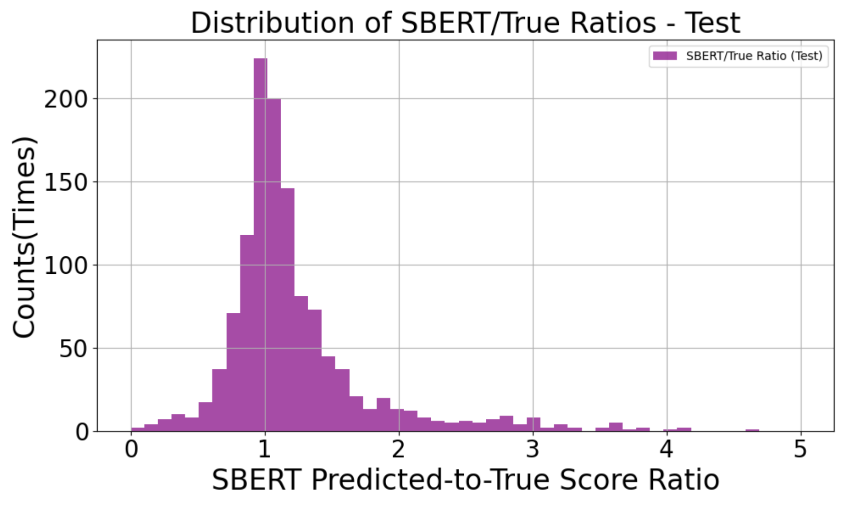
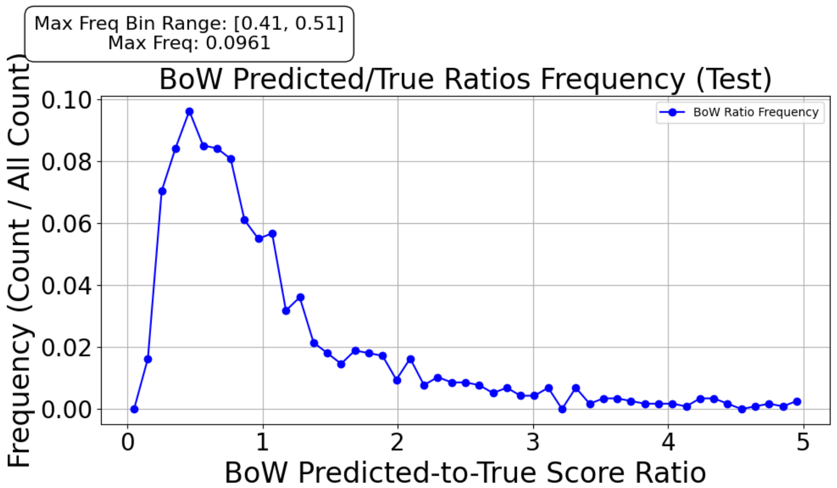
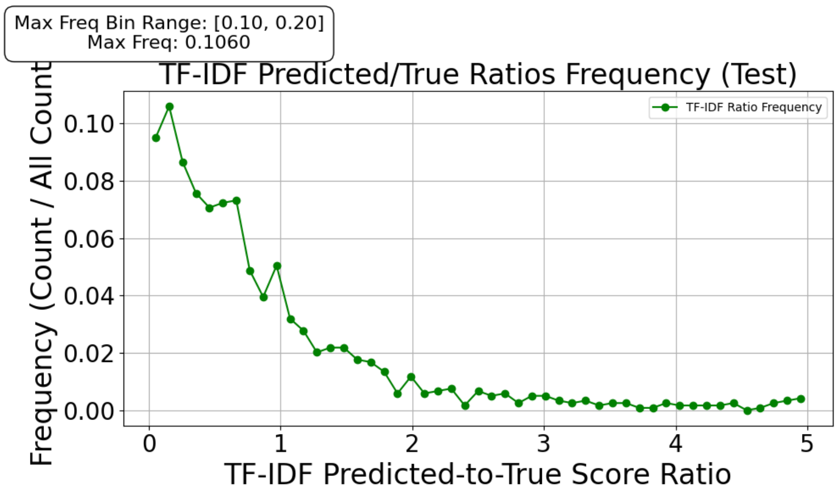


Figure 10: BoW/TF-IDF/SBERT Distribution of Predicted to True Ratios for Test Dataset. The X-axis represents the ratio of the Bow predicted score to the original human-generated (true) score, while the Y-axis shows the counts of each ratio (eg 100 means this ratio appears 100 times).





