AMORTIZED OFF-POLICY EVALUATION FOR LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Accurate evaluation is central to selecting which LLM to deploy, yet testing a candidate on live traffic exposes real users to an unvetted model. Teams therefore evaluate candidates offline, on data produced by already-deployed models. This is off-policy evaluation (OPE), and it faces two distribution shifts: as a model is updated in post-training, its responses diverge from the logged ones (policy shift), and the reward definition under which it is judged changes with business requirements (reward shift). Classical OPE methods are ill-suited to this continual-deployment setting because they are defined per task and require fitting from scratch on every new logged dataset or reward definition. To address this, we propose PFN-OPE, a prior-data fitted network that amortizes OPE across a distribution of contextual-bandit tasks. We pretrain it once on tasks constructed from a pool of LLM responses scored by several reward functions, in which both shifts occur. At test-time it maps a logged dataset and one sampled target response per prompt to a value estimate in a single forward pass, with no per-task fitting. On HelpSteer2 and UltraFeedback with Qwen, Llama, and Gemma policies, PFN-OPE achieves 2.0 to 9.3 times lower error than the best baselines across all tested configurations.

1 INTRODUCTION

Choosing which large language model (LLM) to deploy hinges on accurate evaluation (Zheng et al., 2023), yet testing a candidate on live traffic exposes users to an unvetted model. As such, candidates are often evaluated offline with logged data from previously deployed LLMs. A company has human-rated chat logs and wants to estimate the reward a candidate model would receive on the same conversations. The logs measure the deployed assistant, not the candidate. The candidate's responses differ from the logged ones, which we call *policy shift*, and the rubric the human annotators apply changes with the company's requirements, which we call *reward shift*. Answering the company's question from these logs is off-policy evaluation.

Off-policy evaluation (OPE) for LLM evaluation can be viewed as OPE on a contextual bandit, a one-step interaction in which a context arrives, a policy samples an action, and a reward is observed (Kallus et al., 2022; Wang et al., 2017). For an LLM, the context is the prompt, including any conversation history, the action is the response, and the reward is assigned to the response by an annotator or a reward model (Ouyang et al., 2022). The deployed model that produced the logs is the behavior policy, the candidate is the target policy, and its value is the expected reward it earns on the prompts. OPE estimates that value from the logs, which hold a reward for every behavior response and none for the response the target policy would give to the same prompt.

Three classical estimators exist — inverse propensity scoring (IPS), doubly robust (DR), and the direct method (DM) (Jiang & Li, 2016; Strehl et al., 2010; Dudík et al., 2011)— and the first two are out of reach for many popular LLMs. IPS and DR estimators need the probability of each logged response under both policies, which a black-box model does not expose, and the target policy's responses almost never recur in the logs, so the weights would have little to no support (Saito & Joachims, 2022). LLM evaluation therefore uses the DM: train a reward model on the logs and apply it to the target policy's responses (Ouyang et al., 2022; Rafailov et al., 2024). This has two costs. First, every new logged dataset or reward definition means training a new reward model. Each checkpoint of a post-training run is judged under a rubric that shifts with requirements, so each is a fit from scratch. Second, no reward model can be checked on the target policy's responses, whose rewards are unobserved — exactly where it must extrapolate. Both costs come from fitting one model to one logged dataset at a time.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Prior-data fitted networks (PFNs) remove the per-dataset fit (Müller et al., 2022; Hollmann et al., 2023; 2025). A PFN is a transformer pretrained on datasets sampled from a prior over tasks; given a new dataset and a query in context, it returns a prediction in one forward pass, so a new dataset is a new input rather than new training data. For OPE this offers two things the direct method lacks. The first is amortization (Cranmer et al., 2020): each new logged dataset is a new input to the same network. The second is generalization across the shifts: a reward model fitted to the logs must extrapolate to responses it has never seen, whereas a network pretrained on many tasks in which the target policy differs from the behavior policy has seen such shifts and learned how far the target value departs from the logged rewards. Neither comes for free, because a PFN is only as good as its prior, and no prior for OPE exists. The question is whether one can build a prior whose tasks capture the reward and policy shifts LLM evaluation faces.

