Leveraging LLMs to Automate Options Trading

NATHAN CLARK

Contents

1 INTRODUCTION

Large language models have demonstrated increasing capabilities to match or exceed human performance at various tasks, from natural language understanding and generation to complex decision-making tasks. [Naveed et al., 2024] Pretrained foundation models are often few-shot learners and can be utilized to achieve state-of-the-art performance with far fewer training samples than in traditional supervised learning paradigms, [Brown et al., 2020] and have even demonstrated state-of-the-art performance in numerical time-series predictions. [Gruver et al., 2023]

This paper proposes a purely neural method of integrating news data with real-time trading systems, extracting meaningful information from news using LLMs and translating price predictions into increased expected value using a novel method of encoding model outputs over put/call options. In particular, we propose the following advancements:

- Leveraging LLMs for time series predictions using serialization that leads to more consistent tokenization and hence performance.
- Augmenting our LLM's predictions by retrieving relevant news data and summarize it to the time series model.
- Converting predictions into trading profits using options to align model training and evaluation with actual market performance.

This approach shows promising results for achieving accurate stock price predictions and market gains, demonstrating the clear viability of LLMs to augment existing automated trading platforms and related time-series problems.

2 METHODOLOGY

2.1 Datasets

The following were used as data sources:

(1) The `yfinance` package was used to obtain historical stock prices. [Aroussi, 2024]
(2) The GNews API was used to query for live news feeds. [GNewsAPI, 2024]
(3) The RealNews dataset, itself derived from the common crawl dataset, was used as a source of historical news. [Zellers, 2019]
(4) The Interactive Brokers API was used both as a live feed of stock and option prices as well as a platform on which to simulate model performance.

Collectively these have substantial use and redistribution restrictions. It is important to note that this paper and included software is for educational and academic purposes only.

The raw stock prices from `yfinance` were loaded and saved into compressed numpy arrays as `npz` files. Historical news files were converted to a tarfile and saved on a separate drive due to the considerable size (123 GB) of the RealNews dataset. Both live and historical news are then converted to a common "News" object representing the headline, text, and date of a given article in a consistent format regardless of source.

2.2 Preprocessing & Tokenization

In 2023, [Gruver et al., 2023] demonstrated the performance of large language models at zero-shot time-series forecasting with state-of-the-art results. However, natural language tokenizations are often ineffective for decimal integers. For example, suppose $n = 123$. Then, our tokenization may split the number into tokens representing 12 and 3; we find similar situations for other lengthy numbers where the positional encoding of a token and length of a number in tokens does not allow a model to directly infer positional value. Hence, we instead tokenize a given decimal representation

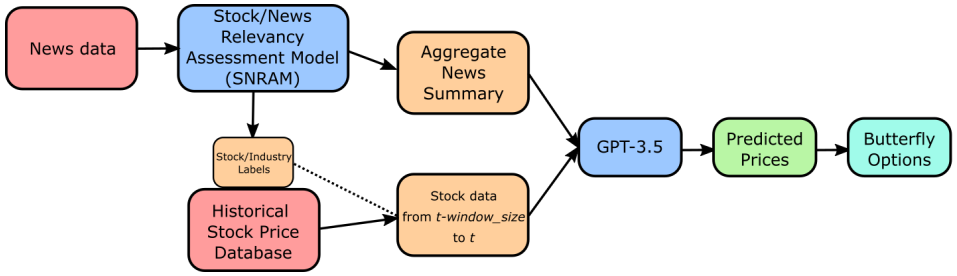

Fig. 1. The overall structure of the model

of a number $n = \sum 10^i a_i$ as $[\text{tokenize}(a_i), \text{tokenize}(a_{i-1}), \ldots \text{tokenize}(a_0)]$ for consistency. We then define an arbitrary separator token to separate multiple numbers in a series.