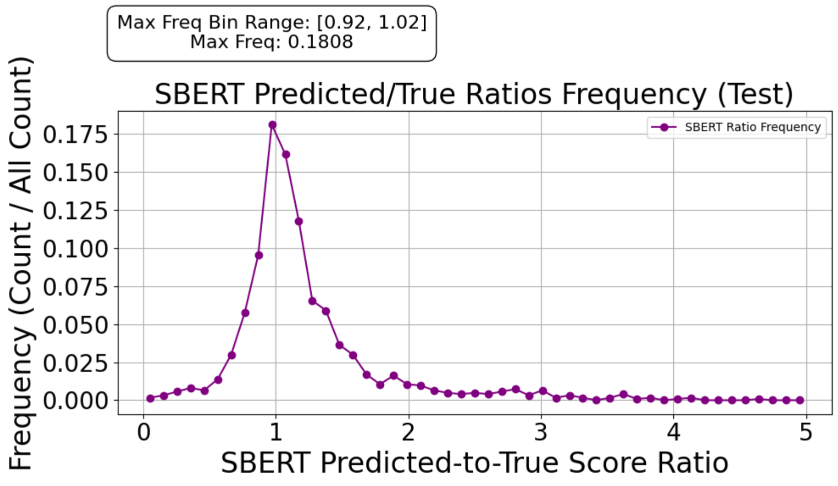
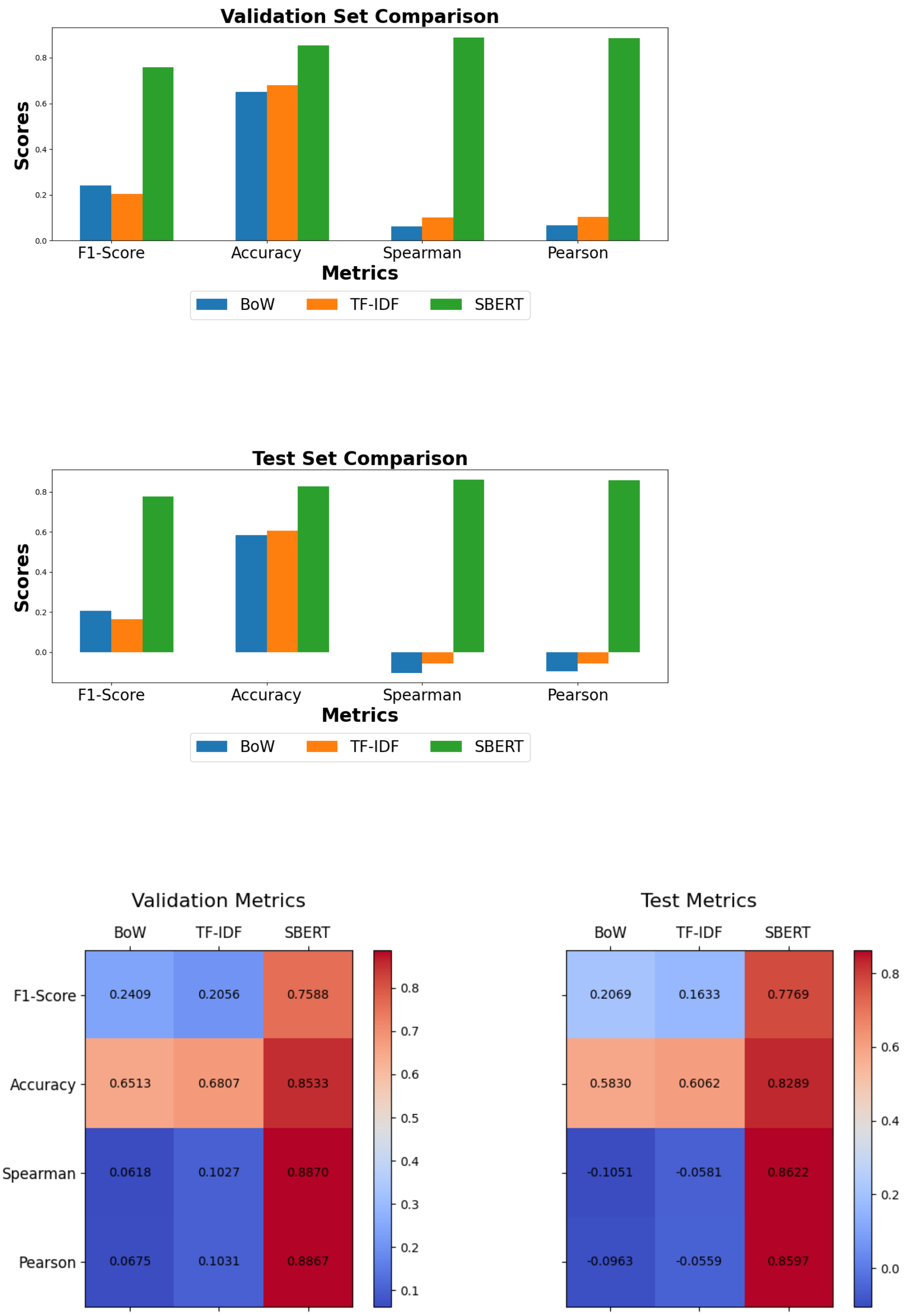


