Alphathon 2024 LLM Problem Submission: Training LLMs for Portfolio Optimization with Parameter-Efficient Online Learning

Ryan Engel, Mario Xerri, Michael Gannon

Abstract

In this submission for the SQA 2024 Alphathon, we chose to work on the LLM problem sponsored by AllianceBernstein. We address this problem by formulating it as a long-term portfolio optimization task with monthly rebalancing. Our approach was to develop an automated portfolio manager trained to predict optimal stock weights based on technical analysis indicators data. We leverage the Chronos-T5 Large Language Model (LLM), originally designed for time-series forecasting, and adapt it for portfolio generation. By exploiting its pre-trained transformer architecture and self-attention mechanisms, we are able to train the model to process diverse technical indicators and make optimized investment decisions. Fine-tuning with Low-Rank Adaptation (LoRA), and following a similar approach to the Parameter-Efficient Reinforcement Learning (PERL) framework, we implement an online learning algorithm to train the model with feedback from its actions each month. We evaluate our strategy in the Quantconnect backtesting platform, where we train the model as if it were a portfolio manager in real time. Over a 10-year backtest, our model demonstrates strong performance relative to the S&P 500 benchmark, generating positive alpha.

1 Introduction

Large Language Models (LLMs) have gathered significant attention for their ability to interpret large amounts of data, and to make meaningful predictions from such data. However, their applications in finance remain an emerging field. Yang et al. [2023], Yu et al. [2023], Nie et al. [2024], Ding et al. [2024], Zhang et al. [2024].

Recent studies show promising results with domain-specific LLM fine-tuning Jeong [2024], Zheng et al. [2024]. However, LLMs are computationally expensive to train and fine-tune, making them impractical in many real-world applications. Much research has gone into solving these problems with Parameter-Efficient Fine-Tuning (PEFT)Mangrulkar et al. [2022] and Low-Rank Adaptation (LoRA) Hu et al. [2021]. Furthermore, these frameworks have been further adapted to reinforcement learning with PERL (Parameter-Efficient Reinforcement Learning) Sidahmed et al. [2024].

To the best of our knowledge, no prior work has applied PERL to train LLMs for financial tasks, such as portfolio optimization. This is because PERL is a relatively new technique, and was originally developed for Reinforcement Learning from Human Feedback (RLHF) in LLM policy alignment tasks.

The main contribution of this work is leveraging the Chronos-T5 LLM Ansari et al. [2024] and training it to optimize a portfolio of stocks based on market feedback rather than human feedback. This approach was inspired by the PERL framework, and we build a similar algorithm for our trading strategy. We evaluate our methodology against the S&P 500's performance over the same time period, and showcase this algorithm's competitive edge against the market.

2 Model Architecture

2.1 Original LLM

The Chronos-T5 LLM is a time-series model pre-trained to handle sequential data. Built on the T5 (Text-to-Text Transfer Transformer) architecture, it includes both an encoder and a decoder. Chronos-T5 was further refined for time-series data, utilizing self-attention mechanisms that excel at capturing long-term dependencies in temporal sequences. This architecture enables Chronos-T5 to focus on the most relevant data points in the input.

2.2 Modified LLM

In our experiments, we utilize the Chronos-T5 Large model (710M parameters). The large parameter space allows Chronos-T5 to capture more complex patterns in the data, improving its ability to generate predictive insights from financial indicators. To adapt this model for portfolio optimization, we added an input projection layer and a final layer, which applies a softmax function to generate portfolio weights. Additionally, we added LoRA adapters to the model, allowing us to fine-tune its parameters with minimal computing power.

