Adaptive Stock Price Forecasting Using Temporal and Market-Guided Architectures

CS 5787 Deep Learning, Fall 2024
Final Project Report
Name: Rahul Vedula, Arnav Kolli, Tarun Venkatasamy
NetID: rv299, ak2677, tv89

Abstract

Stock price forecasting is a critical yet challenging task due to the volatile nature of financial markets. Traditional models often struggle to capture both long-term temporal dependencies and adapt to real-time market signals, leading to suboptimal performance. This paper introduces a novel approach that combines market-aware techniques with advanced temporal representations to enhance stock price prediction accuracy. Our model integrates insights from two key research papers: one focusing on multi-patch prediction for time-series representation and the other on market-guided stock transformers. By employing a parallel fusion architecture, the model processes temporal and market features independently before combining them in a prediction layer, ensuring a simultaneous focus on localized patterns and broader market trends. The temporal pathway uses transformer encoders to process segmented time-series data, while the market-gating pathway dynamically adjusts feature importance based on market conditions. Experiments conducted using historical stock data from major technology companies demonstrate that our parallel fusion model outperforms traditional hierarchical models in prediction accuracy and adaptability. The results highlight the model's ability to extract nuanced feature representations and respond dynamically to market fluctuations. This work underscores the potential of advanced fusion architectures in enhancing financial modeling and decision-making processes, offering a promising direction for future research in stock price forecasting.

Introduction

Stock price forecasting is a critical yet challenging task due to the volatile and complex nature of financial markets. Accurately predicting stock returns, especially in high- and medium-frequency trading, requires models capable of quickly and accurately processing large volumes of temporal data while considering dynamic market conditions. Existing approaches often lack the ability to simultaneously capture long-term temporal dependencies and adapt to real-time market signals, leading to suboptimal performance. Our project aims to address these challenges by designing a model that is able to forecast stock prices by combining market-aware techniques and advanced temporal representations. Our approach aims to provide a high-adaptive model, capable of processing intricate market data and making accurate predictions off of it. Our results show that the combination of these approaches in our model improves prediction accuracy and enhances trading performance when compared to standard financial metrics. The findings show the power of combining both contextual and temporal data in a single model.

Related Work

In order to achieve our intended goal, we looked at the research landscape to get a sense of the feasibility of our idea. We found two papers particularly interesting and relevant to our project goal, and aimed at integrating both of them into our novel transformer-based fusion stock trading model.

The first of the two papers we analyzed was: **Multi-Patch Prediction: Adapting LLMs for Time Series Representation Learning** (arXiv 2402.04852)): This paper introduces a novel time-series representation by segmenting the continuous numerical data (stock prices and such) into patches. Continuous time-series data present a unique challenge for LLMs to understand due to its sequential nature, need to model both short and long term dependencies as well as it irregularity. Patches are smaller, overlapping chunks of the full continuous data. These patches are fed into the transformer-based model proposed by the authors of the paper. Segmenting the data in this fashion enables the model to gain a better understanding of temporal dependencies by focusing on localized (per patch) patterns. The patching approach also reduces the computational overhead of computing very long sequences at once, and allows for a more detailed analysis of the smaller chunks. A self-attention layer within the model allows it to assign varying importance to different elements within the patches. It also allows it to capture intra–patch understands as well as cross-patch relationship, further bolstering its temporal understanding.

Pre-training occurs in two parts: first, patch prediction. The model is passed in a sequence of patches and is tasked with predicting a missing or masked patch within the sequence. Second, sequence reconstruction occurs, where the model reconstructs the entire sequence from a subset of its patches. Additionally, multi-patch fine-tuning occurs, where cross-attention between patches occurs, further boosting the models ability to understand temporal dependencies between patches.

The second paper we analyzed was **MASTER: Market-Guided Stock Transformer for Stock Price Forecasting** (arXiv 2312.15235). This paper proposes a novel framework which addresses the limitations of traditional transformer based models for stock price prediction by incorporating a market-gated mechanism. This mechanism dynamically adjusts features importance based on real-time market signals. This adjustment of features allows the model to prioritize more relevant features while disregarding less relevant features.

