Sentiment Analysis

Juan Jose Gordillo - A00407891Anderson Olave Ibargüen - A00408142

Introduction and Context

The proliferation of online reviews has made sentiment analysis a critical tool for understanding customer opinions. This project develops and compares machine learning models to classify product reviews as positive or negative.

We evaluate three neural network architectures: Dense Networks, vanilla RNN, and LSTM networks, identifying the most effective approach for sentiment classification in multi-domain reviews.

Project Objectives

01

Data Preprocessing

Analyze and prepare a multi-domain review dataset

02

Baseline Models

Implement baseline models to establish minimum performance thresholds

03

Neural Architectures

Develop and optimize Dense NN, RNN, and LSTM networks

04

Comprehensive Evaluation

Evaluate all models using comprehensive performance metrics

05

Comparative Analysis

Identify the best architecture for sentiment analysis tasks

Dataset and Preprocessing

Data Source

UC Irvine ML Repository: 3,000 reviews from Amazon, IMDb, and Yelp

Perfect Balance

500 positive and 500 negative reviews per source

Stratified Split

2,400 training samples and 600 test samples

The preprocessing pipeline transformed raw text into numerical sequences through text cleaning, stopword removal, tokenization with a 10,000-word vocabulary (final result: 4,632 unique words), and uniform padding to 100 tokens.

Impact of Preprocessing

34.2%

Vocabulary Reduction

From 8,015 to 5,277 unique words, improving semantic signal

47.6%

Length Reduction

From 11.83 to 6.20 average words per review

4,632

Final Vocabulary

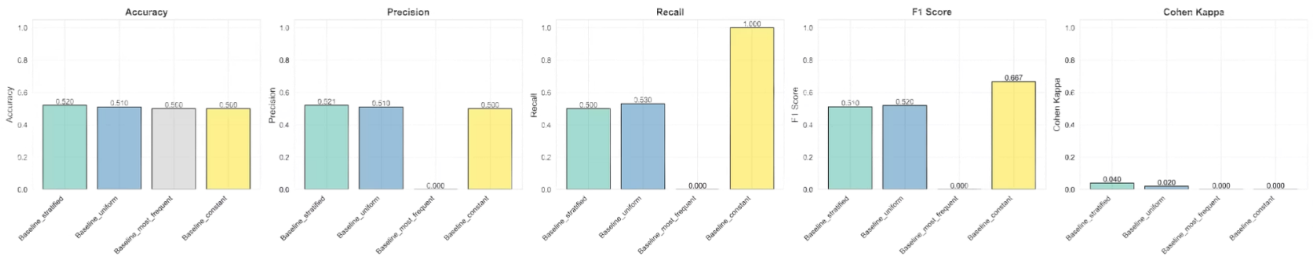
Unique words optimized for training

After preprocessing, the most frequent words correlated directly with sentiment: negative reviews contained "bad", "dont", "worst", while positive ones featured "great", "good", "best".