3 Training Process

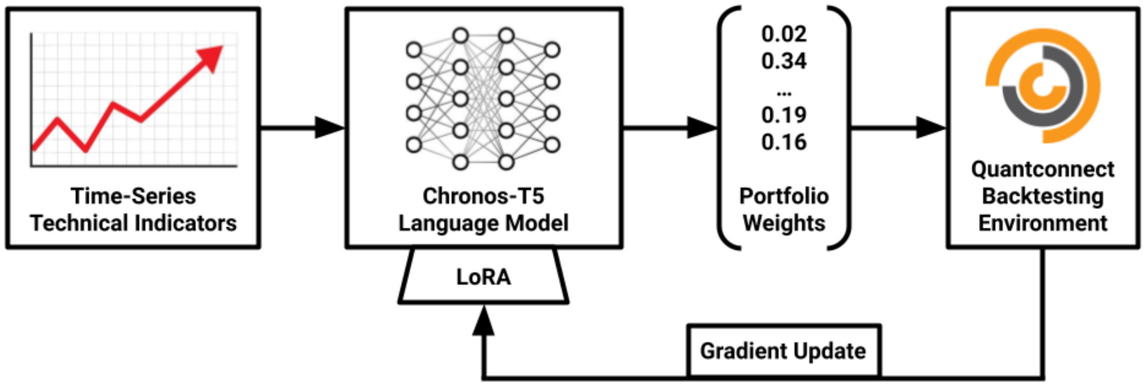

Time-Series Technical Indicators
Chronos-T5 Language Model
LoRA
0.02
0.34
...
0.19
0.16
Portfolio Weights
Quantconnect Backtesting Environment
Gradient Update

Figure 1: Model Training Process

3.1 Algorithm Implementation

We split the algorithm into two sections, where we perform different actions, the beginning of the month and the end of month. In the beginning of the month, we calculate 61 time-series technical indicators, based on each stock's price and volume data. These technical indicators are then used as input to the LLM, which then outputs the portfolio weights for each stock. The model processes this data and outputs portfolio weights for the 100 selected stocks, which are then allocated in the Quantconnect backtesting environment.

At the end of the month, we assess the performance of the allocated portfolio, and calculate the cumulative returns of each stock. These returns are used to calculate the loss function, which measures the model's ability to maximize portfolio returns. The model is updated using Low-Rank Adaptation (LoRA), a parameter-efficient fine-tuning method that adjusts only a small subset of the model's parameters. The loss function is minimized using the AdamW optimizer, which is applied to backpropagate the gradients and update the model's parameters accordingly. This cycle of rebalancing the portfolio and updating the model's parameters based on its performance is repeated monthly, allowing the model to learn and improve its portfolio allocation decisions over time.

3.2 Training with Low-Rank Adaptation

Low-Rank Adaptation (LoRA) is a parameter-efficient fine-tuning method that updates only a small portion of a model's parameters, enabling effective training on limited hardware. This approach retains the general knowledge from pre-training while adapting to domain-specific tasks. By optimizing parameters with less computing power, LoRA allows us to efficiently train LLMs on large historical datasets, making it a practical solution for financial applications without requiring expensive hardware resources.

A key limitation of this research was the restricted GPU resources available through QuantConnect. Training an LLM such as the 710M parameter Chronos-T5 with only a single 32GB GPU is suboptimal, leading us to emphasize parameter efficiency through LoRA. While LoRA mitigated some of these resource limitations, it was not a complete solution. We were constrained to using 16-bit floating point precision (FP16) instead of the more accurate 32-bit (FP32) format, which likely limited the model's performance.

Despite these constraints, LoRA provides a feasible method for fine-tuning LLMs in real-world financial applications. By optimizing parameters with less computing power, LoRA allows us to efficiently train LLMs on large historical datasets, making it a practical solution for financial applications without requiring expensive hardware resources.

3.3 Online Learning and Portfolio Optimization

In our approach, we employ an online learning approach, where the objective is to directly maximize the portfolio's expected cumulative return each month. The model learns from the outcomes of its portfolio allocations and adjusts its parameters to improve future decisions. This allows the model to adapt to market changes as they happen, rather than relying solely on historical time-series forecasting.

Rather than using a traditional reinforcement learning algorithm, which would involve learning a policy and exploring different actions stochastically, we adopt a deterministic optimization approach to reduce the computational complexity of our strategy. In this setup, the portfolio allocation is computed directly from the modified Chronos-T5 LLM. The model takes as input the technical indicators computed over the past month and outputs portfolio weights for the selected stocks.

