

Scaling Point-in-Time Language Models

Enabling Economic Evaluation of Embeddings

Bryan Kelly¹ Semyon Malamud² Johannes Schwab² Teng Andrea Xu³

¹Yale School of Management

²Swiss Finance Institute, EPFL

³AQR Capital Management

NeurIPS 2026 · Evaluations & Datasets Track

1

Introduction

2

Training

3

Results I: language benchmarks

4

Setup: the economic test

5

Results II: economic value

6

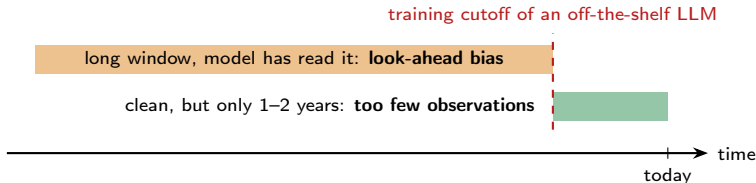
Conclusion

Why we built this: look-ahead bias or a short test window

LLMs are now a standard tool in empirical finance and economics. Every study that uses one to make predictions about the past faces the same choice:

A. test on past data
Glasserman and Lin, 2023; Sarkar and Vafa, 2024

B. test after the cutoff
e.g., Lopez-Lira and Tang, 2023



The dilemma

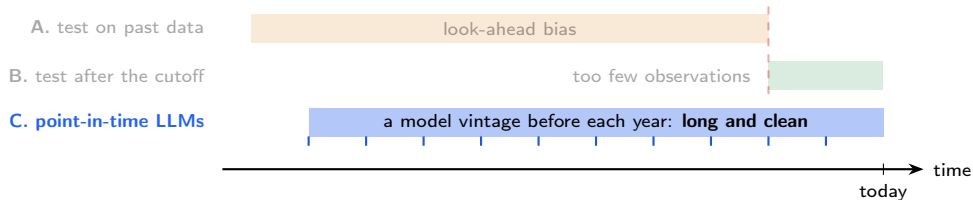
A gives biased results; B gives too few observations to tell skill from luck. Can we get a test window that is both *long* and *clean*?

The solution: point-in-time (PIT) LLMs

Definition

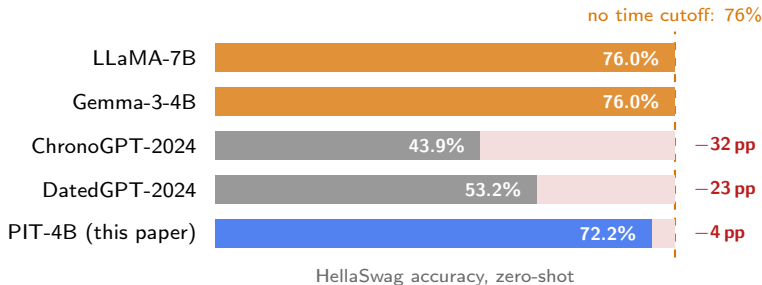
A PIT model with cutoff τ is trained *only* on text published on or before τ . To analyse a document from date t , use a model with $\tau < t$: no look-ahead, by construction.

Keep one model vintage per year, and every year of the test window is scored by a model that has never seen it:



The catch: are PIT models good enough?

Earlier PIT LLMs (He et al., 2025; Yan et al., 2026) show that the idea works, but they lag far behind models trained without a time cutoff. One reason is data: ChronoGPT repeats a 7B-token pre-2000 corpus about 10 times, whereas LLMs normally see each token about once.



Our answer

With scale, the gap to models trained without a time cutoff shrinks from 23–32 pp to about 4 pp.

This paper

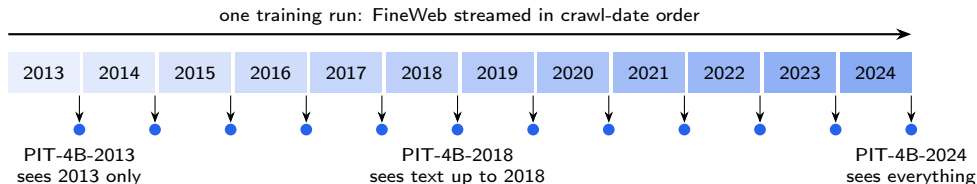
- 1 **Scale PIT pre-training.** A 4.2B-parameter decoder trained on 1T tokens of web text in chronological order, with one model vintage per year, 2013–2024.
- 2 **Keep post-training leakage-free.** LoRA instruction tuning on temporally filtered data, evaluated with IFEval (verifiable, no LLM judge).
- 3 **Evaluate embeddings economically.** Do PIT news embeddings produce profitable stock portfolios, strictly out of sample?
- 4 **Release everything.** Data pipeline, training and evaluation code, base and instruction-tuned checkpoints.

