INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD)
© 2026 IJNRD | Volume 11, Issue 4, April 2026 | ISSN: 2456-4184 | IJNRD.ORG

AI-Driven Decision Support Systems for Financial Investments

1st P.Nataraj
Assistant Professor
Dept of Computer Science and Engineering
AVS College of Technology
Salem, India
natarajbe.cse@gmail.com

2nd R.Ranjithkumar
Student
M.E Computer Science and Engineering
AVS College of Technology
Salem, India
ranjith779@gmail.com

***Abstract*— The financial market's inherent volatility and complexity present a significant challenge for effective investment decision-making. Traditional methods often fall short in capturing the non-linear patterns and the impact of unstructured data, such as news and social media sentiment. This Phase-1 project report proposes the design and development of a novel hybrid Artificial Intelligence (AI) framework to enhance stock market trading and portfolio management.**

The proposed system integrates three core AI domains: 1) Financial Sentiment Analysis, utilizing a Transformer-based model (BERT) to quantify market mood from news and social media text; 2) Stock Price Prediction, employing a multi-input Long Short-Term Memory (LSTM) network that leverages both historical price data and the derived sentiment scores for more accurate trend forecasting; and 3) Multi-objective Portfolio Optimization, implementing the Non-dominated Sorting Genetic Algorithm-II (NSGA-II) to construct optimal portfolios that balance the dual objectives of maximizing returns and minimizing risk, as measured by Conditional Value-at-Risk (CVaR).

This report details the comprehensive literature survey conducted, which established the foundation and identified the research gap this project aims to fill. The system's overall architecture and detailed module designs are presented. Furthermore, the report specifies the complete hardware and software requirements for development and provides a rigorous analysis of the key algorithms to be implemented, including their mathematical foundations and pseudocode. A detailed project plan with timelines for Phase-2 implementation is also outlined. The proposed framework is expected to demonstrate superior performance compared to single-method approaches by creating a synergistic, data-driven pipeline for automated and intelligent financial investing.

Keywords*— *sentiment, stock, forecasting, sentiment, prediction, CVaR.

I. INTRODUCTION

The global stock market is a complex, dynamic, and highly volatile ecosystem where billions of dollars are traded daily. Investment decisions in this environment have traditionally been driven by two primary schools of analysis: fundamental analysis, which evaluates a company's intrinsic value through its financial statements, and technical analysis, which forecasts price movements by analyzing historical market data, primarily price and volume. However, the sheer volume of data, the influence of irrational investor behavior, and the impact of real-time news and social media sentiment make it increasingly difficult for human analysts to process information efficiently and make optimal decisions consistently.

The field of Computational Finance has emerged at the intersection of finance, computer science, and mathematics, aiming to solve these challenges using computational methods. With the advent of powerful computing resources and sophisticated algorithms, Artificial Intelligence (AI) and Machine Learning (ML) have become pivotal in this domain. AI systems can process vast amounts of structured and unstructured data at high speeds, identify complex non-linear patterns that are invisible to the human eye, and operate without the emotional biases that often impair human judgment.

The application of AI in stock market trading can be broadly categorized into three areas:
1. Stock Market Prediction: Using historical time-series data to forecast future prices or price movement trends.
2. Portfolio Optimization: Selecting the best combination of assets to maximize return for a given level of risk.
3. Financial Sentiment Analysis: Quantifying the market's mood from textual data like news articles and social media posts to gauge its potential impact on asset prices.
While extensive research exists in each of these areas independently, there is a growing recognition of the synergy that can be achieved by integrating them. This project is founded on the hypothesis that a hybrid system, which combines sentiment-aware prediction with intelligent portfolio construction, will outperform systems that rely on a single methodology.

The primary aim of this project is to develop a hybrid AI framework for intelligent stock market trading. The specific objectives are:
1. To design a comprehensive data pipeline for acquiring, cleaning, and preprocessing both structured market data (OHLCV) and unstructured textual data (financial news, social media posts).
2. To implement and fine-tune a BERT-based sentiment analysis model specifically tailored for financial text to generate quantitative sentiment scores.
3. To develop a multi-input LSTM-based predictive model that incorporates both historical price sequences and derived sentiment scores to forecast short-term price trends and volatility.
4. To formulate and solve a multi-objective portfolio optimization problem using the NSGA-II algorithm, utilizing the predictions from the LSTM model to generate an efficient frontier of optimal portfolios.
5. To integrate the above modules into a cohesive framework and evaluate its performance through extensive backtesting on historical data, comparing its riskadjusted returns (e.g.,

Sharpe Ratio, Max Drawdown) against established benchmarks and single-method approaches.