The loss function is defined to maximize the expected return of the portfolio with an additional entropy regularization term to encourage diversification in the portfolio:

$$\mathcal{L}(w, r) = -w^T r + \eta \cdot (-w^T \cdot \log(w + \epsilon))$$

where w^T represents the generated portfolio weights, r denotes the returns of each stock over the month, η is a regularization strength parameter, and ϵ is a small constant added for numerical stability. The first term in the loss function maximizes the expected portfolio return, and encourages the model to exploit the strategy it learns. The second term penalizes concentrated allocations, promoting diversity in stock selection. We use the second term for entropy regularization, which helps encourage exploration.

4 Technical Analysis Indicators Data

The technical analysis indicators for this project were calculated using the TA-Lib python library, focusing on various indicators including Cycle Indicators, Momentum Indicators, Volatility Indicators, and Volume Indicators. We deliberately excluded non-time-series indicators that produce output not aligned with our time-series modeling approach.

To enhance model interpretability and performance, we applied a Variance Inflation Factor (VIF) filter to assess and eliminate multicollinearity among the indicators. Any indicators with high VIF values were removed to prevent the model from being influenced by redundant or highly correlated features, ensuring that each selected indicator contributed unique and valuable information to the model's decision-making process. After filtering, we retained a total of 41 distinct technical indicators.

Extensive data cleaning was performed to ensure the integrity of the dataset. This included removing constant or zero-variance columns, handling missing values, and verifying data consistency. The resulting dataset provided a high-quality input for the model, allowing us to simulate a "technical indicators trader" capable of making optimized portfolio decisions based on real-time market trends.

5 Experimental Setup

All experiments were conducted using the QuantConnect backtesting engine, utilizing its online cloud platform and GPU nodes. Model training was conducted on a single Tesla V100S-PCIE-32GB GPU provided by QuantConnect. The backtest period ran from 6/1/2009 to 12/31/2019, aligning with the start of the ETF constituents data and the end of the Alphathon backtesting period.

The stock selection process for our strategy is based closely on that of the S&P 500 universe, using the ETF constituents dataset in QuantConnect. Every three months, we selected the top 100 weighted stocks from the 'SPY' index, adding new stocks and removing delisted ones from the portfolio. Every month, the algorithm explained in section 3.1 is conducted to rebalance the portfolio, and update the model. At the conclusion of the backtest, QuantConnect's metrics were analyzed.

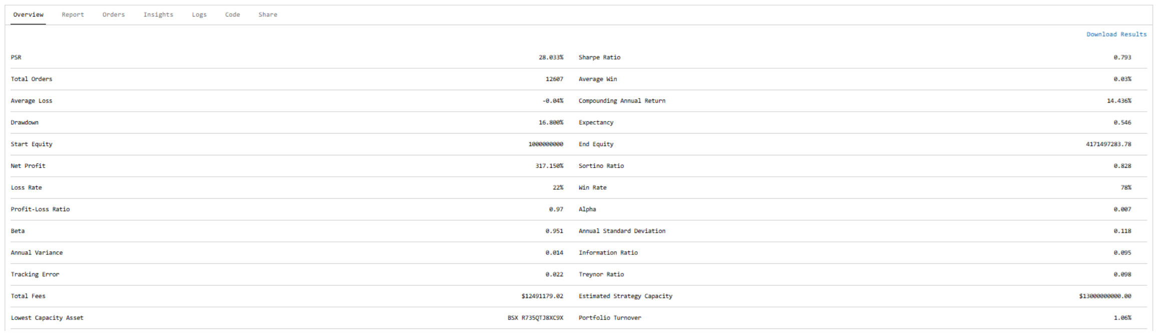
Key hyperparameters, such as learning rate=0.01, eta=0.01, and LoRA parameters, were experimentally derived. Another notable parameter is the model's initial weights, determined by random seed values. These hyperparameters impact performance of the model, making them important factors to consider. We tested many combinations of hyperparameters and ultimately evaluated the setup with the most robust metrics, shown in the results section.

6 Results

The results of our experiments show that our algorithm can consistently outperform the S&P 500, and that this is an efficient and effective strategy for alpha generation. Based on the results of a QuantConnect backtest, which is shown in Figure 2, it is clear that our strategy generates high alpha. The table displays an alpha of 0.7%, while the beta of this strategy is kept under 1 at 0.951. The annualized volatility (11.8%) was calculated to be less than the average annualized volatility of the underlying benchmark of 18%.

