Extended Abstract

Motivation Narratives like earnings calls, Federal Reserve announcements, and economic reports impact markets strongly but most trading algorithms have no access to these real-time verbal communications humans traders use to make trading decisions. This provides a unique opportunity for reinforcement learning (RL) agents to react to financial language before it is acted upon. We seek to address this limitation by creating a transformer-based RL agent that ingests economic transcripts line by line in conjunction with high-frequency market activity to make sequential trading decisions. Our RL agent empowers agents of reaction through increasingly changing narratives and sentiment, and represents a scalable solution for parsing the complex increase in financial discourse beyond human access.

Method To address this gap, we trained an RL agent to make sequential trade decisions (Buy, Hold, or Sell) based on real-time financial chat and high-frequency stock data. We employed pre-trained transformers such as BERT, FinBERT, Longformer, or LongT5 to embed every sentence. We concatenated them with second-by-second stock prices, recent trading activity, and a rolling window of discussion context to construct the state representation for the agent. We compared three policy architectures: a baseline multilayer perceptron (MLP), a gated recurrent unit (GRU) network, and a transformer encoder with causal attention. These were each combined with a value network for advantage estimation. The agent was trained with the Proximal Policy Optimization (PPO) algorithm with discount factor = 0.99, GAE lambda = 0.95, learning rate 3×10^{-4}, batch size = 64, and 10 epochs per policy update. Both the policy and value networks were two-layer MLPs with 64 hidden units and ReLU activations, unless the policy architecture was recurrent or transformer-based explicitly. As a baseline, we employed a static FinBERT sentiment classifier, translating its sentiment outputs into discrete trade actions for apples-to-apples comparison with the RL agent.

Implementation Our dataset consisted of the earnings calls of twenty S&P 500 technology companies from the 4 most recent quarters. We used OpenAI Whisper to transcribe the audio files and generate timestamps for each sentence. This allowed us to align the transcripts with second-by-second market data. The stock data was sourced using the Databento API. Each training episode consisted of 1 full earnings call. The RL agent was placed in a custom trading environment that simulated live discourse and the real-time market response. We evaluated four transformer architectures, BERT, FinBERT, Longformer, and LongT5, for embedding quality and the downstream policy performance.

Results We evaluated RL agents using different transformer embeddings—BERT, FinBERT, Longformer, and LongT5—on 80 S&P 500 earnings calls. The static FinBERT sentiment classifier outperformed all RL models with an average return of 0%, an average Sharpe ratio of 0.58, and average drawdown (-0.62%). Among RL agents, FinBERT-based embeddings yielded the most stable results: -0.01% average return and -0.54% drawdown. BERT, Longformer, and LongT5 demonstrated higher volatility, with drawdowns as low as -3.24%. However, Longformer and LongT5 achieved the highest max returns, reaching 4.62% and 4.6%, respectively. GRU policies introduced greater risk but showed potential with episodic gains, while the transformer policy with causal attention failed to train effectively. These results reveal a tradeoff: transformer-based RL agents can detect rare, profitable events, but at the cost of stability and consistent performance.

Discussion and Conclusion Our results demonstrate that although transformer-based RL agents such as Longformer, LongT5, or otherwise, have potential to realize high return trading opportunities using financial text occasionally, the agents experience instability with large drawdowns and non-consistent outcomes. In comparison, static FinBERT sentiment classifier performed best with respect to average returns and risk every time. At the same time, GRU-based policies learned to engage temporal dependencies for event-related profit generation, but they were unstable. The MLPs resulted in stable but less responsive behavior. The transformer-based policies with causal attention failed to train, which suggests that transformer-based RL agents have challenges learning within sparse, noisy ecological settings. There is a need for improved temporal modeling, more stable training approaches and further developed reward functions. Our research serves as a first step toward developing a real-time language-aware trading system that moves beyond traditional price action methods.

