Implicit Composite Alpha Factors with a Transformer Sequence Model

Shengchun Zhao
University of Michigan
GitHub Code Link: Final Project

***Abstract*—Almost all existing alpha factor discovery frameworks employ simple linear weighting schemes to combine the alpha factors; however, these frameworks are unable to capture nonlinear interactions or temporal dynamics. This project investigates whether the Transformer sequence model can capture the nonlinear structure between factors, thereby constructing a more effective composite alpha. In the simulated investment experiments, the composite alpha factor generated from the Transformer-based model provided higher returns and Sharpe ratios than the linear baselines. This result indicates that successfully capturing the nonlinear relationship between alpha factors can generate a higher rate of return than simple linear aggregation schemes, even if a single alpha contains a weak trading signal.**

I. Introduction

The alpha factors are the mathematical expressions that can capture the predictive signals of the financial market. An effective alpha factor can generate excess returns for traders. Approaches to discovering such alpha factor formulas have evolved from early genetic programming methods to deep learning models. In this paper, rather than generating new alpha factors, we treat an existing library of alpha factors as given and propose a Transformer-based framework that constructs a more powerful composite alpha by learning nonlinear and time-varying combinations of them.

In the early stages of alpha factor mining, genetic programming (GP) was regarded as a core tool. Chen et al. [1] conducted a representative research in which they transformed factor mining into a gene expression discovery problem by using gene expression programming (GEP). In this framework, RankIC is adopted as the fitness function to evaluate and evolve the candidate alpha factor formulas. Applying these GEP-generated alpha factors to the CSI 300 constituents, they found that these alpha factors could generate excess returns, indicating that the genetic algorithm can discover feasible stock trading signals effectively.

Nowadays, reinforcement learning methods for alpha discovery have evolved from simple predictors to more complex frameworks that not only have higher predictive power, but also could enhance alpha factor diversity. Guo et al. [2] introduced a Multi-Frequency Timing Residual (MFTR) network that processes daily 15-minute data with residual LSTM blocks and is trained with the negative Pearson correlation loss. In China A-share backtests, MFTR outperformed the baselines, achieving an annual return of 26.92%. Furthermore, Shi et al. [3] proposed AlphaForge, which is a two-stage framework that uses neural network predictors to guide the generation of alpha factor formula candidates, and then applies a dynamic linear scheme based on their Information Coefficient (IC) values to aggregate a small group of discovered alpha factors. Thus, a time-varying "Mega-Alpha" factor is formed. In the experiments of the CSI 300 Index, AlphaForge provided higher simulated portfolio returns than genetic programming and standard machine learning baselines.

Despite the rapid development of alpha discovery techniques, Transformer, as one of the most powerful models in deep learning, still has relatively limited research in the field of alpha factor mining. Although genetic programming and reinforcement learning generators have been able to generate effective single alpha factors, almost all existing frameworks rely on simple linear combination schemes (e.g., IC weighted averaging) to integrate these factors into the final trading signals. However, this linear aggregation method may fail to capture the complex nonlinear interactions among factors.

These limitations motivated us to shift our research focus from single alpha factor mining to the exploration of more effective combination methods for existing factors. In this project, we study different aggregation strategies based on a fixed classic Alpha101 library [4]. We choose standard linear models such as OLS and LASSO as baseline methods and compare them with the Transformer model that captures nonlinear interactions among factors. Our goal is to evaluate whether Transformer-based aggregators can generate compound alpha factors with stronger predictive performance than linear weighting schemes.

II. Methodology

We utilize an open-source dataset called "S&P 500 stock data" on Kaggle [5], which contains all constituents of the S&P 500 index. For each stock and trading day, the raw dataset provides open, high, low, close, and volume (OHLCV) from early 2013 to February 2018; we also calculate derived quantities such as daily returns and volume-weighted average price (VWAP).

We formulate the composite alpha factor construction as a supervised regression task. For stock i observed on trading day t, with closing price $P_{i,t}$, we calculate the 5-day forward log return in order to build a useful trading signal:

$$r_{i,t\to t+5} = \log(P_{i,t+5}) - \log(P_{i,t})$$

. In addition, we standardize these forward returns cross-sectionally on each date and use this quantity as the training target:

$$y_{i,t} = \frac{\log(P_{i,t+5}/P_{i,t}) - \mu_t}{\sigma_t}$$

, where μ_t and σ_t are the mean and standard deviation of the 5-day log returns on date t.

