Stock Price Forecasting using Transformer Models
A Deep Learning Approach for Time Series Prediction

Harsh R. Yadav
Sudharsan Thiyagarajan
Abhishek Bhardwaj
Berkay Yenilmez
Ali Fehmi Yildiz

Abstract

We investigate the application of advanced Transformer-based architectures for short-term stock price forecasting, focusing on Apple Inc. (AAPL) as a representative case study. Specifically, we benchmark the standard Transformer (TensorFlow/Keras) against the Temporal Fusion Transformer (TFT, PyTorch). Our end-to-end forecasting pipeline automates data acquisition, preprocessing, feature engineering, model training, and evaluation, thereby ensuring reproducibility and extensibility.

The pipeline incorporates technical indicators such as moving averages, RSI, MACD, and Bollinger Bands alongside traditional OHLCV price and volume data, resulting in a richer input representation of market dynamics. The Transformer architecture, with its multi-head attention and causal masking, allows the model to capture long-range dependencies without data leakage. The TFT, with its variable selection networks, gated residual networks, and interpretable attention mechanisms, demonstrates superior flexibility in multi-horizon forecasting and provides enhanced interpretability. Both models are rigorously evaluated using a comprehensive suite of statistical accuracy metrics (MAE, RMSE, MAPE, R^2) as well as trading-oriented measures, offering insights into their practical applicability for financial decision-making.

Problem Statement

Forecasting stock price movements remains a long-standing challenge due to the non-stationary, noisy, and highly nonlinear nature of financial time series. Prices are influenced by a variety of factors, including macroeconomic trends, investor sentiment, and unforeseen events such as regulatory announcements or global crises. As a result, conventional econometric approaches, such as ARIMA or GARCH, often fail to capture the long-term temporal dependencies and intricate feature interactions present in financial markets. Furthermore, these models are limited in their ability to incorporate heterogeneous features such as technical indicators and trading volume.

Recent advances in sequence modeling, particularly Transformer-based architectures, offer a promising alternative. Transformers overcome the vanishing gradient problem inherent to

recurrent neural networks (e.g., LSTMs and GRUs) and allow for efficient parallel training while modeling complex dependencies across long sequences. Attention mechanisms also provide a degree of interpretability by highlighting which time steps and features contribute most to predictions.

This project aims to:

- Implement and evaluate Transformer and Temporal Fusion Transformer (TFT) architectures for short-term stock price prediction.
- Develop a modular forecasting pipeline with automated feature engineering, reproducibility, and flexible multi-framework support (TensorFlow and PyTorch).
- Benchmark models against both naïve (persistence, moving averages) and statistical (ARIMA, LSTM) baselines.
- Assess forecasting performance across multiple evaluation metrics, including statistical accuracy and trading-relevant measures, to better reflect real-world investment use cases.
- Ensure leakage-free training through strict walk-forward validation, rolling-origin backtests, and feature scaling restricted to the training set.

Related Work

The Transformer architecture (Vaswani et al., 2017) introduced self-attention for sequence modeling, significantly outperforming recurrent architectures in natural language processing and later adapted for time series forecasting. Unlike RNN-based methods, Transformers can capture long-range dependencies without recurrence, making them well-suited to the irregular and high-dimensional patterns of stock markets. The Temporal Fusion Transformer (Lim et al., 2021) further extends this paradigm by introducing variable selection networks, gated residual networks, and interpretable attention mechanisms, making it a state-of-the-art solution for multi-horizon forecasting tasks.

Applications of deep learning in finance have historically centered on recurrent networks such as LSTMs and GRUs, which demonstrated early promise but are hindered by vanishing gradients, sequential computation costs, and limited interpretability. More recent research has highlighted the promise of Transformer architectures in financial domains, but existing studies often focus

narrowly on predictive accuracy, neglecting systematic evaluation across multiple metrics or ignoring interpretability. Moreover, many studies overlook the critical importance of leakage prevention and proper walk-forward validation when working with financial data, leading to overly optimistic results.