Preview of results

- PIT-4B: **72.2%** on HellaSwag (prior PIT models: 44–53%; LLaMA-7B: 76%).
- Leakage-free news embeddings: out-of-sample **Sharpe ratio 1.0–1.5** (prior PIT models: 0.3–0.7), on par with or above “leaky” full-sample versions.

- 1 Introduction
- 2 Training**
- 3 Results I: language benchmarks
- 4 Setup: the economic test
- 5 Results II: economic value
- 6 Conclusion

One chronological run gives a whole family of PIT models



- **Corpus:** FineWeb (Penedo et al., 2024), 15T filtered and deduplicated English tokens from 96 Common Crawl dumps (2013–2025). Every document carries the date of the dump it was crawled in.
- **Training:** stream the corpus in time order (continual pre-training) and snapshot the weights at dump boundaries. A snapshot at τ has only seen text crawled up to τ .
- **Vintages:** 12 year-end checkpoints, 2013–2024, from a single run instead of 12 separate ones. Dumps are irregular (2–12 per year), so the 2015 vintage is November.

Two model sizes, one recipe

	ChronoGPT	PIT-1.5B	PIT-4B
Layers	52	52	20
Hidden size d	1,536	1,536	4,096
Heads $\times$ head dim	12×128	12×128	32×128
Parameters	$\approx 1.5\text{B}$	$\approx 1.5\text{B}$	$\approx 4.2\text{B}$
Training tokens	71B + 65B	170B	1T
Epochs	$\approx 10^*$	1	1
Context length	1,792	2,048	2,048

* ChronoGPT's 1999 model makes ≈ 10 passes over a 7B-token pre-2000 corpus (71B tokens); yearly updates 2000–2024 then add a 65B-token corpus (He et al., 2025).

- GPT-2-style decoder on the modded-nanogpt speed-run code base (Muon optimizer, value residuals).
- PIT-4B: $\approx 10\text{k}$ tokens/s per GPU, $\approx 28,000$ GPU-hours on GH200 and H100 GPUs (CSCS Alps).
- **Mostly a data difference:** same architecture as ChronoGPT, but PIT-1.5B reads 170B tokens once; repeating a small corpus ≈ 10 times is non-standard, as repeated data loses value after ≈ 4 epochs (Muennighoff et al., 2023). HellaSwag: 43.9% $\rightarrow$ 64.3%.

Post-training without leaking the future

- Supervised instruction tuning only (no preference tuning) on Tülu 3 subsets (Lambert et al., 2024) and IFEval-style data, with **LoRA** ($r = 16$): $\approx 100\times$ fewer trainable weights per layer, and less forgetting.
- Problem: instruction data is full of dated facts.
- Fix: an LLM classifier (gpt-5-nano) labels every example *timeless* (keep) or *time-sensitive* (drop).
- PIT-4B-FT- τ is LoRA on top of PIT-4B- τ , so it keeps the cutoff τ .

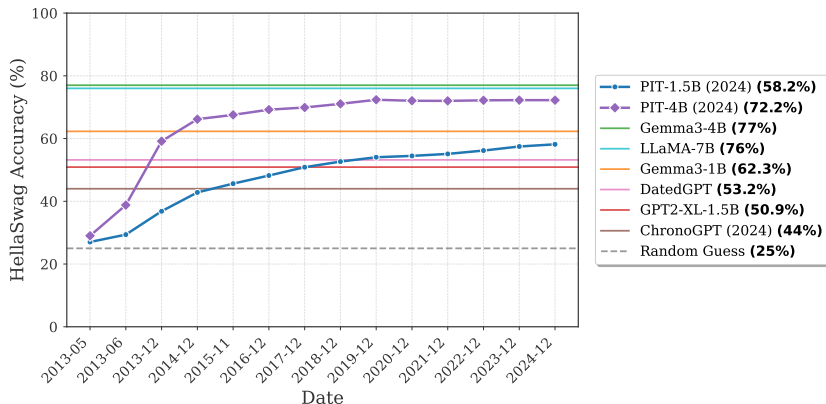
Classifier decisions (from its prompt)

