Journal of Computer Science

Research Article

A Dynamic Lifelong Learning Network for Real-Time Stock Prediction With Reinforcement Tuning

Talent Mawere, Selvaraj Rajalakshmi, Venu Madhav Kuthadi and Otlhapile Dinakanyane

Department of Computing and Informatics, School of Pure and Applied Sciences, Botswana International University of Science and Technology, Palapye, Botswana

Article history
Received: 20-03-2025
Revised: 08-07-2025
Accepted: 29-07-2025

Corresponding Author:
Talent Mawere
Department of Computing and Informatics, School of Pure and Applied Sciences, Botswana International University of Science and Technology, Palapye, Botswana
Email: talentmawere@gmail.com

Abstract: Companies and investors require accurate market forecasting to make more informed decisions. Traditional methods for predicting stock market performance have become less useful considering the dynamic and volatile nature that characterizes stock markets today. In this paper, we propose an innovative Dynamic Lifelong Learning Network for stock market prediction. The model incorporates a hybrid convolutional long short-term memory with an attention mechanism for spatiotemporal feature extraction. To address the problem of static batch processing common in most current machine learning techniques in use, we employ reinforcement learning through a Deep Q-Network for real-time adaptation. We then integrated Elastic Weight Consolidation to address the problem of catastrophic forgetting. The model inputs a comprehensive set of structured features, including historical OHLCV data, technical indicators, and macroeconomic variables such as interest rates and the Consumer Price Index. We applied principal component analysis to optimize dimensionality. The model was trained and tested on APPL data extracted from Yahoo Finance for the period 2010-2023. Macroeconomic features were extracted from the Federal Reserve Economic Data for the same period. Ablation studies confirmed the hypothesis that add-on features such as the attention network and RL-EWC improve the prediction capacity of our model by at least 12.35%. Comparison of our model with literature-identified baselines showed that our model performed much better, with an R^2 of 0.967±0.01 and an MAE of 3.22±0.27. Generalization testing on the SPY ticker, a key representative for the S&P 500 index, shows that the model is robust.

Keywords: Attention Networks, Dynamic Lifelong Learning Network, Elastic Weight Consolidation, Fisher Information Matrix, Hybrid Convolutional Long Short-Term Memory, Stock Prediction, Reinforcement Learning

Introduction

Modern-day society has thrived even in times of turmoil thanks to unprecedented levels of technological advancements. The COVID-19 era bears witness to this as the world continued to revolve even when restrictions were in place. If at all, the pandemic era demonstrated that technological advancement brings more convenience and better service delivery (Fejerskov, 2017; Hossain et al., 2021).

Artificial Intelligence (AI) is at the heart of all this. It has transformed and transcended every part of the economy, from agriculture to healthcare, and education (Adesina et al., 2024). The International Monetary fund predicts that AI will boost productivity through automation, improve decision-making, and create new business models that will stimulate economic growth.

These predictions are already being felt across all industrial sectors, but the greatest effect has been felt on the finance and investment services sector. Its role in the investment decision-making process, particularly in forecasting stock market prices, has been welcomed, especially by novice investors who regarded the domain as a preserve for the elite brokers (Bahoo et al., 2024; Khatwani et al., 2023; Pattnaik et al., 2024). Wang et al. (2024a-c) argues that accurately predicting stock market prices can result in more informed investment choices, thus allowing the optimization of returns while mitigating risk. This argument is premised on the fact that any rational investor will make smart investment decisions if they know which stocks are likely to do well in the future. Furthermore, accurate stock prediction facilitates efficient capital allocation to high-performing entities, thereby fostering economic growth and innovation.

We therefore forward the notion that through accurately predicting stock market trends, businesses and even individual investors can develop more informed financial strategies, including acquisitions, mergers, and other corporate actions. The general public also benefits from correct stock market predictions because they provide people with the information they need to make smart investments, which helps spread wealth more fairly.

It is still very hard to accurately predict stock market values, a challenge that not only has driven continuous innovation in both statistical and machine-learning techniques but also has become a major driver of research in the fields of computer science, economics, and finance (Wu et al., 2022; Ying et al., 2024), including this research.

In this study, we make the following contributions:

1) We develop a novel Dynamic Lifelong Learning Network (DLLN) specifically designed for real-time financial forecasting that continuously adapts to incoming data streams
2) Integrate a reinforcement learning agent in the DLLN implemented via the Lifelong Learning Stock Predictor algorithm to autonomously adjust the model's parameters in response to evolving market dynamics
3) Mitigate catastrophic forgetting in RNN by integrating Elastic Weight Consolidation, which allows the model to retain critical knowledge from previous tasks during continual learning
4) We add an attention mechanism into our model to selectively emphasize the most relevant time steps in the input sequence, which improves both prediction accuracy and computational efficiency

We propose the following hypotheses:

1) Adding Elastic Weight Consolidation modules will improve the predictive performance of the DLLN
2) The DLLN will outperform its industrial benchmarks due to its continual learning ability

Literature Review

Statistical Modelling Techniques

Stock market prediction techniques date back to the 17th century, when the first stock market, as we would recognize it today, emerged. Back then, prediction was solely based on subjective methods such as rumors and insider information. Purely computational techniques did not emerge until the 19th century, following theories such as the Dow theories, the random walk theories, and the efficient market hypothesis theory (Fama, 1970; Malkiel, 1973).

Statistical modeling techniques were among the first technical methods used for stock predictions. Statistical modelling techniques are broadly classified under three groups: time series models like Autoregressive Integrated Moving Average (ARIMA) and GARCH, regression analysis models like linear regression and logistic regression, and stochastic process models like the Geometric Brownian Motion. A comprehensive literature review by Ayyildiz and Iskenderoglu (2024) revealed that models like ARIMA and Markov chains have performed well in market prediction tasks with average accuracy rates typically ranging between 60 and 70% i.e., in studies by Chen et al. (2022).

Existing research has shown, however, that despite these techniques being based on solid mathematical and statistical backgrounds, their predictive performance is average. The extant literature mainly attributes the oversimplification of complex relationships in data as the major inhibitor to better performance. Most models in this category assume that data follows a linear pattern. In financial markets, however, data is nonlinear and multidimensional in nature. Deng et al. (2022) raised skepticism towards these techniques because they rely on short-term historical data, often neglecting longer-term trends.

Machine Learning Techniques

After statistical modelling techniques, AI and Machine Learning (ML) came as the next revolution in stock prediction and financial forecasting. These algorithms are built to handle massive amounts of data to make more informed inferences. Several studies, including Chen et al. (2022), note the transformational impact of ML on stock market prediction. Different studies have demonstrated the superiority of different ML techniques in recent studies. For example, Zheng et al. (2024) demonstrated the predictive superiority of Random Forest (RF), while Kuo and Chiu (2024) proved that the Support Vector Machine (SVM) was better than others. In recent studies, models like the K-Nearest Neighbour (Qin, 2024), Gradient Boost Machine (Liao et al., 2024), and K-means clustering (Chen et al., 2022), were observed to have better predictive capacities compared to some baseline models. The improved predictive capacity of ML models has been attributed not only to their capacity to handle nonlinear data but also to their ability to handle large volumes of data (Ghosh et al., 2024).

However, existing literature also identifies several deficiencies in ML techniques. Chen et al. (2022)

elucidates their challenges in managing complex and high-dimensional data. Ghosh et al. (2024) on the other hand delineate their stuggle with overfitting. An algorithm is overfitting when it works well on data it has been trained on data but poorly on data it has never seen before. This means that the model can't apply what it has learnt to other situations, which makes it less useful for predictive tasks such as stock market forecasting. ML techniques have also been criticized for relying too much on manual feature engineering (Hadizadeh et al., 2025). In practice, poor or inconsistent feature choices erase important predictive signals making the model less accurate and less reliable.

Deep Learning Techniques

Deep learning (DL) techniques are changing the way people predict the stock market. DL is a type of ML that uses multi-layer neural networks. DL algorithms are designed to automatically find and learn important patterns in data, just like how the human brain works. This is very important for working with high-dimensional and unstructured data, which is difficult to do with regular ML algorithms (Li and Bastos, 2020). In multiple studies, DL models have outperformed their ML equivalents (Chatziloizos et al., 2024; Lee et al., 2024; Wei et al., 2024).

Examples of typical DL algorithms that have been applied to stock market prediction include Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM), and Convolutional Neural Networks (CNNs). Current research indicates enhanced efficacy compared to ML and Statistical modelling techniques in stock prediction tasks.