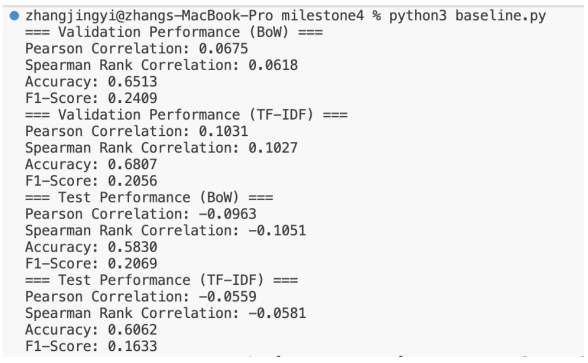
Figure 11: BoW/TF-IDF/SBERT Frequency of Predicted to True Ratios for Test Dataset. The X-axis represents the ratio of the Bow predicted score to the original human-generated (true) score. The Y-axis shows frequency of each ratio,where the frequency is defined as the count of a specific Predicted-to-True Ratio divided by the total count (e.g., 0.1 means that 10 percent of the total ratios fall within this bin)



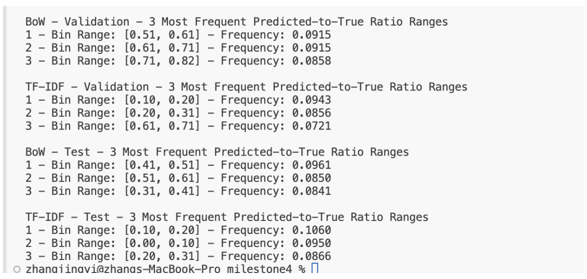
Validation Metrics	BoW	TF-IDF	SBERT
F1-Score	0.2409	0.2056	0.7588
Accuracy	0.6513	0.6807	0.8533
Spearman	0.0618	0.1027	0.8870
Pearson	0.0675	0.1031	0.8867

Test Metrics	BoW	TF-IDF	SBERT
F1-Score	0.2069	0.1633	0.7769
Accuracy	0.5830	0.6062	0.8289
Spearman	-0.1051	-0.0581	0.8622
Pearson	-0.0963	-0.0559	0.8597

Figure 12: F1-Score, Accuracy, Spearman, and Pearson Score comparison for both Validation and Test sets using BoW, TF-IDF, and SBERT models, presented in grid and histogram formats.