The model integrates features such as volatility indexes, macroeconomic indicators, stock price action, and volume as well as many other. Based off of these features the model has a mechanism which learns how to dynamically weigh the importance of each. This mechanism makes sure that the model accurately adjusts to market shifts, and understands what weights to change. Temporal attention occurs, where time-specific patterns are model, capturing relationships within a stocks trading action over different time intervals. Spatial attention also occurs, accounting for dependencies between multiple stocks, which is critical since different stocks affect each other.

The framework proposed by this paper predicts multiple outputs, including future stock returns, providing a comprehensive view on the stocks performance. To do this, a large set of input features are utilized, including historical prices, technical indicators, macroeconomic features, and volatility indexes, giving the model a high-level of predictive power.

Data

Our model relies on a diverse set of data inputs to ensure a very comprehensive and accurate prediction. These include, but are not limited to, historical stock price data, market context features (macroeconomic features), and technical indicators. We utilized a comprehensive stock price dataset comprising historical price and volume data for five major technology companies: Apple Inc. (AAPL), Microsoft Corporation (MSFT), Alphabet Inc. (GOOGL), Amazon.com, Inc. (AMZN), and Meta Platforms, Inc. (META). This data was obtained from yfinance and spans from January 1, 2020, to January 1, 2023. The technical indicators included allow the model to get a better sense of the future trend of each stock based on . These indicators include but are not limited to Relative Strength Index (RSI), which captures whether a stock is overbought or oversold. Bollinger Bands, which measure volatility. Moving Averages, which highlight trends by smoothing out price fluctuations and noise. Trading Volume which indicates the level of market activity as well as interest in specific stocks. Altogether, these indicators capture the patterns and fluctuations in a stock's behavior. In addition to technical indicators, market context features are also passed into the model.

Data preparation additionally takes place to make the input data more suitable for our given model. The raw data goes through several preprocessing steps to ensure compatibility with our model. Missing or inconsistent data points were handled with interpolation and statistical imputation. Normalization occurred, where features were scaled to ensure uniformity and prevent any feature from becoming too dominant. Segmentation took place, where temporal data was segmented into patches (of length 60 i.e. 60 days), which are smaller, overlapping chunks of the continuous numerical data. New features were engineered, as well, such as average trading volume, which should further enhance the predictive power and accuracy of the model. The dataset was also divided into training (70%), validation (10%), and testing (20%) sets to evaluate the models' performance accurately.

Methods

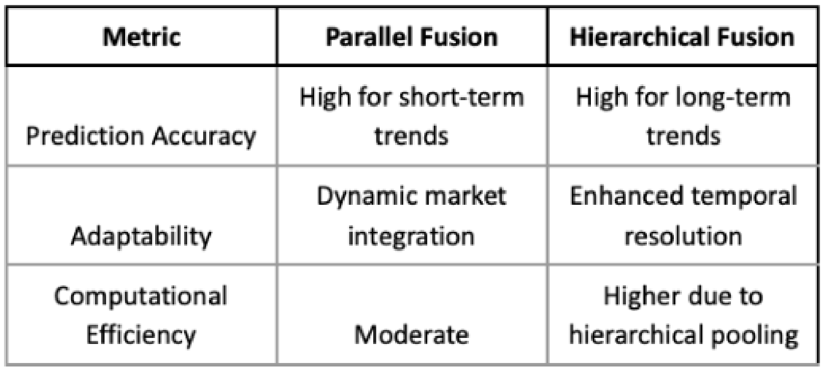
From the multi-patch prediction paper, we sought to incorporate the advanced temporal encoding, allowing our model to get a better temporal understanding of the continuous numerical data. From the MASTER paper, we aimed at incorporating the market-guided feature gating mechanism in our model. This approach would ensure our models adaptability to volatile market conditions, as well as being able to capture both long and short term trends. Our methodology builds upon research insights from state-of-the-art frameworks from recently published papers.

