Al Akhawayn University in Ifrane

Transformer-Based Time Series Forecasting for Stock Market Volatility

CSC 3347–01: Machine Learning

Fall 2025

Project Team

Haytam Raiss

Hiba Tallah Erraji

Mouad Wadi

Contents

1 Project Overview **3**

2 Project Requirements **3**

2.1 Survey of Existing Work 3

2.2 Project Objectives and Hypotheses 5

2.3 Datasets 6

2.4 System Architecture 7

3 Model Descriptions **8**

3.1 Long Short-Term Memory (LSTM) 8

3.2 Transformer Encoder 8

3.3 ARIMA Time-Series Model 9

3.4 GARCH Volatility Model 9

3.5 Model Selection 10

3.6 Evaluation Metrics 10

4 Results and Analysis **12**

4.1 Data Preparation 12

4.2 Model Training Setup 12

4.3 Baseline Results (5-Day Horizon, Level Target) 12

4.4 Hyperparameter Tuning (Transformer) 13

4.5 Longer Horizon and Alternative Target 15

4.6 Rolling Window Backtest and Regime Behaviour 15

4.7 Prediction Interval and Calibration 17

4.8 Comprehensive Experiment Summary 17

4.9 Regime-Specific Error Analysis 18

4.10 Prediction Interval Coverage 19

4.11 Overall Assessment 20

5 Conclusion **20**

6 Deployment **23**

A Appendix: Key References and Resources **25**

A.1 Key Transformer-Based Volatility References 25

A.2 Thirty Related Models, Datasets, and Systems 25

A.3 Reference Links 30

1 Project Overview

This project investigates whether modern deep learning models, in particular Transformer architectures, can improve short-horizon volatility forecasting for the S&P 500 index. The work begins from raw daily OHLCV data for a large universe of U.S. stocks and ETFs, then focuses on a curated S&P-500-centric dataset, `features_target`, which aggregates macroeconomic, options, and index variables engineered specifically for volatility prediction.

A complete time-series pipeline is implemented: data cleaning and feature engineering, sequence construction, model training (Transformer, LSTM, ARIMA, GARCH), systematic evaluation, and an interactive deployment exposing the best model for inference.

2 Project Requirements

2.1 Survey of Existing Work

Volatility forecasting has a long history in financial econometrics and, more recently, in machine learning. Early work focused on modelling conditional variance using parametric time-series models, while more recent research explores non-linear and data-driven approaches, including deep neural networks and attention-based architectures. This subsection summarizes key strands of the literature that motivate the design choices in this project: GARCH-type models, realized volatility and HAR models, traditional machine learning approaches, recurrent neural networks, and Transformers for financial time series.

GARCH and Classical Volatility Models

The starting point for most volatility work is the family of ARCH and GARCH models, which model conditional variance as a function of past squared returns and past variances. These models capture volatility clustering and mean reversion and remain widely used in practice for risk management and derivatives pricing. Numerous extensions have been proposed, including EGARCH, GJR-GARCH, and multivariate GARCH, to allow for leverage effects, asymmetries, and richer dynamics.

ARIMA and related linear time-series models have also been applied directly to volatility or squared returns. These models can capture autocorrelation but do not model conditional heteroskedasticity explicitly, so they are usually considered weaker baselines than GARCH

for volatility itself. Nevertheless, ARIMA remains a useful benchmark because it provides a transparent linear representation and allows for a clean comparison against more complex non-linear approaches.

Realized Volatility and HAR Models

The introduction of high-frequency data enabled realized volatility measures computed from intraday returns. This led to a rich literature on realized volatility forecasting, where the target is the sum of high-frequency squared returns over a day or longer horizon. A key result from this line of work is that realized volatility is highly persistent and exhibits long memory. To capture this behaviour, the Heterogeneous Autoregressive (HAR) model was proposed, decomposing volatility into daily, weekly, and monthly components. HAR-type models often provide strong benchmarks that are simple to estimate yet capture important time-scale effects.

Traditional Machine Learning for Volatility

With the growth of machine learning in finance, many authors have experimented with non-linear models such as decision trees, random forests, gradient boosting, and support vector regression for volatility prediction. These models flexibly capture interactions and non-linearities among explanatory variables. Empirical results typically show that tree-based ensembles can outperform simple linear regressions and sometimes rival or beat GARCH/HAR models when feature engineering is done carefully, although gains are usually modest and sensitive to sample period.

Recurrent Neural Networks and LSTMs

Recurrent neural networks, particularly Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) architectures, have been widely applied to financial time series. LSTMs can, in principle, capture long-term dependencies and non-linear interactions without manual lag engineering. Several studies report that LSTMs outperform classical time-series models on some datasets, especially when multiple correlated inputs (e.g., macro variables, technical indicators) are available. Other work highlights limitations: training instability, overfitting on small or noisy datasets, and improvements that are sometimes small relative to strong econometric baselines.

Transformers and Attention in Finance

More recently, attention mechanisms and Transformer architectures, originally developed for natural language processing, have been adapted to financial time series. In a Transformer, self-attention layers replace recurrence, allowing the model to weigh different time steps and features according to their relevance for the prediction task. Several studies show that Transformers can outperform LSTMs on price prediction, order book forecasting, and realized volatility modelling. Attention is particularly useful when the input includes heterogeneous signals, such as prices, volumes, options data, and macro indicators, with time-varying importance.

Regime Dependence and Limits of Predictability

A recurring theme across both econometric and machine learning literature is that volatility predictability is strongly regime-dependent. Many empirical studies find that models perform better in moderate-volatility periods and degrade during crises or extremely calm markets. Rolling-window and sub-sample analyses often reveal that performance metrics fluctuate significantly over time, sometimes turning negative when structural breaks occur. Overall, short-horizon stock-index volatility appears only partially predictable, even with sophisticated models and rich input features.

2.2 Project Objectives and Hypotheses

The task is a supervised **regression** problem. The main objectives are:

- Predict short-horizon S&P 500 volatility using daily macro, options, and market features.
- Compare deep learning models (LSTM, Transformer) against ARIMA and GARCH baselines.
- Analyze performance across volatility regimes and over time.

Targets include the S&P 500 30-day volatility level shifted 5 days ahead and the daily change in 30-day volatility shifted 5 and 20 days ahead.

Hypotheses:

- The Transformer will outperform LSTM, ARIMA, and GARCH in terms of RMSE, MAE, and R^2.

- Using the engineered `features_target` dataset instead of raw OHLCV will reduce prediction error.
- Volatility predictability will be regime-dependent, with better accuracy in medium-volatility regimes.

