Volatility Prediction Of Stock Market Using Deep Learning Algorithms.

A THESIS

submitted in partial fulfillment of the requirements for the award of the dual degree of

Bachelor of Science

in

Data Science and Engineering

by

Baddepudi E V N M SaiAsrith
(20074)

Department of Data Science and Engineering
INDIAN INSTITUTE OF SCIENCE EDUCATION AND RESEARCH BHOPAL
BHOPAL - 462066
April 2024

भारतीय विज्ञान शिक्षा एवं अनुसंधान संस्थान, भोपाल
Indian Institute of Science Education and Research, Bhopal
Department of Data Science and Engineering
Bhopal Bypass Road, Bhauri Bhopal 462 066

CERTIFICATE

This is to certify that **Baddepudi E V N M SaiAsrith**, BS (Data Science and Engineering), has worked on the project entitled **' Volatility Prediction Of Stock Market Using Deep Learning Algorithms.'** under my supervision and guidance. The content of this report is original and has not been submitted elsewhere for the award of any academic or professional degree.

Dr. Akshay Agarwal (PI) **Dr. Biswajit Patra (Co-PI)**

Apr 2024
IISER Bhopal

Committee Member	**Signature**	**Date**
Dr. Tanmay Basu (Convenor)		
Dr. Vaibhav Kumar		
Dr. Vinod Kurmi		

ACADEMIC INTEGRITY AND COPYRIGHT DISCLAIMER

I hereby declare that this project is my own work and, to the best of my knowledge, it contains no materials previously published or written by another person, or substantial proportions of material which have been accepted for the award of any other degree or diploma at IISER Bhopal or any other educational institution, except where due acknowledgement is made in the document.

I certify that all copyrighted material incorporated into this document is in compliance with the Indian Copyright Act (1957) and that I have received written permission from the copyright owners for my use of their work, which is beyond the scope of the law. I agree to indemnify and save harmless IISER Bhopal from any and all claims that may be asserted or that may arise from any copyright violation.

Apr 2024 **Baddepudi E V N M SaiAsrith**
IISER Bhopal

ACKNOWLEDGMENT

First, I extend my heartfelt gratitude and humble prayers to my gurus, B Vijaya and Sankara Prasad Father, and my sister B Sai Vineetha for their grace and blessings. Without them, none of this would have been possible.

I am immensely grateful to the many individuals who have shown me unwavering support and love throughout my journey. I have been incredibly fortunate to encounter an exceptional group of people who have helped me grow and achieve my goals. I sincerely thank my mentors, Dr. Akshay Agarwal, and Dr. Biswajit Patra, for their invaluable guidance and unwavering support during my thesis.

I also want to express my gratitude to the faculty members of the DSE department for offering us engaging and thought-provoking courses and projects. The knowledge and skills I have gained from them have been beneficial.

My heartfelt thanks go to Abhishek and Roshan for being great friends throughout my journey. Their support and encouragement were invaluable. I am also grateful to my friends from the Ph.D. lab, who made me feel comfortable in the lab. I also thank Ishan, Devashish, and Suraj for being sweet juniors.

Most importantly, thank Amma, Nanna, and Akka for their unwavering support, love, trust, and encouragement. They have been my role models and inspiration since day one.

I also want to express my appreciation to my dear friends, Abhishek, Members of the five states of Matter group, the English gang, Sugandh, Soham, my seniors Akansha Singh and Rita, and many others, who supported and encouraged me with their love and positivity. I also especially thank Shreya, who motivated me. I learned a lot of great things from you, and I enjoyed spending time with you and will remember you guys forever.

Baddepudi E V N M SaiAsrith

ABSTRACT

Volatility prediction of the stock market has been a hot topic in Economics and Finance; because of the complex and volatile nature of the stock market, it is challenging to forecast the volatility accurately of the company's stock. We have introduced an NBEATS plus MTL, a hybrid model, a combination of a time series forecasting model, and a deep learning model to estimate the stock volatility precisely. In this research, a total of two stock indexes and three evaluation metrics have been used. The introduced model outperforms in all three evaluation metrics compared to traditional GARCH and its variants and existing hybrid models such as ANN-GARCH, LSTM-GARCH, MT-GARCH, and MTL-GARCH. So, it demonstrates that NBEATS plus MTL is a more suitable and promising method for market volatility prediction, which has widespread value.

LIST OF SYMBOLS OR ABBREVIATIONS

$\sum$ = Summation
σ_t = Standard Deviation of the error term i.e., Volatility
r_t = Returns at time t
μ = Mean of returns
N = Number of Observations
ϵ_t = Error or Residue at time t
z_t = White noise error term with zero mean and unit variance.
$\alpha_1, \alpha_2, ..., \alpha_p$ are parameters to be estimated.
p = order of ARCH
q = order of GARCH
θ = Measures the leverage effect.
γ = Size of the shocks
β_j= Parameters to be estimated.
$g(\epsilon_t)$ = Function of lagged error terms.
$I_{\epsilon(t-1)<0}$ = An indicator function takes the value 1 if ϵ_{t-1} is negative, otherwise it is 0.
ω_i = parameters to be estimated
L = Lag operator
d = fractional differencing parameter.
Y_t = Actual Volatility at time t.
L_t = Level component of the data at time t.
T_t = Trend Component of the data at time t.
S_t = Seasonal component of the data at time t.
S = Seasonal Cycle Y_{t+h} = Forecasted Volatility h periods ahead. ARCH = Auto-Regressive Conditional Heteroskedasticity.
GARCH = Generalized Auto-Regressive Conditional Heteroskedasticity.
EGARCH = Exponential Generalized Auto-Regressive Conditional Heteroskedas-

ticity.
TGARCH = Threshold Generalized Auto-Regressive Conditional Heteroskedasticity.
Abs-GARCH = Absolute Generalized Auto-Regressive Conditional Heteroskedasticity.
ANN = Artificial Neural Network.
DSVM Deep Stochastic Volatility Model.
LSTM = Long Short-Term Memory.
MT = Multi-Transformer.
MTL = Multi-Transformer-LSTM.

LIST OF FIGURES

LIST OF TABLES

Contents

Chapter 1

Introduction

The prediction of stock market volatility is a crucial area of financial economics, attracting extensive academic and practical interest due to its significant implications for portfolio management, risk assessment, derivative pricing, and strategic decision-making. Following the 2007–2008 financial crisis, several businesses have strengthened their risk management systems to meet specific requirements. These compulsory requirements mainly aim to minimize financial losses from unexpected events.

Volatility is defined by the degree of variation in trading prices over time. It is the central element in accessing the risks and opportunities in financial markets. Accurately predicting the volatility helps investors and financial companies make informed decisions, optimizing returns and mitigating losses. Predicting stock market volatility presents numerous challenges because political events, economic indicators, and investor sentiment influence the financial markets.

Historically, various methods have been used to predict stock market volatility[1]. In the early stages of stock market volatility, prediction mainly relied on time series models such as auto-regressive conditional heteroskedasticity (ARCH), Generalized ARCH, and its variants. These models are particularly noted for their ability to model financial time series data features such as volatility clustering - which is a phenomenon where high volatility events tend to cluster together - often occurs.

Many GARCH model variants, including EGARCH, GJR-GARCH, T-GARCH, and Absolute GARCH, have been developed as the field progresses.

Another category includes stochastic volatility models, such as the deep stochastic volatility model (DSVM), which posits that volatility itself follows a stochastic process[2]. Still, the main challenge of these models is the estimation of the parameters. Hence, in response to the limitations of the econometric, time series, and stochastic models, recent research has been increasingly focused on introducing hybrid models, such as combining the econometric and time series forecasting algorithms with deep learning algorithms. By integrating these methodologies, hybrid models aim to leverage each other's strengths, thereby enhancing the accuracy and reliability of the volatility forecast.