We first aimed at creating a parallel fusion framework. We adopt the patch-based strategy from the Multi-Patch Prediction approach. Specifically, we segment the time-series data into overlapping patches and process each patch through a transformer encoder. This patching procedure enhances our model’s scalability and ability to capture both short- and long-term dependencies without reiterating the full rationale already presented.The self-attention layers of our model will further capture relationships both within and across patches, over varying time scales. The second part of this fusion architecture is the market-guided gating mechanism, which dynamically weights different input features based on real-time market signals. Market gating ensures that the model remains adaptive to the shifts of the market by allowing the model to dynamically “gate” which features influence the predictions. Parallel fusion processes both temporal and market features independently before combining them in the prediction layer. This design ensures a simultaneous focus on localized patterns and broader market trends. The inter and intra stock relationships are effectively captured using multi-head attention. This framework has two pathways. The first, the temporal pathway, projects input features into a high-dimensional latent space. Positional encodings ensure that temporal order is preserved. The transformer encoders process temporal relationships within patches. The second pathway, the market-gating pathway, transforms the market features into gating vectors. Sigmoid activation functions dynamically control the contribution of each feature, depending on their relevance. These two pathways occur in parallel, and once they are both completed, the data reaches the fusion layer. In the fusion layer, adaptive weights adjust the influence of both the temporal and market-gating pathways, deciding on how relevant each is, enhancing prediction accuracy.

The hierarchical fusion architecture expands on the parallel fusion framework by incorporating multi-scale temporal patterns. It does this by capturing short-term dependencies in finer temporal representations (patches), while capturing long-term trends at coarser scales using hierarchical pooling. This framework uses time-scale attention where intra-level dependencies are modeled, which ensures that critical temporal features are emphasized. Cross-scale attention also occurs, where features across different hierarchical levels are integrated, ensuring a comprehensive temporal representation.

We compared our approach to other, common alternative approaches out there. The first alternative we considered was Long Short-Term Memory (LSTM) Models. These models are commonly used to process continuous numerical data as they are effective in capturing temporal dependencies, both short term and long term. They, however, struggle with very long sequences and are computationally intensive compared to transformers (due to less parallelizability). Traditional statistical models such as ARIMA were also considered. ARIMA is suitable to handle simple time series data and is effective at capturing linear patterns. It, however, fails to capture complex, non-linear market activity, and lacks adaptability. We thought that our approach was the best, since it combines the strengths of temporal representation and market adaptations, while also being highly parallelizable, overcoming the shortcomings of other approaches. By integrating insights from these other approaches, and extending their capabilities, we are trying to ensure robustness and scalability.

Here’s a table outlining the comparison:



Metric	Parallel Fusion	Hierarchical Fusion
Prediction Accuracy	High for short-term trends	High for long-term trends
Adaptability	Dynamic market integration	Enhanced temporal resolution
Computational Efficiency	Moderate	Higher due to hierarchical pooling

Experiments

Parallel Fusion Architecture

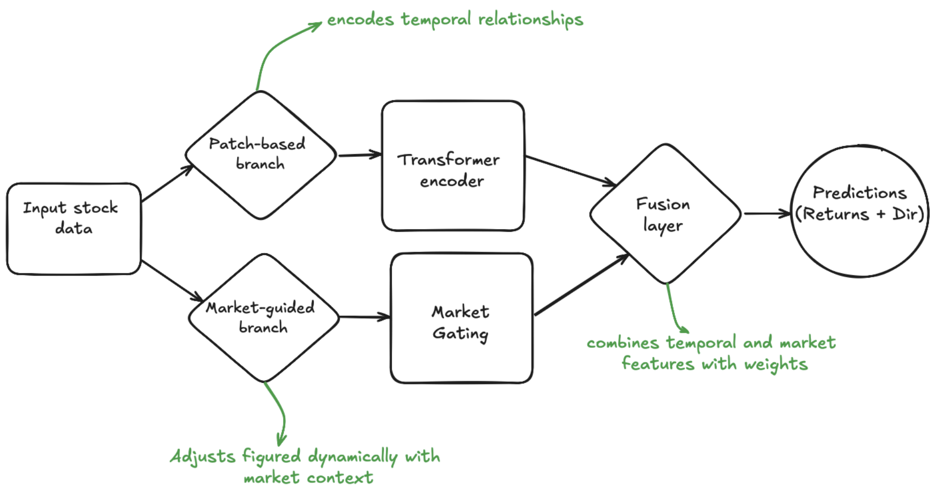
The Parallel Fusion model was the first attempt at creating a model whose architecture combined components from both *MASTER* and *aLLM4TS*. It integrates temporal features and market-driven patterns through separate pathways that run in parallel, combining them adaptively in the final prediction layer. It incorporates both time-series dependencies and market conditions - one from each branch - ensuring that the predictions reflect real-time market behavior. The key components of the model are as follows:

Key Components

1. **Intra-StockAttention (Temporal Pathway)**: This processes the input stock time-series data to capture temporal dependencies using a transformer encoder:
    a. A linear layer projects the input features to a higher-dimensional latent space.
    b. Positional encoding is added to retain the temporal order.
    c. A transformer encoder layer is applied to model temporal relationships within stock sequences.
2. **MarketGating (Market Pathway)**:
    a. Market features are projected to the same latent dimension as the time-series features.
    b. A gating mechanism applies a sigmoid activation to dynamically select relevant features.
3. **CrossTimeAttention**: Models dependencies across time periods and integrates inter-stock correlations:
    a. Multi-head self-attention refines temporal embeddings by considering relationships across time steps.
    b. Layer normalization and residual connections stabilize training.
4. **Patch-Based Processing**:
    a. The input sequence is divided into smaller, non-overlapping patches

    b. Patches are processed through a linear layer followed by layer normalization to reduce dimensionality and capture intra-patch relationships.
    c. The processed patches are then used in the fusion mechanism instead of the full sequence, enabling scalable processing.
5. **Fusion Layer**: Combines temporal and market features adaptively:
    a. Pooling used to aggregate weights from both features and learnable weights used to adjust the relative importance of each pathway during training.
    b. The combined features are passed through fully connected layers for final prediction.
6. **Optimizer and Loss Functions**: The parallel fusion architecture used a dual-objective loss to optimize both prediction and directional accuracies.
    a. **Optimizer**: Adam Optimizer was used with a learning rate scheduler used to adjust the learning rate during training.
    b. **Returns Loss**: Minimizes Mean Square Error (MSE) between predicted and actual returns.
    c. **Directional Loss**: Binary Cross-Entropy (BCE) with logit loss used to evaluate predicted and true direction of returns.
    d. **Final Loss**: Weighted sum of both losses balanced by hyperparameter ***λ***
7. **Hyperparameters**
    a. **Batch Size**: A consistent batch size of 128 was used across all experiments.
    b. **Training Epochs**: Models were trained for a max of 20 epochs and had an early stopping patience of 3 to prevent overfitting.

The architecture of the model can be seen below:


encodes temporal relationships
Patch-based branch
Transformer encoder
Input stock data
Fusion layer
Predictions (Returns + Dir)
Market-guided branch
Market Gating
combines temporal and market features with weights
Adjusts figured dynamically with market context

Given a constraint on the computational resources for this investigation, the parallel fusion model was designed with scalability and efficiency in mind. These metrics informed design choices with regards to reference papers and model components. By integrating algorithms such as patch-based processing from *aLLM4TS* and various attention mechanisms from *MASTER*, the result was a flexible and efficient fusion model whose results will be discussed in subsequent sections.

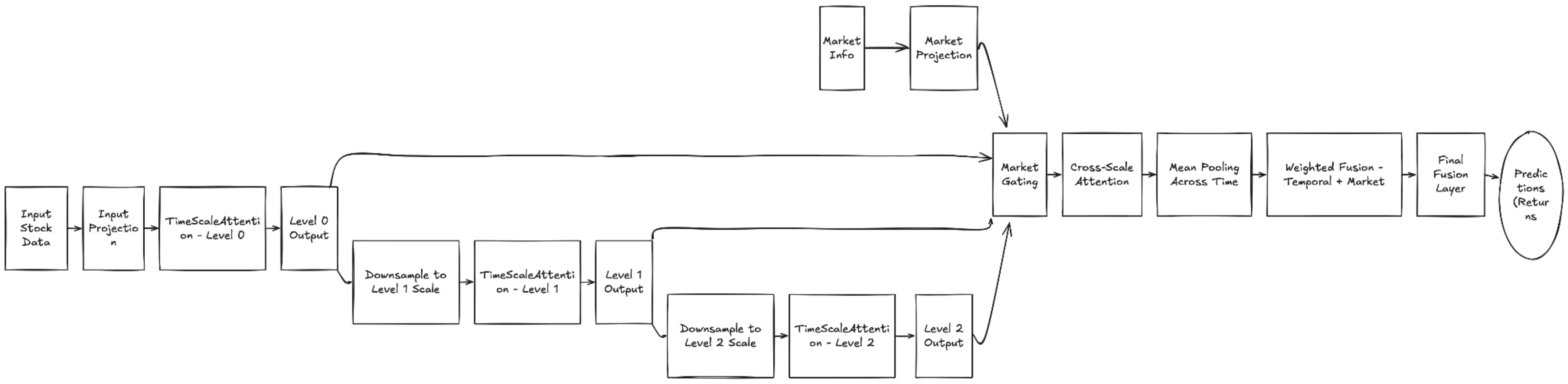
Market-Aware Hierarchical Fusion Architecture

