Debiasing LLMs by Fine-tuning

Zhenyu Gao, Wenxi Jiang, Yutong Yan *

March 2026

Abstract

Prior research shows that large language models (LLMs) exhibit systematic extrapolation bias when forming predictions from both experimental and real-world data, and that prompt-based approaches appear limited in mitigating this bias. We propose a supervised fine-tuning (SFT) approach that uses Low-Rank Adaptation (LoRA) to train off-the-shelf LLMs on instruction datasets constructed from rational benchmark forecasts. By intervening at the parameter level, SFT changes how LLMs map observed information into forecasts and thereby corrects extrapolation bias. We evaluate the fine-tuned model in two settings: controlled forecasting experiments and cross-sectional stock return prediction. In both applications, the fine-tuned model eliminates extrapolation bias, and the improvements hold out of sample. We provide a low-cost and generalizable method for debiasing LLMs.

*Gao, Jiang, and Yan are at the Department of Finance, CUHK Business School, The Chinese University of Hong Kong. Our correspondences are gaozhenyu@baf.cuhk.edu.hk, wenxijiang@baf.cuhk.edu.hk, and yutong.yan@link.cuhk.edu.hk, respectively. First draft: March 2026.

1 Introduction

Algorithms and data-driven tools have already proven useful in financial decision-making (e.g., D'Acunto, Prabhala, and Rossi, 2019; Cao, Jiang, Wang, and Yang, 2024). More recently, AI agents built on large language models (LLMs) have been shown to replicate human behavior in economic experiments (Horton, 2023).[1] For these agents to be trusted with greater autonomy, however, the behavioral biases embedded in their underlying models become an economically important concern.

A notable example is extrapolation bias, a well-documented tendency among human forecasts to place excessive weight on recent trends (e.g., Da, Huang, and Jin, 2021; Afrouzi et al., 2023). Recent work shows that LLMs exhibit the same pattern: Chen, Green, Gulen, and Zhou (2024) find that LLMs extrapolate trends when forecasting stock returns, and that this bias is difficult to correct. Prompt-based approaches, such as instructing the model to reason in a fully rational way, have little effect, suggesting the bias is encoded in the model's learned representations rather than driven by how the prompt is framed.

To address this challenge, we develop a supervised fine-tuning (SFT) approach to systematically mitigate behavioral biases in LLMs. The key insight is that if extrapolation bias is embedded in the model's parameters during pre-training (e.g., Gallegos et al., 2024), then correcting it requires intervening at the parameter level rather than through prompting. To do so, we first construct instruction datasets: each prompt presents the model with a sequence of past data (e.g., stock returns), and the target response encodes a rational benchmark forecast. Then, we fine-tune on these instruction data using Low-Rank Adaptation (LoRA; Hu et al., 2022), which directly reshapes how the model maps observed data to predictions. This replaces biased extrapolation patterns with rational forecasts while preserving the model's general language understanding.

Modern LLMs are built in two stages: (1) **pre-training** on a massive corpus of raw text to predict the next token in a sequence, and (2) **alignment** where the model is fine-tuned on curated examples and human feedback to produce a helpful conversational assistant. The pre-training corpus is rich in financial content: journalism, analyst reports, earnings call transcripts, and investment forums, where extrapolative language about firm performance

[1] Also see, e.g., Brown et al. (2020); Chen et al. (2021); Achiam et al. (2023); Bai et al. (2023); Chowdhery et al. (2023); Touvron et al. (2023) for foundational LLM architectures and Yao et al. (2022); OpenAI (2023); Wang et al. (2023a); Park et al. (2023); Schluntz and Zhang (2024) for LLM-based agents.

is pervasive. Because the model learns by internalizing the statistical patterns in this text, its parameters encode not only factual knowledge and reasoning ability, but also the systematic biases present in the data. Alignment shapes how the model communicates, including its tone, safety, and instruction-following, but does not correct its underlying beliefs about how data are generated. As a result, prompting-based approaches fail to mitigate LLM forecasting biases as they operate entirely at inference time without touching the model's parameters.

Our framework introduces an additional fine-tuning step after alignment, directly targeting the model's forecasting behavior before deployment. We maintain a strict separation between training, validation, and test data. To establish a baseline, we first present the model with a held-out set of forecasting prompts and record its raw predictions. Each prompt provides a history of past stock returns and asks the model to forecast the next period, with no guidance on how to form that forecast. The model simply produces whatever prediction its pre-trained weights generate. Comparing these predictions against rational benchmarks reveals the systematic biases we aim to correct. These prompts are reserved as the test set and never exposed to the model during training.

We then construct a separate instruction dataset of prompt-response pairs. The prompts are identical to those in the previous step, while the responses encode rational forecasts derived either from a rational expectations benchmark or from realized future returns. Each pair is a corrective example: it presents the model with the same input it already sees, but pairs it with the answer a disciplined forecaster would give. A portion of this instruction dataset is also held out as a validation set, used to monitor generalization during training and to determine the stopping rule.

A key technical choice is how to update the model's parameters. In earlier, smaller models such as BERT-large (340 million parameters), the standard approach was full fine-tuning: updating every weight on task-specific data (Devlin, Chang, Lee, and Toutanova, 2019; Huang, Wang, and Yang, 2023). Modern LLMs, however, are orders of magnitude larger. Qwen3-32B, the model we use, contains 32 billion parameters, making full fine-tuning problematic for two reasons. First, it is computationally prohibitive, requiring hundreds of gigabytes of GPU memory. Second, and more importantly, it risks *catastrophic forgetting* (Goodfellow et al., 2013): modifying all parameters simultaneously can degrade the model's general language understanding, instruction-following, and numerical reasoning.

Since our goal is to correct a specific forecasting bias while preserving all other capabilities, full fine-tuning is poorly suited to the task.

We instead fine-tune using Low-Rank Adaptation (LoRA; Hu et al., 2022), the most widely used parameter-efficient fine-tuning method in machine learning. Rather than updating the full model, LoRA freezes the original pre-trained weights and attaches a small set of trainable parameters to each attention layer. Only these added parameters are updated during training, and because they typically represent less than 1% of the full model, LoRA reduces computational cost by several orders of magnitude. More importantly, keeping the original weights frozen preserves the model's general language understanding while adjusting only the forecasting behavior of interest. Once training is complete, the added parameters are merged back into the original weights, so the fine-tuned model can be deployed with no additional inference overhead. We use early stopping based on validation performance to guard against overfitting.

We implement the framework on Qwen3-32B (Yang et al., 2025), an open-weight LLM with 32 billion parameters. We choose an open-weight model deliberately: unlike proprietary LLMs accessed via APIs, open-weight models allow researchers to inspect and modify internal parameters, which is a prerequisite for our fine-tuning approach. The procedure is also computationally inexpensive. The entire training runs on two NVIDIA GPUs at a cost of a few hundred dollars on a commercial cloud cluster, negligible relative to the millions required to pre-train a frontier LLM from scratch.

We validate our fine-tuned model in two settings. The first replicates the controlled forecasting experiment of Afrouzi et al. (2023), replacing human subjects with the LLM. In that study, participants are assigned to one of six AR(1) processes with persistence $\rho \in 0.0, 0.2, 0.4, 0.6, 0.8, 1.0$, observe 40 historical realizations, and submit one- and two-period-ahead forecasts over 40 rounds. Overreaction is quantified by regressing forecast errors on forecast revisions: a negative coefficient ($\hat{b}$) indicates that upward revisions systematically predict negative errors, a sign of overreaction.

We adapt this design for a text-based LLM. The off-the-shelf Qwen model closely replicates the pattern found in the human data: the overreaction coefficient b is negative and statistically significant at the 1% level across all six persistence conditions, most negative at $\rho = 0.0$ ($\hat{b} = -0.456$, $t = -19.05$) and monotonically less so as persistence increases, reaching $\hat{b} = -0.260$ ($t = -10.37$) at $\rho = 1.0$. Overreaction is strongest for transitory

processes and weakest for random walks, exactly as observed in the human data.

We then fine-tune on a training dataset of approximately 30,000 round-level observations, constructed from 128 independent AR(1) realizations for each of the six persistence values, using conditional expectations as learning targets rather than ex-post realizations.[2] After fine-tuning, the overreaction bias becomes statistically insignificant: point estimates of b range from -0.073 ($t = -1.54$) at $\rho = 0.0$ to -0.027 ($t = -0.97$) at $\rho = 1.0$.

The second exercise turns to forecasting actual stock returns. Both studies document a pronounced extrapolation bias: human forecasts (Da, Huang, and Jin, 2021) and ChatGPT-4 (Chen, Green, Gulen, and Zhou, 2024) alike place excessive weight on recent past returns. We adopt a similar design and prompt the model to forecast monthly returns for SP 500 constituents using trailing twelve-month return histories. We anonymize the prompt, supplying only numerical return histories with no firm or date identifiers, to mitigate lookahead bias. We quantify extrapolation by regressing the model's forecasts on lagged stock returns. The baseline model loads positively on all lags, with the coefficient on the most recent-month return equal to 0.394 ($t = 53.92$) and declining with lag length, confirming excessive weight on recent performance.

To implement the SFT procedure, we divide the sample into three non-overlapping periods following Gu, Kelly, and Xiu (2020): training (January 2001–December 2011), validation (January 2012–December 2015), and test (January 2016–December 2024). Unlike the previous exercise, we use realized next-month returns as the learning target, since the conditional expected return is not directly observable. By training on the empirical distribution of monthly returns conditional on trailing histories, the model learns the short-term reversal pattern in stock returns. After fine-tuning, the extrapolative loading is eliminated: coefficients on all lagged returns turn negative, with the most recent lag equal to -0.120 ($t = -23.21$). The fine-tuned model has internalized from the training data that recent outperformers tend to reverse, producing forecasts that better reflect the actual return-generating process.