In this thesis, we benchmark the existing models such as ARCH, GARCH, EGARCH, TGARCH, GJR-GARCH, LSTM, and some hybrid models such as ANN-GARCH, LSTM-GARCH, Transformer-GARCH, Multi-Transformer-GARCH, Transformer-LSTM-GARCH, Multi-Transformer-LSTM-GARCH. Later, based on the analysis of the existing models, we developed a novel hybrid algorithm named NBEATS-Multi-Transformer-LSTM. The proposed algorithm yields state-of-the-art surpasses the existing algorithm on multiple databases.

Chapter 2

Literature Review

This section will study what exactly volatility is and how it is measured. In later sections, we have discussed Econometric models, time series forecasting, and hybrid models, which measure and predict the stock market's volatility.
Volatility measures how much the price of the market index changes over time. Therefore, volatility measures the standard deviation of logarithmic returns observed over fixed intervals.

$$\sigma = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(r_i - \bar{\mu})^2}$$

and

$$r_t = \log\frac{(Price_t - Price_{t-1})}{(Price_{t-1})} \tag{2.1}$$

where
r_i = returns at period i.
$price_t$ = closed price of the market at period t.
$price_{t-1}$ = closed price of the market at period t-1.

There are two types of volatility:

1. **Realized volatility:** It calculates the financial instrument's actual price change over a given time frame. It is often calculated as the standard de-

viation of the instrument returns. When computing realized volatility, the mean returns are typically used to estimate the average price movement.

2. **Implied Volatility:** It represents expected future asset volatility calculated using the current option prices.

In this thesis, we have used historical data to predict the future volatility. The study aims to develop a reliable model for predicting the stock market volatility of banks and indexes using time series, econometrics, and deep learning models. The study uses a multi-variate dataset of stock prices, trading volume, and economic indicators to capture the stock market data's complex patterns, such as trend, seasonality, and correlation structure. The results of this study offer insightful information about the stock market volatility and help investors and institutions make informed decisions about risk management and asset allocation.

2.1 Econometric models:

2.1.1 The ARCH model:

Robert Angle introduced the arch model [3]. It is a statistical model to analyze time series data, especially for modeling financial market volatility. The key idea of this model is that the variance of the current error term (or residual) is a function of the sizes of the errors in the previous periods. This reflects the observed clustering of volatility in financial markets where more considerable changes succeed in significant changes, and more minor changes succeed in small changes. The model can be expressed as

$$r_t = \mu + \epsilon_t \tag{2.2}$$

$$\epsilon_t = \sigma_t.z_t \tag{2.3}$$

The variance of the error term is modules as:

$$(\sigma_t)^2 = \alpha_0 + \alpha_1.(\epsilon_{t-1})^2 + ... + \alpha_p.(\epsilon_{t-p})^2 \tag{2.4}$$

where $\alpha_1, \alpha_2, ..., \alpha_n$ are the parameters to be estimated, and p is the order of arch.

The parameters of ARCH are theoretically estimated using Maximum Likelihood Estimation.
Limitations:

- Complexity with high lags: Whenever there is a high volatility clustering, the ARCH model might require many lagged error terms, i.e., P value in equation 2.4, leading to complexity and overfitting issues.
- Ignoring leverage effects and does not address the influence of the external variable on volatility prediction.

2.1.2 GARCH model:

GARCH stands for Generalized Auto-Regressive Conditional Heteroskedasticity, an extension of the ARCH model widely used to forecast financial market volatility. The GARCH works effectively when there is a high lag period. As mentioned in [4], the volatility clustering phenomenon—where periods of high volatility are followed by periods of high volatility and times of low volatility are followed by periods of low volatility—is best captured by the GARCH model. The GARCH model also includes a moving average component, allowing it to more effectively handle situations where the effect of the past stocks on current volatility decreases over time. [4] The advantages of using the GARCH model are its Efficiency with a long memory, and the model can be adopted and extended in various ways, such as Exponential-GARCH, GJR GARCH, Threshold-GARCH, etc., to capture different aspects of the financial time series and standard GARCH (P, Q) model is typically defined as

$$r_t = \mu + \epsilon_t \tag{2.5}$$

$$(\sigma_t)^2 = \alpha_0 + \sum_{i=1}^{P} (\alpha_i.(\epsilon_{t-i}))^2 + \sum_{i=1}^{Q} (\beta_j.\sigma_{t-j})^2 \tag{2.6}$$

Limitations:

- The model can become complex, particularly for the higher order of P and Q.
- There are certain conditions that the GARCH model should satisfy, like Non-

Negativity, stationary, etc., to ensure that the model is well-defined and the variance is favorable.

- Choosing the correct order of P and Q can be challenging and may require extensive empirical testing.

2.1.3 EGARCH:

Exponential GARCH is a variant of GARCH specifically designed to address certain limitations of the GARCH model. As mentioned in [5], It is particularly effective in modeling financial time series data that exhibit asymmetric volatility responses often observed in financial markets. The model is well-known for reflecting the leverage effects—the asymmetric impact of positive and negative shocks on volatility. The EGARCH model accounts for this asymmetry in financial markets, where negative news typically influences volatility more than positive news of equal scale. The conditional variance in this model is specified in the logarithmic form, which guarantees the predicted conditional variance to be always positive. [6] The logarithm of the conditional variance in an Exponential GARCH model is given as:

$$log(\sigma_t)^2 = \alpha_0 + \sum_{i=1}^{P} \alpha_i . g(\epsilon_{t-i}) + \sum_{j=1}^{Q} \beta_j . log(\sigma_{t-j})^2 \tag{2.7}$$

where

$$g(\epsilon_{t-i}) = \theta . \epsilon_{t-i} + \gamma(|\epsilon_{t-i}| - E[|\epsilon_t|]) \tag{2.8}$$

Limitations:

1. The model is mathematically more complex than the standard GARCH models, making it more challenging as implementations are computationally intensive because of the likelihood function.

2.1.4 TGARCH:

[7] Threshold GARCH: The model also works very similarly to the EGARCH model, specifically designed for capturing the asymmetric impacts of the positive and negative shocks on volatility. And it tends to raise future volatility more than

positive shocks of the same magnitude. It is also used to analyze the financial time series data, where negative and positive returns impact the volatility differently and where the parameters are estimated using the maximum likelihood estimation. [7] The conditional variance equation of the Threshold GARCH model is expressed as

$$\sigma_t^2 = \alpha_0 + \alpha_1.\epsilon_{t-1}^2 + \gamma.\epsilon_{t-1}^2.I_{|\epsilon t-1<0|} + \beta_1.\sigma_{t-1}^2 \tag{2.9}$$

Limitations:

1. Deciding the appropriate lag structure requires careful empirical testing and can lead to model selection issues.

2.1.5 Absolute value GARCH

[8] The model is comparable to the conventional Generalized ARCH model, which forecasts volatility by considering volatility and prior returns' absolute values.

$$\sigma_t = \omega + \sum_{i=1}^{q} \alpha_i |r_{t-i}| + \sum_{i=1}^{p} \beta_i |\sigma_{t-i}| \tag{2.10}$$

2.1.6 GJR GARCH

An extension of the Generalized-ARCH model. It specifically addresses the phenomenon of the leveraging effect, where adverse shocks to the returns on the financial asset raise future volatility more than positive shocks of the same magnitude. Conditional Variance in GJR-GARCH(p,o,q) is given by the expression:

$$\sigma_t^2 = \omega + \sum_{i=1}^{q} \alpha_i.r_{t-i}^2 + \sum_{i=1}^{o} \gamma_i.r_{t-i}^2.I_{r_{t-1}<0} + \sum_{i=1}^{p} \beta_i.\sigma_{t-i}^2 \tag{2.11}$$

where
p is the order of GARCH, o is the leverage term, and q is the order of ARCH.

