SEMANTIC TEXTUAL SIMILARITY

TEAM PLAYER NO.456

- Sasidhar Chavali - 2021101111
- Rohit Gowlapalli - 2021101113
- Abhinav Reddy Boddu - 2021101034

INTRODUCTION

Semantic Textual Similarity (STS) quantifies the extent to which two text snippets share the same meaning. Given a pair of sentences, the model assigns a similarity score on a continuous scale from 0 to 1,where 0 denotes no semantic overlap and 1 indicates full equivalence.

The model's performance is evaluated by calculating the correlation between its similarity scores and human judgments.

DATASETS

SICK DATASET

- Focuses on compositional distributional semantics.
- Contains sentence pairs labeled with relatedness (1–5) and entailment.
- Relatedness scores normalized to [0, 1] for our task.

STS BENCHMARK

- 8,628 sentence pairs from news, captions, and forums.
- Curated from SemEval STS tasks (2012–2017).
- Human-annotated similarity scores ensure label quality.

FINAL DATASET STRATEGY

- Initially included MSR & Quora, but removed due to binary label bias.
- Final dataset: only SICK + STS Benchmark.
- Split: 70% train, 10% validation, 20% test.
- Corresponding splits from both datasets merged for model use.

METRICS

PEARSON CORRELATION

- Semantic similarity is typically framed as a continuous, real-valued regression task (e.g., predicting scores like 0.3, 0.7, 0.9).
- We care about how well the predicted scores linearly track the true human-labeled similarity scores.
- Pearson measures the *strength and direction of the linear relationship* between predicted and true scores.
- almost all official benchmarks (like STS-B, SICK) use this.

SPEARMAN CORRELATION

- Measures rank order consistency, ignoring the actual distance between scores.
- Doesn't tell you if your predicted scores match the magnitude or scale — just the order.
- Good as a secondary metric to check if ranking quality is preserved, but not as the main metric.

METRICS

MEAN ABSOLUTE ERROR (MAE)

- Measures the average absolute deviation between predicted and true scores
- Sensitive to scale shifts or constant offsets; doesn't capture whether the relative movement between pairs is correct.
- Okay as a sanity check, but not meaningful if since the main goal is relational similarity.

Mean Squared Error (MSE)

- Same as MAE, but penalizes large errors more heavily.
- Can over-amplify outliers, and like MAE, ignores rank or relational consistency.
- Mostly useful to check numeric fit, but often misleading for similarity tasks where we care more about correlations.

METRICS

Why choose Pearsons?

- Pearson’s correlation measures linear agreement
- Captures both rank and value agreement
- Scale-invariant, robust to shifts and scaling

N-GRAM BASED APPROACH

Statistical:

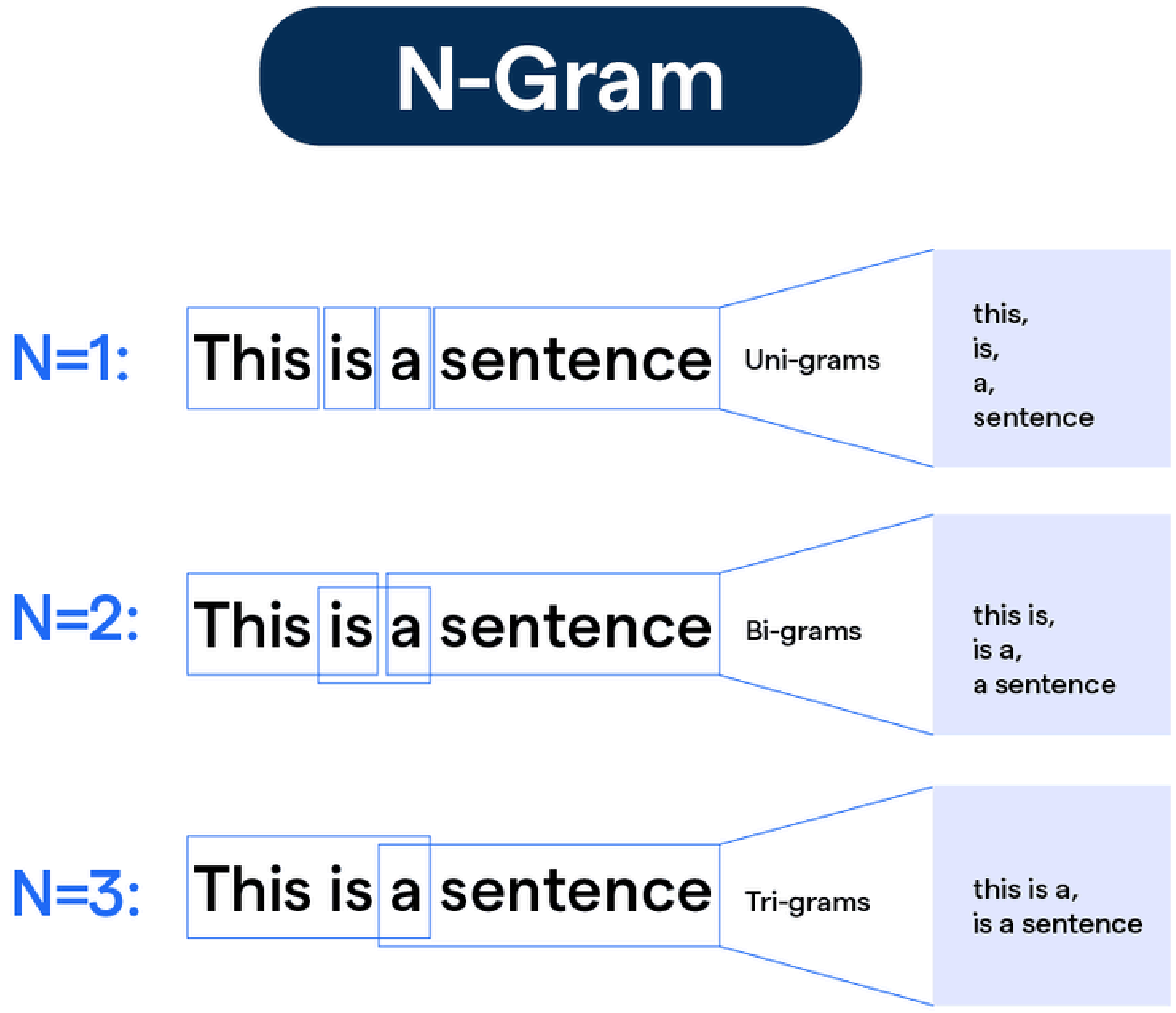
- Compared word/n-gram overlaps

Semantic (WordNet-based):

- Retrieved synsets for each token
- Aligned token pairs with similarity > 0.3
- Compared all tokens across both sentences

Example:

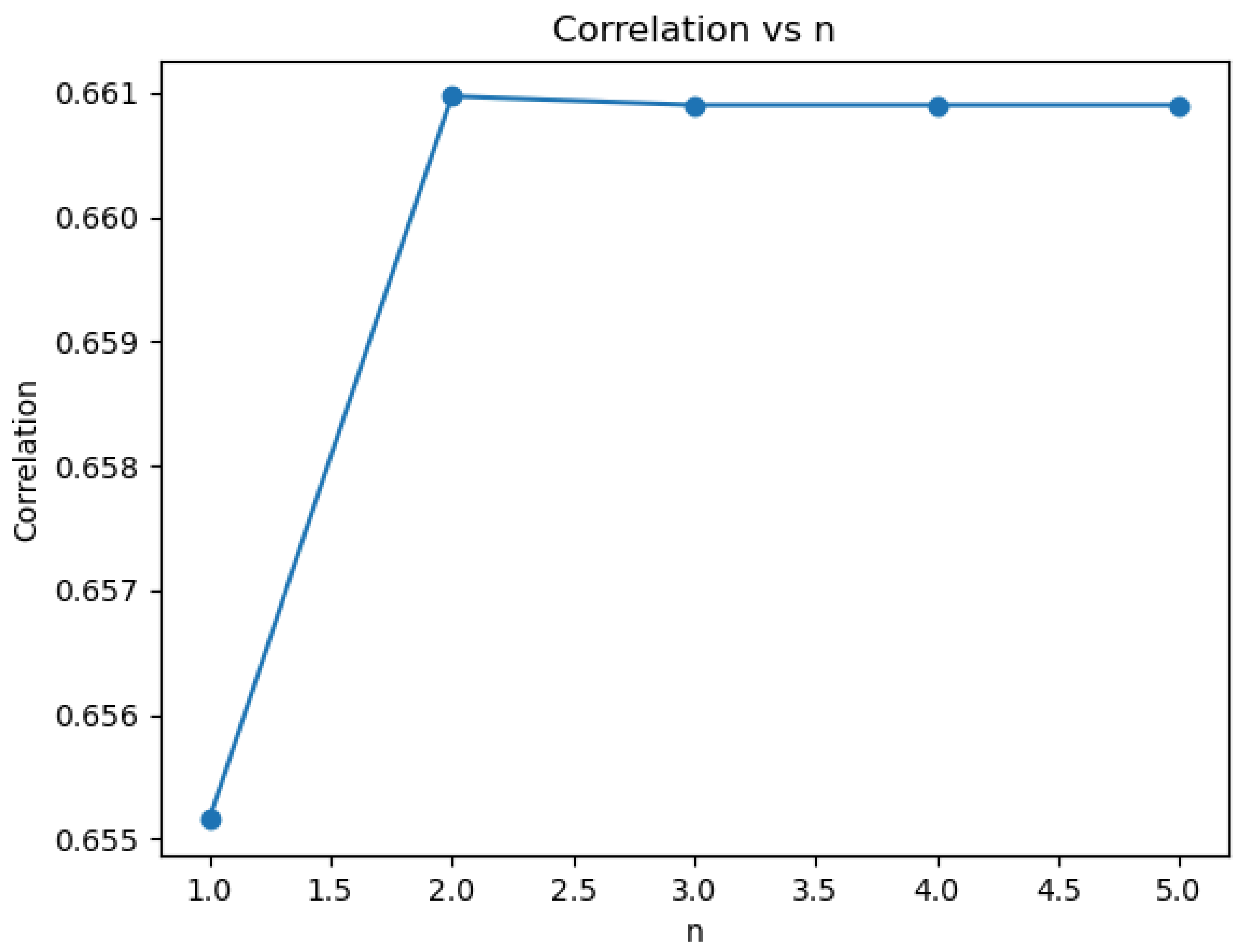
- "began" ↔ "started", "journey" ↔ "trip" (semantic match)



$$Sim(s1, s2) = \frac{n \cdot \sum_{al \in AL_{s1,s2}} al.s}{|T1| + |T2|}$$

Where, |T1| and |T2| are the number of tokens in sentence 1 and sentence 2 respectively. 'n' is the number of tokens contributing to the similarity score 's'.

ANALYSIS



- The Pearson correlation values vary with increasing N values from 1 to 5. The correlation starts at 0.6551 for N = 1 and slightly increases to 0.6609 by N = 3. Beyond this point, the correlation remains constant for N = 3, 4, and 5, indicating no further improvement.
- This pattern suggests that increasing N helps capture more meaningful patterns up to N = 2, but beyond that, there is no added benefit. This likely indicates that the information captured by phrases longer than 2 words does not contribute significantly to the correlation, possibly due to the absence or rarity of meaningful longer n-grams in the dataset.

BUT...

Ignores word order and semantics. Fails on paraphrases and reordered sentences

sentence-1: A man is playing a bamboo flute
sentence-2: A flute is being played by the man

True: 0.86, predicted ~0.86 (ok here), but fails on:

sentence-1: A bike is next to a couple women
sentence-2: A child next to a bike
True: 0.40 → **overestimated due to "bike"**

Doc2Vec to THE RESCUE

TRY SENTENCE-LEVEL EMBEDDINGS FROM AN UNSUPERVISED, PRETRAINED MODEL.

AIMED TO GO BEYOND WORD COUNTS AND GET FIXED-LENGTH SENTENCE VECTORS CAPTURING OVERALL MEANING.

Doc2Vec

Embedding Generation

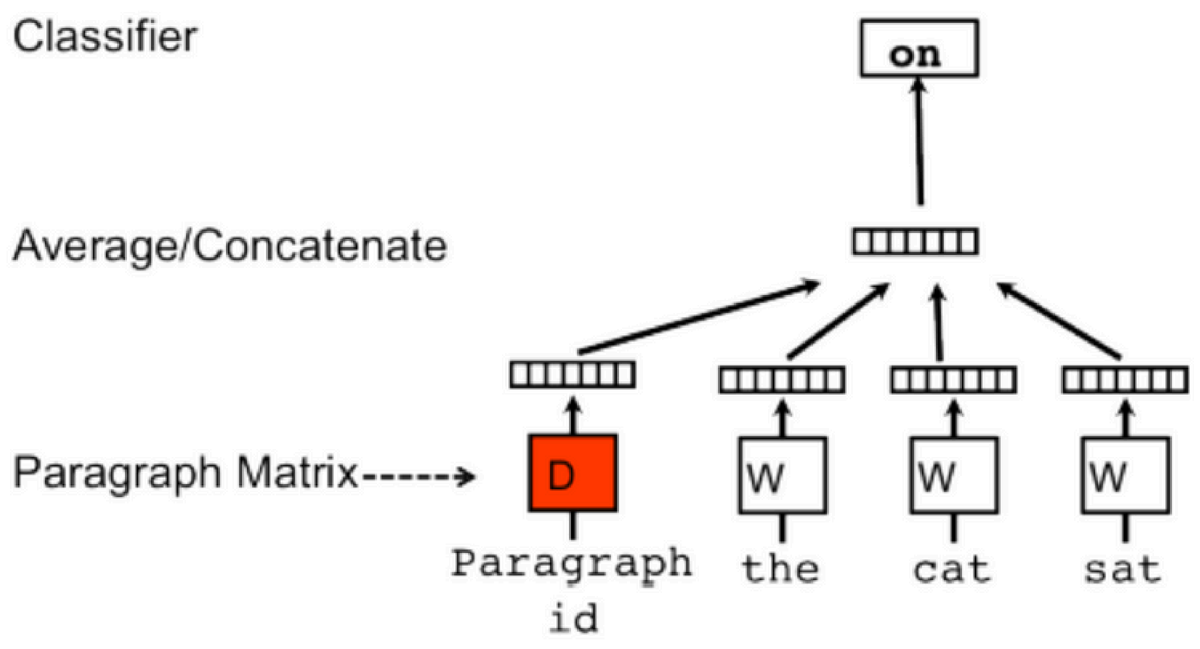
- Model: Doc2Vec (vector size = 25, window = 6)
- Sentences tokenized & converted to fixed-size vectors
- Embeddings inferred via infer_vector()

Similarity Methods

- **Cosine Similarity (Normalized):** Measures the angle between two sentence vectors. The score is scaled to fall within a 0–1 range for interpretability.
- **BiLSTM Regression:** Concatenates both sentence embeddings, passes them through a bidirectional LSTM, and uses a final layer to predict a similarity score. Trained using Mean Squared Error loss.