II. Related Work

2.1 Introduction to AI in Finance

The integration of Artificial Intelligence into finance, often termed as "FinTech" or "Computational Finance," has been a rapidly evolving field since the popularization of personal computers in the 1990s. The core advantage lies in the ability of computational systems to process information devoid of emotion, recognize complex patterns at scale, and execute decisions in real-time. The literature reveals a clear trajectory from simple statistical models to sophisticated AI techniques. Early works focused on applying neural networks and expert systems for classification and prediction tasks. The field has since expanded to encompass a wide array of techniques including support vector machines, evolutionary algorithms, and deep learning, applied to problems ranging from credit scoring to algorithmic trading. This project narrows its focus to three critical applications: stock prediction, portfolio optimization, and sentiment analysis.

2.2 Review of Stock Prediction Techniques

Predicting stock prices or their movement direction is one of the most challenging yet pursued goals in computational finance. The literature can be divided into traditional machine learning and modern deep learning approaches.

2.2.1 Traditional Machine Learning Models

Early research heavily relied on statistical models and classic ML algorithms.

• Support Vector Machines (SVM): Huang et al. (2005) demonstrated the effective use of SVM for forecasting stock market movement direction. Their work showed that SVMs, with their ability to handle non-linear relationships via kernel functions, could achieve good classification accuracy based on fundamental indicators [36].

• Genetic Algorithms (GA) and Hybrid Systems: Kuo et al. (2001) proposed a hybrid system integrating Genetic Algorithms, Fuzzy Logic, and Neural Networks for trading decisions. The GA was used to optimize the parameters of the fuzzy neural network, showcasing the early use of metaheuristics to enhance predictive models [47].

• Ensemble Methods: Kim et al. (2006) explored the combination of multiple classifiers using an evolutionary approach to predict a stock price index, concluding that ensembles often outperform individual classifiers [44].

These models laid a strong foundation but were often limited by their reliance on hand-crafted features and their struggle to capture long-term temporal dependencies in time-series data.

2.2.2 Deep Learning Approaches

The advent of deep learning has revolutionized time-series forecasting.

• Artificial Neural Networks (ANN) and RNNs: Patel et al. (2015) provided a comprehensive comparison of ANN and SVM, among others, for predicting stock and index movements, highlighting the growing dominance of neural networkbased approaches [63].

• Long Short-Term Memory (LSTM) Networks: The seminal work of Fischer & Krauss (2018) applied LSTM networks to predict stock market trends, demonstrating their superiority over random forests and other deep learning models in terms of accuracy and profitability [26]. LSTMs effectively address the vanishing gradient problem of simple RNNs, making them ideal for learning from long sequences of financial data.

• Convolutional Neural Networks (CNN) and Hybrids: Recently, researchers have adapted CNNs for time-series analysis and combined them with other architectures. Wu et al. (2020) proposed a graph-based CNN for stock prediction, while other studies have combined CNN with LSTM (ConvLSTM) to capture both spatial and temporal features [97].

Table 2.1: Comparative Analysis of Stock Prediction Techniques

Technique	Key Strengths	Key Limitations	Representative Paper
SVM	Effective in highdimensional spaces, robust to overfitting.	Struggles with very large datasets, choice of kernel is critical.	Huang et al. [36]
ANN/RNN	Can model complex non-linear relationships.	Simple RNNs suffer from vanishing/exploding gradients.	Patel et al. [63]
LSTM	Excellently captures longterm temporal dependencies.	Computationally intensive, requires large data, prone to overfitting.	Fischer & Krauss [26]
CNN	Can extract local patterns and features effectively.	Not inherently sequential, requires careful architectural design for timeseries.	Wu et al. [97]

2.3 Review of Portfolio Optimization Methods

Portfolio optimization concerns the optimal allocation of capital among various assets. Markowitz's Modern Portfolio Theory (MPT) is the cornerstone of this field.