DL techniques have made predictions much more accurate, but they have some problems that limit their use. Examples include limited computational efficiency, catastrophic forgetting and exploding gradients. These limitations, most notably high computational complexity, pose significant challenges for the practical deployment of artificial neural networks in real-time financial markets. Work in the field currently revolves around the development of lightweight DL architecture. Continuous learning integration in market prediction tasks, which can potentially address some of these limitations, is also an active research area, one that this research seeks to contribute towards.

Continuous Learning Integration in DL

One of the biggest problems is that most AI models can't keep up with new and changing data trends (De Lange et al., 2022). The standard approach is to train models on static datasets. This way of training our models means that we must train it repeatedly to keep it useful. In infinite data streams which characterize stock markets data, this is very impractical and computationally expensive. In this paper, we propose a system capable of lifelong learning. In this system, an agent's knowledge evolves dynamically as it learns new data patterns while preserving critical knowledge. Lifelong learning is inspired by biological learning systems, where an agent retains and refines knowledge over time (Kudithipudi et al., 2022).

Research in continual learning systems is growing but there are problems like concept drift that need to be solved. Continual learning systems are also built to address the problem of catastrophic forgetting. A number of techniques have been proposed in building such systems. In this paper we present an adaptive forecasting framework with lifelong learning capabilities. More specifically, our study proposes the use of Elastic Weight Consolidation algorithm as well as a reinforcement learning technique (Deep Q Network) for continuous learning capabilities. Our design guarantees that the model adapts to changing market conditions.

Previous studies, such as Wei et al. (2024), have presented a Gated Recurrent Unit (GRU) with attention mechanisms to tackle the challenges of capturing long-term dependencies. It has been noted that neglecting this may result in catastrophic forgetting in DL models (Parisi et al., 2019). The GRU layer's main job is to keep long-term dependencies while keeping unnecessary data from building up. The attention mechanism, on the other hand, facilitates the model's capacity to concentrate its analytical efforts on most important parts of the data sequence necessary for making predictions. Oğuz and Ertuğrul (2023) posits that when working with a lot of data, GRU networks increase the speed and performance of LSTM networks. Although GRU-only architectures have been observed to capture long-term dependencies, they have been criticized for overlooking rapid intraday fluctuations. Similarly, Jin (2024) undertook a study to develop a hybrid model that combines the strengths of graph neural networks (GNNs) and CNNs for short-term price trend prediction. The model remarkably outperformed the traditional CNN-based models and graph techniques, showing some promise as a viable candidate for integration into real-world trading systems as a prediction module. Table 1 summaries some of the recent literature.

While prior studies have demonstrated promising advances in stock market prediction using deep learning, several methodological and conceptual gaps remain insufficiently addressed. Most notably, existing models tend to operate under static learning regimes, lacking the capacity to dynamically recalibrate in response to non-stationary market behaviour. Furthermore, despite the proliferation of transformer-based and hybrid neural models, there is limited exploration of mechanisms to systematically preserve learned knowledge over time, particularly in multi-task or longitudinal financial settings.

Table 1: Summary of literature review

Author	Context	Method	Dataset	Key contribution	Gaps
Wu et al. (2022)	Stock price prediction (regression)	S_I_LSTM (Att-LSTM + CNN)	5 Chinese A-share stocks, EastMoney.com news/posts	Combines traditional OHLCV + technical indicators + sentiment index. Sentiment derived via CNN on financial text.	No real-time adaptation; no continual learning; Single-market dataset; Limited ablation study
Dang et al. (2020)	Sentiment Polarity Classification	DNN, CNN, RNN using TF-IDF & Word Embedding	Sentiment140, IMDB, Twitter Airline, SemEval, Book, Music & Movie Reviews	Conducted a systematic evaluation of DNN, CNN, and RNN across eight datasets with 2 input schemes; found CNN + Word Embedding delivered the best trade-off (accuracy ~90%) vs. training time; RNN achieved highest accuracy but incurred long runtime; models were tested with k-fold CV and metrics included accuracy, F1, AUC, and CPU time	No attention mechanisms; only tested polarity (not aspect); no multilingual benchmarks; did not incorporate ensemble or transformer models; lacks real-time or continual learning focus
Berti and Kasneci (2025)	Stock Price Trend Prediction	TLOB (Transformer w/ Dual Attention), MLPLOB	FI-2010, Tesla, Intel (LOB data)	Proposed TLOB: transformer with temporal & spatial attention + bilinear normalization; F1 = 92.81% (FI-2010); MLPLOB outperforms prior SoTA for shorter horizons	No discussion of sentiment signals; macroeconomic indicators, or transaction profitability backtesting; no multilingual datasets
Chauhan et al. (2025)	Stock Index Prediction (Regression & Classification)	SenT-In (Informer + CNN-GRU Sentiment Awareness Model)	S&P 500, FTSE, SSE, Nifty 50	Developed a hybrid model combining financial news sentiment (CNN-GRU) with time-series market data via Informer. Introduced sentiment-aware fusion mechanism. Outperformed baselines (LSTM, GRU, CNN, SVM) across accuracy, F1, AUC-ROC, PR-AUC. For S&P 500, achieved Acc: 0.8159, F1: 0.8174, AUC-ROC: 0.9331, PR-AUC: 0.9465.	No reinforcement learning or continual learning; sentiment limited to structured financial news (no social media); no profitability/backtesting analysis; interpretability not evaluated in depth
Wang and Vo (2025)	Stock Price Prediction (Regression)	ARIMAX with Dual Important Indicators (VWAP, WCP, FWMA, Decay, ZLMA)	Vietnam Stock Index (VNINDEX) 2013–2023; Pham & Ta (2010–2021); Do & Trang (2001–2019)	Proposed a novel feature selection strategy using lag-0 and lag-1 XGBoost to identify 5 "dual important indicators" relevant in both past and future. Demonstrated superior forecasting using ARIMAX, outperforming LSTM, GALSTM, XGBoost, Meta Prophet. Achieved MAPE = 9.05×10^{-16} and R^2 = 99.99% on VNINDEX	No sentiment integration; relies solely on technical indicators; limited generalization across different geographic markets; potential selection bias in dual indicator extraction
Sharma et al. (2025)	Stock Price Prediction (Regression on NFLX close)	LSTM-based Sequential Model with feature correlation analysis	Netflix stock data (2010–2024) from Yahoo Finance via yfinanc	Developed LSTM-based model trained on 14 years of OHLCV data; architecture: 2 stacked LSTM layers (128 & 64 units) + Dense layers; achieved visually reasonable forecasts for future 100 days of Netflix closing prices; performed correlation and candlestick chart analysis	Lacks quantitative evaluation metrics (e.g., R^2, MAE); no comparison with benchmarks (e.g., ARIMA, Transformer, GRU); limited feature diversity (no sentiment, macro factors); no real-time or continual learning integration
Wang (2025)	Stock Market Stability Prediction	CFBSPM (Contradictory-Factor-Based Stability Prediction Model using Sigmoid & Non-Sigmoid DL layers)	Stock data (Feb–May 2023) for 4 companies (Kaggle) + CSI 300 stability index	Proposed a novel sigmoid-based deep learning model that jointly evaluates stock market stability using economic, sentiment, and trading factors; Combines sigmoid layers for probabilistic stability classification with non-sigmoid layers for abrupt deviation detection; Outperformed baseline models (e.g., SMP-DL, HDFM, PPO-TLSTM) in accuracy (96.12%), stability matching (94.68%), and detection time (0.05s); Conducted a detailed empirical analysis using 7 influencing factors across multiple months.	No incorporation of attention mechanisms or memory-based layers like LSTM in final model; limited market diversity (mostly CSI 300 and Kaggle-listed firms); no sentiment sourced from social media; model lacks interpretability/explainability modules (e.g., SHAP)
Oyewole et al. (2024)	Stock Price Prediction (Forecasting, Classification)	Multiple ANN variants (MLP, RNN, LSTM, CNN, GAN); Qualitative synthesis	Global literature (e.g. S&P 500, NIFTY 50, NSE, NYSE)	Conducted a comprehensive review and synthesis of neural network architectures for stock prediction; Highlighted data preprocessing, interpretability challenges, architecture-specific strengths, and trade-offs (e.g. LSTM vs GAN vs CNN); Emphasized role of high-quality/quantity data, ensemble models, and hybrid designs; Reported	Qualitative meta-review only; lacks unified experimental benchmark; no deep comparative metrics; no real-time testing; interpretability still unresolved in most models; no direct profitability/backtesting outcomes