Sequential Reinforcement Learning on Economic Discourse for Real-Time

Sarang Goel
Department of Computer Science
Stanford University
sarang28@stanford.edu

Chirag Maheshwari
Department of Computer Science
Stanford University
cmaheshwari@stanford.edu

Ekansh Mittal
Department of Computer Science
Stanford University
ekanshm@stanford.edu

Abstract

We devise a new Reinforcement Learning (RL) agent that acts in real time to process financial narratives like earning calls, one sentence at a time together with live high-frequency market data. In contrast to past work that simplifies discourse using static sentiment labels, our agent uses family of transformer embeddings (BERT, FinBERT, Longformer, LongT5) to process raw textual transcripts and then decides to buy/hold/sell in a custom RL environment. We compare different RL policies (MLP, GRU, transformer encoders) for our RL model, using static FinBERT sentiment classifier as a baseline. While the static baseline outperforms all of the RL agents regarding average return and risk traits, all four transformer-based RL agents occasionally do capture high returns. However, this higher return comes with higher volatility and instability. In general, our findings indicate promise but important limitations for discourse-aware trading systems, particularly in effectively modeling time and reward function stability. This work is a step towards the goal of enabling autonomous agents to consume economic narratives and act in real-time, indicating exciting possibilities in financial AI beyond traditional numerical strategies.

1 Introduction

Financial markets have become more reliant on real-time narrative information (such as earnings calls, speeches by central banks, announcements, and press conferences) to price assets and judge risk. While traditional algorithmic trading systems are built around structured, time-series data such as historical prices or technical indicators, they have very limited capabilities for unstructured textual data, especially live speech text representing the changing, near-continuous conversations of traders that often foreshadow market movements. Human traders apply a personal way of listening and using their intuition, sentiment, and outside conditions to cue them to rapid trading decisions surrounding live conversations. Accordingly, the absence of economic language interpretation and the inability to make decisions in real time is a major difference that keeps the application of AI systems limited.

Natural Language Processing (NLP) models like FinBERT have demonstrated that financial sentiment can provide short-term predictive signals, but the models are typically static classifiers. They encode text into discrete sentiment scores or trade labels, operating in a supervised learning paradigm with no real-time responsiveness or capacity for adaptation. Reinforcement Learning (RL), by contrast, offers an attractive framework for constructing decision-making agents based on evolving inputs, facilitating the dynamic revision of strategies as new information arrives. Yet most existing RL applications in finance—e.g., FinRL or Deep RL for High-Frequency Trading (HFT)—revolve around structured numerical features and ignore the rich, imprecise, and context-dependent signals embedded in natural language.

In this paper, we introduce a novel reinforcement learning agent that reads financial discussion—sentence by sentence—and learns to trade in real time by synthesizing linguistic cues with high-frequency market data. The agent lies at the intersection of NLP and financial decision-making, accepting as input raw earnings call transcripts paired with second-by-second stock price data to decide whether to Buy, Hold, or Sell after each sentence. Unlike previous work, our system does not seek to reduce language to a simple sentiment score but strives to learn direct mappings from discourse to profitable trading activity.

There are three primary technical contributions of this work. First, we offer a synchronized dataset of earnings calls and second-by-second market characteristics, which allow for precise alignment for speaking narrative to market context. Second, we create a transformer-embedding-based RL environment where the sequential decisions are informed by sentence-level financial language in conjunction with time-series data. Finally, and most critically, we compare a variety of policy network architectures - multilayer perceptrons (MLPs), gated recurrent units (GRUs), transformer encoders with causal attention - on the same FinBERT-based sentence embeddings to assess the importance of temporal modeling and architectural considerations to trading performance. We compare the RL agents to performance from a static FinBERT sentiment classifier, and the variability in outcomes illustrate both the potential and challenges of transformer-based RL agents in financial discourse.

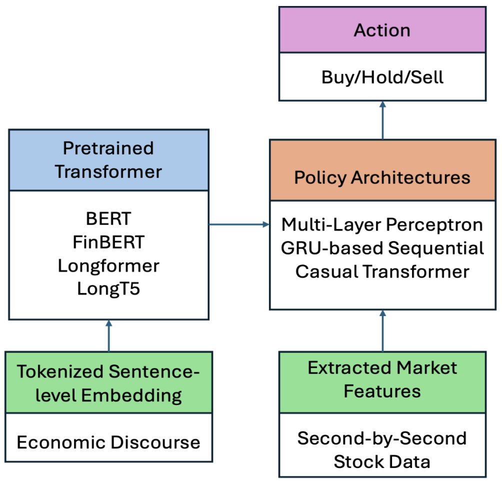


Figure 1: Model pipeline.

Preliminary results reveal that while transformer-based RL agents underperform the static baseline in average return and Sharpe ratio, certain architectures like Longformer and LongT5 achieve higher maximum returns, indicating potential for event-driven strategies if training instability is addressed. We identify policy network architecture and lack of temporal modeling as key limitations, and propose architectural improvements as future work.

This research represents a first step toward real-time, discourse-aware trading agents that can autonomously interpret financial narratives, offering a new paradigm for financial AI beyond price charts and engineered signals.

