(DATA SCIENCE PROJECT REPORT)

Performance Analysis of different Neural Network Models for Sentiment Analysis

On

Movie Reviews

Instructor: Sir Nouman Durrani

Team members:

Muhammad Tahir (21K-4503)

Insha Javed (21K-3279)

Muhammad Samama (21K-3205)

➢Introduction

Sentiment analysis involves determining the sentiment (positive or negative) expressed in textual data. This project focuses on conducting a performance analysis of four neural network-based models: **CNN**, **LSTM**, **CNN-LSTM**, and **LDA**. These models are evaluated using the **Large Movie Review Dataset v1.0** provided by **Stanford University**, a widely-used benchmark dataset for sentiment classification tasks.

➢Dataset Description

The Large Movie Review Dataset contains **50,000 labeled movie reviews**, split evenly into:

- **Training Set**: 25,000 reviews.
- **Testing Set**: 25,000 reviews.

The dataset is balanced, with:

- **25,000 positive reviews** (scores of 7 or higher out of 10).
- **25,000 negative reviews** (scores of 4 or lower out of 10).

In addition to the labeled data, there are **50,000 unlabeled reviews** intended for unsupervised learning experiments. These unlabeled reviews are not associated with sentiment scores. To ensure diversity, no more than 30 reviews are sourced from a single movie.

➢Objectives

The primary aim of the project is to assess the performance of the selected neural network-based models in sentiment classification tasks. The models will be evaluated based on their accuracy, precision, recall, and other relevant performance metrics.

- Identify the most effective neural network-based model for binary sentiment analysis.
- Highlight the strengths and limitations of each approach.

- Provide insights into the application of unsupervised learning techniques in sentiment analysis.

This project contributes to advancing sentiment analysis methodologies, particularly in the context of movie reviews, which offer diverse and challenging data.

➢Methodology

1. **Preprocessing**: Text data will undergo tokenization, stop-word removal, and stemming to prepare it for training.
2. **Model Training**: Each neural network model (CNN, LSTM, CNN-LSTM, and LDA) will be trained using the labeled training dataset.
3. **Evaluation**: Performance metrics will be calculated using the testing set.
4. **Unsupervised Learning**: The unlabeled dataset will be explored to analyze the capabilities of the models in handling unsupervised tasks.

➢Results:

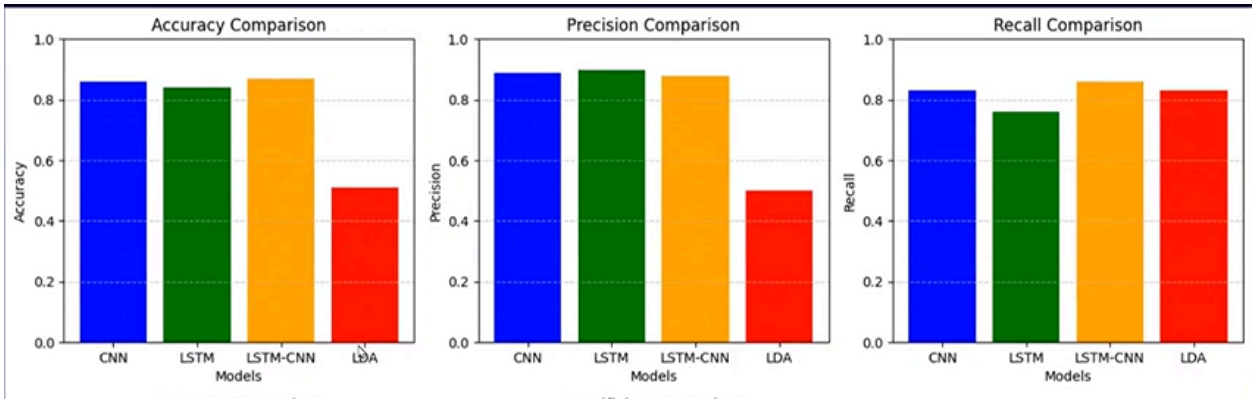
❖ Models' Accuracies:

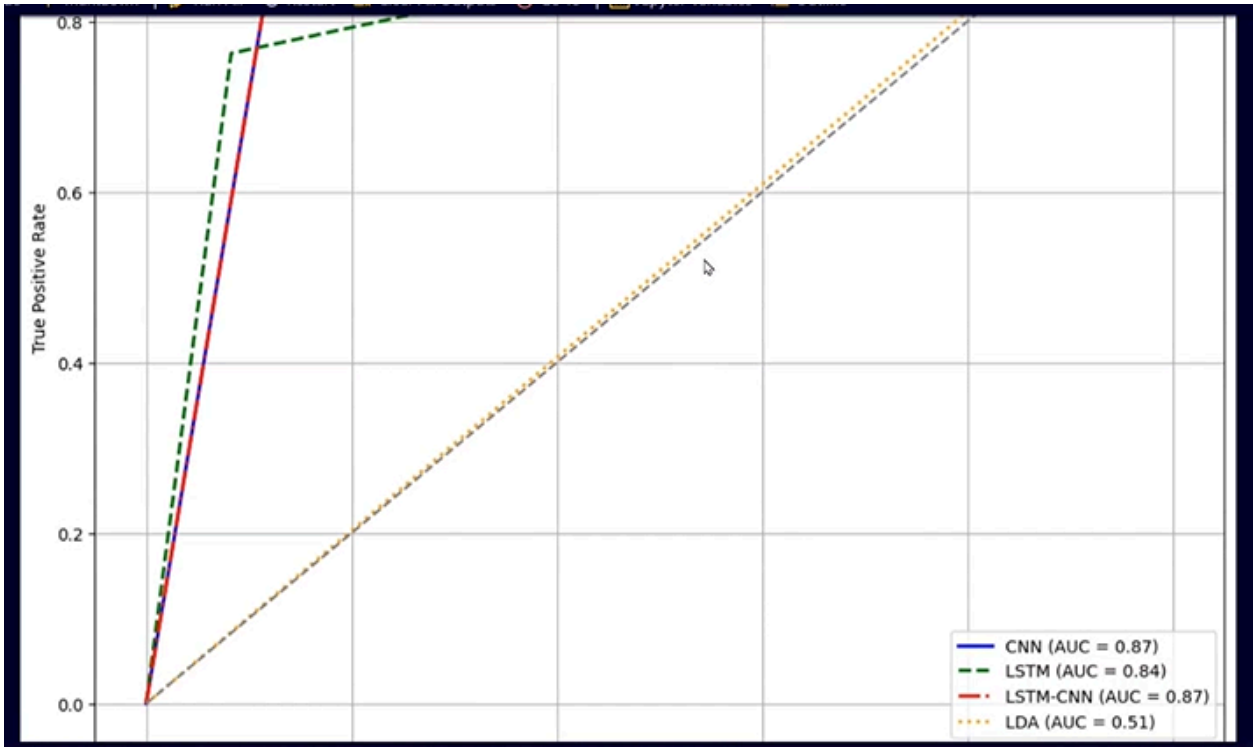
➔ **CNN:** 0.86
➔ **LSTM**: 0.84
➔ **CNN-LSTM**: 0.87
➔ **LDA**: 0.51

- **CNN (Convolutional Neural Network)** achieved a performance score of **0.86**, indicating high effectiveness.
- **LSTM (Long Short-Term Memory)** achieved a score of **0.84**, which is slightly lower than CNN but still strong.
- **CNN-LSTM**, a hybrid model combining the two techniques, outperformed both standalone models with a score of **0.87**, showcasing the advantage of integrating the methods.