The Market-Aware Hierarchical Fusion model represents an advanced approach to stock prediction by processing data at multiple time scales while incorporating market conditions. It uses a hierarchical structure to capture patterns at different temporal granularities (daily, weekly, monthly) and combines them with market information through an adaptive fusion mechanism.

Key Components

1. **TimeScaleAttention (Multi-Scale Processing):**
    a. Implements multi-head attention at each time scale
    b. Includes layer normalization and residual connections
    c. Dropout for regularization during training
2. **HierarchicalProcessor (Temporal Pathway):**
    a. Projects input features to model dimension
    b. Processes data at multiple time scales (typically 3 levels: daily, weekly, monthly)
    c. Uses adaptive pooling to progressively reduce sequence length
    d. Maintains separate processors for each hierarchical level
3. **Market Gating Mechanism:**
    a. Projects market information to model dimension
    b. Separate gating for each hierarchical level
    c. Dynamic feature weighting based on market conditions
    d. Sigmoid activation for gate values between 0 and 1
4. **Cross-Scale Attention:**
    a. Integrates information across different time scales
    b. Allows interaction between hierarchical levels
    c. Maintains batch-first processing for efficiency
5. **Adaptive Fusion Layer:**
    a. Learnable weights for temporal and market features
    b. Concatenates features from all hierarchical levels
    c. Final prediction through fully connected layers
    d. Dropout for regularization
6. **Optimizer and Loss Functions:**
    a. Returns Loss: MSE for price return prediction
    b. Direction Loss: BCE for movement direction
    c. Combined Loss: Weighted sum with 0.5 factor for direction loss
    d. Adam optimizer with learning rate scheduling
7. **Hyperparameters**
    a. Number of Hierarchical Levels: 3 (daily, weekly, monthly)
    b. Model Dimension: 32
    c. Number of Attention Heads: 4
    d. Dropout Rate: 0.1
    e. Sequence Length: 60 days
    f. Batch Size: 32

Here is the model architecture:

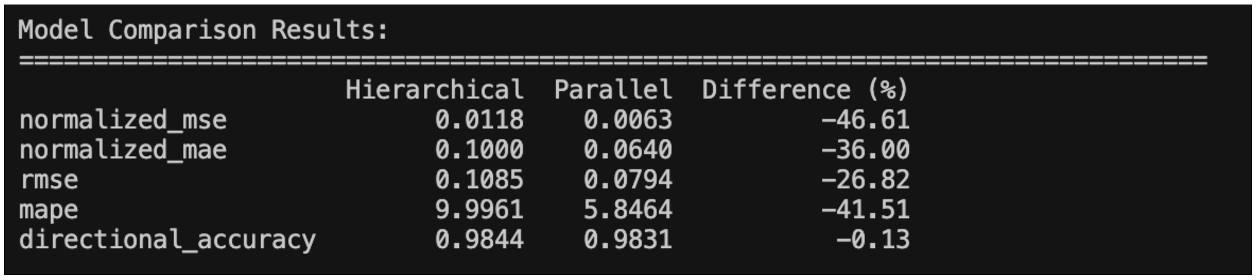


Comparative Summary

Feature	**Parallel Fusion**	**Hierarchical Fusion**
Fusion Mechanism	Adaptive weights for temporal and market	Cross-scale attention
Prediction Outputs	Returns and directional trends	Returns
Scalability enhancements	Optional patch-based processing	Hierarchical scaling without patching
Focus	Balances temporal and market patterns	Emphasizes temporal patterns
Gating Mechanism	Applied once after intra-stock attention using a unified gating mechanism to the entire temporal representation at once.	Applied at each hierarchical level using a gate at each hierarchical level, and at different temporal scales(daily, weekly, monthly)

