Utilizing a Transformer and Imbalances in Market Fair Value for an Algorithmic Trading System

Bora Yimenicioglu

Abstract

In algorithmic trading, supervised machine learning techniques like decision trees and linear regression are commonly used in tandem with well-known technical analysis strategies. Examples include moving average crossovers, or tracking oscillators like the relative strength index (RSI), or moving average convergence/divergence (MACD). This project explores the idea of using a deep learning algorithm, specifically the transformer, instead of more conventional supervised machine learning algorithms, to predict future market sentiment.

A detailed review of the data preprocessing steps, model architecture, and evaluation metrics are provided to help with the clarity and replicability of this research. The significance of this project comes from using a lesser-known retail trading strategy where buy-side and sell-side imbalances are targeted within the market together with the neural network. This is achieved by engineering a dataset where each entry consists of the last 15 days of price action, including the open, high, low, and close prices for each day, paired with indicators like volume. To use the imbalances in conjunction with the algorithm, each entry should include the prices' distance from a buy-side and sell-side imbalance. The output for each entry is the overall bullish/bearish sentiment for the following five days.

Through this approach, the transformer is designed to capture long-term trends in the data to predict future market direction. After the model was trained on this dataset, it was evaluated using accuracy and precision metrics, with results indicating the transformer model successfully captured long-term trends better than traditional machine learning models with the addition of imbalance information improving model accuracy. This model could help stock traders improve their existing strategies and provide additional confirmation by providing accurate market predictions.

Keywords: **Transformer, Algorithmic Trading, Fair Value Gaps, Deep Learning, Financial Time Series**

Introduction

Algorithmic trading automates trading processes for both institutional market makers and retail traders, with 60 to 75% of trading volume being automated in major markets (Groette, 2024). These strategies follow logical rules created by traders to automatically place trades (Seth, 2023). Historically, market makers used significant capital and high-frequency trading (HFT), which limited individual traders from entering the field (Chen, 2024). However, advances in technology have made automated systems more accessible to retail traders. This paper focuses on retail algorithmic trading because the resources used by market makers are beyond the scope of retail traders.

Conventional retail algorithmic trading methods have limitations that prevent them from providing a real market edge. Common strategies include trend-following techniques like the Simple Moving Average (SMA) and oscillators like the Moving Average Convergence/Divergence (MACD) indicator. Other methods include mean reversion strategies, such as the Relative Strength Index (RSI) and Bollinger Bands, which track price divergences and convergences (Chen, 2024). These strategies rely on historical data and simple indicators, failing to capture important price dependencies. Moreover, they lack flexibility to consider broader market context and sentiment due to their strict rules. While retail traders incorporate discretion into their strategies, automated systems lack this subjectivity, which can be both a strength and a limitation.

Machine learning has become more accessible, offering tools like linear regression, decision trees, random forests, and neural networks, such as recurrent neural networks (RNNs) and long short-term memory (LSTM) networks. LSTMs handle sequential time-series data, capturing complex temporal dependencies in price, and provide a deeper understanding of patterns, introducing a sense of discretion and subjectivity that rule-based systems lack. However, these algorithms have drawbacks. Linear regression fails to capture the nonlinear nature of price. Decision trees and random forests suit nonlinear data but struggle in unseen market conditions. LSTMs falter with large datasets, critical when trends span decades.

A key feature of this study is the use of Fair Value Gaps (FVGs), representing price inefficiencies when there is a gap between market value and the "fair value" of a financial asset. Illustrated by a three-candle sequence, these gaps arise from imbalances between buy-side and sell-side orders. These patterns are quantified into a dataset for model training.

The limitations of conventional retail algorithmic trading strategies and the constraints of machine learning pose challenges for retail traders seeking an edge with automated systems. They struggle to capture long-term dependencies in data, detect anomalies, and consider broader context in decision-making. This research explores transformer models to investigate their potential in financial time series, aiming to provide retail traders with a greater advantage in automated trading systems.

Literature review

Fang et al. (2014) evaluated the profitability of 93 technical market indicators, including advance/decline lines, volatility indices, and the Arms index. They found little evidence that these indicators

could reliably predict stock market returns, and the indicators did not outperform a simple buy-and-hold strategy.

Wen et al. (2023) reviewed the application of transformers to time series modeling, highlighting their appeal for capturing long-range dependencies. They explored modifications like learnable positional encodings and efficient attention mechanisms. Combining transformers with generative models like Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs) has led to improved performance in applications like time series forecasting and anomaly detection, though performance can degrade with longer input sequences.