The correction holds out of sample in both the controlled experimental and empirical settings, ruling out the possibility that the behavioral change is an artifact of in-sample fitting. More broadly, our approach offers a low-cost and generalizable approach to aligning

[2]Training on realizations would introduce an irreducible noise component that mechanically biases the diagnostic regression through the common noise shared between the forecast error and the forecast revision. Training on conditional expectations eliminates this source of mechanical bias.

LLM behavior with rational benchmarks across economic settings. This is a prerequisite for the responsible deployment of AI agents in financial decision-making.

Related Literature

Use of LLMs in Finance A growing body of work applies LLMs to extract economically meaningful signals from financial text. Studies in this vein use corporate disclosures and earnings calls to predict firm actions (e.g., Cao, Jiang, Yang, and Zhang, 2023; Jha, Qian, Weber, and Yang, 2024), financial news to forecast stock returns (e.g., Chen, Kelly, and Xiu, 2022; Lopez-Lira and Tang, 2023), central bank communications to extract macroeconomic expectations (e.g., Hansen and Kazinnik, 2024), historical news to generate economic beliefs (Bybee, 2023), and mutual fund manager reports and analyst research to study belief formation (e.g., Gao, Xiong, and Yuan, 2024; Ke, 2024). Beyond using LLMs as text processors, a parallel strand deploys them as autonomous agents capable of replicating experimental economics findings, generating emergent market dynamics in simulated trading environments, and nowcasting stock returns from autonomously gathered information (e.g., Horton, 2023; Lopez-Lira, 2025; Chen and Pu, 2026). A separate line of work seeks to open the LLM black box at inference time: Chen, Didisheim, and Somoza (2024) show that next-token conditional probabilities provide an interpretable measure of prediction uncertainty that improves portfolio performance, while Chen, Didisheim, Somoza, and Tian (2025) apply Gemma Scope sparse autoencoders (Lieberum et al., 2024), developed by Google Deepmind, to extract interpretable concepts driving LLM forecasts and adjust model behavior along dimensions like risk aversion and optimism by adjusting linear feature vectors within the model.

Our work instead intervenes at the alignment stage, using SFT to directly modify how a chat-LLM forms its beliefs.

LLM biases in behavioral economics. Most closely related to our work is the emerging literature examining whether LLMs inherit human-like behavioral biases. Chen, Green, Gulen, and Zhou (2024) show that LLMs over-extrapolate when forecasting stock returns and that prompt-based interventions, such as role-based instructions, have a negligible effect on this bias. Bini, Cong, Huang, and Jin (2025) conduct a comprehensive battery of behavioral economics experiments and find that while rational prompting can modestly re-

duce biases in some settings, it does not fully eliminate them. Our paper focuses on the stock return forecasting setting of Chen, Green, Gulen, and Zhou (2024), where prompting is least effective, and shows that parameter-level intervention through SFT can succeed where prompting fails. Other studies reinforce the broader picture that LLMs exhibit human-like behavioral biases across economic tasks (Ross, Kim, and Lo, 2024; Fedyk, Kakhbod, Li, and Malmendier, 2024; Wu, Xi, and Xie, 2025).

Our SFT approach contributes to this literature by showing that LLM biases can be effectively corrected through targeted parameter-level intervention. The method is low-cost and potentially generalizable to aligning LLM behavior with rational benchmarks across a broad range of economic settings.

2 Methodology

2.1 Fine-Tuning LLMs

Figure I summarizes the modern LLM training pipeline, which consists of a training phase, in which the model learns from data, and a prompting phase, in which users interact with the trained model. To fix ideas, consider the progression from GPT-3 (Brown et al., 2020) to ChatGPT-3.5. GPT-3 was trained on approximately 570 GB of filtered text via next-token prediction, producing a base model capable of generating fluent text but not designed to follow instructions or engage in dialogue. OpenAI subsequently fine-tuned this base model using supervised learning on curated prompt-response dataset and reinforcement learning from human feedback (RLHF; Ouyang et al., 2022), yielding ChatGPT-3.5, the conversational assistant released to end users. This two-stage process, pre-training on raw text followed by alignment to human preferences, is now the standard paradigm across all major LLM families.

Training begins with a massive corpus of raw, unlabeled text, typically on the order of tens of terabytes, assembled from diverse sources including web crawls (e.g., CommonCrawl), books, academic papers, code repositories (e.g., GitHub), and encyclopedic sources (e.g., Wikipedia). This corpus constitutes the model's entire exposure to the world. Critically for our purposes, it includes financial journalism, analyst commentary, earnings call transcripts, and online investment forums, sources in which extrapolative language about asset returns is pervasive.

A Transformer neural network (Vaswani et al., 2017), consisting of repeated layers of self-attention and feed-forward blocks, is then trained on this corpus via next-token prediction. Given a sequence of tokens $(x_1, x_2, \ldots, x_{t-1})$, the model learns to minimize the negative log-likelihood of each successive token:

$$\mathscr{L}_{\text{PT}} = -\sum_{i=1}^{T} \log P_\theta(x_i \mid x_{<i}), \tag{1}$$

where θ denotes the model's parameters and the sum runs over all tokens in the corpus. This objective is entirely self-supervised: no human labels or curated examples are required. By predicting the next token across trillions of training examples, the model internalizes the statistical regularities of its training data, including factual knowledge, reasoning patterns, and, unavoidably, any systematic biases present in the text. The result is a *base model*: a powerful but unrefined text generator that is not yet designed to follow instructions or interact with users.

The base model is then aligned to produce helpful, harmless, and instruction-following responses. The alignment stage typically begins with supervised fine-tuning (SFT) on curated prompt-response pairs, teaching the model the format and style of a conversational assistant. This is followed by preference optimization methods, such as direct preference optimization (DPO; Rafailov et al., 2023) or group relative policy optimization (GRPO; Shao et al., 2024), that further refine the model's outputs based on human judgments of response quality. The aligned model, often referred to as a Chat-LLM, is the version deployed to end users. Alignment shapes *how* the model responds, its tone, safety, and adherence to instructions, but does not target the *substance* of its beliefs about data-generating processes or economic relationships.

At inference time, users interact with the aligned Chat-LLM through natural language prompts. The model receives a query, generates a response token by token, and returns its output. This is the stage at which all existing attempts to mitigate LLM forecasting biases operate: by modifying the prompt (e.g., role-based instructions, few-shot demonstrations, chain-of-thought reasoning) without altering the model's parameters. As documented by Chen, Green, Gulen, and Zhou (2024), prompt-level interventions have limited efficacy against extrapolation bias. This bias is baked into the model's parameters during training and cannot be corrected by reframing the input alone.

The central insight of our approach is that correcting a bias embedded in the model's parameters requires intervening at the parameter level. We introduce an additional fine-tuning step between the standard alignment phase and deployment. Figure II illustrates this procedure.

We begin by constructing a dataset of N prompt-response pairs in which each prompt presents a return sequence and each response contains the corresponding rational benchmark forecast. Starting from the aligned Chat-LLM, we then apply supervised fine-tuning (SFT) to teach the model to produce rational forecasts in place of its default overextrapolative predictions (Wang et al., 2023b). The SFT trainer minimizes the cross-entropy loss over the target completions:

$$\mathcal{L}_{\text{SFT}} = -\sum_{i=1}^{N}\sum_{j=1}^{T_i} \log p_\theta\Big(y_j^{(i)} \mid y_{<j}^{(i)}, x^{(i)}\Big), \tag{2}$$

where $x^{(i)}$ is the i-th input prompt, $y^{(i)} = (y_1^{(i)}, \ldots, y_{T_i}^{(i)})$ is the target rational forecast, and θ denotes the model's parameters. By training the model to assign high probability to rational forecasts conditional on observed return sequences, SFT directly reshapes the internal representations that govern how the model maps historical data to predictions.

A naive approach would be to update all of the model's parameters on the instructional dataset, known as full fine-tuning. In the earlier era of smaller language models, full fine-tuning was standard practice. BERT-large (Devlin, Chang, Lee, and Toutanova, 2019), for example, contains 340 million parameters, small enough that researchers could update every weight on a single GPU in a matter of hours. Many influential applications of language models in finance, such as sentiment analysis of financial text (e.g., Huang, Wang, and Yang, 2023), rely on full fine-tuning of BERT-scale models. However, modern LLMs are orders of magnitude larger: Qwen3-32B contains 32 billion parameters, roughly 100 times the size of BERT-large. At this scale, full fine-tuning becomes problematic for two reasons. First, it is computationally prohibitive: storing and updating a separate copy of every weight demands hundreds of gigabytes of GPU memory and far more training time. Second, and more fundamentally, full fine-tuning risks *catastrophic forgetting*: by modifying the entire parameter space, the model may lose the general language understanding, instruction-following behavior, and numerical reasoning capabilities that it acquired during pre-training. Since our goal is to correct a specific bias in how the model forms forecasts

while leaving all other capabilities intact, full fine-tuning is poorly suited to the task.

We instead adopt Low-Rank Adaptation (LoRA; Hu et al., 2022), illustrated in Figure III. LoRA is the most widely used method for parameter-efficient fine-tuning of LLMs. The key idea is that the adjustment needed for a specific task can be captured by a small number of additional parameters, without modifying the original model. As shown in Figure III, LoRA freezes the entire set of pretrained weights and attaches a compact pair of trainable matrices alongside each attention layer. When the model processes an input, the output is the sum of the original pretrained computation and the small task-specific adjustment. During fine-tuning, only these added parameters are updated; the pretrained weights remain untouched. Because the added parameters typically represent less than 1% of the full model, LoRA reduces the computational budget by several orders of magnitude while preserving the model's general language understanding, reasoning ability, and instruction-following behavior. Once training is complete, the added parameters can be merged back into the original weights, so the resulting fine-tuned Chat-LLM can be queried through standard prompting, exactly as the original model, but now produces forecasts consistent with the rational benchmark rather than over-extrapolating recent trends.

2.2 Debiasing Framework

Our debiasing framework rests on a simple intuition: if an LLM systematically overreacts to recent returns or extrapolates transitory trends, we can teach it not to. The key is to show the model what rational forecasts look like and let it learn the correction itself. We implement this idea through supervised fine-tuning (SFT), maintaining a strict separation between the data used to train the model and the data used to evaluate whether debiasing has succeeded. Figure IV summarizes the framework.