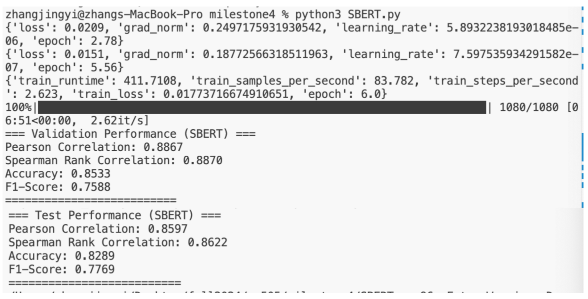


```
● zhangjingyi@zhangs-MacBook-Pro milestone4 % python3 baseline.py
  === Validation Performance (BoW) ===
  Pearson Correlation: 0.0675
  Spearman Rank Correlation: 0.0618
  Accuracy: 0.6513
  F1-Score: 0.2409
  === Validation Performance (TF-IDF) ===
  Pearson Correlation: 0.1031
  Spearman Rank Correlation: 0.1027
  Accuracy: 0.6807
  F1-Score: 0.2056
  === Test Performance (BoW) ===
  Pearson Correlation: -0.0963
  Spearman Rank Correlation: -0.1051
  Accuracy: 0.5830
  F1-Score: 0.2069
  === Test Performance (TF-IDF) ===
  Pearson Correlation: -0.0559
  Spearman Rank Correlation: -0.0581
  Accuracy: 0.6062
  F1-Score: 0.1633
```

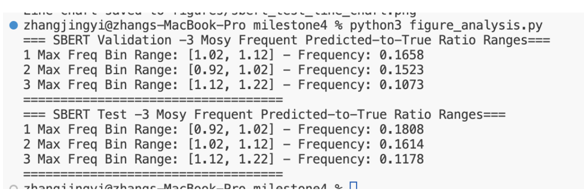


```
  BoW - Validation - 3 Most Frequent Predicted-to-True Ratio Ranges
  1 - Bin Range: [0.51, 0.61] - Frequency: 0.0915
  2 - Bin Range: [0.61, 0.71] - Frequency: 0.0915
  3 - Bin Range: [0.71, 0.82] - Frequency: 0.0858

  TF-IDF - Validation - 3 Most Frequent Predicted-to-True Ratio Ranges
  1 - Bin Range: [0.10, 0.20] - Frequency: 0.0943
  2 - Bin Range: [0.20, 0.31] - Frequency: 0.0856
  3 - Bin Range: [0.61, 0.71] - Frequency: 0.0721

  BoW - Test - 3 Most Frequent Predicted-to-True Ratio Ranges
  1 - Bin Range: [0.41, 0.51] - Frequency: 0.0961
  2 - Bin Range: [0.51, 0.61] - Frequency: 0.0850
  3 - Bin Range: [0.31, 0.41] - Frequency: 0.0841

  TF-IDF - Test - 3 Most Frequent Predicted-to-True Ratio Ranges
  1 - Bin Range: [0.10, 0.20] - Frequency: 0.1060
  2 - Bin Range: [0.00, 0.10] - Frequency: 0.0950
  3 - Bin Range: [0.20, 0.31] - Frequency: 0.0866
○ zhangjingyi@zhangs-MacBook-Pro milestone4 %
```