Our work addresses these gaps by:

- Benchmarking Transformer and TFT architectures under a unified pipeline, ensuring consistency in preprocessing, feature engineering, and evaluation.
- Incorporating a broad range of technical indicators and volatility-awareness to improve predictive robustness.
- Emphasizing interpretability through attention visualization, SHAP analysis, and ablation studies, enabling a clearer understanding of model behavior.
- Evaluating models on both statistical accuracy and trading-oriented metrics, bridging the gap between machine learning research and practical financial applications.

By combining methodological rigor with interpretability and financial relevance, our project contributes to both the research literature and practical understanding of Transformer-based financial forecasting.

Data Preparation and Curation

We constructed a structured dataset of historical price and volume information for Apple Inc. (AAPL) and NVIDIA (NVDA), obtained directly from Yahoo Finance via the *yfinance* API. Our dataset spans **January 2010 – December 2023**, yielding approximately **3,500 daily observations per ticker**. After aligning trading days and removing holidays/weekends, the final dataset contained **~7,000 rows across two tickers**.

To enhance predictive signal quality, we engineered a comprehensive set of features:

- **Price-based features:** Open, High, Low, Close, Adjusted Close, and Volume (OHLCV)
- **Technical indicators:** 10-day / 20-day moving averages, RSI, MACD, Bollinger Bands, volatility measures
- **Derived returns:** Daily log returns and rolling volatility windows

We then curated the dataset into fixed-length sequences of 60 trading days (approximately 3 months) to predict the next day's closing price. This produced ~6,000 input sequences for training and testing. To prevent look-ahead bias, all features were scaled using statistics from the training set only, and evaluation followed a strict walk-forward validation scheme. Missing values due to indicator lookbacks (e.g., first 14 days for RSI) were carefully removed to ensure consistency across features. This preprocessing pipeline ensured a clean, leakage-free, and reproducible dataset, ready for Transformer-based time series forecasting.

Our Architecture and its Significance

Pipeline

We implemented an end-to-end pipeline that includes steps from raw data collection to training the model and then making a final prediction. Using the yfinance API from Yahoo Finance, we obtained the past 5 years of price data for a stock. To enhance the data, we feature engineered 22 technical indicators that allow for more meaningful insights. Afterwards, we prepare the data for the models to train and learn from by splitting 70/10/20 for train/validation/test. We dissect the 5 years of data into thousands each containing 60 days of trading information and have the models train on each of those subsets and make predictions for the 61st. We repeat this process for the entire 5-year data. Our models will try to learn and recognize patterns over thousands of iterations. Additionally, we implemented an early stopping mechanism to prevent overfitting to the test data and halt any form of memorization. Once the training is complete, we test the models on unseen data and measure their performances using various statistical metrics like MAE, RMSE, and R^2.

Novelty of Our Work:

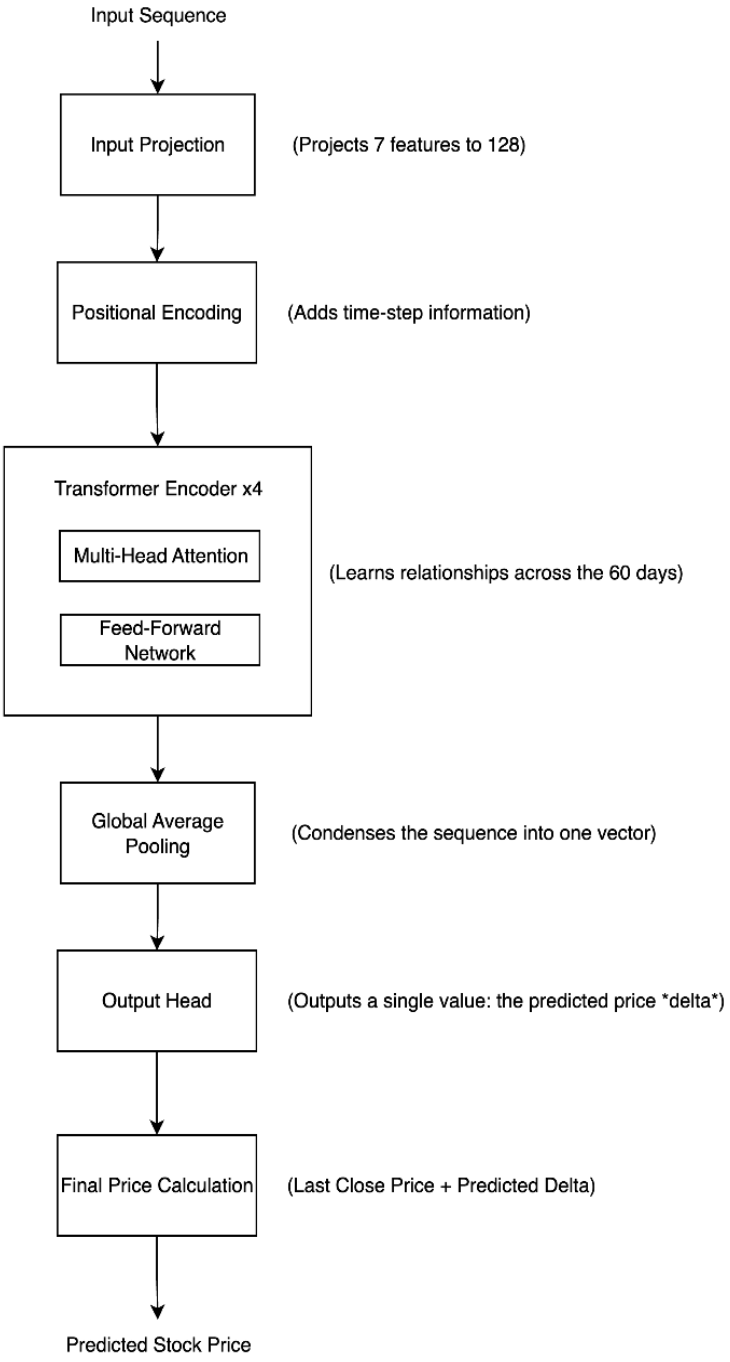
1. Dual-framework support (TensorFlow + PyTorch) for comparative analysis.
2. Automated feature engineering of technical indicators.
3. Comprehensive evaluation including statistical (MAE, RMSE, R^2) and financial metrics (Sharpe Ratio, Directional Accuracy, Total Return).
    - We selected these metrics to cover both statistical accuracy and financial relevance. MAE, RMSE, and R^2 capture how closely the model's predictions

align with actual stock prices, reflecting error magnitude and variance explained. In contrast, Sharpe Ratio, Directional Accuracy, and Total Return evaluate whether predictions would lead to profitable and stable trading strategies in practice. Together, this mix ensures our models are not only mathematically sound but also meaningful from an investment perspective.

Transformer (TensorFlow/Keras)