We begin with a diagnostic exercise. We present the baseline LLM with a held-out set of forecasting prompts and collect its raw predictions. Each prompt supplies the model with a history of past returns and asks it to forecast the next one or two periods ahead. The model receives no guidance on how to form its forecast; it simply produces whatever prediction its pretrained weights generate. Comparing these predictions against rational benchmarks, whether derived from a rational expectations model or from ex-post realizations, exposes the systematic biases we aim to correct. A model exhibiting overreaction, for instance, will chase recent momentum too aggressively: after a sequence of negative returns, it predicts

further steep declines even when the data-generating process implies mean reversion. A model exhibiting extrapolation will latch onto short-run patterns and project them forward as if they were permanent, ignoring the transitory nature of the underlying shocks. This diagnostic phase pins down both the direction and the severity of the distortion. Critically, the prompts used in this phase are set aside as our test set and are never exposed to the model during training.

The central ingredient of the framework is the instructional dataset. We construct a separate collection of prompt-response pairs in which the prompts share the same structure as those in the test set, presenting the model with a return history and asking for a forecast, but the responses now reflect what a rational forecaster would predict. These target responses can be sourced in two ways: from the conditional forecasts implied by a rational expectations benchmark, or from the realized future returns that the model is trying to predict. In either case, the target encodes the behavior we want the model to internalize. Where the baseline LLM might dramatically overweight a recent crash and forecast continued freefall, the instructional response instead shows a measured, mean-reverting prediction. Where the baseline might project a brief rally into an extended boom, the instructional response shows a tempered, stabilizing path. In essence, each prompt-response pair is a corrective example, pairing the same information the model already sees with the answer a disciplined forecaster would give. We further partition a portion of this instructional data into a validation set, which plays no role in updating model weights but is used to track generalization performance during training and to inform our stopping rule. Together, the training and validation splits constitute the data the model learns from, entirely disjoint from the test set reserved for final evaluation.

With the instructional dataset in hand, we fine-tune the LLM using Low-Rank Adaptation (LoRA; Hu et al., 2022). Rather than updating all of the model's parameters, LoRA freezes the original pretrained weights and injects a small number of low-rank trainable matrices into the model's attention layers. During fine-tuning, only these added parameters are updated, while the vast majority of the model remains untouched. This design choice is deliberate and serves two purposes. First, it is computationally efficient: fine-tuning billions of parameters from scratch would be prohibitively expensive, whereas LoRA requires only a fraction of the memory and compute[3]. Second, and more important for our setting,

[3]Effective training size is usually less than 1% of the full model.

it preserves the LLM's general language understanding and reasoning abilities while surgically adjusting only the forecasting behavior of interest. The model retains its capacity to parse numerical inputs, follow instructions, and produce coherent outputs; what changes is the mapping from observed return histories to predicted future returns. Throughout training, we monitor the loss on the training set alongside the model's forecasting performance on the validation set. We employ early stopping: once validation performance ceases to improve, we halt training and select the checkpoint with the best validation performance. This guards against overfitting to the idiosyncrasies of the training sample and ensures that the debiasing generalizes beyond the specific examples the model has seen.

The final and most important test is whether the debiased model generalizes to data it has never encountered. We return to the held-out test set from the diagnostic phase and feed the same prompts to the fine-tuned LLM. Because these prompts were excluded before training began, the model's responses to them constitute a clean, out-of-sample evaluation. We then compare the debiased model's forecasts against the same rational benchmarks used in the diagnostic phase. If fine-tuning has worked, the gap between the model's predictions and the rational forecasts should narrow substantially: overreaction to recent returns should be attenuated, extrapolation of transitory patterns should diminish, and the overall distribution of forecast errors should shift toward the behavior implied by the benchmarks. This out-of-sample evaluation discipline is what allows us to claim that the debiasing is genuine, a learned change in how the model processes return histories, rather than an artifact of in-sample fitting to the particular sequences it trained on.

2.3 Implementation Details

We implement the debiasing framework on Qwen3-32B (Yang et al., 2025), an open-weight LLM with 32B parameters, as it is the most popular dense model with over 30B parameters and offers the best balance between capability and accessibility among the open-source models. We choose an open-weight model deliberately: unlike proprietary LLMs accessed through APIs, open-weight models allow researchers to inspect, modify, and retrain the model's internal parameters, which is a prerequisite for our fine-tuning approach. Qwen3-32B offers a strong balance between forecasting capability and tractability; it is large enough to exhibit the sophisticated behavioral patterns we aim to correct, yet small enough to fine-tune efficiently with LoRA.

After downloading the pretrained model weights, we attach LoRA adapter layers to the model's attention modules. These adapters introduce a small number of trainable parameters while leaving most of the parameters entirely frozen. The model then undergoes supervised fine-tuning on our instructional dataset: at each training step, the model receives a forecasting prompt, generates a prediction, and updates only the adapter weights to bring that prediction closer to the rational target response. We monitor performance on the validation set throughout training and apply early stopping to select the checkpoint at which debiasing is strongest without sacrificing out-of-sample generalization.

We implement the training pipeline using the Hugging Face `transformers` library (Wolf et al., 2020) for model loading, tokenization, and training orchestration, and the Hugging Face `peft` library (Mangrulkar et al., 2022) for LoRA integration. The entire procedure runs on two NVIDIA A100 GPUs. The estimated cost is a few hundred dollars on a commercial cloud cluster, a negligible expense relative to the millions of dollars typically required to pretrain a frontier LLM from scratch.

For prompting, we use `vLLM` (Kwon et al., 2023), an open-source, high-throughput inference library designed to process large volumes of prompts in parallel for LLMs efficiently. Because our experimental design requires eliciting forecasts across a large number of prompts, often numbering in the tens of thousands, efficient inference is essential to keeping the overall pipeline tractable. We set the sampling temperature to zero to produce deterministic outputs.[4]

3 Results

3.1 AR(1) process

In this exercise, we generally follow the approach of Afrouzi et al. (2023), with the difference that we use LLM as our participants, while they recruit humans. Afrouzi et al. (2023) design a controlled forecasting experiment to measure biases in expectations.

In their baseline experiment, they recruit approximately 207 participants from Amazon's

[4]Greedy decoding with zero temperature selects the highest-probability token at each step, but floating-point operations on GPUs are not perfectly associative. Changes in batch size alter the order of parallel computations within the attention mechanism, introducing small numerical discrepancies that can occasionally shift which token ranks highest. This is a well-documented property of parallelized GPU inference rather than a limitation specific to our setting. We have verified that our qualitative results are robust to varying the batch size.

Mechanical Turk platform and randomly assign each to one of six AR(1) processes,

$$x_t = \rho\, x_{t-1} + \varepsilon_t, \qquad \varepsilon_t \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma^2), \tag{3}$$

with persistence $\rho \in \{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$, a long run mean of zero, and a conditional volatility of $\sigma = 20$. Each participant first observes 40 historical realizations of the assigned process displayed as a time series graph. The forecasting game then proceeds for 40 rounds.

At each round t, participant i sees the realized history $x_{t-39}, \ldots, x_{t-1}, x_t$ and submits two forecasts: a one period ahead forecast $F_{i,t}\, x_{t+1}$ and a two period ahead forecast $F_{i,t}\, x_{t+2}$. The true value x_{t+1} is then revealed, the graph is updated, and round $t+1$ begins. Because participants forecast both one and two periods ahead in every round, two forecasts of the same target x_{t+1} are collected from each participant across consecutive rounds: the two period ahead forecast $F_{i,t-1}\, x_{t+1}$ submitted in round $t-1$ (when x_{t-1} was the latest observation), and the one period ahead forecast $F_{i,t}\, x_{t+1}$ submitted in round t (after x_t is observed). The difference $F_{i,t}\, x_{t+1} - F_{i,t-1}\, x_{t+1}$ is the forecast revision, capturing how participant i updated the prediction of x_{t+1} upon seeing the new realization x_t. The forecast error $x_{t+1} - F_{i,t}\, x_{t+1}$ is realized once x_{t+1} is revealed at the start of round $t+1$.

To quantify overreaction, the authors estimate, for each level of ρ, the following panel regression:

$$\underbrace{x_{t+1} - F_{i,t}\, x_{t+1}}_{\text{forecast error}} = a + b \underbrace{\left(F_{i,t}\, x_{t+1} - F_{i,t-1}\, x_{t+1}\right)}_{\text{forecast revision}} + v_{i,t}, \tag{4}$$

A negative coefficient b indicates that upward revisions systematically predict negative forecast errors, which is the signature of overreaction.

We take this experimental framework as our starting point and replicate it with Qwen3-32B serving as participants. Since the original experiment relies on a visual interface in which participants observe and interact with a time series graph, we adapt the design for text based language models that do not process visual input. Specifically, at each round t, we provide the model with the numerical values of the past realizations $x_{t-39}, \ldots, x_{t-1}, x_t$ and ask it to submit forecasts of changes of x_{t+1} and x_{t+2}. A notable feature of our prompt design is that we elicit predicted *changes* rather than predicted *levels*. This choice is motivated by the cognitive process that the original experiment documents: when human subjects observe a time series graph, they anchor on the last realized value and form a judgment about the magnitude and direction of the next movement. Eliciting changes from the LLM

parallels this anchoring process, prompting the model to condition explicitly on the most recent realization before producing each forecast. Figure V displays an example prompt and model response. We otherwise retain the same process parameters as Afrouzi et al. (2023), setting the long run mean to zero, the conditional volatility to 20, and the persistence to each of $\rho \in \{0, 0.2, 0.4, 0.6, 0.8, 1\}$. For each value of ρ, we sample 32 independent AR(1) realizations as our test set.