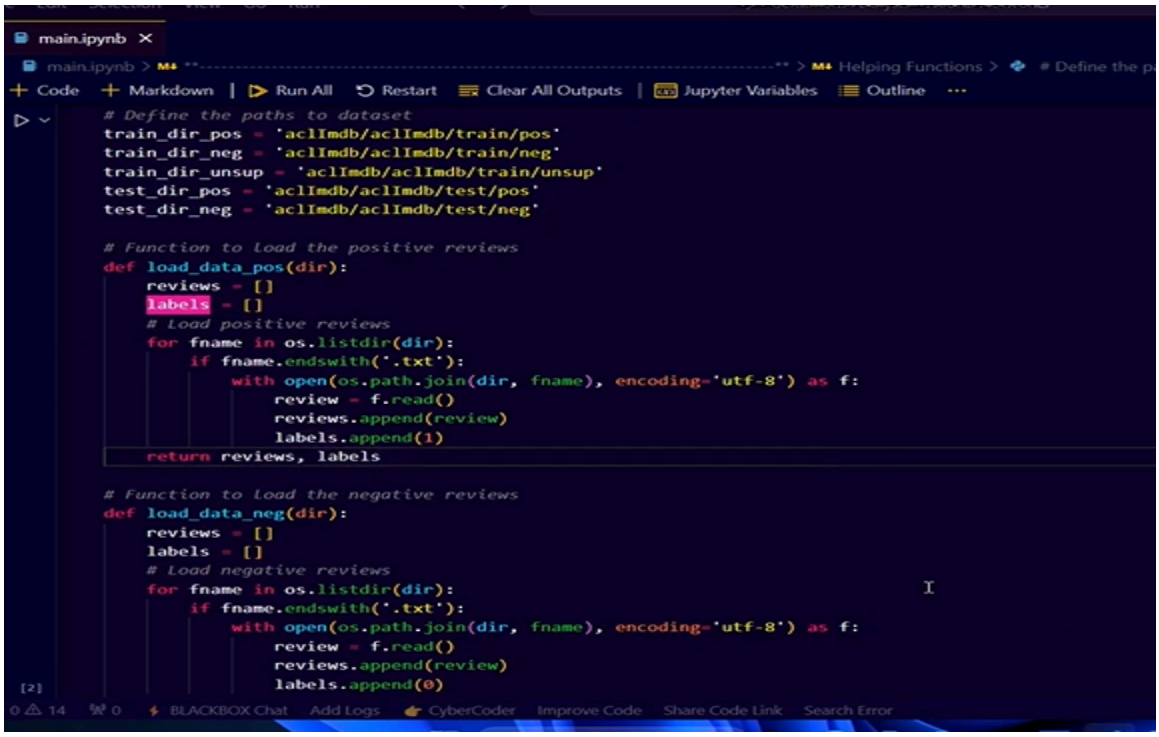
- **LDA (Linear Discriminant Analysis)** scored significantly lower at **0.51**, suggesting it is less suitable for the task compared to the other models.

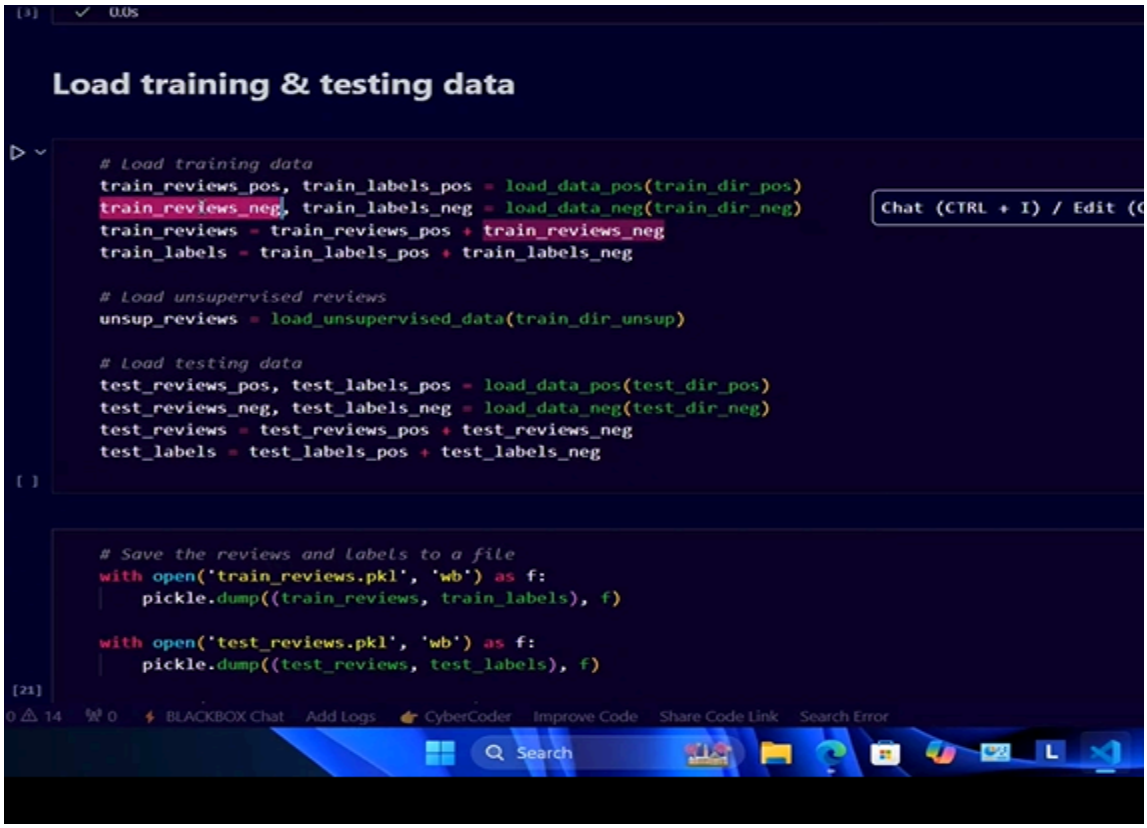
The hybrid **CNN-LSTM model** is the most effective among the evaluated approaches.