Training

- BiLSTM: 10 epochs
- Batch Size: 10, Learning Rate: 0.001
- Pearson Correlation:
  - Train: 0.82
  - Validation/Test: ~0.40



BUT...

CONTEXT-INDEPENDENT; STRUGGLES WITH NUANCED SIMILARITY OR SENTENCE STRUCTURE.

CNN TO THE RESCUE

USE LOCAL PATTERN DETECTORS; LET THE MODEL LEARN N-GRAM-LIKE FEATURES FROM DATA.

GO BEYOND FIXED EMBEDDINGS; LEARN MEANINGFUL LOCAL PATTERNS.

NEURAL NETWORK MODELS - CNN

Preprocessing & Feature Engineering

- Lowercased & punctuation removed , Tokenized using NLTK
- GloVe embeddings (unknowns → zero vector)
- Padded to 30 tokens

Added features:

- Word overlap flag , Numeric match flag , One-hot POS tags (NLTK)

CNN for Sentence Embeddings

- 1D CNN (300 filters, filter size = embedding dim)
- ReLU activation + max pooling
- Fixed-length sentence vector
- No dropout; early stopping used

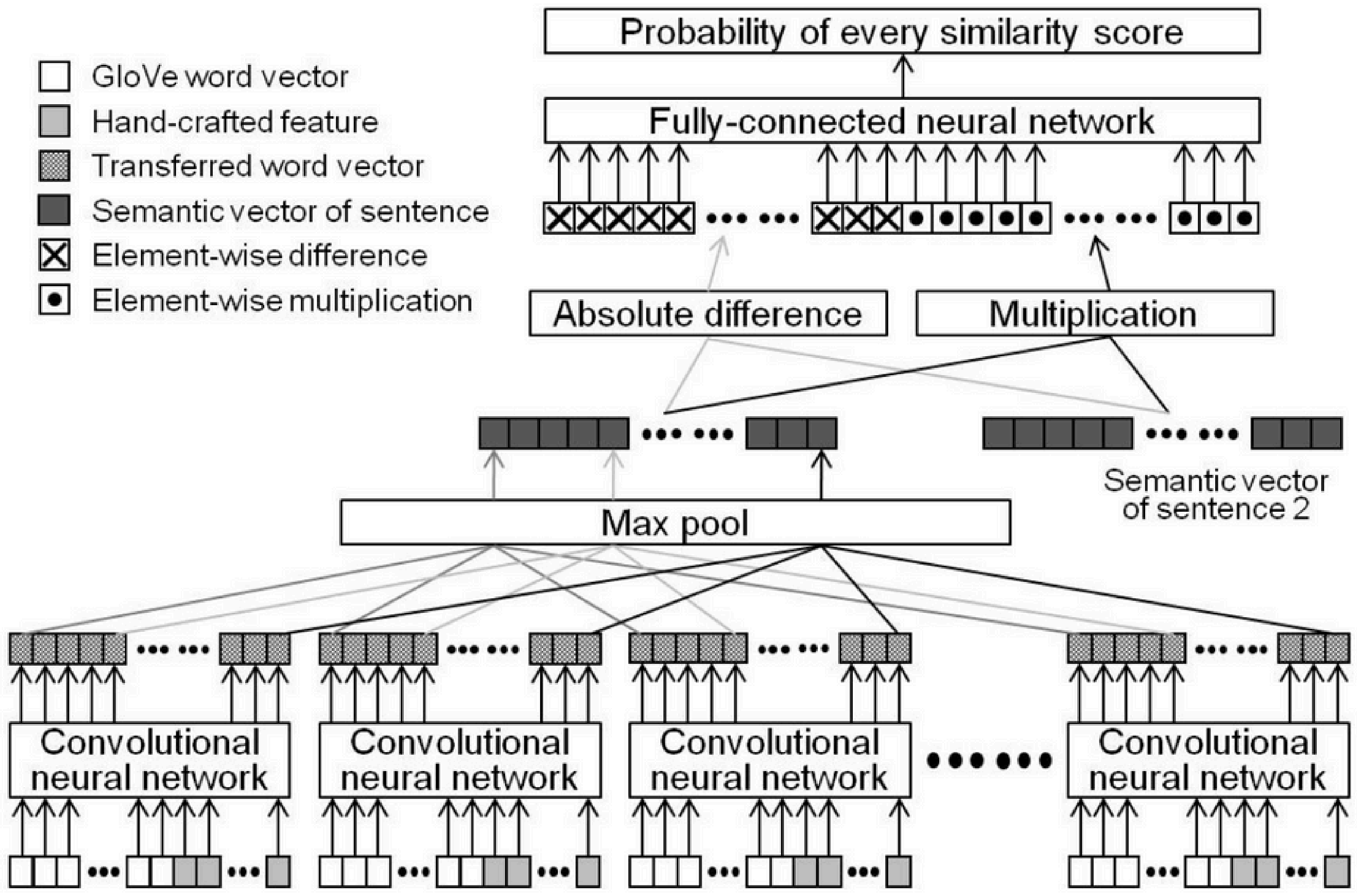
Semantic Similarity Scoring

- Concatenated absolute diff + Hadamard product
- Captures contrast + commonality

Fully Connected Network (FCNN)

- Input: 600-dim difference vector
- FC Layer 1: 300 units, tanh
- FC Layer 2: 6 softmax outputs (STS labels)
- No regularization/dropout

HCTI at SemEval-2017



Pearson Coefficient - 0.7423

BUT...

Over-focus on keywords; cannot distinguish roles or structure.

Examples:

sentence-1: Supreme Court to hear Voting Rights Act case
sentence-2: Supreme Court to hear corporate human rights case
True: 0.28 → CNN: 1.02 (mistakes surface similarity for meaning)

sentence-1: A yellow vested person is doing road work
sentence-2: A person is doing well on a skateboard
True: 0.04 → CNN: 0.77 (fails due to similar surface structure)

RNN TO THE RESCUE

🧠💡 TRY SENTENCE-LEVEL EMBEDDINGS FROM AN UNSUPERVISED, PRETRAINED MODEL.

AIMED TO GO BEYOND WORD COUNTS AND GET FIXED-LENGTH SENTENCE VECTORS CAPTURING OVERALL MEANING.