Results and Evaluation

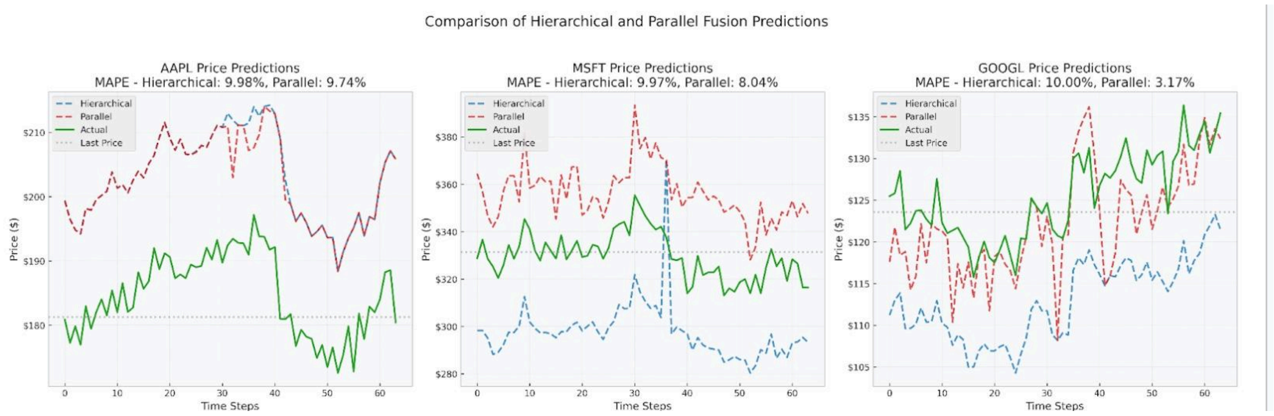
After the architecture was built out, every component of the model was tested individually and in conjunction with the other components (attention, gating, and temporal mechanisms). Both models were run on the same test data over the same time frame as mentioned above. The following evaluation metrics were recorded for both model architectures after comparing them to actual stock prices:

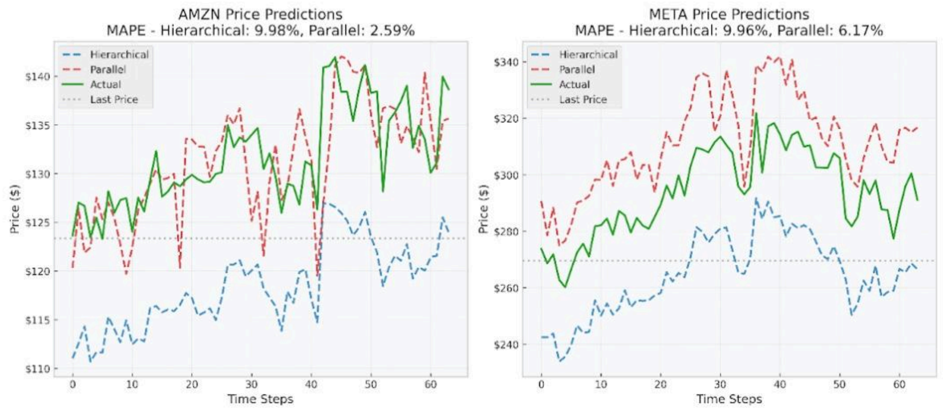


```
Model Comparison Results:
================================================================================
                      Hierarchical  Parallel  Difference (%)
normalized_mse              0.0118    0.0063          -46.61
normalized_mae              0.1000    0.0640          -36.00
rmse                        0.1085    0.0794          -26.82
mape                        9.9961    5.8464          -41.51
directional_accuracy        0.9844    0.9831           -0.13
```

As can be seen from the results above, the parallel fusion model performs significantly better than its hierarchical counterpart across most of the metrics that were recorded. This suggests that the parallel model was able to extract temporal and market dynamics better than its hierarchical counterpart. This might have to do with the way each of the two models perform market gating. Parallel fusion performs market gating once after the intrastock attention and uses a single, unified, gating mechanism whereas hierarchical fusion gates at every level *separately* at every hierarchical level using different gating modules.

