SEMANTIC TEXTUAL SIMILARITY

Team Linguists

OVERVIEW

- Introduction
- Datasets
- Experiments
- Results
- Analysis
- Conclusion

INTRODUCTION

- Our project focuses on Semantic Textual Similarity (STS), determining how alike two pieces of text are in meaning.
- We aim to develop a model that assigns a score ranging from 0 to 5 to indicate the similarity between two sentences.
- Our focus is on English-English sentence pairs.

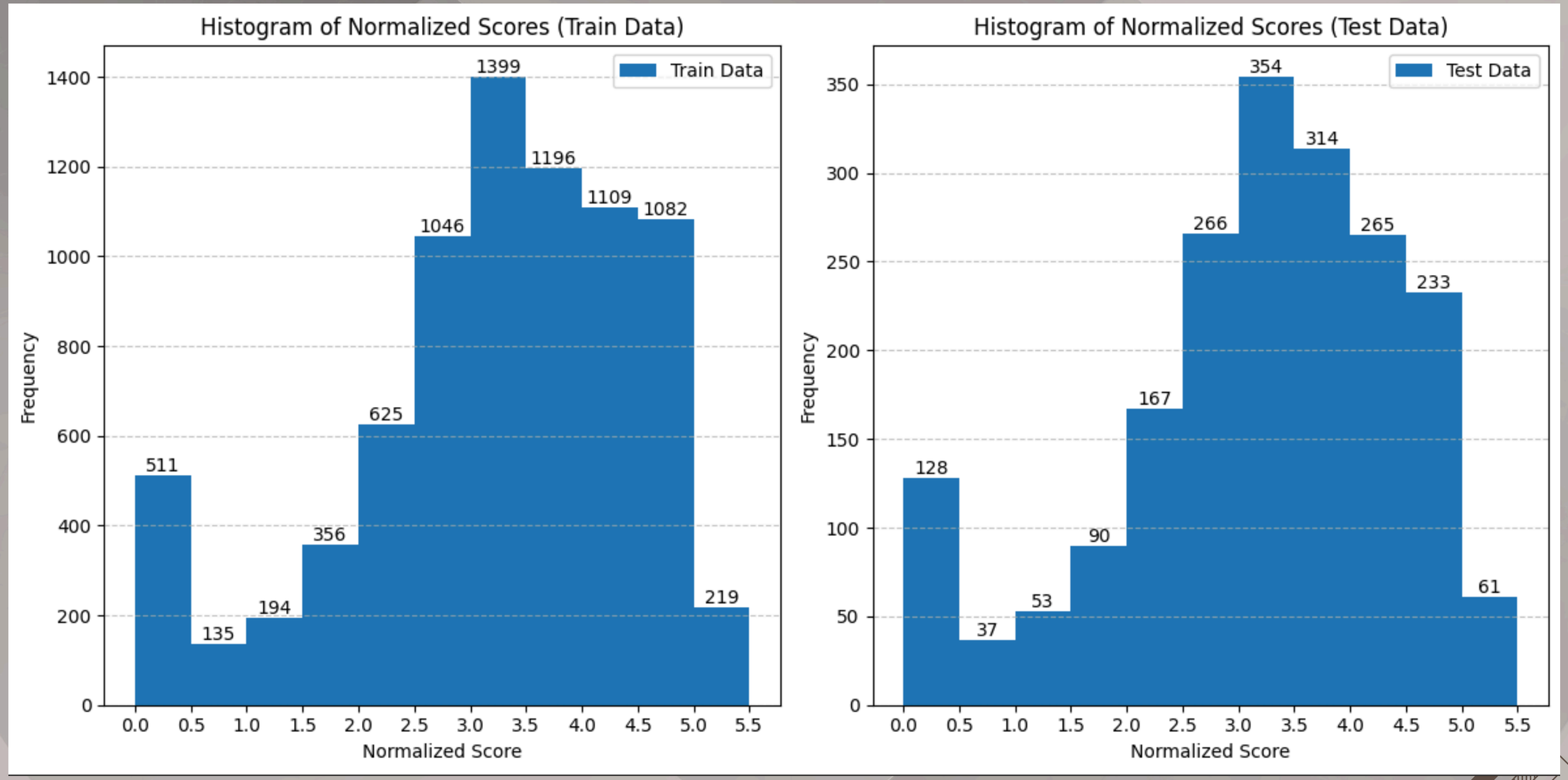
DATASETS

SICK dataset

- We used the Sentences Involving Compositional Knowledge (SICK) dataset, designed for studying compositional distributional semantics.
- It features numerous sentence pairs illustrating lexical, syntactic, and semantic phenomena.
- Each pair is annotated for relatedness and entailment, with relatedness scores ranging from 1 to 5.
- To align with our scale of 0 to 5, we normalized all labels to values between 0 and 1, then multiplied by 5 for consistency.

