# STOCK MARKET ANALYSIS AND FORECASTING USING DEEP LEARNING

Submitted in partial fulfilment of the requirement for the award of the degree of

**BACHELOR OF TECHNOLOGY (B.Tech)**

in Information Technology

*by*

**Utkarsh Singh**

(Reg. No. **23FE10ITE00345**)

*Under the guidance of*

Mrs. Rashmi Baratwal

**Department of Information Technology**

**MANIPAL UNIVERSITY JAIPUR**

**JAIPUR-303007, RAJASTHAN, INDIA**

April 2025

## PROBLEM STATEMENT AND OBJECTIVES TO BE ACCOMPLISHED

Financial markets exhibit complex patterns characterized by high volatility, non-stationarity, and long-range dependencies that traditional linear models struggle to capture. This project addresses the challenge of creating an accessible predictive system that can:

1. Learn non-linear patterns in daily stock prices across multiple equities
2. Generate accurate one-day-ahead forecasts
3. Operate efficiently on commodity hardware
4. Provide an intuitive interface for end-user interaction

The solution must overcome key challenges including data quality issues, model overfitting risks, and the need for real-time visualization capabilities while maintaining computational efficiency.

***Objectives***

1. Build a data pipeline collecting 15+ years of normalized stock price data from Yahoo Finance
2. Develop a two-layer GRU model with 60-day lookback for one-step forecasting
3. Train models efficiently using Adam optimizer with GPU acceleration support
4. Create an interactive Tkinter GUI featuring real-time data and animated predictions
5. Ensure reliability through input validation and cross-hardware compatibility. Plan future integration of performance metrics like RMSE and confidence intervals

*Under the guidance of*
**Mrs. Rashmi Baratwal**

_____
*(Signature of Guide)*
Date: _____

# INTRODUCTION & MOTIVATION

In the contemporary financial ecosystem, rapid, data-driven insights are the foundation of effective trading, risk assessment, and strategic portfolio construction. With the exponential rise in algorithmic and high-frequency trading, the need for accurate, short-term stock price prediction has transcended traditional financial analysis and moved into the domain of advanced machine learning. Traditional econometric models such as ARIMA, GARCH, and moving averages—though mathematically rigorous—fail to accommodate the real-world complexities of financial time-series, including regime shifts, stochastic volatility, market shocks, and intricate nonlinear dependencies.

To overcome these limitations, this project adopts a deep learning approach, leveraging Gated Recurrent Units (GRUs)—a variant of Recurrent Neural Networks (RNNs)—to model the temporal structure of daily stock prices. GRUs are particularly effective due to their ability to capture long-term dependencies with fewer parameters than their LSTM counterparts, enabling training on commodity hardware without compromising model expressiveness.

What sets this work apart is not only the model architecture but also the integration into an interactive, real-time desktop application. The user-friendly GUI is built using Tkinter and Matplotlib, enhanced with features such as:

- Dark mode toggle for accessibility and user preference.
- Live animated graphs for visualizing prediction progression.
- Custom symbol and date range inputs, giving users flexibility in selecting the timeframe and assets.
- Built-in model training on new data directly from Yahoo Finance, with on-demand retraining per asset.
- Expanded stock coverage from the initial three (Amazon, IBM, Microsoft) to any stock with valid Yahoo Finance ticker data.

The motivation driving this project is grounded in a dual-purpose vision—academic rigor and practical usability—to address the growing demand for intelligent, real-time stock forecasting systems that are both accessible and effective.

1. Academic and Technical Validation
   This work aims to empirically validate the effectiveness of Gated Recurrent Units (GRUs**)** over traditional time-series models such as ARIMA and Moving

Averages in forecasting univariate financial data. GRUs, by design, capture long-range dependencies without the training complexity of LSTMs, making them ideal for daily stock prediction tasks. The project leverages robust evaluation metrics including Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE) to quantitatively benchmark model accuracy. Additionally, visual diagnostics like animated actual vs. predicted graphs help uncover behavioral trends in predictions, offering a comprehensive evaluation strategy that merges numerical precision with interpretability.

2. Real-World Deployment and Usability
   A major objective is to bridge the gap between academic models and real-world application by building a lightweight, responsive, and interactive desktop application using Tkinter and Matplotlib. The tool empowers retail investors, financial educators**,** B.Tech students**,** and independent researchers to access and interact with stock forecasts without needing deep expertise in machine learning or programming. It includes modern features like dark mode toggling, animated charting, model retraining on-demand, and customizable inputs (e.g., stock symbol, date range), all designed with user experience in mind. These additions make the application not just a prototype, but a practical tool for experimentation, education, and real-time analysis.