"The 2024 Olympics were held in Paris" drop
"Tesla's market cap exceeded \$1 trillion in 2024" drop
"Solve: $2x + 3 = 7$ " keep

SFT data	examples	kept	share
Code (2 sets)	141.7k	134.9k	95%
Math (2 sets)	114.2k	113.7k	99.6%
Instruction following (2 sets)	486.1k	411.9k	85%
Total	742.1k	660.4k	89%

- 1 Introduction
- 2 Training
- 3 Results I: language benchmarks**
- 4 Setup: the economic test
- 5 Results II: economic value
- 6 Conclusion

Capability grows with every vintage



- Each point is a model that has only seen text up to that date (HellaSwag, zero-shot).
- PIT-4B reaches **72.2%**: +28 pp over ChronoGPT and +19 pp over DatedGPT, and within $\approx 4\text{--}5$ pp of LLaMA-7B and Gemma-3-4B.

Horizontal lines are published numbers (Gemma 3 reports 10-shot); the table on the next slide uses our own zero-shot evaluations.

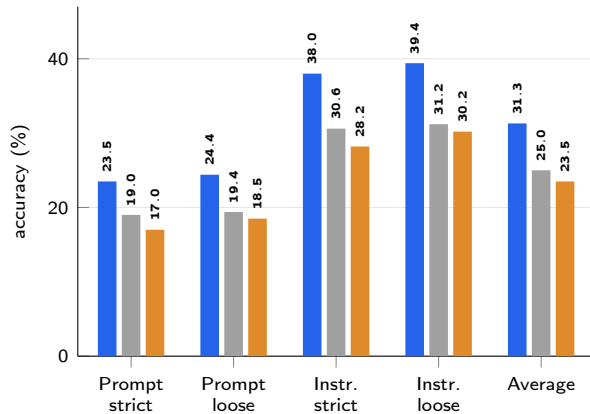
Zero-shot common-sense reasoning

Model	BoolQ	PIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA	Avg.
<i>point-in-time models</i>								
DatedGPT-2024	–	70.5	53.2	–	52.0	34.7	–	(52.6) [†]
ChronoGPT-2024	60.4	66.5	43.9	54.9	52.5	29.5	34.8	48.9
PIT-1.5B-2024 (ours)	61.9	76.1	64.3	59.4	49.5	30.4	34.8	53.8
PIT-4B-2024 (ours)	63.0	78.9	72.2	64.2	54.4	35.1	39.0	58.1
<i>trained without a time cutoff</i>								
Gemma-3-1B	66.4	74.8	62.0	58.9	72.2	38.3	37.0	58.5
Gemma-3-4B	79.0	80.0	76.0	69.5	81.8	54.9	43.0	69.2
LLaMA-7B	76.8	79.7	76.0	69.6	72.1	44.3	44.4	66.1
PIT-4B – LLaMA-7B	–13.8	–0.8	–3.8	–5.4	–17.7	–9.2	–5.4	–8.0

- The best point-in-time model on every task.
- Close to LLaMA-7B on PIQA and HellaSwag; clear gaps remain on knowledge-heavy QA (ARC-e, BoolQ).

[†]Numbers from Yan et al. (2026), which reports four of the seven tasks; PIT-4B averages 60.2 on those four.

Instruction following: IFEval



- We benchmark the fine-tuned (FT) models on **IFEval** (Zhou et al., 2023).
- Average accuracy:
 - PIT-4B-FT-2024 (ours): **31.3%**
 - ChronoGPT-Instruct-2024: 25.0%
 - Qwen1.5-1.8B (no cutoff): 23.5%

Greedy decoding, default harness settings.

Outline

- 1 Introduction
- 2 Training
- 3 Results I: language benchmarks
- 4 Setup: the economic test**
- 5 Results II: economic value
- 6 Conclusion

Finance basics: returns, portfolios, backtests

Return. $R_{t+1,s}$ is the percentage gain of stock s over the next month. Stacking all N stocks gives $R_{t+1} \in \mathbb{R}^N$.

Portfolio. A weight vector $\pi_t \in \mathbb{R}^N$ chosen at month t :

- $\pi_{t,s} > 0$: buy the stock (“long”);
- $\pi_{t,s} < 0$: borrow it and sell it (“short”), which pays off if the price falls.