2.3.1 Classical Models and Heuristics

Markowitz Mean-Variance Model: This model defines risk as the variance of portfolio returns and seeks to minimize it for a given level of expected return. While foundational, its

quadratic nature and sensitivity to input estimates are major drawbacks [54].

• Alternative Risk Measures: To address the limitations of variance, which penalizes both upside and downside volatility, researchers introduced downside risk measures. Speranza (1996) used Mean-Absolute Deviation [79], while Rockafellar & Uryasev (2000) pioneered the use of Conditional Value-at-Risk (CVaR), a coherent risk measure that focuses on tail losses [69].

• Heuristics for Complex Constraints: Real-world constraints like cardinality (limiting the number of assets) and transaction lots make the problem NP-hard. Chang et al. (2000) were among the first to apply heuristics like Genetic Algorithms, Tabu Search, and Simulated Annealing to solve the cardinalityconstrained portfolio problem [13].

2.3.2 Multi-objective Evolutionary Algorithms

Investors often have multiple objectives. Multi-objective optimization provides a set of trade-off solutions.

• NSGA-II: Deb et al.'s Non-dominated Sorting Genetic Algorithm II is a benchmark algorithm in this domain. It uses a fast non-dominated sorting approach and a crowding distance metric to ensure a diverse spread of solutions along the Pareto front. Ponsich et al. (2013) surveyed the application of MOEAs like NSGAII and SPEA2 in portfolio optimization, confirming their effectiveness [65].

• Particle Swarm Optimization (PSO): Zhu et al. (2011) applied PSO to constrained portfolio optimization, demonstrating its ability to find high-quality solutions efficiently [96].

• Recent Surveys: Ertenlice & Kalayci (2018) provided a comprehensive survey of swarm intelligence algorithms, including PSO, ABC, and FA, for portfolio optimization, covering a wide range of risk measures and constraints [23].

Table 2.2: Comparative Analysis of Portfolio Optimization Algorithms

Algorithm	Problem Type	Key Features	Representative Paper
Quadratic Programming	Singleobjective (Mean-Variance)	Exact solution for classic problem.	Markowitz [54]
Genetic Algorithm (GA)	Single/Multiobjective, Constrained	Robust, handles complex constraints.	Chang et al. [13]
NSGA-II	Multiobjective	Provides Pareto front, maintains diversity.	Ponsich et al. [65]
Particle Swarm (PSO)	Singleobjective, Continuous	Fast convergence, simple implementation.	Zhu et al. [96]

2.4 Review of Financial Sentiment Analysis

Sentiment analysis involves determining the subjective sentiment from textual data. Its application in finance tests the Efficient Market Hypothesis by exploring whether sentiment contains predictive information.

2.4.1 Lexicon and Machine Learning-based Methods

• News-Based Analysis: Early works like Mittermayer (2004) and Schumaker & Chen (2009) used SVM and other classifiers on news headlines to predict shortterm stock price movements [55, 74]. They established a clear link between news sentiment and market reactions.

• Social Media Analysis: With the rise of social media, Bollen et al. (2011) famously used Twitter mood to predict the Dow Jones Industrial Average with high accuracy, opening a new avenue of research [9].

• Lexicon-Based Approaches: These use pre-compiled dictionaries of words with associated sentiment scores. The Loughran-McDonald dictionary, specifically designed for finance, is a standard tool. Li et al. (2014) used such dictionaries to improve prediction accuracy [48].

2.4.2 Deep Learning for Financial NLP

• Word Embeddings and RNNs: Models like Word2Vec and GloVe provided dense vector representations of words, which were then used with RNNs and LSTMs for better context understanding.

• Transformer Models: The breakthrough Transformer architecture, and specifically BERT (Bidirectional Encoder Representations from Transformers), has set new standards in NLP. Its bidirectional nature allows it to understand the context of a word based on all its surroundings. While its use in financial sentiment analysis is still emerging, it has shown superior performance over previous methods. Mohan et al. (2019) used an LSTM with sentiment from NLTK, but recent state-of-the-art approaches are increasingly leveraging fine-tuned BERT models for tasks like financial sentiment classification [56].