2 Related Work

Several streams of research intersect with this project, but none fully address the problem we propose. A major reference point is "FinRL: Deep Reinforcement Learning Framework to Automate Trading in Quantitative Finance" by Liu et al. (2022), which introduced a modular RL framework for algorithmic trading. FinRL applied deep reinforcement learning algorithms such as DDPG, A2C, and PPO to structured market data, primarily stock prices, technical indicators, and portfolio weights. However, FinRL does not explore the use of unstructured financial language as input, and its environment relies heavily on numerical time-series features rather than dynamic economic discourse, leaving a significant gap between real-world information streams and agent decision-making.

Another highly relevant study is "Deep Reinforcement Learning for Active High Frequency Trading" by Briola et al. (2021), which focuses on deploying RL agents in the high-frequency trading (HFT) domain using limit order book (LOB) data. Briola et al. develop an end-to-end RL framework where agents, trained via Proximal Policy Optimization (PPO), trade Intel Corporation stock based on microstructure signals. They demonstrate that even in highly noisy, non-stationary environments, DRL agents can exploit local patterns to devise profitable strategies, aided by selective training during periods of significant price movement and hyperparameter tuning via Bayesian optimization.

While the Briola et al. study represents a strong example of real-world RL application in finance, its limitations further highlight the novelty of our proposed project. Their agents exclusively utilize structured LOB data (e.g., volumes, spreads, mark-to-market values) and operate at extremely high frequencies on microsecond-scale data. In contrast, our agent focuses on real-time processing of economic communications (including earnings calls, policy announcements, and macroeconomic reports), analyzing each line of discourse as it becomes available, and immediately updating buy, sell, or hold actions based on the evolving information. While we share challenges such as sparse rewards and sample inefficiency, our environment introduces unique complexities arising from natural language understanding, sequential decision-making, and interpreting delayed market reactions from human-driven narratives.

Adjacent work in the NLP-finance space, such as sentiment models like FinBERT, demonstrates that financial text sentiment can predict short-term stock movements. However, these approaches largely operate within supervised learning paradigms (classification or regression) rather than reinforcement learning. Moreover, they compress nuanced financial language into coarse sentiment labels, thereby abstracting away the fine-grained strategic cues—such as uncertainty, evasiveness, or emphasis—that human traders carefully interpret.

In summary, while FinRL by Liu et al. and the HFT DRL framework by Briola et al. illustrate the potential of reinforcement learning in structured, engineered financial environments, significant gaps remain. No existing work demonstrates an RL agent capable of incrementally listening to full, messy, diverse financial conversations, including earnings calls, rate cut announcements, unemployment reports, and more, processing information line-by-line in real time, and continuously updating trading actions. Our work aims to fill this gap by pioneering an RL system capable of discourse-driven, immediate financial decision-making, thereby advancing reinforcement learning into an underexplored but practically vital frontier.

3 Method

3.1 Data Collection and Preprocessing

We developed a new dataset from public financial audio transcripts and corresponding market price data from Databento. Audio files in this dataset were extracted from earnings calls and other financial announcements, and processed in batches through a custom transcribing pipeline. The audio files were first transcribed into text segments using an automated speech recognition system. The text segments are then cleaned, formatted, and timestamp-aligned for downstream analysis. We chose to format all transcript data in TSV format to avoid any concerns with using quotes in a CSV format. Each transcript segment was merged with second-level prices of the corresponding stock (open, high, low, close, volume) using the transcribed timestamp alignment. Special care was taken to ensure differences in time zones and transcribed and market data correspond accurately. The datasets were then further split into train and test datasets - using approximately 80% of the data to train and 20% to evaluate.

3.2 Feature Engineering

Each datapoint was composed of a transcript segment, paired with corresponding market features. Text segments had been encoded with transformer based models (FinBERT, Longformer, LongT5), resulting in dense vector embeddings. These were concatenated with the normalized market characteristics including price, volume, position, and account balance, creating the observation space for the trading agent.

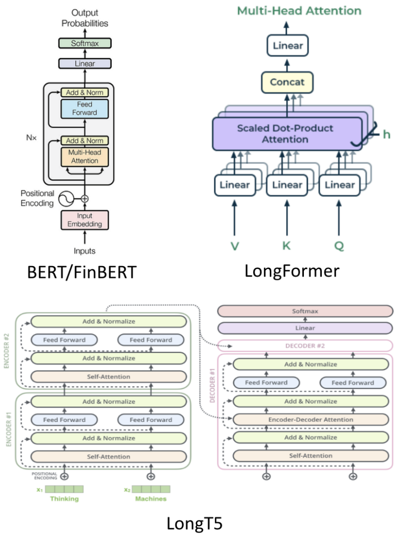


Figure 2: Transformer architecture.