Bilokon and Qiu (2023) compared LSTMs and transformers in financial time series using high-frequency limit order book data. They found that transformers offer advantages in absolute price sequence prediction, while LSTM models perform better in predicting price differences in specific trading strategies.

Lara-Benítez et al. (2021) evaluated transformers for univariate time series forecasting, comparing their accuracy and efficiency against LSTM and CNN models. Transformers outperformed LSTMs and CNNs in accuracy, although they required more time to generate forecasts.

Zhang et al. (2024) explored transformers in financial markets using a custom transformer architecture for quantitative trading strategies. They found that transformers, using transfer learning from sentiment analysis, showed success in forecasting future market sentiment, outperforming many traditional factor-based strategies.

This existing literature provides a strong starting point for exploring the potential of transformer models in financial time series analysis and where they fill the gaps in terms of limitations with past strategies and methodologies.

Methods

This research aims to explore how effective transformer models can be in predicting market sentiment in the context of financial time series analysis. The methodologies involved several different transformer architectures to capture both short and long term dependencies in the data.

1. Data Preparation.

c. **Source:** Historical price data of Apple Inc's stock (AAPL) over the period from January 1, 2005, to December 31, 2023, were collected from Yahoo Finance.

d. **Data Points:** Each entry consists of open, high, low, close prices (OHLC) for the last 15 days, volume data, and distances from buy-side and sell-side imbalances (FVGs).

e. **Fair Value Gap Identification:**

i. **Criteria:**

1. **Bullish FVG (Buy-side Imbalance Sell-side Inefficiency - BISI):** Identified using a three-candle pattern where the first and third candles are consolidation candles, and the middle candle is a strong bullish displacement. There is a gap between the first and third candle wicks.
2. **Bearish FVG (Sell-side Imbalance Buy-side Inefficiency - SIBI):** Similar but with bearish displacement.

ii. **Implementation**

1. The *Imbalance* class checks for valid bullish or bearish imbalances using methods *is_bullish_valid()* and *is_bearish_valid()*.
2. Distances from the current price to the identified FVGs are calculated and included as features.

f. **Feature Engineering:**

i. **Dataset Construction:**

1. **OHLCV Data:** Open, high, low, close prices, and volume for the past 15 days.
2. **Imbalance Features:** Number of bullish and bearish imbalances, distances to nearest bullish and bearish FVGs.
3. **Target Variables:** Total number of bullish and bearish days in the next 10 days.

g. **Data Scaling and Normalization:**

i. Applied StandardScaler to normalize features.

ii. Data split into training (January 1, 2005, to December 31, 2023) and testing sets (January 1, 2024, to July 16, 2024).

2. Model/Architecture Implementations

Five transformer-based models were implemented in this study, each designed to capture both short and long term dependencies in financial time series data. The decision behind choosing transformer models and the specific architectures are discussed below.

Rationale

Transformer models are very effective in capturing long-range dependencies within sequential data due to their self-attention mechanisms. In financial time series, patterns and dependencies can be seen across many different time scales, and capturing these relationships is very important for accurate predictions.

Explanation of Architectural Components and Their Unique Roles

1. **Transformer-based CNN:**
   a. Purpose and Rationale: CNN layers effectively capture short-term patterns (rapid price change) whilst the transformer encoder is excellent at modeling long-term relationships.
   b. **Architecture:**
      i. CNN Layers: Three Conv1D layers with ReLU activations and MaxPooling to capture short-term patterns.
      ii. Transformer Encoder: Four layers with eight attention heads; sinusoidal positional encoding added to retain temporal order.
      iii. Fully Connected Layer: Maps output to desired dimension.
2. **Masked Attention Encoder-Decoder Transformer Model:**
   a. **Purpose and Rationale:** Model sequential data while

preventing the model from accessing future information during training.

b. **Architecture:**
   i. Encoder and Decoder Layers:
      1. Each with six layers and eight attention heads.
   ii. Masked Self-Attention Mechanism:
      1. Masks future positions in the decoder to prevent information leakage.
   iii. Sinusoidal positional encoding added to input embeddings