Base Models and Dense Network

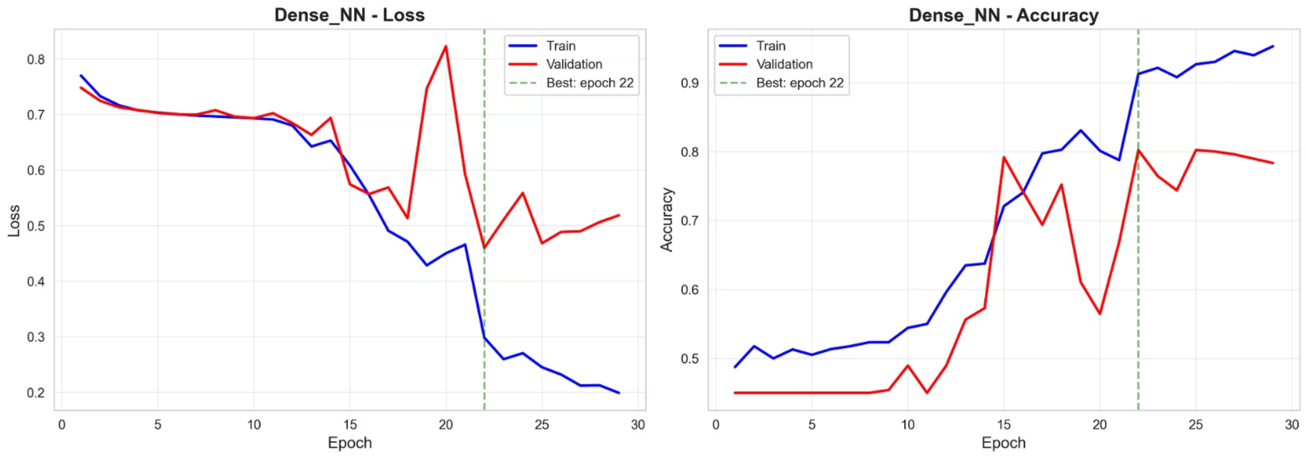
Dummy Classifiers

We established four baseline strategies: most frequent, stratified, uniform, and constant. The constant strategy achieved the highest baseline F1-score of 0.6667.



Dense Neural Network

Feedforward architecture with 128-dimensional embedding, global average pooling, three dense layers (128, 64, 32 units) with ReLU activation, 30% dropout, and 617,729 trainable parameters.