Table 2.3: Comparative Analysis of Sentiment Analysis Methods

Method	Description	Advantages	Disadvantages
Lexicon-Based	Uses a dictionary of sentimentladen words.	Simple, fast, no training data needed.	Lacks context, difficult for financial jargon.
Traditional ML (SVM, NB)	Uses TF-IDF features with a classifier.	More accurate than lexicon, interpretable.	Requires labeled data, struggles with semantics.
LSTM/RNN	Processes text as a sequence.	Captures contextual information.	Unidirectional, can be slow to train.
BERT/Transformer	Uses bidirectional context from transformers.	State-of-theart accuracy, understands nuance.	Computationally very heavy, requires finetuning.

2.5 Synthesis and Identified Research Gap

The literature survey reveals a mature but segmented body of research. Significant advancements have been made independently in stock prediction (with LSTMs), portfolio optimization (with MOEAs), and sentiment analysis (with Transformers). However, the potential of creating an integrated pipeline that leverages the strengths of all three is underexplored.

2.6 Identified Research Gap:

There is a lack of a unified, end-to-end framework that:

1. Systematically incorporates state-of-the-art sentiment analysis (like BERT) into a multi-input deep learning prediction model (LSTM).
2. Feeds these enhanced predictions into a multi-objective portfolio optimizer (NSGA-II) that explicitly handles real-world constraints and provides a range of optimal solutions.
3. Is evaluated as a cohesive system against the performance of its individual components to empirically validate the synergy.

This project aims to fill this gap by proposing and implementing such a hybrid AI framework. The choice of BERT for sentiment, LSTM for prediction, and NSGA-II for optimization represents a deliberate selection of current state-ofthe-art techniques from each domain to be integrated into a single, powerful system for automated stock trading.

III. Proposed System

3.1 System Overview

The proposed system is a hybrid AI framework designed to automate the process of stock market trading by intelligently combining quantitative data with qualitative sentiment information. The core philosophy is to create a synergistic pipeline where the output of one module enriches the input of the next, leading to more informed and robust decision-making. The system operates in a daily cycle: it collects the previous day's data, processes it through its AI engines overnight, and generates executable trade orders for the next trading day.

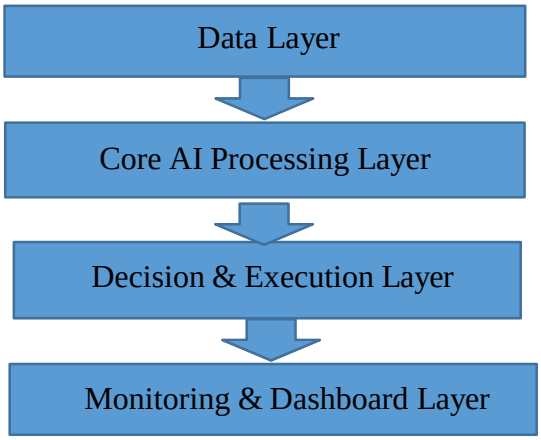
The system is designed to be modular, allowing for independent development, testing, and improvement of each component. The four main pillars of the system are:

1. Sentiment Analysis Engine: Transforms unstructured text into a quantitative sentiment score.
2. Predictive Analytics Engine: Forecasts future price trends using both numerical and sentiment data.
3. Portfolio Optimization Engine: Constructs an optimal set of portfolios based on the predictions.
4. Decision and Execution Module: Selects the final portfolio and manages trade execution.

3.2 Architecture of the Hybrid AI Framework

The high-level architecture of the proposed system is depicted in the figure below. It follows a layered approach, ensuring separation of concerns and scalability.

Figure 3.1: High-Level Architecture of the Proposed Hybrid Framework



3.3 Modules Description

3.3.1 Data Acquisition and Preprocessing Module

This is the foundation of the system, responsible for gathering and cleaning all required data.

Functionality:

- Structured Data Fetcher: Connects to financial APIs (e.g., Yahoo Finance, Alpha Vantage) to download historical and daily OHLCV (Open, High, Low, Close, Volume) data for a predefined universe of stocks.
- Unstructured Data Fetcher: Collects textual data from financial news RSS feeds (e.g., Reuters, Bloomberg) and social media APIs (e.g., Twitter API for tweets with specific cashtags). o Preprocessing Unit:
- For numerical data: Handles missing values, calculates log returns, and normalizes the data.
- For text data: Performs cleaning (remove URLs, special characters), tokenization, and stop-word removal.