We propose *prior-data fitted network for off-policy evaluation* ***PFN-OPE*** that maps a logged dataset and one sampled response per prompt from the target policy to an estimate of the target policy's value, in one forward pass and with no per-task fitting. Our key insight is that the label a PFN needs, the target policy's value, is unobservable on real logs but computable on tasks built from a pool of stored responses with rewards: when every policy is a distribution over the pool, every value is a weighted sum of stored rewards. We build the pool once, from several LLMs and several reward functions, and sample tasks from it in which both shifts occur: affine combinations of the reward functions play the role of new reward definitions, and reward-tilted reweightings of an LLM's responses play the role of post-trained policies. Pretrained on these tasks, the network learns to read the shift off the observation itself, how the target responses differ from the logged ones and what the logged rewards say under the current definition, and by the PFN argument it approximates the minimum-MSE estimator for tasks drawn from the prior.

Our contributions are:

- **Amortized OPE via prior-data fitted in-context learning.** One network maps a logged dataset and sampled target responses to a value estimate in one forward pass, with no per-task fitting. This is a new problem setting for PFNs: the input is a set of paired responses, one with a reward and one without, and the label is a functional of the whole set rather than the label of one query (Section 3.1).
- **A task prior for LLM off-policy evaluation.** Tasks are built from a pool of LLM responses with rewards from several reward functions: affine combinations of the reward functions as reward definitions, reward-tilted and routed policies as policy shifts, and target values computed from the pool (Section 3.2).
- **Evaluation with LLM policies.** On HelpSteer2 and UltraFeedback, PFN-OPE has the lowest MSE under all five reward definitions, four of them absent from pretraining, by 2.0 to 9.3 times over the best of the baselines. It has the lowest MSE in 11 of 12 behavior–target pairs when the target LLM and the reward model are both absent from pretraining (Section 4).

2 Background

Off-policy evaluation. We consider off-policy evaluation (OPE) in a contextual-bandit setting. In a contextual-bandit interaction, the environment provides a context $x \in \mathcal{X}$, a policy samples an action $a \in \mathcal{A}$, and the environment returns a reward r with a reward function $R : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$. The behavior policy $\mu : \mathcal{X} \to \Delta(\mathcal{A})$ samples the logged data $\mathcal{D}_\mu := \{(x_i, a_i, r_i)\}_{i=1}^n$ from an environment, while the target policy π is the policy whose value we want to estimate. OPE estimates the target policy value $V(\pi)$ as the expected reward of π: $V(\pi) := \mathbb{E}_{x\sim p(\cdot), a\sim\pi(\cdot|x)}\left[R(x, a)\right]$.

The main challenge in OPE is that the behavior and target policies may induce different action distributions, resulting in distribution shift (Saito & Joachims, 2022). The logged data contain rewards only for actions sampled by the behavior policy, leaving the rewards of actions sampled from the target policy unobserved. The distribution shift becomes more severe as the behavior and target policies diverge.

In our application, x is a prompt, a is a generated response, and r is the reward assigned to that response by a human annotator. The behavior policy μ is the LLM that produced the logged responses, and π is the LLM being evaluated.

Off-policy evaluation estimators. OPE estimators fall into three methods: inverse propensity scoring (IPS), direct method (DM), and doubly robust (DR) estimation (Jiang & Li, 2016). IPS estimates the target policy value by correcting the behavior policy induced action distribution to that of the target policy by reweighting logged rewards with IPS weights (Saito & Joachims, 2022). IPS is unbiased given behavior policy. However, its variance grows as the action space increases. The direct method approximates the reward function from logged data and therefore has low variance but high bias due to estimation error from distribution shift. DR combines the advantages of both IPS and DM (Farajtabar et al., 2018), doubly robust estimator is unbiased when either IPS or DM is unbiased and has lower variance compared to IPS.

Conventional OPE estimators do not amortize inference across evaluation datasets. For IPS and DR, they must recompute importance weights for each behavior–target policy pair and require policy probability information that may be unavailable for black-box LLMs. DM also requires re-training when the ground-truth reward function changes.

Our work focuses on improving the DM's ability to predict rewards for target policy responses with amortized off-policy evaluation capability. This focus is motivated by the high variance of IPS in the large response spaces induced by LLM policies. The resulting reward model can also serve as the DM component of a DR estimator when combined with an IPS method.