3. **Traditional Vanilla Encoder Transformer with Sin/Cos Positional Encoding:**
   a. **Purpose and Rationale:** Simpler architecture with fixed positional information.
   b. **Architecture:**
      i. **Input Embedding:** Linear layer projecting input features to a model dimension.
      ii. **Sinusoidal Positional Encoding:** Provides explicit information about the position of data points.
      iii. **Transformer Encoder Layers:** Four layers with four attention heads.
      iv. **Output Layer:** Linear layer mapping the encoder's output to the target dimension.
4. **Traditional Vanilla Encoder Transformer with Learnable Encoding:**
   a. **Purpose and Rationale:** Simpler architecture with learnable positional information.
   b. **Architecture:**
      i. Input Embedding:
         1. Same as above.
      ii. Learnable Positional Encoding:
         1. Positional encodings are parameters learned during training.
      iii. Transformer Encoder Layers:
         1. Same as above.
      iv. Linear Output Layer\
5. **Transformer-Based Variational Autoencoder (VAE):**
   a. **Purpose and Rationale:** VAEs are powerful as they learn the latent representations of data distributions, potentially capturing underlying patterns in the data. The *stochastic* nature of VAEs allow them to account for uncertainty in the data, thus reacting better to anomalies.
   b. **Architecture:**
      i. **Embedding Layer:** Projects input features to a model dimension.
      ii. **Encoder:** Six layers encoding the input sequence into a latent space.
      iii. **Latent Space Representation:** Computes mean and log-variance for the latent distribution.
      iv. **Reparameterization Trick:** Allows sampling from the latent space during training.
      v. **Decoder:** Reconstructs the input sequence from the latent representation.
      vi. **Output Layers:** Predict sentiments from the decoder outputs.

3. Training Process:
   a. **Hyperparameters:**
      i. **Optimizer:** Adam optimizer with learning rates:
         1. Models 1, 2, 5: 0.001
         2. Models 3, 4: 0.0001
      ii. **Loss Functions:**
         1. Models 1-4: Mean Squared Error (MSE) loss.
         2. Model 5 (VAE): Combination of Reconstruction Loss (MSE), Kullback-Leibler Divergence (KLD), and MSE losses for bullish and bearish predictions.
      iii. **Batch Size:** 32 or 64 depending on the model.
      iv. **Epochs:** 100 to 1000 depending on convergence.
   b. **Training Steps:**
      i. **Forward Pass:** Input data is fed through the model to obtain predictions.
      ii. **Loss Computation:** Calculated using the specified loss functions.
      iii. **Backward Pass:** Backpropagation is performed to compute gradients.
      iv. **Parameter Update:** Model weights are updated using the optimizer.
      v. **Regularization Techniques:**
         1. Gradient clipping with a maximum norm of 1.0.
         2. Dropout with a rate of 0.1 in transformer layers.
      vi. **Learning Rate Scheduler:** For some models, *ReduceLROnPlateau* reduces the learning rate once the validation loss starts plateauing.

4. Evaluation:
   a. Models are evaluated using mean squared error (MSE).
      i. Measures average squared difference between predicted and actual values.
      ii. Results: MSE calculated on training and test data for all models to assess performance.
   b. **Backtesting:**
      i. Custom Backtesting Script:
         1. Uses the model predictions to simulate trading decisions based on a set of rules.
      ii. **Trading Rules:**
         1. Buy Signal: Model predicts higher bullish sentiment.
         2. Sell Signal: Model predicts higher bearish sentiment.
      iii. **Metrics Calculated:**
         1. Total return over the backtesting period.
         2. Number of profitable trades.
         3. Final account balance.
   c. **Visualization:**
      i. Prediction Plots: Comparing true values and model predictions over time.
      ii. Loss Curves: For viewing training convergence.

5. Tools and Software:

a. Programming Language: Python 3.11
b. Libraries and Frameworks:
   i. **Data Handling:** Pandas, NumPy.
   ii. **Data Visualization:** Matplotlib, Plotly.
   iii. **Machine Learning:** PyTorch for model implementation.
c. Hardware: Tensors moved to and trained on a Nvidia RTX 4070 for efficiency

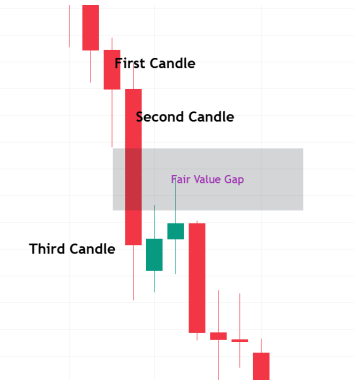

Fig. 1. Retracement and Respect of Bearish FVG. Figure 1. shows an example of a bearish fair value gap or SIBI.