Figure 13: the baseline(BoW/TF-IDF) training results, including the distribution characteristics of predicted-to-true ratios; and the main evaluation metrics for both the validation and test sets.



```
zhangjingyi@zhangs-MacBook-Pro milestone4 % python3 SBERT.py
{'loss': 0.0209, 'grad_norm': 0.2497175931930542, 'learning_rate': 5.8932238193018485e-
06, 'epoch': 2.78}
{'loss': 0.0151, 'grad_norm': 0.18772566318511963, 'learning_rate': 7.597535934291582e-
07, 'epoch': 5.56}
{'train_runtime': 411.7108, 'train_samples_per_second': 83.782, 'train_steps_per_second
': 2.623, 'train_loss': 0.01773716674910651, 'epoch': 6.0}
100%|████████████████████████████████████████████████████| 1080/1080 [0
6:51<00:00,  2.62it/s]
=== Validation Performance (SBERT) ===
Pearson Correlation: 0.8867
Spearman Rank Correlation: 0.8870
Accuracy: 0.8533
F1-Score: 0.7588
==========================
 === Test Performance (SBERT) ===
 Pearson Correlation: 0.8597
 Spearman Rank Correlation: 0.8622
 Accuracy: 0.8289
 F1-Score: 0.7769
 ==========================
```



```
  Line chart saved to figures/sbert_test_line_chart.png
● zhangjingyi@zhangs-MacBook-Pro milestone4 % python3 figure_analysis.py
  === SBERT Validation -3 Mosy Frequent Predicted-to-True Ratio Ranges===
  1 Max Freq Bin Range: [1.02, 1.12] - Frequency: 0.1658
  2 Max Freq Bin Range: [0.92, 1.02] - Frequency: 0.1523
  3 Max Freq Bin Range: [1.12, 1.22] - Frequency: 0.1073
  ===================================
  === SBERT Test -3 Mosy Frequent Predicted-to-True Ratio Ranges===
  1 Max Freq Bin Range: [0.92, 1.02] - Frequency: 0.1808
  2 Max Freq Bin Range: [1.02, 1.12] - Frequency: 0.1614
  3 Max Freq Bin Range: [1.12, 1.22] - Frequency: 0.1178
  ===================================
○ zhangjingyi@zhangs-MacBook-Pro milestone4 %
```

Figure 14: the SBERT training process and results, including metrics such as loss, gradient, and learning rate changes during training; the distribution characteristics of predicted-to-true ratios; and the main evaluation metrics for both the validation and test sets.

Results&Experiments

As we mentioned before, the experiment utilized the **Semantic Textual Similarity Benchmark (STSb)** dataset to evaluate model performance from two perspectives: quantitative metrics (including F1-Score, Accuracy, Spearman Correlation, and Pearson Correlation) and the frequency distribution of predicted-to-true score ratios. The comparison involved three models: **BoW**, **TF-IDF** (baseline model), and **SBERT** (the advanced model).

The figures provide a visual overview of the performance of BoW, TF-IDF, and SBERT models. Figures 8 and 9 illustrate the distribution and frequency of predicted-to-true score ratios on the validation set, while Figures 10 and 11 present similar results for the test set. Figure 12 compares the main evaluation metrics, including F1-Score, Accuracy, Spearman, and Pearson correlations, across the three models on both validation and test sets. Figures 13 and 14 show the terminal reports, detailing the ratio distributions and corresponding evaluation results. The following analysis will be based on these results (results of validation set).

The results clearly show SBERT's superior performance and reliability over baseline models, as summarized below:

- **Evaluation Metrics Comparison:**
  - **F1-Score:**SBERT achieves an F1-Score of 0.7588, which is 2-3 times higher than BoW (0.2409) and TF-IDF (0.2056). This indicates that SBERT better balances prediction precision and coverage in sentence similarity tasks.
  - **Accuracy:** SBERT achieves an accuracy of 0.8533, significantly outperforming BoW (0.6513) and TF-IDF (0.6807).This demonstrates that SBERT more accurately captures semantic relationships between sentences.
  - **Correlation Metrics (Spearman and Pearson):** SBERT achieves Spearman and Pearson correlation coefficients of 0.8870 and 0.8867, respectively, showing a high consistency with human-labeled scores. In contrast, BoW scores only 0.0618 and 0.0675, while TF-IDF scores 0.1027 and 0.1031. The large gap highlights SBERT's superior semantic modeling capabilities.
- **Predicted-to-True Ratio Distribution Analysis:**
  - **SBERT Prediction Stability:**
    - ∗ SBERT exhibits high concentration in its predicted-to-true ratio distribution. The highest frequency range is [1.02, 1.12], with a frequency of 0.1658.
    - ∗ The top three high-frequency ranges, [0.92, 1.02], [1.02, 1.12], and [1.12, 1.22], are all close to 1.0, with a combined frequency exceeding 40%. **A ratio close to 1 indicates a high similarity between predicted and true scores**, demonstrating SBERT's superior accuracy and robust prediction stability.
  - **BoW and TF-IDF Dispersion:**In contrast, BoW and TF-IDF exhibit more dispersed predicted-to-true ratio distributions:
    - ∗ BoW's highest frequency range is [0.51, 0.61], with a frequency of only 0.0915. Its predicted-to-true ratios are widely spread away from 1.0, and the tail regions (extreme values) constitute a significant proportion, indicating larger prediction errors and lack of stability.
    - ∗ TF-IDF's highest frequency range is [0.10, 0.20], with a frequency of 0.0943. Other high-frequency ranges (e.g., [0.20, 0.31]) are sparsely distributed, making it less effective in accurately reflecting sentence similarity.