				accuracies in cited works exceeding 90% (e.g., ANN = 93% on Indian stocks; CNN = 99% on Iraqi data).	
Lee et al. (2024)	Predict S&P 500 index by integrating ESG sentiment (from news) with technical indicators using deep learning.	ESG sentiment score via FinBERT; Models: Bi-RNN, Bi-LSTM; Evaluation: MAPE (%); Hyperparameters: Window sizes (3/4/5), batch sizes (2-8), hidden layers (4-8)	ESG News: 14,049 LexisNexis articles (Jan 2016–Jul 2023) Market Data: S&P 500 historical prices/indicators (investing.com)	Best Model: Bi-LSTM (window = 3, batch = 64, hidden = 32/64, layers = 2) MAPE = 3.05%; ESG + technical indicators outperformed baselines: - Only price: MAPE = 3.81-4.87% - Price + technical: MAPE = 3.48-3.75% - Full model (ESG + technical + price): MAPE = 3.05-3.55%; Ablation tests confirmed ESG causality with S&P 500.	S&P 500 focus limits generalizability; Reliance on news data only for ESG sentiment; Industry-specific ESG impacts not explored
Song et al. (2024)	Gold Stock Price Forecasting (Regression)	BO-LSTM (Bayesian-Optimized LSTM)	Gold stock price data (exact period not specified)	Proposed a novel LSTM model optimized via Bayesian networks for hyperparameter tuning (e.g., learning rate, hidden layers). Achieved improved prediction accuracy, robustness, and noise resistance over baseline LSTM. Empirical results show reductions in RMSE (e.g., 13.88 vs 17.50) with visual forecast alignment.	Focused solely on gold stocks; lacks broader market generalization; no integration of sentiment or macroeconomic factors; no benchmark against transformer-based or ensemble models; explainability and interpretability not explored in depth
Nejad and Ebadzadeh, (2024)	Stock Price Forecasting (Regression)	DRAGAN + Feature Matching + GRU Generator + CNN Discriminator	12 U.S. stocks (2010–2020, Yahoo Finance); daily OHLCV + 8 technical indicators	Proposed a stabilized GAN framework (DRAGAN) with windowing and conditioning to capture temporal and feature correlations. Integrated feature matching to mitigate mode collapse. Outperformed baseline models (LSTM, Basic GANs, WGAN-GP) across RMSE, MAE, and R^2 (e.g., Apple RMSE = 0.92, R^2 = 0.98; avg RMSE across stocks ≈ 1.105). Supported many-to-many mappings and demonstrated high distributional fidelity between predicted and real prices.	No sentiment integration; lacks macroeconomic variables; limited interpretability methods; some training instability remains; not evaluated in live trading or profitability context
Tan, (2024)	Stock Price Prediction (Regression)	LSTM, Random Forest, Linear Regression	NVIDIA stock prices (1999–2023), focus on 2016–2023 closing prices	Compared three models, LSTM, RF, and LR, on predicting NVIDIA closing prices using historical data. RF outperformed others: MSE = 10.6, RMSE = 3.3, MAE = 1.0, R^2 = 0.997. LSTM followed (R^2 = 0.984), with LR performing worst (R^2 = 0.731). Provided visual plots and quantitative analysis across all methods.	Focused solely on univariate close prices; no technical indicators, sentiment, or macroeconomic variables; lacked advanced model tuning or explainability tools; no live trading or economic implication testing
Lu and Xu 2024)	Stock Price Forecasting (Regression)	TRNN (Time-series Recurrent Neural Network) with extrema compression + sliding window preprocessor	Dow Jones Index (1990–2019), Chinese stock markets (9.9M records)	Proposed an RNN-based model enhanced with two-dimensional price-volume encoding and a trend-based extrema compression algorithm. Achieved faster convergence and better accuracy than RNN, ARIMA, and GARCH; performed nearly as well as LSTM with significantly less training time (e.g., error ≈ 9.23% vs. LSTM's 14.4%–16%). Ablation studies confirmed strongest gains came from the price-volume dimensionality upgrade.	Did not incorporate attention mechanisms, macroeconomic features, or sentiment; LSTM still outperformed TRNN at higher iteration counts; no live trading simulation or interpretability methods used

To simultaneously capture rapid intraday swings and enduring trends, we propose employing a hybrid CNN–LSTM–Attention architecture, where convolutional layers extract high-frequency, localized patterns, LSTMs encode longer-term dependencies, and an attention module focuses on the most informative time steps. Inspired by Zhang et al. (2023)'s ConvBiLSTM, our streamlined hybrid CNN–LSTM–Attention pipeline delivers end-to-end spatio-temporal forecasting, and we further advance it by integrating reinforcement-learning-driven online adaptation plus Elastic Weight Consolidation (EWC) to prevent catastrophic forgetting. To prove its robustness, we validate the model on an entirely different market than Zhang et al. (2023) used. Table 1 synthesizes leading DL approaches to stock market forecasting, highlights their principal limitations, and shows precisely how our proposed framework overcomes each of those gaps.

Materials and Methods

The model development will be implemented in phases following sequential steps as guided by the framework in Figure 1. The process begins by loading a dataset into a python environment and normalising it so that all input sources are consistent. Then comes a full feature extraction phase, which adds both technical indicators and macroeconomic variables to the input space. Model training uses a mix of optimisation methods, including EWC to stop catastrophic forgetting and reinforcement learning methods to help the model learn in a flexible way. Hyperparameter tuning is done by combining Bayesian optimisation and grid search to get the best predictive performance. The last step in the evaluation process uses quantitative metrics like Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) as well as feature importance analysis to see how well the model works and how easy it is to understand when it comes to predicting the future of finance.

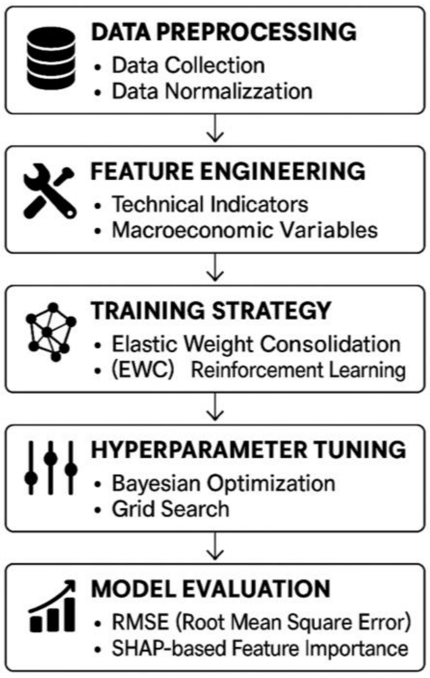


Fig. 1: Model Development Framework

DLLN Model Implementation Algorithm

The Lifelong Learning Stock Predictor (LLSP) algorithm is employed in the implementation of the DLLN model. The pseudocode for LLSP is presented below, accompanied by a concise description of its functionality.

Algorithm: *Lifelong Learning Stock Predictor (LLSP)*

```
Beginning the LLSP
Input: Historical data D, Feature set F, and hyperparameters H
Output: Trained DLLN model M
Step 1.  Initialize the model
1. Initialize the model M with hyperparameter H
Step 2.  Data Preprocessing
2. Preprocess stock market data
2.1      Normalize and scale data to ensure consistency
2.2      Handling missing values using interpolation
Step 3.  Feature Engineering
3. Enhance feature set F to improve predictive power
3.1      Extracting technical indicators (RSI, MACD)
3.2      Add macroeconomic variables (i.e., Interest rates, inflation, CPI)
Step 4.  Data Splitting
4. Split data D into training set T and validation set V
Step 5.  Model training
5. Train the DLLN model M on training set T:
5.1      For each epoch do:
5.1.1    Update weights via the Adam optimizer
5.1.2    Evaluate performance on the validation set V using MAE
Step 6.  Forgetting mitigation via Elastic Weight Consolidation (EWC)
6. Apply EWC to prevent catastrophic forgetting
6.1      Estimate parameter importance using the Fisher Information Matrix
6.2      Regularize updates by minimizing the modified loss function
Step 7.  Hyperparameter Tuning
7. Fine-tuning hyperparameters H via grid search to optimize performance
Step 8.  Reinforcement Learning Integration
8. Enhance adaptability by integrating reinforcement learning (RL)
8.1      Initializing RL agent A
8.2      Training RL agent A via Deep Q-Learning
8.3      Combining RL agent A with the DLLN model M to dynamically adjust learning rates and attention threshold.
Step 9.  Return Final Model
9. Return trained DLLN model M
End LLSP
```

Data Preprocessing

To train and test our model we used a comprehensive set of features. These include historical Open, High, Low, Closing and Volume (OHLCV) values, and Macroeconomic indicators like the Gross Domestic

Product (GDP), unemployment rate, real interest rates, and the Consumer Price Index (CPI). Finally, we also computed and incorporated two technical features, Relative Strength Index (RSI), and Moving Average Convergence Divergence (MACD). The MACD is lagged at three-time steps to increase the feature space.