Prior-data fitted networks. A prior-data fitted network (PFN) is a single transformer q_θ pretrained across many datasets: given a dataset and a query in context, it outputs a predictive distribution for the query's label in one forward pass, so a new dataset is a new input rather than new training data (Müller et al., 2022). The datasets come from a synthetic prior over tasks $p(\tau)$. In the standard PFN a task τ is a relationship between inputs z and labels y, so sampling $\tau \sim p(\tau)$ samples a function; from τ we draw a dataset $\mathcal{D} \coloneqq \{(z_i, y_i)\}_{i=1}^n$ and a query z_* with hidden label y_*. Pretraining minimizes the negative log-likelihood of the hidden label,

$$\mathcal{L}_{\mathrm{PFN}}(\theta) \coloneqq \mathbb{E}_{\tau\sim p(\tau)}\, \mathbb{E}_{(\mathcal{D},z_*,y_*)\sim p(\cdot\mid\tau)} \big[-\log q_\theta(y_* \mid \mathcal{D}, z_*)\big], \tag{1}$$

an expectation over tasks and, within each task, over the dataset and query drawn from it.

The property PFN-OPE inherits is that this loss is minimized by the posterior predictive, $q^\star(y_* \mid \mathcal{D}, z_*) = p(y_* \mid \mathcal{D}, z_*) = \int p(y_* \mid z_*, \tau)\, p(\tau \mid \mathcal{D})\, d\tau$, whose mean is the minimum-MSE estimate of y_* for tasks drawn from the prior (Müller et al., 2022). A trained PFN thus performs Bayesian inference under $p(\tau)$ without representing the posterior, and the prior is its entire inductive bias: which tasks the prior contains decides what the network infers from a small dataset and how far it transfers. For a real-valued label, a PFN's output is a categorical distribution over fixed intervals of the label's range (Müller et al., 2022). PFN-OPE keeps this pretraining recipe and the Bayes argument. Section 3 changes what a task, a dataset, and a prediction target are.

3 PFN-OPE: Prior-Data Fitted Network for Off-Policy Evaluation

We propose PFN-OPE for efficient off-policy evaluation of LLM policies under policy and reward shifts. Unlike conventional OPE methods, which fit an estimator separately for each task, PFN-OPE amortizes estimation across a prior of OPE tasks, allowing a single pretrained network to estimate target policy values without per-task fitting. We first formulate OPE as in-context prediction in Section 3.1, then define the task prior used to generate pretraining tasks in Section 3.2, and finally describe the transformer architecture in Section 3.3.

3.1 OPE as in-context prediction

PFN-OPE formulates OPE as an in-context prediction problem in which the target policy value is estimated from a behavior policy sampled logged data and target policy sampled target responses.

Input. We assume access to a logged dataset $\mathcal{D}_\mu \coloneqq \{(x_i, a_i^\mu, r_i^\mu)\}_{i=1}^n$, where x_i is a prompt, a_i^μ is the response sampled from the behavior policy μ, and $r_i^\mu = R(x_i, a_i^\mu)$ is its observed reward. We also assume black-box sampling access to the target policy π. For each prompt x_i, we sample one target response $a_i^\pi \sim \pi(\cdot \mid x_i)$ and form the target prompt–response dataset $\mathcal{D}_\pi \coloneqq \{(x_i, a_i^\pi)\}_{i=1}^n$. Together, $\mathcal{D}_\mu$ and $\mathcal{D}_\pi$ constitute the OPE observation $O \coloneqq (\mathcal{D}_\mu, \mathcal{D}_\pi)$, which serves as the input to

the PFN-OPE transformer. Equivalently, the input can be represented as $O = \{(x_i, a_i^\mu, r_i^\mu, a_i^\pi)\}_{i=1}^n$. The target-response rewards are not included in O and must be inferred by the model.

Prediction target. PFN-OPE is trained to predict the correction value C in Equation (2) given O rather than predicting target-policy value directly. Subtracting the average logged reward from $V(\pi)$ centers the target on the observed behavior reward and focuses prediction on the target-policy correction.

$$C \coloneqq V(\pi) - \bar{r}, \qquad \bar{r} \coloneqq \frac{1}{n}\sum_{i=1}^{n} r_i. \tag{2}$$

Pretraining. During pretraining, PFN-OPE learns an inference rule across various OPE tasks. Tasks τ are sampled from the task prior described in Section 3.2. A sampled task τ specifies a reward function, and behavior and target policies. Given τ, we generate the observation dataset $O \sim p(O \mid \tau)$ for pretraining. Because reward functions and policies are known, PFN-OPE can be supervised to predict the correction target and consequently the target policy value.