We estimate the error revision regression in Equation (4) separately for each level of ρ using the LLM generated forecasts. Figure VII and Table III report the results. The estimated coefficient b is negative and statistically significant at the 1% level for all six persistence conditions, confirming that the language model's forecasts systematically overreact to recent information. Moreover, the magnitude of overreaction varies with process persistence in the same direction as in the human data: b is most negative at $\rho = 0.0$ ($\hat{b} = -0.456$, $t = -19.05$) and becomes monotonically less negative as persistence increases, reaching $\hat{b} = -0.260$ ($t = -10.37$) at $\rho = 1.0$. This monotonic pattern is the central empirical finding of Afrouzi et al. (2023), and the LLM replicates it qualitatively: overreaction is strongest for transitory processes and weakest for random walks.

Training dataset and validation dataset. To construct a benchmark against which the baseline LLM behavior can be compared, we curate separate training and validation datasets. The training dataset is used to fine-tune the LLM toward the rational expectations forecast, while the validation dataset serves to monitor out-of-sample performance during the fine-tuning process and to guard against overfitting.

The training dataset comprises a total of around 30,000 round-level observations, constructed from 128 independent AR(1) realizations for each of the six persistence values. The validation dataset is constructed with the same dimensions as the test set. For each observation in both the training and validation datasets, we define learning targets that encode the conditionally optimal forecast.

We set learning targets to conditional expectations rather than ex-post realizations. If the model were trained on realizations, its forecasts would contain an irreducible noise component. In the diagnostic regression (4), this noise enters the forecast error on the left-hand side with a negative sign, since the forecast is subtracted from the realization, and enters the forecast revision on the right-hand side with a positive sign, since the current-

period forecast is added in computing the revision. Even if the model's point forecasts are conditionally unbiased, this common noise term induces a negative covariance between the dependent variable and the regressor, mechanically biasing the OLS estimate of b downward. Training on conditional expectations removes this noise from the learning targets and eliminates this source of mechanical bias.

For the one-period-ahead change $\Delta x_{t+1} \equiv x_{t+1} - x_t$, the rational forecast conditional on the observed history is $\mathbb{E}[x_{t+1} \mid x_t] - x_t = (\rho - 1)\, x_t$, given that the long-run mean of the process is zero. Because this target equals zero identically when $\rho = 0$, a training set composed entirely of exact zeros would present the fine-tuning procedure with a degenerate signal. To avoid this issue, we perturb the target by adding independent Gaussian noise with mean zero and standard deviation 0.1. [5]

For the two-period-ahead change $\Delta x_{t+2} \equiv x_{t+2} - x_{t+1}$, the learning target is drawn from a normal distribution with mean zero and standard deviation 20.[6] The economic rationale for this choice is as follows. Under rational expectations, the optimal forecast revision for the two-period-ahead target should be zero in expectation: conditional on information available at time $t-1$, the forecast $F_{i,t-1}\, x_{t+2}$ already incorporates the best available prediction, so the arrival of x_t should not produce a systematic upward or downward revision of $\mathbb{E}[x_{t+2}]$ once the information is appropriately discounted. By setting the learning target to unpredictable noise centered at zero, we train the model to treat the two-period-ahead change as unpredictable, discouraging it from extrapolating the most recent innovation into the two-period-ahead prediction.

Fine-tuned LLM corrects the bias. The preceding analysis establishes that the pretrained LLM systematically overreacts to recent information, replicating the central finding of Afrouzi et al. (2023) with human subjects. A natural question is whether this bias reflects a fundamental limitation of the model architecture or whether it can be eliminated through supervised learning on rational expectations targets. To investigate, we fine-tune Qwen3-32B on the training dataset described above and evaluate the resulting model on the same held-out test set.

Figure VIII and Table IV present the estimates of the error revision regression in Equa-

[5] We have verified that the results are robust to alternative choices of the perturbation scale, including a standard deviation of 0.5.

[6] We have also experimented with standard deviation of 10 and find qualitatively identical results.

tion (4) using forecasts from the fine-tuned model. The overreaction bias documented in the baseline specification is eliminated. Across all six persistence conditions, the estimated coefficient on the forecast revision is small in magnitude and statistically indistinguishable from zero at conventional significance levels. Point estimates range from $\hat{b} = -0.073$ ($t = -1.54$) at $\rho = 0.0$ to $\hat{b} = -0.027$ ($t = -0.97$) at $\rho = 1.0$, with none exceeding the 10% critical value. Taken together, these results indicate that the overreaction bias exhibited is a learned regularity that can be corrected through fine-tuning on rational targets.

3.2 Stock Return Prediction

We now turn from forecasting synthetic AR(1) processes to forecasting actual stock returns in the cross-section. In this exercise, we generally follow the approach of Da, Huang, and Jin (2021) and Chen, Green, Gulen, and Zhou (2024). Da, Huang, and Jin (2021) use novel data from the Forcerank platform, a crowdsourcing contest in which participants rank stocks by their expected returns over the coming week, to study how investors form beliefs about short-horizon returns. They document that investors extrapolate from stocks' recent past returns, placing greater weight on more recent realizations. Chen, Green, Gulen, and Zhou (2024) adopt the same experimental framework, replacing human participants with ChatGPT-4. They find that LLM generated rankings load positively on lagged returns with a pronounced recency effect, mirroring the pattern documented in human forecasts.

Our analysis departs from both studies in the forecasting target. While Da, Huang, and Jin (2021) and Chen, Green, Gulen, and Zhou (2024) focus on the Forcerank contest setting, where participants rank a curated set of stocks, we instead ask the LLM to forecast monthly returns for S&P 500 constituents. This shift in scope reflects both a practical constraint and a substantive advantage. Because we do not have access to the proprietary Forcerank platform data, we construct our own forecasting exercise using WRDS CRSP for S&P 500 constituents. The S&P 500 offers a natural compromise: it captures approximately 80% of total U.S. equity market capitalization and spans the full range of sectors represented in the broader market, while keeping the prompting budget tractable. We note that our methodology is readily extensible to the full cross-section of listed equities as inference costs decline.

As in the AR(1) exercise, we use Qwen3-32B as the forecasting agent, replacing human

participants in the original experimental setting. Figure VI displays an example prompt and model response. For each stock i in each month t, we provide the model with the trailing twelve monthly returns $r_{i,t-11}, r_{i,t-10}, \ldots, r_{i,t}$ and ask it to forecast the stock's return over the following month, $r_{i,t+1}$. As in the AR(1) exercise, we anonymize the prompt by supplying only numerical return data with no firm identifying information and no date information, thereby mitigating potential look-ahead bias from the model's pretraining corpus.[7]

To quantify the degree of extrapolation, we estimate the following regression:

$$F_{i,t}\, r_{i,t+1} \;=\; \alpha_i \;+\; \delta_t \;+\; \sum_{s=0}^{n} \beta_s\, r_{i,t-s} \;+\; \varepsilon_{i,t}, \tag{5}$$

where $F_{i,t}\, r_{i,t+1}$ denotes the LLM's forecast of stock i's return in month $t+1$, formed at the end of month t, and $r_{i,t-s}$ is the realized return s months prior. A positive coefficient β_s indicates that the model extrapolates from past returns when forming its forecast.

Table V reports the results. The test set uses a data sample from January 2016 to December 2024. Consistent with the findings of Chen, Green, Gulen, and Zhou (2024), the LLM exhibits extrapolation from recent past returns when forming beliefs about future stock returns. Column (1) estimates Equation (5), regressing the LLM's forecast on trailing monthly returns with firm and month fixed effects. All coefficients on lagged returns are positive and statistically significant at the 1% level, confirming that the model systematically extrapolates from past performance. The coefficient on the most recent return $r_{i,t}$ is 0.394 ($t = 53.92$), and the coefficients generally decline with lag length, falling to 0.111 at lag one, 0.035 at lag two, and 0.025 at lag eleven. The estimated coefficients indicate that the model places excessive weight on recent performance, replicating the central pattern documented in human subjects by Da, Huang, and Jin (2021).

Training dataset and validation dataset. As in the AR(1) exercise, we curate separate training and validation datasets to fine-tune the LLM toward a rational benchmark. We divide the sample into three non-overlapping periods following the convention in Gu,

[7] A growing body of work has examined the potential for lookahead bias in LLM-based predictions (e.g., Glasserman and Lin, 2023; Sarkar and Vafa, 2024; Gao, Jiang, and Yan, 2025; Lopez-Lira, Tang, and Zhu, 2025; Ludwig, Mullainathan, and Rambachan, 2025). Mitigation strategies include entity-neutering prompting (Glasserman and Lin, 2023; Engelberg, Manela, Mullins, and Vulicevic, 2025) and training models under controlled information sets (e.g., Sarkar, 2024; He, Lv, Manela, and Wu, 2025a,b; Yan et al., 2026). We adopt the former approach, masking firm names and dates to limit data contamination. Moreover, if residual lookahead bias were present, the LLM would plausibly generate forecasts that earn positive returns rather than exhibit the extrapolative patterns we document.

Kelly, and Xiu (2020): a training sample spanning January 2001 through December 2011, a validation sample spanning January 2012 through December 2015, and the test sample spanning January 2016 through December 2024 on which the baseline results above are estimated. The training dataset contains 59,846 stock-month observations. As before, the validation dataset is used to monitor out-of-sample performance during the fine-tuning process and to guard against overfitting.

Unlike the AR(1) setting, where we set learning targets to conditional expectations to avoid mechanical bias in the diagnostic regression, here we define the learning target for each observation to be the realized next-month return $r_{i,t+1}$. The distinction reflects a difference in the downstream objective. In the AR(1) exercise, the goal of fine-tuning is to eliminate a specific bias in the error-revision regression, and training on realizations would introduce a mechanical negative correlation between the forecast error and the forecast revision. In the stock return setting, the goal of fine-tuning is to discipline the model's forecasts with realized market outcomes so that the model learns to form predictions that are informed by the historical joint distribution of past and future returns.

Training on realized returns also has a practical advantage: in the cross-section of equities, the conditional expected return is not directly observable, whereas realized returns are available. By exposing the model to the empirical distribution of monthly returns conditional on trailing twelve-month histories, the fine-tuning procedure allows the model to internalize the weak mean-reverting tendency in short-horizon stock returns, thereby attenuating the extrapolative bias that the LLM inherits from its training data.