DATASETS

SICK dataset



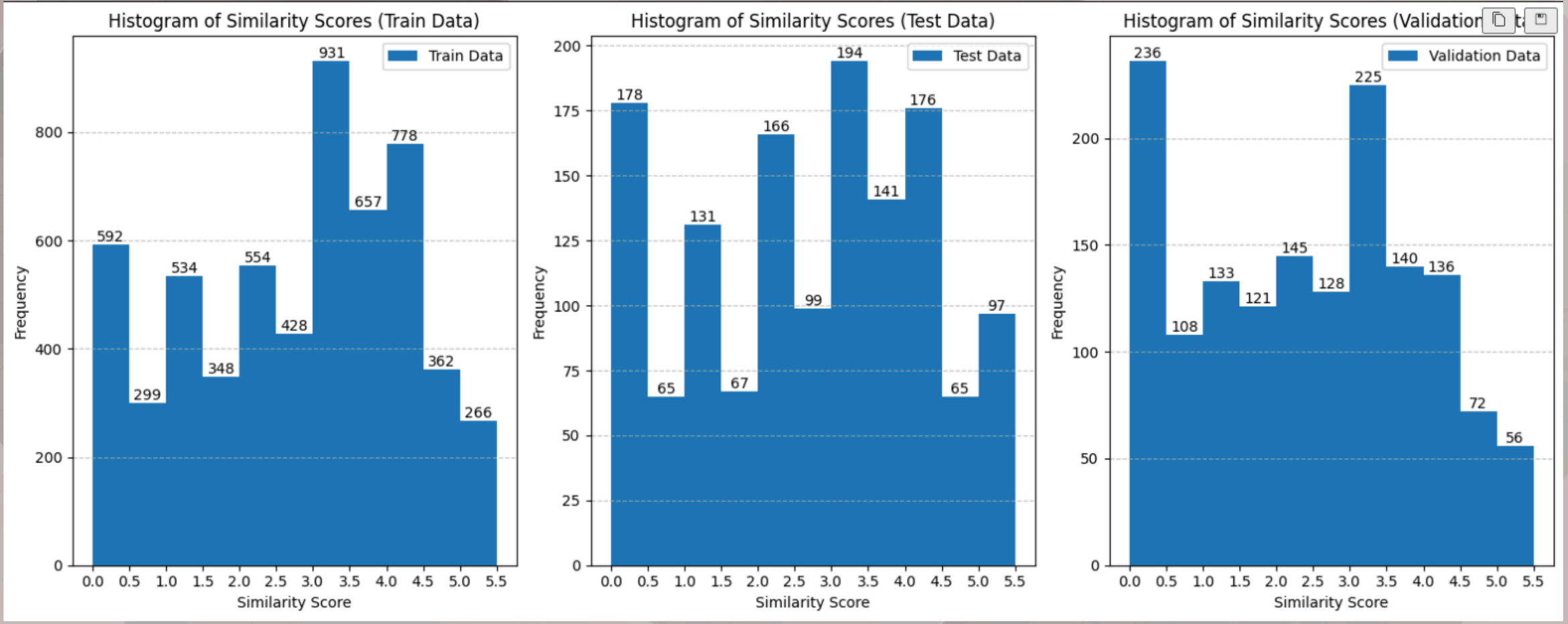
DATASETS

STS Benchmark

- We used the STS Benchmark dataset, containing 8,628 sentence pairs across news, image captions, and forum discussions.
- These sentences were carefully chosen from English datasets used in STS tasks during SemEval competitions (2012-2017), showcasing various language styles.
- A significant aspect of this dataset is its reliance on human judgments for similarity scores, ensuring reliable labels.

STS Benchmark

DATASETS



EXPERIMENTS

Statistical approach

- We tokenized and lemmatized both sentences, considering phrases up to a certain length (i.e., joining tokens up to n-grams). This helps capture phrases with similar meanings even if they're expressed differently.

- Next, using wordnet we obtain synsets for tokens. Calculate the similarity score for every synset pair i.e., for every token’s synset in the first sentence with the second sentence. The tokens with similarity scores greater than the threshold(0.3) are added to the alignments list.

EXPERIMENTS

Statistical approach

Example

Sentence 1: “The old guy died at the age of seventy.”