Model	Limitations
GARCH	Assumes symmetry and ignores negative and positive shocks
EGARCH	Computationally expensive due to the presence of logarithm in variance.
TGARCH	It can overfit complex time series data.
AVGARCH	It is sensitive to large shocks because of the absolute value specification.
AVGARCH	It is sensitive to large shocks because of the absolute value specification.
GJRGARCH	More parameters are needed to estimate the complex model.
FIGARCH	Complex to implement and interpret.

Table 2.1: Limitations of the GARCH model

2.1.7 FIGARCH

Fractional Integrated GARCH model. The conditional variance dynamic is given by [9]

$$\sigma_t = \omega + [1 - \beta.L - \phi.L(1-L)^d].\epsilon_t^2 + \sigma.h_{t-1} \tag{2.12}$$

L is the lag operator, and d is the fractional differencing parameter.

2.2 Time Series Algorithms

The following sub-section contains the time series algorithms used for forecasting volatility.

2.2.1 Naive forecast

It is the most straightforward forecasting approach. As mentioned in [10], this model assumes that volatility observed in the past will also persist into the next period. For example, If the volatility of the stock was 5 % yesterday, then the naive forecast model also assumes that it will be 5 % today as well.

$$\sigma_{t+1} = \sigma_t \tag{2.13}$$

where σ_{t+1} represents volatility of the time period t+1 i.e next volatility period and σ_t represents volatility of the time period t.

Limitations:

1. This model does not capture the volatility clustering phenomenon or adjust the forecast in response to the new information. So it shows poor performance in the changing conditions.

2.2.2 Moving Average forecast

The model uses the [11] average of the historical data points to forecast future values. The historical data can be of any length.

$$\sigma_{t+1} = \frac{1}{N} \sum_{i=t-n+1}^{t} \sigma_i \tag{2.14}$$

LHS represents volatility for t+1 time period, and RHS represents volatility for period t. Limitations:

1. This model is a lagging indicator, which means it relies entirely on historical data. Due to this, the model is slow to respond to the recent changes.
2. Choice of window length and poor performance in nonstationary markets.

2.2.3 Holt Winter Exponential Smoothing

The technique is an extension of exponential smoothing used to capture seasonality in addition to the level and trend components in the data. This method is not a traditional approach to calculating volatility but can be adopted under certain conditions. [12] There are three components in this method

1. Level Component: It is the average value inside the series.
2. Trend Component: This is the series value rising or falling.
3. Seasonality Component: The recurring short-term cycle in the series.

Formulation: The model has three equations for level, trend, and seasonal components.

1. Level Equation: It updates the level components considering the latest observations and the previous level and trend components.

2. Trend Equation: It updates the trend equation components according to the trend in the previous period.

3. Seasonal Equation: It updates the seasonal components based on the most recent period.

There are two types of variants in this exponential smoothing [12]

1. Additive method: Used when the series's seasonal variants are roughly constant.

2. Multiplicative method: Used when seasonal variants change proportionality to the series level.

Additive model:
1. Level Equation:

$$L_t = \alpha(Y_t - S_{t-s}) + (1 - \alpha)(L_{t-1} + T_{t-1} \tag{2.15}$$

2. Trend Equation:

$$T_t = \beta(L_t - L_{t-1}) + (1 - \beta)(T_{t-1}) \tag{2.16}$$

3. Seasonal Equation:

$$S_t = \gamma(Y_t - L_t) + (1 - \gamma)(S_{t-s}) \tag{2.17}$$

4. Forecast Equation

$$Y_{t+h} = L_t + hT_t + S_{t-S+1+(h-1)modS} \tag{2.18}$$

Limitations:

1. Volatility may not exhibit clear and consistent seasonal patterns, a core assumption of the Holt winter exponential smoothing.

2. Stationary requirement: May not react quickly to sudden market shocks or

irregular events as it depends on past patterns to forecast future values.

2.2.4 ARIMA

Autoregressive Integrated Moving Average forecast: A statistical approach for time series forecasting, famous for forecasting future values in a series from its past values. There are two components involved in this model: As mentioned in [13]

1. AR(Auto Regressive part):
   (I) This part shows the relationship between the variable and its lagged values.
   (II) An auto-regressive term with order p implies that the current values are related to its p previous value of the series.

2. I(Integrated part):
   (I) This part adds the differences to make the series stationary. The order of differencing is denoted as d.

3. MA(Moving Average part):
   (I) Captures the association between a series and the residual error from a moving average model applied to past observations.
   (II) An MA term of order q involves using the error terms of the model's lagged forecast.

Limitations:

1. ARIMA models assume constant variance over time, which is impossible in finance time series data.

2. The model does not account for the asymmetric effects of leverage in which positive and negative shocks affect the volatility differently.

3. Selecting the appropriate order of ARIMA(p,o,q) can be complex as it requires careful analysis.

2.2.5 Theta forecast

The theta forecast decomposes the time series into two components:
1. A smoothing component.
2. A trend component.
The smoothing component captures the most stable underlying level of the series, and the trend component captures the direction and pace of the series over time. In [14], the original time series is first de-trended by applying simple or double exponential smoothing, resulting in a smoothed series. Then, it involves adjusting the curvature of the time series by applying different theta coefficients, which essentially modify the second difference in the series. A theta value of zero corresponds to a linear trend, while other values allow for capturing nonlinear trends. Finally, the forecasts are generated by recombining the smoothed and trend components.
Limitations:

1. Assumptions of consistent patterns.
2. Lack of responsiveness.
3. Ignores external factors.
4. Simplicity and over-generalization.

2.2.6 NBEATS

NBEATS is a time series forecasting algorithm which was developed in 2020. Initially, it was used to predict time series datasets. Later, it was used for cryptocurrency prediction, volatility prediction, etc., so the application of NBEATS gradually transitioned from being applied on average time series data to real market data.
In this algorithm [15], they have defined that the backcast is used during training, where the model attempts to generate outputs that match the given input data as closely as possible. In this way, the model trains to capture the underlying trends and seasonality of the given data, and then the output is forwarded to forecast.
The following is the procedure for how the NBEATS algorithm works:

1. The input to the model is a window of historical time series data.

2. The model uses its layers to produce two outputs:
   (I) A forecast for future points.
   (II) A backcast for the input window.

3. The backcast is compared to the actual historical data, and the difference, i.e., residue, is used to adjust the model parameters through backpropagation. Then, the residue is passed to the next block as the input for further learning.

In [15], the model is characterized by a unique architecture known as a double branch structure, which consists of two paths:
1. A trend stack
2. A seasonality stack

Trend stack:
1. It is designed to model and forecast long-term progress in the data by learning a set of basic functions that represent different possible trends in the time series data.
2. The stack consists of a series of fully connected feed-forward neural network blocks, each producing a forecast and a backcast.
3. The forecast output is the model prediction of the future values, and the backcast is used to reconstruct the backcast values.
4. The outputs of these blocks are combined to produce the trend forecast.

Seasonality trend:
1. Operates parallel to the trend stacks and is responsible for capturing periodic fluctuations that occur at regular intervals.
2. It is also very similar to a trend stack and comprises fully connected feed-forward neural blocks, each specializing in modeling different seasonal patterns.