However, we run into yet another problem: vastly differing scales exist for different stocks, with some ranging near to the thousands (e.g. Regeneron) and others staying in the dozens (e.g. Ford). To solve this problem, we dynamically rescale our input to max out at $(1 + \beta)$ times the 95% percentile value, using this instead of the maximum to avoid outliers. A cursory hyperparameter grid search finds $\beta = 0.3$ gave the best performance. We also ignore exact floating point precision in favor of a lower precision, based on the clear considerable reduced value of predictions more granular than this and the restricted resolution of option strikes being a limiting factor on the utility of higher precision predictions, which we will explore later.

In the text embedding of stock tickers we use whenever providing a stock to a large language model, we always provide it as the full company name, trimmed of any generic terms that do not disambiguate the company's identity, removing terms such as "Ltd." and "Inc."

2.3 Architecture

The amount of data and investment strategies we wish to make available to our model represents an ineffably large space of potential inputs and outputs unfit for any single model to take as input/output. Instead, we relegate the tasks of gradually reducing our input news and stock price data to specialized submodules, and reduce the output to price predictions which we use classical financial tools to convert into options.

2.3.1 SNRAM. News articles in our historical news database do not come with labels as to their relevance to a particular stock. Keyword approaches are viable to obtaining explicit mentions to stocks – however, this is hardly the full picture of potential impacts of a given event. Instead, our model leverages natural language processing to better encode all potential related stocks, developing a stock/news relevancy assessment model (hereafter "SNRAM").

SNRAM uses GPT-3.5-Turbo that has been provided system and user prompts detailing all existing stocks available in the stock price database, and is asked to provide only the name of the relevant company. This is then converted directly back into the stock ticker using a dictionary mapping stock names back to tickers. While outputs are technically somewhat subjective in what stocks could potentially be considered most relevant to a news article, we find that hallucinations and improperly formatted outputs are exceedingly rare. In particular, in a test of the live script that involved 2000 calls[1] to SNRAM, only 2 failed due to misformatted SNRAM outputs.

[1]SNRAM was called 20 times per timestep over 100 steps, for 20×100 calls

We then apply a summarization model over several news articles with the same relevant stocks, in order to leverage and compress the available potentially relevant news into an agggregate news summary to make best use of the model's context window.

2.3.2 GPT-3.5. We use GPT-3.5 over a historical stock data up until the simulated prediction time (or the current time if the live system) over each hour of a particular stock's price, using the tokenization method mentioned above in Preprocessing & Tokenization. We further restrict the outputs using logit bias to only the digits and numerical separator. Considering the stochastic nature of large language models run with nonzero temperatures, we run this model over n parallel times representing potential timelines based on the model's predictions. Considering the recursive nature of how a transformer generates multiple tokens, i.e. that the outputed probabilities over tokens is based on the tokens the transformer already produced that are then re-fed as the new inputs, it is reasonable to intuitively consider these as alternate timelines of the stock price evolution. If we assume all are equal in probability, increasing n leads to an increasingly detailed probability mass function over time describing the stock price's potential evolution.

We briefly note here that GPT-3.5 has been shown to be a more effective model than those that have undergone alignment (i.e. RLHF) at time-series predictions, informing the use of this model as opposed to other options. [Gruver et al., 2023]

Using retreival-augmented generation, the aggregate news summary from SNRAM is used as context for the GPT-3.5 time-series model. This is included as part of the prompt before the historical stock data, enabling the model to make use of real-world events and new information.

2.3.3 Model uncertainty. Not all model outputs are equal in importance. In order to filter price predictions that are less likely to be accurate, we propose the following parameters which we calculate for each proposed prediction, which consists of a 2D matrix where each row is a parallel time-series predictions of the stock price at each hourly timestep into the future. First, we take the median value along each column to obtain the median price evolution. We then compute the variance of the samples along this prediction, supposing that this will with more timesteps approximate the variance of the underlying Brownian noise defining the stock price's change over time. We then use the variance along each column as a proxy to the model's uncertainty: greater variance at each timestep indicates a greater potential range of values according to the model. The effect of filtering on each metrics was evaluated and used to adjust their weightings; full details of evaluation are under Conclusions.