Results

The the following section goes over the results of backtesting conducted on the transformer-based models

1. CNN-Transformer Model

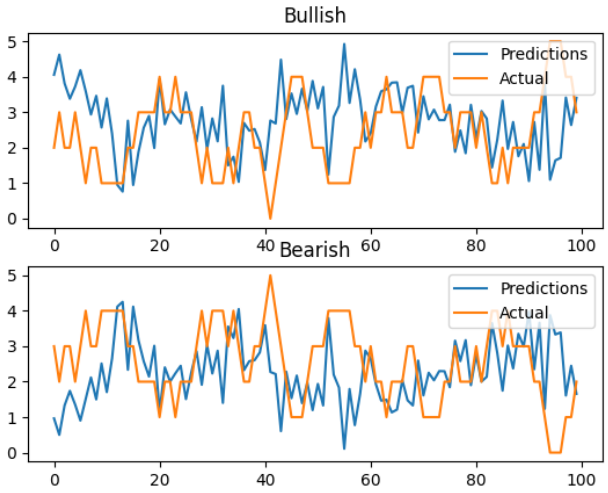

Fig. 2. Results of Evaluating CNN-Transformer

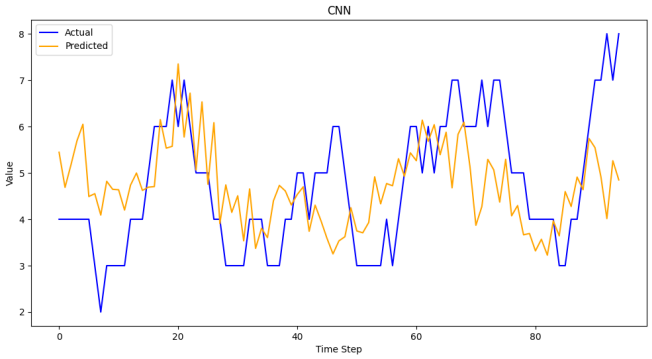

Fig. 3. Results of Evaluating CNN-Transformer on a Different Test Set

Performance Overview:

- MSE on Test Set: 1.792 Observations:

Observations:

- Demonstrated moderate accuracy in predicting both bullish and bearish sentiment.
- Effectively captured short-term patterns due to the CNN layers but struggled with long-range dependencies and high volatility.

2. Vanilla Encoder-Only Transformer

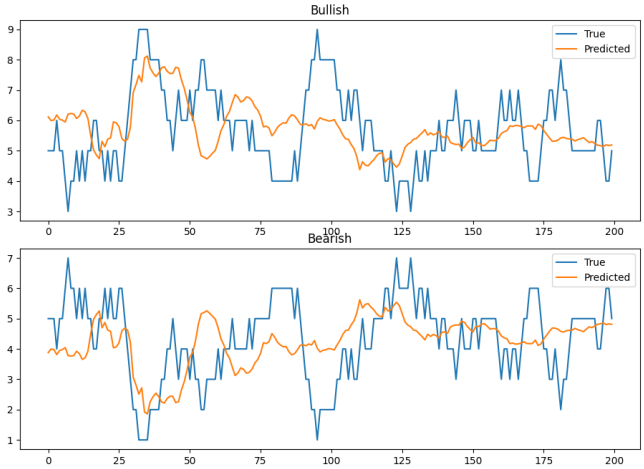

Fig. 4. Results of Evaluating Vanilla Encoder-Only Transformer on a Test Set Performance Overview:

Performance Overview:

- MSE on Test Set: 1.092

Observations:

- Had greater difficulty in predicting sentiment compared to other models.
- Captured some short-term patterns but overall performance was less satisfactory.

3. Masked-Attention Encoder-Decoder Transformer

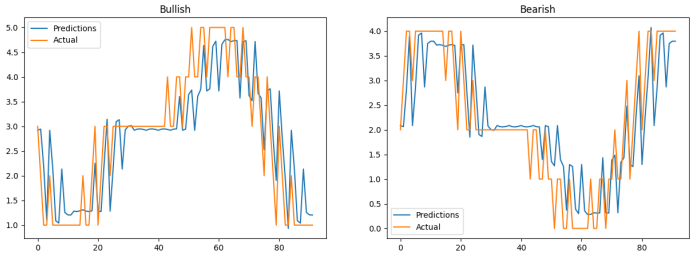

Fig. 5. Results of Evaluating Masked-Attention Encoder-Decoder Transformer Performance Overview:

Performance Overview:

- MSE on Test Set: 0.1967

Observations:

- Showed significant improvement in predicting sentiment compared to the CNN-Transformer.
- Better generalization to unseen data due to the masked attention mechanism.

