National College of Ireland

Enhancing CNN-LSTM Models for Financial Time-Series Forecasting through Hyperparameter Optimization and Regularization

MSc Research Project
MSC DATA ANALYTICS

Rishabh Saudagar
Student ID: X23288914

School of Computing
National College of Ireland

Supervisor: Sallar Khan

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Rishabh Saudagar
Student ID:	X23288914
Programme:	MSC Data Analytics **Year:** 2024-2025
Module:	MSC Data Analytics Research Project
Supervisor:	Sallar Khan
Submission Due Date:	11/08/2025
Project Title:	Enhancing CNN-LSTM Models for Financial Time-Series Forecasting through Hyperparameter Optimization and Regularization
Word Count:	6478 **Page Count** 21

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Rishabh Deepak Saudagar
Date:	11/08/2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Enhancing CNN-LSTM Model for Financial Time-Series Forecasting through Hyperparameter Optimization and Regularization

Rishabh Deepak Saudagar
X23288914

Abstract

The work in this paper explores the enhancements of the CNN-LSTM hybrid model of predicting the closing prices of the following days of three financial indices, the NSEI of India, the ISEQ of Ireland, and the S&P 500 of America. The focus is on the improvement of the prediction accuracy and generalization through the combination of hyperparameters optimization and over-fitting alleviation approaches. A decade of OHLCV data was used to construct technical indicators such as moving averages, RSI, MACD and volatility on each having a lookback period of 60 days. Each market RMSE and MAE were the main evaluations used to train and test the baseline CNN-LSTM model separately per market. The later hyperparameter optimization with the Keras Tuner showed quite modest results. Comparative performance in terms of markets was explained through visualizations, such as plots of real vs anticipated and bar charts of RMSE/MAE. The USA model performed better and this shows the relevance of the approach in mature markets. Forecasts and results were exported to be analysed within Power BI. The project offers the replicable model of financial forecasting with deep learning.

Keywords: CNN-LSTM, Financial Forecasting, Hyperparameter Optimization, Overfitting, Deep Learning

1. Introduction

The combination of spatial patterns and temporal connection is one of the reasons why deep learning models, in particular, the hybrid advances, like CNN-LSTM, have become widely employed in financial forecasting. The CNN layers are also quite useful in finding localized patterns based on financial indicators whereas LSTM layers are the best at handling sequences of dependencies, which makes this combination best suited towards time-series. (Cheema & Moon, 2022)

Despite the theoretical capabilities, the CNN-LSTM systems have a weak generalization to their theoretical advantages and require careful hyperparameters adjustments to ensure good generalization especially in the unpredictable and nonlinear world of the stock markets. This paper attempts to stabilize and improve the accuracy of CNN-LSTM by systematic hyperparameter tuning and regularization techniques. (Takuya Akiba, 2019) (Nitish Srivastava, 2014)

Its focus is the prediction of the next day closing price of three common stock index, NSEI (India), ISEQ (Ireland) and S&P 500 (USA). The regular modeling process was adopted over all the databases with technical indicators and past data followed by optimization and evaluation. The final results were displayed and exported in order to be later used in business intelligence tools. (Vantage, 2017) (Microsoft, 2023)

Research Questions:

1. How can hyperparameter optimization improve the stability and predictive performance of CNN-LSTM models in financial time-series forecasting?

2. To what extent do the regularization procedures help minimize the tendency of overfitting and increase generalization in CNN-LSTM models in the case of using them on the financial data?

Paper Structure:

The later parts of this report are structured in the following manner. In section 2, a literature review is provided which highlights the use of CNN-LSTM models to make financial predictions but addresses the problem of overfitting. The third section outlines data-gathering methodology, the feature engineering processes, and the preparation of sequences that were used. A description of the architecture of the proposed model and specification of the key design choices during its formulation are outlined in section 4. Section 5 talks of the performance of the model and the training technique of the model. The measurements, results of the experiments, and graphs are described in part 6 accordingly. Section 7 finishes with conclusions overall results of research on the main findings and the directions to further research.

2. Related Work

The realization that, financial time-series forecasting is relevant to the algorithmic trade, risk management and portfolio optimization has always elicited interests. Traditional ways of doing things such as ARIMA, GARCH, and SVMs dominated the past, but now with deep learning available it has provided a set of strong models capable of capturing the non-linearity and long-term dependence in financial data. The CNN-LSTM hybrid model is trendy due to its architectural dominance against capturing the spatio-temporal dynamics. (Neha Solanki, 2024) (Jin, et al., 2023)

Sezer et al. (2020) reported a comprehensive literature review of deep learning in financial prediction, noting the growing importance of such models as the CNN-LSTM and leaving behind those small networks with single layers. The same authors (Widiputra et al., 2021) then expanded on this idea by using a multivariate CNN-LSTM model to simultaneously classify multiple financial indexes and showed the higher accuracy of the model in comparison to baseline methods. (Ahoora Rostamian, 2022) applied the CNN-LSTM model within the directional change framework and demonstrated its superior predictive capabilities in terms of turbulent market conditions. (Jin, et al., 2023)

There is an abundance of modifications proposed to remedy the pervading problems of overfitting and hyperparameterity in CNN-LSTM models. To illustrate the point, (Jimmy Ming-Tai, 2021) offered a graph-based CNN-LSTM framework based on the use of leading indicators and found that it showed positive results in terms of an improved performance curve in all the financial datasets it was tested with. (Hui Liu, 2020) incorporated the attention and memory methodology to CNN-STLSTM-AM architectures, and this led to a great improvement in predicting exchange rates. Feeling that the Bidirectionality reveal the hidden complex connections, (Neha Solanki, 2024) applied the CNN-BiLSTM architecture to a decade of stock data. (Takuya Akiba, 2019)

Tweaking hyperparameter techniques are very crucial in improving the generalizability of a model. Other examples of research provided by (Takuya Akiba, 2019) and Bergstra et al. (2011) present the available automated search architectures like Optuna and Bayesian Optimization that proves to be efficient in navigating large parameter spaces with the goal of optimizing the model performance. There have been significant improvements in validation measures with the application of these tuners to CNN-LSTM architectures.