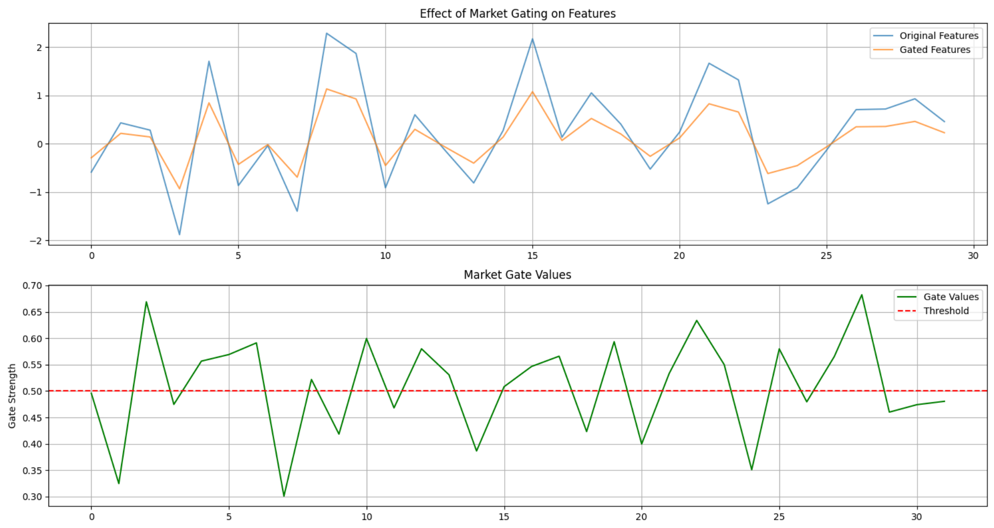
The results were visualized as follows:





As can be seen in the graphs, neither model can very accurately predict returns for stocks at this point in time. While both models achieved high directional accuracies, the actual predictions are off with the models' predictions being shifted above or below the actual price(s) of the stock(s). In general, however, parallel fusion performs better than hierarchical and its curve is closer to the actual price curve.

The experimental evaluations underscore the efficacy of the Parallel Fusion model in stock price forecasting tasks. The integration of parallel attention mechanisms, coupled with market-aware gating, facilitates robust feature extraction and dynamic adaptation to market conditions. Comparative analyses reveal that the Parallel Fusion model outperforms the Hierarchical Fusion baseline across crucial evaluation metrics, including error rates and directional accuracy.



Key Insights:

- <u>Enhanced Feature Extraction</u>: Parallel attention allows the model to concurrently capture multiple temporal dependencies, leading to more nuanced feature representations.
- <u>Dynamic Market Adaptation</u>: The market gating mechanism effectively modulates feature contributions, enhancing the model's responsiveness to market fluctuations.
- <u>Improved Prediction Accuracy</u>: Quantitative and qualitative results consistently demonstrate the Parallel Fusion model's superiority in predicting stock prices accurately.

Implications for Future Work:

- <u>Incorporation of Additional Market Indicators</u>: Extending the model to include more diverse market indicators could further enhance prediction capabilities.
- <u>Scalability and Generalization</u>: Testing the model across a broader range of stocks and market conditions would validate its scalability and generalizability.
- <u>Integration with Deep Learning Enhancements</u>: Exploring advanced attention mechanisms or integrating with other deep learning architectures might yield additional performance gains.

<u>Conclusion</u>

The comprehensive experiments conducted affirm the Parallel Fusion model's robust performance in stock price forecasting. By leveraging parallel attention mechanisms and market-aware gating, the model achieves superior predictive accuracy and adaptability compared to traditional hierarchical approaches. These findings highlight the potential of advanced fusion architectures in financial modeling and decision-making processes.

Appendix

Our code: https://github.com/Tarunv711/DLfinal

Multi-Patch Prediction: Adapting LLMs for Time Series Representation Learning (arXiv 2402.04852)) code: https://github.com/yxbian23/aLLM4TS/tree/main

MASTER: Market-Guided Stock Transformer for Stock Price Forecasting (arXiv 2312.15235) code: https://github.com/ SJTU-Quant/MASTER