This framework also enhances an investor's Sharpe Ratio (0.793 for the in-sample backtest results displayed), compared to the S&P 500, whose 10-year average is 0.71. The Sortino ratio of 0.828 was lower than the S&P 500 benchmark's Sortino ratio of 0.89, indicating that the benchmark outperformed our portfolio in terms of risk-adjusted returns focused on downside volatility. The annualized return of the S&P 500 over the backtest window came in at 12.31%, while the annualized return of the portfolio utilizing our LLM strategy was calculated to be 14.44%. This represents a substantial 213 bps improvement, not adjusted for risk. Finance [2024]

Our model demonstrated the ability to capture upward and downward market trends effectively, leveraging time-series technical indicators to identify which stocks are likely to outperform. The highest alpha was generated during upward market trends, where the model accurately predicted which stocks would deliver the most value. This is shown in Figure 3.



PSR	28.033%	Sharpe Ratio	0.793
Total Orders	12607	Average Win	0.03%
Average Loss	-0.04%	Compounding Annual Return	14.436%
Drawdown	16.800%	Expectancy	0.546
Start Equity	1000000000	End Equity	4171497283.78
Net Profit	317.150%	Sortino Ratio	0.828
Loss Rate	22%	Win Rate	78%
Profit-Loss Ratio	0.97	Alpha	0.007
Beta	0.951	Annual Standard Deviation	0.118
Annual Variance	0.014	Information Ratio	0.095
Tracking Error	0.022	Treynor Ratio	0.098
Total Fees	$12491179.02	Estimated Strategy Capacity	$13000000000.00
Lowest Capacity Asset	BSX R735QTJ8XC9X	Portfolio Turnover	1.06%

Figure 2: Performance Metrics Screenshot

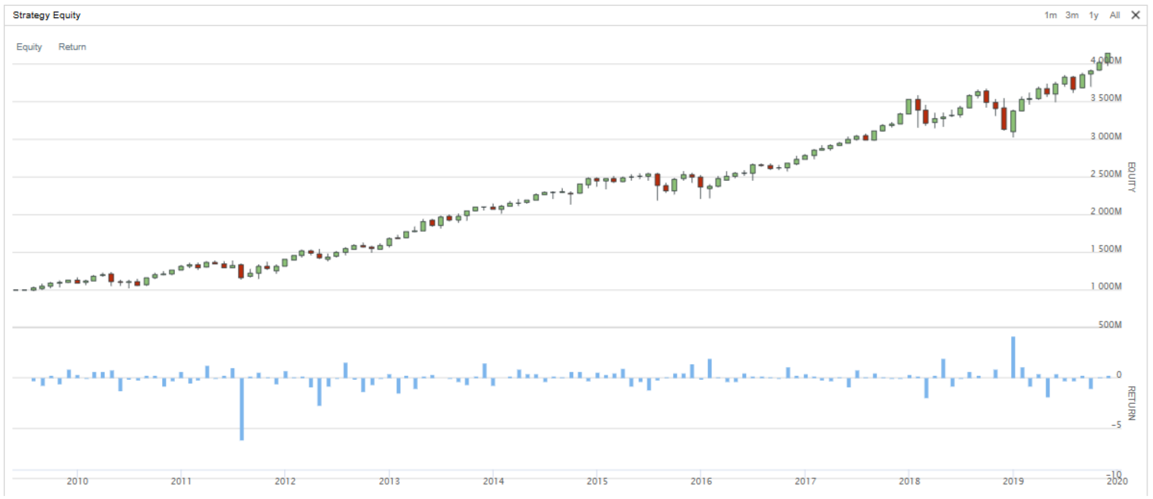


Figure 3: Performance Metrics Screenshot

Overall, our results show that training the LLM for portfolio optimization based on technical analysis indicators proved to be a powerful tool for generating alpha. Our model consistently outperforms the S&P 500 and also demonstrated the potential of LLMs in driving sophisticated, data-driven financial strategies, offering a significant advantage over traditional benchmarks.