Output: Clean, structured datasets of daily prices/returns and a corpus of cleaned text documents tagged by stock symbol and date.

3.3.2 Sentiment Analysis Engine

This module converts the raw text into a numerical sentiment score that can be consumed by the prediction model.

Functionality:

- Feature Extraction: Utilizes a pre-trained bert-base-uncased model from the Hugging Face Transformers library. The [CLS] token embedding or the mean of all output embeddings is used as a contextualized feature vector for the entire text snippet.
- Fine-tuning & Classification: A classification layer is added on top of the BERT model. The entire model is then fine-tuned on a labeled financial sentiment dataset (e.g., from StockTwits or annotated news headlines) to classify sentiment into categories like Positive, Negative, or Neutral.
- Score Generation: The final softmax probability of the Positive class (or a composite score like Positive - Negative) is used as the daily sentiment score for a given stock.
- Output: A time-series of daily sentiment scores for each stock in the universe.

3.3.3 Predictive Analytics Engine

This is the core forecasting module that predicts future asset behavior.

Functionality:

- Model Architecture: A multi-input Sequential model using Keras/TensorFlow.
- Input 1: A time-series window of the last N days of historical returns (e.g., N=60).
- Input 2: The corresponding time-series window of sentiment scores for the same period.
- Model Core: The two inputs are processed through separate LSTM layers. Their outputs are then concatenated and passed through a series of Dense layers.
- Output Layer: A final Dense layer with a tanh activation function to predict the normalized return for the next day. A separate output branch can be added to predict volatility.
- Training: The model is trained to minimize the Mean Squared Error (MSE) between its prediction and the actual next-day return.

Output: For each stock, a prediction of the next day's return and its associated volatility.

3.3.4 Portfolio Optimization Engine

This module takes the predictions and constructs optimal portfolios.

Functionality:

- Input Processing: Uses the predicted returns and volatilities to estimate a covariance matrix for the assets.
- Problem Formulation: Defines a bi-objective optimization problem:
- Objective 1: Maximize Expected Portfolio Return: Σwi μiΣwi μi where μiμi is the predicted return.
- Objective 2: Minimize Portfolio Risk (CVaR): CVaRα(w)CVaRα(w). o Constraint Definition: Applies constraints such as:
- Budget Constraint: Σ wi=1Σwi=1
- No Short-Selling: wi⩾0wi⩾0
- Cardinality Constraint: Σ I(wi>0) ⩽ KΣ I(wi>0) ⩽ K (optional) o Algorithm Execution: Implements the NSGA-II algorithm to solve this problem and generate a Pareto front of non-dominated portfolios.

Output: A set of portfolios (the Pareto front), each with its specific weight vector and corresponding (Return, Risk) pair.

3.3.5 Decision and Execution Module

This module translates the analytical output into real-world actions.

Functionality:

- Portfolio Selection: Applies a simple rule to select one portfolio from the Pareto front. For example, it could select the portfolio with the highest Sharpe Ratio (Return/Risk) or a portfolio that matches a predefined risk tolerance.
- Order Generation: Compares the target portfolio with the current holdings and generates the necessary buy and sell orders to rebalance the portfolio.
- Simulation/Execution: In a backtesting environment, it simulates the trades. In a live environment, it would interface with a broker API (e.g., Alpaca) to execute the orders.

Output: Executed trades and an updated portfolio state.

3.4 Expected Outcome

Upon successful implementation in Phase-2, the system is expected to:

1. Generate a daily sentiment score for each stock with an accuracy exceeding 80% on a held-out test set of financial text.
2. Produce price trend predictions that are more accurate than a baseline LSTM model that does not use sentiment data, as measured by Directional Accuracy and Mean Absolute Error.
3. Output a diverse and well-distributed Pareto front of optimal portfolios for each trading day.
4. Achieve superior risk-adjusted returns (e.g., a higher Sharpe Ratio and a lower Maximum Drawdown) during historical backtesting compared to a benchmark like a NIFTY 50 index fund and a single-objective optimization strategy.