Linear interpolation was applied to handle missing data where the gaps are small. For filling bigger gaps, we used forward and backward filling. We handled outliers by applying Winsorization technique. Min-Max scaling was used for normalization enabling the input distributions to remain balanced and the learning dynamics of the DLLN model to remain steady.

Hyperparameter Optimization

We use a dual-strategy technique that combines grid search with Bayesian optimization to improve hyperparameters systematically. Grid search enables the full testing of specified parameter grids, including ConvLSTM-specific settings such as (32, 64, 128), kernel size (1, 1), and activation functions (ReLU, Tanh). The same goes for dense layer units (25, 50, 100), batch sizes (16, 32, 64), and look-back windows (30, 60). We used the Bayesian optimisation to identify the best configurations faster. The Adam optimizer and Mean Absolute Error (MAE) is used for model training. These optimization methods work together to make dynamic financial forecasting settings more stable and accurate.

Integrating Attention Mechanism

To find the attention weights alpha_*t*, we applied the following equation:

$$\alpha_t = softmax(v^T \, tanh(W \cdot h_t + b)) \tag{1}$$

Where v is a learnable context vector. W is the weight matrix. h_t is the hidden state at time t and b is the bias term.

Elastic Weight Consolidation for Forgetting Mitigation

To handle catastrophic forgetting we will use Elastic Weight Consolidation (EWC) algorithm based on the Equation 2. EWC makes it difficult to change important weights for old tasks while allowing less important weights to change freely to learn the new task. If the calculated weight is greater than one, then the previous weight is discarded gracefully. If its less than one, the current weight is maintained:

$$L_{total} = L_{new(\theta)} + \frac{\gamma}{2}(\sum_{i=1}^{N_{params}} (F_i(\theta_i - \theta^*_{A,i})^2 \tag{2}$$

Where $L_{new(\theta)}$ represents the standard loss for the new task. γ is a hyperparameter that controls how important it is to remember the old task vs learning the new one. Σ goes over all the weights i to N in the network. θ_i represent the current value of weight i. $\theta^*_{A,i}$ represents the optimal value of weight i after learning Task A and F_i is the Fisher information weight i.

The Fisher Information is a calculated value that measures the importance of the weight i of the previous task. It's based on Equation 3:

$$F_i = E_{\{(x,y)\sim D_{AAPL}\}} \left[\left(\frac{\partial}{\partial \theta_i} log\, p(y\,|x\,,\theta^*) \right)^2 \right] \tag{3}$$

Where θ* represents the parameters optimized on the AAPL task, and F_i quantifies how much a small change in θ_i would affect the log-likelihood on that task.

EWC keeps AAPL-specific patterns by "locking" weights with large F_i values. This helps avoid catastrophic forgetting while allowing the model to learn from new data.

Adding Reinforcement Learning

Reinforcement learning is one of the important drivers of continual learning systems allowing an agent to adjust its strategy in non-stationary environments. We applied the Deep Q Network (DQN) in our framework to meta-learn optimal adaptation policies. The DQN agent works by observing a state representation such as the recent OHLCV values and the model's prediction error which serve as a proxy for market stability. Based on the observed state, the agent then select an appropriate set of actions from a discrete set, such as increasing the learning rate for rapid adaptation. Alternatively, the model could increase the attention thresholds to allow for focus on important features during volatile periods. This makes the model more sensitive to changing patterns by lowering the prediction loss function during times of high volatility through context aware hyperparameters. The Bellman equation (Equation 4) is used to update the Q-value. The agent gradually improves its approach over time by using experience replay and Q-learning to make sure that the best parameter changes are made:

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha \left[r_t + \gamma \, max\, Q(s_{\{t+1\}}, a) - Q(s_t, a_t)\right] \tag{4}$$

Where $Q(s_t, a_t)$ is the predicted reward for doing action a_t in state s_t. α is the learning rate. γ is the discount factor, and r_t is the reward right now.

Proposed DLLN Architecture

The final architecture of the proposed DLLN framework is shown in Figure 2. Several layers, including CNN, LSTM, and attention layers are systematically synthesized to predict future market price of a given asset.

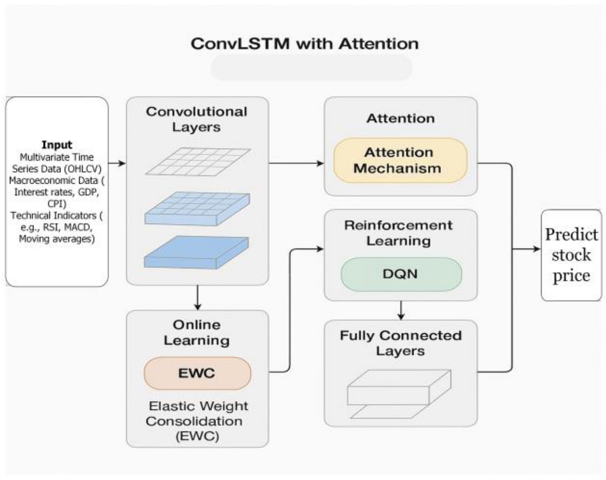
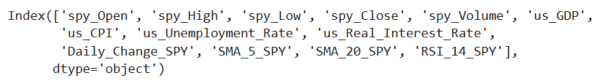

ConvLSTM with Attention
Input
Multivariate Time Series Data (OHLCV)
Macroeconomic Data (Interest rates, GDP, CPI)
Technical Indicators (e.g., RSI, MACD, Moving averages)
Convolutional Layers
Attention
Attention Mechanism
Reinforcement Learning
DQN
Online Learning
EWC
Elastic Weight Consolidation (EWC)
Fully Connected Layers
Predict stock price

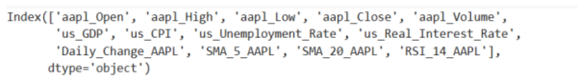
Fig. 2: Proposed Model architecture

Datasets



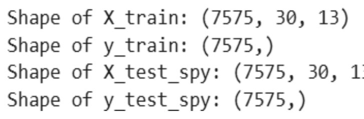
```
Index(['spy_Open', 'spy_High', 'spy_Low', 'spy_Close', 'spy_Volume', 'us_GDP',
       'us_CPI', 'us_Unemployment_Rate', 'us_Real_Interest_Rate',
       'Daily_Change_SPY', 'SMA_5_SPY', 'SMA_20_SPY', 'RSI_14_SPY'],
      dtype='object')
```

a) APPL features



```
Index(['aapl_Open', 'aapl_High', 'aapl_Low', 'aapl_Close', 'aapl_Volume',
       'us_GDP', 'us_CPI', 'us_Unemployment_Rate', 'us_Real_Interest_Rate',
       'Daily_Change_AAPL', 'SMA_5_AAPL', 'SMA_20_AAPL', 'RSI_14_AAPL'],
      dtype='object')
```

b) SPY features



```
Shape of X_train: (7575, 30, 13)
Shape of y_train: (7575,)
Shape of X_test_spy: (7575, 30, 13)
Shape of y_test_spy: (7575,)
```

c) Final tuned datasets

Fig. 3: Features ingested into the model

Trained on the AAPL dataset extracted from Yahoo Finance API, our model ingests various other data steams to make sure that the features are as comprehensive as possible. These features include the OHLCV values, macroeconomic indicators, and technical indicators as shown in Figure 3. The period of the data from 2002 to 2022 is selected to ensure that lived realities such as the 2008 global financial crisis and the COVID-19 pandemic era are modelled. The specific macroeconomic features extracted include unemployment rate, consumer price index, real interest rate, and gross domestic product extracted from the FRED API. We also add technical indicators like the Relative Strength Index and lagged aspects of moving averages at three-time steps that were calculated directly from the time-series data.