Fine-tuned LLM corrects the bias. The preceding analysis establishes that the pretrained LLM extrapolates from recent past returns when forming beliefs about future stock returns, replicating the central finding of Da, Huang, and Jin (2021) and Chen, Green, Gulen, and Zhou (2024). We now examine whether fine-tuning on realized return data can eliminate this bias. We fine-tune Qwen3-32B on the training dataset described above and evaluate the resulting model on the same out-of-sample test period spanning January 2016 through December 2024.

Table VI presents the results. Column (1) re-estimates Equation (5), regressing the fine-tuned model's forecasts on lagged returns with firm and month fixed effects. The extrapolation bias documented in the baseline specification is eliminated. The estimated coef-

ficients on lagged returns are uniformly negative, indicating that the fine-tuned model has internalized the weak mean-reverting tendency in short-horizon stock returns rather than extrapolating from recent performance. The coefficient on the contemporaneous return $r_{i,t}$ is -0.120 ($t = -23.21$), and the magnitudes generally decrease with lag length, falling to -0.005 ($t = -1.56$) at lag eleven. This pattern is consistent with the model having learned from the training data that stocks with recent positive returns tend to experience subsequent reversals. Taken together, these results indicate that the extrapolation bias exhibited by the pretrained LLM is not a hard-wired feature of the model architecture. Fine-tuning on realized returns corrects the bias and produces forecasts that reflect the empirical return-generating process.

4 Conclusion

As AI agents built on LLMs move from narrow decision support toward autonomous delegation at scale, the behavioral biases encoded in their underlying models become economically consequential. We document that LLMs exhibit systematic extrapolation bias in both controlled forecasting environments and the cross-section of stock returns, replicating patterns documented in human subjects. Prompt-level interventions leave these biases largely intact, consistent with the view that the distortion is embedded in the model's internal representations rather than in the surface-level framing of the input. By intervening directly at the parameter level through supervised fine-tuning on rational benchmark forecasts, we eliminate overreaction to recent information in AR(1) forecasting tasks and remove the extrapolative loading on lagged returns in cross-sectional stock return predictions, with both corrections holding strictly out of sample. The fine-tuning procedure relies on Low-Rank Adaptation (LoRA), which preserves the model's general language understanding while remaining computationally feasible. More broadly, our approach offers a low-cost and generalizable methodology for aligning LLM behavior with rational benchmarks across economic settings, a prerequisite for the responsible deployment of AI agents in financial decision-making.

References

J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

H. Afrouzi, S. Y. Kwon, A. Landier, Y. Ma, and D. Thesmar. Overreaction in expectations: Evidence and theory. *The Quarterly Journal of Economics*, 138(3):1713–1764, 2023.

J. Bai, S. Bai, Y. Chu, Z. Cui, K. Dang, X. Deng, Y. Fan, W. Ge, Y. Han, F. Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

P. Bini, L. W. Cong, X. Huang, and L. J. Jin. Behavioral economics of AI: LLM biases and corrections. *Available at SSRN 5213130*, 2025.

T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

J. L. Bybee. The ghost in the machine: Generating beliefs with large language models. *arXiv preprint arXiv:2305.02823*, 2023.

S. Cao, W. Jiang, B. Yang, and A. L. Zhang. How to talk when a machine is listening: Corporate disclosure in the age of ai. *The Review of Financial Studies*, 36(9):3603–3642, 2023.

S. Cao, W. Jiang, J. Wang, and B. Yang. From man vs. machine to man+ machine: The art and ai of stock analyses. *Journal of Financial Economics*, 160:103910, 2024.

H. Chen, A. Didisheim, and L. Somoza. Out of the black box: Uncertainty quantification for llms via conditional probabilities. *Available at SSRN*, 2024.

H. Chen, A. Didisheim, L. Somoza, and H. Tian. A financial brain scan of the llm. *arXiv preprint arXiv:2508.21285*, 2025.

M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. D. O. Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

S. Chen, T. C. Green, H. Gulen, and D. Zhou. What does chatgpt make of historical stock returns? extrapolation and miscalibration in llm stock return forecasts. *arXiv preprint arXiv:2409.11540*, 2024.

Y. Chen, B. T. Kelly, and D. Xiu. Expected returns and large language models. *Available at SSRN 4416687*, 2022.

Z. Chen and D. Pu. Autonomous market intelligence: Agentic ai nowcasting predicts stock returns. *arXiv preprint arXiv:2601.11958*, 2026.

A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, et al. Palm: Scaling language modeling with pathways. *Journal of machine learning research*, 24(240):1–113, 2023.

Z. Da, X. Huang, and L. J. Jin. Extrapolative beliefs in the cross-section: What can we learn from the crowds? *Journal of Financial Economics*, 140(1):175–196, 2021.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186, 2019.

F. D'Acunto, N. Prabhala, and A. G. Rossi. The promises and pitfalls of robo-advising. *The Review of Financial Studies*, 32(5):1983–2020, 2019.

J. Engelberg, A. Manela, W. Mullins, and L. Vulicevic. Entity neutering. *Available at SSRN*, 2025.

A. Fedyk, A. Kakhbod, P. Li, and U. Malmendier. AI and perception biases in investments: An experimental study. *Available at SSRN*, 4787249, 2024.

I. O. Gallegos, R. A. Rossi, J. Barrow, M. M. Tanjim, S. Kim, F. Dernoncourt, T. Yu, R. Zhang, and N. K. Ahmed. Bias and fairness in large language models: A survey. *Computational linguistics*, 50(3):1097–1179, 2024.

Z. Gao, W. Xiong, and J. Yuan. Structured beliefs and fund performance: An LLM-based approach. *Available at SSRN*, 2024.

Z. Gao, W. Jiang, and Y. Yan. A test of lookahead bias in LLM forecasts. *arXiv preprint arXiv:2512.23847*, 2025.

P. Glasserman and C. Lin. Assessing look-ahead bias in stock return predictions generated by gpt sentiment analysis. *arXiv preprint arXiv:2309.17322*, 2023.

I. J. Goodfellow, M. Mirza, D. Xiao, A. Courville, and Y. Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211*, 2013.

S. Gu, B. Kelly, and D. Xiu. Empirical asset pricing via machine learning. *The Review of Financial Studies*, 33(5):2223–2273, 2020.

A. L. Hansen and S. Kazinnik. Can chatgpt decipher fedspeak? *Available at SSRN 4399406*, 2024.

S. He, L. Lv, A. Manela, and J. Wu. Chronologically consistent large language models. *arXiv preprint arXiv:2502.21206*, 2025a.

S. He, L. Lv, A. Manela, and J. Wu. Instruction tuning chronologically consistent language models. *arXiv preprint arXiv:2510.11677*, 2025b.

J. J. Horton. Large language models as simulated economic agents: What can we learn from homo silicus? Technical report, National Bureau of Economic Research, 2023.

E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, et al. LoRA: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

A. H. Huang, H. Wang, and Y. Yang. Finbert: A large language model for extracting information from financial text. *Contemporary Accounting Research*, 40(2):806–841, 2023.

M. Jha, J. Qian, M. Weber, and B. Yang. Chatgpt and corporate policies. *arXiv preprint arXiv:2409.17933*, 2024.

S. Ke. Analysts' belief formation in their own words. *Available at SSRN 5025830*, 2024.

W. Kwon, Z. Li, S. Zhuang, Y. Sheng, L. Zheng, C. H. Yu, J. E. Gonzalez, H. Zhang, and I. Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

T. Lieberum, S. Rajamanoharan, A. Conmy, L. Smith, N. Sonnerat, V. Varma, J. Kramár, A. Dragan, R. Shah, and N. Nanda. Gemma scope: Open sparse autoencoders everywhere all at once on gemma 2. In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 278–300, 2024.

A. Lopez-Lira. Can large language models trade? testing financial theories with llm agents in market simulations. *arXiv preprint arXiv:2504.10789*, 2025.

A. Lopez-Lira and Y. Tang. Can chatgpt forecast stock price movements? return predictability and large language models. *arXiv preprint arXiv:2304.07619*, 2023.

A. Lopez-Lira, Y. Tang, and M. Zhu. The memorization problem: Can we trust LLMs' economic forecasts? *arXiv preprint arXiv:2504.14765*, 2025.

J. Ludwig, S. Mullainathan, and A. Rambachan. Large language models: An applied econometric framework. Technical report, National Bureau of Economic Research, 2025.

S. Mangrulkar, S. Gugger, L. Debut, Y. Belkada, S. Paul, B. Bossan, and M. Tietz. PEFT: State-of-the-art parameter-efficient fine-tuning methods. https://github.com/huggingface/peft, 2022.

OpenAI. Chatgpt plugins. https://openai.com/blog/chatgpt-plugins, 2023.

L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

J. S. Park, J. O'Brien, C. J. Cai, M. R. Morris, P. Liang, and M. S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pages 1–22, 2023.

R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

J. Ross, Y. Kim, and A. Lo. LLM economicus? mapping the behavioral biases of LLMs via utility theory. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Rx3wC8sCTJ.

S. K. Sarkar. Storieslm: A family of language models with time-indexed training data. *Available at SSRN 4881024*, 2024.

S. K. Sarkar and K. Vafa. Lookahead bias in pretrained language models. *Available at SSRN 4754678*, 2024.

E. Schluntz and B. Zhang. Building effective agents, Dec. 2024. URL https://www.anthropic.com/engineering/building-effective-agents. Anthropic.

Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, X. Bi, H. Zhang, M. Zhang, Y. Li, Y. Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

G. Wang, Y. Xie, Y. Jiang, A. Mandlekar, C. Xiao, Y. Zhu, L. Fan, and A. Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023a.

Y. Wang, Y. Kordi, S. Mishra, A. Liu, N. A. Smith, D. Khashabi, and H. Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 13484–13508, 2023b.