NEURAL NETWORK MODELS - RNN

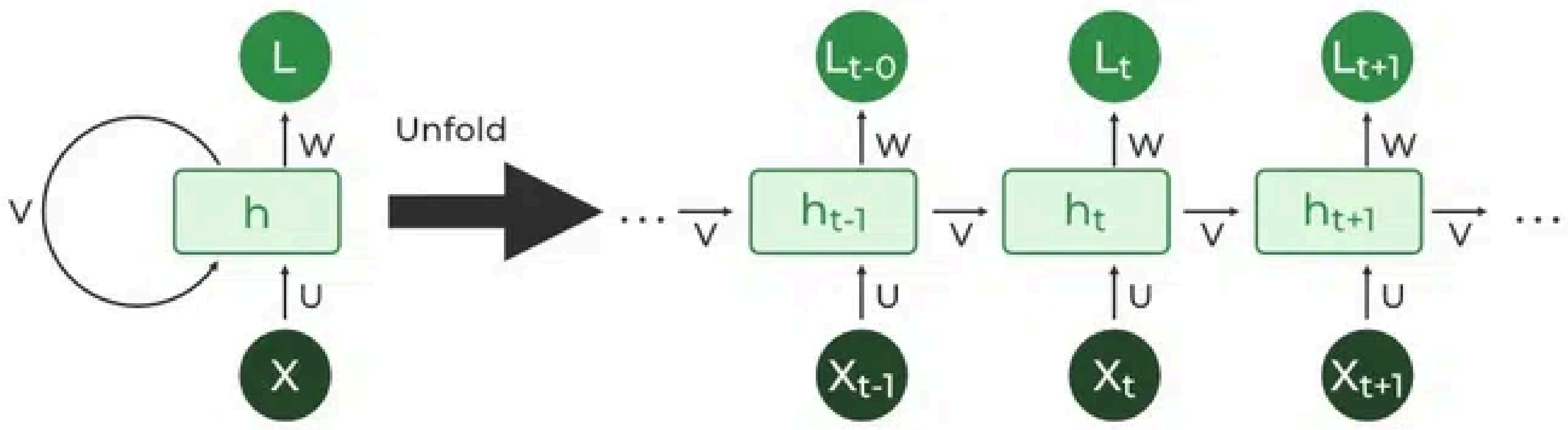
Preprocessing & Features
- Same steps as CNN: clean, lowercase, tokenize (NLTK)
- GloVe embeddings (840B Common Crawl)
- Padding: 30 tokens

Added features:
- Word match flag ,Numeric match flag , One-hot POS tags

RNN for Sentence Embedding
- RNN processes tokens sequentially
- Final hidden state = sentence representation
- Captures sentence semantics

Semantic Similarity Scoring
- Element-wise sum → vector 1
- Element-wise product → vector 2
- Concatenated (vector 1 + vector 2) for similarity features

Fully Connected Network (FCNN)
- FC Layer 1: 300 units, tanh
- FC Layer 2: 6 units, softmax (STS labels)
- No dropout or regularization



Pearson Coefficient - 0.6661

BUT...

FAILS ON LONG SEQUENCES OR SUBTLE SHIFTS IN MEANING.

sentence-1: You may have to experiment and find what you like
sentence-2: You have to find out what works for you

True: 1.0 → RNN predicts ~0.21 (underestimates clear paraphrase)

LSTM TO THE RESCUE

FIX RNN'S SHORT-TERM MEMORY, HANDLES LONGER DEPENDENCIES.

BETTER REMEMBER EARLIER CONTEXT OVER LONG SENTENCES.

NEURAL NETWORK MODELS - LSTM

Preprocessing & Features

- Same as CNN/RNN: lowercase, clean, tokenize (NLTK)
- GloVe embeddings (840B Common Crawl)
- Padding: 30 tokens

Added features:

• Word match flag , Numeric match flag , One-hot POS tags

LSTM for Sentence Embedding

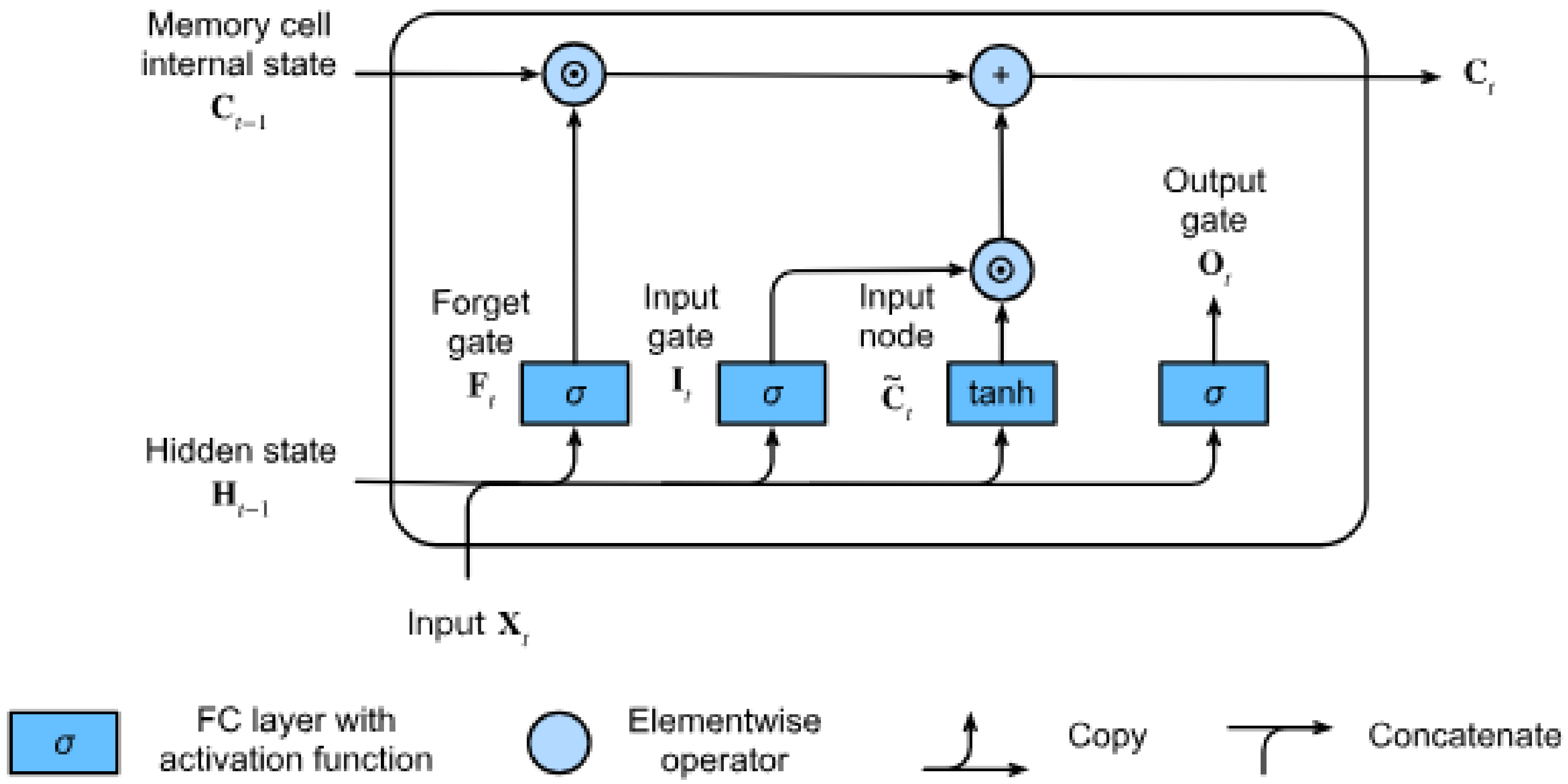
- LSTM processes tokens sequentially
- Final hidden state = sentence embedding
- Captures long-range dependencies

