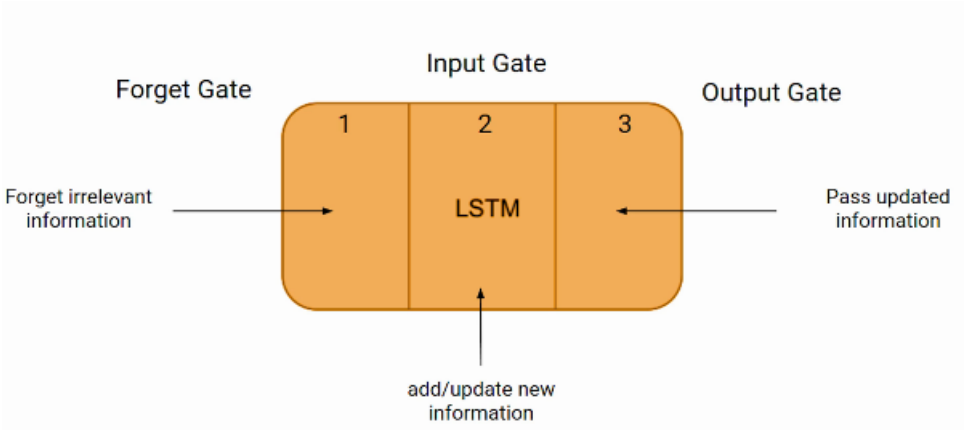
Integrating Sentiment Analysis and LSTM Networks for Predicting Stock Prices



Brayden K. Mi

bkm9592@nyu.edu

New York University

Courant Institute of Mathematical Sciences

August 8, 2024

1 Introduction

Due to the dynamic nature of the stock market, stock price prediction is a difficult task. Traditional methods often fall short in capturing the complex patterns present in time-series data, and many machine learning methods overlook crucial societal sentiment data in training their models.

In this proposal and proof of concept, I present an Long Short-Term Memory (LSTM) based model that integrates sentiment analysis from news data and historical stock prices to predict next day closing price. The model utilizes a 30 day sliding window, which is updated daily to ensure the most recent information is always incoporated.

Preliminary backtesting was conducted on two stocks with different volatility profiles: NVIDIA (NVDA), a stable stock with significant news coverage, and INNOVATE Corp. (VATE), a highly volatile stock with limited news coverage. The results demonstrate that the model performs well on stable stocks, achieving a positive return on investment (ROI) with unsupervised trading over a period of one to six months. However, when it comes to volatile stocks, it performs poorly, often resulting in a negative ROI.

2 LSTM Background

The base architecture of this price prediction model is the Long Short-Term Memory (LSTM) network, which is well-known for its ability to manage long-term dependencies and mitigate the vanishing gradient problem through its additive update learning strategy.
The LSTM network is structured as follows:

- **Input Vector** X_t: Comprises the daily sentiment score and the historical closing price of the stock for that day which forms the input sequence fed into the LSTM network.
- **Memory Cell** C_t: Stores the information across different time steps. The LSTM cell updates C_t at each time step based on the previous memory cell state C_{t-1}, the current input vector X_t, and the network's gates. The update mechanism is as follows:

$$f_t = \sigma(W_f \cdot [h_{t-1}, X_t] + b_f) \quad \text{(forget gate)} \tag{1}$$
$$i_t = \sigma(W_i \cdot [h_{t-1}, X_t] + b_i) \quad \text{(input gate)} \tag{2}$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, X_t] + b_C) \quad \text{(candidate cell state)} \tag{3}$$
$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \quad \text{(updated cell state)} \tag{4}$$

- **Output** h_t: Computed using the updated memory cell C_t and the current input vector X_t. The output serves as the hidden state, capturing the relevant information from both the current input and the previous states:

$$o_t = \sigma(W_o \cdot [h_{t-1}, X_t] + b_o) \quad \text{(output gate)} \tag{5}$$
$$h_t = o_t \cdot \tanh(C_t) \quad \text{(hidden state)} \tag{6}$$

- **Dense Layers**: These layers are responsible for mapping the LSTM outputs to the final stock price prediction by learning complex relationships between the input features and the target variable.
- **Output Layer**: Consists of a single neuron, which provides the predicted stock price for the next day. This output is a continuous value, suitable for the regression task at hand.