7 Conclusion

In this submission for the Alphathon 2024 competition, we introduce an innovative approach to portfolio optimization by fine-tuning the Chronos-T5 LLM on time-series financial data. By utilizing Low-Rank Adaptation (LoRA) and concepts from the Parameter-Efficient Reinforcement Learning (PERL) framework, we trained the model to optimize stock portfolio weights based on technical analysis indicators and feedback based on its performance.

Rather than using LLMs on news data or for sentiment analysis, our approach leverages Chronos-T5's deep time-series understanding and self-attention mechanisms to capture market trends and make precise investment decisions. Backtests on QuantConnect consistently outperformed the S&P 500, demonstrating the capabilities of this approach. Despite resource constraints, this method proved both efficient and cost-effective, illustrating the practicality of our approach, and offering a promising direction for future research in quantitative finance.

References

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C. Maddix, Hao Wang, Michael W. Mahoney, Kari Torkkola, Andrew Gordon Wilson, Michael Bohlke-Schneider, and Yuyang Wang. Chronos: Learning the language of time series, 2024. URL `https://arxiv.org/abs/2403.07815`.

Han Ding, Yinheng Li, Junhao Wang, and Hang Chen. Large language model agent in financial trading: A survey, 2024. URL `https://arxiv.org/abs/2408.06361`.

Yahoo Finance. S&p 500 index, 2024. URL `https://finance.yahoo.com/quote/%5EGSPC/`. Accessed: 2024-09-30.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. URL `https://arxiv.org/abs/2106.09685`.

Cheonsu Jeong. Domain-specialized llm: Financial fine-tuning and utilization method using mistral 7b. *Journal of Intelligence and Information Systems*, 30(1):93–120, March 2024. ISSN 2288-4882. doi: 10.13088/jiis.2024.30.1.093. URL `http://dx.doi.org/10.13088/jiis.2024.30.1.093`.

Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and B Bossan. Peft: State-of-the-art parameter-efficient fine-tuning methods. *Younes Belkada and Sayak Paul," PEFT: State-of-the-art Parameter-Efficient Fine-Tuning methods*, 2022. URL `https://github.com/huggingface/peft`.

Yuqi Nie, Yaxuan Kong, Xiaowen Dong, John M. Mulvey, H. Vincent Poor, Qingsong Wen, and Stefan Zohren. A survey of large language models for financial applications: Progress, prospects and challenges, 2024. URL `https://arxiv.org/abs/2406.11903`.

Hakim Sidahmed, Samrat Phatale, Alex Hutcheson, Zhuonan Lin, Zhang Chen, Zac Yu, Jarvis Jin, Roman Komarytsia, Christiane Ahlheim, Yonghao Zhu, Simral Chaudhary, Bowen Li, Saravanan Ganesh, Bill Byrne, Jessica Hoffmann, Hassan Mansoor, Wei Li, Abhinav Rastogi, and Lucas Dixon. Perl: Parameter efficient reinforcement learning from human feedback, 2024. URL `https://arxiv.org/abs/2403.10704`.

Hongyang Yang, Xiao-Yang Liu, and Christina Dan Wang. Fingpt: Open-source financial large language models, 2023. URL `https://arxiv.org/abs/2306.06031`.

Yangyang Yu, Haohang Li, Zhi Chen, Yuechen Jiang, Yang Li, Denghui Zhang, Rong Liu, Jordan W. Suchow, and Khaldoun Khashanah. Finmem: A performance-enhanced llm trading agent with layered memory and character design, 2023. URL `https://arxiv.org/abs/2311.13743`.

Wentao Zhang, Lingxuan Zhao, Haochong Xia, Shuo Sun, Jiaze Sun, Molei Qin, Xinyi Li, Yuqing Zhao, Yilei Zhao, Xinyu Cai, Longtao Zheng, Xinrun Wang, and Bo An. A multimodal foundation agent for financial trading: Tool-augmented, diversified, and generalist, 2024. URL `https://arxiv.org/abs/2402.18485`.

Jiawei Zheng, Hanghai Hong, Xiaoli Wang, Jingsong Su, Yonggui Liang, and Shikai Wu. Fine-tuning large language models for domain-specific machine translation, 2024. URL `https://arxiv.org/abs/2402.15061`.