Portfolio return. Just a dot product:

$$r_{t+1} = \pi_t^\top R_{t+1}.$$

Long–short. If $\sum_s \pi_{t,s} = 0$, the shorts pay for the longs, so the position costs nothing up front. It is a pure bet on *which stocks beat which*, not on whether the market goes up.

Out-of-sample backtest. Pick π_t with information up to t only, record r_{t+1} , move one month forward, repeat. The output is a time series of monthly returns $r_1, \dots, r_T$, which we then evaluate.

ML analogy

Choosing π_t is a cross-sectional prediction problem that is scored by the realized payoff $\pi_t^\top R_{t+1}$, not by squared error.

Finance basics: the Sharpe ratio

$$SR = \frac{\text{mean}(r)}{\text{sd}(r)}, \quad SR_{\text{annual}} = \sqrt{12} SR_{\text{monthly}}$$

- The **signal-to-noise ratio** of a strategy: average gain per unit of risk.
- **Leverage-proof**: doubling every position doubles both mean and risk, so SR is unchanged. It rates the signal, not the size of the bet.
- **Calibration**: U.S. stock market ≈ 0.4 – 0.5 in the long run.
- **Our downstream metric**: like linear-probe accuracy, but scored in money and strictly out of sample.

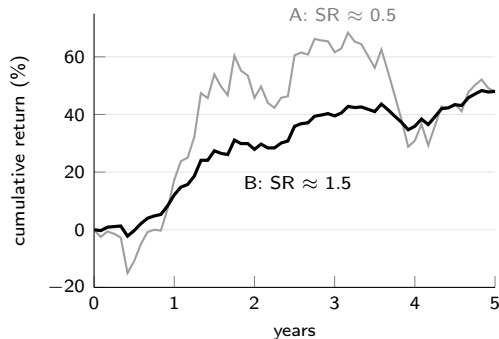
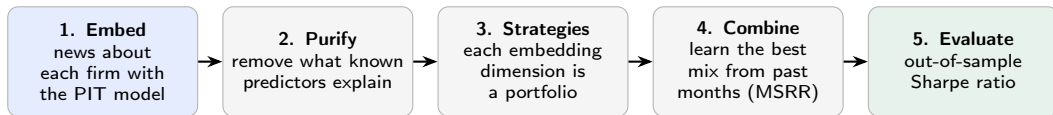


Illustration: same average return ($\approx 10\%$ a year). B is $3\times$ less noisy, so its Sharpe ratio is $3\times$ higher.

The economic test: can news embeddings pick stocks?



- **Data:** Dow Jones Newswire articles (1979–2020) matched to stocks, and monthly stock returns.
- **Out-of-sample window:** January 2014 to May 2020. The first PIT vintage is December 2013, and the news data end in May 2020.
- **Models compared:** ChronoGPT (base, instruct), PIT-4B (base, FT), and *full-sample* versions of PIT-4B.
- Same pipeline for every model (following Chen et al., 2022; Didisheim et al., 2026): only the embedding function changes.

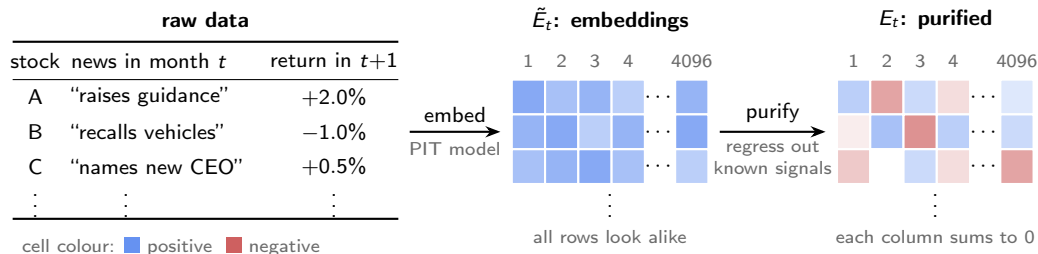
Keeping every step point-in-time

News from	2014	2015	2016	2017	2018	2019	2020*
PIT model	Dec 2013	Dec 2014	Nov 2015	Dec 2016	Dec 2017	Dec 2018	Dec 2019
Full-sample model	Dec 2024	Dec 2024	Dec 2024	Dec 2024	Dec 2024	Dec 2024	Dec 2024

Each cell is the cutoff of the checkpoint that embeds that year's news. * January to May 2020.