The model is further validated on the SPY ETF, a diversified proxy for the S&P 500 index. Both datasets are extracted from the US stock market. SPY dataset was specifically selected as it gives access to a wider and more varied selection of sectors and volatility profiles than the AAPL ticker. This cross-market validation provided critical insights into the model's adaptability to varied financial landscapes.

Evaluation Metrics

Several evaluation metrics, including the R-squared (R^2), Mean Absolute Error (MAE), Root Mean Square Error (RMSE) and mean absolute percentage error (MAPE), are used. These evaluation metrics are described below.

The MAE equation measures the average magnitude of the errors in a set of predictions without considering their direction. It is the average over the test sample, of the absolute differences between the prediction and actual observations where all individual differences have equal weights. The model with the lowest MAE is the better model:

$$MAE = \frac{1}{n}\sum_{i=1}^{n} \mid x_i - x \mid \tag{5}$$

Where:
n = The total number of data points
x_i = Predicted value
x = True value

The RMSE measures the square root of the average of the squared differences between the prediction and actual observations. This metric gives relatively high weight to large errors. The model with an RMSE closest to zero is considered to perform better:

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(x_i - x)^2}{n}} \tag{6}$$

Where:
n = The number of observed values
x_i = Predicted value
x = Observed value

The MAPE measures the average magnitude of the errors as a percentage of the true values, providing a scale-independent measure:

$$\mathrm{MAPE} = \frac{1}{\mathrm{n}}\sum_{\mathrm{i}=1}^{\mathrm{n}} \left| \frac{\mathrm{x_i} - \mathrm{x}}{\mathrm{x_i}} \right| \tag{7}$$

Where:
n = Number of observed values
x_i = Predicted value
x = Observed value

The MAPE value can be expressed as a percentage by multiplying the result by 100% with values that are closer to 0%, indicating greater accuracy of the model (De Myttenaere et al., 2016).

R^2 is a measure of the proportion of the variance in the dependent variable that is predictable from the independent variable (s) (feature (s)) via the regression model. The R^2 values range from 0 to 1, where 0 indicates that the model does not explain any of the variance in the target variable and 1 indicates that the model explains all the variance in the target variable. R^2 can be expressed as a percentage by multiplying by 100%:

$$R^2 = 1 - \frac{\Sigma(y_i - \hat{y}_i)^2}{\Sigma(y_i - \bar{y}_i)^2} \tag{8}$$

Where:

y_i = Actual value
$\hat{y}_i$ = Predicted value
$\bar{\boldsymbol{y}}_i$ = Mean value of y

The DLLN model is also compared to several baseline deep learning architectures identified in literature. These are the LSTM, CNN, BiLSTM, and GRU-BiLSTM. An ablation study is done to see how each part of the hybrid structure improves the overall performance of the model.

The study employs a diverse array of Python packages to facilitate data preparation, model training, evaluation, and visualization. Libraries used include Pandas, NumPy, Scikit-learn, Scikeras, Matplotlib, and Plotly. PyTorch, is used to create the EWC and RL components.

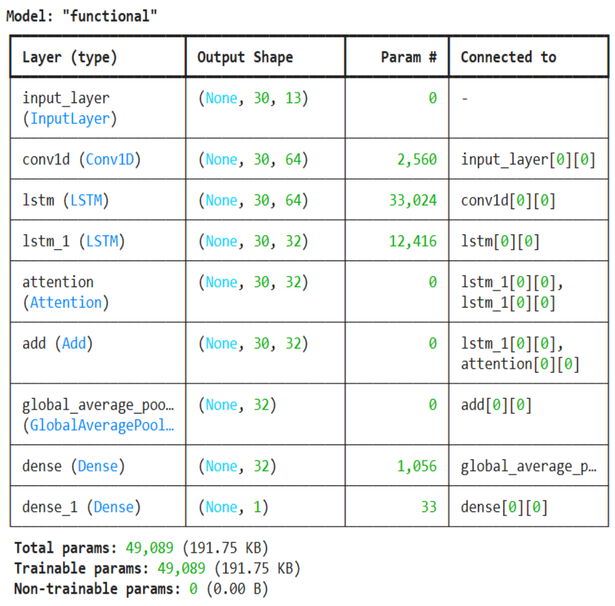
Results and Discussion

We implemented our model using Python in the Google Colab environment as shown in Figure 4. The model takes input sequences that are in the shape (30, 13). This means that each sample has 30-time steps and 13 characteristics. There is only one 1D CNN layer in the design, which has 64 filters and a kernel size of 3. This layer captures local temporal patterns in the input data. A ReLU activation function adds nonlinearity to make the model more expressive, and a 30% dropout rate is used to keep it from overfitting.

The BiLSTM layer with 256 units follows the CNN layer. It is set up with return_sequences=True so that the whole sequence output may be used for attention processing later on. A dropout of 30% is again used to make training more regular. Then, a custom attention mechanism is used on the output of the BiLSTM. This layer gives learnt weights to distinct time steps. This property improves the model's ability to focus on salient segments of the input sequence. The mechanism has trainable weight parameters (W, b), and a tanh activation function that gives alignment scores and a SoftMax function that gives the normalised attention weights (α). The resulting weighted context vector summarizes the sequence for final prediction.

The output layer is a dense layer of 5 units, corresponding to the forecast horizon (FORECAST_LENGTH) of 5 steps. The model is trained with the Adam optimizer, chosen for its adaptive learning capability and the MAE loss function to ensure precise regression performance.

Figure 5 shows the resultant Loss and MAE plot run over 50 epochs. As depicted in the plot, both the MAE curve and the combined loss shows a gradual descent with an increase in epochs. This indicates improved predictive accuracy as training progresses and effective model optimization. The convergence of both metrics demonstrates that the model balances learning stability with adaptability thereby mitigating catastrophic forgetting while benefiting from dynamic reward-guided fine-tuning.



Model: "functional"

Layer (type)	Output Shape	Param #	Connected to
input_layer (InputLayer)	(None, 30, 13)	0	-
conv1d (Conv1D)	(None, 30, 64)	2,560	input_layer[0][0]
lstm (LSTM)	(None, 30, 64)	33,024	conv1d[0][0]
lstm_1 (LSTM)	(None, 30, 32)	12,416	lstm[0][0]
attention (Attention)	(None, 30, 32)	0	lstm_1[0][0], lstm_1[0][0]
add (Add)	(None, 30, 32)	0	lstm_1[0][0], attention[0][0]
global_average_poo… (GlobalAveragePool…	(None, 32)	0	add[0][0]
dense (Dense)	(None, 32)	1,056	global_average_p…
dense_1 (Dense)	(None, 1)	33	dense[0][0]

Total params: 49,089 (191.75 KB)
Trainable params: 49,089 (191.75 KB)
Non-trainable params: 0 (0.00 B)

Fig. 4: Baseline model architecture overview

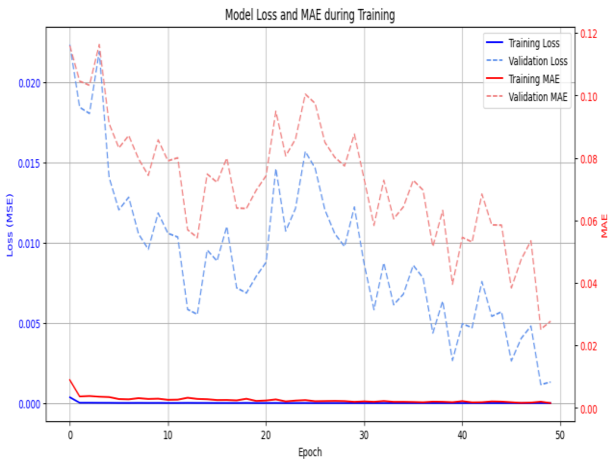


Fig. 5: DLLN loss function plot

Table 2 summarizes the hyperparameter tuning results. The optimal setup used a convolutional filter size of 32, 32 units in the first LSTM layer, 16 units in the second LSTM layer, a learning rate of 0.01, and a batch size of 64. This configuration has the lowest validation loss of 0.005 and the lowest MAE of 0.047. It also showed the most efficient training profile, taking only 132.1 seconds to compute. Given these results, this best setup was made the default for all benchmark models so that they would be consistent and easy to compare.