Double branch Integration
1. The final output of the NBEATS is the sum of the forecast of both the trend and seasonality stacks. This allows the model to simultaneously consider and combine long-term trends and repetitive patterns in the time series data.
The following is the architecture of the NBEATS algorithm, which consists of backcast and forecast and also blocks with stack layers:

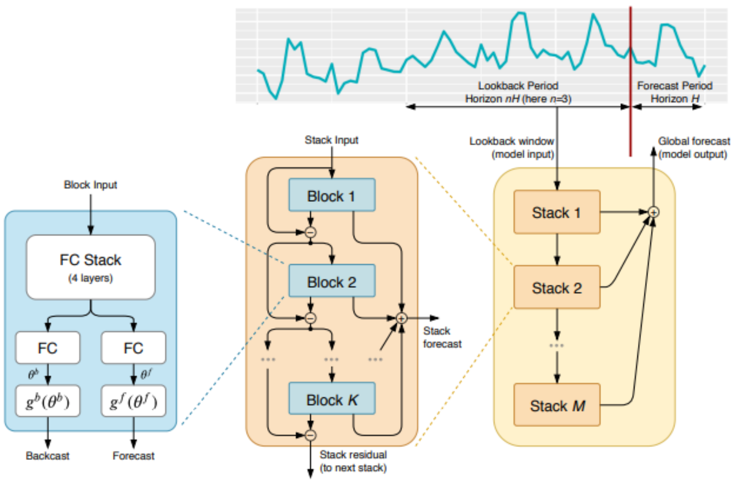


Figure 2.1: NBEATS architecture horizontal look

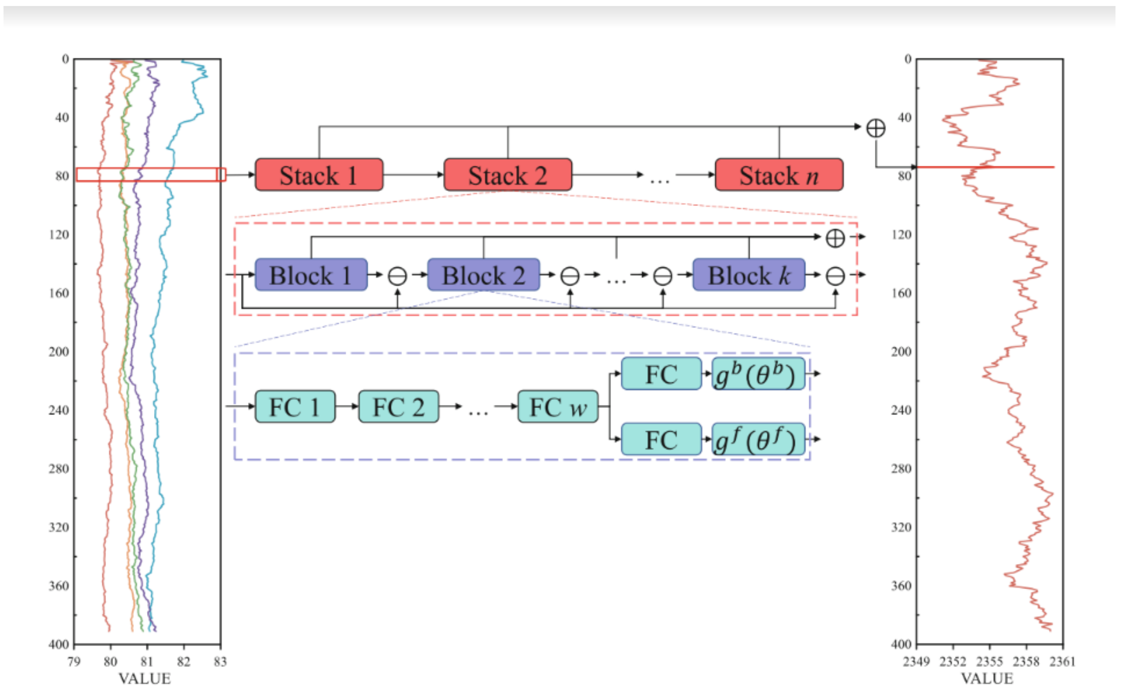


Figure 2.2: NBEATS Architecture vertical look

Till now, we have seen Econometric models and Time Series Algorithms and the limitations of each of them. Now, let us focus on the central part, where data science plays a crucial role in predicting volatility in the stock market.

2.3 Machine Learning and Deep Learning Algorithms

2.3.1 LSTM(long short term memory)

In addition, as shown in [16], we intend to create an LSTM neural network to recurrently predict stock values during volatility with a short forward-looking time. Stock prices are normalized to lie between zero and one. An exact-shaped two-dimensional array serves as the LSTM's input.
shapes of the array = Samples x look-back
Samples denote the total number of data points and look back denotes the time (up to the look-back) the LSTM will take to provide a prediction. We will predict volatility using a dropout of 0.2. The process for forecasting stock market volatility is as follows.

1. Load the dataset.
2. Set the stock prizes to a uniform value of one to zero.
3. From the test dataset, extract segments of 60 observations each and stack these segments sequentially. Each row in the resulting structure will consist of 60 steps.
4. Make the required predictions.
5. After the required estimation, we will use inverse transform to return the stock prices.

Limitations: This model cannot find trends and seasonality in the financial market.

2.3.2 DSVM (Deep Stochastic Volatility Model)

These models operate within the deep latent variable model framework, employing adaptable deep learning techniques to autonomously identify how past returns, volatility, and similar factors influence future volatility. The Deep Stochastic Volatility Model (DSVM) excels over many well-known alternative volatility models, particularly in high-risk market conditions. We have used one-step-ahead forecasting as did in [17] to predict future values.
Let σ_t represent the asset's volatility at time step t, and let r_t represent the asset return at that time. We also include z_t since the volatility at time step t is influenced by continuous stochastic noise. It is assumed that this stochastic noise has a normal distribution and that earlier stochastic noises impact its properties.[17]

$$z_t = (N(m_t^{(p)}, (v_t^{(p)})^2) \tag{2.19}$$

$$m_t^{(p)} = f_1(z_{t-1}), v_t^{(p)} = f_2(z_{t-1}) \tag{2.20}$$

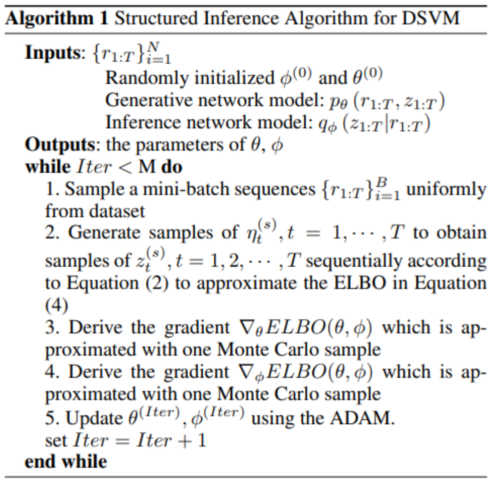
where f_1, f_2 rely on z_{t-1}, and z_t is a gaussian variable with $m_t^{(p)}$, v_t^p provided by the MLP models.