In the area of financial forecasting, overfitting is a recurrent problem. This phenomenon is addressed by a set of methods, the main ones of which are dropout (Nitish Srivastava, 2014), batch normalization (Sergey Ioffe, 2015), and early halting (Han, 2016). Such interventions increase model robustness and lead to an increase in high-quality generalization, especially in situational settings that tend towards volatility. (Sidra Mehtab, 2020) in their research proved that CNN-LSTM the ensemble can be reinforced with dropout layer modalities and thus prove to be more robust to such high-frequency signal noise. Here, complementary preprocessing methods are involved in calculating log returns, moving averages, and RSI value indicators, which (Brownlee, 2016), along with (Hui Liu, 2020), approves of.

The empirical results confirm that CNN-LSTM models have predictive power in relation to financial forecasting and at the same time address the need of having customized regularization processes that can avert their inherent shortcomings. Making use of optimized procedures in the construction of the model, feature engineering procedures, and hyperparameter optimization practices, the present research builds on a replicable setup flexible to embrace cross-market predictions. The following sections are divided into the following blocks. In section 2, a literature review and survey are given with an emphasis on CNN-LSTM models as applied to financial forecasting and on the measures taken to overcome the overfitting problem. Section 3 describes the data-collection strategy, feature-engineering and sequence-preparation techniques used. In section 4, the architecture of the proposed model is explained along with design choices that were made throughout its creation. The execution and training methodology of its model is provided in section 5. Section 6 shows the metrics of evaluation, experimental results, and related plots. In section 7, the main results are summarized and avenues where further research can be carried out are defined.

3. Research Methodology

The works apply a repeatable and systematic procedure to predict the end-of-business values of three of the most popular stock indices globally, which are the Indian NSEI, Ireland ISEQ, and the S&P 500 in the USA, the prices of the indices at the close of trading the following day. The process is structured into six consecutive steps, namely data collecting and pre-processing, feature engineering, sequence formation, model building, model testing, and final selection. The steps are performed in the same way across all markets in order to be comparable and replicable. (Anon., n.d.)

3.1Data Acquisition and Cleaning

The given research starts with the acquisition of the ten-year span (about 2500 trading days) of daily OHLCV (Open, High, Low, Close, Volume, Adjusted Close) information on all of these indices with the help of the Python package "***yfinance***". Archiving The data was archived in comma-separated value (CSV) files in order to encourage transparency and reproducibility. All CSVs were then loaded into a Pandas DataFrame with the 'Date' column converted to the "***datetime***" type and turned into the index. The other columns were normalized to a same schema "*[Adj Close], [Close], [High], [Low], [Open], [Volume]*". Even though the percentage of missing values was dominant, the found values were removed so that no anomalies appear when the model training was conducted. This process led to the acquisition of clear structured data of the three respective marketplaces.

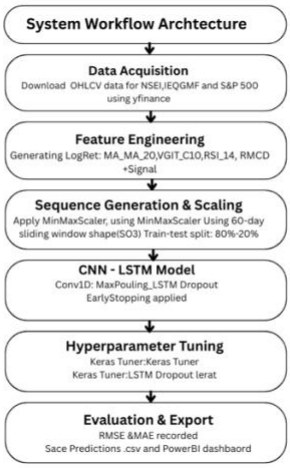

Figure 1: System Workflow Architecture Diagram Here

3.2 Feature Engineering

The second stage focused on generating predictive financial features from the raw data. The following indicators were calculated for each market:

- **Log Returns (LogRet):** Computed as the natural logarithm of the ratio between consecutive closing prices.
- **Moving Averages:** 5-day (MA_5) and 20-day (MA_20) simple moving averages to capture short- and medium-term trends.
- **Volatility (Vol_10):** 10-day rolling standard deviation of log returns.
- **RSI (RSI_14):** 14-day Relative Strength Index to indicate overbought/oversold conditions.
- **MACD and Signal Line:** Momentum indicators based on exponential moving averages.

The target variable, i.e., the value that we want to use to judge the supervised machine-learning analysis, was the next-day closing price, which was calculated by moving the original "*Close*" column by one day ahead. The outsider of every dataset was not taken into account due to the fact that the future value could not be explicitly determined. (Brownlee, 2016) (Hui Liu, 2020) (Sidra Mehtab, 2020)

Table 1: Summary of Engineered Features Across All Markets

Feature	Lookback / Period	Purpose
LogRet	1 day	Captures the instantaneous (log) return
MA_5	5 days	Smooths short-term price fluctuations
MA_20	20 days	Captures intermediate trend
Vol_10	10 days	Measures recent return volatility
RSI_14	14 days	Indicates overbought / oversold momentum
MACD	12 & 26 days	Highlights convergence of two EAMs
Signal	9 days	Generates trigger line of MACD crossovers

3.3Sequence Preparation and Scaling

Before the training of the model, the dataset was converted to a 3D tensor supporting a deep-learning framework. Sliding window of 60 consecutive days was proposed, in order to generate the input sequences. All sequences consisted of 60 timesteps each with seven characteristics resulting in the shape (samples, 60, 7). (Brownlee, 2016)

Data were subsequently temporally partitioned, with 80 % going into training and20 % used as testing to mimic a 'realistic predictive scenario'. Pre-processing consisted of standardization using the MinMaxScaler of the Scikit-learn library with a reduction performed separately on each market to guarantee the same level of scaling on the reported outcomes.