T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. Le Scao, S. Gugger, M. Drame, Q. Lhoest, and A. Rush. Transformers: State-of-the-art natural language processing. In Q. Liu and D. Schlangen, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, Oct. 2020. Association for Computational Linguistics. doi:

10.18653/v1/2020.emnlp-demos.6. URL https://aclanthology.org/2020.emnlp-demos.6/.

J. C. Wu, J. Xi, and S. Xie. LLM survey framework: Coverage, reasoning, dynamics, identification. Technical report, National Bureau of Economic Research, 2025.

Y. Yan, R. Tang, Z. Gao, W. Jiang, and Y. Lu. DatedGPT: Preventing lookahead bias in large language models with time-aware pretraining. *arXiv preprint arXiv:2603.11838*, 2026.

A. Yang, A. Li, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Gao, C. Huang, C. Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. R. Narasimhan, and Y. Cao. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*, 2022.

Figure I. Pre-training and Instructional Tuning Pipeline

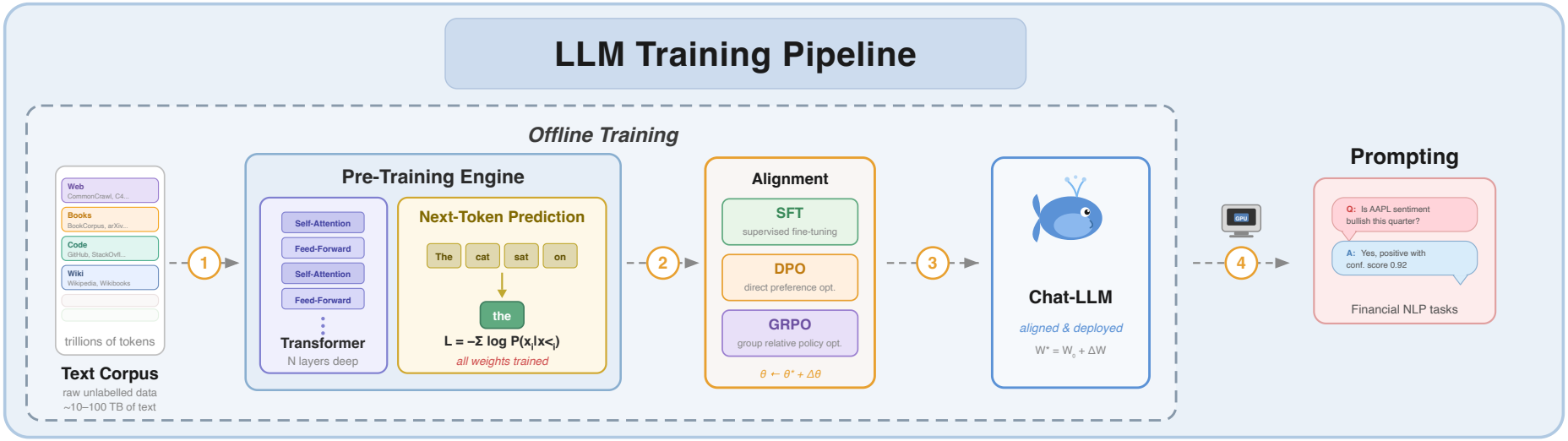


Figure II. Supervised Fine-Tuning (SFT) Pipeline

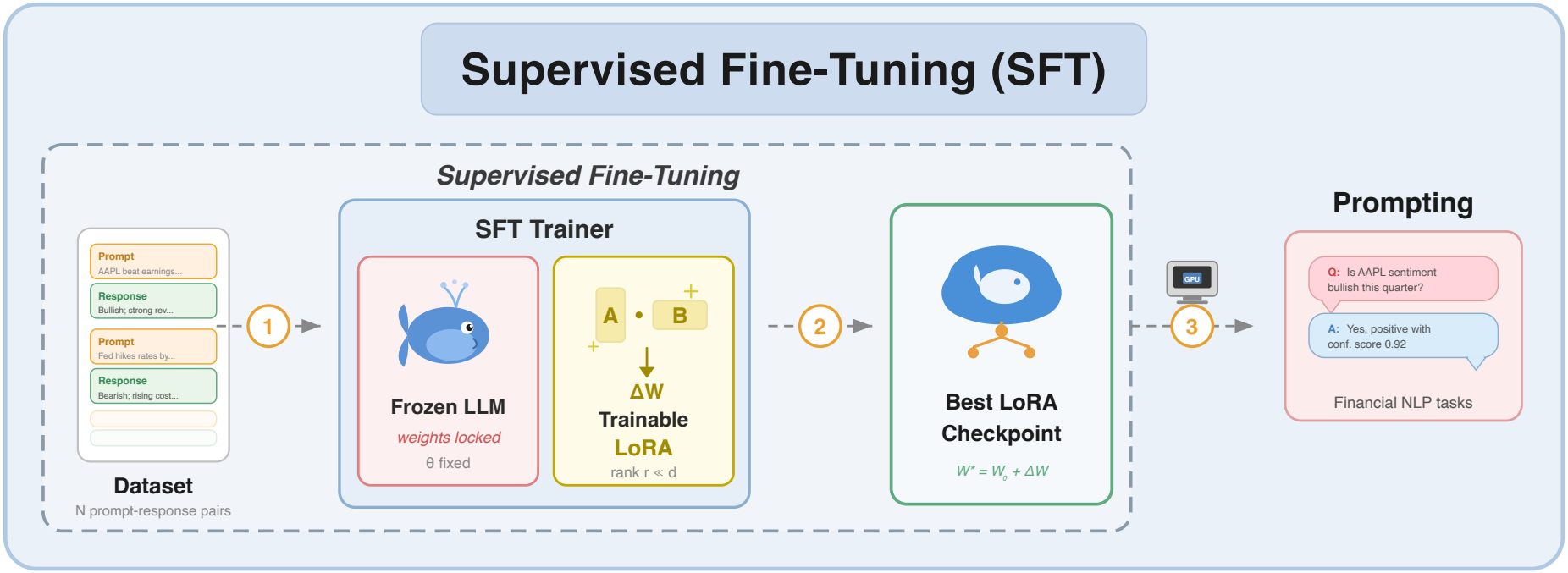


Figure III. LoRA Reparametrization

This figure from Hu et al. (2022) illustrates the LoRA reparametrization. The pretrained weights (left) are frozen during fine-tuning. A small pair of trainable matrices (right, in orange) is attached alongside the frozen weights to capture the task-specific adjustment. Only these added parameters are updated during training, typically less than 1% of the full model, making fine-tuning feasible.

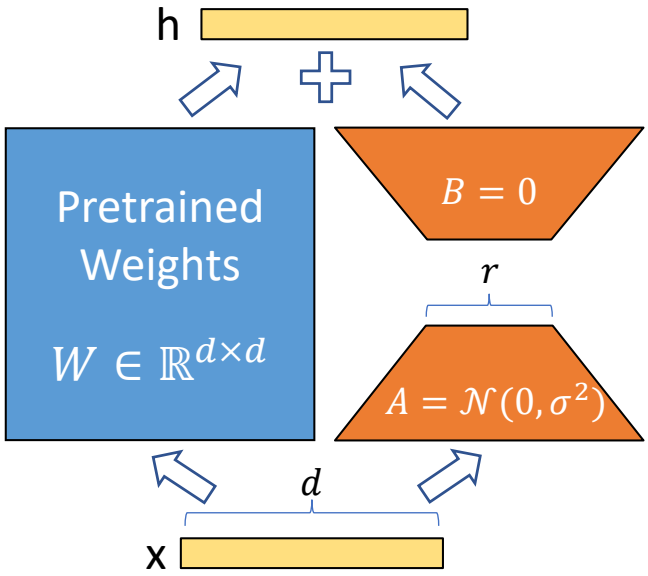


Figure IV. Debiasing Framework

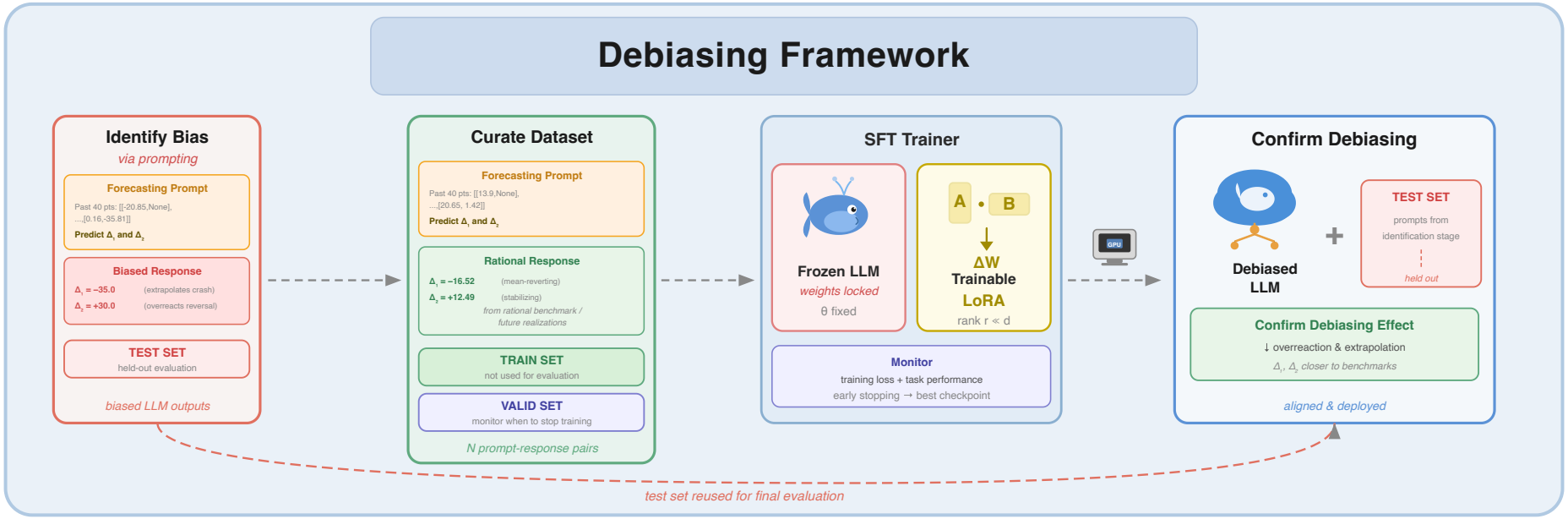


Figure V. Example Prompt and Response for AR(1) Exercise

Example Prompt:

You are an expert forecaster in an economic forecasting experiment. Your final score will be divided by 600 and paid in real U.S. dollars. Maximize your score by making accurate predictions. Errors are penalized by squared error (large mistakes hurt more).