Model Generalizability and Robustness

To explore the benefits of each additional module in our model, an ablation study was conducted. Here, the model was tested in three configurations, utilizing SPY dataset. The basic setup used the ConvLSTM-Attention architecture. In the second setup, EWC was included to stop catastrophic forgetting. Finally, a reinforcement learning module was included. We compared the performance measures Loss, MAE, RMSE, and R-squared for all three variations and included them in Table 3. Adding the EWC part improved performance by 7.72% and adding the RL part improved it by 4.63% more. All in all, both components led to a 12.35% improvement in performance. These findings confirm our earlier hypothesis that adding both EWC and RL modules improves the model's predictive performance, its stability and generalization capabilities.

A good predictive model's trend line should follow that of the actual prize as closely as possible even when exposed to a different dataset other than the one it was trained on. To further assert our claims of the contributions of each additional component of our model, we plotted the closing against predicted price over our test dataset as shown in Figure 6. The figure shows normalized values of the closing price (y-axis) over a 20-year period (x-axis) running against three variations of our model. The DLLN model's EWC and RL modules facilitated the model to perform well and to adaptively learn from evolving patterns as evidenced in Figure 6. The figure, especially (c), with all the components added demonstrates strong alignment between predicted and actual SPY closing prices as most key trend structures and turning points were captured. Of note is the ability of the model to capture the 2008 recession precisely as can be observed between time steps 2000 to 3000.

However, despite its earlier robust performance, the model exhibited limitations during the COVID-19 period (time steps 5000 to 7000) where extreme volatility was experienced. This means that the model was unable to respond to this particular black swarm event. Our assumption on what might have contributed to this shortcoming is that the response by the market to the COVID-19 pandemic had nothing to do with the structured data which we used to train our model. Given that COVID-19 triggered unprecedented panic and uncertainty, largely communicated through news outlets and social media, the model's purely quantitative design lacked access to timely psychological signals that often precede market movements. Without real-time sentiment input we observe that the DLLN failed to fully recognize the severity and timing of the downturn.

To test the significance of our initial hypothesis that adding additional components such as the EWC and RL leads to a better performing model, we applied two statistical tests, the Diebold-Mariano test and the paired t-test. The DM test evaluates whether the forecast error distributions of two models differ significantly. Negative values indicate superior predictive accuracy from the second model. The paired t-test on the other hand assesses whether the mean difference in prediction errors is statistically distinguishable from zero. Results of the two tests are shown in Table 4. Both tests confirm that the additional EWC and RL modules significantly improved the model's predictive performance.

Table 2: Grid search results for Hyperparameter settings

Conv_filter	LSTMunit1	LSTMunit2	Learning rate	Batch size	Value loss	ValueMAE	Time
32	32	16	0.010	32	0.037	0.142	184.4
32	32	16	0.010	32	0.005	0.045	286.9
32	32	16	0.010	64	0.005	0.047	132.1
32	32	16	0.010	32	0.045	0.163	173.6
32	32	16	0.001	32	0.018	0.098	188.8

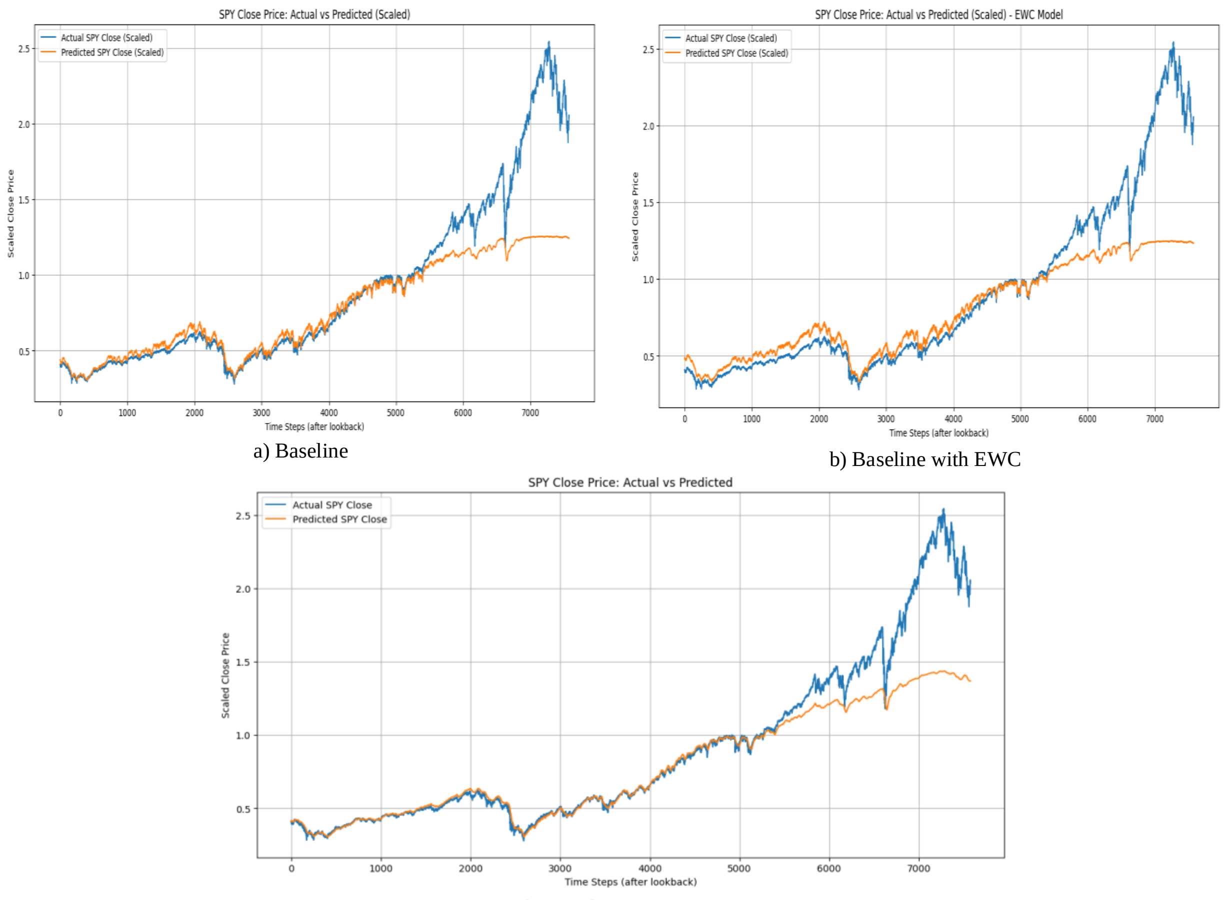
Table 3: Performance Comparison on SPY Test Set

	Metric	Baseline model	EWC model	RL-EWC Model
1	Loss (MSE)	0.1085±0.0064	0.0837±0.0049	0.0635±0.0038
2	MAE	0.1758±0.0112	0.1279±0.0076	0.1021±0.0060
3	R-Squared	0.6618±0.0215	0.7390±0.0173	0.7853±0.0147
4	RMSE	0.3285±0.0099	0.2886±0.0082	0.2241±0.0069

Table 4: Statistical Significance Testing

	Diebold-Mariano test		Paired t-test	
	DM statistic	P Value	T-Statistic	P value
Baseline vs EWC	-2.15	0.032*	2.47	0.019*
Baseline vs RL	-2.78	0.007**	2.93	0.005**
EWC vs RL-EWC	-1.95	0.051*	2.11	0.042*

*P<0.05; **P<0.01



a) Baseline

b) Baseline with EWC

c) Baseline with RL-EWC (DLLN)

Fig. 6: DLLN Variants Forecasting performance on SPY

Table 5: Performance evaluation of comparable models over the Apple ticker

	MAE	RMSE	R2	MAPE
LSTM	3.98±0.31	5.07±0.43	0.919±0.10	13.79±0.88
CNN	8.23±0.50	9.92±0.72	0.788±0.12	20.69±1.31
Bi-LSTM	12.48±0.66	13.64±0.81	0.600±0.21	21.54±1.48
GRU-BILSTM	13.71±0.58	14.52±0.69	0.547±0.31	22.02±1.52
DLLN	3.22±0.27	3.95±0.32	0.967±0.01	22.55±1.74

Performance Evaluation of the DLLN Against Comparative Models

In our second hypothesis, we claimed that our model significantly outperforms other benchmarks identified in literature. We present the findings of the DLLN against several benchmark architectures identified in literature such as LSTM (Lu and Xu, 2024), CNN (Oyewole et al., 2024), Bi-LSTM (Lee et al., 2024) and the GRU-LSTM (Shaban et al., 2024) in Table 5. We evaluated the performance of these models based on our earlier described evaluation framework including MAE, RMSE, R-squared, and MAPE. As illustrated in Table 5, the DLLN model consistently outperformed its counterparts across most metrics. The model achieved a notably high R^2 value of 0.967 and a very low MAE of 3.22 when applied to the Apple ticker dataset. These findings confirm that the DLLN model's predictive capacity is superior to the other baselines.