2.3 Datasets

The final dataset, `features_target`, is constructed from:

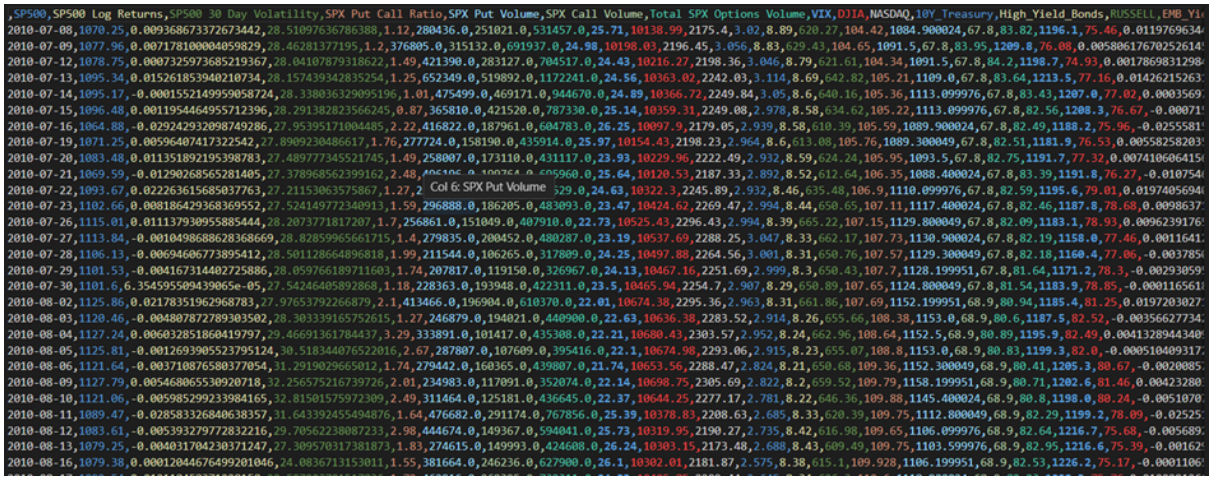
- Market indices: S&P 500 level and log returns, 30-day volatility, DJIA, NASDAQ, Russell 2000, MSCI World and their log returns.
- Options data: SPX put–call ratio and log changes of put, call, and total options volume.
- Macro and credit variables: 10-year Treasury yield, high-yield bond index, EMB yield.
- Commodities and FX: gold, oil, USD index and their log returns.
- Sentiment: consumer sentiment level and log change.

Labels are derived as:

- 5-day-ahead S&P 500 30-day volatility level.
- 5-day-ahead and 20-day-ahead changes in 30-day volatility.

After cleaning and dropping missing values, the dataset contains roughly 2,200 daily observations. Features are standardized using training-set mean and variance. Sequences of length 60 days are built to produce samples of shape $60 \times F$ with scalar volatility targets.

Time-based splits are used: 70% train, 15% validation, 15% test, plus additional rolling windows for robustness. The data involve no personal information, so privacy and fairness risks are minimal.



2.4 System Architecture

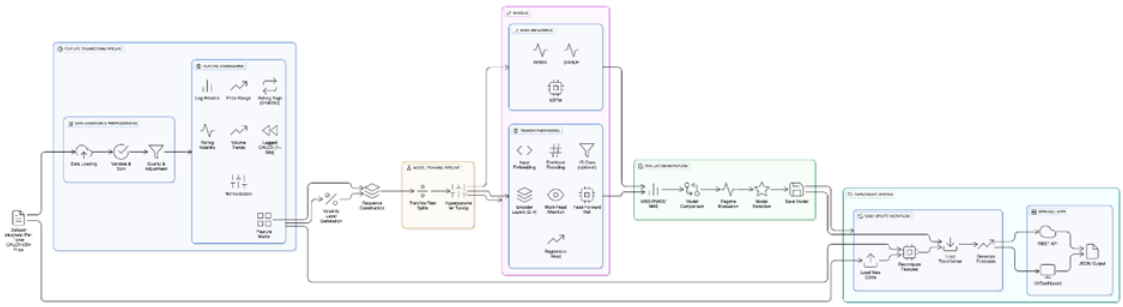
The system has four stages:

1. **Ingestion and preprocessing:** load `features_target`, clean columns, remove redundant raw volumes, handle missing values, and standardize numerical features.

2. **Feature engineering and sequence construction:** construct volatility level or change targets by shifting the series forward in time, then form 60-day rolling windows for model input; compute volatility-regime labels (low/medium/high) via quantiles for analysis.

3. **Model training pipeline:** implement LSTM and Transformer models in PyTorch with configurable hyperparameters; fit ARIMA and GARCH models using standard libraries; use mini-batch training, gradient clipping, and early stopping on validation loss with checkpointing of the best model.

4. **Evaluation and deployment:** compute metrics (RMSE, MAE, R^2), regime-specific scores, rolling-window performance, residual plots, Q–Q plots, and prediction-interval coverage; expose the best Transformer model through an interactive dashboard for inference.



3 Model Descriptions

This section briefly explains the main models used in the project: LSTM and Transformer as deep learning sequence models, and ARIMA and GARCH as classical time-series baselines.

3.1 Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) networks are a type of recurrent neural network designed to model sequential data. An LSTM processes an input sequence step by step and maintains a hidden state that is updated over time. Each LSTM cell contains three gates: an input gate, a forget gate, and an output gate. These gates control how much past information is kept, how much new information is added, and what part of the internal state is exposed at each time step.

In this project, the LSTM takes as input a 60-day window of features and produces a single forecast of future volatility (or volatility change). The architecture uses stacked LSTM layers followed by a fully connected output layer. LSTMs are well-suited to capturing non-linear temporal patterns and have been widely used in financial forecasting, but they can be difficult to train and may overfit when data are noisy or limited.

3.2 Transformer Encoder

The Transformer is a sequence model based entirely on attention mechanisms, without recurrence or convolution. Instead of processing observations one by one, a Transformer encoder looks at the entire 60-day window simultaneously. Each layer uses multi-head self-attention to compute weighted combinations of all time steps, allowing the model to focus on the most relevant days and features for the prediction. A position encoding or learned projection is used so that the model is aware of ordering in the sequence.

In this project, the Transformer encoder projects the input features into a latent space of dimension d_{model}, passes them through several self-attention and feed-forward layers, and then aggregates the final hidden state (typically the last time step) through a linear output layer to obtain a volatility forecast. Transformers handle long-range dependencies and heterogeneous features more flexibly than LSTMs and can parallelize computation across time, which makes them attractive for financial time series with many correlated signals.

3.3 ARIMA Time-Series Model

The Autoregressive Integrated Moving Average (ARIMA) model is a classical linear model for univariate time series. An ARIMA(p, d, q) model combines:

- an autoregressive (AR) component of order p, which regresses the series on its own lags,
- an integration (I) component of order d, which applies differencing to remove trends or non-stationarity,
- a moving-average (MA) component of order q, which models the error term as a linear combination of past shocks.

In this project, ARIMA is fitted to the volatility (or volatility-change) series alone, without additional exogenous variables. It provides a transparent benchmark that captures linear persistence but does not model time-varying variance explicitly. ARIMA is useful to show how much improvement is obtained by more flexible non-linear and attention-based models.

3.4 GARCH Volatility Model

The Generalized Autoregressive Conditional Heteroskedasticity (GARCH) model explicitly models the conditional variance of a time series. A GARCH(1,1) specification assumes that the conditional variance at time t depends on a constant term, the squared return (or volatility innovation) at time $t-1$, and the previous conditional variance at time $t-1$. This structure captures volatility clustering: periods of high volatility tend to be followed by high volatility, and similarly for low volatility.