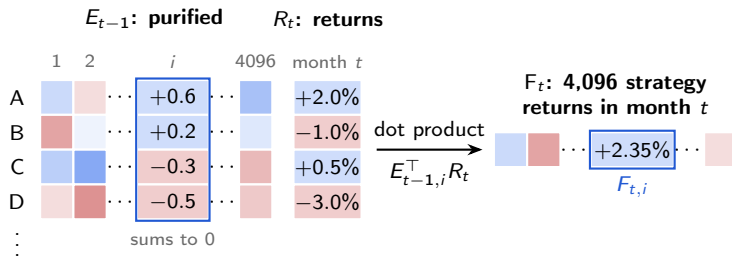
- **Embeddings:** news from year t is embedded by the last checkpoint *before* year t .
- **Portfolio weights:** re-estimated every month on past data only (expanding window, 12-month warm-up).
- **Full-sample benchmark:** what you get with one fixed model, here the final 2024 checkpoint. The downstream pipeline is identical, but the embedding function has seen the future, so the result is not a valid backtest.

Step 1: from news to “pure news” embeddings



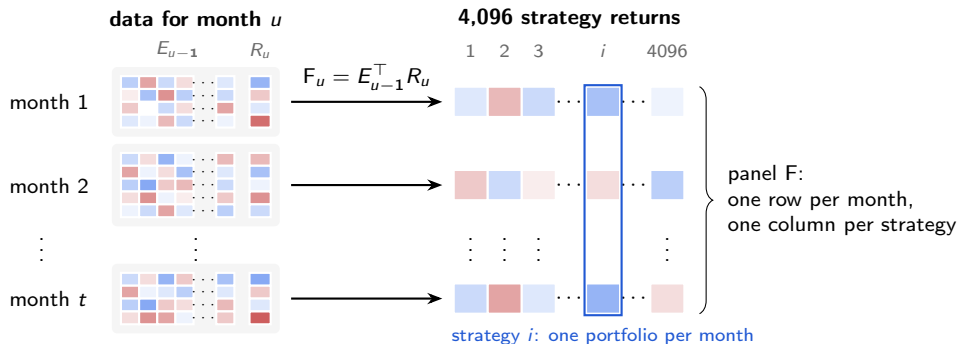
- **Embed**: final-layer hidden state of each article’s last token, averaged per stock and month: one row of 4,096 numbers per stock.
- **Purify**: regress out firm characteristics already known to predict returns (size, value, momentum, ...) (Jensen et al., 2023). The residual is the *news* those signals miss, without the direction shared by all LM embeddings (Ethayarajh, 2019).

Step 2: every embedding dimension is a trading strategy



- Column i of E_{t-1} = **portfolio weights**: buy the stocks that score high on dimension i (A, B), short those that score low (C, D). The weights sum to zero, so it is a zero-cost long-short bet.
- Its return is a dot product, $F_{t,i} = E_{t-1,i}^\top R_t$, with no “predict returns first” step. The 4,096 dimensions give 4,096 strategy returns every month.

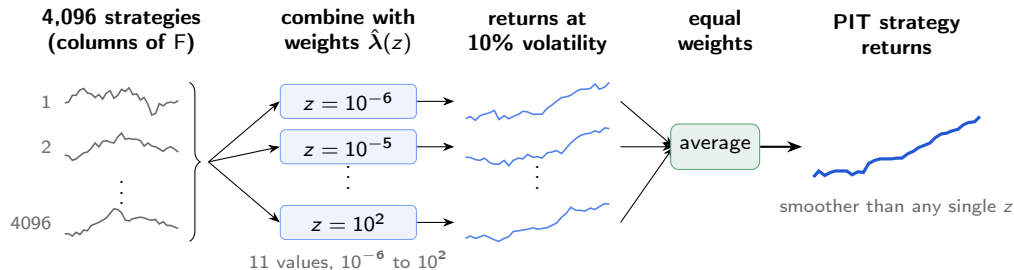
Step 2: repeat every month to build a panel



- Each month u is one data point: embeddings of last month's news E_{u-1} and this month's returns R_u . The number of stocks varies; every row of F has 4,096 entries.
- Column i of F is **strategy i** : every month, hold the portfolio given by dimension i and record its return. **Next**: combine the 4,096 strategies into one.