$$h_t = f_h(\sigma_{t-1}, r_{t-1}, z_t, h_{t-1}) \tag{2.21}$$

$$\sigma_t = f_3(h_t) \tag{2.22}$$

MLP model is indicated by f_3 and an RNN model by f_h. As a result, returns, stochastic noises, and prior volatility all influence present volatility, as observed in the above equation. As maximum likelihood estimation is computationally and theoretically complex, we will not utilize it to estimate the parameters. Instead, we take the optimal lower bound of variational evidence, or ELBO(θ, ϕ), and maximize it, where ϕ is the chosen posterior parameter. The ELBO is an equation as follows

$$\begin{aligned} \text{ELBO}(\theta, \phi) = \sum_{t=1}^{T} \Big[\log p_\theta(r_t|\sigma_t) = f_\sigma(\sigma_{1:t-1}, r_{1:t-1}, z_{1:t}^{(s)}) \Big] \\ - \sum_{t=1}^{T} \Big[KL \Big[q_\phi(Z_t|Z_{t-1}^{(s)}) || p_\theta(z_t|z_{t-1}^{(s)}) \Big] \Big] \end{aligned} \tag{2.23}$$



```
Algorithm 1 Structured Inference Algorithm for DSVM
  Inputs: $\{r_{1:T}\}_{i=1}^{N}$
          Randomly initialized $\phi^{(0)}$ and $\theta^{(0)}$
          Generative network model: $p_\theta\left(r_{1:T}, z_{1:T}\right)$
          Inference network model: $q_\phi\left(z_{1:T} | r_{1:T}\right)$
  Outputs: the parameters of $\theta$, $\phi$
  while $Iter < $ M do
    1. Sample a mini-batch sequences $\{r_{1:T}\}_{i=1}^{B}$ uniformly
    from dataset
    2. Generate samples of $\eta_t^{(s)}, t = 1, \cdots, T$ to obtain
    samples of $z_t^{(s)}, t = 1, 2, \cdots, T$ sequentially according
    to Equation (2) to approximate the ELBO in Equation
    (4)
    3. Derive the gradient $\nabla_\theta ELBO(\theta, \phi)$ which is ap-
    proximated with one Monte Carlo sample
    4. Derive the gradient $\nabla_\phi ELBO(\theta, \phi)$ which is ap-
    proximated with one Monte Carlo sample
    5. Update $\theta^{(Iter)}, \phi^{(Iter)}$ using the ADAM.
    set $Iter = Iter + 1$
  end while
```

Figure 2.3: The algorithm presents the step-wise DSVM implementation to predict the volatility.

Limitations:

1. Computationally complex because variational inference is chosen, which is tough to train.

2. Model specification and sensitivity: In choosing appropriate prior.

The rolling window technique will fit the upcoming architectures [18], widely applied in finance and other fields. The model is fitted using a fixed sample length, and the subsequent time step is forecasted. Specifically, the window size is established at 650, with a forecast horizon of one day. This approach involves fitting the model using data from the most recent 650 days to forecast the volatility for the following day. This process is repeated until volatility predictions have been made for the entire period under analysis.
Also, in the upcoming hybrid algorithms, the GARCH model implies the combination of GARCH and its variants such as EGARCH, TGARCH, GJR-GARCH, Abs-GARCH, FI-GARCH.
As multi-transformers, transformers, and LSTM layers can handle the time series, fitting the layers involves considering a lag of the last ten observations of the previous variables. Thus, the input variables are:
1. $X_1 = (\sigma_{t-1}, \sigma_{t-2}, ..., \sigma_{t-10})$

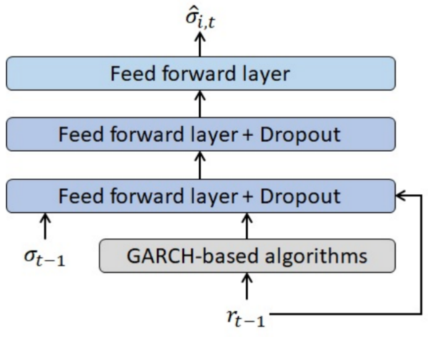


Figure 2.4: The ANN-GARCH model's flow chart is shown in the figure.

2. $X_2 = (r_{t-1}, r_{t-2}, ..., r_{t-10})$
The realized volatility is used as the response variable

$$Y = \sigma_{i,t} = \sqrt{\frac{\sum_{n=0}^{i-1}(r_{t+n} - E[r_f]^2}{i-1}} \tag{2.24}$$

where

$$E[r_f] = \sum_{n=0}^{i-1} \frac{r_{t+n}}{i} \tag{2.25}$$

Thus, the realized volatility is the standard deviation of future logarithmic returns.

2.3.3 ANN-GARCH:

ANN GARCH [1] is a hybrid model that combines an artificial neural network and six different GARCH models, which include GARCH, GJR-GARCH, TGARCH, EGARCH, AVGGARCH, and FIGARCH. The output of the GARCH models is the conditional variance of the Garch models and their parameters, with past volatility given as input to the feed-forward layers. The neNetworktwork consists of 2 different dense layers, one with 16 neurons and the other with eight neurons, where a dropout of 0.15 is added to each dense layer to prevent overfitting. The model is trained using the ADAM optimizer and the Binary Entropy loss. The final feed-forward layer is the conditional variance.
Limitations:

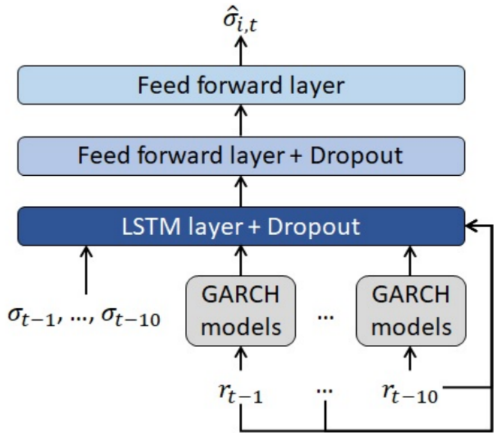


Figure 2.5: The lstm-garch model's flow chart and operation are depicted in the above figure.

1. Cannot estimate accurate temporal dependency in the data.

2.3.4 LSTM-GARCH

It is a hybrid model developed by [1] and is formed using the LSTM and GARCH models. The input to the LSTM layer is the volatility estimates from all six GARCH models, past returns, and past volatility. Here, we will be using a lag value of 10, which means the network uses data from 10 previous predictions to make the current step prediction. The LSTM layer is used here to capture the temporal dependencies, and feed-forward layers are used to capture further pattern recognition and make predictions. The size of the lstm layer is 32 units, indicating the output space's dimensionality. The dimensionality of the two feed-forward layers is eight neurons and one neuron, respectively.
Limitations:

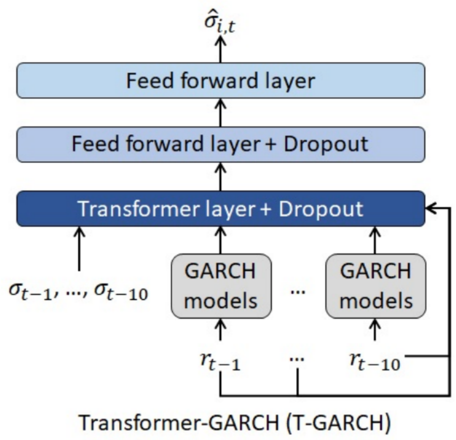
1. Long-term dependencies: Short-term memory is developed to capture long-term dependencies but struggles to capture long-term dependencies when the relevant information is spread over significant time gaps.
2. Volatility clustering and parameter estimation.
3. Financial data show regime shifts, as the process creates dates that require a long time for LSTM to adjust.