Semantic Similarity Scoring

- Element-wise sum + product → 2 vectors
- Concatenated → semantic difference vector

Fully Connected Network (FCNN)

- FC Layer 1: 300 units, tanh
- FC Layer 2: 6 units, softmax (STS labels)
- No dropout or regularization



Pearson Coefficient - 0.7112

ANALYSIS

Key Observations:

- All neural models outperform the N-Gram on test data, especially CNN and LSTM.
- LSTM > RNN: Handles long-range dependencies better via memory cells.
- CNN captures local patterns well and generalizes better than sequential-only models.
- N-Gram relies on surface-level word overlap—misses semantic nuance.

Reasons Neural Models Excel:

- Contextual understanding: Neural models encode word meaning in context.
- Word order and structure: RNN/LSTM capture sequential dependencies.
- Semantic features: CNNs learn phrase-level interactions.
- N-Grams fail on paraphrases/synonyms (e.g., "start" vs. "begin").

BUT...

Fails on logical negations or slight contradictions.

sentence-1: You should do it
sentence-2: You should prime it first

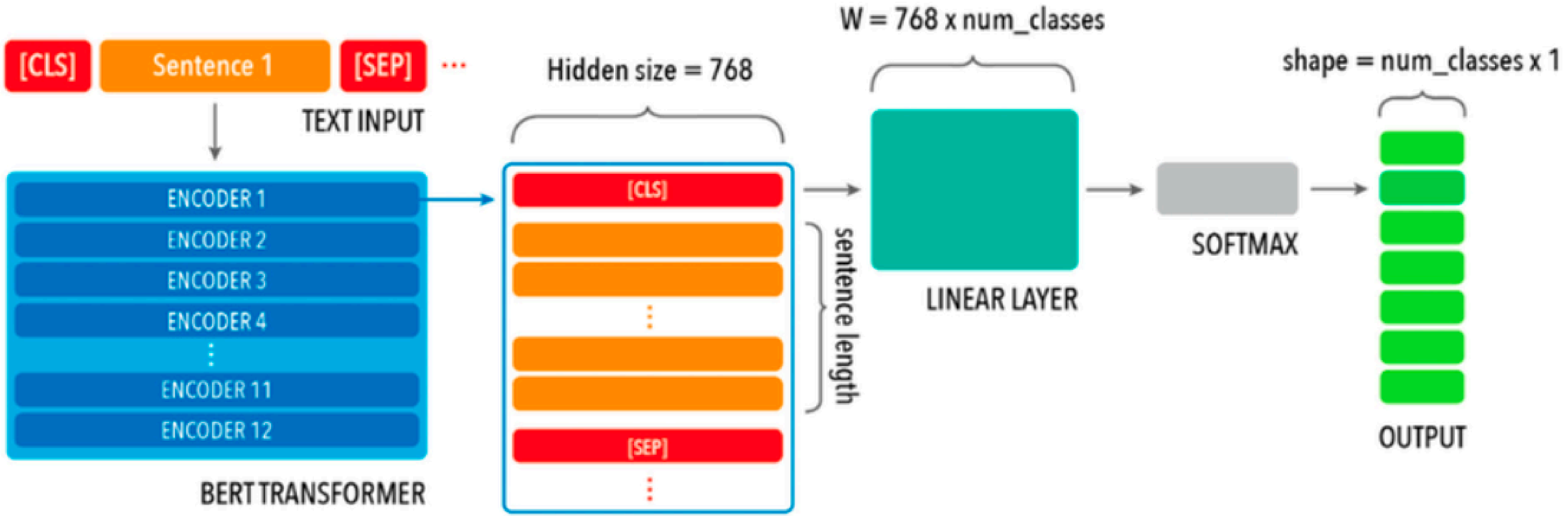
True: 0.0 → LSTM predicts ~0.83 (misses crucial difference: "prime" changes meaning)

BERT TO THE RESCUE

USE BIDIRECTIONAL CONTEXT; LEVERAGE PRETRAINED DEEP TRANSFORMER.

LEARN RICH CONTEXTUAL REPRESENTATIONS; CAPTURE COMPLEX MEANING.

BERT



[CLS]
Sentence 1
[SEP]
...
TEXT INPUT
ENCODER 1
ENCODER 2
ENCODER 3
ENCODER 4
ENCODER 11
ENCODER 12
BERT TRANSFORMER
Hidden size = 768
[CLS]
[SEP]
sentence length
W = 768 x num_classes
LINEAR LAYER
SOFTMAX
shape = num_classes x 1
OUTPUT

BERT

BERT-Base-Uncased

- 12 layers, 768 hidden units, 110M parameters
- Used for sequence classification (regression)

Fine-Tuning Setup

- Input Processing
- Tokenized & padded to max length (95th percentile)
- Classification Head
- Single regression layer for similarity score
- Loss Function
- Mean Squared Error (MSE)
- Optimizer
- Adam (lr=1e-5, betas=(0.5, 0.99))

Pearson Coefficient - 0.7902

BUT...

Sometimes fails on pragmatics or very subtle paraphrase logic.

- **millions of parameters (e.g., 110M in BERT-Base), requiring substantial GPU memory, especially during fine-tuning.**
- **Slow Inference & Training Time. Due to its deep architecture (12+ layers), BERT is computationally expensive and slower compared to lighter models like DistilBERT.**

sentence-1: Work into it slowly
sentence-2: It seems to work
True: 0.0 → BERT: 0.80 (doesn't detect unrelated intent)

sentence-1: You can do it, too
sentence-2: Yes, you can do it
True: 1.0 → BERT: 0.24 (underestimated paraphrase confidence)

DISTILLATION TO THE RESCUE

- Achieves a compact model with significantly reduced memory and computational cost, while preserving performance close to the teacher (BERT) model.
- Leverages both soft targets from the teacher and ground-truth labels to enhance the student model's ability to predict nuanced similarity scores.