2.4 Options

Thus far, this paper has only presented an architecture for obtaining price predictions. But predictions do not *per se* generate trading revenue. In order to convert our probabilistic predictions of what a stock price will look like at a given time, one can buy and sell options with payoff and expiry date providing optimal payoffs if predictions are correct. As an additional benefit, options give a substantial leveraging effect as the responsiveness δ of a given option with respect to its underlying stock price can be quite large in relation to the ratio of the premium (i.e. cost) of the option and the price of the underlying stock.

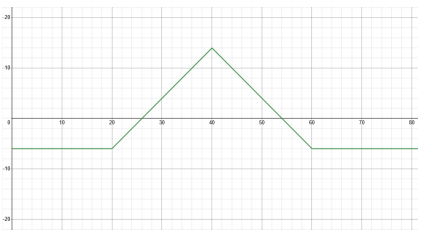

Fig. 2. The payoff of a butterfly spread with $k = 40$, $s = 20$, $p_{net} = 6$

2.4.1 Butterfly spreads. In order to ensure losses are bounded (i.e. the maximum cost is not infinite) while maximizing gains at a given predicted price p at time t, we choose butterfly spreads with central strike $k = p$ and expiry date t corresponding to our price predictions. This strategy consists of 3 call contracts at strikes $k - s$, k, and $k + s$, resulting in a net payoff of:

$$q(\max(k - x - s, 0) + \max(k - x + s, 0) - 2\max(k - x, 0) - p_{net})$$

where q is the quantity purchased, s is a hyperparameter defining how wide or narrow the spread is, and p_{net} is the sum of the premiums of the options. It is important to note here that option strikes are usually quantized to the nearest \$5, so price predictions are rounded to the nearest \$5. We choose $s = \$20$ and dynamically calculate the quantity q to buy or sell to match our desired overall portfolio amount.

2.5 Evaluation

One might consider using a model's accuracy directly over the predictions on the data above; however, this will not necessarily reflect the actual power of our model in maximizing profits. While we would like to fully simulate our model on full markets, this is fundamentally impossible without actually trading money due to both the theoretical effect of depth of market in how trading affects prices, and in substantial pragmatic limitations on what data is freely available. Hence, we must develop effective proxies to market performance and simulate performance to the greatest extent reasonable. We implement the following methods:

- Simulate a model on market data and estimate the value of the purchased butterfly spreads as their value at expiration based on historical stock data
- Use an IBKR paper trading account to simulate gain on actual market data

We mitigate data leakage by validating on news data separate than what was used for hyperparameter optimization, using only market data after GPT-3.5-Turbo's cutoff date of September 2021 [OpenAI, 2024], and ensuring the date of all news articles are always before the timesteps the model is asked to predict.

We make the following assumptions about each approach. Since the investment strategy involves options, and detailed option pricing data is prohibitively expensive, we use the fact that options are generally never optimal to exercise to compute payoffs at expiry. We also assume the effects of market depth are negligible (see appendix for details).

3 CONCLUSIONS

Model performance according to both methods of evaluation were promising and show significant latent potential for application in advancing the state of the art in time series predictions. Of note, the model's performance was often the optimal option strategy of all outputs within the options embedding; that is to say, the model's predictions were so accurate they were limited by the quantization necessary to convert raw predictions to options. In particular, on the 24 hour to expiry task of predicting prices a stock's price a day into the future, even without removing outliers or implementing any of the strategies used to remove uncertain and high-variance model outputs, the model's price prediction was within \$5 of the actual price 76% of the time.

