S&P 500 Price Prediction Using Enhanced Temporal Convolutional Networks

Abdelaziz ElHelaly
Zewailcity of Science, Technology & Innovation
Email: s-abdelaziz.eleisawy@zewailcity.edu.eg

***Abstract*—This study presents a deep learning-based approach to forecast the S&P 500 index using Enhanced Temporal Convolutional Networks (TCN). Two methods were developed: the first averages predictions from models trained on various input sequence lengths; the second dynamically selects the best-performing prediction per time point across these sequence lengths. The model integrates financial indicators, macroeconomic data, and news sentiment, with advanced preprocessing techniques such as wavelet denoising and feature lagging, enhancing both accuracy and interpretability.**

I. Introduction

Forecasting financial markets, particularly the S&P 500 index, poses a substantial challenge due to the non-stationary, noisy, and nonlinear nature of time-series data. Traditional approaches like ARIMA or exponential smoothing lack the ability to capture long-term dependencies and exogenous inputs, particularly when data stems from multiple sources.

Recent advancements in deep learning have introduced architectures tailored for sequential data, such as RNNs and LSTMs. However, these models often suffer from vanishing gradients and inefficient training on long sequences. Temporal Convolutional Networks (TCNs), which rely on dilated convolutions and causal architecture, offer a scalable, memory-efficient, and accurate solution.

This work enhances the classical TCN model through a dual strategy: (1) leveraging ensemble outputs of different sequence lengths, and (2) implementing an adaptive weighting system across multi-resolution temporal branches to dynamically select optimal predictions.

II. Data Acquisition and Preprocessing

The dataset was constructed from several high-quality sources:

- **Alpha Vantage API**: Provided historical price data for the SPY ETF, a close proxy for the S&P 500 index, along with technical indicators such as Simple Moving Average (SMA), Relative Strength Index (RSI), and Bollinger Bands.
- **FRED API**: Supplied macroeconomic indicators including the Federal Funds Rate and the 10-year Treasury yield, offering economic context to market behavior.
- **News API**: Delivered daily financial headlines from which sentiment polarity scores were computed using the TextBlob natural language processing tool.

Additional feature engineering was performed:

- Temporal encoding: Day of the week, month, and year.
- Lag features: Close prices lagged by 1, 7, 30, and 252 trading days.
- Rolling statistics: 7-day moving average and 30-day rolling standard deviation to capture trend and volatility.
- Wavelet Denoising: Applied to close price, volume, and volatility features using the Daubechies wavelet to reduce noise without distorting the signal.

Missing values from economic indicators were forward-filled, while missing sentiment scores were imputed with 0 to denote neutral sentiment. All numerical features were normalized using MinMaxScaler.

III. Temporal Convolutional Network (TCN) Architecture

The core TCN architecture comprises several Temporal Blocks, each including:

- 1D Causal Convolutions: Prevent information leakage from future timesteps.
- Dilation: Each layer's dilation factor increases exponentially, allowing the receptive field to cover longer histories efficiently.
- Residual Connections: Enhance gradient flow and accelerate training convergence.
- Chomp1D Layer: Ensures output size consistency after padding.
- Dropout and ReLU: Applied after each convolution for regularization and non-linearity.

These blocks are stacked to create a deep architecture capable of modeling long-range dependencies. The architecture is extended to a multi-branch configuration, where each branch is trained on a different sequence length. Outputs from all branches are merged using two strategies detailed below.

IV. Prediction Strategies and Training Methodology

Two different methodologies were implemented:

A. 1) Fixed Sequence Averaging

For this method, multiple TCN models were trained individually using sequence lengths of 50, 100, 150, and 200. Each model independently predicts the next-day close price. The outputs are then averaged, leveraging ensemble learning to reduce variance and improve generalization. This is particularly effective when capturing both short-term patterns (via shorter sequences) and longer-term trends (via longer sequences).

B. 2) Adaptive Sequence Selection with Multi-Scale TCN

In this more advanced method, a shared model is trained using multiple input sequences simultaneously. Each sequence branch passes through its own TCN layers. The outputs are fed into a learned Softmax-based gating mechanism that determines the relative importance (weight) of each branch for every prediction.