We train the network via quantile regression. A quantile head returns $\widehat{Q}_\theta^\kappa(O)$, an estimate of the κ-quantile of C given O, for every level κ in $\mathcal{K}$, and pretraining minimizes the pinball loss in Equation (3) (Koenker & Bassett, 1978). We follow TabICLv2 (Qu et al., 2026), which found quantile regression more accurate than TabPFN's discretized output for regression targets. Another advantage is that a categorical head over fixed intervals Section 2 fixes the range of C before pretraining, whereas the quantile levels adapt to each task's predictive distribution.

For each level κ, the expected pinball loss is minimized by the κ-quantile of the posterior predictive $p(C \mid O)$ induced by the prior (Koenker & Bassett, 1978), by the argument behind Equation (1) with (O, C) in place of $(\mathcal{D}, z_*, y_*)$. A network that attains the minimum is therefore the posterior-quantile estimator for the prior, and the prior is its inductive bias, as in Section 2.

$$\mathcal{L}(\theta) \coloneqq \mathbb{E}_{\tau\sim p(\tau)}\,\mathbb{E}_{O\sim p(O|\tau)}\Big[\frac{1}{|\mathcal{K}|}\sum_{\kappa\in\mathcal{K}} \ell_\kappa\big(C - \widehat{Q}_\theta^\kappa(O)\big)\Big], \qquad \ell_\kappa(u) \coloneqq u\,\big(\kappa - \mathbf{1}\{u < 0\}\big). \tag{3}$$

Inference. At test-time, PFN-OPE receives an observation O from a test-time OPE task. The target-policy rewards and value are unknown. The pretrained network uses O to predict the correction. In a single forward pass, it produces the quantile estimates $\{\widehat{Q}_\theta^\kappa(O)\}_{\kappa\in\mathcal{K}}$. We aggregate these quantiles as $\widehat{C}_\theta(O)$ and estimate the target policy value $\widehat{V}_\theta(\pi; O)$:

$$\widehat{C}_\theta(O) \coloneqq \frac{1}{|\mathcal{K}|}\sum_{\kappa\in\mathcal{K}} \widehat{Q}_\theta^\kappa(O), \qquad \widehat{V}_\theta(\pi; O) \coloneqq \bar{r} + \widehat{C}_\theta(O). \tag{4}$$

Averaging the predicted quantiles approximates the mean of the predictive distribution of the correction. Hence, $\widehat{V}_\theta(\pi; O)$ approximates the posterior-mean estimate of the target-policy value under the task prior, which is the minimum-MSE estimator for tasks drawn from that prior.

PFN-OPE handles both policy and reward shifts through the observation, without parameter updates. A policy shift changes the target-response component $\mathcal{D}_\pi$ in O, because the new target policy produces different responses for the logged prompts. A reward shift changes the rewards assigned to the logged responses when they are rescored under a new reward function, while the target responses remain unscored. Both shifts are reflected in O and processed by the same pretrained inference rule.

3.2 Constructing Task Prior from a Response Pool

The task prior determines the reward and policy shifts during pretraining, and therefore serves as the estimator's inductive bias. Broader coverage of reward and policy shifts during pretraining generally improves PFN-OPE's ability to generalize to related shifts at test-time. Each pretraining task must also provide a target-policy value $V_\tau(\pi)$ for supervision.

Generating and scoring new LLM responses for every pretraining task would be computationally expensive. We instead construct a finite response pool before pretraining and reuse it across tasks.

Each task is obtained by selecting a prompt distribution $p(x \mid \tau)$, a reward function R_τ, a behavior policy μ, and a target policy π. The target-policy value $V_\tau(\pi)$ can then be evaluated as a finite sum over the pool, allowing us to sample pretraining tasks without additional LLM generation or reward-model evaluation.

Response pool. We assume that a set of LLMs $\mathcal{G}$ is available for pretraining. An LLM $g \in \mathcal{G}$ is a model's parameters together with its system prompt and decoding settings, and $g(a \mid x)$ is its response distribution. For every prompt x in a finite set $\mathcal{X}_0 \subset \mathcal{X}$ and every $g \in \mathcal{G}$, we sample K responses from g once and store them as $\mathcal{A}_g(x) \coloneqq \{a_g^{(1)}(x), \ldots, a_g^{(K)}(x)\}$. Reward functions $R_1, \ldots, R_J$, the heads of one or more reward models, assign a reward to every stored response, so the pool is the set of tuples $(x, g, a, R_1(x,a), \ldots, R_J(x,a))$.