In the experiments, a GARCH(1,1) model is estimated on the volatility (or volatility-change) series. The one-step-ahead conditional variance output is interpreted as the volatility forecast. GARCH serves as a strong econometric baseline because it is specifically designed for volatility dynamics and is widely used in risk management and option pricing. Comparing Transformer and LSTM forecasts against GARCH makes it possible to assess whether complex deep learning architectures provide added value beyond established statistical models.

3.5 Model Selection

Baselines:

- **ARIMA**: captures linear autocorrelation in the volatility or volatility-change series.
- **GARCH(1,1)**: models conditional heteroskedasticity and serves as a standard volatility benchmark.

Deep learning models:

- **LSTM**: two layers with hidden size 128, dropout 0.2 and a linear output head. The model is trained with Adam (learning rate 10^{-4}), batch size 64, and early stopping.
- **Transformer**: an encoder-only architecture with input projection to $d_{\text{model}} \in \{64, 128\}$, 2–3 layers, 4 attention heads, feed-forward dimension 128, and dropout 0.2–0.3. Grid search over d_{model}, depth, dropout, and learning rate identifies the best configuration $d_{\text{model}} = 128$, 2 layers, dropout 0.2, learning rate 10^{-4}.

3.6 Evaluation Metrics

Because the project is a regression task (forecasting continuous volatility), evaluation focuses on standard forecast-error metrics.

Mean Squared Error (MSE)

Definition: the average of squared differences between predicted and true volatility values:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2.$$

Justification: squaring penalizes large errors more than small ones, which is important in risk management because severely underestimating volatility is especially harmful. **Usage:** many realized-volatility and risk-forecasting studies report MSE or related measures when comparing econometric and machine learning models.

Root Mean Squared Error (RMSE)

Definition: the square root of MSE:

$$\text{RMSE} = \sqrt{\text{MSE}}.$$

Justification: expressed in the same units as the target volatility, making it easier to interpret, while still weighting large errors strongly. **Usage:** RMSE is one of the most common metrics in volatility-forecasting work and is used in many comparisons between GARCH, HAR, LSTM, and Transformer-based models.

Mean Absolute Error (MAE)

Definition: the average absolute difference between predicted and true values:

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i|.$$

Justification: less sensitive to outliers than MSE/RMSE and directly interpretable as the typical forecast error in volatility units. **Usage:** MAE is frequently reported alongside RMSE to give a more robust view of error magnitude and to highlight whether performance is driven by a few large mistakes.

Coefficient of Determination (R^2)

Definition: the proportion of variance in the target explained by the model relative to a constant-mean baseline:

$$R^2 = 1 - \frac{\sum_i(\hat{y}_i - y_i)^2}{\sum_i(y_i - \bar{y})^2}.$$

Justification: summarizes how much of the variability in volatility is captured by the model; negative values indicate performance worse than simply predicting the sample mean. **Usage:** many volatility-forecasting papers report R^2 in addition to error metrics to assess explanatory power across models trained on the same dataset.

4 Results and Analysis

4.1 Data Preparation

The project started from raw daily OHLCV data for many U.S. stocks and ETFs. After exploratory analysis, the focus moved to a single aggregate dataset called `features_target`, built for the S&P 500. This dataset contains engineered features such as index returns, 30-day volatility, option-based indicators (put–call ratio, options volume changes), macro and credit spreads, and commodity and FX variables. The final prediction target is the S&P 500 30-day volatility (or its daily change), shifted forward in time so that the models forecast future volatility using only information available at the prediction date.

4.2 Model Training Setup

The data are transformed into rolling windows of 60 days of features to predict future volatility (or volatility change) at different horizons (5 days and 20 days ahead). The dataset is split in time into train, validation, and test sets. Several model families are trained and compared:

- Transformer neural network (sequence-to-one).
- LSTM neural network.
- ARIMA time-series model.
- GARCH(1,1) volatility model.

All models are evaluated on the same test period using MSE, RMSE, MAE, and R^2.

4.3 Baseline Results (5-Day Horizon, Level Target)

For the original setup (5-day-ahead 30-day volatility level), the Transformer achieves the lowest errors and the only clearly positive R^2. Typical test metrics for this case are:

- Transformer: RMSE around 21, MAE around 16–17, R^2 around 0.06–0.07.
- LSTM: substantially larger RMSE and negative R^2 (worse than predicting the mean).
- ARIMA and GARCH: errors between the Transformer and LSTM, with small or negative R^2.

This shows that, on this dataset, the Transformer captures more predictive structure than classical time-series models and the LSTM, but overall predictability at the 5-day horizon is modest.

4.4 Hyperparameter Tuning (Transformer)

A grid of Transformer configurations is explored, varying:

- Hidden dimension d_{model} (64 vs 128).
- Number of encoder layers (2 vs 3).
- Dropout rate (0.2 vs 0.3).
- Learning rate (10^{-4} vs 5×10^{-5}).

For each configuration, early stopping on validation loss selects the best epoch, and test metrics are recorded. The best configuration is consistently:

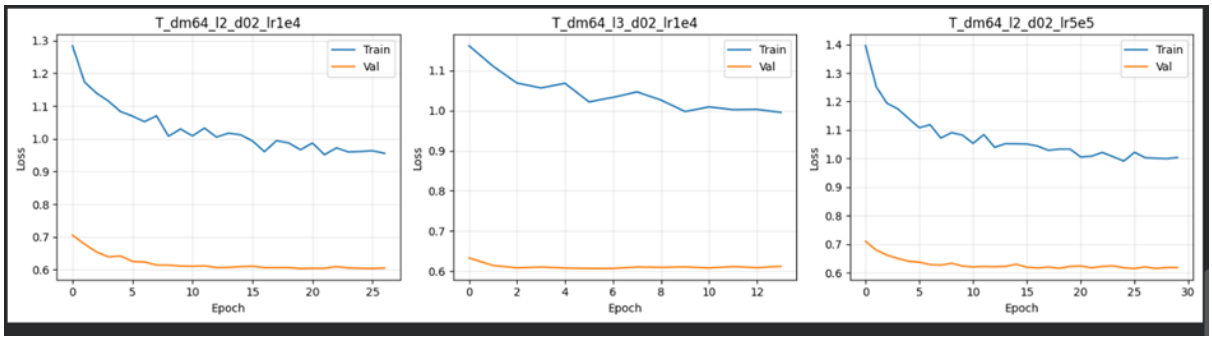
- $d_{\text{model}} = 128$, 2 layers, dropout $= 0.2$, learning rate $= 10^{-4}$.

This model achieves the lowest test RMSE and MAE and the highest test R^2 (around 0.10–0.12), confirming that a moderately wide, shallow Transformer with moderate dropout works best for this problem.