This adaptive attention mechanism is dynamic over time and offers more flexibility in volatile market conditions. The model is trained end-to-end using PyTorch Lightning with Mean Squared Error as the loss function and Adam optimizer. Wavelet-denoised features were used for both strategies.

V. Results and Visualization

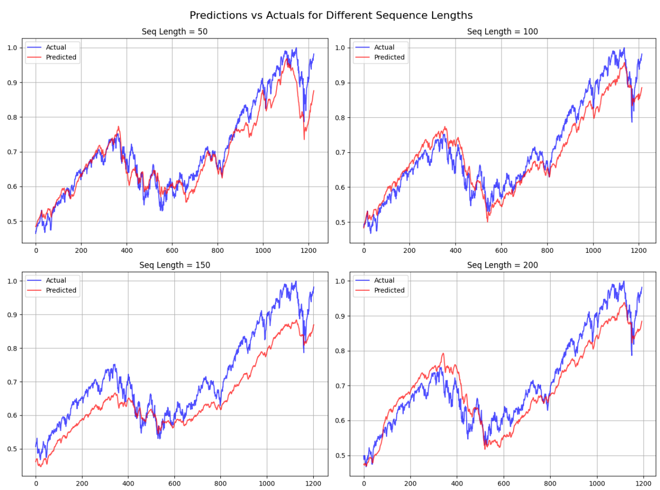

Fig. 1: Predictions vs Actuals for Different Sequence Lengths: Each subplot demonstrates model performance for input lengths of 50, 100, 150, and 200 timesteps. Shorter sequences react faster to abrupt changes; longer sequences preserve trend consistency.

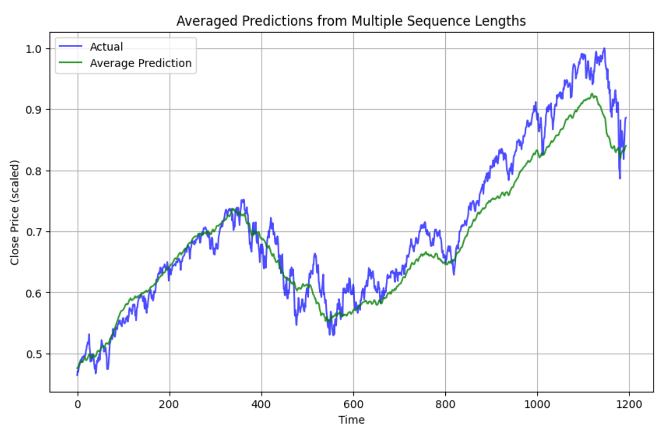

Fig. 2: Averaged Predictions from Multiple Sequence Lengths: Combining outputs from all models enhances stability and generalization, especially during turbulent market periods.

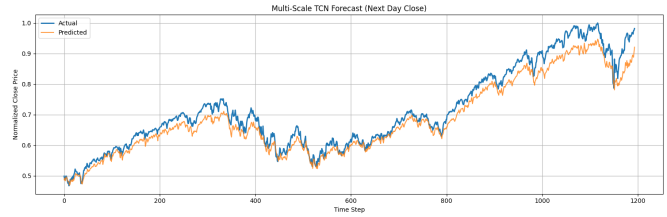

Fig. 3: Multi-Scale TCN Forecast with Adaptive Weights: The model dynamically chooses the best sequence length for each prediction. It excels at both local reactivity and long-term tracking.

The fixed averaging strategy produced robust forecasts across a variety of scenarios but sometimes lagged during sharp market reversals. In contrast, the adaptive approach achieved lower average error by allocating more weight to the best-performing sequence length on a per-instance basis. This flexibility made it more resilient to noise and non-linearity.

VI. Conclusion

We presented two TCN-based methods for stock market forecasting. The first approach relies on ensemble averaging from models trained at different temporal resolutions, balancing global and local patterns. The second introduces an adaptive method that dynamically selects the best sequence length per forecasted time point.

Experimental results validate the efficiency of both methods, with the adaptive model offering superior flexibility and sharper alignment with actual price movements. For future work, hybrid models incorporating attention layers or reinforcement learning policies may further enhance accuracy in volatile markets.