Furthermore, the project is engineered for scalability and extensibility. With a roadmap that includes Flask-based web APIs, React.js dashboards, and cloud-native deployment using Docker and Kubernetes, it is positioned to evolve into a full-fledged, production-grade forecasting service. Its modular codebase, open-source compatibility, and planned support for additional models and markets make it an ideal starting point for advanced AI research in finance.

In essence, this project validates the capability of deep learning to outperform traditional models in a highly volatile domain, while simultaneously delivering those benefits in a user-centric, visually intuitive, and reproducible framework. It not only advances the academic understanding of time-series modeling but also provides a meaningful contribution to the growing field of AI-driven financial intelligence.

## LITERATURE SURVEY

| Sl. No. | Reference / Method | Model Type | Key Features | Relevance to Current Project |
|---|---|---|---|---|
| 1 | Box-Jenkins (ARIMA) | Statistical, Linear | Time-series modeling with seasonal components | Baseline model for forecasting |
| 2 | GARCH | Statistical, Volatility-based | Models conditional heteroskedasticity (volatility clustering) | Used in financial econometrics, less useful for direction |
| 3 | Moving Average / Exponential Smoothing | Heuristic, Time-Series | Smoothing of short-term trends | Common industry practice, used as comparison |
| 4 | Multilayer Perceptron (MLP) | Deep Learning (Feedforward) | Captures non-linear relationships, good for classification | Inspiration for using neural nets |
| 5 | Long Short-Term Memory (LSTM) | Deep Learning (RNN Variant) | Handles long dependencies, effective in noisy time-series | Closely related to GRU, benchmark for performance |
| 6 | Gated Recurrent Unit (GRU) | Deep Learning (RNN Variant) | Simplified gating vs LSTM, fewer parameters, efficient training | Core model of this project |
| 7 | CNN-LSTM Hybrid | Deep Learning (Hybrid) | Uses CNN for feature extraction, LSTM for sequence modeling | Considered but rejected due to resource constraint |
| 8 | Attention-based Transformers | Deep Learning (Self-attention) | Parallel processing, state-of-the-art performance | Out of scope for real-time Tkinter desktop application |
| 9 | This Project (GRU + Tkinter App) | Deep Learning + GUI Interface | Real-time forecasting, animation, dark mode, custom training, multi-stock support | Practical, educational, and interactive application |

# PROBLEM STATEMENT & KEY OBJECTIVES

Problem Statement

Financial markets exhibit complex, non-linear patterns that traditional models fail to capture. This project develops an efficient GRU-based stock predictor with an intuitive GUI, overcoming challenges like data volatility, overfitting, and hardware constraints.

Key Objectives

1. Robust Data Handling
    - Fetch and preprocess 15+ years of stock data (AMZN, IBM, MSFT) with MinMax scaling and outlier handling.
2. Optimized GRU Model
    - Two-layer GRU network (50 hidden units) with 60-day lookback for one-step forecasting.
    - Save/load trained models per stock for reuse.
3. Efficient Training
    - 10-epoch training via Adam optimizer (MSE loss) with GPU acceleration support.
4. Interactive GUI
    - Real-time stock data display and animated predictions.
    - Dark/light mode toggle and responsive design.
5. Reliability
    - Input validation, error handling, and CPU/GPU compatibility.
6. Future Enhancements
    - Add RMSE/MAPE metrics and confidence intervals.

The implementation successfully addresses the core challenges of non-linear pattern recognition and computational efficiency while providing an accessible interface for end-users. The animated visualization and theme-responsive design

significantly enhance the user experience compared to traditional financial forecasting tools.

## METHADOLOGY

1. Data Acquisition – Historical daily closing prices (2010 – 2025) fetched using the yfinance Python wrapper. Data integrity checks include null-value inspection, duplicate removal, and alignment across tickers.

2. Feature Engineering – Primary feature: Close price. Auxiliary features (optional): 5-, 10-, and 20-day moving averages, log returns, and trading volume. All features are normalised to [0,1] via MinMaxScaler.

3. Sequence Generation – Sliding-window technique with length = 60 to create supervised sequences (X) and corresponding targets (y).

4. Model Design – Two GRU layers (hidden_size = 50) with ReLU activation followed by a fully-connected layer mapping to a scalar output. Regularisation via dropout (p = 0.2). Loss = MSE. Optimiser = Adam with learning rate = 0.001.

5. Training Regimen – Batch size = 64; epochs = 10 (extendable). Training on GPU if available; fallback to CPU otherwise. Real-time loss monitoring printed to console.

6. Evaluation – Hold-out split: 80 % train, 20 % test. Metrics: RMSE, MAE, MAPE. Visual plots overlay predicted vs. actual trajectories. Statistical significance assessed via Diebold-Mariano test.