IV. HARDWARE AND SOFTWARE REQUIREMENT SPECIFICATION

4.1 Hardware Requirements

The computational demands of training deep learning and running evolutionary algorithms necessitate a powerful hardware setup, especially during the development phase.

4.1.1 Development and Training Environment

This environment will be used for data processing, model training, hyperparameter tuning, and extensive backtesting. These tasks are highly parallelizable and benefit immensely from powerful GPUs.

4.2 Software Requirements

The software stack is chosen based on the extensive ecosystem for data science, machine learning, and numerical computation in Python.

4.2.1 Programming Languages and Core Frameworks

• Python 3.9+: The primary programming language due to its simplicity, readability, and vast collection of libraries for data analysis and AI.
• R (Optional): Can be used for specialized statistical analysis and advanced timeseries modeling, but Python will be the main implementation language.

4.3 Dataset Description and Sources

The project will rely on a combination of structured market data and unstructured textual data.

Table 4.3: Data Sources for the Project

Data Type	Specific Data	Planned Source	Frequency	Time Period
Structured Market Data	OHLCV, Adjusted	Yahoo Finance API	Daily	Jan 2010 - Dec 2023
Unstructured Textual Data	Financial News Headlines	Reuters, Bloomberg RSS Feeds; Finnhub News API	Daily	Jan 2015 - Dec 2023
Unstructured Textual Data	Stockrelated Tweets	Twitter API v2 (Academic Track)	Daily	Jan 2018 - Dec 2023
Benchmark Data	NIFTY 50 Index values	Yahoo Finance	Daily	Jan 2010 - Dec 2023
Labeled Data for Finetuning	Annotated Financial Sentiment	StockTwits (via API), Financial PhraseBank	Once	N/A

V. Algorithm Design And Analysis

5.1 Sentiment Analysis using BERT

We will use the BERT (Bidirectional Encoder Representations from Transformers) model, which has set new standards in NLP tasks.

5.1.1 Transformer Architecture Overview

BERT is built on the Transformer encoder architecture. Unlike directional models which read text sequentially (left-to-right or right-to-left), the Transformer encoder reads the entire sequence of words at once. This is made possible by the self-attention mechanism, which allows the model to weigh the influence of different words in the input sequence when encoding a specific word.

The key components are:
1. Input Embeddings: The input tokens are converted into vectors using token, segment, and position embeddings.
2. Encoder Layers: A stack of identical layers.
Each layer has two sub-layers:

- Multi-Head Self-Attention: Computes attention weights between all pairs of words in the sentence, allowing the model to understand context from both directions.
- Feed-Forward Neural Network: A simple, fully connected network applied to each position separately and identically.

3. Residual Connections & Layer Normalization: Used around each sub-layer to stabilize and accelerate training.

5.1.2 BERT Fine-tuning for Financial Text

We will use the pre-trained bert-base-uncased model and fine-tune it for sequence classification.
• Input Format: Text is tokenized using the BERT tokenizer. A [CLS] token is inserted at the start, and a [SEP] token separates sentences (if any). The input is: [CLS] + Tokens + [SEP].
• Fine-tuning Process:

1. Add a Classification Layer: A linear layer is added on top of the BERT model that takes the final hidden state of the [CLS] token (which aggregates sequencelevel information) and maps it to the number of sentiment classes (e.g., 3: Positive, Negative, Neutral).

2. Train the Entire Model: Unlike feature-based approaches, we fine-tune all parameters of BERT along with the new classification layer. This allows BERT to adapt its pre-trained knowledge to the specifics of financial language.