Reward functions. A task's reward function is an affine combination of the stored rewards, each standardized over the pool, $\tilde{R}_j \coloneqq (R_j - m_j)/s_j$ with m_j and s_j the mean and standard deviation of R_j on the pool:

$$R_\tau(x,a) \coloneqq \sum_{j=1}^{J} \lambda_{\tau j}\, \tilde{R}_j(x,a), \qquad \lambda_\tau \sim \mathcal{N}\big(w,\ \mathrm{diag}(w) - ww^\top\big), \tag{5}$$

with evaluation weights λ_τ drawn once per task from a Gaussian centered on base weights $w \in \Delta^{J-1}$. The covariance is singular, so every draw satisfies $\sum_j \lambda_{\tau j} = 1$ We standardize the rewards so that the weights compare criteria on one scale, and we allow negative weights because an evaluation criterion can penalize an aspect, as helpfulness minus verbosity does. The same weights apply to every prompt and response, so the logged rewards of a task describe one consistent evaluation criterion, and each new draw of λ_τ is a new reward definition obtained from the stored rewards alone.

Behavior and target policies. A task's policies are distributions over stored responses. For LLM g and prompt x, the pool policy is the uniform distribution on $\mathcal{A}_g(x)$,

$$\widehat{g}(a \mid x) \coloneqq \frac{1}{K} \sum_{k=1}^{K} \mathbf{1}\big\{a = a_g^{(k)}(x)\big\}, \tag{6}$$

the K-sample Monte Carlo approximation of $g(a \mid x)$. A task picks a behavior LLM $g \in \mathcal{G}$ and a target LLM $g' \in \mathcal{G}$ and sets $\mu \coloneqq \widehat{g}$. The target policy starts from $\widehat{g}'$ and is shifted in the way reward-guided post-training shifts a policy, toward the responses a reward model prefers (Ouyang et al., 2022). The preference is a selection reward $S_\tau(x,a) \coloneqq \sum_{j=1}^{J} \omega_{\tau j} \tilde{R}_j(x,a)$ with selection weights $\omega_\tau \in \Delta^{J-1}$, and it acts through a selection strength $\eta \geq 0$ and a routing probability $\alpha \in [0,1]$:

$$\nu_\eta(a \mid x) \coloneqq \frac{\widehat{g}'(a \mid x)\, e^{\eta S_\tau(x,a)}}{\sum_{a'} \widehat{g}'(a' \mid x)\, e^{\eta S_\tau(x,a')}}, \qquad \pi \coloneqq (1-\alpha)\,\mu + \alpha\,\nu_\eta. \tag{7}$$

The tilt ν_η has the form of the optimal policy of a KL-regularized reward objective (Rafailov et al., 2023), which is how post-training moves a policy: larger η favors responses with higher S_τ, and $\eta = 0$ with $\alpha = 1$ returns $\widehat{g}'$ itself, a change of the LLM with no reward guidance. Routing mixes ν_η with μ because the value gap $V_\tau(\pi) - V_\tau(\mu)$ is then linear in α, so the prior covers shifts from none ($\alpha = 0$) to the full gap ($\alpha = 1$); a target response routed to μ is a fresh draw from $\widehat{g}$, not a copy of the logged response. The selection weights ω_τ are drawn independently of λ_τ because the reward that steered a policy need not be the reward under which it is judged.

Training labels. A task fixes its prompt distribution $p(x \mid \tau)$ on $\mathcal{X}_0$ before any observation is drawn. The value of the target policy π at a prticular task τ is computed as

$$V_\tau(\pi) = \sum_{x \in \mathcal{X}_0} p(x \mid \tau) \sum_{a} \pi(a \mid x)\, R_\tau(x,a), \tag{8}$$

The label $C = V_\tau(\pi) - \bar{r}$ of Equation (2) is defined relative to the observed logged mean, so two observations of one task carry different labels, and the network learns to predict the correction for the dataset in hand rather than the task's average gap. Algorithm 1 summarizes pretraining.