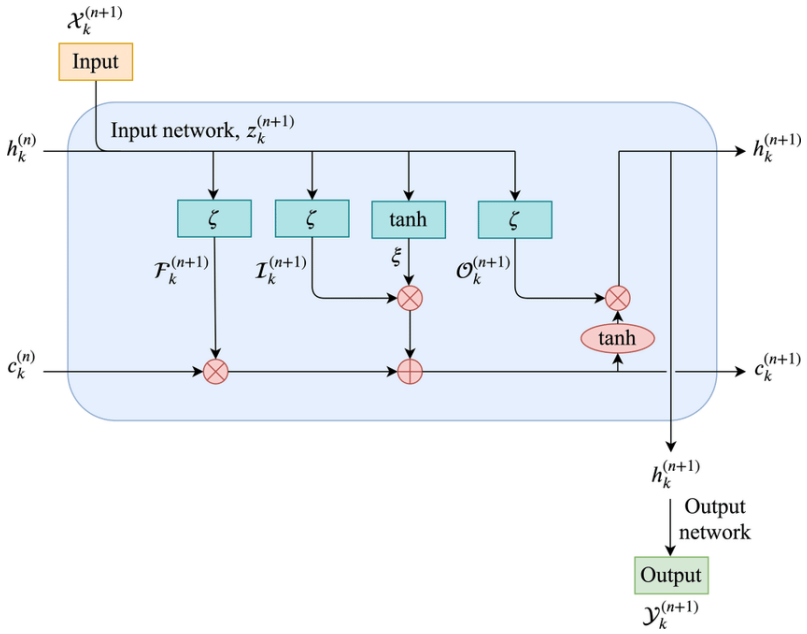


Figure 1: Architecture of a basic LSTM model from Traian Iliescu

3 Model Workflow

The model's workflow is composed of the following steps:

3.1 Data Collection and Preprocessing

- **News Data Collection and Preprocessing**:
  - **Collection**: Relevant news articles and daily digests related to the target stock are collected using a web scraping tool designed for this model.
  - **Sentiment Analysis**: The collected articles undergo sentiment analysis to extract sentiment scores. This project uses the Valence Aware Dictionary and Sentiment Reasoner (VADER) sentiment analysis model with a finance specific modified weighted lexicon.
- **Retrieve Historical Prices**:
  - **Data Source**: Historical closing prices for the stock are retrieved from Yahoo's financial database using the yfinance library in python.
  - **Normalization**: The prices are normalized (using min-max scaling) to match the scale of the sentiment scores as required.
- **Feature Engineering**:
  - **Feature Vector**: Each day within the 30-day window is represented as a feature vector containing the normalized sentiment score and the normalized historical closing price.
  - **Sliding Window**: A 30-day sliding window is used to create sequences of feature vectors, which serve as input for the LSTM network.

3.2 Training

- **Loss Function**:
  - **Mean Squared Error (MSE)**: The loss function used is Mean Squared Error (MSE), which measures the average squared difference between predicted and actual stock prices.

- **Equation**:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

where y_i is the actual price, $\hat{y}_i$ is the predicted price, and N is the number of predictions.

- **Backpropagation Through Time (BPTT)**:
  - BPTT is used to update the LSTM network's weights by propagating errors backward through time and adjusting the parameters to minimize the loss.

3.3 Prediction/Inference

- **Next Day Price Prediction**:
  - The trained LSTM model takes the current 30-day input sequence and predicts the stock price for the next day.
  - The output is a continuous value representing the forecasted price.

4 Future Work

While the current model shows promising results, several areas need improvement to enhance its overall performance and applicability to real-time trading scenarios. Key issues identified during preliminary backtesting include the models poor performance with volatile stocks, inadequate handling of sparse news coverage, and slow training and processing times that hinder the real-time applicability. To address these issues, the next steps for this model are as proposed:

- **Optimize News Scraping**:
  - **Enhanced Data Collection**: Improve the efficiency and breadth of the news scraping process to ensure more comprehensive and timely data collection. This includes establishing a more efficient scraping and sentiment analysis pipeline, along with expanding the range of news sources
- **Implement Twitter/X Scraper**:
  - **Social Media Integration**: Develop and integrate a scraper for Twitter/X to capture real time market sentiment and expand signal sources to posts from influential figures.
- **Include Technical Signals**:
  - **Feature Engineering**: Incorporate technical indicators, namely Moving Average Convergence Divergence (MACD) and Bollinger Bands (BB) as additional input features to the LSTM model and assess their impact on prediction accuracy.
  - **Technical Indicators vs. Sentiment Analysis**: Evaluate the effectiveness of technical trading signals compared to sentiment analysis to determine the most predictive features for the model.
- **Optimize Real-Time Performance**:
  - **Speed and Efficiency**: Due to the lack of access to efficient hardware and high performance computing clusters, the current model trains too slowly to keep pace with the market. Optimizing the model architecture and training process will reduce computation time so forward testing can be conducted. Additionally, the model could then be scaled to predict stock price in an hourly window.
  - **Incremental Training**: Implement more efficient incremental or online learning methods that allow the model to update with new data in real time without requiring full retraining.