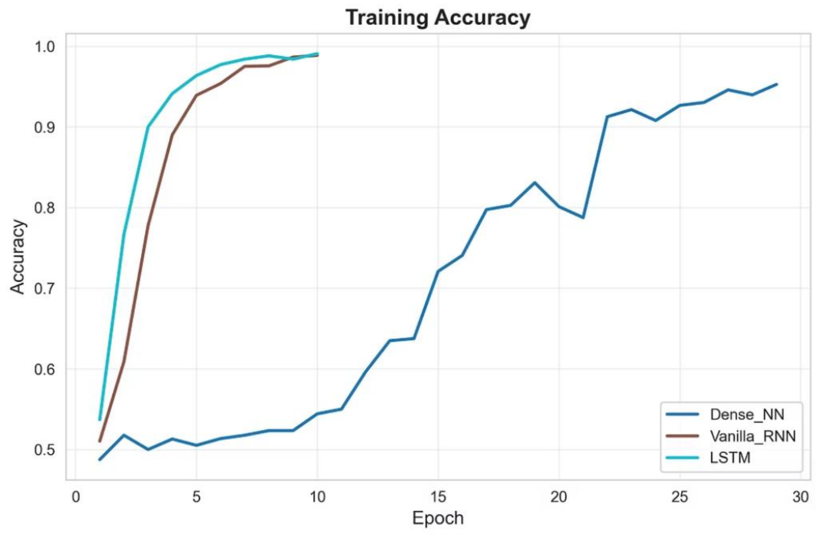
Vanilla RNN and LSTM Network

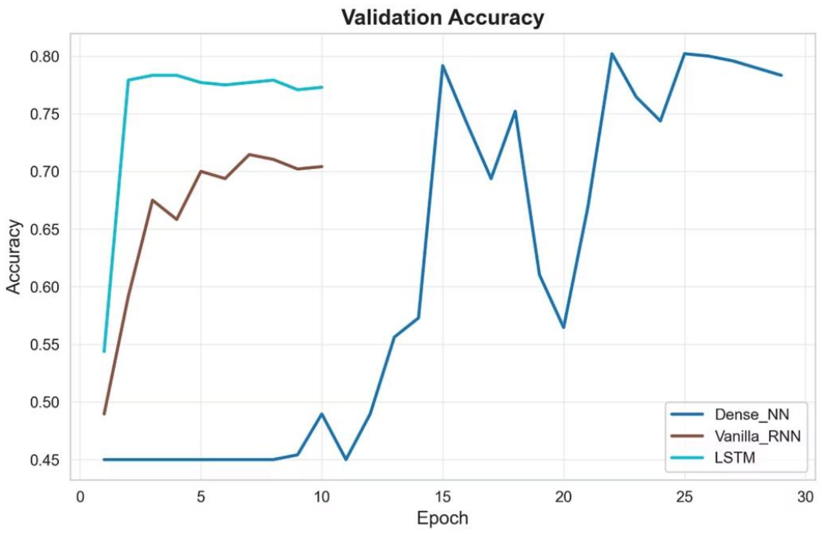
Vanilla RNN

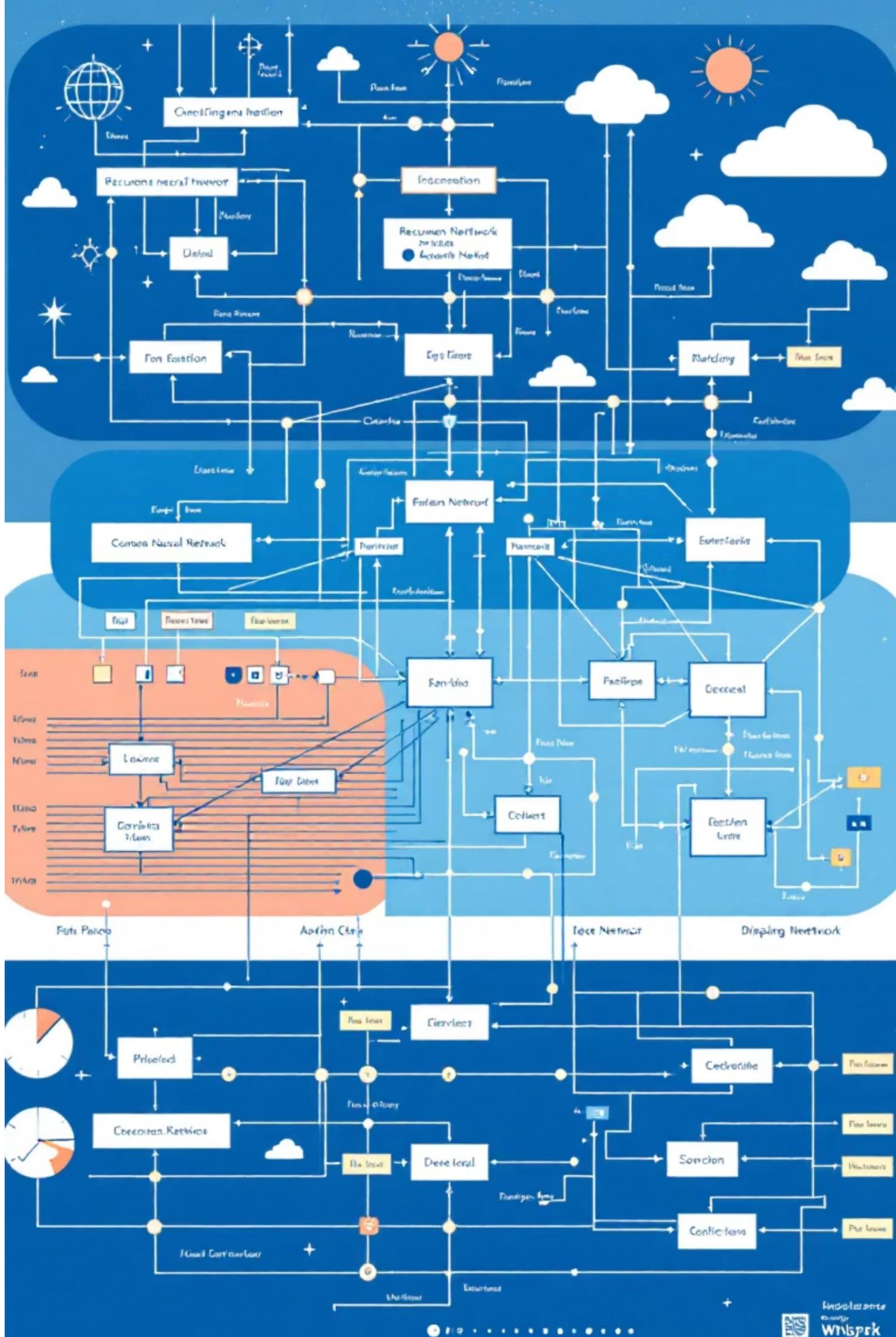
SimpleRNN with 64 units, 128-dimensional embedding, 30% dropout, and 605,313 parameters. Fast training of 7.40 seconds but premature convergence at epoch 3 with 67.50% validation accuracy.