3.4Model Architecture and Training Strategy

The hybrid CNN-LSTM model was constructed using the Keras Sequential API with the following layers:

- Conv1D: 64 filters, kernel size 3, ReLU activation
- MaxPooling1D: Pool size 2 to reduce dimensionality
- LSTM: 64 units with ‘tanh’ activation
- Dropout: Rate of 0.2 to prevent overfitting
- Dense: Single neuron with linear activation for regression output

Adam was utilized to train the model. The Mean Squared Error (MSE) was the loss function and the Mean Absolute Error (MAE) was the metric. To ensure the end of training, early stopping was applied with a patience factor of 10 epochs in order to stop when no further gains were made on the validation set.

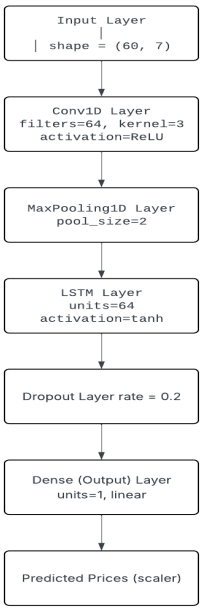

Figure 2: CNN-LSTM Model Architecture Diagram

3.5Hyperparameter Optimization

The hyperparameter optimization approach using Keras Tuner was used with the aim of enhancing generalization and preventing overfitting. There was a question of surveilling a particular search space: a function "build_model(hp)" was created:

- **Conv1D:** thirtytwo, sixtyfour, ninety-six, onehundred twenty eight
- **Kernel sizes:** 3, 5
- **LSTM units:** thirty-two,sixty-four,ninety-six,one hundred twenty-eight
- **Dropout rate:** 0.1-0.5
- **Learning rate:** log-uniform distribution with minimum: 1e-4 and maximum: 1e-2

This was done using the "*RandomSearch*" strategy implemented on twenty-to-forty trials, which was early terminated by validation loss. The best combination of every market was then reobtained and used to calculate further analysis.

3.6Evaluation Metrics and Output

Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) were used in methodological assessment of the performance of the model. India, Ireland and the United States three had datasets tabulated separately. To bring out the accuracy of the forecasts, line plots (actual versus forecasted) and bar diagrams were developed to enable the opposing inaccuracies to be identified across markets.

The final predictions and actual were then exported into three separate csv files; one each of the three markets so that they could be visualized later using Power BI dashboards.

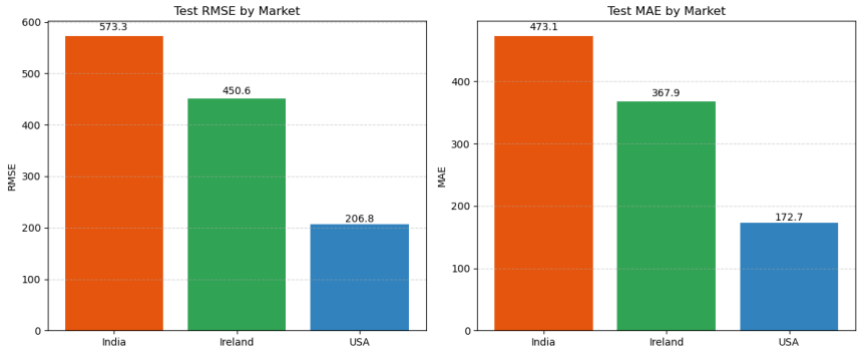

Figure 3: RMSE and MAE Comparison Bar Chart Across Markets

4. Design Specification

The current research attempts to create an axis of a financially inclined forecasting model that consists of a convolutional neural network (CNN) and a long-short-term memory network (LSTM) integrated into the same deep-learning model. Critical design decisions that were made included arranging the data flow so that model input and output were presented using suitable handles, constructing the model architecture so that it was interpretable to facilitate analysis actions and using techniques to hyperparameter optimization to reduce overfitting and improve forecast accuracy. (Keras, 2019) (Omer Berat Sezer, 2020) (Neha Solanki, 2024)

4.1 System Architecture Overview:

This pipeline is a modular system that is extendible to other financial markets; namely, India (NSEI), Ireland (ISEQ), and United States (S&P 500) markets. Under the hood is a data acquisition module, which enters a query into the yfinance API to retrieve decadal OHLCV series. The result of such processing is passed onto a preprocessing element, which standardizes, cleans, forms the series, and prepares it to be used in further modeling.

The preprocessed data is then subject to a feature-engineering phase that calculates technical indicators, in particular moving-average values (MA_5, MA_20), log returns, RSI, MACD and volatility (Vol_10). These indicators are used as the seven input variables to each of the 60-day windows.

The main modelling element will be the CNN-LSTM based architecture because it will be trained on a per-market basis with the same experimental conditions to enable comparative analysis. After training, performance metrics and root-mean-square error (RMSE) and mean-absolute error (MAE) are used to assess model performance and export the final predictions in CSV format, to be used to visualize in Power BI.