In particular, on a test of $n = 100$ running the model directly on aggregate news summaries with a knowledge cutoff of before January 1st, 2024 and hourly stock history from before January 1st, 2024, the model's predictions over various stocks have an absolute difference of less than \$5 from the actual value at later dates 76 out of 100 attempts. Potential data leakage was mitigated by selecting only models with knowledge cutoffs before 2024, as GPT-3.5 has a knowledge cutoff of September 2021 [OpenAI, 2024], and by skipping any news samples that were ever used in

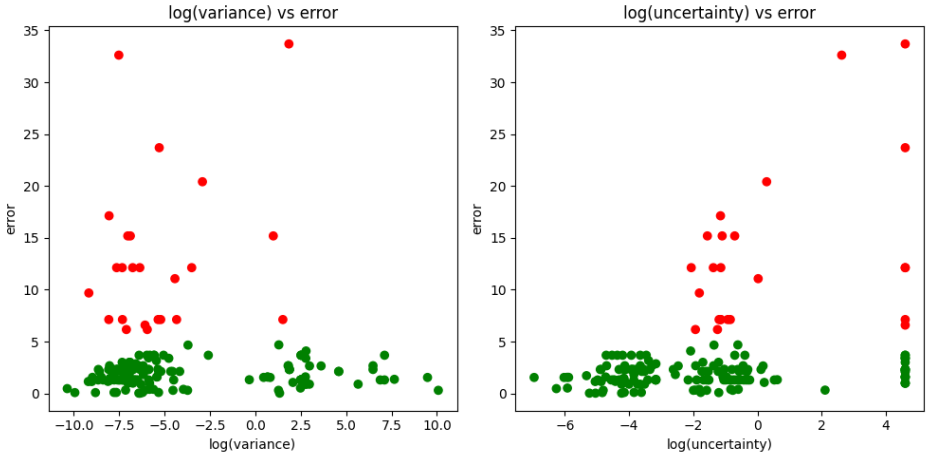

Fig. 3. Graphs of absolute error vs model uncertainty and model variance

previous hyperparamter tuning. To reiterate, this means the model's output was the optimal output producing the maximum profit possible by a butterfly spread with our range of strikes 76% of the time.

While net profit is harder to fully evaluate on historical data with any precision given the lack of freely available historical options prices, the advantage of the butterfly spread includes that maximum losses are bounded by the net premium, hence even particularly unreasonable model outputs have a limited potential to eliminate existing gains. That being said, implementing our strategy for pruning high-variance and uncertain model outputs, we find that 10 out of the 11 trades that would meet criteria for execution are optimal, with absolute differences in predicted vs actual prices of: 0.105, 2.13, 0.475, 1.31, 0.09, 1.56, 1.83, 0.90, 2.68, 2.32, 11.07, in USD.[2] We note that the only difference suggesting suboptimal option strategy of a difference of 11.07 is still within our selected parameter of butterfly strategy spread of $20, i.e. the difference between option strikes, and so failing highly unusual market conditions exactly when that trade would be executed such a trade would still be profitable.

This evaluation supports the consistent and incredible performance on the IBKR paper trading platform, with the live system achieving a final liquidation value of 1,380,594 USD. However, there are important considerations limiting the actual performance of this model in practice, as discussed in the Limitations section of the appendix. The highly leveraged nature of the options strategy used does create significant risk and increases the potential for loss just as it increases the potential for profit. Protecting and hedging trades is necessary; and, the market is in a constant state of innovation towards increasingly sophisticated methods for trading, so the actual expected performance of this model over a longer time period would likely be considerably less.

The correlation coefficients between log variance and absolute error is -0.04, and between log uncertainty and absolute error is 0.31. Initially, we might note the correlation between model uncertainty and performance is somewhat weak; however, note that most samples (green in the figure above) have errors that are solely the result of the quantization in scaling, i.e. they are the best outputs the model could possibly produce and result in the best options strategy possible

[2]Differences of half a cent, i.e. $0.005, are possible due to midpoint pricing, e.g. for bid=$1.00 and ask=$1.01

with available strikes. The positive correlation between uncertainty and error suggests our use of uncertainty as a means of deciding which model predictions to use for options trading is reasonable.

The lack of correlation between variance and error is unexpected, as options pricing theory usually dictates that increasing variance lowers the expected payoff of butterfly spreads. This suggests the parallel timelines that our time-series model computes may not be easily approximate a stock price probability mass function evolution over time, i.e. the model's many parallel outputs as to potential stock price values are not useful considered as samples to some probability distribution. This observation is what justified implementing only the simple selection method of only the median value; however, it is possible with higher sample sizes we begin to observe an approximate probability distribution from the time-series model's parallel outputs.