Step 3: combine the strategies with Maximum Sharpe Ratio Regression

$$\hat{\lambda}_t(z) = \arg \min_{\lambda \in \mathbb{R}^{4096}} \frac{1}{T} \sum_{u \leq t} (1 - \lambda^\top F_u)^2 + z \|\lambda\|_2^2 \implies \pi_t(z) = E_t \hat{\lambda}_t(z)$$

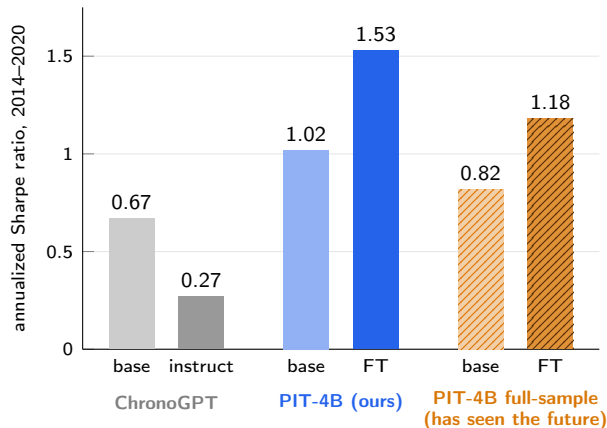


- **Ridge regression with every label set to 1**: the mix whose payoff stays closest to “+1 every month” has high mean and low variance, hence the highest Sharpe ratio (Britten-Jones, 1999).
- **No tuning on test data**: one portfolio per penalty z , each scaled to 10% volatility, then averaged. Refit every month on past months only.

Outline

- 1 Introduction
- 2 Training
- 3 Results I: language benchmarks
- 4 Setup: the economic test
- 5 Results II: economic value**
- 6 Conclusion

Result: PIT embeddings earn high Sharpe ratios out of sample



- 1 **Real economic value without look-ahead:** SR 1.02 (base) and 1.53 (fine-tuned).
- 2 **Scale matters:** the prior PIT models reach 0.67 and 0.27.
- 3 **Instruction tuning helps:** 1.02 $\rightarrow$ 1.53.
- 4 **Look-ahead does not help here:** the full-sample versions earn 0.82 and 1.18.

For calibration: the U.S. stock market has $SR \approx 0.4\text{--}0.5$.

Does one signal subsume another? Spanning tests

Regress the monthly returns of a *target* strategy on a *benchmark*:

$$r_t^{\text{target}} = \alpha + \beta r_t^{\text{bench}} + \varepsilon_t$$

- βr^{bench} : the part you could copy by holding the benchmark.
- α : what the benchmark *cannot* reproduce.
- $\alpha \neq 0 \Leftrightarrow$ combining both beats the benchmark alone:
 $SR_{\text{both}}^2 = SR_{\text{bench}}^2 + (\alpha/\sigma_\varepsilon)^2$.
- ML analogy: does feature set A add information beyond feature set B?

Other model m	annualized α when regressing ...	
	PIT-4B-FT on m	m on PIT-4B-FT
ChronoGPT-base	13.1%***	3.4%
ChronoGPT-instruct	13.7%***	0.7%
PIT-4B	11.4%***	3.2%
PIT-4B-full	12.5%***	5.1%
PIT-4B-FT-full	11.3%***	4.5%*

Newey–West standard errors; * $p < 0.10$, *** $p < 0.01$ (t between 3.0 and 3.8).

- PIT-4B-FT earns 11–14% a year that none of the other models can replicate.
- No model earns an alpha over PIT-4B-FT that is significant at 5%.
- ChronoGPT earns no significant alpha over *any* of our four models: our embeddings span theirs, not the reverse.

Why doesn't the “leaky” model win?

- One would expect look-ahead to inflate backtests. Here the full-sample models earn **0.82 / 1.18**, vs. **1.02 / 1.53** for their PIT counterparts.
- A plausible reason: LMs are trained to predict the next token, not returns. Future text reshapes the embedding geometry without necessarily adding return-relevant directions, and the change can even hurt.
- Other recent work finds the same pattern (He et al., 2025; Didisheim et al., 2026).
- We do not claim that look-ahead translates into higher returns; characterizing its effect on the embeddings is left to future work.

The point is validity, not performance

A leaky backtest cannot be interpreted whichever way it comes out, because nobody knows whether the leakage helped or hurt. With PIT models the number is valid by construction.