2.3.5 Transformer based models

Originally developed for natural language processing (NLP), transformer layers require certain adaptations for volatility forecasting. Unlike LSTM models, transformers do not incorporate recurrence within their architecture. Instead, these layers utilize two primary components tailored to handle time series data:
1. Positional Encoder
2. Multi-head Attention

1. Positional encoder: As mentioned earlier, transformer layers lack recurrence, necessitating the inclusion of information about the relative positions within the time series. To address this, positional encoding is incorporated into the input time data. [19] the following wave function is employed as a positional encoder:
$$PE_{pos,2_i} = sin(pos/1000^{2i/dim}) \tag{2.26}$$
$$PE_{pos,2_{i+1}} = cos(pos/1000^{2i/dim}) \tag{2.27}$$
where I = (1,2,...., dim-1) and pos denotes the observation's position inside the time series data. Since the volatility model does not accept words as input. This positional encoding alters the input data according to the time series lag and the **embedding dimensions used for the words**. Adjustments are made to the positioning encoder to prevent deviations. As a result, the positional encoding values remain constant across various explanatory factors but vary according to the lag value.
$$PE_{pos} = Cos(\pi.\frac{pos}{N_{pos}-1}) \tag{2.28}$$

2. Multi-head attention: Proposed by [19], Multiple scaled dot-product attention units operating in parallel make up the multi-transformer. The computation of scaled dot product attention is as follows:
$$Attention(Q,K,V) = softmax(\frac{Qk^T}{\sqrt{d_k}})V \tag{2.29}$$



$\hat{\sigma}_{i,t}$
Feed forward layer
Feed forward layer + Dropout
Transformer layer + Dropout
$\sigma_{t-1}, \ldots, \sigma_{t-10}$
GARCH models
...
GARCH models
r_{t-1}
...
r_{t-10}
Transformer-GARCH (T-GARCH)

Figure 2.6: The Transformer-GARCH model's flow chart and operation are depicted in the above figure.

Q, K, and V are the input metrics; d_k is the number of input variables considered. This layer separates the variables into distinct heads to execute the multiple-scaled dot product attention in units simultaneously. The outputs are then concatenated after distinct heads are calculated. According to [19], the transformer layer consists of a feed-forward layer with ReLu activation, the multi transformers uses this layer as multi head attention and then normalization takes place for final volatility prediction. As mentioned in [2], the transformer layer has two residual connections; because of this, the model will automatically decide at key points throughout the fitting process whether or not the training of specific layers needs to be avoided. The purpose of this modified TransfoTransformerforecast volatility.

As discussed in the above section, the Transformer-GARCH architecture combines six different structures, integrating TransfoTransformered-forward layers to predict volatility, and in the Transformer-LSTM-GARCH model, which includes an LSTM with 32 units, the LSTM layer recognizes and models the temporal structure of the data, eliminating the need for a positional encoder in the transformer layer.

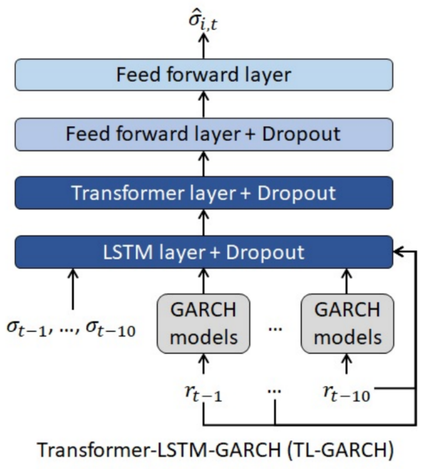

$\hat{\sigma}_{i,t}$
Feed forward layer
Feed forward layer + Dropout
Transformer layer + Dropout
LSTM layer + Dropout
$\sigma_{t-1}, ..., \sigma_{t-10}$
GARCH models
...
GARCH models
r_{t-1}
...
r_{t-10}
Transformer-LSTM-GARCH (TL-GARCH)

Figure 2.7: The Transformer-lstm-garch model's flow chart and operation are depicted in the above figure.

2.3.6 Multi Transformer based models

This study uses a variant of the transformer layer, known as the multi-transformer-based architecture [19]. The primary distinction is that in this version, T unique random input data samples are generated, each randomly selected for processing in T different attention layers inside the TransfoTransformerinal attention matrix is obtained by computing the average of the units. Thus, the following definition can be applied to a typical multi-head mechanism of a multi-transformer:

$$AMH(Q, K, V) = \frac{\sum_{t=1}^{T} Concat(head_{1,t}, ..., head_{h,t})W_t^O}{T} \tag{2.30}$$

$$head_{i,t} = Attention(Q_t W_{i,t}^Q, k_t W_{i,t}^K, V_t W_{i,t}^V \tag{2.31}$$

To increase stability and precision, the multi-transformer layers use bagging [20] to the attention mechanism. This approach applies bagging only to the layered architecture's attention mechanism, not to all of the data supplied as input to the model.

Advantages:

1. Bagging preserves the bias.

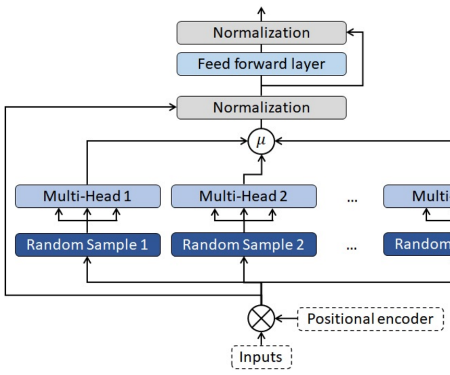


Figure 2.8: Multi Transformer Architecture

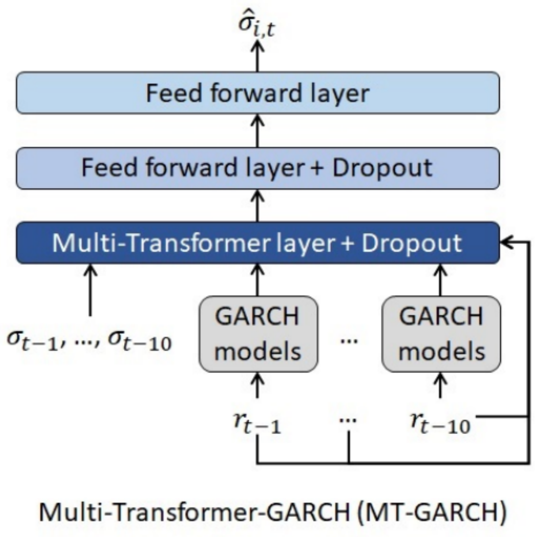
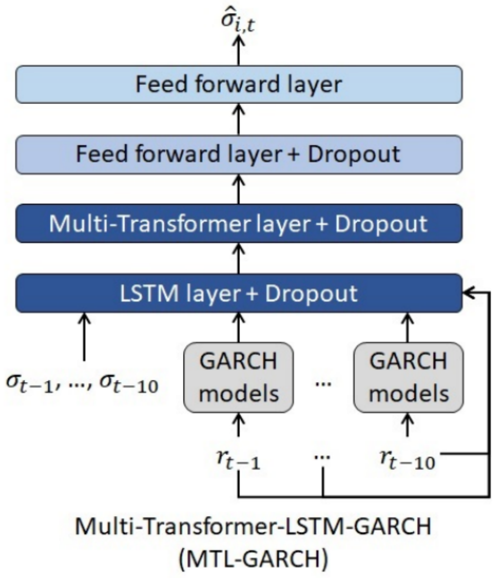

$\hat{\sigma}_{i,t}$
Feed forward layer
Feed forward layer + Dropout
Multi-Transformer layer + Dropout
$\sigma_{t-1}, \ldots, \sigma_{t-10}$
GARCH models
...
GARCH models
r_{t-1}
...
r_{t-10}
Multi-Transformer-GARCH (MT-GARCH)

Figure 2.9: The workflow of the MT GARCH algorithm



$\hat{\sigma}_{i,t}$
Feed forward layer
Feed forward layer + Dropout
Multi-Transformer layer + Dropout
LSTM layer + Dropout
$\sigma_{t-1}, \ldots, \sigma_{t-10}$
GARCH models
...
GARCH models
r_{t-1}
...
r_{t-10}
Multi-Transformer-LSTM-GARCH
(MTL-GARCH)