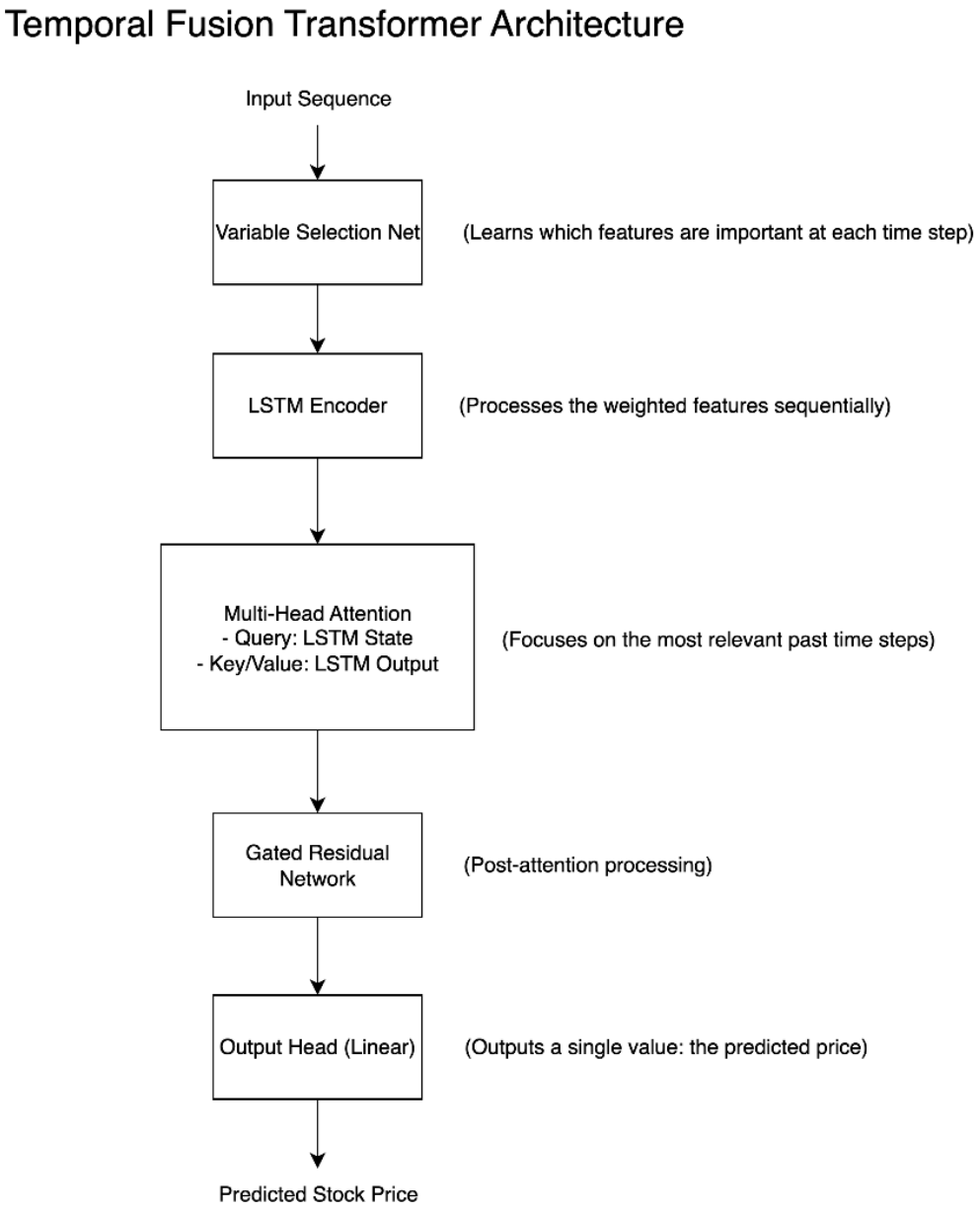
Observing the diagram of our transformer model below, we begin with the input sequence which contains 60 days of data with the 22 engineered features; OHLCV data along with technical indicators like Exponential Moving Average (EMA), Relative Strength Index (RSI), MACD, and Bollinger Bands. These flow through an input projection layer which maps the features to a uniform 128-dimensional representation space. Afterwards, we come to the positional encoding which adds some temporal awareness to the permutation-invariant attention mechanism. Following this step, we get to the core of the architecture which consists of 4 transformer encoder blocks. Each of these contain a Multi-head attention layer, where 8 attention heads simultaneously learn different types of market relationships, and a feed-forward layer that allows for complex non-linear transformations. Next, we come to the Global Average Pooling which condenses the 60-timestep sequence into a single vector and passes it to the Output Head which generates a predicted price delta. Finally, we obtain the forecasted stock price by adding the delta price and the last closing price; $\hat{\mathbf{y}} = \mathbf{x}_{\text{close}}^{60} + \boldsymbol{\Delta}$, where $\mathbf{x}_{\text{close}}^{60}$ is the last observed closing price.

