Sentiment Analysis on Review Datasets using Deep Learning

Final Project Report

November 21, 2025

Abstract

This report presents a comparative analysis of machine learning models applied to sentiment classification of user reviews. Using a dataset comprising Amazon, IMDb, and Yelp reviews, we implemented preprocessing pipelines and trained three distinct neural network architectures: A Dense Neural Network (DNN), a Vanilla Recurrent Neural Network (RNN), and a Long Short-Term Memory (LSTM) network. The Dense NN utilizing TF-IDF features achieved the highest accuracy of 81.87%, outperforming the sequential models on this specific dataset.

1 Introduction

Sentiment analysis is a critical task in Natural Language Processing (NLP) used to determine the emotional tone of a text. This project aims to classify sentences as either Positive (1) or Negative (0). The ability to automate this classification allows organizations to process vast amounts of feedback efficiently.

2 Objectives

- To implement a robust text preprocessing pipeline using NLTK.
- To implement and tune three architectures: Dense NN, RNN, and LSTM.
- To establish a baseline using a Dummy Classifier.
- To evaluate and compare models using Accuracy, F1-Score, and Cohen's Kappa.

3 Dataset Description

The dataset (Kotzias et al., KDD 2015) contains sentences labeled with sentiment extracted from:

- **IMDb:** Movie reviews.
- **Amazon:** Product reviews.
- **Yelp:** Restaurant reviews.

After cleaning corrupt data and duplicates, the dataset consists of **2,726 instances** with a balanced class distribution between positive and negative labels.

4 Preprocessing Steps

Preprocessing was performed using Python's NLTK library. The pipeline included:

1. **Normalization:** Lowercasing all text.
2. **Cleaning:** Removing punctuation and alphanumerics using RegEx.
3. **Tokenization:** Splitting strings into tokens.
4. **Lemmatization:** Using POS tags to lemmatize words (reducing dimensionality).
5. **Stopword Removal:** Removing non-informative words.
6. **Splitting:** Data was split into **80% Training** (2,180 samples) and **20% Testing** (546 samples) using stratification.

5 Models and Architecture

5.1 Baseline (Dummy Classifier)

A stratified dummy classifier was used to determine the random chance performance floor, achieving an accuracy of $\approx 52\%$.

5.2 Dense Neural Network (DNN)

- **Input:** TF-IDF Vectors (Max features: 5,000).
- **Structure:** Two Dense layers (64, 32 units) with ReLU activation and Dropout (0.3). Output layer with Sigmoid activation.
- **Best Optimizer:** RMSprop.

5.3 Vanilla RNN

- **Input:** Padded Sequences (Length 11).
- **Structure:** Embedding Layer (50 dim) $\rightarrow$ SpatialDropout $\rightarrow$ SimpleRNN (32 units) $\rightarrow$ Dense Output.
- **Regularization:** High dropout (0.6) to prevent overfitting.

5.4 LSTM

- **Input:** Padded Sequences (Length 14).
- **Structure:** Embedding Layer (50 dim) $\rightarrow$ SpatialDropout $\rightarrow$ LSTM (64 units) $\rightarrow$ Dense Output with L2 Regularization.
- **Best Optimizer:** Adam.

6 Evaluation Metrics and Comparative Analysis

Hyperparameter tuning was conducted using GridSearchCV. The final evaluation on the test set yielded the following results:

Table 1: Performance Comparison of Models

Model	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Kappa**
Baseline	0.5220	0.52	0.50	0.51	0.04
Vanilla RNN	0.7418	0.73	0.76	0.75	0.48
LSTM	0.7875	0.76	**0.84**	0.80	0.57
DNN	**0.8187**	**0.84**	0.78	**0.82**	**0.64**

The Dense Neural Network (TF-IDF) outperformed the sequence models. This indicates that for short sentences, specific keywords (handled well by TF-IDF) are stronger predictors than word order. The LSTM performed better than the RNN, showing its capability to handle dependencies better, though both suffered from overfitting due to the small dataset size.

7 Feature Importance

Analysis of the Dense Network weights identified the following top predictors:

- **Positive:** Love, Quality, Service, Great.
- **Negative:** Cheap, Tired, Poor, Loud.

8 Conclusions and Future Work

The project demonstrated that simple architectures like DNNs with TF-IDF can outperform complex sequence models on small, short-text datasets. While LSTM showed promise with a high recall, it requires more data to generalize effectively.

Future Work:

1. Implementation of pre-trained embeddings (GloVe/Word2Vec).
2. Utilization of Transformer-based models (BERT).
3. Data augmentation to improve RNN/LSTM generalization.