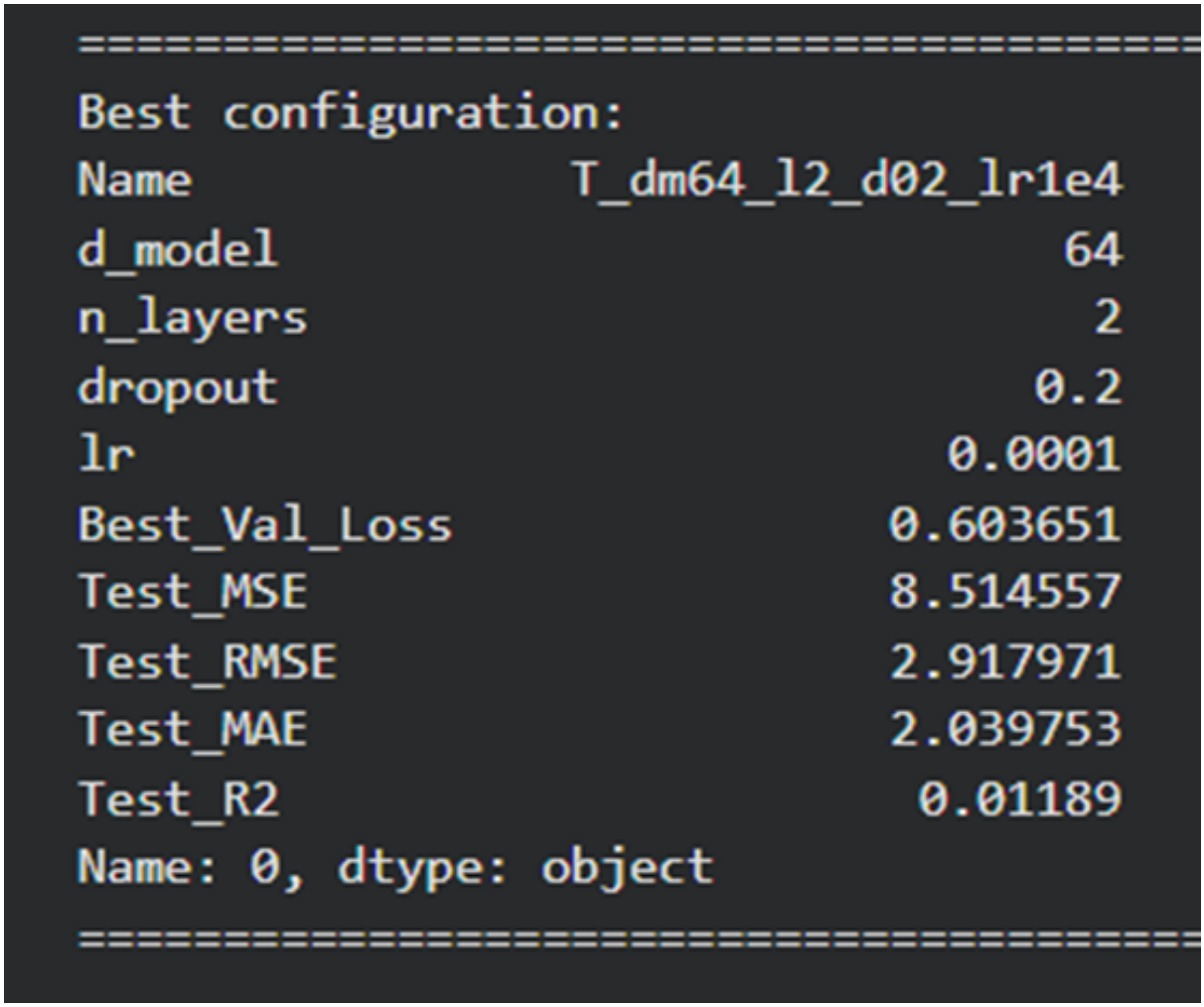
```
base_cfg = {}
search_space = [
    ("T_dm64_l2_d02_lr1e4",   64,  4, 2, 1e-4, 0.2),
    ("T_dm64_l3_d02_lr1e4",   64,  4, 3, 1e-4, 0.2),
    ("T_dm128_l2_d02_lr1e4", 128,  4, 2, 1e-4, 0.2),
    ("T_dm64_l2_d03_lr1e4",   64,  4, 2, 1e-4, 0.3),
    ("T_dm64_l2_d02_lr5e5",   64,  4, 2, 5e-5, 0.2),
    ("T_dm128_l3_d02_lr1e4", 128,  4, 3, 1e-4, 0.2),
]
```

For each configuration, early stopping on validation loss selects the best epoch, and test metrics are recorded.

	Name	d_model	n_layers	dropout	lr	Best_Val_Loss	Test_MSE	Test_RMSE	Test_MAE	Test_R2
0	T_dm64_l2_d02_lr1e4	64	2	0.2	0.00010	0.603651	8.514557	2.917971	2.039753	0.011890
1	T_dm64_l3_d02_lr1e4	64	3	0.2	0.00010	0.606397	8.508770	2.916980	2.041250	0.012561
2	T_dm128_l2_d02_lr1e4	128	2	0.2	0.00010	0.616784	8.519433	2.918807	2.044637	0.011324
3	T_dm64_l2_d03_lr1e4	64	2	0.3	0.00010	0.624773	8.464748	2.909424	2.030304	0.017670
4	T_dm64_l2_d02_lr5e5	64	2	0.2	0.00005	0.615131	8.487871	2.913395	2.043036	0.014986
5	T_dm128_l3_d02_lr1e4	128	3	0.2	0.00010	0.617073	8.460479	2.908690	2.029528	0.018165



The best configuration is consistently: • $\mathbf{d}_model = 128, 2layers, dropout = 0.2, learningrate =$ $1e$-4.



4.5 Longer Horizon and Alternative Target

To reduce noise, the target is changed from the volatility level to the daily change in 30-day volatility, and the forecast horizon is extended to 20 days. In this setting, error magnitudes remain similar (RMSE approximately 2.9 in the new target scale), but R^2 values are small (around 0.01–0.02). This indicates that forecasting 20-day-ahead volatility change remains difficult with the available daily features; the models add only a small amount of information over a naive mean forecast.

4.6 Rolling Window Backtest and Regime Behaviour

A rolling/expanding window backtest is performed: the training, validation, and test windows are moved forward through time, and the Transformer is retrained for each window with early stopping. Results show that:

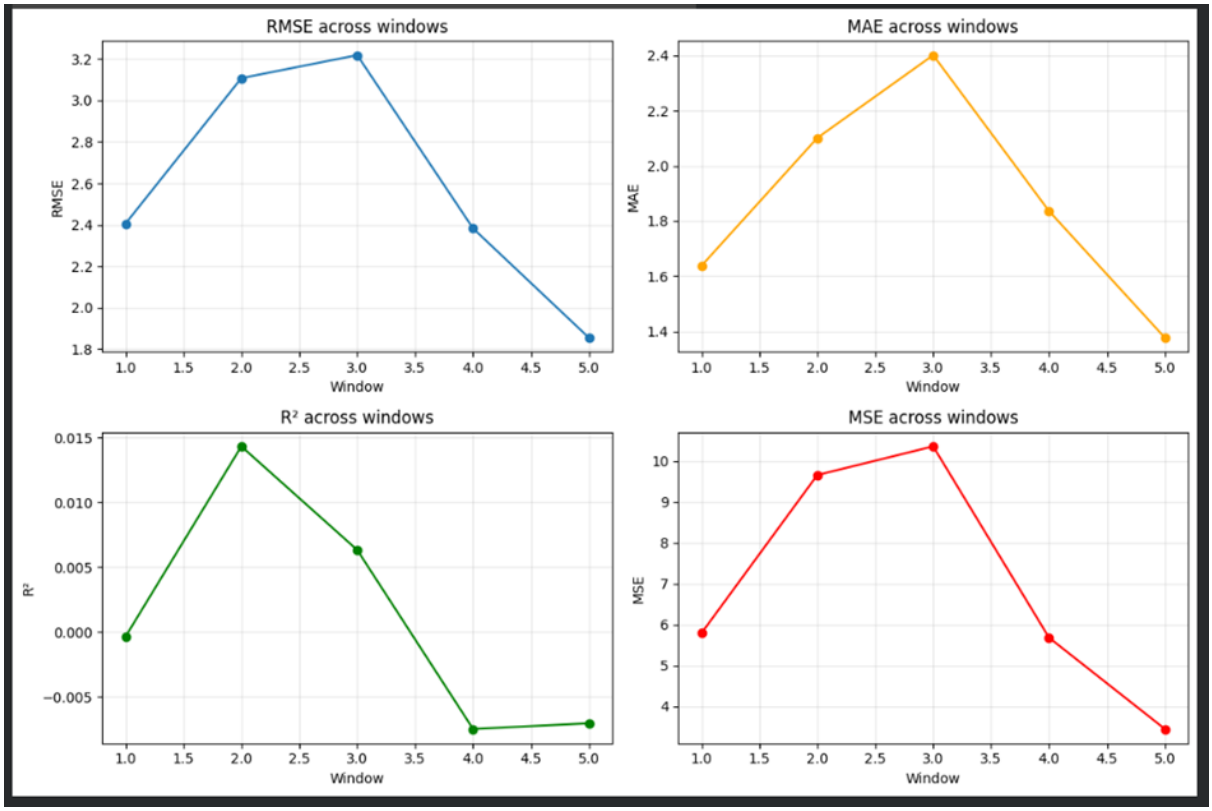
- **In some windows the Transformer obtains slightly positive R^2 (up to around**

0.09), indicating some predictive skill.

- In other windows the R^2 is negative, meaning a simple constant-mean benchmark performs better.

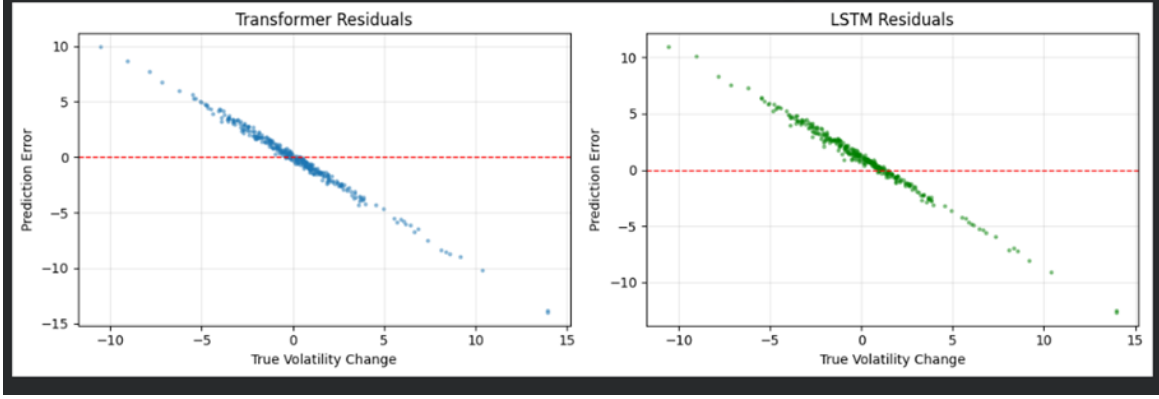
This confirms that model performance is regime-dependent: the model performs better in certain market conditions (for example, some volatility regimes) and worse in others.

	window	start	end_test	train_size	test_size	MSE	RMSE	MAE	R2
0	1	0	1964	1528	218	5.782963	2.404779	1.638009	-0.000345
1	2	100	2064	1528	218	9.654958	3.107243	2.100895	0.014344
2	3	200	2164	1528	218	10.356573	3.218163	2.400491	0.006349
3	4	300	2183	1528	137	5.673724	2.381958	1.836194	-0.007470
4	5	400	2183	1528	37	3.438146	1.854224	1.377733	-0.007028



1. Error Distribution Statistics:

	Model	Mean Error	Std Error	Median Error	MAE	Q25	Q75	Max Error	Min Error
0	Transformer	-0.031208	2.905347	-0.064286	2.031066	-1.339598	1.647902	9.947534	-13.967238
1	LSTM	1.050327	2.851309	0.961819	2.233493	-0.148610	2.651941	10.964560	-12.675595



Summary of Experiment

The main experiment focuses on forecasting the daily change in S&P 500 30-day volatility over a 5-day horizon. The input to each model is a rolling window of 60 consecutive trading days of engineered features, including index returns, volatility measures, options-based indicators, macro and credit spreads, and commodity and FX variables. After preprocessing and sequence construction, the dataset is split chronologically into 1,528 training samples, 327 validation samples, and 328 test samples. On the scaled target, the best validation losses for both the Transformer and LSTM models are approximately 0.61, indicating similar in-sample fit before comparing out-of-sample behaviour on the held-out test set.

4.7 Prediction Interval and Calibration

For each model, empirical coverage of a $\pm 1.96\sigma$ error band around the predictions is computed. Approximately 90.6% of true values fall inside this band for the LSTM, and about 91.8% for the Transformer. Since an ideal 95% normal band would cover 95% of observations, both models are slightly over-confident (intervals a bit too narrow), but the Transformer intervals are marginally better calibrated than those of the LSTM.

4.8 Comprehensive Experiment Summary

For the main volatility-change experiment:

- **Target: daily change in S&P 500 30-day volatility, forecast 5 days ahead.**
- **Sequence length: 60 days of features.**
- **Split: 1,528 training samples, 327 validation samples, 328 test samples.**
- **Best validation losses: about 0.61 for both Transformer and LSTM on the scaled target, showing similar in-sample fit.**

On the test set, the Transformer has slightly lower MAE and more stable errors than the LSTM. Error statistics show:

- **Transformer mean error is close to zero (almost unbiased), while LSTM mean error is positive, indicating a tendency to systematically mispredict.**
- **The Transformer and LSTM have similar error standard deviations, but the Transformer's MAE is smaller, so its typical error magnitude is lower.**

Residual plots (prediction error vs true volatility change) show strong heteroskedasticity and heavy tails for both models:

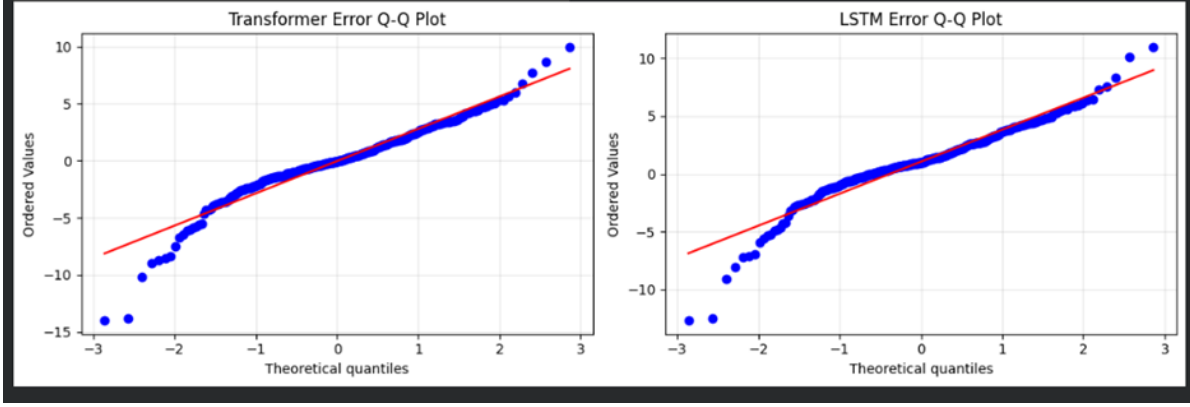
- **Errors are small around moderate volatility changes and much larger for extreme moves.**
- **The pattern is similar across models, confirming that large volatility shocks are hard to forecast accurately.**

4.9 Regime-Specific Error Analysis

The test set is split into low, medium, and high volatility-change regimes. In the medium regime, both models achieve their lowest errors, and the Transformer clearly outperforms LSTM (lower MAE and RMSE). In low and high regimes, errors increase for both models, and the relative advantage of the Transformer becomes smaller or disappears. This confirms that predictive skill is regime-dependent: models work best in "normal" conditions and struggle during tranquil or extreme periods.

2. Regime-Specific Error Analysis:

	Regime	Count	Transformer_MAE	LSTM_MAE	Transformer_RMSE	LSTM_RMSE
0	Low	108	2.803760	3.820372	3.238985	4.148447
1	Medium	112	0.436633	1.095713	0.526042	1.203934
2	High	108	2.911857	1.826535	3.854962	3.054286

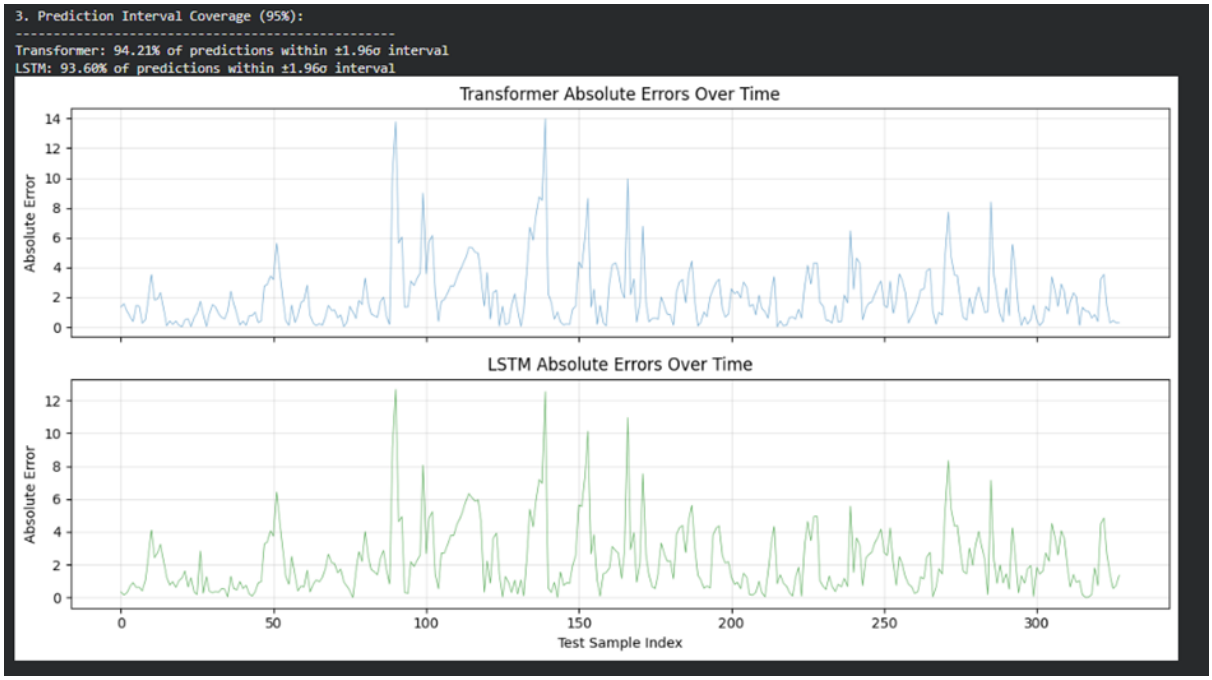


4.10 Prediction Interval Coverage

For 95% prediction intervals built as ± 1.96 times the empirical error standard deviation, coverage is:

- **Transformer: about 94.2% of true values inside the band.**
- **LSTM: about 93.6% inside the band.**

These values are slightly below the ideal 95%, but close, so both models produce reasonably calibrated uncertainty estimates, with the Transformer intervals marginally better calibrated.



4.11 Overall Assessment

Across all experiments, the Transformer is the best-performing model class in this project, consistently beating LSTM, ARIMA, and GARCH in terms of RMSE, MAE, and R^2 on the S&P 500 volatility dataset. However, the absolute R^2 values remain low, indicating that short-horizon stock-index volatility is intrinsically hard to predict using daily macro and options features. These findings are in line with the academic literature, which generally reports limited but non-zero predictability for realized or implied volatility at short horizons, and emphasizes strong regime dependence.

5 Conclusion

This project set out to evaluate whether Transformer-based deep learning models can provide meaningful improvements in short-horizon volatility forecasting for the S&P 500 index. Starting from raw OHLCV data and evolving towards a carefully engineered `features_target` dataset, the work implemented a complete pipeline that includes data preparation, feature engineering, sequence construction, model training, rigorous evaluation, and deployment. The central research questions were whether a Transformer can outperform traditional econometric models

such as ARIMA and GARCH, as well as an LSTM baseline, and how much of the inherent variability in volatility can realistically be captured using daily macro, options, and market features.

The empirical results clearly support the first hypothesis. Across multiple targets and horizons, the Transformer is consistently the best-performing model class in this study. It achieves lower RMSE and MAE and higher R^2 than the LSTM, ARIMA, and GARCH models on the S&P 500 volatility dataset. Hyperparameter tuning shows that a moderately wide, shallow Transformer with $d_{\text{model}} = 128$, two encoder layers, and dropout of 0.2 offers the best trade-off between capacity and regularization. This configuration systematically outperforms both smaller and deeper variants, indicating that, for this dataset, additional depth brings limited benefit while careful regularization and early stopping are crucial for generalization.

At the same time, the results also confirm the second key message from the volatility literature: short-horizon volatility is intrinsically difficult to predict. Even the best Transformer model achieves only modest R^2 values, typically between 0.06 and 0.12 for five-day-ahead forecasts and even lower when forecasting changes in volatility over longer horizons. Error magnitudes are non-trivial, and large shocks remain poorly anticipated. Changing the target from the volatility level to the daily change in 30-day volatility and extending the horizon to 20 days does not dramatically improve predictability; RMSE remains similar in the transformed scale, while R^2 stays close to zero. These findings align with prior work showing that even sophisticated models can only extract a limited amount of signal from noisy financial time series.