Figure 2.10: The workflow of the MTL GARCH algorithm

2. Bagging significantly reduces the error variance, leading to high accuracy and lowering the risk of overfitting.

Limitation:

1. Computational power needed for bagging is the only limitation of this technique.

2.4 Proposed Hybrid Model for Volatility Prediction

2.4.1 Nbeats Plus Multi-Transformer LSTM

The concept originated from insights gained through [21]. The developed algorithm merges the predictive capabilities of a time series forecasting model with the advanced techniques of deep learning. Prior analysis revealed that the foundation of N-BEATS is the progressive adjustment of input residuals conveyed across its layered structure [22]. Input traverses a series of 'n' blocks within the N-BEATS

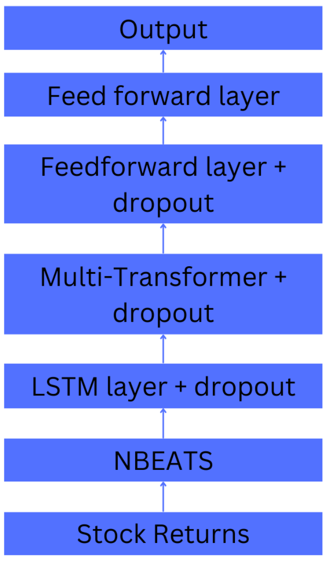


Figure 2.11: The workflow of NBEATS plus MTL algorithm

framework, each providing dual outputs: the forecast and the backcast. [15] The backcast essentially represents the residual—differences between the predicted and the actual input—which is then relayed forward for additional refinement. Conversely, the forecast reflects the predictive output for future values, and these forecasts are aggregated from each block to form the final predictive output.

Transformers were designed for natural language processing, so it is necessary to make certain changes to their application to our dataset. The transformer layers are central to the proposed architecture, with the multi-head attention mechanism as its foundation. This multi-head mechanism divides the input data into separate segments, facilitating parallel operations of the scaled dot-product attention units. Post-processing involves concatenating outputs from various heads, followed by normalization and passage through a feed-forward layer.

Similarly, [19] the Multi-Transformer creates T different random samples from the input, which are then processed through multiple multi-head attention units.

The outputs of these units are averaged to formulate the final attention matrix. Subsequently, this matrix feeds into a forward layer with an activation function, culminating in market volatility prediction. In the multi-transformer bagging technique is applied only to attention layers.
The proposed model has the following characteristics:

1. ADAM algorithm [23] is used to update the weights of the respective layers.
2. The feed-forward layer has a dropout layer.
3. The loss function used in Nbeats was the Minimum Quantile loss function where, as in the remaining architecture, Mean Square Error.
4. Batch size, which we used, is 64, and the number of epochs to train the model is 50.

The only limitation of the proposed algorithm is it requires substantial computational power as the algorithm uses the bagging technique [20].

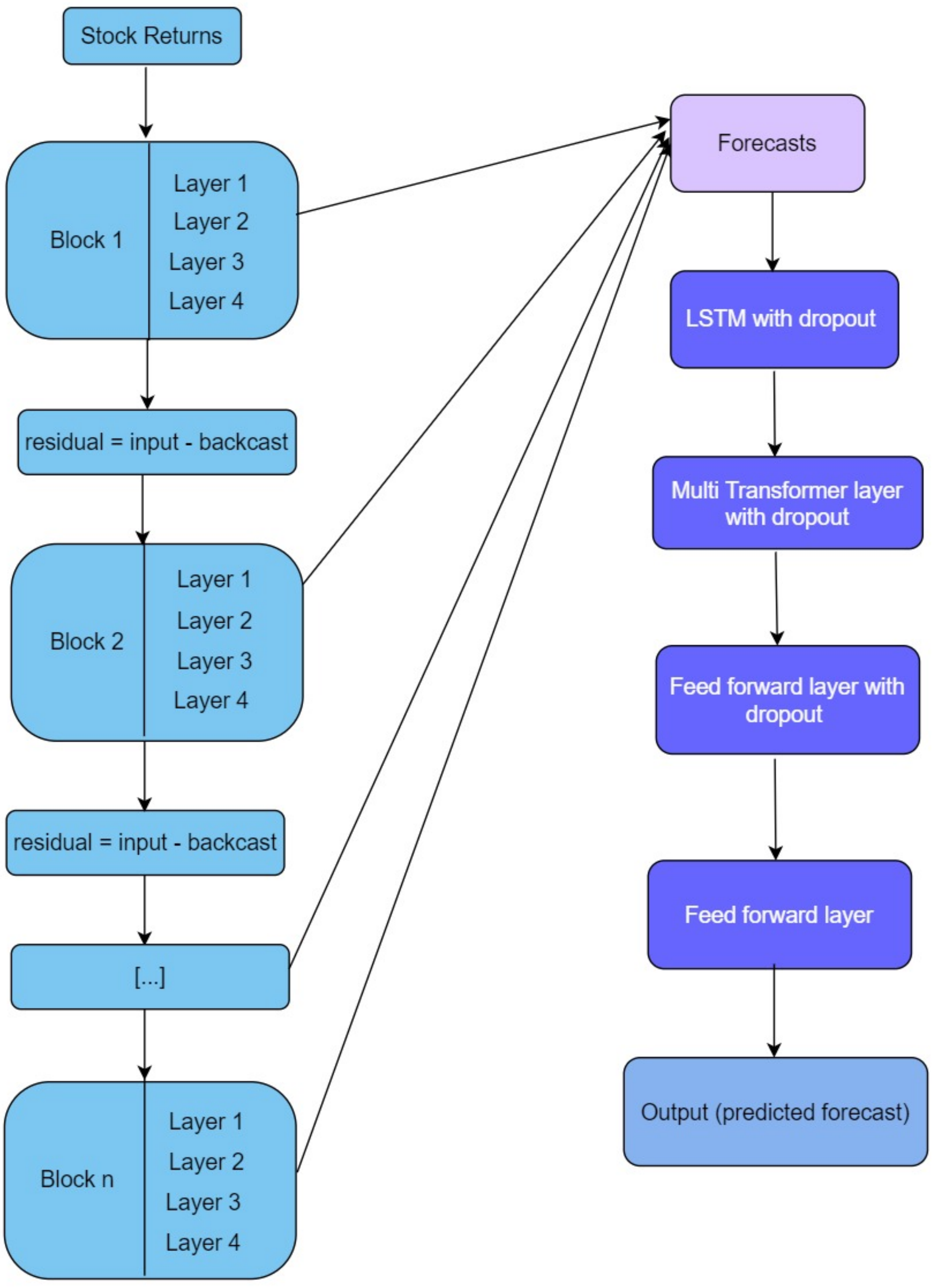


Figure 2.12: NBEATS MTL workflow

Chapter 3

Experiment Results and Analysis

3.1 Dataset

To predict the volatility of a stock market, we have used real-world market data, which is available in Yahoo Finance.We have used three different datasets: a bank dataset, and two stock indexes. We have used the Bajaj Finance dataset, which we can download from the website, Investing India, S and P 500 and Nifty 500 are the two stock indexes which we have used for Volatility prediction. The data of these indexes are directly extracted from Yahoo finance.

3.2 Evaluation metrics

In this section we discuss about the evaluation metrics which have been used for comparative study and the results of the dataset.