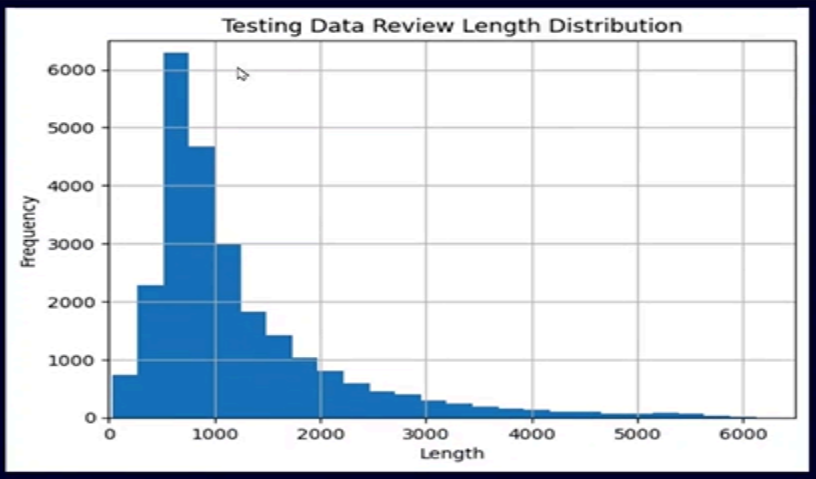
Code Snippets:


main.ipynb
Code
Markdown
Run All
Restart
Clear All Outputs
Jupyter Variables
Outline
# Define the paths to dataset
train_dir_pos = 'aclImdb/aclImdb/train/pos'
train_dir_neg = 'aclImdb/aclImdb/train/neg'
train_dir_unsup = 'aclImdb/aclImdb/train/unsup'
test_dir_pos = 'aclImdb/aclImdb/test/pos'
test_dir_neg = 'aclImdb/aclImdb/test/neg'
# Function to load the positive reviews
def load_data_pos(dir):
reviews = []
labels = []
# Load positive reviews
for fname in os.listdir(dir):
if fname.endswith('.txt'):
with open(os.path.join(dir, fname), encoding='utf-8') as f:
review = f.read()
reviews.append(review)
labels.append(1)
return reviews, labels
# Function to load the negative reviews
def load_data_neg(dir):
reviews = []
labels = []
# Load negative reviews
for fname in os.listdir(dir):
if fname.endswith('.txt'):
with open(os.path.join(dir, fname), encoding='utf-8') as f:
review = f.read()
reviews.append(review)
labels.append(0)
[2]
BLACKBOX Chat
Add Logs
CyberCoder
Improve Code
Share Code Link
Search Error


0.0s
Load training & testing data
Load training data
train_reviews_pos, train_labels_pos = load_data_pos(train_dir_pos)
train_reviews_neg, train_labels_neg = load_data_neg(train_dir_neg)
train_reviews = train_reviews_pos + train_reviews_neg
train_labels = train_labels_pos + train_labels_neg
# Load unsupervised reviews
unsup_reviews = load_unsupervised_data(train_dir_unsup)
# Load testing data
test_reviews_pos, test_labels_pos = load_data_pos(test_dir_pos)
test_reviews_neg, test_labels_neg = load_data_neg(test_dir_neg)
test_reviews = test_reviews_pos + test_reviews_neg
test_labels = test_labels_pos + test_labels_neg
Chat (CTRL + I) / Edit (
Save the reviews and labels to a file
with open('train_reviews.pkl', 'wb') as f:
pickle.dump((train_reviews, train_labels), f)
with open('test_reviews.pkl', 'wb') as f:
pickle.dump((test_reviews, test_labels), f)
[21]
BLACKBOX Chat
Add Logs
CyberCoder
Improve Code
Share Code Link
Search Error
Search