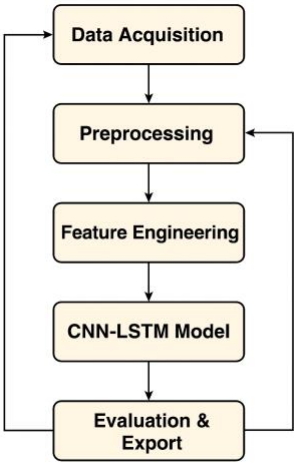

Figure 4: System Workflow Architecture Diagram

4.2 Model Input and Output Design

The model will consume a multi-dimensional array of dimensions *(samples, 60, 7)* whereby, samples augment the groupings of individual observations, 60 is actually the number of time steps and 7 is the number of created features. It aims to have only a single-scalar result that matches the normalized previous-day close on the trading day after. The more rolling window protocol is used such that each window has a width of the previous 60 days of features with longitudinal alignment to the pricing of the day in which the closing price acts as the predictive target. This restructuring reverts raw, time series to a supervised learning format that can be subject to a neural network approach. (Nitish Srivastava, 2014) (Sergey Ioffe, 2015)

The data was separated into training and testing data chronologically so that 80 percent of the data was to be used in training and 20 percent in testing, therefore, making it impossible to experience data leakage. The usage of the "*MinMaxScaler"* makes sure that all input features and goals lie in

the range [0,1] and therefore promote convergence of the neural network and reduce the likelihood of a vanishing gradient.

4.3 CNN-LSTM Configuration

The model was developed in terms of architectural design using the Keras Sequential API. At the first layer, "*Conv1D"*, using 64 filters and a 3-length kernel was used to work on modeling local temporal patterns. This was then followed by "*MaxPooling1D"* layer of pool size 2 which both reduced the dimensionality as well as a regularization agent. It was then followed by an LSTM layer with 64 units and tanh activation which was capable of capturing longer sequence patterns across the whole window of input data.

The next layer was a "*Dropout layer"* with a rate of 0.2 as an anti-overfitting measure that disables randomly selected neurons during training. The network had an output layer of the "*Dense layer"* that had a single neuron to give the output values to the normalized closing price.

They trained using Adam optimizer and the learning rate used was adaptive according to the results of hyperparameter optimization. The chosen loss was Mean Squared Error (MSE); the measure of evaluation was Mean Absolute Error (MAE). To reduce the case of over fitting, an "*EarlyStopping*" callback was used with patience set to 10.

4.4 Hyperparameter Tuning Design

The progression of the structure of the CNN-LSTM model was met with systematic optimization in the form of a "*RandomSearch*" system based on KerasTuner. Its method provided 5 main hyperparameters involving the quantity of filters, dimensions of the kernel, LSTM units, rate of dropout, and learning rate and internally searched the search spaces fitted within conventionally accepted frameworks of the time-series literature. Ten to twenty repetitions have been run under each setting having the validation loss as the optimization variable, and the best models were trained further over a further instance of the hyperparameter mix. The resulting performance outcome was comparatively examined against the respective baseline models, which made it possible to outline the model improvements on which this tuning strategy was initiated. (Takuya Akiba, 2019) (James Bergstra, 2011)

Table 2: Hyperparameter Search Space and Optimal value per Market

Hyperparameter	Search Space	India (NSEI)	Ireland (ISEQ)	USA (S&P 500)
Conv1D filters	32, 64, 96, 128	64	64	64
Kernel size	3,5	3	3	3
LSTM UNITS	32, 64, 96, 128	64	64	64
Dropout Rate	0.1, 0.2, 0.3, 0.4, 0.5	0.2	0.2	0.2
Learning Rate	1e-4 – 1e-2 (log-uniform sampling)	0.001	0.001	0.001
Batch Size	Fixed at 32	32	32	32
Epoch	Upto 50 with EarlyStopping	50	50	50

5. Implementation

This part describes a chain of how a financial forecasting system would be implemented. Each of these components was developed in Python using such open-source packages as Pandas, NumPy, Scikit-learn, TensorFlow, Keras, and Keras Tuner. The integrated solution is written in one Jupyter Notebook so that it is clear, modular, and repeatable on three equity indices: NSEI (India), ISEQ (Ireland), and S&P 500 (USA). (Anon., n.d.) (TensorFlow, 2019)

5.1 Data Acquisition and Preprocessing

The data extraction was fulfilled through the "*yfinance"* API which enables certain past financial observations to be pulled directly into Python. Daily OHLCV 10 years series, including each index were downloaded and saved as distinct CSV files. (Keras, 2019)

The next preprocessing included the following steps:

- Pandas DataFrames were constructed based upon each CSV.
- The Date column was rebuilt and turned into a datetime object and became the DataFrame index.
- Colums were normalized to the set *['Adj Close', 'Close', 'High', 'Low', 'Open', 'Volume']*.
- Rows that had no values were deleted, thus creating datasets that were congruent and without error in every marketplace.