To sum up,SBERT demonstrates significant superiority over BoW and TF-IDF in both quantitative metrics and prediction stability. This validates the advantage of Transformer-based pre-trained models in sentence similarity tasks and underscores the importance of adopting stronger semantic modeling methods to enhance performance and reliability in natural language processing tasks.

Conclusion

The experimental results clearly demonstrate that Sentence-BERT (SBERT) significantly outperforms traditional models such as Bag-of-Words (BoW) and TF-IDF in semantic similarity tasks. Based on performance evaluation and analysis, the following conclusions are drawn:

- **Semantic Understanding Ability**: SBERT excels in capturing deep semantic relationships and contextual information between sentences, enabling it to handle complex semantic tasks effectively. In contrast, BoW and TF-IDF, which rely on word frequency and keyword matching, are limited in understanding nuanced semantics.

- **Model Performance**: SBERT surpasses baseline models across all evaluation metrics, including F1-Score, Accuracy, Pearson Correlation, and Spearman Correlation. For instance, SBERT achieves significantly higher F1-Score compared to BoW and TF-IDF, demonstrating its superior accuracy and coverage in predictions.

- **Prediction Stability**: SBERT's predicted-to-true score ratios are highly concentrated around 1.0, indicating minimal prediction bias and high reliability. In contrast, BoW and TF-IDF exhibit more dispersed distributions, reflecting inconsistencies in handling complex tasks.

- **Applicability and Robustness**: With its profound understanding of sentence semantics and stable prediction performance, SBERT emerges as an ideal choice for semantic similarity tasks. Its ability to accurately capture semantic relationships makes it highly applicable in real-world scenarios that require precise semantic judgment.

In summary, this study validates the significant advantages of Transformer-based pre-trained models in semantic similarity tasks and provides a reliable reference for related research in natural language processing. These findings also highlight SBERT's potential to improve practical applications such as information retrieval, machine translation, and question answering.

Applications and Benefits of This Project

This project delivers a precise solution for sentence semantic similarity tasks, enhancing natural language understanding and application intelligence across industries. Key applications include:

- **Information Retrieval:** Enhances search engines and recommendation systems for faster, more accurate information access.

- **Machine Translation:** Improves translation quality by evaluating semantic consistency.

- **Question Answering:** Boosts the accuracy of intelligent assistants and Q&A systems.

- **Education and Research:** Aids in analyzing semantic consistency in student work and academic studies.

- **Human-Computer Interaction:** Enhances chatbot and virtual assistant understanding for better user experiences.

Potential Improvements

- **Handling Zero Values:** Currently, data points with `True Score` and `Predicted Score` equal to 0 are ignored because the predicted-to-true ratio becomes undefined or invalid. While this simplification aids processing, it may result in information loss. Future work could explore better handling methods, such as defining alternative metrics for zero values or implementing special markers.

- **Expanding the Dataset:** Using only the STS Benchmark dataset may limit the model's generalization capability. Future studies could introduce additional datasets, including multilingual or cross-domain corpora, to enhance adaptability to diverse textual contexts.

- **Parameter Optimization:** Further hyperparameter tuning, such as adjusting learning rate, batch size, or the number of training epochs, could be explored to better adapt the model to various tasks and scenarios, improving overall performance.

Replicability:

The code, input datasets, output results, and detailed instructions are available on our GitHub repository: https://github.com/jyz0328/Semantic-Similarity-Detection.