Knowledge Distillation

Teacher Model

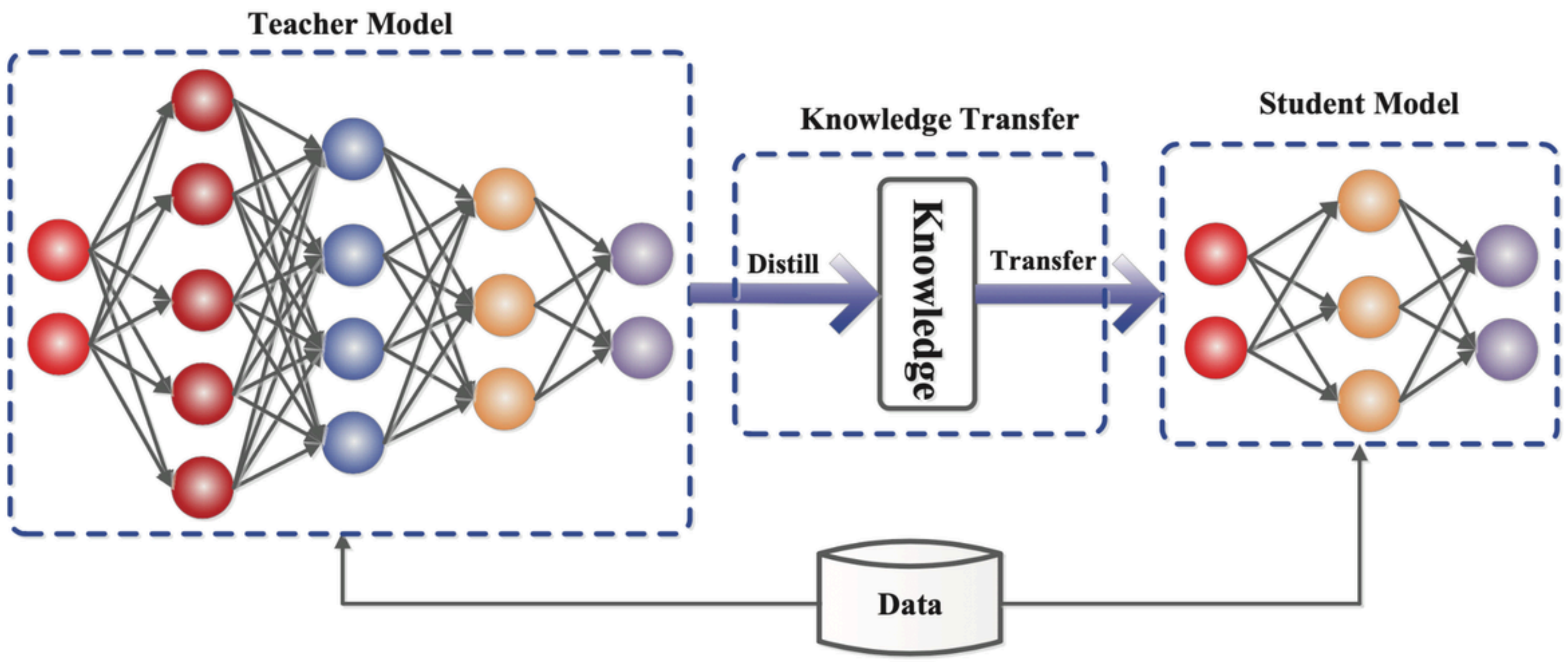
- Large fine-tuned BERT (high accuracy, slow inference)

Student Model

- MiniLM-L12-H384-uncased
- 12 layers, hidden size 384
- Lightweight, faster, good performance retention

Loss Function

- Distillation Loss = 0.5 × Hard Loss + 0.5 × Soft Loss
- α = 0.5 chosen after tuning



$$Loss = \alpha * hardLoss + (1 - \alpha) * softLoss$$

$$hardLoss = \frac{\Sigma(prediction - ground\ truth)^2}{N}$$

$$softLoss = \frac{\Sigma(prediction - BERT\ prediction)^2}{N}$$

Pearson Coefficient - 0.8942 ‼

Surprise!

Pearson Coefficient - 0.8942

To our surprise, this distilled model performed far better than the original BERT model.

- **Distillation is not just compression — it's a form of regularization:**
  - When you train a student model (MiniLM) using the soft labels from BERT, you're not just copying hard labels.
  - Soft labels from the teacher capture rich relational knowledge
  - This gives the student a smoother, better-shaped loss landscape, which often generalizes better on downstream tasks.

- **MiniLM architecture is extremely efficient:**
  - MiniLM was specifically designed for distillation
  - It matches the attention relations of large teachers.
  - getting a student that is not just smaller, but smarter in task-specific generalization.

Surprise!

Pearson Coefficient - 0.8942

To our surprise, this distilled model performed far better than the original BERT model.

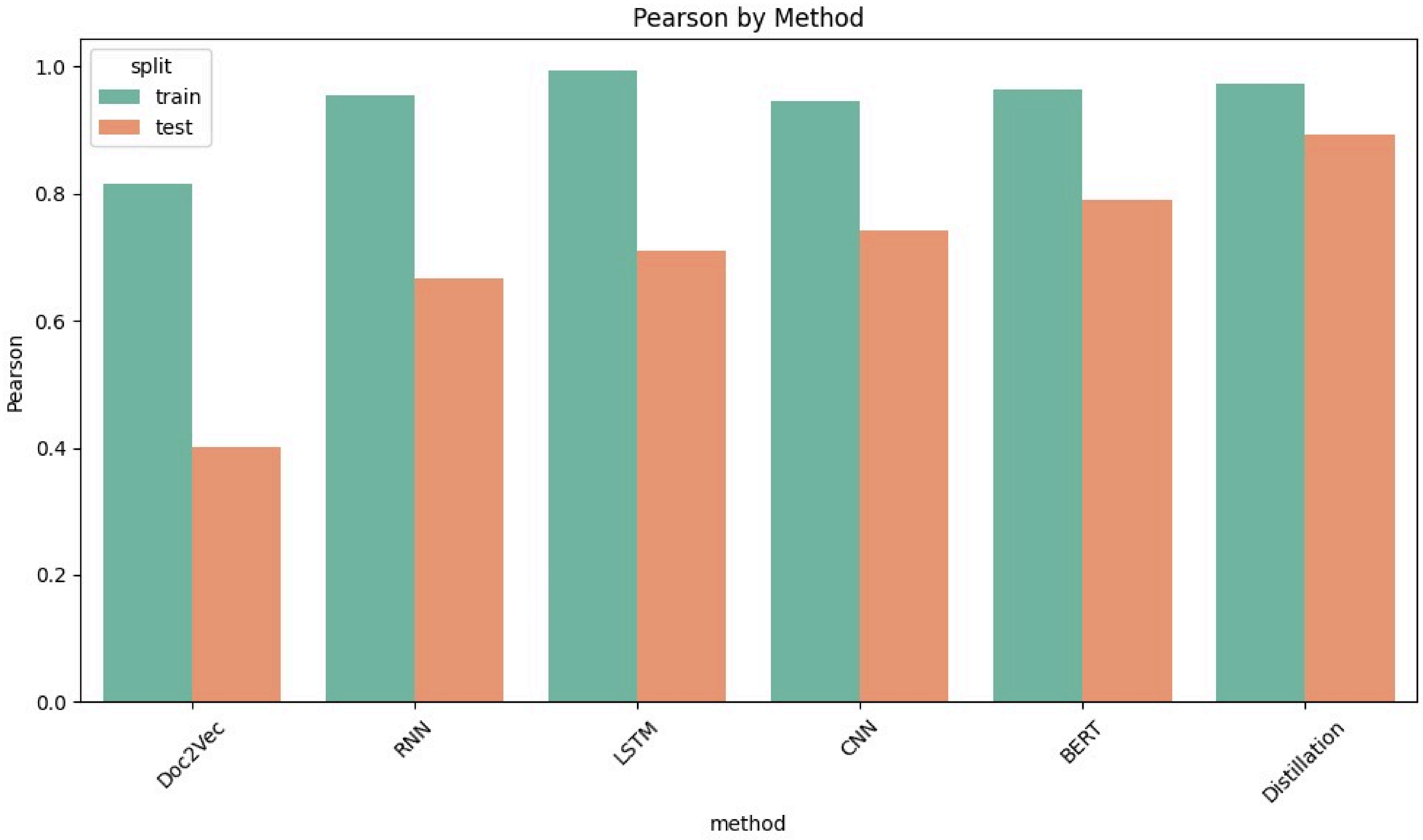
- **Semantic similarity tasks are forgiving to model size:**
  - Small datasets, Relatively shallow reasoning,Strong signal in relational patterns.
  - This means that smaller models can match or even outperform large models when distilled, especially when the task does not demand heavy linguistic depth or long-range reasoning.