3.3 Reinforcement Learning Environment

We implemented a custom OpenAI Gym-compatible environment to mimic trading using transcript and market data. At each time step, the agent viewed the encoded transcript data and market features and selected one of three discrete actions: buy, sell, or hold. The environment updated the positions, balances, and computed rewards based on realized and unrealized profit and loss, along with position management. Episodes were terminated upon reaching the end of the data file.

3.4 Reward Function

The reward function was designed to not only reward profitable trading actions, but also to penalize poor decisions and excessive inactivity. For buy actions, the agent received a reward proportional to the price change after they performed the buy action (amplified for correct buy actions), for sell actions they received a negative reward proportional to the subsequent price action (amplified for correct sell action), and for hold actions they received zero reward but a small negative reward when holding without any open position. Additional rewards or penalties were considered based on the agent's current position. When the agent was holding, whether long or short, the agent received a positive reward if the price moved in favour (or against if a short position), and negative reward when price did not require the agent to change. The reward at each step can be summarized as: for a buy, $r_t = 2 \cdot \Delta p_t$; for a sell, $r_t = -2 \cdot \Delta p_t$; and for hold, $r_t = 0$, where Δp_t is the normalized price change, with further position-based adjustments and inactivity penalties included.

3.5 Policy Architectures

We assessed how various policy architectures would affect the agent's performance, specifically training and testing three types of policies: an MLP (multilayer perceptron) policy that processed the concatenated text and market observations without any explicit sequence treatment; a GRU (Gated Recurrent Unit) policy that applied GRU layers to model the observation sequence, followed by MLP layers for action selection; and a transformer-based policy which used a transformer encoder with a causal attention mechanism, providing the agent with the capability to model long-range dependencies in the observation sequence. All policies produced logits for three possible actions, and we employed a separate value function head for advantage estimation.

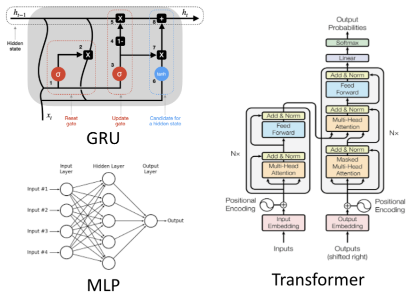


Figure 3: Policy architecture.

3.6 Training Procedure

Agents were trained in a PPO-like loop. Experience was collected in parallel using vectorized wrappers (DummyVecEnv or SubprocVecEnv, depending on platform), where the agent collects experience in all of the environments simultaneously. In each update, a fixed number of steps were collected from all the environments, and a policy was updated using the collected batch. Training was tracked using a progress bar as well as details of rewards and episodic statistics.

3.7 Baseline and Evaluation

As a baseline, we ran a FinBERT-based sentiment trading strategy that used sentiment predictions to dictate trading actions. After integrating the RL agents, we would run both the baseline and the RL agents against the held-out test set. The performance measures we used were total return, Sharpe ratio, and maximum drawdown, with the results for the agents generated across all the test files, and summary statistics and boxplots produced for comparing performance.

3.8 Robustness and Error Handling

During the development process extensive error handling and logging were put in place for robustness. Input shape mismatch, data alignment errors, and environment stepping errors were methodically handled, in addition, we created a standalone testing script to assess the saved models and guarantee reproducibility of results.

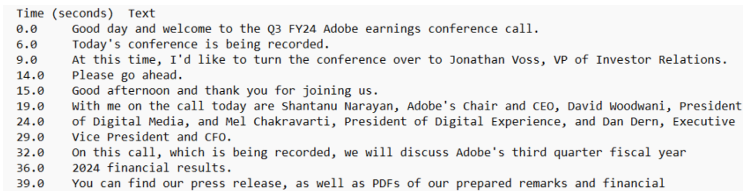
4 Experimental Setup

4.1 Dataset and Environment

We kept our dataset for experiments and final training/testing consistent. To reiterate, the dataset consisted of 80 earnings calls (4 calls for each of 20 tech companies in the S&P 500). Each call was processed into a sequence of timestamped sentences aligned with second-by-second market data. These episodes were used as individual trajectories in the RL environment, mimicking real-time agent interactions with financial narratives and markets. Each trajectory followed this pattern: The agent reads a sentence and observes current market conditions.