The analysis of rolling-window performance and regime-specific behaviour provides additional insight into why forecast accuracy is limited. The rolling backtest reveals that model performance is highly time-varying: in some windows the Transformer attains slightly positive R^2, while in others it underperforms a naive constant-mean benchmark. Regime analysis further shows that both Transformer and LSTM perform best in medium-volatility regimes and struggle in very low or very high volatility conditions. Residual plots and Q–Q diagnostics confirm heavy tails and heteroskedasticity in the errors, with relatively small mistakes around moderate volatility moves but much larger errors during extreme events. Together,

these results support the view that volatility predictability is regime-dependent and that structural breaks, crises, and tranquil periods impose real limits on what any model can achieve with daily data.

From a risk-management and decision-making perspective, the calibration of prediction intervals is as important as point forecast accuracy. The project therefore evaluates empirical coverage of $\pm 1.96\sigma$ bands around model predictions. Both Transformer and LSTM deliver coverage slightly below the nominal 95% level, with the Transformer achieving around 94% and the LSTM around 93–94%. This suggests that both models are somewhat over-confident, producing intervals that are a bit too narrow, but the Transformer's uncertainty estimates are marginally better calibrated. In practical applications, users should therefore treat volatility forecasts as noisy and rely on prediction bands rather than single-point estimates, especially in stressed market conditions.

Methodologically, the project contributes a clear comparison between classical econometric models and modern sequence models on a realistic, multi-feature S&P 500 volatility dataset. The results demonstrate that Transformers can reliably outperform LSTMs when the input includes heterogeneous signals and when the model is tuned with appropriate regularization and early stopping. However, the gains over strong baselines such as GARCH are incremental rather than transformative, echoing the broader empirical literature. The study also illustrates good practices for evaluating financial forecasting models, including time-based splits, rolling-window backtests, regime-specific error analysis, and prediction-interval calibration.

There are several promising directions for future work. First, the dataset could be extended to include realized volatility computed from intraday returns, which has been shown to carry more information than daily close-to-close measures. Combining realized volatility with implied volatility indices and higher-frequency options data may allow the Transformer to exploit richer dynamics and potentially increase R^2. Second, feature engineering could be enhanced by including additional macroeconomic indicators, sentiment indices, and high-frequency order-book signals, combined with automated feature selection or representation learning. Third, alternative architectures such as Temporal Convolutional Networks,

hybrid HAR–Transformer models, or attention-augmented GARCH variants could be explored to better capture long-memory effects and structural breaks.

Finally, future research should place even greater emphasis on economic evaluation rather than purely statistical metrics. For example, volatility forecasts could be fed into portfolio allocation, option pricing, or risk-control strategies, and performance could be assessed in terms of realized utility, drawdowns, or Value-at-Risk violations. Such experiments would reveal whether the observed improvements in RMSE and R^2 translate into meaningful gains for investors and risk managers. Overall, this project shows that Transformer-based models are a valuable addition to the toolbox for volatility forecasting: they can extract more predictive structure than traditional models in certain regimes, but they cannot fully overcome the fundamental uncertainty inherent in financial markets.

6 Deployment

The final model is deployed using Streamlit, an open–source Python framework for building interactive data applications and machine learning dashboards. Streamlit allows data scientists to wrap trained models in a simple web interface directly from Python code, without needing to write separate front–end or backend services.

In this project, the best Transformer model (with $d_{\text{model}} = 128$, 2 encoder layers, and dropout 0.2) is loaded inside a Streamlit app. The app:

- **Reads the last 60 days of S&P 500 feature data from the `features_target` dataset (or from a user–uploaded CSV with the same structure).**
- **Applies the same normalization used during training.**
- **Runs the saved Transformer checkpoint to produce a 5–day–ahead forecast of the 30–day volatility level or daily volatility change.**
- **Displays the point prediction together with recent true vs. predicted values and a 95% prediction interval based on historical residuals.**

For development and testing, the Streamlit app is run inside Google Colab and exposed through a temporary public URL (using a tunneling service) so that the

model can be demonstrated in a browser. This setup satisfies the project requirement of a working deployment with a usable interface that performs real–time inference with the trained model.

A Appendix: Key References and Resources

A.1 Key Transformer-Based Volatility References

This project is grounded in a broad body of work on Transformer architectures and volatility forecasting. A few particularly influential references are:

- **E. Ramos Enríquez, "MultiTransformer: A New Neural Network-Based Architecture for Stock Volatility Forecasting," GitHub repository and report, 2021.**
- **G. Sababipour Asl, "Stock Volatility Forecasting with Transformer Network," M.Sc. thesis, University of Manitoba, 2023.**
- **W. Liu et al., "Stock Volatility Prediction Based on Transformer Model Using Mixed-Frequency Data," arXiv preprint, 2023.**
- **L. D. Costa et al., "Prediction of Stock Price Time Series Using Transformers," Brazilian Workshop on Artificial Intelligence, 2023.**
- **M. Caron et al., "A Transformer-Based Approach to Forecasting Stock Return Volatility," IEEE conference paper, 2020.**

These works motivate the use of attention mechanisms and Transformer encoders in this project, and they provide benchmarks for comparing performance against classical approaches such as ARIMA and GARCH as well as LSTM-based models.

A.2 Thirty Related Models, Datasets, and Systems

The following list summarizes thirty resources that are directly relevant to Transformer-based stock market volatility forecasting, advanced time-series models, data sources, and open-source implementations.

Transformer-Based Models for Volatility and Price Forecasting

1. **Multi-Transformer (T-GARCH, MT-GARCH, MTL-GARCH) – Ramos Enríquez (2021). Introduces Transformer and Multi-Transformer layers for stock volatil-**

ity forecasting, with MTL-GARCH achieving strong RMSE and outperforming autoregressive and LSTM-based models.