- **Teacher bias filtering:**
  - Sometimes the teacher (BERT) overfits or carries bias from pretraining.
  - The student (MiniLM) may absorb only the useful signal and discard some noise during distillation, effectively making it a better version on the specific task.

RESULTS

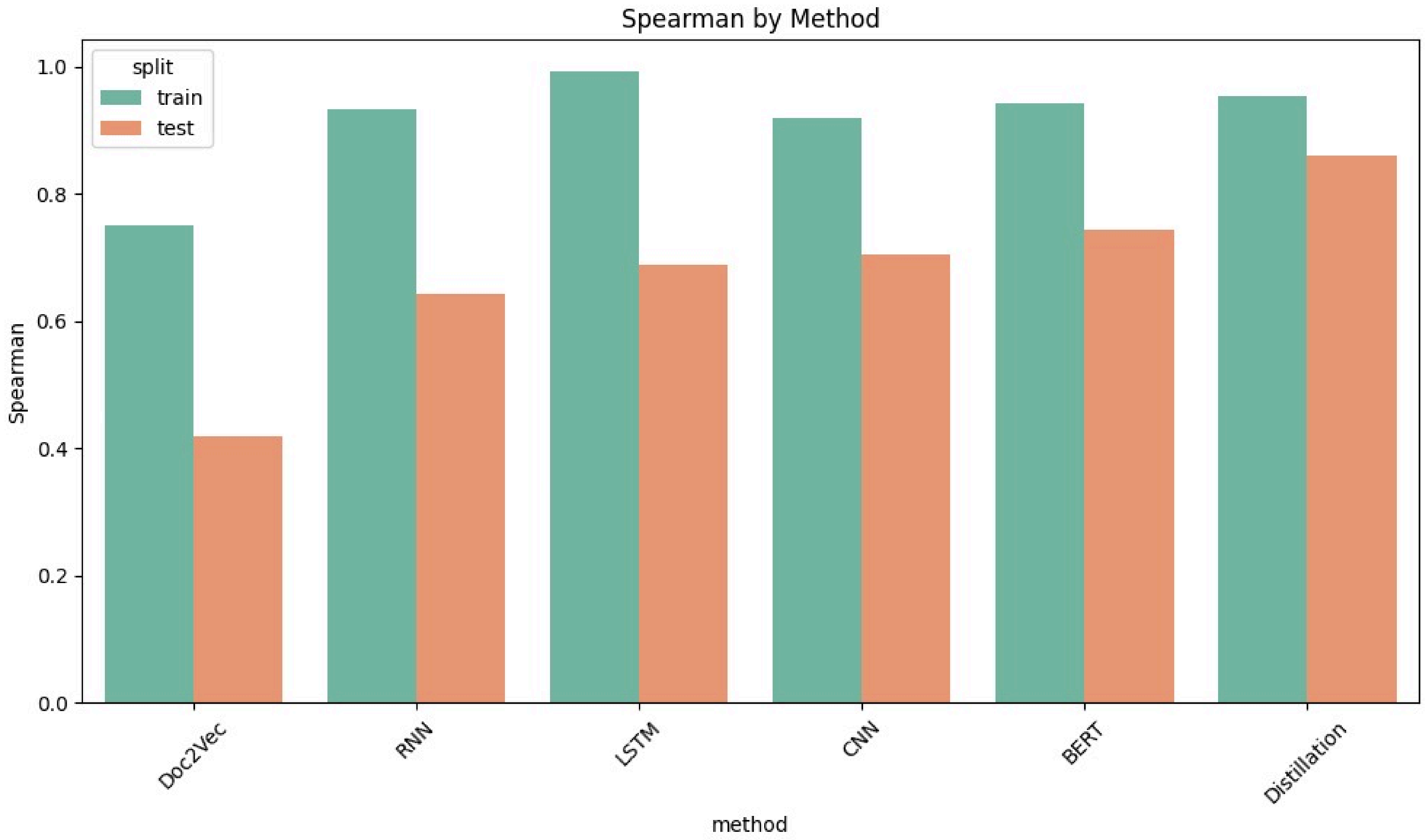
Model	Train Data Pearson Coefficient	Test Data Pearson Coefficient
N-Gram (N=2)	0.674	0.660973883729884
Doc2Vec BiLSTM	0.815907151976909	0.401701737067514
RNN	0.955988361808126	0.666618317525793
LSTM	0.993805120105462	0.711290768305084
CNN	0.944998063680744	0.742325919071415
BERT	0.96353090275641	0.790292680181099
Distillation	0.97275080364215	0.894339293111913