LSTM Network

Bidirectional LSTM with 128 units per direction, 50% dropout, 20% recurrent dropout, and 856,321 parameters. Training of 73.02 seconds with better performance at epoch 3 and 78.33% validation accuracy.

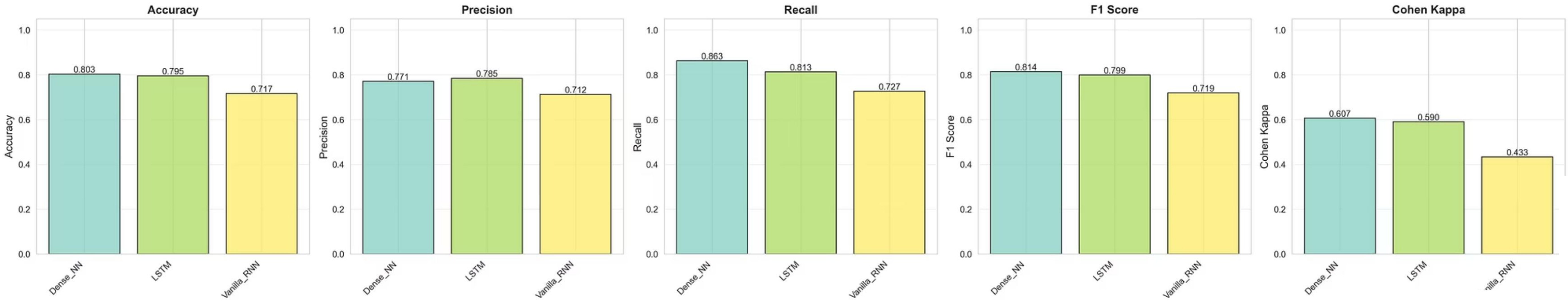




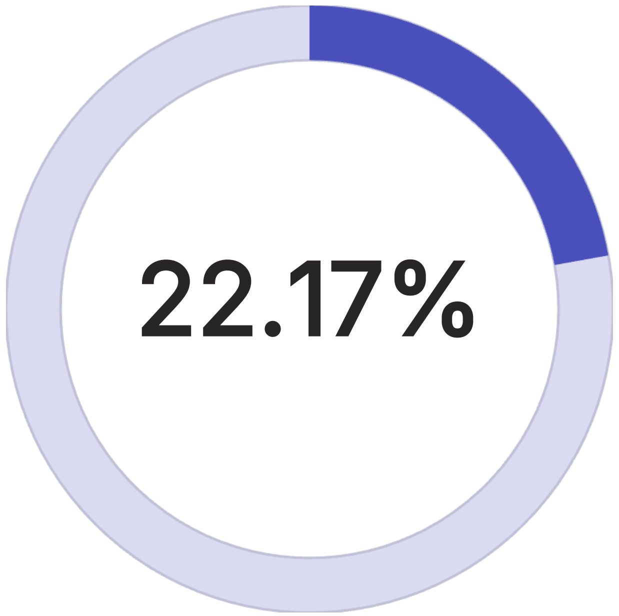


Performance Evaluation

Model	Accuracy	Precision	Recall	F1-Score	Kappa
Baseline (Constant)	0.5000	0.5000	1.0000	0.6667	0.0000
Baseline (Stratified)	0.5200	0.5208	0.5000	0.5102	0.0400
Dense NN	**0.8033**	0.7708	**0.8633**	**0.8145**	**0.6067**
Vanilla RNN	0.7167	0.7124	0.7267	0.7195	0.4333
LSTM	0.7950	0.7846	0.8133	0.7987	0.5900