- 1 Introduction
- 2 Training
- 3 Results I: language benchmarks
- 4 Setup: the economic test
- 5 Results II: economic value
- 6 Conclusion**

Takeaways

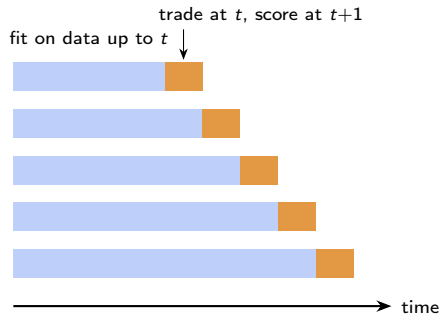
- **PIT LLMs can now be strong LLMs:** PIT-4B beats earlier PIT models on every task we report and comes close to LLaMA-7B and Gemma-3-4B on PIQA and HellaSwag, without ever seeing the future.
- **Temporal validity and economic value are compatible:** leakage-free news embeddings deliver out-of-sample Sharpe ratios of 1.0–1.5.
- **For ML evaluation:** time is a contamination axis. Any study that asks “what could a model have known at time t ?” (backtests, forecasting, historical or in-silico policy simulations) needs PIT models.
- **Open:** code at github.com/DjoFE2021/Point-In-Time-LLM; 36 checkpoints (PIT-1.5B, PIT-4B and PIT-4B-FT for each year 2013–2024) and the pre-training tokens at huggingface.co/Diamegs (PIT-1.5B is named PIT-1B there).

Thank you!

Backup slides

Backtests: finance's walk-forward evaluation

- A **backtest** replays history month by month: at date t , decide what to buy or sell using only what was knowable at t ; score the decision with what happened next.
- ML view: rolling-origin evaluation on a time series. Never shuffle, never peek.
- **Look-ahead bias**: any channel through which information from after t reaches the decision at t . With LLMs, the channel is the pre-training corpus, i.e. the weights (Glasserman and Lin, 2023; Sarkar and Vafa, 2024).
- Consequences: inflated backtests, biased estimates of risk premia, invalid causal and counterfactual conclusions.



an LLM pre-trained up to 2024 has already
read the text of every orange month

Anticipated questions

- **Is the improvement really due to scale?** A matched control (the same 4B run with the time order shuffled) would roughly double the $\approx 28k$ GPU-hours. PIT-1.5B shares ChronoGPT's architecture, so ChronoGPT $\rightarrow$ PIT-1.5B mostly varies the data, while PIT-1.5B $\rightarrow$ PIT-4B varies size within one pipeline.
- **Is the economic gain caused by PIT training?** No: ChronoGPT and PIT-4B are both point-in-time. The comparison shows that strict temporal validity is compatible with much better performance at scale.
- **Are the timestamps safe?** Each document is placed by the Common Crawl dump it was crawled in, so anything in a dump dated $\leq \tau$ was online by τ .
- **Doesn't the gpt-5-nano filter leak?** It only removes examples; it never adds text to the training data.
- **Why year-end, not monthly, checkpoints?** Common Crawl dumps are irregular (2–12 per year). We snapshot at dump boundaries and release year-end vintages; the 2015 vintage is November 2015 (the last 2015 dump), which makes the 2016 evaluation slightly more conservative.

Pairwise spanning regressions (full table)

Benchmark (row) \ target (column)	(1)	(2)	(3)	(4)	(5)	(6)
(1) ChronoGPT-base	—	−1.42% (2.88%)	4.54%** (2.09%)	13.07% *** (3.68%)	4.39% (3.33%)	4.99%** (2.05%)
(2) ChronoGPT-instruct	3.77% (2.34%)	—	4.93%** (1.99%)	13.65% *** (3.63%)	6.88%** (3.51%)	6.15%*** (2.14%)
(3) PIT-4B	3.14% (3.03%)	−0.81% (3.56%)	—	11.41% *** (3.66%)	3.66% (3.70%)	4.73%** (2.19%)
(4) PIT-4B-FT	3.36% (3.24%)	0.65% (3.95%)	3.20% (2.22%)	—	5.07% (4.24%)	4.49%* (2.36%)
(5) PIT-4B-full	1.58% (2.55%)	−1.07% (3.31%)	3.51%* (1.95%)	12.46% *** (3.67%)	—	4.34%** (1.99%)
(6) PIT-4B-FT-full	1.28% (2.87%)	−0.59% (3.69%)	3.40% (2.11%)	11.34% *** (3.74%)	2.41% (3.65%)	—

Each cell is the annualized alpha from regressing the monthly returns of the column model on those of the row model, with Newey–West standard errors in parentheses. * $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$. Sample: January 2014 to May 2020.