Sentence 2: “The old guy kicked the bucket at the age of 70.”

```
shravya@shravya-inspiron-11:~/Desktop/sem 6/INLP/Semantic_Textual_Similarity/Method-1$ python3 alignments.py
/usr/lib/python3/dist-packages/scipy/__init__.py:146: UserWarning: A NumPy version >=1.17.3 and <1.25.0 is required for thi
y (detected version 1.26.3
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
[('old', 'old', 1.0), ('guy', 'guy', 1.0), ('age', 'age', 1.0), ('70', 'seventy', 1.0), ('kick_the_bucket', 'die', 1.0)]
```

EXPERIMENTS

Statistical approach

- Once the alignments are obtained, the similarity score between two sentences is calculated as follows:

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

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

EXPERIMENTS

Neural network based approach
Data pre-processing

- All punctuations are removed and all words are lowercased.
- All sentences are tokenized by Natural Language Toolkit (NLTK).
- All words are replaced by pre-trained GloVe word vectors (Common Crawl, 840B tokens). Words not existing in the pre-trained embeddings are set to the zero vector.
- All sentences are padded to a static length l = 30 with zero vectors.

EXPERIMENTS

Neural network based approach
Data pre-processing

Hand-crafted to enhance the GloVe word vectors

- If a word appears in both sentences, add a TRUE flag to the word vector, otherwise, add a FALSE flag.
- The part-of-speech (POS) tag of every word according to NLTK is added as a one-hot vector.

EXPERIMENTS

Neural network based approach CNN

- A Convolutional Neural Network (CNN) with 300 one-dimensional filters with RELU activation function is passed over every token to get corresponding word embedding.
- Sentence level embeddings are calculated by max pooling over every dimension of the transformed word-level embedding.
- Semantic similarity between sentences is calculated using a semantic difference vector that concatenates absolute differences and element-wise multiplication of sentence embeddings.

EXPERIMENTS

Neural network based approach CNN

- A Fully connected Neural Network (FCNN) transforms the semantic difference vector to a similarity score used by STS, with two layers and tanh/Linear activation functions.

Semantic difference vector

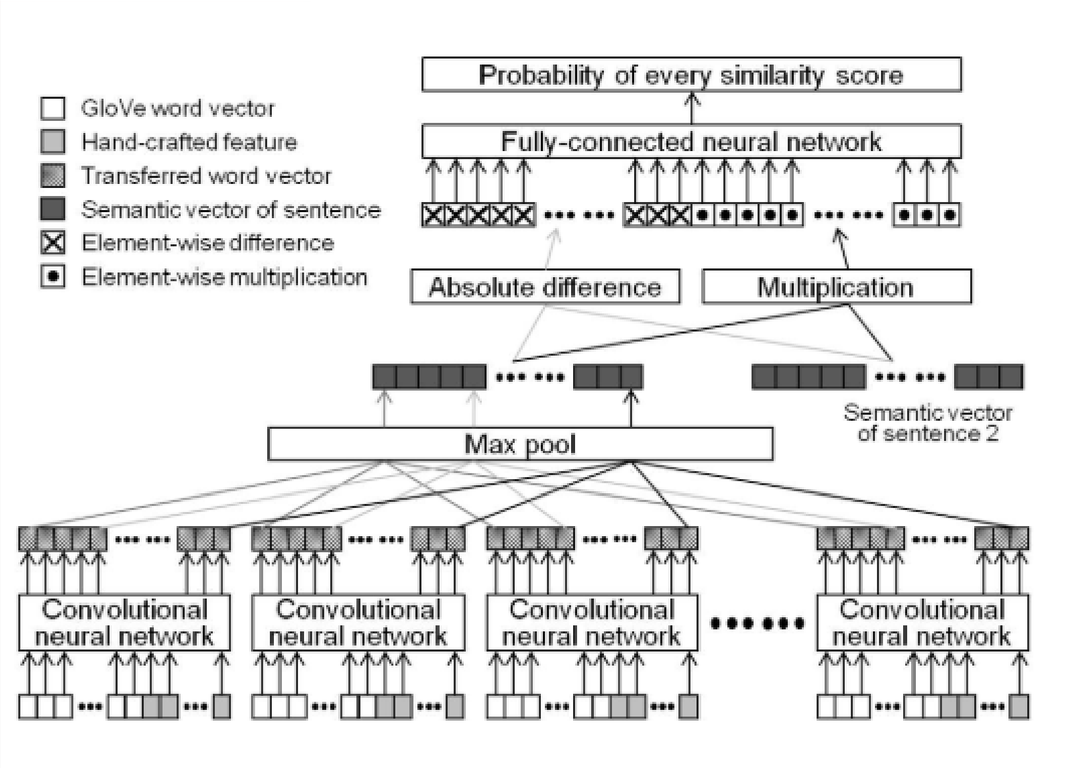
$$\vec{SDV} = (|\vec{SV}1 - \vec{SV}2|, \vec{SV}1 \circ \vec{SV}2)$$