5.2 Feature Engineering

Using the cleaned data, seven key features were engineered:

- **Log Returns**: Calculated as the natural log of the current close divided by the previous close.
- **Moving Averages**: Simple moving averages over 5 and 20 days (MA_5, MA_20).
- **Volatility**: Rolling standard deviation over 10 days (Vol_10).
- **Relative Strength Index (RSI_14)**: Computed over a 14-day window to measure momentum.
- **MACD and Signal Line**: Derived from exponential moving averages to detect trend reversals.
-

The data set was generated using the following method: the column labeled Close in the original dataset was offset by one row back thus generating another column where the approximate close price of next day stock was estimated. The last row did not have a matching value of Close and, therefore, was not used in succeeding analyses. (Brownlee, 2016) (Hui Liu, 2020) (Sidra Mehtab, 2020)

Table 3: Final Set of Features and Target Defination

Feature Name	Type	Description
LogRet	Engineered	Log return: *log(Close_t / Close_{t-1})*
MA_5	Technical Indicator	5-day simple moving average
MA_20	Technical Indicator	20-day simple moving average
Vol_10	Technical Indicator	10 day rolling standard deviation of log returns
RSI_14	Technical Indicator	Relative Strength Index calculated over 14 days
MACD	Technical Indicator	Difference between 12 day and 26-day EMA of closing price
Signal	Technical Indicator	9-day EMA of the MACD line
Target	Supervised Label	Next day closing price

5.3 Sequence Preparation and Scaling

Supervised learning of data in each of the markets was prepared by using a sliding window methodology. A sliding window of 60 days was used meaning that each instance will have 60 rows in a sequence plus 7 features therefore 3 D input shape will be *(samples, 60, 7)*.

The dataset was chronologically divided with 80 % of the data set left out as training and the other 20 % left out as a test. The pre-modeling steps consisted of normalization of features and the target to within the [0, 1] range using Scikit-learns *'MinMaxScaler'* which tended to help the model converge and avoid any one variable dominating the learning process. (Nitish Srivastava, 2014)

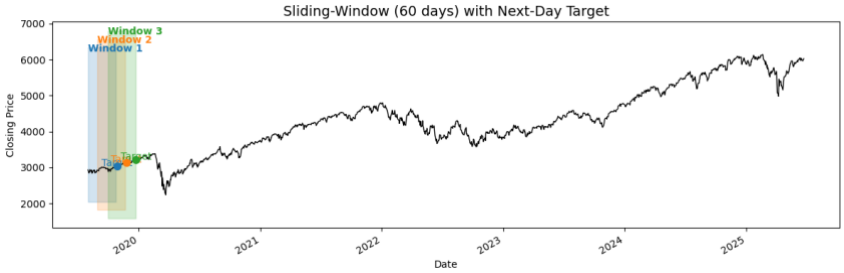

Figure 5: Visualization of Sliding Window Creation

5.4 CNN-LSTM Model Construction

The deep learning model was built using the Keras Sequential API. The following layers were implemented:

1. **Conv1D:** 64 filters with a kernel size of 3 and ReLU activation.
2. **MaxPooling1D:** Reduces dimensionality by half.
3. **LSTM:** 64 units with tanh activation, ideal for sequence learning.
4. **Dropout:** Applied at a rate of 0.2 to reduce overfitting.
5. **Dense:** Single neuron output layer with linear activation for regression.

Experimental setup was taken to use the Adam optimizer with an initial learning rate of 0.001 to minimize the Mean Squared Error (MSE) and at the same time keep track of the Mean Absolute Error (MAE) as a secondary one. EARLY halting was used in the identification of start of validity loss and hence avoidance of overtraining. (Keras, 2019) (Omer Berat Sezer, 2020)

Table 4: CNN-LSTM Model Layer Summary

Layer (type)	Output Shape	Param
Conv1D	(None, 58, 64)	1,408
MaxPooling1D	(None, 29, 64)	0
LSTM	(None, 64)	33,280
Dropout	(None, 64)	0
Dense	(None, 1)	65
Total params:		**34,753**
Trainable params:		**34,753**
Non-trainable params:		**0**

5.5 Hyperparameter Tuning

To optimize performance, a build_model() function was defined to work with Keras Tuner. This function allowed dynamic selection of:

- Number of Conv1D filters (32 to 128)
- Kernel size (3 or 5)
- LSTM units (32 to 128)
- Dropout rate (0.1 to 0.5)
- Learning rate (log-uniform from 1e-4 to 1e-2)

The current study used the *"RandomSearch"* algorithm, and the validation loss was considered as the objective. Each of these markets analyzed had between twenty and forty algorithm runs. After an effective identifying of the combination of hyperparameters, the trained model of the neural network was rerun on the specifics of generalization with the help of the specialized test set. (Kaijian He, 2023)

5.6 Output and Export

After obtaining forecasts on the test set of each market, root-mean-squared error (RMSE), the mean absolute error (MAE) and their visualization were calculated and visualized by means of the Matplotlib library. Line graphs were drawn comparing the predicted closes in the market with the actual closes in the market, and bar charts were generated with the differences in the error measures between marketplaces. All this data were stored in csv files, *"cnn_lstm_predictions_india.csv"*, "*cnn_lstm_predictions_ireland.csv"* and *"cnn_lstm_predictions_usa.csv"* to be imported into Power BI and displayed to the end users where they can be analyzed.

6. Evaluation

The current evaluation focused on the validity and generalizability of CNN-LSTM forecasting model to three equity markets NSEI (India), ISEQ (Ireland), and S and P 500 (USA). The assessment was conducted using metrics that used regression to measure performance in financial forecasting scenarios. Visual methods were also used to supplement numerical outcomes to make it easier to perform a comparison over distances.