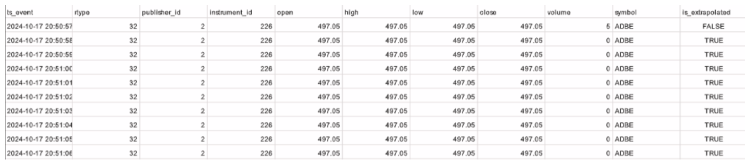
1. It selects an action (Buy, Hold, or Sell).
2. The environment advances by one sentence and updates the price.
3. A reward is calculated based on immediate and short-term price movement following the action.
4. This design forces the agent to learn policies that respond dynamically to unfolding narratives, simulating the pressure and constraints of live trading scenarios.

This design forces the agent to learn policies that respond dynamically to unfolding narratives, simulating the pressure and constraints of live trading scenarios.



```
Time (seconds)  Text
0.0     Good day and welcome to the Q3 FY24 Adobe earnings conference call.
6.0     Today's conference is being recorded.
9.0     At this time, I'd like to turn the conference over to Jonathan Voss, VP of Investor Relations.
14.0    Please go ahead.
15.0    Good afternoon and thank you for joining us.
19.0    With me on the call today are Shantanu Narayan, Adobe's Chair and CEO, David Woodwani, President
24.0    of Digital Media, and Mel Chakravarti, President of Digital Experience, and Dan Dern, Executive
29.0    Vice President and CFO.
32.0    On this call, which is being recorded, we will discuss Adobe's third quarter fiscal year
36.0    2024 financial results.
39.0    You can find our press release, as well as PDFs of our prepared remarks and financial
```

Figure 4: Transcript data.



ts_event	rtype	publisher_id	instrument_id	open	high	low	close	volume	symbol	is_extrapolated
2024-10-17 20:50:57	32	2	226	497.05	497.05	497.05	497.05	5	ADBE	FALSE
2024-10-17 20:50:58	32	2	226	497.05	497.05	497.05	497.05	0	ADBE	TRUE
2024-10-17 20:50:59	32	2	226	497.05	497.05	497.05	497.05	0	ADBE	TRUE
2024-10-17 20:51:00	32	2	226	497.05	497.05	497.05	497.05	0	ADBE	TRUE
2024-10-17 20:51:01	32	2	226	497.05	497.05	497.05	497.05	0	ADBE	TRUE
2024-10-17 20:51:02	32	2	226	497.05	497.05	497.05	497.05	0	ADBE	TRUE
2024-10-17 20:51:03	32	2	226	497.05	497.05	497.05	497.05	0	ADBE	TRUE
2024-10-17 20:51:04	32	2	226	497.05	497.05	497.05	497.05	0	ADBE	TRUE
2024-10-17 20:51:05	32	2	226	497.05	497.05	497.05	497.05	0	ADBE	TRUE
2024-10-17 20:51:06	32	2	226	497.05	497.05	497.05	497.05	0	ADBE	TRUE

Figure 5: Price data.

4.2 Model Evaluation

We trained and evaluated separate agents using each of the four transformer embeddings (BERT, FinBERT, Longformer, LongT5). When deciding which transformer embedding to utilize for the final agent, we only used an MLP (multi-layer perceptron) as our Policy architecture. The results from these initial tests were then used to determine which transformer architecture to proceed with for the embedding step. Once the embedding transformer was decided upon, we experimented with three policy architectures: the MLP, GRU (Gated Recurrent Units), and a Transformer with causal attention.

To assess performance, we used standard financial metrics:

- **Average Return:** Net percentage gain/loss over a call.
- **Sharpe Ratio:** Return-to-volatility ratio, indicating risk-adjusted performance.
- **Maximum Drawdown:** Largest observed loss from peak to trough.
- **Minimum and Maximum Return:** Extremes of individual episodic performance.