It is common practice to measure performance of a model against time complexity (Zhou et al., 2021). This metric is often evaluated based on the time it takes to execute the model. In terms of this measure, the DLLN proved to be efficient, requiring only 45.56 seconds to train, making it comparable only to the smaller CNN model. In a domain that requires quick data processing for decision making, such as financial markets prediction, the DLLN has shown to be suitable due to its efficiency and effectiveness. This is achieved through the model's ability to learn continuously enabling it to adapt to sudden changes in trends.

Figure 7 shows the relationship between actual stock prices and outputs from five forecasting models: CNN, LSTM, BiLSTM, Bidirectional GRU, and the proposed DLLN architecture. To ensure a fair comparison, optimum hyperparameter for each model are first scanned using the grid search and Bayesian search optimization algorithms. The y axis represents the price, and the x-axis represents the time steps over a 20-year period. The training vs testing dataset was split on a 60:40 ratio. We used the Apple dataset extracted from Yahoo Finance as articulated in our methodology. Among these, the DLLN model, shown in brown, consistently follows the actual stock price path (blue) more accurately than the others, especially during times of high volatility.

The trend line for the DLLN model shows closer depiction to the actual prices even capturing quick spikes and declines that other models smooth out. This confirms that the model is efficient in adapting to rapid market changes. These results further support our second hypothesis that our model outperforms other industrial standards identified in literature. This ability is important for making accurate financial predictions in the real world. The architecture of the DLLN, which combines CNN, LSTM, Attention network, EWC and RL allows the model to be aware of short-term changes while maintaining an understanding of larger trend patterns. The model shows a clear ability to track near trend inflexion points, exhibiting their real-time reliability.

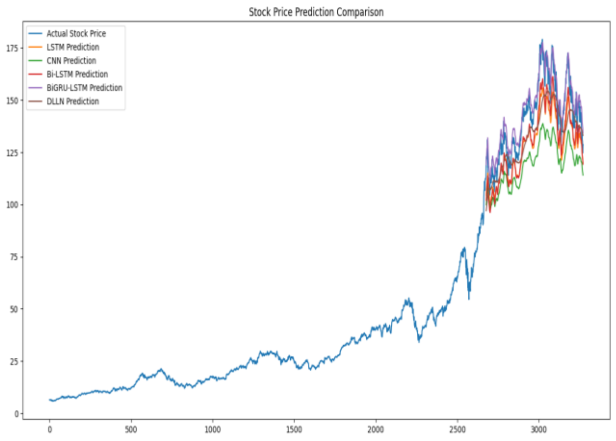


Fig. 7: Training and testing models over the Apple ticker

Conclusion and Future Study

In this paper, we introduced the dynamic lifelong learning network for stock market price forecasting. The neural network is built by combining a convolutional neural network with a long short-term memory network. An attention mechanism is also integrated to assist the model to focus on more critical time steps in the input sequence thereby improving accuracy and computational efficiency. Lifelong learning capability was specifically implemented to avoid expensive model retraining in changing data trends unlike the standard batch training approach used in most machine learning models. This flexibility is achieved by using a reinforcement learning (Deep Q Network) implemented through our novel LLSP algorithm. The capability is further enhanced by implementing Elastic Weight Consolidation which facilitates the model to retain vital knowledge from past tasks. This strategy reduces the risk of catastrophic forgetting a major problem in DL. The novel hybrid architecture outperformed several baselines identified from literature.

While our model performed quite well there remain some limitations that future research should address. For a start, our model was based on purely structured data like OHLCV values. The model could be enhanced by integrating even unstructured data like news and social media sentiments. Using NLP techniques could improve the model's understanding of context, though it needs careful adjustment of specific sentiment models to prevent misclassification or bias.

The other limitation is the reliance of the model on data extracted from the same market. A cross-market study, exploring the financial markets from different regions than the S&P 500 could possibly assist in verifying the generalizability of the model to different market regimes. Ultimately, the results underscore the value of continual, context-sensitive learning in financial modelling. This offers a blueprint for systems that evolve in sync with volatile markets.

Acknowledgment

We are very grateful to all contributors who gave useful insights into the research. Their input greatly enhanced the depth and breadth of this research. We also appreciate the editorial team for their thorough review and constructive feedback which significantly improved the quality and clarity of the manuscript.

Funding Information

No funding was received for this research.

Author's Contributions

Talent Mawere: Conceptualized, designed, and implemented the DLLN. He executed all technical experiments including development of the processing pipeline and the LLSP algorithm. Talent drafted the initial manuscript.

Selvaraj Rajalakshmi: Provided strategic oversight and critical guidance on the research framework. She reviewed and revised the manuscript for theoretical accuracy. She also contributed to optimizing real-time systems and enhancing computational efficiency.

Venu Madhav Kuthadi: Provided oversight over the methodological design including feature engineering, evaluation metrics and statistical assessment. He advised on data selection, and benchmarking against baseline models.

Otlhapile Dinakenyane: Supported the technical validation and strategies for implementation. She reviewed visual presentations and manuscript structure to ensure clarity and compliance with academic standards.

Ethics

No ethical issues are anticipated to arise after the publication of this manuscript.

Data Availability Statement

Data used for this research is available via FRED and Yahoo Finance APIs. The code can be obtained from the corresponding author upon reasonable request.

References

Adesina, A. A., Vanessa Iyelolu, T., & Okpeke Paul, P. (2024). Optimizing Business Processes with Advanced Analytics: Techniques for Efficiency and Productivity Improvement. *World Journal of Advanced Research and Reviews*, *22*(3), 1917–1926. https://doi.org/10.30574/wjarr.2024.22.3.1960

Ayyildiz, N., & Iskenderoglu, O. (2024). How effective is machine learning in stock market predictions? *Heliyon*, *10*(2), e24123. https://doi.org/10.1016/j.heliyon.2024.e24123

Bahoo, S., Cucculelli, M., Goga, X., & Mondolo, J. (2024). Artificial intelligence in Finance: a comprehensive review through bibliometric and content analysis. *SN Business & Economics*, *4*(2), 23. https://doi.org/10.1007/s43546-023-00618-x

Berti, L., & Kasneci, G. (2025). TLOB: A Novel Transformer Model with Dual Attention for Stock Price Trend Prediction with Limit Order Book Data. *Statistical Finance*. https://doi.org/10.48550/arXiv.2502.15757

Chatziloizos, G.-M., Gunopulos, D., & Konstantinou, K. (2024). Deep Learning for Stock Market Prediction Using Sentiment and Technical Analysis. *SN Computer Science*, *5*(5), 446. https://doi.org/10.1007/s42979-024-02651-5

Chauhan, J. K., Ahmed, T., & Sinha, A. (2025). A novel deep learning model for stock market prediction using a sentiment analysis system from authoritative financial website’s data. *Connection Science*, *37*(1), 2455070. https://doi.org/10.1080/09540091.2025.2455070

Chen, Y., Wu, J., & Wu, Z. (2022). China’s commercial bank stock price prediction using a novel K-means-LSTM hybrid approach. *Expert Systems with Applications*, *202*, 117370. https://doi.org/10.1016/j.eswa.2022.117370

Dang, N. C., Moreno-García, M. N., & De la Prieta, F. (2020). Sentiment Analysis Based on Deep Learning: A Comparative Study. *Electronics*, *9*(3), 483. https://doi.org/10.3390/electronics9030483

De Lange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G., & Tuytelaars, T. (2021). A continual learning survey: Defying forgetting in classification tasks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 44(7), 3366-3385. https://doi.org/10.1109/tpami.2021.3057446

de Myttenaere, A., Golden, B., Le Grand, B., & Rossi, F. (2016). Mean Absolute Percentage Error for regression models. *Neurocomputing*, *192*, 38–48. https://doi.org/10.1016/j.neucom.2015.12.114

Deng, C., Huang, Y., Hasan, N., & Bao, Y. (2022). Multi-step-ahead stock price index forecasting using long short-term memory model with multivariate empirical mode decomposition. *Information Sciences*, *607*, 297-321. https://doi.org/10.1016/j.ins.2022.05.088

Fejerskov, A. M. (2017). The New Technopolitics of Development and the Global South as a Laboratory of Technological Experimentation. *Science, Technology, & Human Values*, *42*(5), 947–968. https://doi.org/10.1177/0162243917709934

Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work. *The journal of Finance*, *25*(2), 383-417.