SDV is the semantic difference vector, SV1 and SV2 are the semantic vectors of the two sentences

EXPERIMENTS

Neural network based approach
CNN

Overview of the system



EXPERIMENTS

Neural network based approach
RNN

- Pre-processed sentences are passed through RNN to get sentence-level embeddings.
- The sentence embeddings obtained using RNN are used for calculating semantic difference vectors.
- A Fully connected Neural Network (FCNN) transforms the semantic difference vector to a similarity score used by STS, with two layers and tanh/Linear activation functions.

EXPERIMENTS

Neural network based approach LSTM

- Pre-processed sentences are passed through LSTM to get sentence-level embeddings.
- The sentence embeddings obtained using LSTM are used for calculating semantic difference vectors.
- A Fully connected Neural Network (FCNN) transforms the semantic difference vector to a similarity score used by STS, with two layers and tanh/Linear activation functions.

EXPERIMENTS

Fine-tuning BERT

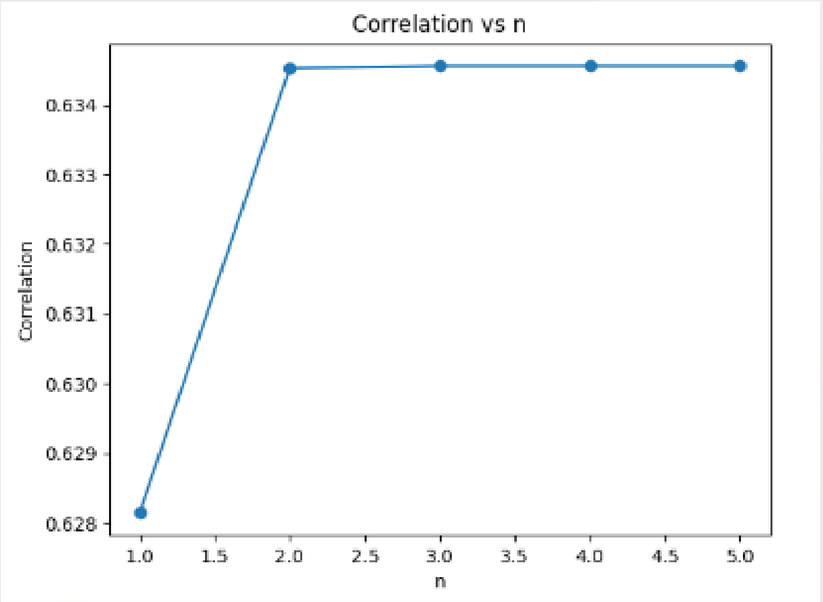
- Sentences are tokenized using the BERT tokenizer. The pair of sentences whose similarity is to be determined are concatenated using special tokens and passed as input to BERT model.

- The BERT model is frozen and the contextual embeddings obtained from the BERT model are then passed to a fully connected neural network to predict the similarity score.

RESULTS
STATISTICAL

- **Pearson's R** is used as a metric to measure the correlation between predicted and actual scores.
- **STS Benchmark**

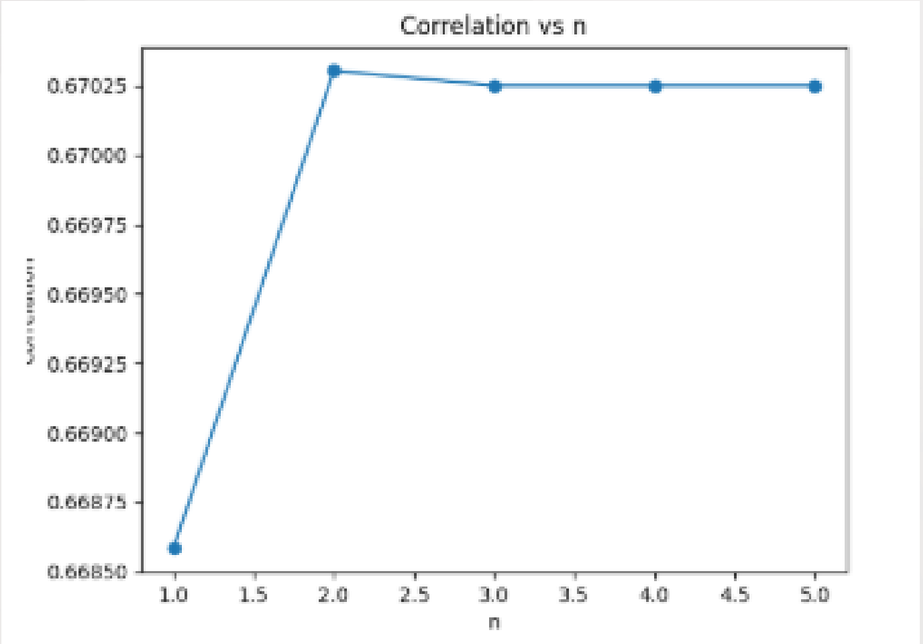
```
Correlation for n = 1: 0.6281480789895671
Correlation for n = 2: 0.6345255804180845
Correlation for n = 3: 0.6345563104924982
Correlation for n = 4: 0.6345563104924982
Correlation for n = 5: 0.6345563104924982
```