Data Set	Training	Testing
Bajaj	01-01-2000 to 30-12-2022	02-01-2023 to 01-01-2024
Nifty 500	01-01-2012 to 31-12-2016	01-01-2017 to 21-01-2024
S and P 500	01-01-2008 to 31-12-2015	01-01-2016 to 22-01-2024

Table 3.1: Table above shows the training and testing periods of the datasets which we have used for volatility prediction.

We have used three different evaluation metrics, they are RMSE, MSE and MAE.

1. RMSE: Root mean square error, can be expressed as the square root of the average square of the discrepancies between the predicted and actual values.

$$RMSE = \sum_{t=1}^{N} \frac{(\sigma_{i,t} - \hat{\sigma_{i,t}})^2}{N} \tag{3.1}$$

2. MSE: Mean square error, is the mean of the square of the variations between the predicted volatility and the actual volatility.

3. MAE: Mean absolute error, is the average absolute difference between the predicted volatility and the actual volatility.

$$MAE = \sum_{t=1}^{N} \frac{|\sigma_{i,t} - \hat{\sigma_{i,t}}|}{N} \tag{3.2}$$

where $\sigma_{i,t}$ implies the actual volatility, $\hat{\sigma_{i,t}}$ implies the predicted volatility and N is the number of observations.

3.3 Results and Analysis

We have used dropout value of 0.15 for all the models so that it will be easy to compare with the bench mark models. The lower the rmse, mse and mae values the better the model has performed in predicting the volatility of the stock market.

1. Models based on merging Artificial neural network, LSTM with GARCH models outperformed when compared to traditional GARCH models and Time series forecasting models.

2. NBEATS algorithm outperformed well when compared to traditional GARCH models.

3. Multi transformer LSTM and Transformer LSTM performed well compared to merging models such as lstm garch, ann garch and transformers.

Model	RMSE	MSE	MAE
Naive	1.24	1.53	0.86
Moving Average	1.24	1.53	0.86
Holt winter	1.62	2.63	1.35
ARIMA	1.23	1.53	0.86
Theta forecast	1.62	2.63	1.35
ARCH	0.1692	0.0286	0.1684
GARCH	0.1589	0.0253	0.1584
EGARCH	0.1462	0.0214	0.1453
TGARCH	0.1564	0.0245	0.156
LSTM	0.1878	0.0353	0.1358
Transformers	0.61	0.37	0.5
ANN GARCH	0.0181	3.31E-04	0.01464
LSTM GARCH	0.0079	6.34E-05	0.00547
TGARCH	0.0078	6.13E-05	0.00540
MT GARCH	0.0080	6.54E-05	0.00573
TL GARCH	0.00754	5.69E-05	0.00533
MTL GARCH	0.00599	3.58E-05	0.005289
NBEATS	0.025	0.000631	0.0154
NBEATS + MTL	**0.00417**	**1.71E-05**	**0.004038**

Table 3.2: The table above shows the evaluation results of all the applied models on the Bajaj dataset.

Model	RMSE	MSE	MAE
NBEATS	0.015383	0.000237	0.006207
MTL GARCH	0.013796	0.000190	0.01100
NBEATS MTL	**0.003744**	**1.402E-05**	**0.003646**

Table 3.3: The table shows the rmse, mse and mae values for three different models for the Nifty 500 index.

Model	RMSE	MSE	MAE
NBEATS	0.024992	0.000623	0.014568
MTL GARCH	0.013253	0.000176	0.010561
NBEATS MTL	**0.003810**	**1.45E-05**	**0.003620**

Table 3.4: The table shows the rmse, mse and mae values for three different models for the S and P 500 index.

4. The proposed algorithm NBEATS plus Multi transformer LSTM outperformed all the existing models for all the datasets used in the experimentation.

Chapter 4

Conclusion and Future work

4.1 Conclusion and Future Work

The objective of this thesis is to build a model which predicts the volatility accurately when compared to existing models. Here we introduced NBEATS plus MTL for forecasting the realized volatility. The study does so by comparing NBEATS plus MTL forecast performance with recently introduced NBEATS algorithm and MTL GARCH algorithm which is a hybrid model. Three datasets have been used for the investigation and the NBEATS plus MTL outperformed well when compared to the existing models and the NBEATS plus MTL have improved by a factor of 10 when compared to MTL GARCH, but the only limitation of this model is the computational power, so in future we can introduce such models which consume less computational power and there is also possibility of model which is formed as a result of NBEATS and Econometric model for better evaluation when compared to existing models.

Bibliography

[1] Eduardo Ramos-Pérez, Pablo J Alonso-González, and José Javier Núñez-Velázquez. Multi-transformer: A new neural network-based architecture for forecasting s&p volatility. *Mathematics*, 9(15):1794, 2021.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[3] Tim Bollerslev, Robert F Engle, and Daniel B Nelson. Arch models. *Handbook of econometrics*, 4:2959–3038, 1994.

[4] Gary GJ Lee and Robert F Engle. A permanent and transitory component model of stock return volatility. *Available at SSRN 5848*, 1993.

[5] Markus Haas, Stefan Mittnik, and Marc S Paolella. A new approach to markov-switching garch models. *Journal of financial Econometrics*, 2(4):493–530, 2004.

[6] Isiaq O Oseni and Philip I Nwosa. Stock market volatility and macroeconomic variables volatility in nigeria: An exponential garch approach. *Journal of Economics and Sustainable Development*, 2(10):28–42, 2011.

[7] Jing Wu. Threshold garch model: Theory and application. *The University of Western Ontario*, pages 1–42, 2010.

[8] Markus Haas. Persistence in volatility, conditional kurtosis, and the taylor property in absolute value garch processes. *Statistics & probability letters*, 79(15):1674–1683, 2009.

[9] Maryam Tayefi and TV Ramanathan. An overview of figarch and related time series models. *Austrian journal of statistics*, 41(3):175–196, 2012.

[10] David Meyer. Naive time series forecasting methods. *R news*, 2(2):7–10, 2002.

[11] Seng Hansun. A new approach of moving average method in time series analysis. In *2013 conference on new media studies (CoNMedia)*, pages 1–4. IEEE, 2013.

[12] Prajakta S Kalekar et al. Time series forecasting using holt-winters exponential smoothing. *Kanwal Rekhi school of information Technology*, 4329008(13):1–13, 2004.

[13] Jamal Fattah, Latifa Ezzine, Zineb Aman, Haj El Moussami, and Abdeslam Lachhab. Forecasting of demand using arima model. *International Journal of Engineering Business Management*, 10:1847979018808673, 2018.

[14] Vassilis Assimakopoulos and Konstantinos Nikolopoulos. The theta model: a decomposition approach to forecasting. *International journal of forecasting*, 16(4):521–530, 2000.

[15] Boris N Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. N-beats: Neural basis expansion analysis for interpretable time series forecasting. *arXiv preprint arXiv:1905.10437*, 2019.

[16] Yigit Alparslan and Edward Kim. Extreme volatility prediction in stock market: When gamestop meets long short-term memory networks. *arXiv preprint arXiv:2103.01121*, 2021.

[17] Xiuqin Xu and Ying Chen. Deep stochastic volatility model. *arXiv preprint arXiv:2102.12658*, 2021.

[18] Norman R Swanson. Money and output viewed through a rolling window. *Journal of monetary Economics*, 41(3):455–474, 1998.

[19] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[20] Leo Breiman. Bagging predictors. *Machine learning*, 24:123–140, 1996.

[21] Attilio Sbrana, André Luis Debiaso Rossi, and Murilo Coelho Naldi. N-beats-rnn: Deep learning for time series forecasting. In *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 765–768. IEEE, 2020.

[22] Hugo Gobato Souto and Amir Moradi. Introducing nbeatsx to realized volatility forecasting. *Expert Systems with Applications*, 242:122802, 2024.

[23] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.