Furthermore, we construct a library of 39 candidate formulaic alpha factors in a separate Python module by following the public Alpha101 library instruction [4]. Of these 39 candidate expressions, some of them rely heavily on intraday high-frequency data (e.g., 15-minute prices) or produce many missing values. We therefore retain only those factors that (i) can be computed solely from daily OHLCV data and (ii) exhibit no massive missing values (NA) or infinite values. This filtering procedure yields a final set of $K = 36$ usable alpha factors.

For each stock i and date t, the model input is a 20-day history of these $K = 36$ alpha-factor values. Specifically, we construct a matrix

$$\mathbf{X}_{i,t} \in \mathbb{R}^{L\times K}, \quad L = 20$$

, where each row is a past trading day (from $t-19$ to t), and each column is the value of one of $K = 36$ formulaic alpha factors drawn from the Alpha101 library [4]. The learning objective is to estimate a function g such that

$$g(\mathbf{X}_{i,t}) = \hat{y}_{i,t}$$

, where the output $\hat{y}_{i,t}$ is the predicted value for the standardized 5-day forward return $y_{i,t}$. We train g by minimizing the mean squared error between $y_{i,t}$ and $\hat{y}_{i,t}$ on the training set.

Given the sequence input $\mathbf{X}_{i,t} \in \mathbb{R}^{L\times K}$ constructed previously, we use a small Transformer encoder model to map each 20-day history of 36 alpha-factor values to a scalar prediction $\hat{y}_{i,t}$. For each stock-date pair (i,t), the model takes $\mathbf{X}_{i,t}$ as input and outputs the prediction $\hat{y}_{i,t}$:

$$g_\theta(\boldsymbol{X}_{i,t}) = \hat{y}_{i,t}$$

, the goal is to capture nonlinear cross-factor interactions that linear models cannot capture.

We train the Transformer end-to-end on the same target $y_{i,t}$ as in the linear baselines, minimizing mean squared error with respect to $y_{i,t}$. The model is optimized with AdamW. The architecture is intentionally kept small to reduce overfitting while still allowing the model to learn nonlinear relationships among the 36 precomputed alpha factors.

In the next stage, we benchmark our approach against two standard linear models: ordinary least squares (OLS) regression and LASSO regression with an L_1 penalty. Both models use the same 20-day factor-history inputs $\mathbf{X}_{i,t}$ and target $y_{i,t}$. This process can be interpreted as a simple linear factor-aggregation scheme.

To apply linear models to the 20×36 alpha factor-history inputs, we first flatten each $\mathbf{X}_{i,t}$ into a one-dimensional vector. Specifically, for each stock-date pair (i,t), we reshape the $L \times K$ matrix into a vector $\mathbf{x}_{i,t} \in \mathbb{R}^{LK}$ by stacking all L days of the K alpha-factor values into one dimension. This transformation preserves all information in the alpha-history window but ignores temporal ordering.

The OLS baseline models the standardized 5-day forward return as a linear function of the flattened features:

$$\hat{y}_{i,t} = \beta_0 + \boldsymbol{\beta}^\top \mathbf{x}_{i,t}$$

where β_0 is an intercept and $\boldsymbol{\beta} \in \mathbb{R}^{LK}$ are regression coefficients. We estimate β_0 and $\boldsymbol{\beta}$ on the training set using the standard *LinearRegression* implementation in scikit-learn.

The LASSO baseline uses the same linear prediction form but adds an L_1 penalty on the coefficients. We use the scikit-learn Lasso estimator with $\lambda = 0.01$ and train it on the same flattened inputs and target as OLS.

Both OLS and LASSO provide simple and interpretable linear baselines that combine the alpha factors with fixed weights. These models can only capture linear relationships in the feature space and fail to capture nonlinear cross-factor interactions.

III. Results

We evaluate the Transformer-based factor aggregator compared with the OLS and LASSO baselines on the same test set. We report three types of metrics: (i) mean square error (MSE) on the standardized 5-day forward return, (ii) 5-day cross-sectional rank information coefficient (RankIC), and (iii) the performance of a long-short portfolio.

Based on the result of the first table, all models achieve test MSE values very close to 1, which reflects that there exists a high noise level in the 5-day returns of a single stock. In other words, neither the linear methods nor the Transformer meaningfully reduces prediction error at the individual-stock level; all three models are weakly informative.