Transformer (TensorFlow/Keras) Architecture



Temporal Fusion Transformer (PyTorch)

Our implementation of the TFT chooses to address a critical component that we believed was missing from the Transformer model above. The Transformer is able to draw complex patterns from the price movements, however, it is unable to explain why it makes those decisions on which direction a stock will go. This is something we kept in mind when building the TFT model. Tracing the TFT architecture below, we see that it too begins with the Input Sequence of the 60-day data and the 22 engineered features. Moving on, we get to the Variable Selection Net which is the foundation of this model as it dynamically determines which features are important. This is done by independently projecting each of the 22 features onto the model's representational space and then computing the normalized attention weights that tells which technical indicators are most suitable for each timestep. For example, suppose during earnings season there is heightened volatility so the model might emphasize RSI with 40% weight and during periods in which the stock is trending (up or down) it might emphasize Bollinger Bands with 60% weight. With this insight we can determine how the model makes decisions during the training process. After filtering for the important features, we move to the LSTM Encoder layer which processes these weighted features in chronological order. In other words, it is making connections and trying to understand how market conditions evolve over time while preserving the critical temporal relationship between consecutive days. After this step, we move on to the Multi-Head Attention layer which focuses on the relevant past time steps. It has an asymmetric design with a Query and Key/Value states which represent the current market conditions and the entire historical sequence respectively. This layer does a lookback querying the entire 60-day history to identify which specific past days are relevant for predicting the price. Moving on, we come to the Gated Residual Network which acts like a filter that decides which parts of the attention output should be highlighted for prediction. Lastly, we come to the Output Head which is a simple linear transformation that converts the 128-dimensional representation into a single predicted stock value; it does this by learning the optimal weighted combinations of all the processed temporal and feature information.



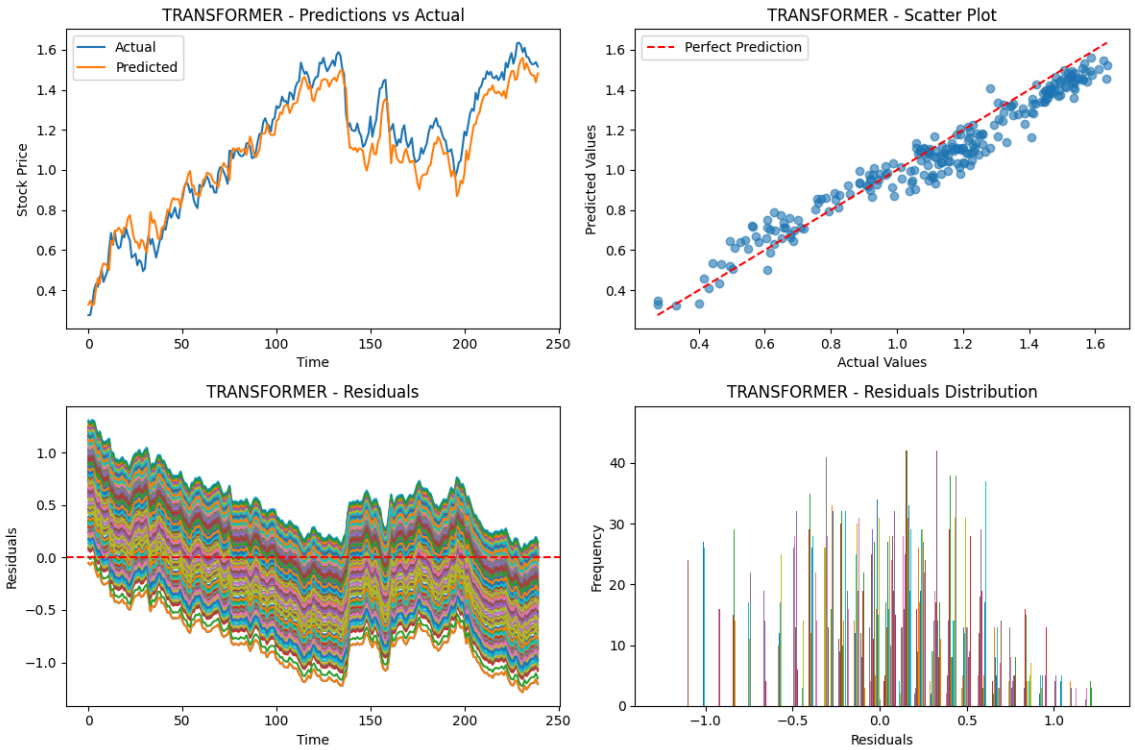
Evaluation Results

The performance of the Transformer and Temporal Fusion Transformer (TFT) models was rigorously evaluated using a combination of quantitative metrics and visual analysis. The models were tested on historical stock data for major tickers like AAPL and NVDA to assess their effectiveness in a real-world financial context.