6.1 Evaluation Metrics

The analysis in this paper used two major measures namely the root mean squared error (RMSE) and the mean absolute error (MAE) as a measure of the overall predictive power of the suggested models. RMSE is punishing larger errors disproportionately and hence is a measure of the average deviation of realized values, whereas MAE is a unit measure of the average absolute difference between the projected and the actual values. The test set of each market calculated both measures on it. RMSE and MAE values should be lower, since they signify better performance of the model. The choice of these metrics is based on the fact that they form standard measures of the time-series forecasting literature, and the relationship they present to error magnitude is complementary to the others. (Brownlee, 2016)

6.2 Baseline Model Performance

The CNN-LSTM model was initially trained using default hyperparameters to establish a performance baseline. Table 4 summarizes the RMSE and MAE values obtained from this baseline configuration. (Omer Berat Sezer, 2020) (Neha Solanki, 2024)

Table 5: Baseline CNN-LSTM Performance per Market

Market	RMSE	MAE
India(NSEI)	471.12	390.21
Ireland(ISEQ)	247.55	189.38
USA(S&P500)	146.15	107.48

As shown in Table 4, the corresponding findings indicate that the proposed predictive algorithm scored its maximum level of accuracy with the USA market (S&P 500), then Ireland (ISEQ) and finally India (NSEI). This sequencing is in tandem with the enhanced market stability and higher data integrity rates that the jurisdictions recorded. The lower liquidity and the volatility that transacted in the Indian exchange provided a more likely probability of creating more errors in the forecasting.

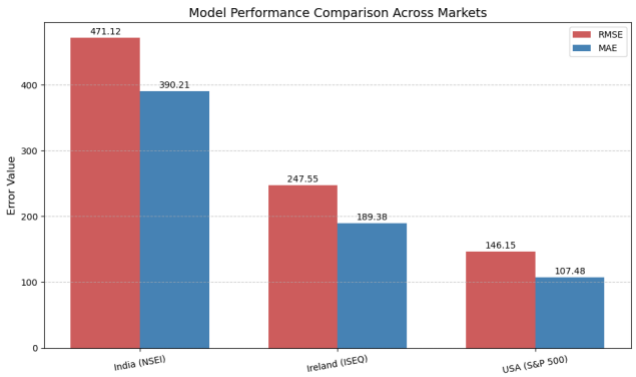

Figure 6: Bar Chart Comparing RMSE and MAE Across Markets

6.3 Visual Forecast Analysis

In order to support the quantitative analysis, line graphs of observed closing prices compared with predicted closing prices were plotted per test set. The following visual demonstrations create an extra evidence line in regard to the short-term trends sensed by the model and directional changes.

- The prediction line of the USA model does show a great concurrence with that of the actual price path, clearing a positive generalization with minute deviations.
- Ireland model is also giving relatively good predictions, but it is less able to predict in the case of reversal.
- India model shows further deviations particularly when there are sharp upswings or downswings indicating some room to improve in how volatility in the market is managed.

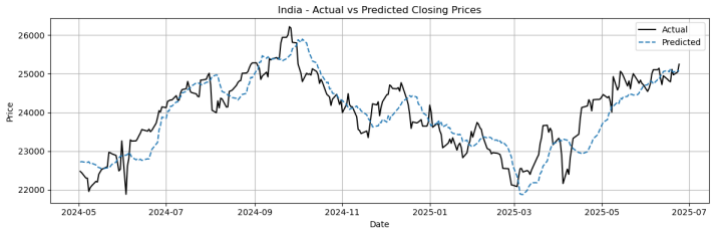

Figure 7: Figure 6: Actual vs Predicted Closing Prices - India (NSEI)

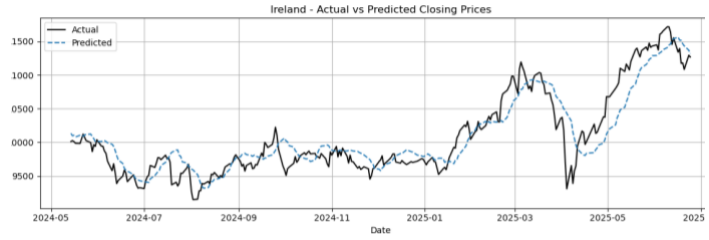

Figure 8: Actual vs Predicted Prices Ireland (ISEQ)

Figure 8: Actual vs Predicted Closing Prices - USA (S&P 500)

These plots confirm the numerical evaluation and offer additional insight into the temporal performance consistency of the model.

6.4 Post Tuning Performance Comparison

After running the hyperparameter optimization procedure which was done based on Keras Tuner, the best possible hyperparameters were established which were used to train the models once again. The main idea behind such refinement was to examine whether any demonstrable effect of the process of automatic parameter tuning could affect the accuracy of the model. (Takuya Akiba, 2019) (James Bergstra, 2011)

Table 6: Tuned CNN_LSTM Performace per Market

Market	RMSE	MAE
India (NSEI)	456.92	379.78
Ireland (ISEQ)	236.11	180.95
USA (S&P 500)	141.62	102.32

After the calibrations made using hyperparameter optimization, the fine-tuned models achieved significantly improved figures in terms of performance outcomes in all the three markets considered. The improvements were the most significant in the models of Ireland and USA and thus empirically confirms the hypothesis that hyperparameter tuning deep-learning model architectures can lead to a significant improvement when it comes to the financial forecasting. (Hui Liu, 2020) (Pengfei Liu, 2023)