Data: You observe the past 40 data points as `[value, change from previous]`: `[[`x_{t-39}`, null], [`x_{t-38}`,` $x_{t-38} - x_{t-39}$`], ..., [`x_t`,` $x_t - x_{t-1}$`]]`. The last observed value is $x_t = -7.73$.

Definitions: $\Delta_{t+1} = x_{t+1} - x_t$ (change from last observed to first prediction); $\Delta_{t+2} = x_{t+2} - x_{t+1}$ (change from first prediction to second prediction).

Task: (1) **Step 1:** Predict `step_one_delta`, where $x_{t+1} = -7.73 +$ `step_one_delta`. Explain briefly. (2) **Step 2:** Using the same x_{t+1}, predict `step_two_delta`, where $x_{t+2} = x_{t+1} +$ `step_two_delta`. Explain briefly.

Output format (strict; valid JSON only):

```
{ "step_one_delta": float,
  "step_one_explanation": "string (≤100 words)",
  "step_two_delta": float,
  "step_two_explanation": "string (≤100 words)" }
```

Input: `Past 40 data points: [[21.53, None], ..., [−0.97, 34.3], [−7.73, −6.76]]`

Example Response (Qwen3-32B):

```
{ "step_one_delta": 34.3,
  "step_one_explanation": "The recent trend shows large fluctuations,
  and the last change was -6.76. Predicting a reversal with a significant
  positive delta of 34.3 aligns with the historical volatility.",
  "step_two_delta": -34.3,
  "step_two_explanation": "Given the high volatility and the large
  positive step_one_delta, a symmetric negative delta of -34.3 is
  predicted to balance the trend and reflect potential mean reversion." }
```

Example Response (Qwen3-32B-Fine-tuned):

```
{ "step_one_delta": 4.64,
  "step_one_explanation": "The time series shows a recent downward
  trend, but the last few values suggest a potential reversal. I predict
  a moderate recovery, so x_{t+1} will increase by 4.64.",
  "step_two_delta": -1.85,
  "step_two_explanation": "The recent upward movement is likely to
  moderate, with a small decline expected in the next step. I predict
  a decrease of 1.85 for x_{t+2}." }
```

Figure VI. Example Prompt and Responses for Stock Return Prediction

Example Prompt:

You will receive one JSON object with:
- x: a 12×11 matrix, ordered oldest → newest, where each row represents one calendar month from the past 12 months.
If a row is an empty list, it could be due to trading halts, etc.
Each row contains the following variables (in order):
`[stock_return, market_value_weighted_return, market_equal_weighted_return, close_price, bid_price, ask_price, monthly_low_bid, monthly_high_ask, share_volume, shares_outstanding, stock_return_ex_dividends]`
Note: `volume` is a raw integer.

Task:

1. Predict the sign and strength of the next-month return as a float (`prediction`).
2. Provide a short `explanation` of at most 100 words, referencing patterns in `x` such as momentum, reversal, market correlation, liquidity (bid-ask spread), volume trends, or intramonth price moves, using **only** the variables listed above.

Output format (STRICT; JSON only, no markdown, no thinking blocks):
`{"prediction": float, "explanation": "string"}`

INPUT x:
```
[[−0.021, −0.018, −0.047, 62.49, 62.47, 62.48, 62.16, 64.59, 357184,
505664, −0.021],
 ...,
 [−0.118, −0.091, −0.1, 56.23, 56.29, 56.32, 56.23, 69.18, 614243,
498860, −0.126],
 [−0.003, 0.077, 0.05, 56.07, 56.08, 56.09, 53.48, 59.81, 655444, 498860,
−0.003]]
```

Example Response (Qwen3-32B):

```
{ "prediction": −0.05,
  "explanation": "The stock has shown a recent downward trend with
  negative returns in the last three months, including a sharp −12%
  drop. Momentum appears weak, and the spread between bid and ask
  prices has widened, indicating reduced liquidity. High volume in the
  most recent month suggests increased selling pressure, reinforcing a
  potential continuation of the downward move." }
```

Example Response (Qwen3-32B-Fine-tuned):

```
{ "prediction": 0.06,
  "explanation": "The stock has shown a reversal pattern after a sharp
  decline in the previous month, with a positive market move and a
  narrowing bid-ask spread. Volume has increased, indicating improved
  liquidity. The stock's return is expected to continue in the positive
  direction, albeit at a moderate pace." }
```

Figure VII. AR(1) Exercise: Base Model

This figure plots the estimated overreaction coefficient b from the error-revision regression $x_{t+1} - F_{i,t}\, x_{t+1} = a + b\,(F_{i,t}\, x_{t+1} - F_{i,t-1}\, x_{t+1}) + v_{i,t}$ for each persistence parameter $\rho \in \{0, 0.2, 0.4, 0.6, 0.8, 1\}$, using forecasts from the base model (Qwen3-32B). Error bars denote the 95% confidence interval.

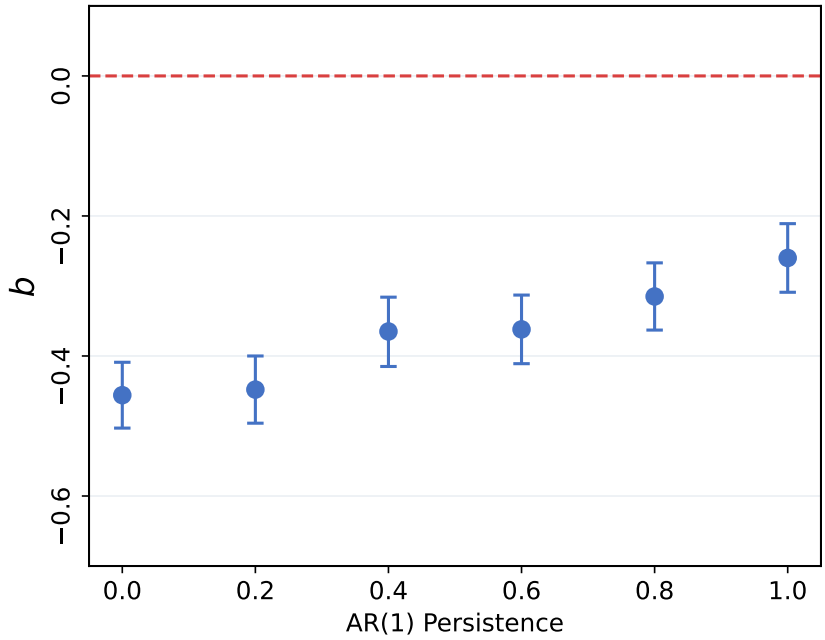


Figure VIII. AR(1) Exercise: Fine-tuned LLM

This figure plots the estimated overreaction coefficient b from the error-revision regression $x_{t+1} - F_{i,t}\, x_{t+1} = a + b\,(F_{i,t}\, x_{t+1} - F_{i,t-1}\, x_{t+1}) + v_{i,t}$ for each persistence parameter $\rho \in \{0, 0.2, 0.4, 0.6, 0.8, 1\}$, using forecasts from the fine-tuned Qwen3-32B. Error bars denote the 95% confidence interval.

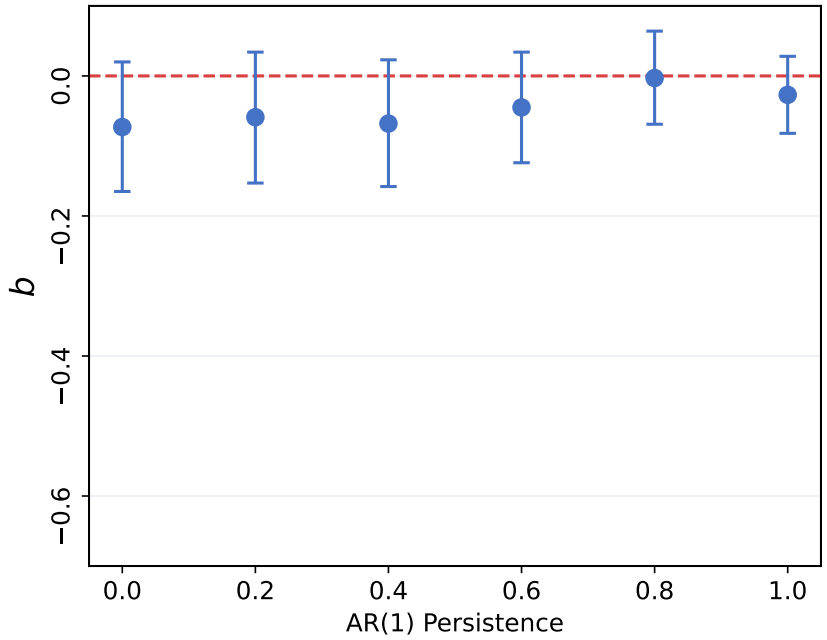


Table I. Descriptive Statistics: AR(1) Exercise

This table presents descriptive statistics for the predicted values in the AR(1) exercise. $F_{i,t}\, x_{t+1}$ and $F_{i,t-1}\, x_{t+1}$ denote the one-period-ahead and two-period-ahead predictions, respectively. Statistics are reported separately for the base model (Qwen3-32B) and the fine-tuned model across each value of ρ. Each cell contains 1,248 observations.