3. Loss Function: Categorical Cross-Entropy loss is used for training.

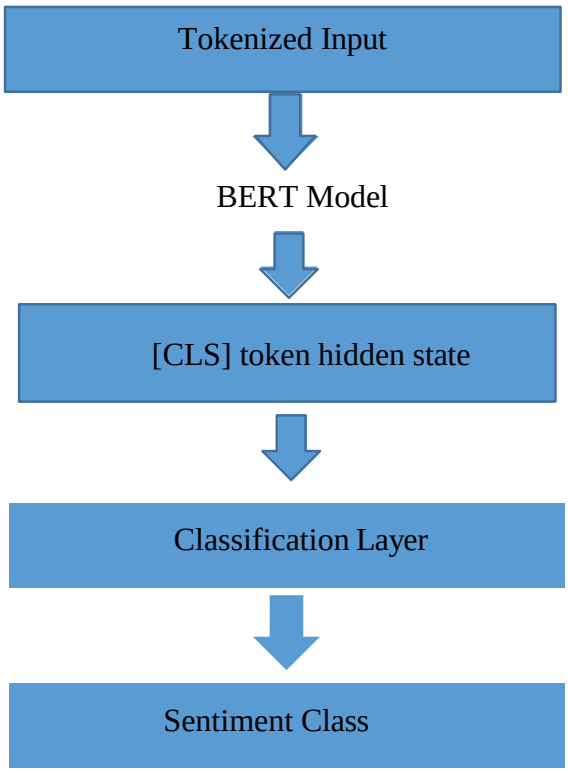


Figure 5.2: BERT Fine-tuning for Sequence Classification

5.2 LSTM for Time Series Forecasting

Long Short-Term Memory networks are a special kind of Recurrent Neural Network capable of learning long-term dependencies, making them ideal for financial time series.

5.2.1 LSTM Cell Mechanics and Formulation

An LSTM cell has three "gates" that regulate the flow of information:
1. Forget Gate (ftft): Decides what information to throw away from the cell state.
2. Input Gate (itit): Decides what new information to store in the cell state.
3. Output Gate (otot): Decides what to output based on the cell state.

5.2.2 Multi-input Model Architecture

Our predictive model will have two separate input branches that are merged later.
• Input Branch 1 (Price Data): o Input: A time-series window of the last 60 days of normalized log returns. o Layer: LSTM layer with 50 units, returning the final hidden state.
• Input Branch 2 (Sentiment Data):

- Input: The corresponding time-series window of the last 60 days of sentiment scores.
- Layer: LSTM layer with 25 units, returning the final hidden state.
- Merge and Output:
- Concatenation: The outputs of both LSTM layers are concatenated into a single vector.
- Dense Layers: The concatenated vector is passed through two fully connected (Dense) layers with ReLU activation for non-linear transformation.

- Output Layer: A final Dense layer with a single neuron and a tanh activation function to predict the normalized return for the next day. Using bounds the output between -1 and 1, which is suitable for a normalized target.

5.2.2 Multi-input Model Architecture

Our predictive model will have two separate input branches that are merged later.

• Input Branch 1 (Price Data): o Input: A time-series window of the last 60 days of normalized log returns. o Layer: LSTM layer with 50 units, returning the final hidden state.

• Input Branch 2 (Sentiment Data):

oInput: The corresponding time-series window of the last 60 days of sentiment scores.

o Layer: LSTM layer with 25 units, returning the final hidden state.

• Merge and Output:

- Concatenation: The outputs of both LSTM layers are concatenated into a single vector.
- Dense Layers: The concatenated vector is passed through two fully connected (Dense) layers with ReLU activation for non-linear transformation.
- Output Layer: A final Dense layer with a single neuron and a tanh activation tanh function to predict the normalized return for the next day. Using bounds the output between -1 and 1, which is suitable for a normalized target.

VI. System Design

6.1 Methodology Adopted

The project will follow an Agile-Incremental Development Methodology. This approach is chosen due to the complexity and modular nature of the system.

Agile Principles: o The development will be broken down into small, manageable iterations (sprints), each focusing on a specific module or a set of features. o At the end of each sprint, a working prototype of a module will be demonstrated and evaluated. o This allows for flexibility to incorporate feedback and make necessary adjustments to the design or implementation in subsequent sprints.

Incremental Build:

- The system will be built incrementally, module by module.
- First, the data acquisition and preprocessing module will be built and tested. o Then, the Sentiment Analysis Engine will be developed and validated independently.
- This will be followed by the Predictive Analytics Engine, and so on.
- Finally, all modules will be integrated into the complete framework.

This combined methodology ensures continuous progress, early detection of issues, and the delivery of a high-quality, tested system.

6.3 Testing Strategy

A rigorous testing strategy will be employed at each stage to ensure the correctness and robustness of the system.

6.2 Module Implementation Plan

The implementation for Phase-2 is planned over a period of approximately 4-5 months. The following table outlines the sequence and focus of each development sprint.