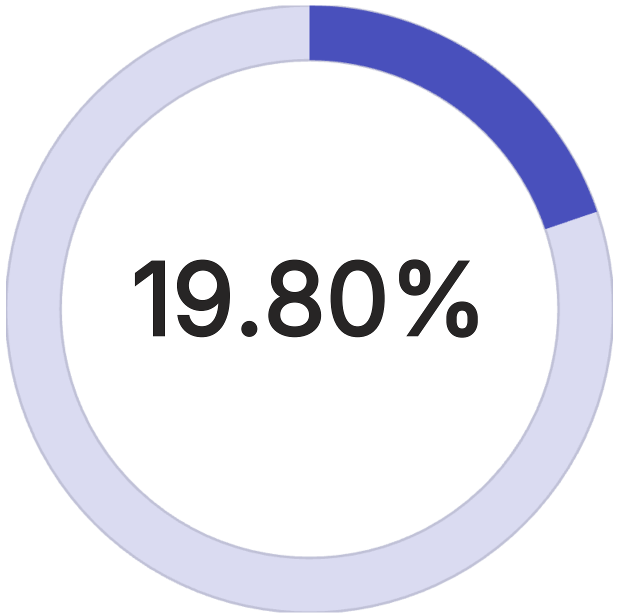


Comparative Analysis and Key Findings



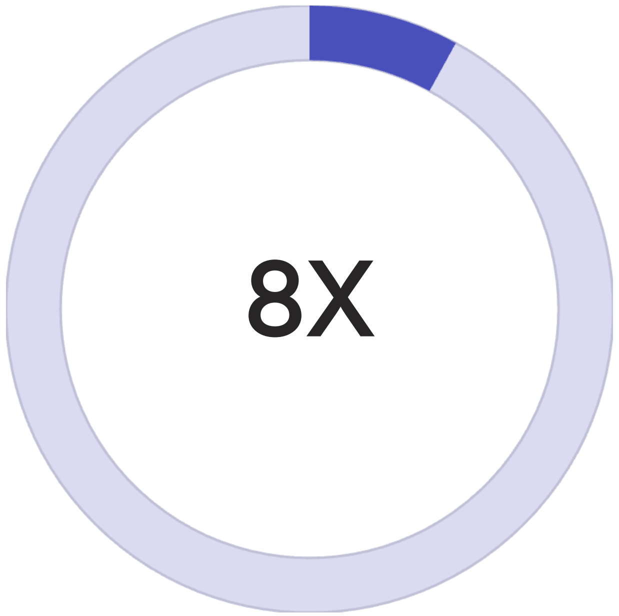
Dense NN Improvement

Exceeded baseline in F1-score



LSTM Improvement

Above the target threshold of 15%



Dense NN Speed

Faster than LSTM with superior performance

The Dense NN achieved the best performance with 80.33% accuracy and 81.45% F1-score, surpassing even the more sophisticated LSTM architecture. This remarkable finding demonstrates that simpler architectures can outperform complex models when well-optimized and aligned with data characteristics.

The success of the Dense NN is attributed to short sequences (6.20 average words), a three-layer architecture with effective global pooling, optimal hyperparameter configuration, and simplicity that matches the task's complexity.

Conclusions

Conclusions

- The Dense Network emerged as the best model with 80.33% accuracy and a 22.17% improvement over baseline
- Simple architectures can outperform complex models when optimized and aligned with data characteristics
- Extensive preprocessing was critical, reducing text by 47.6% while improving semantic signal
- Hyperparameter optimization using GridSearchCV significantly improved performance