ρ	Variable	Mean	SD	P25	Median	P75
Panel A: Base Model (Qwen3-32B)						
0.0	$F_{i,t}\, x_{t+1}$	−0.03	18.89	−10.45	2.15	10.52
	$F_{i,t-1}\, x_{t+1}$	2.99	21.22	−10.94	2.70	17.79
0.2	$F_{i,t}\, x_{t+1}$	−0.62	21.05	−13.07	2.37	11.92
	$F_{i,t-1}\, x_{t+1}$	2.60	21.71	−12.54	2.80	17.69
0.4	$F_{i,t}\, x_{t+1}$	0.07	21.30	−12.95	2.15	12.93
	$F_{i,t-1}\, x_{t+1}$	2.01	23.00	−14.10	2.39	18.02
0.6	$F_{i,t}\, x_{t+1}$	0.98	24.88	−14.87	3.03	16.38
	$F_{i,t-1}\, x_{t+1}$	1.69	25.56	−15.73	2.65	19.30
0.8	$F_{i,t}\, x_{t+1}$	0.87	34.53	−23.45	3.51	22.67
	$F_{i,t-1}\, x_{t+1}$	1.13	35.20	−23.96	2.63	25.78
1.0	$F_{i,t}\, x_{t+1}$	−31.63	164.32	−132.41	−43.38	38.45
	$F_{i,t-1}\, x_{t+1}$	−33.96	164.34	−131.97	−46.24	38.74
Panel B: Fine-tuned Model						
0.0	$F_{i,t}\, x_{t+1}$	0.17	3.99	−0.01	0.00	0.77
	$F_{i,t-1}\, x_{t+1}$	6.02	12.08	−1.29	10.13	14.86
0.2	$F_{i,t}\, x_{t+1}$	0.05	6.14	−3.07	0.00	3.06
	$F_{i,t-1}\, x_{t+1}$	6.19	11.79	−1.76	7.01	14.90
0.4	$F_{i,t}\, x_{t+1}$	0.03	9.52	−5.13	0.00	5.21
	$F_{i,t-1}\, x_{t+1}$	5.97	12.18	−2.63	4.28	15.03
0.6	$F_{i,t}\, x_{t+1}$	−0.03	14.79	−9.66	0.09	9.17
	$F_{i,t-1}\, x_{t+1}$	4.23	15.97	−7.22	1.78	14.82
0.8	$F_{i,t}\, x_{t+1}$	−0.10	25.85	−17.78	0.80	16.82
	$F_{i,t-1}\, x_{t+1}$	3.06	26.49	−15.04	1.19	19.80
1.0	$F_{i,t}\, x_{t+1}$	−32.55	157.08	−124.09	−41.54	32.07
	$F_{i,t-1}\, x_{t+1}$	−27.95	157.10	−116.35	−38.00	34.58

Table II. Descriptive Statistics: Stock Return Prediction

This table presents descriptive statistics for key variables used in the stock return prediction analysis. Statistics reported include mean, standard deviation (SD), percentiles (P25, Median, P75), and the number of observations (N).

Variable	Mean	SD	P25	Median	P75	N
r_{t+1}	0.011	0.089	−0.038	0.011	0.060	51,724
$F_{i,t}\, r_{i,t+1}$ (Base)	0.011	0.045	−0.020	0.020	0.036	51,724
$F_{i,t}\, r_{i,t+1}$ (Fine-tuned)	0.029	0.022	0.010	0.020	0.030	51,724

Table III. AR(1) Exercise: Base Model

This table reports estimates of the error-revision regression $x_{t+1} - F_{i,t}\, x_{t+1} = a + b\,(F_{i,t}\, x_{t+1} - F_{i,t-1}\, x_{t+1}) + v_{i,t}$ using forecasts from the base model (Qwen3-32B). Each column corresponds to a different persistence parameter ρ. A negative coefficient b indicates overreaction. t-statistics are reported in parentheses. ***, **, and * denote significance at the 1%, 5%, and 10% levels, respectively.

	(1)	(2)	(3)	(4)	(5)	(6)
	$\rho = 0.0$	$\rho = 0.2$	$\rho = 0.4$	$\rho = 0.6$	$\rho = 0.8$	$\rho = 1.0$
$F_{i,t}\, x_{t+1} - F_{i,t-1}\, x_{t+1}$	−0.456***	−0.448***	−0.365***	−0.362***	−0.315***	−0.260***
	(−19.05)	(−18.38)	(−14.35)	(−14.52)	(−12.88)	(−10.37)
R^2	0.226	0.213	0.142	0.145	0.118	0.079
N	1,248	1,248	1,248	1,248	1,248	1,248

Table IV. AR(1) Exercise: Fine-tuned LLM

This table reports estimates of the error-revision regression $x_{t+1} - F_{i,t}\, x_{t+1} = a + b\,(F_{i,t}\, x_{t+1} - F_{i,t-1}\, x_{t+1}) + v_{i,t}$ using forecasts from the fine-tuned Qwen3-32B. Each column corresponds to a different persistence parameter ρ. A negative coefficient b indicates overreaction. t-statistics are reported in parentheses. ***, **, and * denote significance at the 1%, 5%, and 10% levels, respectively.

	(1)	(2)	(3)	(4)	(5)	(6)
	$\rho = 0.0$	$\rho = 0.2$	$\rho = 0.4$	$\rho = 0.6$	$\rho = 0.8$	$\rho = 1.0$
$F_{i,t}\, x_{t+1} - F_{i,t-1}\, x_{t+1}$	−0.073	−0.059	−0.068	−0.045	−0.003	−0.027
	(−1.54)	(−1.24)	(−1.47)	(−1.12)	(−0.08)	(−0.97)
R^2	0.002	0.001	0.002	0.001	0.000	0.001
N	1,248	1,248	1,248	1,248	1,248	1,248

Table V. Stock Return Extrapolation: Base Model

This table examines extrapolation in the base model (Qwen3-32B) stock return forecasts. The sample covers S&P 500 constituents from January 2016 through December 2024. Column (1) regresses the LLM's forecast $F_{i,t}\, r_{i,t+1}$ on lagged returns with firm and month fixed effects. Column (2) regresses realized returns on lagged returns. Column (3) regresses realized returns on the LLM's forecast with firm and month fixed effects. Column (4) reports Fama–MacBeth estimates. Standard errors are double-clustered by stock and month in columns (1)–(3). t-statistics are reported in parentheses. ***, **, and * denote significance at the 1%, 5%, and 10% levels, respectively.

	(1)	(2)	(3)	(4)
	$F_{i,t}\, r_{i,t+1}$	$r_{i,t+1}$	$r_{i,t+1}$	$r_{i,t+1}$
$F_{i,t}\, r_{i,t+1}$			-0.074^{**}	-0.004
			(-2.05)	(-0.16)
$r_{i,t}$	0.394^{***}	-0.048^{*}		
	(53.92)	(-1.85)		
$r_{i,t-1}$	0.111^{***}	-0.020		
	(27.31)	(-1.14)		
$r_{i,t-2}$	0.035^{***}	-0.026		
	(12.98)	(-1.16)		
$r_{i,t-3}$	0.038^{***}	-0.017		
	(13.16)	(-0.94)		
$r_{i,t-5}$	0.027^{***}	-0.009		
	(11.73)	(-0.50)		
$r_{i,t-7}$	0.018^{***}	-0.038		
	(6.88)	(-1.50)		
$r_{i,t-9}$	0.024^{***}	-0.008		
	(7.75)	(-0.39)		
$r_{i,t-11}$	0.025^{***}	-0.019		
	(7.17)	(-1.28)		
Controls	Yes	Yes	No	No
Firm FE	Yes	Yes	Yes	No
Month FE	Yes	Yes	Yes	No
Estimation	Panel	Panel	Panel	FM
Within R^2	0.632	0.007	0.001	0.024
N	51,724	51,724	51,724	51,724

Table VI. Stock Return Extrapolation: Fine-tuned LLM

This table examines extrapolation in the fine-tuned Qwen3-32B's stock return forecasts. The sample covers S&P 500 constituents from January 2016 through December 2024 . The model is fine-tuned on realized returns from January 2001 through December 2011, with January 2012 through December 2015 used for validation. Column (1) regresses the fine-tuned LLM's forecast $F_{i,t}\, r_{i,t+1}$ on lagged returns with firm and month fixed effects. Column (2) regresses realized returns on the fine-tuned LLM's forecast with firm and month fixed effects. Column (3) reports Fama–MacBeth estimates. Standard errors are double-clustered by stock and month in columns (1)–(2). t-statistics are reported in parentheses. ***, **, and * denote significance at the 1%, 5%, and 10% levels, respectively.

	(1)	(2)	(3)
	$F_{i,t}\, r_{i,t+1}$	$r_{i,t+1}$	$r_{i,t+1}$
$F_{i,t}\, r_{i,t+1}$		0.136**	0.016
		(2.26)	(0.31)
$r_{i,t}$	−0.120***		
	(−23.21)		
$r_{i,t-1}$	−0.062***		
	(−20.37)		
$r_{i,t-2}$	−0.037***		
	(−16.18)		
$r_{i,t-3}$	−0.022***		
	(−7.40)		
$r_{i,t-5}$	−0.011***		
	(−5.50)		
$r_{i,t-7}$	−0.012***		
	(−6.41)		
$r_{i,t-9}$	−0.002		
	(−0.99)		
$r_{i,t-11}$	−0.005		
	(−1.56)		
Controls	Yes	No	No
Firm FE	Yes	Yes	No
Month FE	Yes	Yes	No
Estimation	Panel	Panel	FM
Within R^2	0.301	0.001	0.027
N	51,724	51,724	51,724

A Online Appendix

Table A.1. Top Dense Text Generation Models on Hugging Face as of March 2026

Model	Total Params	Active Params	Downloads
Qwen/Qwen2.5-7B-Instruct	8B	8B	20.9M
Qwen/Qwen3-8B	8B	8B	8.97M
meta-llama/Llama-3.1-8B-Instruct	8B	8B	7.62M
Qwen/Qwen3-32B	32B	32B	4.81M
dphn/dolphin-2.9.1-yi-1.5-34b	34B	34B	4.61M