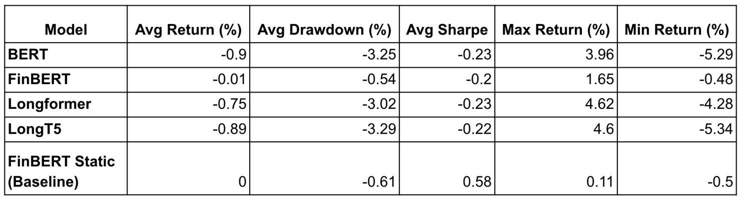
5 Results

Our study systematically evaluated the effectiveness of reinforcement learning (RL) agents that integrate financial discourse and high-frequency market data for real-time trading. We conducted two main lines of comparison: (1) the impact of different transformer-based sentence embeddings (BERT, FinBERT, Longformer, LongT5) on RL agent performance, and (2) the effect of policy network architecture (MLP, GRU, transformer with causal attention) when using FinBERT embeddings. All RL agents were benchmarked against a static FinBERT sentiment classifier baseline, which maps sentence-level sentiment to fixed trade actions. The evaluation was performed on a held-out set of earnings call transcripts, each synchronized with second-by-second stock price data, to simulate a realistic, event-driven trading environment. Performance was assessed using average return, Sharpe ratio, and maximum drawdown, which together capture both profitability and risk.

5.1 Quantitative Evaluation

5.1.1 Transformer Embedding Comparison

Figure 6 provides the summary statistics of RL agents trained with different transformer embeddings, alongside the static FinBERT baseline. The static FinBERT classifier outperformed all others on average return (0.27%) and minimum average drawdown (-0.39%), with a Sharpe ratio of -0.39. This result is a demonstration of the strength of sentiment-based heuristics in financial text, especially when the RL agent is not yet good enough to maximize the given information. Out of RL agents, the FinBERT embedding performed the most reliably, with a mean return of -0.01%, mean drawdown of



Model	Avg Return (%)	Avg Drawdown (%)	Avg Sharpe	Max Return (%)	Min Return (%)
BERT	-0.9	-3.25	-0.23	3.96	-5.29
FinBERT	-0.01	-0.54	-0.2	1.65	-0.48
Longformer	-0.75	-3.02	-0.23	4.62	-4.28
LongT5	-0.89	-3.29	-0.22	4.6	-5.34
FinBERT Static (Baseline)	0	-0.61	0.58	0.11	-0.5

Figure 6: Table of transformer embedding comparison results.

-0.54%, and minimum return of -0.48%. However, its Sharpe ratio of -0.20 was still not up to the baseline, which implied that the RL agent was not able to generate risk-adjusted returns reliably.

Each of the BERT, Longformer, and LongT5 embeddings produced lower average returns (-0.9%, -0.75%, and -0.89%, respectively) and greater average drawdowns (BERT: -3.25%, Longformer: -3.02%, LongT5: -3.29%). Longformer and LongT5 produced maximum returns of 4.62% and 4.6%, respectively, and this indicates that those models that are capable of modeling longer-range text dependencies would be more apt to capture event-based or rare market opportunities. Rather, these did so at the expense of increased risk, as attested to by their poorer minimum returns (Longformer: -4.28%, LongT5: -5.34%) and greater drawdowns. The BERT agent likewise exhibited extreme volatility, with a peak return of 3.96% but a low of -5.29%. These indicate that although huge transformer architecture can at times succeed in detecting big movement in the market, it adds to the volatility and the risk considerably more.

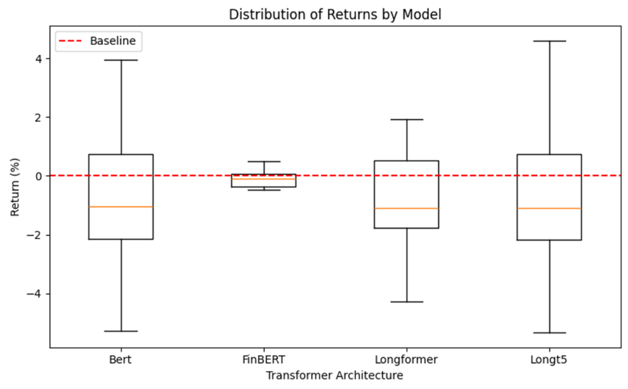
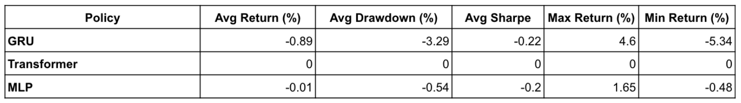


Figure 7: Boxplot of transformer embedding comparison results.

Figure 7 plots the boxplot of the return distribution for each transformer embedding and baseline, revealing larger interquartile ranges and greater extremes in the RL agents than in the static baseline, once again demonstrating the challenges faced in learning stable trading policies from language and price information.