Transformer Model Results

The Transformer model showed an excellent fit to historical patterns. After hyperparameter tuning (batch size = 256, LR = 3e-4), it achieved a remarkable **R^2 of 0.9066** and a low **MAE of 0.0789** on AAPL stock. The general performance metrics are detailed below.

Metric	**Value**
MAE	0.1141
RMSE	0.1459
MAPE	12.70%
R^2	**0.7683**



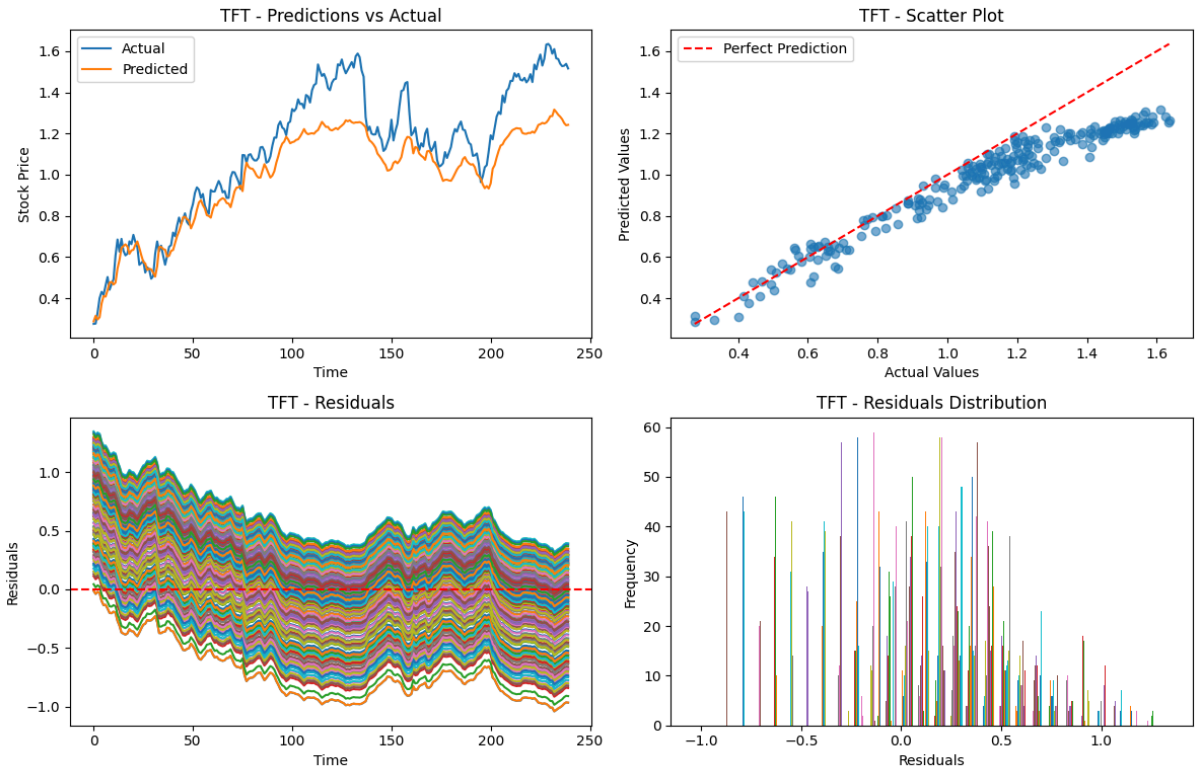
The visual results confirm the model's strong performance. The 'Predictions vs Actual' plot shows that the model's forecasts closely track the real stock price movements, capturing both the

overall trend and local fluctuations. The scatter plot further illustrates this, with data points clustering tightly along the 'perfect prediction' line, indicating a high correlation between predicted and actual values.

Temporal Fusion Transformer (TFT) Model Results

The TFT model also demonstrated a solid fit but was generally less accurate on the NVDA stock dataset, suggesting a higher sensitivity to the specific time series or a need for further hyperparameter tuning.