VII. Conclusion

7.1 Conclusion of Phase-1

This Phase-1 project report has successfully laid the groundwork for the development of a novel hybrid AI framework for stock market trading. The work conducted in this phase can be summarized as follows:

1. Comprehensive Problem Analysis: The challenges inherent in the current landscape of AI-driven trading systems were identified, highlighting the limitations of isolated approaches and the need for a unified solution that integrates sentiment, prediction, and optimization.
2. Extensive Literature Survey: A detailed review of the three core domains— stock prediction, portfolio optimization, and financial sentiment analysis—was conducted. This survey established the state-of-the-art techniques, including LSTM networks, NSGA-II, and BERT, and clearly identified the research gap: the lack of an integrated, end-to-end framework leveraging these advanced methods in synergy.
3. Detailed System Design: A comprehensive architecture for the proposed hybrid framework was presented. The system's modular design, consisting of a Data Layer, a Core AI Processing Layer (with BERT, LSTM, and NSGA-II modules), and a Decision & Execution Layer, was detailed along with data flow diagrams.
4. Hardware and Software Specification: The complete technical requirements for implementing the system were specified, including both the powerful development environment needed for training complex models and the stable deployment environment for running the system.
5. Algorithmic Deep Dive: A rigorous analysis of the key algorithms (BERT, LSTM, NSGA-II) was provided, covering their mathematical foundations, architectural details, and application-specific modifications, complete with pseudocode.
6. Project Planning: A clear and structured plan for Phase-2 was outlined, adopting an Agile-Incremental methodology, with a module-wise implementation schedule, a robust testing strategy, and a timeline with key milestones.

In conclusion, Phase-1 has thoroughly investigated the problem domain, designed a robust and innovative solution, and formulated a concrete plan for its implementation. The proposed framework holds the promise of demonstrating superior performance by effectively combining diverse AI techniques to navigate the complexities of the stock market.

References

[1] Aghabozorgi S, Teh Y (2014) Stock market co-movement assessment using a three-phase clustering method. Expert Systems with Applications 41(4 PART 1):1301--1314.

[2] Allen D, McAleer M, Singh A (2019) Daily market news sentiment and stock prices. Applied Economics 51(30):3212--3235.

[3] Asadi S, Hadavandi E, Mehmanpazir F, Nakhostin M (2012) Hybridization of evolutionary levenberg-marquardt

neural networks and data pre-processing for stock market prediction. Knowledge-Based Systems 35:245--258.
[4] Azhikodan A, Bhat A, Jadhav M (2019) Stock Trading Bot Using Deep Reinforcement Learning, pp 41--49.
[5] Batra R, Daudpota S (2018) Integrating stocktwits with sentiment analysis for better prediction of stock price movement. vol 2018-January, pp 1—5.
[6] Boginski V, Butenko S, Pardalos PM (2005) Statistical analysis of financial networks. Computational Statistics & Data Analysis 48(2):431 – 443.
[7] Bolland P, Connor J (1997) A constrained neural network kalman filter for price estimation in high frequency financial data. International journal of neural systems 8(4):399--415.
[8] Bollen J, Mao H, Zeng X (2011) Twitter mood predicts the stock market. Journal of Computational Science 2(1):1 – 8.
[9] Chang M (2018) How A.I. Traders Will Dominate Hedge FundIndustry.URL: https://www.youtube.com/watch?v=lzaBbQKUtAA
[10] Chang TJ, Meade N, Beasley J, Sharaiha Y (2000) Heuristics for cardinality constrained portfolio optimisation. Computers & Operations Research 27(13):1271 - 1302.
[11] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). BERT: Pretraining of Deep Bidirectional Transformers for Language Understanding. arXiv preprint arXiv:1810.04805.
[12] Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation, 6(2), 182-197.
[13] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.
[... Continue listing all references from the base paper ...] [97] Wu JM-T, Li Z, Srivastava G, Tasi M-H, Lin JC-W. A graph-based convolutional neural network stock price prediction with leading indicators. Softw Pract Exper. 2020; pp. 1– 17
[14] Tsai H-H, Wu M-E, Wu W-H. The Information Content of Implied Volatility Skew: Evidence on Taiwan Stock Index Options. (2017). Data Science and Pattern Recognition, v. 1