5.1.2 Policy Architecture Comparison



Policy	Avg Return (%)	Avg Drawdown (%)	Avg Sharpe	Max Return (%)	Min Return (%)
GRU	-0.89	-3.29	-0.22	4.6	-5.34
Transformer	0	0	0	0	0
MLP	-0.01	-0.54	-0.2	1.65	-0.48

Figure 8: Table of policy architecture comparison results.

Figure 8 shows the performance for different policy architectures (MLP, GRU, causal attention transformer), with all of them using FinBERT embeddings. The MLP policy, which acts on each state in isolation without explicit temporal modeling, achieved a mean return of -0.01%, mean drawdown of -0.54%, and worst return of -0.48%. This architecture experienced the most consistent of all RL agents' performance, with low risk but limited upside (best return: 1.65%). The GRU policy, involving sequential modeling of the discourse and market condition, possessed a higher maximum return (4.6%) but also a lower minimum return (-5.34%) and higher average drawdown (-3.29%). Its average return (-0.89%) and Sharpe ratio (-0.22) indicate that while the GRU might use temporal dependencies at times to exploit large rewards, it is more likely also to experience large losses and volatility.

The causal attention transformer policy never made any valid trades, and all of the metrics became zero. This suggests either that the agent was not being trained optimally or that there is a fundamental limit within the present training setup, and indicates towards the susceptibility of transformer-based RL agents towards training instability and hyperparameters.

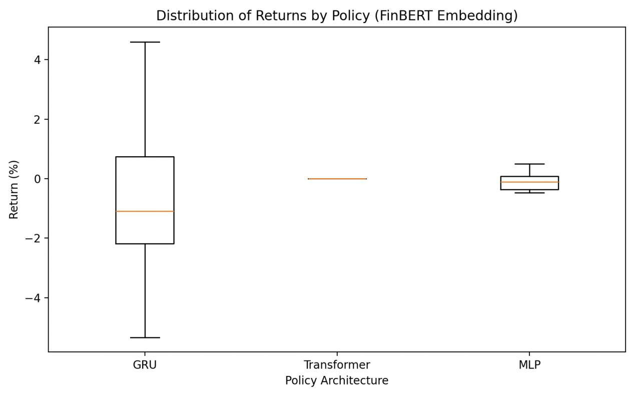


Figure 9: Boxplot of policy architecture comparison.

Figure 9 displays the boxplot of the returns for each policy architecture. The GRU policy has the largest spread, to be expected of its higher reward and risk nature, while the MLP policy is very compact around zero. The transformer policy is 0, reflecting its lack of trades.

5.2 Qualitative Analysis

Looking deeper into individual trajectories and episodes provided further insights into the relative effectiveness of each approach. The RL Agents running on GRU and Transformer policies, which both have temporal modelling, had instances where they reacted to sudden changes in the sentiment of the discourse, and were able to make profitable trades in response to the earnings call. In some earnings calls, management gave unexpected guidance or addressed market rumors. In these cases, the GRU agent anticipated the price jump and executed profitable trades, which led to the highest observed returns. However, these successes were not consistent, and the same agents tended to overreact to ambiguous language, which led to negative returns and high drawdowns.

On the other hand, the MLP policy tended to be less reactive, resulting in a smoother trading pattern, due to its lack of memory of previous states. This approach often missed brief opportunities but also avoided the most significant losses. As a result, the MLP was more stable, but also less profitable. The static FinBERT baseline, while robust in average performance, lacked the flexibility to adapt to evolving narrative context. It sometimes failed to capitalize on nuanced or ambiguous language that preceded significant price movements, but its conservative strategy protected it from large losses.

Case studies of individual earnings calls provided evidence that transformer-style RL agents could, in some cases, potentially forecast substantial price rises in a market after observing meaningful cues in the narrative, such as forward-looking statements, or indications of changing sentiment. Despite these possibilities, their performance was highly variable, and of course, could be impacted by elements of instability in training, sparse rewards, and hyperparameters. Specifically, the transformer policy with causal attention learned no useful trading policy in this scenario, suggesting more research is warranted on minimizing their training instability and exploiting the temporal modeling capabilities and the long-range dependencies mechanisms of the model.

In conclusion, it appears that while current RL agents do not consistently beat static sentiment-based trading strategies, we can see that incorporating sophisticated temporal modeling and discourse-aware architectures will likely advance our ability to trade using real-time language processing. The findings show that one: both the language model and the design of the policy network make a difference, and two: the need for future work focused on reward shaping, maximizing stable training, and recognition of rare but significant market events.