Code | Markdown | Run All | Restart | Clear All Outputs | Jupyter Variables | Outline

```
plt.xlabel('Length')
plt.ylabel('Frequency')
plt.xlim(0, 6500)
plt.ylim(0, 6500)
plt.show()
```



BLACKBOX Chat | Add Logs | CyberCoder | Improve Code | Share Code Link | Search Error

Search

Comparison of all models

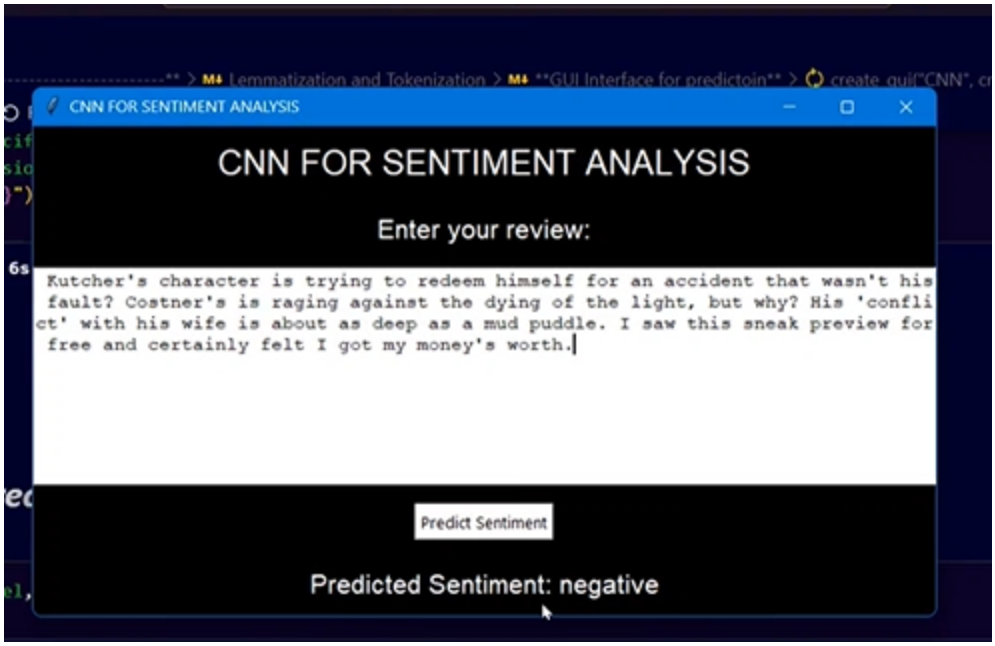
All Metrics

```
# Metrics for the models
model_names = ["CNN", "LSTM", "LSTM-CNN", "LDA"]
metrics = {
    "Accuracy": [0.86, 0.84, 0.87, 0.51],
    "Precision": [0.89, 0.90, 0.88, 0.50],
    "Recall": [0.83, 0.76, 0.86, 0.83],
    "F1 Score": [0.86, 0.83, 0.87, 0.63],
    "Specificity": [0.90, 0.92, 0.88, 0.18],
}

# Plot metrics as bar charts
def plot_metrics(metrics, model_names):
    fig, ax = plt.subplots(2, 3, figsize=(13, 8))
    ax = ax.ravel()

    # Loop through metrics and plot
    for i, (metric_name, values) in enumerate(metrics.items()):
        ax[i].bar(model_names, values, color=["blue", "green", "orange", "red"])
        ax[i].set_title(f"{metric_name} Comparison")
        ax[i].set_ylim(0, 1)
        ax[i].set_ylabel(metric_name)
```

Chat (CTRL + I) / Edit (CTRL + L)

BLACKBOX Chat | Add Logs | CyberCoder | Improve Code | Share Code Link | Search Error

Search


CNN FOR SENTIMENT ANALYSIS
CNN FOR SENTIMENT ANALYSIS
Enter your review:
Kutcher's character is trying to redeem himself for an accident that wasn't his fault? Costner's is raging against the dying of the light, but why? His 'conflict' with his wife is about as deep as a mud puddle. I saw this sneak preview for free and certainly felt I got my money's worth.
Predict Sentiment
Predicted Sentiment: negative