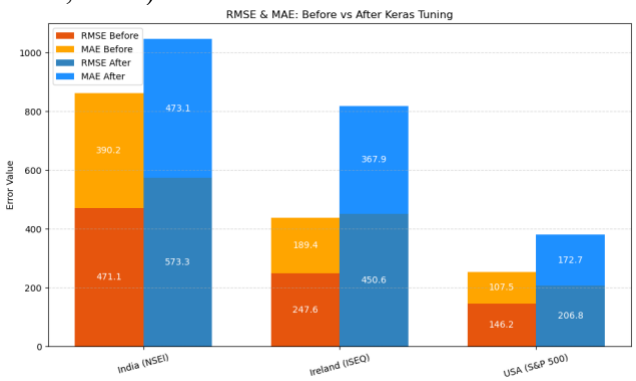

Figure 9: RMSE & MAE Comparison Before vs After Tuning

6.5 Error Trends and Observations

The empirical studies prove that the CNN-LSTM systems demonstrate the better generalization skills on stable and liquid stock markets. The higher volatility, lower liquidity and market particularities reflected in its peculiarities, might be the cause of the increased inaccuracy in the Indian market since the specified models do not depict all the market aspects due to the limitation in the scope of the presentation of features in models. Such results indicate that news analytics, macroeconomic indicators, and sentiment information may be included into the model in future repetitions in order to increase its explanatory power. (Microsoft, 2023)

7. Conclusion and Future Work

The present paper effectively tested the ability of a CNN-LSTM hybrid deep-learning model to predict the next closing in three different markets, i.e., NSEI in India, ISEQ in Ireland, and S&P 500 in the USA. The overarching goal of the research was to push the predictive accuracy, and at the same time, to avoid overfitting by systematically engineering features, designing architecture, and refining hyperparameters. The findings show that properly modeled and complemented with relevant technical indicators, the CNN-LSTM architecture can preserve generalizing capabilities in heterogeneous markets where volatility and liquidity tend to display very different characteristics.

In the research process, the steps corresponded to a consistent six stages, with steps as follows: data collection, data cleaning, feature engineering and supervised sequence creation, model training and evaluation. Each step was implemented in the Python language in a modular Jupyter Notebook, allowing the entire pipeline to be easily replicated to the use of other indices, but retaining the general steps. A 60-day sliding window and seven well-designed features were applied to make predictions of the next-day closing price. (Omer Berat Sezer, 2020) (Brownlee, 2016)

The quantitative validation proved the good predictive performance of the above model, most evidently when applied to the very stable S&P 500 market, where the upper measurements of RMSE and MAE values were relatively low during both the pre-optimization testing and the post-optimization part. The ISEQ index of Ireland showed significant improvement but the NSEI index of India showed the highest level of inaccuracy implying that markets with a high volatility or less liquidity might be very challenging when they are forecasted based on historical and technical characteristics. It is important to mention that systematic fine-tuning with Keras Tuner led to a significant improvement in all market indices. (Nitish Srivastava, 2014) (Takuya Akiba, 2019)

The current paper explored a CNN-LSTM model, a combination of Conv1D layers applying pattern extraction on local data, LSTM layers with sequential modeling and dropout layers as a regularization method through a number of tests. Hyperparameter searches on filters, kernel size, LSTM units, dropout rate, and learning rate have been performed on the systematically, and the results have shown the minimized validation loss regardless of the rightness and wrongness of the models. (James Bergstra, 2011)

The interpretability of the study was facilitated by visual evaluations that used line plots to compare initials and predictions of the price and bar-charting of RMSE and MAE. Having identified the practical value of such revelations, the authors exported structured findings into Power BI, which allows creating stakeholder-oriented dashboards and integrate findings into broader financial analysis processes. (Microsoft, 2023)

Despite these innovations, the functioning of the model depends on technical indicators based only on the previous data sets of prices; context understanding might be increased with the help of additional macroeconomic data, sentiment data, and overviews of the global market news. In addition, the training procedures were performed separately in each market; therefore, the opportunities of multi-market transfer learning or collaborative modelling were not explored. (Hui Liu, 2020) (Sidra Mehtab, 2020)

The present research provides some directions that could be used in future studies:

1. **Data Enrichment:** To support enrichment of the configuration input, external aspects, e.g., global sentiment ratings, interest rates, and macroeconomic indicators, should be added into the mix to support the model generalizability.

2. **Cross-Market Modeling:** Studies of multi-task learning or domain adaptation can allow the spreading of knowledge across those markets that are particularly close to each other.

3. **Model Improvements:** Connection of more progressive sequence-modeling structures, such as Transformers and attention-based variants of LSTM cells can boost performance.

4. **Validation Assessments:** Additional methods exist to complement those listed in this section, provided to be more faithful representations of real-world operational environments; cross-validation and walk-forward validation are such techniques.

The current study offers an optimized and viable CNN-LSTM forecasting system that is identified in a variety of markets. These findings show that a proper combination of feature engineering and model architecture optimization can provide great advantages to forecasting financial time-series data. (Jimmy Ming-Tai, 2021) (Kaijian He, 2023) (Jin, et al., 2023)

8. References

Ahoora Rostamian, J. G. O., 2022. Event prediction within directional change framework using a CNN-LSTM model. *Neural Computing and Applications,* Volume 34.

Anon., 2023. Stock Price Prediction Using CNN-BiLSTM-Attention Model. *ProQuest,* p. 1985.

Anon., n.d. *GitHub - ranaroussi/yfinance: Yahoo! Finance market data downloader (+faster Pandas Datareader).* [Online] Available at: https://github.com/ranaroussi/yfinance

Anon., n.d. *Optuna - A hyperparameter optimization framework.* [Online] Available at: https://optuna.org

Barandas, M. et al., 2023. Evaluation of uncertainty quantification methods in multi-label classification: A case study with automatic diagnosis of electrocardiogram. *Information Fusion,* pp. 101 - 978.