Visualization:

- Predictions aligned more closely with actual market sentiment.

Fig. 6. Results of Backtesting Vanilla Encoder-only Transformer with FVG Data

Vanilla Transformer with FVG Data: Demonstrated clear profitability in a simulated trading environment.

- Initial Balance: $10,000
- Final Balance after 100 Trades: $10,244
- Total Return: 2.44%

Statistical Analysis

The MSE metric was calculated for predictions. The vanilla transformer with FVG data had an MSE of 1.79, showing a clear distinction in accuracy compared to models without FVG data. The masked attention model had a lower MSE, indicating better generalization to unseen data.

Discussion

The findings highlight the potential of transformer models in capturing future market sentiment and providing practical value to algorithmic trading systems. Models integrated with FVG data generally outperformed those with only OHLCV data.

Effectiveness of Transformer Models:

- The masked-attention transformer showed the best performance, aligning with Wen et al. (2023), indicating the importance of preventing information leakage from future time steps.

Importance of Fair Value Gaps (FVGs):

- The inclusion of FVG data improved model accuracy, suggesting that specialized market indicators enhance performance.

Comparison with Previous Studies:

- The results align with conclusions from previous research about the effectiveness of transformer models in time series forecasting.

Conclusion

This study demonstrates the effectiveness of transformer models in predicting future market sentiment for algorithmic trading systems. The integration of buy-side and sell-side imbalances through Fair Value Gaps provided a significant edge over traditional datasets and models, as evidenced by the improved accuracy and profitability in backtesting.

Key Takeaways:

- Transformer models, especially those with masked attention mechanisms, can capture complex temporal dependencies in financial time series data.
- Feature engineering, including the usage of technical market indicators like FVGs, improves model performance.
- The models developed can potentially help stock traders enhance existing strategies by providing more accurate market predictions.

Further Research:

Model Enhancements:

- Explore combining transformers with Generative Aversial Networks (GANs) or Long Short-Term Memory (LSTM) networks to potentially improve prediction accuracy.
- Explore different attention mechanisms, such as prob-sparse or adaptive attention, to handle larger sequences.

More Data:

- Include more technical/macroeconomic indicators and news sentiment analysis features to provide a broader context of market conditions.

Real-world Application:

- Test the model in live trading to test its practicality and adaptability to real-time market conditions.
- Implement adaptive learning techniques that would allow the model to adapt to changing market dynamics.

Bibliography

Bilokon, P. & Qiu, Y. (2023, October 17). Transformers versus LSTMs for electronic trading [Preprint]. arXiv. https://arxiv.org/abs/2309.11400

Chen, J. (2024, March 11). Algorithmic trading: Definition, how it works, pros & cons. *Investopedia*. Retrieved from https://www.investopedia.com/terms/a/algorithmictrading.asp

Fang, J., Qin, Y., & Jacobsen, B. (2014, June 14). Technical market indicators: An overview. *Journal of Behavioral and Experimental Finance*, 4, 25–56. https://doi.org/10.1016/j.jbef.2014.09.001

Groette, O. (2024, April 7). What percentage of trading is algorithmic? (Algo trading market statistics). *Quantified Strategies*. Retrieved from https://www.quantifiedstrategies.com/what-percentage-of-trading-is-algorithmic/

Lara-Benitez, P., Carranza-Garcia, M., & Riquelme, J. C. (2021, March 22). An experimental review on deep learning architectures for time series forecasting [Preprint]. arXiv. https://arxiv.org/abs/2103.12057

Seth, S. (2023, December 14). Basics of algorithmic trading: Concepts and examples. *Investopedia*. Retrieved from https://www.investopedia.com/articles/active-trading/101014/basics-algorithmic-trading-concepts-and-examples.asp

Wen, Q., Zhou, T., Zhang, C., Chen, W., Ma, Z., Yan, J., & Sun, L. (2023, February 15). Transformers in time series: A survey [Preprint]. arXiv. https://arxiv.org/abs/2202.07125

Zhang, Z., Chen, B., Zhu, S., & Langrene, N. (2024, October 23). Quantformer: From attention to profit with a quantitative transformer trading strategy. arXiv. https://arxiv.org/abs/2404.00424