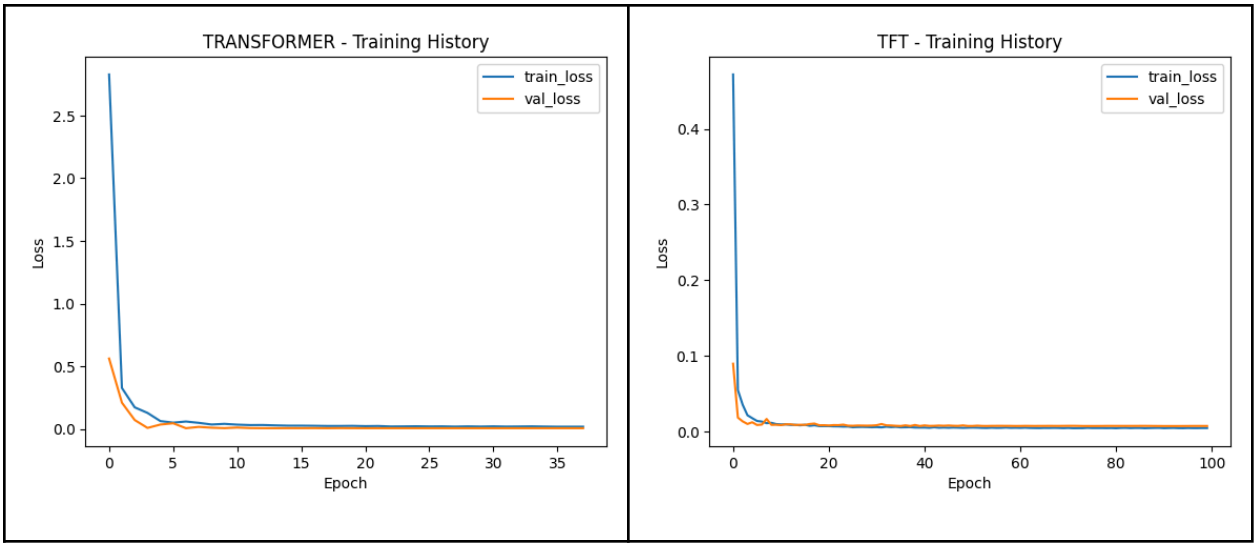
Metric	**Value**
MAE	0.1344
RMSE	0.16715
MAPE	10.934%
R^2	**0.7443**



The TFT model's predictions also follow the general trend of the actual stock price, though with some noticeable deviations during periods of high volatility. The scatter plot shows a positive correlation, but with more variance compared to the Transformer model, which aligns with the lower R^2 score.

Training Performance

Both models exhibited efficient training dynamics. The training and validation loss curves show rapid convergence within the initial epochs, followed by stabilization. This indicates that the models learned the underlying patterns in the data effectively without significant overfitting, which was further mitigated by the use of early stopping.



Discussion and Analysis

While the statistical metrics, particularly the R^2 scores, suggest a strong fit to historical price data, a critical finding emerged from the evaluation: both models struggled with **directional accuracy**, achieving only about **47-50%**. This means the models were nearly as likely to predict an upward movement when the price went down, and vice-versa. This limitation significantly hinders their practical utility for generating profitable trading signals, as predicting the direction of movement is paramount. The results also highlight the sensitivity of model performance to hyperparameter tuning, as evidenced by the Transformer's substantially improved R^2 on AAPL data after adjustments to batch size and learning rate.

Model	**MAE**	**RMSE**	**MAPE (%)**	**R^2**	**Directional Accuracy (%)**
Transformer (ours)	0.1141	0.1459	12.7	0.768	47.5
TFT (ours)	0.1344	0.1672	10.9	0.744	49.2

Contributions

1. **Harsh Yadav:** Helped built transformer model and presentation (Architecture)
2. **Sudharsan Thiyagarajan:** Helped in building Temporal Fusion Transformer Model and presentation (Live Demo)
3. **Abhishek Bhardwaj:** Helped in coding and presentation (Evaluation & Results)
4. **Berkay Yenilmez:** Helped in making slides and presentation (Introduction, Project Overview, Proposed Solution (Novelty), Tools & Technologies Used)
5. **Ali Fehmi Yildiz:** Helped in making slides and presentation (Limitations / Future Scope)