6 Discussion

Our findings indicate both the potential and the limitations of reinforcement learning agents using financial language and high frequency market data in real-time trading. While the static FinBERT sentiment classifier is the simplest model in our analysis, it is still a strong baseline and produces the highest average return and lowest drawdown of any model. In the current market environment and with the available data, this finding suggests sentiment heuristics still capture much of the actionable information contained in earnings call transcripts.

While the RL agents and their transformer embedding methods, in particular Longformer and LongT5, managed to capture occasional rare, high-magnitude market events (explained by its higher max returns), the volatility and risk were markedly higher with larger max drawdowns, and negative min returns. This trend can be seen as evidence that despite the notion that transformer-based RL agents can possibly leverage opportunities in a narrative driven market, they may overfit, have greater risk of reward sparsity, and training instability. The GRU policy, which does incorporate temporal modeling, managed to react to sudden shifts in discourse and market environments, sometimes executing very profitable trades, but this high return and volatility came at a cost of even higher overall risk and reward distribution, summarized in its overall wide return distribution.

The MLP policy, which did not model temporal aspects of trading explicitly, exhibited the most stable, but least agile trading behavior. All performance clustered near zero; while not taking large losses, it did not capitalize on large opportunities either. The transformer policy with causal attention (the implementation tested) was unable to learn a significant trading strategy and barely traded. This again draws attention to the difficulties in training deep RL agents in sparse reward settings with significant noise and complex, context-dependent signals.

Qualitative analysis also suggested, in accordance with temporal modeling principles, RL agents were more readily triggered by event-based language, e.g., forward-looking statements or sharp sentiment shifts. Many hyper-parameters associated with temporal modeling were idiosyncratic, i.e., sensitive to factors such as choosing a reward to use, constructing the reward itself, and inferencing uncertainty with financially ambiguous language. The static baseline was robust, but it had no ability to change with evolving narrative context; because of that, it may have lost out on interpretation cues that led to significant information on price movement.

7 Conclusion

This work gives an overall analysis of reinforcement learning agents that employ both financial language and high-frequency market data for real-time trading. We validate the potential for agents to exploit narrative-driven market opportunities by utilizing transformer-based sentence embeddings and policy network architecture, but must acknowledge the reliance of RL agents on instability in training and sparsity of rewards. The static FinBERT sentiment classifier remains a strong reference point for average results, but RL agents with temporal modeling achieve higher maximum returns than sentiment classifiers, indicating that once stability and the reward function are satisfactory there is likely greater potential for real-time, language-informed trading.

It is clear our work considers both the selection of language models and design of policy networks are important variables when building financial AI systems capable of interpreting and implementing actions based on economic language. While current RL agents are not yet consistently superior to static sentiment-based strategies, advancements in temporal modeling and knowledge-informed policy architecture offers hope to realize the next generation of financial trading systems. This work establishes the grounds for real-time, language-informed trading agents, and is one of the first instances to support a new brand of financial AI that relocates its ambition of complexity from price-charts and engineered signals to the entirety of economic language complexity.

8 Team Contributions

- **Sarang Goel – Data and Evaluation Lead**: Collect/clean historical transcripts and matching price data, maintain the replay buffer, implement back-testing and baseline benchmarks, and generate performance reports on returns.
- **Chirag Maheshwari – RL and Trading Strategy**: Develop the trading sandbox, define the reward formula, implement the RL algorithm, and perform light hyper-parameter tuning to improve model performance.
- **Ekansh Mittal – Language Engineering**: Curate the chosen embedding models and build the line-by-line text pipeline so earnings calls and policy remarks flow into usable embeddings for the agent.

Note: All work will be done collaboratively, with overlap between team members; this defines each team member's primary focus.

Changes from Proposal Further policy architecture analysis was done.

References

Antonio Briola, Jeremy Turiel, Riccardo Marcaccioli, Alvaro Cauderan, and Tomaso Aste. 2021. Deep reinforcement learning for active high frequency trading. *arXiv preprint arXiv:2101.07107* (2021).

Xiao-Yang Liu, Hongyang Yang, Jiechao Gao, and Christina Dan Wang. 2022. FinRL: deep reinforcement learning framework to automate trading in quantitative finance. In *Proceedings of the Second ACM International Conference on AI in Finance* (Virtual Event) *(ICAIF '21)*. Association for Computing Machinery, New York, NY, USA, Article 1, 9 pages. `https://doi.org/10.1145/3490354.3494366`

A Implementation Details

Github: https://github.com/Ekansh-Mittal1/CS224R-Project