IFEval by vintage: early vs. late checkpoints

Metric	2013–2020	2021–2024	Δ (pp)
Base, prompt strict	17.14%	15.40%	−1.74
Base, prompt loose	18.31%	16.43%	−1.89
Base, instr. strict	29.70%	28.02%	−1.68
Base, instr. loose	31.25%	29.27%	−1.98
FT, prompt strict	20.05%	23.05%	+3.00
FT, prompt loose	21.16%	23.62%	+2.46
FT, instr. strict	33.42%	37.85%	+4.43
FT, instr. loose	34.60%	38.80%	+4.20

- Averages over the annual checkpoints in each period, base (PIT-4B) and fine-tuned (PIT-4B-FT) separately.
- Base models are flat to slightly worse in later vintages.
- After fine-tuning, later vintages gain 2.5–4.4 pp: later checkpoints adapt better to instruction tuning.

The temporal-leakage classifier (abridged prompt)

Prompt given to gpt-5-nano for every SFT example

You are a binary classifier. Determine if text contains TIME-SENSITIVE FACTS that could become outdated or incorrect over time. Output ONLY "0" (timeless) or "1" (time-aware).

Output "1" ONLY if the text states facts that could BECOME OUTDATED, such as: who currently holds a political office; recent or dated events; current statistics, prices, or rankings; a real person doing something specific at a specific time; laws or policies tied to a specific time frame.

Output "0" if the text is: math problems or solutions; programming/coding tasks or tutorials; general educational content about real software; generic advice or best practices; hypothetical scenarios; instruction-following tasks; scientific facts that don't change; abstract reasoning or logic puzzles.

KEY DISTINCTION: Mentioning a real tool, company, or product by name does NOT make text time-aware. Only FACTUAL CLAIMS THAT COULD BECOME OUTDATED do.

The full prompt, with worked examples, is in the paper's appendix.

References I

- Mark Britten-Jones. The sampling error in estimates of mean-variance efficient portfolio weights. *The Journal of Finance*, 54(2): 655–671, 1999.
- Yifei Chen, Bryan T Kelly, and Dacheng Xiu. Expected returns and large language models. *Available at SSRN 4416687*, 2022.
- Antoine Didisheim, Bryan T. Kelly, Mo Pourmohammadi, and Hanqing Tian. The inefficient pricing of news. Working paper, SSRN 6540399, 2026.
- Kawin Ethayarajh. How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65. Association for Computational Linguistics, 2019.
- Paul Glasserman and Caden Lin. Assessing look-ahead bias in stock return predictions generated by GPT sentiment analysis. *arXiv:2309.17322*, 2023.
- Songrun He, Linying Lv, Asaf Manela, and Jimmy Wu. Chronologically consistent large language models. *arXiv:2502.21206*, 2025.
- Theis Ingerslev Jensen, Bryan Kelly, and Lasse Heje Pedersen. Is there a replication crisis in finance? *The Journal of Finance*, 78(5): 2465–2518, 2023.
- Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tulu 3: Pushing frontiers in open language model post-training. *arXiv:2411.15124*, 2024.
- Alejandro Lopez-Lira and Yuehua Tang. Can ChatGPT forecast stock price movements? Return predictability and large language models. *arXiv:2304.07619*, 2023.
- Niklas Muennighoff, Alexander M. Rush, Boaz Barak, Teven Le Scao, Aleksandra Piktus, Nouamane Tazi, Sampo Pyysalo, Thomas Wolf, and Colin Raffel. Scaling data-constrained language models. In *Advances in Neural Information Processing Systems*, 2023.

References II

- Guilherme Penedo, Hynek Kydlíček, Anton Lozhkov, Margaret Mitchell, Colin A Raffel, Leandro Von Werra, Thomas Wolf, et al. The FineWeb datasets: Decanting the web for the finest text data at scale. *Advances in Neural Information Processing Systems*, 37: 30811–30849, 2024.
- Suproteem K Sarkar and Keyon Vafa. Lookahead bias in pretrained language models. In *ICML 2025 Workshop on Reliable and Responsible Foundation Models*, 2024.
- Yutong Yan, Raphael Tang, Zhenyu Gao, Wenxi Jiang, and Yao Lu. DatedGPT: Preventing lookahead bias in large language models with time-aware pretraining. *arXiv:2603.11838*, 2026.
- Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv:2311.07911*, 2023.