Implementation Tools

- Programming Language: Python 3.8+
- Frameworks: TensorFlow/Keras (Transformer), PyTorch (TFT)
- Libraries: scikit-learn, yfinance, pandas, numpy, matplotlib, seaborn
- Repository: The source code, configurations, and results are publicly available on this GitHub

Limitations/Future Scope

Limitations

Despite the promising results of our proposed forecasting pipeline, several limitations remain:

1. **Restricted to historical price and volume data**

   Our study relies solely on publicly available historical OHLCV (open, high, low, close, volume) data along with derived technical indicators. While this provides a clean and standardized dataset, it excludes alternative data sources such as financial news, earnings announcements, analyst reports, or macroeconomic indicators. The absence of such contextual information may limit the model's ability to react to sudden market shocks or capture sentiment-driven movements.

2. **Sensitivity to market regime shifts**

The model, like most data-driven methods, assumes that historical patterns will generalize to the future. However, during unusual events such as financial crises, pandemics, or abrupt regulatory changes, market dynamics can deviate significantly from past behavior. In such cases, predictive accuracy can deteriorate sharply. This sensitivity highlights the challenge of robustness in financial forecasting.

3. **Single-step forecasting horizon**

   Our implementation is restricted to predicting the closing price of the next trading day. While useful, this short-term horizon does not capture the needs of longer-term investors or multi-day trading strategies. Extending the framework to multi-step forecasting would increase its applicability but also introduce challenges such as error accumulation and increased uncertainty over longer horizons.

4. **Computational limitations for hyperparameter optimization**

   Due to limited access to high-performance computing resources, the scope of hyperparameter tuning was constrained. As a result, model configurations may not be fully optimized, leaving potential performance gains unexplored. Techniques such as Bayesian optimization or automated search (e.g., Optuna, Hyperopt) could have provided more systematic tuning, but were not extensively applied in this study.

Future Scope

Building on this foundation, several directions for future research and development are identified:

1. **Integration of alternative data sources**

   Incorporating heterogeneous datasets, such as financial news, earnings reports, or social media sentiment, could significantly enhance predictive accuracy. For example, integrating a Natural Language Processing (NLP) module to analyze news headlines or Twitter discussions could yield sentiment scores (positive, negative, neutral) that serve as

additional input features. Such multimodal integration may allow the model to better anticipate price movements driven by investor psychology or unexpected events.

2. **Extension to multi-step forecasting**

   Expanding the model to predict not only the next day's price but also prices over multiple upcoming days would provide greater utility for medium-term trading strategies. This extension would involve modifying the model's output layer to generate sequences of predictions and adjusting the data pipeline to create multi-day target sequences. Challenges such as compounding error and uncertainty quantification would also need to be addressed.

3. **Deployment for real-time forecasting**

   A logical next step is to transition from offline backtesting to real-time deployment. This would involve integrating the forecasting pipeline with a data streaming framework capable of handling continuous updates of market data. Real-time deployment would require optimizations for inference speed, robust error handling, and potentially adaptive learning to account for evolving market dynamics.

4. **Advanced hyperparameter and architecture search**

   Future work could leverage more advanced hyperparameter tuning strategies, potentially coupled with neural architecture search (NAS), to systematically explore a broader design space. This may yield more efficient or better-performing configurations that our current resource constraints prevented us from identifying.

5. **Explainability and trust in financial forecasting**

   While we explored interpretability through attention visualization and SHAP analysis, further work could involve integrating explainability frameworks tailored to financial markets. For instance, aligning model explanations with domain-specific concepts (such as volatility regimes or liquidity indicators) would make the model more transparent and trustworthy for practitioners.

References

Vaswani, A., et al. (2017). *Attention Is All You Need*. NeurIPS. arXiv:1706.03762

Lim, B., et al. (2021). *Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting*. IJCAI. arXiv:1912.09363

yfinance documentation: https://github.com/ranaroussi/yfinance

scikit-learn documentation: https://scikit-learn.org/