Sentiment Classification with Neural Networks

Saisrijith Reddy Maramreddy

December 2025

1 Background on Sentiment Analysis and Neural Text Classification

Sentiment analysis aims to interpret opinions and emotional tone in text, a task made difficult on Twitter due to informal and highly variable language. CNNs address these challenges by learning embeddings and hierarchical n-gram features directly from raw text, enabling them to capture sentiment cues such as negation and contrast. Their ability to scale and model local context makes them well suited for large sentiment datasets like Sentiment140.

1.1 Why Model Architecture and Hyperparameters Matter

A CNN's performance is shaped by its architecture—kernel sizes, filter counts, and depth determine how much linguistic structure the model can capture. Training hyperparameters, especially learning rate, strongly influence convergence stability and generalization. By varying these factors, this project evaluates how model capacity and optimization choices affect accuracy and efficiency in large-scale sentiment classification.

2 Data Description and Objective

The Sentiment140 dataset, consisting of 1.6 million tweets labeled as either positive or negative, provides a realistic and challenging benchmark for neural text classification. Tweets contain informal grammar, emojis, abbreviations, and noise—mirroring real-world sentiment expression. To ensure balanced learning, the dataset was equalized across classes and split into 80% training, 10% validation, and 10% testing.

2.1 Objective

The main objective of this project is to determine how architectural modifications and learning rate choices affect the performance of a CNN-based sentiment classifier. Specifically, the study evaluates whether deeper and wider convolutional architectures provide meaningful accuracy gains relative to their increased computational cost.

3 Model Architectures

3.1 Baseline CNN

The baseline architecture follows a standard convolutional design for text classification. It consists of:

- an embedding layer that maps each token to a 100-dimensional vector,
- a single 1D convolutional layer with 128 filters and a kernel size of 5, used to capture local n-gram features,
- a global max pooling layer that extracts the most salient activation across the sequence,
- a fully connected output layer with sigmoid activation for binary sentiment prediction.

This architecture serves as a lightweight benchmark, offering fast training and a clear reference point for evaluating the impact of deeper and wider convolutional structures.

3.2 Modified CNN

The modified architecture expands the model's capacity by deepening the convolutional stack and widening its receptive field. Specifically:

- a second Conv1D layer (256 filters) is added on top of the baseline's 128-filter layer, enabling hierarchical feature extraction—the first layer captures short n-grams, while the second composes them into higher-level sentiment patterns,

- a larger kernel size (7 instead of 5) increases the receptive field, allowing the model to capture longer phrases and broader contextual cues,
- the increased filter count provides richer feature diversity for handling informal sentiment markers common in tweets.

These changes help the model detect sentiment spread across multi-word expressions or contrastive statements, though they also increase parameter count, computation time, and sensitivity to learning rate settings.

4 Training Setup

- Optimizer: SGD, batch size = 32
- Learning rates tested: {0.1, 0.01, 0.001}
- Maximum epochs: 20 with early stopping (patience = 3)
- Training metrics: training loss, validation loss, validation error rate, epoch time ‘
- Evaluation metrics: accuracy, false positive rate (FPR), false negative rate (FNR), total error rate

Across all runs, training behavior depended strongly on the choice of learning rate: a very small rate (0.001) produced stable but slow convergence; a moderate rate (0.01) converged faster but showed mild sensitivity to validation fluctuations; and a high rate (0.1) provided the quickest improvement and consistently achieved the lowest validation and test error without divergence.

5 Results

5.1 Performance Comparison

Table 1: Training, validation, and test accuracy across learning rates for both CNN architectures.

Model	Parameters	Learning Rate	Train (%)	Val (%)	Test (%)
Baseline CNN	1,064,357	0.001	79.67	76.89	76.81
Modified CNN	1,507,749	0.001	81.68	76.22	76.19
Baseline CNN	1,064,357	0.010	80.97	77.55	77.62
Modified CNN	1,507,749	0.010	85.22	77.00	77.14
Baseline CNN	1,064,357	0.100	81.80	**78.09**	**78.14**
Modified CNN	1,507,749	0.100	83.25	77.60	77.59

The modified network contains roughly 40% more parameters due to the additional convolutional layer and expanded filter bank, which also increased training time. Across all learning rates tested, this added complexity did not improve performance. In every setting, the baseline CNN outperformed the modified architecture on both validation and test sets. These results indicate that the deeper, wider model offered no measurable benefit for this sentiment classification task, and the simpler baseline network generalized more effectively despite its lower capacity.

6 Discussion

Overall, the results show that the baseline CNN consistently performs as well as—or slightly better than—the modified architecture across learning rates. At the optimal learning rate of 0.1, the baseline model achieved the highest test accuracy (78.52%), despite having substantially fewer parameters. These findings indicate that the added depth and filter capacity in the modified CNN did not translate into stronger performance and may introduce mild overfitting at certain learning rates. In this setting, model simplicity offered more reliable generalization than increased architectural complexity.

Appendix

All source code, model training, and analysis notebooks used in this project are available at:
`Sentiment Classification with Neural Networks Notebook`