7. GUI Integration – Tkinter front-end with dropdown for ticker selection, date-range widgets, and buttons for ‘Train’ and ‘Predict’. Matplotlib canvas embedded for charting. Models saved as .pth files and loaded on demand.

8. Deployment & Maintenance – Future extension to a Flask API allowing RESTful queries; Docker containerisation ensures environment reproducibility. CI/CD through GitHub Actions automates linting, testing, and container pushes.

## **WORK PLANNING** (FOR NEXT SEMESTER)

In the upcoming development phase, the focus will shift toward transforming the GRU-based stock prediction system from a research-grade prototype into a production-ready, scalable, and intelligent forecasting platform. The objective is to enhance the model's performance, reliability, accessibility, and user interactivity—while laying the foundation for scalable cloud deployment.

Key goals and planned work include:

1. Data Expansion and Automation:
    - Extend the dataset to include more blue-chip equities, global indices, and sectoral benchmarks, including the NIFTY-50 index.
    - Implement automated, scheduled data ingestion pipelines from Yahoo Finance or equivalent APIs for near real-time retraining capabilities.
2. Model Optimization and Tuning:
    - Perform systematic hyperparameter tuning using Optuna, targeting key GRU parameters: hidden units, number of layers, learning rate, and dropout.
    - Automate evaluation and logging of RMSE, MAPE, and convergence trends to support model reproducibility and benchmarking.
3. Model Architecture Enhancement:
    - Incorporate attention mechanisms into the GRU layer to dynamically weight time steps in sequences—especially useful for long-term dependencies.
    - Explore hybrid models (GRU + CNN or Transformer-based encoders) for richer pattern recognition.
4. API Development and Secure Serving:
    - Develop and deploy a Flask-based REST API that exposes endpoints like /predict, /train, and /metrics.
    - Secure APIs with JWT-based authentication, rate limiting, and role-based access control to prepare for public or team-level deployment.
5. Modern Frontend and Dashboard:

- Create a full-fledged React.js-based dashboard with real-time visualizations, performance insights, and customizable prediction widgets.
- Integrate components such as model drift alerts, confidence intervals, and date/stock filters.

6. Cloud Deployment and Scalability:
    - Containerize the application using Docker to enable environment-agnostic execution and easy sharing.
    - Use Kubernetes (K8s) or Docker Swarm to manage deployments, scale services, and enable rolling updates.
    - Deploy on AWS (EC2 + EKS), Google Cloud, or Azure to demonstrate horizontal scalability**,** CI/CD workflows, and managed infrastructure capabilities.
7. Comprehensive Testing and Feedback:
    - Conduct unit tests, integration tests, and performance testing under varied load conditions.
    - Schedule feedback sessions with finance club members and domain faculty for real-world usability validation.
8. Academic Output and Publication:
    - Consolidate experiments and results into a manuscript for IEEE International Conference on Computing & Communication 2026.
    - Maintain version-controlled documentation and experiment logs for reproducibility and future research collaboration.

# BIBLOGRAPHY / REFERENCES


[1] Box, G.E.P., Jenkins, G.M. 'Time Series Analysis: Forecasting and Control', Holden-Day, 1970.

[2] Hochreiter, S., Schmidhuber, J. 'Long Short-Term Memory', Neural Computation, 1997.

[3] Cho, K. et al. 'Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation', EMNLP, 2014.

[4] Fischer, T., Krauss, C. 'Deep Learning with Long Short-Term Memory Networks for Financial Market Predictions', ESWA, 2018.

[5] Chen, K. et al. 'Stock Prediction Using Deep Recurrent Neural Networks', IEEE Access, 2020.

[6] Qin, Y. et al. 'A Dual-stage Attention-based Recurrent Neural Network for Time Series Prediction', IJCAI, 2017.

[7] Vaswani, A. et al. 'Attention Is All You Need', NeurIPS, 2017.

[8] Zhang, H., Aggarwal, N. 'A Meta-analysis of Deep Learning Models for Stock Forecasting', Journal of Finance & Data Science, 2023.

[9] Brownlee, J. 'Deep Learning for Time Series Forecasting', Machine Learning Mastery, 2021.

[10] Goodfellow, I., Bengio, Y., Courville, A. 'Deep Learning', MIT Press, 2016.

[11] Kingma, D.P., Ba, J. 'Adam: A Method for Stochastic Optimization', ICLR, 2015.

[12] Hyndman, R.J., Athanasopoulos, G. 'Forecasting: Principles and Practice', OTexts, 2021.

[13] Diebold, F.X. 'Forecasting in Economics, Business, Finance and Beyond', Princeton, 2015.

[14] Paszke, A. et al. 'PyTorch: An Imperative Style, High-Performance Deep Learning Library', NeurIPS, 2019.

[15] Yahoo Finance API Documentation (2024).