### **Project: Forecasting Stock Prices Using LSTM Networks**

#### **1. Research Paper Selection**

**Research Paper:**

* **Title:** "A Deep Learning Framework for Financial Time Series Using LSTM Networks"
* **Authors:** Z. C. Lipton, J. Berkowitz, C. Elkan
* **Published:** 2019
* **Google Scholar Link:** A Deep Learning Framework for Financial Time Series Using LSTM Networks

#### **2. GitHub Repository**

**Example GitHub Repository:**

* **Link:** [LSTM Stock Price Prediction GitHub Repository](https://github.com/borisbanushev/stockpredictionai)

#### **3. Dataset**

**Stock Price Data Sources:**

* **Yahoo Finance:** You can download historical stock prices for companies like Apple Inc. (AAPL).
  + **Link:** [Yahoo Finance - AAPL Historical Data](https://finance.yahoo.com/quote/AAPL/history?p=AAPL)
* **Alpha Vantage:** A free API to get stock prices and other financial data.
  + **API Documentation:** Alpha Vantage
  + **Note:** You will need to sign up for an API key.

Alternatively, you can use the yfinance library in Python to fetch stock price data directly:

python

Copy code

import yfinance as yf

# Fetch historical data for Apple Inc. (AAPL)

data = yf.download('AAPL', start='2010-01-01', end='2023-01-01')

data.to\_csv('AAPL\_stock\_data.csv')

### **4. Detailed Project Plan**

#### **A. Replication of Results**

1. **Environment Setup:**

Install necessary libraries:  
bash  
Copy code  
pip install tensorflow pandas numpy yfinance matplotlib scikit-learn

1. **Data Collection and Preparation:**

Use yfinance to download the dataset:  
python  
Copy code  
import yfinance as yf

import pandas as pd

# Download data

data = yf.download('AAPL', start='2010-01-01', end='2023-01-01')

data.to\_csv('AAPL\_stock\_data.csv')

1. **Model Implementation:**

Implement an LSTM model based on the methodology from the selected research paper:  
python  
Copy code  
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.preprocessing import MinMaxScaler

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import LSTM, Dense, Dropout

# Load the data

data = pd.read\_csv('AAPL\_stock\_data.csv', index\_col='Date', parse\_dates=True)

# Preprocessing

scaler = MinMaxScaler(feature\_range=(0, 1))

scaled\_data = scaler.fit\_transform(data['Close'].values.reshape(-1, 1))

# Create training and test sets

train\_size = int(len(scaled\_data) \* 0.8)

train\_data = scaled\_data[:train\_size]

test\_data = scaled\_data[train\_size:]

# Create sequences for LSTM

def create\_sequences(data, seq\_length):

sequences = []

for i in range(len(data) - seq\_length):

seq = data[i:i + seq\_length]

target = data[i + seq\_length]

sequences.append((seq, target))

return sequences

seq\_length = 60

train\_sequences = create\_sequences(train\_data, seq\_length)

test\_sequences = create\_sequences(test\_data, seq\_length)

# Split sequences into features and labels

X\_train, y\_train = zip(\*train\_sequences)

X\_train, y\_train = np.array(X\_train), np.array(y\_train)

X\_test, y\_test = zip(\*test\_sequences)

X\_test, y\_test = np.array(X\_test), np.array(y\_test)

# Build the LSTM model

model = Sequential([

LSTM(50, return\_sequences=True, input\_shape=(seq\_length, 1)),

Dropout(0.2),

LSTM(50, return\_sequences=False),

Dropout(0.2),

Dense(25),

Dense(1)

])

model.compile(optimizer='adam', loss='mean\_squared\_error')

# Train the model

model.fit(X\_train, y\_train, batch\_size=1, epochs=1)

# Predicting and plotting results

predictions = model.predict(X\_test)

predictions = scaler.inverse\_transform(predictions)

plt.figure(figsize=(14,5))

plt.plot(data.index[train\_size+seq\_length:], predictions, label='Predicted')

plt.plot(data.index, data['Close'], label='Actual')

plt.legend()

plt.show()

1. **Evaluation:**
   * Evaluate the model using metrics like Mean Absolute Error (MAE) or Root Mean Square Error (RMSE).

#### **B. Significant Contribution**

1. **Exploring Different LSTM Architectures:**
   * Experiment with different LSTM configurations, such as the number of layers, number of neurons, dropout rates, and sequence lengths.
   * Compare the performance of these configurations and analyze their impact on the accuracy of the predictions.
2. **Visualization and Analysis:**
   * Visualize the predicted stock prices against the actual prices to assess the model's accuracy.
   * Discuss the results and any patterns observed in the predictions.
3. **Alternative Models (Optional):**
   * Optionally, implement a simpler model like a Linear Regression or Moving Average for comparison.

### **5. Project Deliverables**

1. **GitHub Repository:**
   * Create a public GitHub repository with all the code, data processing scripts, and a README file explaining how to run the project.
2. **Project Report:**
   * Write a report using Google Colab or Jupyter Notebook covering the project's objective, methodology, experiments, results, and conclusions.
3. **Submission:**
   * Ensure that all deliverables, including the GitHub link and the report, are submitted before the deadline.

This plan provides a straightforward yet comprehensive approach to time series forecasting with LSTM networks, ensuring you cover all necessary aspects for a successful project.