It is interesting to observe that all errors greater in magnitude than \$5 were always in the negative direction, likely because our scaling strategy which allows the model to freely output down to \$0 but restricts the model's output to $\beta = 30$ % above the current historical max. While initial hyperparameter tuning suggests significantly higher β has lower performance, further tuning of the scaling parameters may mitigate this effect. A more sophisticated approach, such as assymetrical butterfly spreads offering more free parameters for the model to approximate a stock price probability distribution, may also mitigate this limitation. Future work to fully realize the time-series prediction potential of LLMs includes finetuning, training on more comprehensive historical options data, and further prompt engineering.

A APPENDIX

A.1 Simulation artefacts

There exist several important differences between the dynamics of a real market and the simulation through IKBR. The expected effects of these various simulation-specific artifacts both enable unrealistic (i.e. not realizable in real-world markets) gains and hinder performance, but ultimately altogether cast some restrictions on what one can infer from the results. In particular:

- IKBR presents market data at a 15 minute delay
- Depth of market is not simulated

The delayed market data clearly makes any high-frequency trading strategies infeasible if we used IKBR as a data source, but more interestingly, it also causes subtle issues with limit orders. A limit order places an upper limit on the price actually paid for a stock, lessening the uncertainty of actual execution inherent in market orders. However, when the algorithm was written to allow this limit order to be handled automatically, occasionally the limit price would not reflect recent market changes and cause delayed orders that are inherently counter to existing market trends. This can be mitigated by using market orders or manually setting higher limit prices, but essentially means that trading only from IKBR data is trading blind to recent market movements.

Depth of market, or order book depth, refers to the fact that ask/bid prices exist on a spectrum with different traders willing to purchase or sell at different prices. Making any purchase or sale influences later prices; while for individual investors this effect may usually be trivially small, this effect is significant enough to motivate dark pools consisting of 40% of trading activity in 2019. Hence for a sufficiently large institutional investor, the strategies presented in this paper may have different empirical results.

However, even casting this consideration aside, finite market liquidity means that market prices will change as the purchase or sale is made. For example, for a financial instrument in which 1 unit is for sale at \$1 and an additional 5 units are for sale at \$2, while our simulated trading via IBKR will purchase 6 units for \$6, a real trader would pay \$1 + 5 * \$2 = \$11.

Depth of market fundamentally is impossible to simulate perfectly without actual execution of trades, so these same considerations exist regardless of the simulation method used. It is hence similarly impossible to quantify the effect of depth of market on this strategy's performance; however, it is likely this effect is more pronounced on especially low-volume markets where some of the largest profits were realized.

A.2 Dataset limitations

High-quality data is often quite expensive, and in the world of quantitative finance, this is especially true. Several cost or licensing restrictions prevent the inclusion of crucial tests that would otherwise allow for more thorough backtesting, analytics, an improved model. In particular, historical options pricing data is especially pricey, with the full ORATS historical dataset priced at $2000 as of the writing of this paper. This and similar datasets would allow for more training directly on options pricings over time, as well as offer a more thorough means of testing the model's performance.

REFERENCES

Ran Aroussi. 2024. *yfinance.* https://pypi.org/project/yfinance/

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. arXiv:2005.14165 [cs.CL]

GNewsAPI. 2024. *GNews API.* https://gnews.io/

Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew Gordon Wilson. 2023. Large Language Models Are Zero-Shot Time Series Forecasters. arXiv:2310.07820 [cs.LG]

Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. 2024. A Comprehensive Overview of Large Language Models. arXiv:2307.06435 [cs.CL]

OpenAI. 2024. *GPT-3.5 Turbo Updates.* https://help.openai.com/en/articles/8555514-gpt-3-5-turbo-updates

Rowan Zellers. 2019. *RealNews Dataset.* https://github.com/rowanz/grover/tree/master/realnews