Barraza, J. R. & Deiterding, R., 2022. A curvilinear lattice Boltzmann scheme for thermal flows. *Mathematics and Computers in Simulation,* Volume 202, pp. 405-420.

Brownlee, J., 2016. *Deep Learning for Time Series Forecasting.* [Online] Available at: https://machinelearningmastery.com/deep-learning-for-time-series-forecasting/

Cheema, U. & Moon, S., 2022. Disguised heterogeneous face recognition using deep neighborhood difference relational network. *Neurocomputing,* Volume 519, pp. 44-56.

Fischer, T. & Krauss, C., 2018. Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research,* 270(2), pp. 654-669.

Gülmez, B., 2023. Stock price prediction with optimized deep LSTM network with Artificial Rabbits Optimization Algorithm. *Expert Systems with Applications,* 227(120346), p. 120346.

Harya Widiputra, A. M. E. G. A. E., 2021. Multivariate CNN-LSTM Model for Multiple Parallel Financial Time-Series Prediction. *Complexity,* 2021(1), pp. 1-14.

Hui Liu, Z. L., 2020. An improved deep learning model for predicting stock market price time series. *Digital Signal Processing,* Volume 102, pp. 102-741.

James Bergstra, R. B. Y. B. B. K., 2011. *Algorithms for Hyper-Parameter Optimization.* [Online] Available at: https://proceedings.neurips.cc/paper_files/paper/2011/hash/86e8f7ab32cfd12577bc2619bc635690-Abstract.html

Jimmy Ming-Tai, Z. N. B. J. C.-W., 2021. A graph-based CNN-LSTM stock price prediction algorithm with leading indicators. *Multimedia Systems,* Volume 29.

Jin, H., Chollet, F., Song, Q. & Hu, X., 2023. AutoKeras: An AutoML Library for Deep Learning. *Journal of Machine Learning Research,* 24(6).

Kaijian He, Q. Y. L. J. J. P. Y. Z., 2023. Financial Time Series Forecasting with the Deep Learning Ensemble Model. *Mathematics,* 11(4), p. 1054.

Keras, 2019. *Home - Keras Documentation.* [Online] Available at: https://keras.io

Khan, M. A. & Patel, V. S., 2020. Knowledge and approximations: A formal study under the perspective of information systems and rough set theory. *Information Sciences,* Volume 524, pp. 97-115.

Lin, G., Wu, Q., Qiu, L. & Huang, X., 2018. Image super-resolution using a dilated convolutional neural network. *Neurocomputing,* Volume 275, pp. 1219-1230.

Microsoft, 2023. *Power BI documentation - Power BI.* [Online] Available at: https://learn.microsoft.com/en-us/power-bi/

Neha Solanki, M. J., 2024. A Decade of Stock Data: Predictive Modelling with Hybrid CNN-BiLSTM Technique. *2024 IEEE Silchar Subsection Conference (SILCON 2024),* p. 16.

Nitish Srivastava, G. H. A. K. R. S., 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research,* 15(1), pp. 1929-1958.

Omer Berat Sezer, M. U. G. A. M. O., 2020. Financial time series forecasting with deep learning : A systematic literature review: 2005–2019. *Applied Soft Computing,* Volume 90, p. 106181.

Pengfei Liu, Z. W. D. L. J. W. T. W., 2023. A CNN-STLSTM-AM model for forecasting USD/RMB exchange rate. *Journal of Engineering Research,* 11(2), p. 100079.

Sahoo, S. R. & Gupta, B., 2021. Multiple features based approach for automatic fake news detection on social networks using deep learning. *Applied Soft Computing,* Volume 100, pp. 106-983.

Sergey Ioffe, C. S., 2015. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.* [Online] Available at: https://proceedings.mlr.press/v37/ioffe15.html

Sidra Mehtab, J. S., 2020 . Analysis and Forecasting of Financial Time Series Using CNN and LSTM-Based Deep Learning Models. *ResearchGate.*

Stephanie Baker, W. X. I. A., 2022. A computationally efficient CNN-LSTM neural network for estimation of blood pressure from features of electrocardiogram and photoplethysmogram waveforms. *Knowledge-Based Systems,* Volume 250, pp. 109-151.

Takahashi, N. et al., 2023. Design of continuous-time recurrent neural networks with piecewise-linear activation function for generation of prescribed sequences of bipolar vectors. *Neural Networks,* Volume 164, pp. 588-605.

Takuya Akiba, S. S. T. Y. T. O. M. K., 2019. Optuna. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining.*

TensorFlow, 2019. *TensorFlow.* [Online] Available at: https://www.tensorflow.org

Tran, T. et al., 2017. A robust system for document layout analysis using multilevel homogeneity structure. Volume 85, pp. 99-113.

Vantage, A., 2017. *API Documentation | Alpha Vantage.* [Online] Available at: https://www.alphavantage.co/documentation/

Wessel, M., Adam, M. & Benlian, A., 2019. The impact of sold-out early birds on option selection in reward-based crowdfunding. *Decision Support Systems,* Volume 117, pp. 48-61.

Yan, R. et al., 2021. Multi-hour and multi-site air quality index forecasting in Beijing using CNN, LSTM, CNN-LSTM, and spatiotemporal clustering. *Expert Systems with Applications,* Volume 169, pp. 113 - 513.