RESULTS
STATISTICAL

- SICK Dataset

```
Correlation for n = 1: 0.6685840197683521
Correlation for n = 2: 0.6703033712436781
Correlation for n = 3: 0.6702504205407234
Correlation for n = 4: 0.6702504205407234
Correlation for n = 5: 0.6702504205407234
```



RESULTS
STATISTICAL

- **Analysis**

The maximum correlation value obtained for SICK data is 0.67 and for STS benchmark is 0.63. Though the correlation values increased slightly for n=1,2 and 3, we observed that there is no much significant increase in correlation for n = 3,4 and 5. This suggests that there are no idioms/phrases containing 4 or more words.

RESULTS
NN - BASED AND BERT

MODEL	STS	SICK
CNN	0.628	**0.8059**
RNN	0.619	-
LSTM	0.707	0.799
BERT	**0.741**	0.736

RESULTS
NN - BASED AND BERT

ANALYSIS

On the STSbenchmark dataset, LSTM-based and BERT models show the highest correlation coefficients of 0.707 and 0.741, respectively. This indicates their proficiency in extracting semantic features and capturing sentence similarities. CNN and RNN models exhibit lower coefficients (0.628 and 0.619), suggesting they may be less effective in capturing nuanced semantic relationships.

When assessed on the SICK dataset, the CNN-based model leads with a correlation coefficient of 0.8059, surpassing others. The LSTM model follows closely with a coefficient of 0.799, indicating consistent performance across datasets. However, BERT, despite its strong performance on STSbenchmark, shows a lower coefficient of 0.736 on SICK, suggesting variability in its effectiveness across different datasets.

RESULTS
NN - BASED AND BERT

ANALYSIS

The RNN-based model struggled on the SICK dataset, indicating limitations in capturing semantic textual similarities in certain contexts. Differences in label distributions between SICK and STSbenchmark datasets, along with variations in sentence types and semantic complexities, could influence model performance.

Deep learning architectures should be chosen based on dataset and task specifics. LSTM and BERT perform well overall, but CNN excels on the SICK dataset. Knowing the strengths and limitations of each architecture is vital for building reliable semantic textual similarity models.

RESULTS
ENSEMBLE

ANALYSIS

- Ensemble approaches, combining multiple deep learning architectures, significantly enhance semantic textual similarity models' performance.
- On **STSbenchmark**, combining RNN, CNN, LSTM, and BERT achieves a Pearson's correlation coefficient of **0.760**, surpassing individual models.
- Similarly, on **SICK**, combining CNN, LSTM, and BERT outperforms all individual models with a correlation coefficient of **0.8579**.
- Ensembles leverage diverse strengths, leading to superior performance across datasets and scenarios, enhancing accuracy and reliability in natural language understanding tasks.

CONCLUSION

In conclusion, while deep learning (DL) architectures like LSTM and BERT demonstrate strong performance in semantic textual similarity tasks, their effectiveness varies depending on the dataset. CNN emerges as particularly effective on the SICK dataset, emphasizing the importance of architecture selection tailored to specific requirements. Ensemble methods, combining multiple DL architectures, significantly enhance model performance, outperforming individual models on both STSbenchmark and SICK datasets. Moreover, **statistical methods < DL < DL Ensemble**, indicating the superiority of ensemble methods in capturing semantic textual similarity effectively. Overall, harnessing the collective intelligence of multiple models through ensemble approaches offers a robust solution for improving accuracy and reliability in natural language understanding tasks.

THANK YOU