RESULTS



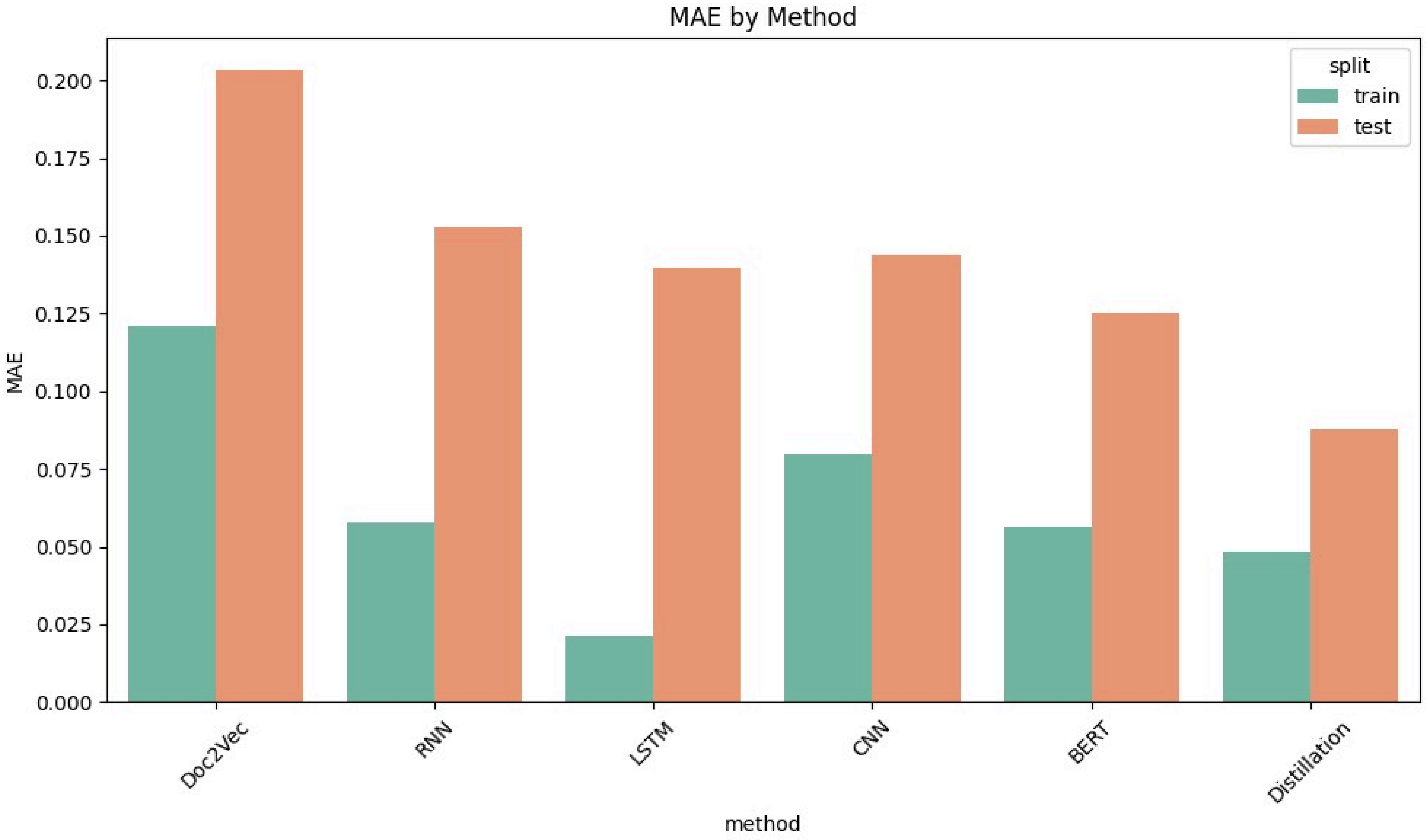
Pearson by Method
split
train
test
1.0
0.8
0.6
0.4
0.2
0.0
Pearson
Doc2Vec
RNN
LSTM
CNN
BERT
Distillation
method

RESULTS



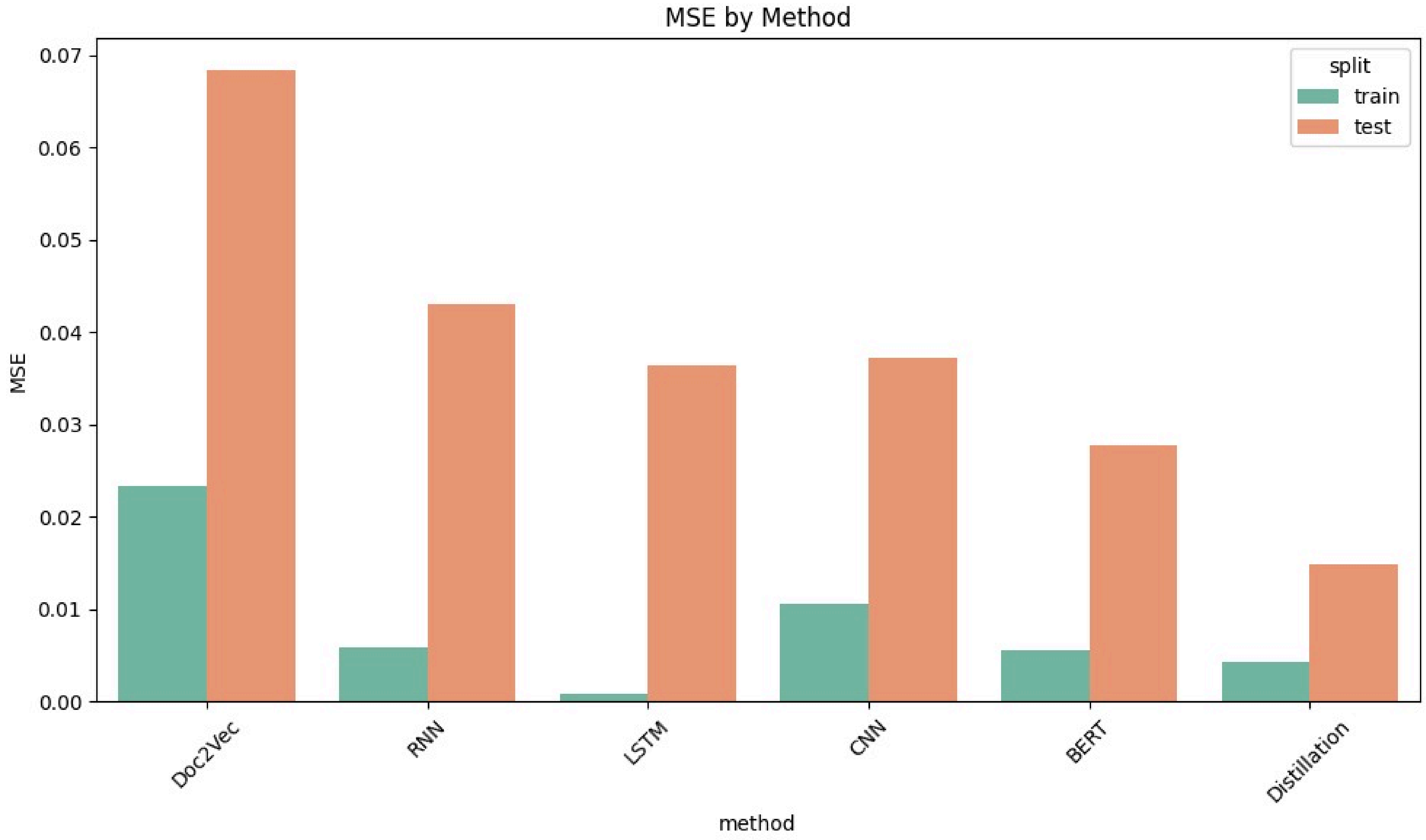
Spearman by Method
split
train
test
1.0
0.8
0.6
0.4
0.2
0.0
Spearman
Doc2Vec
RNN
LSTM
CNN
BERT
Distillation
method

RESULTS



MAE by Method
split
train
test
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
MAE
Doc2Vec
RNN
LSTM
CNN
BERT
Distillation
method

RESULTS



MSE by Method
split
train
test
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0.00
MSE
Doc2Vec
RNN
LSTM
CNN
BERT
Distillation
method

CONCLUSION

- Neural models (RNN, LSTM, CNN) outperform statistical baselines (N-Gram, Doc2Vec), confirming their strength in modeling semantic similarity.
- LSTM achieves the highest train score, indicating strong fitting capability and effective context handling.
- CNN generalizes better on test data than RNN/LSTM, suggesting better robustness.
- BERT performs well overall, leveraging deep contextual understanding.
- Distilled MiniLM surpasses all models on test data with best generalization (highest test Pearson), offering BERT-level accuracy with faster inference.

✅ Distillation provides the best balance between performance and efficiency.

THANK YOU