Ghosh, B. P., Bhuiyan, M. S., Das, D., Nguyen, T. N., Jewel, R. M., Mia, M. T., ... & Shahid, R. (2024). Deep learning in stock market forecasting: Comparative analysis of neural network architectures across NSE and NYSE. *Journal of Computer Science and Technology Studies*, *6*(1), 68-75. https://doi.org/10.32996/jcsts.2024.6.1.8

Hossain, E., Shariff, M. A. U., Hossain, M. S., & Andersson, K. (2021). A Novel Deep Learning Approach to Predict Air Quality Index. *Advances in Intelligent Systems and Computing*, *194*, 367–381. https://doi.org/10.1007/978-981-33-4673-4_29

Jin, Y. (2024). GraphCNNpred: A stock market indices prediction using a Graph based deep learning system. *Proceedings of the 2024 2nd International Conference on Artificial Intelligence, Systems and Network Security*, 170–178. https://doi.org/10.1145/3714334.3714364

Jintong, S., Qishuo, C., Bai, X., Wei, J., & Guangze, S. (2024). LSTM-Based Deep Learning Model for Financial Market Stock Price Prediction. *Journal of Economic Theory and Business Management*, *1*(2). https://doi.org/https://doi.org/10.5281/zenodo.10940654

Khatwani, R., Mishra, M., Bedarkar, M., Nair, K., & Mistry, J. (2023). Impact of Blockchain on Financial Technology Innovation in the Banking, Financial Services and Insurance (BFSI) Sector. *Journal of Statistics Applications & Probability*, *12*(1), 181–189. https://doi.org/10.18576/jsap/120117

Kuo, R. J., & Chiu, T.-H. (2024). Hybrid of jellyfish and particle swarm optimization algorithm-based support vector machine for stock market trend prediction. *Applied Soft Computing*, *154*, 111394. https://doi.org/10.1016/j.asoc.2024.111394

Kudithipudi, D., Aguilar-Simon, M., Babb, J., Bazhenov, M., Blackiston, D., Bongard, J., ... & Siegelmann, H. (2022). Biological underpinnings for lifelong learning machines. *Nature Machine Intelligence*, *4*(3), 196-210. https://doi.org/10.1038/s42256-022-00452-0

Lee, H., Kim, J. H., & Jung, H. S. (2024). Deep-learning-based stock market prediction incorporating ESG sentiment and technical indicators. *Scientific Reports*, *14*(1), 10262. https://doi.org/10.1038/s41598-024-61106-2

Li, A. W., & Bastos, G. S. (2020). Stock Market Forecasting Using Deep Learning and Technical Analysis: A Systematic Review. *IEEE Access*, *8*, 185232–185242. https://doi.org/10.1109/access.2020.3030226

Liao, C., Chen, G., & Cai, S. (2024). Stock Market Volatility Prediction Based on Robust GBM-GRU Model. *Fluctuation and Noise Letters*, *23*(03), 2450021. https://doi.org/10.1142/s0219477524500214

Lu, M., & Xu, X. (2024). TRNN: An efficient time-series recurrent neural network for stock price prediction. *Information Sciences*, *657*, 119951. https://doi.org/10.1016/j.ins.2023.119951

Malkiel, B. G. (1973).A Random Walk Down Wall Street. New York: W. W. Norton 8c Co.

Nejad, F. S., & Ebadzadeh, M. M. (2024). Stock market forecasting using DRAGAN and feature matching. *Expert Systems with Applications*, *244*, 122952. https://doi.org/10.1016/j.eswa.2023.122952

Oğuz, A., & Ertuğrul, Ö. F. (2023). Introduction to deep learning and diagnosis in medicine. *Deep Learning Applications for Medical Diagnosis*, 1–40. https://doi.org/10.1016/b978-0-323-96129-5.00003-2

Oyewole, A. T., Bukola Adeoye, O., Afua Addy, W., Chinazo Okoye, C., Chrisanctus Ofodile, O., & Esther Ugochukwu, C. (2024). Predicting Stock Market Movements Using Neural Networks: A Review and Application Study. *Computer Science & IT Research Journal*, *5*(3), 651–670. https://doi.org/10.51594/csitrj.v5i3.912

Pattnaik, D., Ray, S., & Raman, R. (2024). Applications of artificial intelligence and machine learning in the financial services industry: A bibliometric review. *Heliyon*, *10*(1), e23492. https://doi.org/10.1016/j.heliyon.2023.e23492

Qin, W. (2024). Predictive Analysis of AAPL Stock Trend by Random Forest and K-NN Classifier. *Highlights in Business, Economics and Management*, *24*, 1418–1422. https://doi.org/10.54097/pgh78d83

Shaban, W. M., Ashraf, E., & Slama, A. E. (2024). SMP-DL: a novel stock market prediction approach based on deep learning for effective trend forecasting. *Neural Computing and Applications*, *36*(4), 1849–1873. https://doi.org/10.1007/s00521-023-09179-4

Sharma, U., Kumar, A., Kumar, R., Manchanda, M., & Hooda, S. (2025). Netflix stock market price prediction. *AIP Conference Proceedings*, 020061. https://doi.org/10.1063/5.0248341

Tan, J. (2024). NVIDIA Stock Price Prediction by Machine Learning. *Highlights in Business, Economics and Management*, *24*, 1072–1076. https://doi.org/10.54097/dsz8ns50

Wang, H., Liu, D., Zhang, J., Wang, J., Ma, Y., & Lian, Y. (2024a). Log-periodic power law hybrid model based on BP neural network. *Evolutionary Intelligence*, *17*(1), 123–131. https://doi.org/10.1007/s12065-020-00552-z

Wang, J., Chen, J., Zhang, K., & Sigal, L. (2024b). Training feedforward neural nets in Hopfield-energy-based configuration: A two-step approach. *Pattern Recognition*, *145*, 109954. https://doi.org/10.1016/j.patcog.2023.109954

Wang, J., Hong, S., Dong, Y., Li, Z., & Hu, J. (2024c). Predicting Stock Market Trends Using LSTM Networks: Overcoming RNN Limitations for Improved Financial Forecasting. *Journal of Computer Science and Software Applications*, *4*(3), 1–7. https://doi.org/https://doi.org/10.5281/zenodo.12200708

Wang, K. (2025). Multifactor prediction model for stock market analysis based on deep learning techniques. *Scientific Reports*, *15*(1), 5121. https://doi.org/10.1038/s41598-025-88734-6

Wang, P.-C., & Vo, T. T. H. (2025). Stock price prediction based on dual important indicators using ARIMAX: A case study in Vietnam. *Journal of Intelligent Systems*, *34*(1), 20240101. https://doi.org/10.1515/jisys-2024-0101

Wei, Y., Gu, X., Feng, Z., Li, Z., & Sun, M. (2024). Feature Extraction and Model Optimization of Deep Learning in Stock Market Prediction. *Journal of Computer Technology and Software*, *3*(4). https://doi.org/https://doi.org/10.5281/zenodo.13622489

Wu, S., Liu, Y., Zou, Z., & Weng, T.-H. (2022). S_I_LSTM: stock price prediction based on multiple data sources and sentiment analysis. *Connection Science*, *34*(1), 44–62. https://doi.org/10.1080/09540091.2021.1940101

Ying, Z., Cheng, D., Chen, C., Li, X., Zhu, P., Luo, Y., & Liang, Y. (2024). Predicting stock market trends with self-supervised learning. *Neurocomputing*, *568*, 127033. https://doi.org/10.1016/j.neucom.2023.127033

Zhang, J., Ye, L., & Lai, Y. (2023). Stock Price Prediction Using CNN-BiLSTM-Attention Model. *Mathematics*, *11*(9), 1985. https://doi.org/10.3390/math11091985

Zheng, J., Xin, D., Cheng, Q., Tian, M., & Yang, L. (2024). *The Random Forest Model for analyzing and Forecasting the US Stock Market under the background of smart finance*. *62,* 82–90. https://doi.org/10.2991/978-94-6463-419-8_11