To assess whether the models can rank stocks by future performance, we compute a daily RankIC on the test set: for each date, we calculate the Spearman rank correlation between the model's predicted scores and the real 5-day forward returns across all S&P 500 stocks. From the result, these average

Table I
PREDICTION METRICS ON THE TEST SET

Model	Test MSE	RankIC Mean	RankIC S.D.
OLS	1.0016	0.0024	0.0713
LASSO	0.9956	-0.0003	0.0979
Transformer	0.9957	-0.0013	0.1225

Table II
LONG-SHORT PORTFOLIO PERFORMANCE

Model	Return (%)	Volatility(%)	Sharpe
OLS	-0.80	4.41	-0.18
LASSO	-0.32	5.32	-0.06
Transformer	2.11	6.85	0.31

RankIC values are all very close to zero, indicating that none of the models produces a strong cross-sectional ranking signal.

Because investors ultimately care about portfolio performance rather than raw prediction error, we evaluate the model scores using a simple long-short backtest over the test period. On each rebalancing date, we rank all S&P 500 stocks by the predicted standardized 5-day return, go long the top 20% of the stocks, go short the bottom 20%, and hold this portfolio for the next five trading days. From the resulting daily return series, we compute the annualized return, annualized volatility, and the Sharpe ratio (annualized return divided by annualized volatility, using a zero risk-free rate).

From the second table, we can clearly find that the two linear baselines do not generate a positive annualized return. By contrast, the Transformer-based portfolio achieves a positive annualized return of 2.11%, with a Sharpe ratio of 0.31. Although this Sharpe ratio is modest, the Transformer clearly outperforms the OLS and LASSO baselines on this long-short portfolio metric.

IV. Conclusion

In our experiments, the three models perform similarly on evaluation metrics such as test MSE and average RankIC. However, in the same long-short backtest, the Transformer-based portfolio earns a small positive Sharpe ratio (about 0.31), while OLS- and LASSO-based portfolios have negative Sharpe ratios. Taken together, these results suggest that the Transformer can extract nonlinear structural information from the alpha factors that linear models cannot capture, even though its prediction performance at the individual stock level is very similar to that of the linear baselines.

A natural explanation for the weak performance of all three models is that the alpha factors we chose do not provide useful information. In this project, we focus on 36 pre-defined formulaic alpha factors taken from the public Alpha101 library. Alpha factors are time-sensitive; once they are widely known and used in the financial market, their predictive power can significantly decrease. Therefore, among the 36 factors we selected, many may only provide weak trading signals on the S&P 500 dataset, which means that even more powerful models can only extract limited useful information from the input. From this perspective, test MSE values close to one, and RankIC values close to zero are not evidence that the models are flawed, but rather that the alpha factors set we use does not contain strong and useful trading signals.

It is encouraging that even when these alpha factors can only generate weak signals, the Transformer model is still able to extract additional information and produce positive returns in the performance of long-short portfolios, while the results of the linear OLS and LASSO models are negative. These indicate that the Transformer can extract nonlinear patterns from the existing alpha factors that simple linear weighting schemes cannot capture. In other words, even if a single alpha factor may have weakened and only provided a weak trading signal, the combined alpha generated by the Transformer model can still effectively utilize this information and achieve better performance than the linear benchmark model at the portfolio level.

In future work, we aim to extend our framework to larger and more recent alpha-factor libraries, to experiment with loss functions that are more closely aligned with portfolio-level objectives, and to evaluate several alternative Transformer architectures.

References

[1] T. Chen, W. Chen, and L. Du, "An empirical study of financial factor mining based on gene expression programming," in *2021 4th International Conference on Advanced Electronic Materials, Computers and Software Engineering (AEMCSE)*, 2021, pp. 1113–1117. DOI: 10.1109/AEMCSE51986.2021.00228.

[2] Y. Guo, R. Zhao, and C. Gou, "Mining alpha factors in financial markets using multi-frequency timing residual networks," in *2023 9th International Conference on Computer and Communications (ICCC)*, IEEE, 2023, pp. 2508–2513. DOI: 10 . 1109 / ICCC59590 . 2023 . 10507427.

[3] H. Shi et al., *Alphaforge: A framework to mine and dynamically combine formulaic alpha factors*, 2024. arXiv: 2406.18394 [q-fin.CP]. [Online]. Available: https://arxiv.org/abs/2406.18394.

[4] Z. Kakushadze, "101 formulaic alphas," *Wilmott*, vol. 2016, no. 84, pp. 72–81, 2016. DOI: https://doi.org/10.1002/wilm.10525. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/wilm.10525. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/wilm.10525.

[5] C. Nugent. "S&p 500 stock data," Kaggle, Accessed: Nov. 18, 2025. [Online]. Available: https://www.kaggle.com/datasets/camnugent/sandp500/data.