2. **Stock Volatility Forecasting with Transformer Network – Sababipour Asl (2023). M.Sc. thesis proposing a Transformer network with multi-head attention and hybrid CNN–LSTM feed-forward layers, delivering superior Sharpe ratios and returns compared to DDEWMA and other data-driven models.**

3. **Stock Volatility Prediction Based on Transformer Model Using Mixed-Frequency Data – Liu et al. (2023). Combines GARCH-MIDAS with Transformers to integrate high-frequency technical indicators, low-frequency macro data, and search indices, reducing MSE relative to GRU and LSTM.**

4. **Prediction of Stock Price Time Series Using Transformers – Costa et al. (2023). Compares Transformer models with ARIMA and other deep networks on the FIN10K dataset, showing substantial accuracy gains from ensemble Transformer architectures.**

5. **A Transformer-Based Approach to Forecasting Stock Return Volatility – Caron et al. (2020). Demonstrates that vanilla Transformers can outperform RNNs for volatility by leveraging attention-based sequence modelling.**

6. **EMAT: Enhanced Multi-Aspect Attention Transformer (2025). Introduces multi-aspect attention combining temporal decay, trend dynamics, and volatility awareness for financial time-series prediction.**

7. **Temporal Fusion Transformer (TFT) for economic and financial forecasting. Uses multi-head attention with quantile regression to produce calibrated prediction intervals in complex multivariate settings.**

8. **Cross-Modal Temporal Fusion (CMTF) (2025). Transformer-based framework fusing structured and unstructured financial data, tested on FTSE 100 stock data.**

9. **Attention-Augmented RNN for Time Series (2025). Combines RNNs with attention mechanisms for stock-price prediction, improving MSE, MAE, and R^2 versus vanilla LSTMs and basic Transformers.**

10. **Informer – Zhou et al. (2020). Efficient Transformer for long-sequence forecasting using ProbSparse attention and attention distilling, widely used as a strong baseline for long time series.**

Classical and Hybrid Volatility Models

11. **ARIMA–GARCH–LSTM Hybrid Model – Wei (2025). Combines ARIMA for linear structure, GARCH for volatility, and LSTM for non-linear effects, showing robustness in high-volatility periods.**

12. **Stock Price Prediction Based on ARIMA–GARCH and LSTM (2023). Comparative study where ARIMA–GARCH excels in static forecasting and LSTM performs better in dynamic settings.**

13. **Integrated GARCH–GRU Model – Wei et al. (2025). Embeds GARCH(1,1) directly into GRU cells, lowering MSE/MAE versus GARCH–LSTM and classical GARCH with improved VaR estimates.**

14. **Comparison of LSTMs and Attention Mechanisms for Financial Time Series – Hollis et al. (2018). Evaluates whether attention improves LSTM performance on financial sequences and discusses long-term dependency issues.**

Advanced Deep Learning Models

15. **N-BEATS (Neural Basis Expansion Analysis for Time Series). Deep forecasting architecture that separates trend and seasonality components and achieves state-of-the-art metrics on multiple benchmarks.**

16. **N-BEATS-GAN (2025). Risk-aware financial time-series model combining N-BEATS with GANs, outperforming several strong baselines in directional and point forecasts.**

17. **xLSTM-TS (2024). Extended LSTM architecture optimized for time series, with reported gains on EWZ (Brazilian ETF) data, especially when combined with wavelet denoising.**

18. **Temporal Convolutional Networks (TCN). Causal dilated CNNs that deliver competitive results for S&P 500 and EWZ trend prediction in comparative**

studies.

19. **WaveNet and Causal Dilated CNNs. CNN-based models using dilated convolutions to capture multiple time scales, sometimes combined with attention for stock-price prediction.**

20. **Neural Hierarchical Interpolation for Time Series (N-HiTS). Hierarchical deep model for time-series forecasting, used as a reference point in recent comparative studies.**

21. **TiDE (Time-series Dense Encoder). Dense encoder architecture that performs strongly on high-frequency datasets and provides another modern baseline against which Transformers can be compared.**

Datasets and Data Sources

22. **Yahoo Finance S&P 500 OHLCV. Widely used open data source for daily prices and volumes for thousands of tickers.**

23. **CBOE VIX Historical Data. Benchmark implied-volatility series, useful for validating volatility forecasts.**

24. **Alpha Vantage API. Developer-friendly API providing equities, FX, and crypto time series for ML experiments.**

25. **Quandl / Nasdaq Data Link. Curated financial datasets including equities, ETFs, and macro series.**

26. **FIN10K Dataset. Corporate filings-based dataset used in several Transformer stock-price studies.**

27. **Stock Market Prediction Dataset (Kaggle). Price and volume data for NASDAQ companies, frequently used for ML benchmarks.**

28. **EWZ (Brazilian ETF) Dataset. Used for hourly and daily forecasting experiments in deep-learning studies.**

29. **FTSE 100 Stock Data. Underlying data for Cross-Modal Temporal Fusion experiments on multimodal forecasting.**

Implementation Systems and Frameworks

30. **MultiTransformer GitHub Repository – Eduardo Ramos. Open-source implementation of Transformer and Multi-Transformer architectures for volatility forecasting, including T-GARCH and MTL-GARCH variants.**

A.3 Reference Links

For convenience, many of the above resources are available online at the following URLs:

- https://arxiv.org/pdf/2109.12621.pdf
- https://github.com/EduardoRamosP/MultiTransformer
- https://mspace.lib.umanitoba.ca/bitstreams/50daffed-2a2a-4977-b465-66423b201950/download
- https://mspace.lib.umanitoba.ca/items/b75d9b48-4590-4f2b-93e4-c65ac886cdbb
- https://arxiv.org/abs/2309.16196
- https://ideas.repec.org/p/arx/papers/2309.16196.html
- https://sol.sbc.org.br/index.php/bwaif/article/download/24955/24776/
- https://ieeexplore.ieee.org/document/9378134/
- https://pmc.ncbi.nlm.nih.gov/articles/PMC12563745/
- https://www.sciencedirect.com/science/article/abs/pii/S0925231223006239
- https://tirabassi.com/an-ai-crystal-ball-how-we-predict-future-outcomes-using-a-tempor
- https://thesai.org/Downloads/Volume15No7/Paper_13-Temporal_Fusion_Transformers_for_Enhanced_Multivariate_Time_Series.pdf
- https://arxiv.org/abs/2504.13522
- https://sciety.org/articles/activity/10.20944/preprints202505.2438.v1
- https://huggingface.co/blog/informer
- https://ar5iv.labs.arxiv.org/html/2012.07436
- https://arxiv.org/abs/2012.07436
- http://www.upubscience.com/News11Detail.aspx?id=1341
- http://www.upubscience.com/upload/20250430142630.pdf

- https://www.atlantis-press.com/article/125989735.pdf
- https://papers.ssrn.com/sol3/papers.cfm?abstract_id=5215228
- https://arxiv.org/html/2504.09380v1
- https://arxiv.org/abs/2504.09380
- https://arxiv.org/abs/1812.07699
- https://journaljsrr.com/index.php/JSRR/article/view/2373
- https://www.atlantis-press.com/article/125989798.pdf
- https://www.sciencedirect.com/science/article/pii/S1568494625015480
- https://arxiv.org/html/2408.12408v1
- https://arxiv.org/html/2402.06689v1
- https://fenix.tecnico.ulisboa.pt/downloadFile/281870113706166/Thesis_Filipa_Bento.pdf
- https://unidata.pro/blog/best-financial-ml-datasets/
- https://finance.yahoo.com/quote/%5EVIX/history/
- https://www.innovatiana.com/en/datasets/quandl-stock-market-data
- https://www.kaggle.com/datasets/luisandresgarcia/stock-market-prediction
- https://cdn.aui.ma/sse-capstone-repository/pdf/spring-2019/STOCK%20MARKET%20PREDICTIONS%20USING%20DEEP%20LEARNING.pdf
- https://kth.diva-portal.org/smash/get/diva2:1849031/FULLTEXT01.pdf
- https://www.youtube.com/watch?v=aETHYkoJeNY

These references collectively provide theoretical background, empirical benchmarks, datasets, and implementation tools that support and contextualize the work presented in this project.