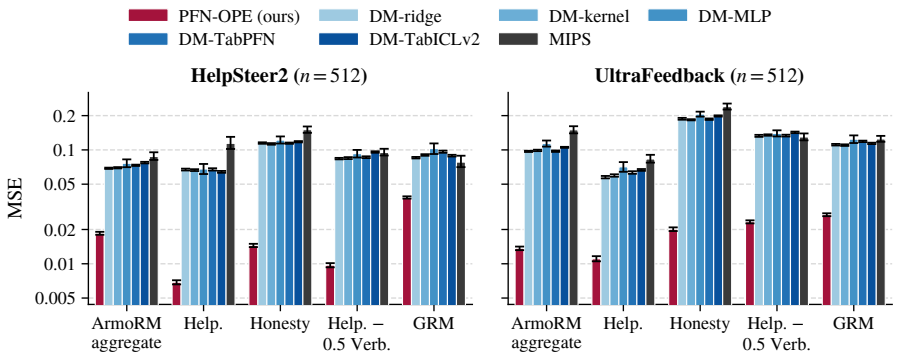


Figure 1: **One checkpoint transfers to four rewards absent from pretraining.** MSE at $n = 512$ over the six pairs of seen LLMs, with 95% intervals. The ArmoRM aggregate is the seen reward; the other four are unseen. PFN-OPE's MSE is 2.0 to 9.3 times lower than the best baseline's.

3.3 PFN-OPE Architecture

PFN-OPE follows the PFN framework, but adapts its transformer architecture to the structure of the OPE observation O. In particular, we modify three components of the standard PFN architecture: the input representation, the attention mask, and the readout.

Input representation. The input is the OPE observation O, which is represented by two tokens:

$$L_i = f_\mu\big(\phi_x(x_i), \phi_a(a_i^\mu), r_i^\mu\big), \qquad Q_i = f_\pi\big(\phi_x(x_i), \phi_a(a_i^\pi), \mathrm{MASK}\big), \tag{9}$$

where L_i represents the logged response and its observed reward, whereas Q_i represents the target response whose reward is unobserved. Prompts and responses are mapped to frozen text embeddings $\phi_x(x)$ and $\phi_a(a)$ using Qwen3-Embedding-4B (Zhang et al., 2025). The symbol MASK denotes a learned vector that replaces the unobserved target reward.

Attention mask. For the input structure, we use a directional attention mask. Each logged token L_i attends only to the logged tokens, whereas each target token Q_i attends to all logged and target tokens. Target tokens attend to the logged tokens because the reward of a target response must be inferred from the logged responses and their observed rewards. They also attend to each other because PFN-OPE predicts the correction C, which depends jointly on the entire set of sampled target responses.

Readout. The transformer includes a learned summary token whose final representation is passed to an output head that predicts the quantiles $\{\widehat{Q}_\theta^\kappa(O)\}_{\kappa\in\mathcal{K}}$ of the correction C. We obtain $\widehat{C}_\theta(O)$ by averaging these predicted quantiles, as defined in Equation equation 4.

4 Experiments

The experiments ask whether PFN-OPE replaces per-task fitting under the *reward shift* and *policy shift*. The metric is the MSE between $\widehat{V}(\pi)$ and $V_\tau(\pi)$. (Saito et al., 2021). Five questions organize the section:

- **Q1** Does PFN-OPE generalize to rewards absent from pretraining? (Section 4.2)
- **Q2** Does it generalize to unseen target LLMs under an unseen reward model? (Section 4.3)
- **Q3** Mechanism: why does PFN-OPE win, and how much logged data does it need? (Section 4.4)
- **Q4** Prior ablation: which axis of the prior carries the gain? (Section 4.5)
- **Q5** Prior coverage: what do the shifted policies add? (Section 4.6)

4.1 Common experimental setup

Pool, prompts, and LLMs. Prompts come from HelpSteer2 (Wang et al., 2024b) and UltraFeedback (Cui et al., 2024), reported separately. Each dataset holds out 1,024 evaluation prompts that never

Table 1: **PFN-OPE achieves the lowest MSE in 11 of 12 behavior–target pairs when both the target LLM and reward model are absent from pretraining.** MSE ($\times 10^{-2}$) at $n = 512$ under GRM; the small grey number is the larger half-width of the 95% interval. Bold marks the lowest MSE within each dataset and column, determined before rounding.

	Target: Llama-3.2-3B			Target: Gemma-3-1B		
Behavior LLM	Qwen2.5 7B	Qwen2.5 1.5B	Llama-3.2 1B	Qwen2.5 7B	Qwen2.5 1.5B	Llama-3.2 1B
HelpSteer2						
PFN-OPE (ours)	**0.041** ±0.006	**0.87** ±0.16	**0.97** ±0.23	**1.54** ±0.30	**2.01** ±0.21	**2.75** ±0.42
DM-ridge	0.130 ±0.047	17.78 ±1.10	8.45 ±0.66	5.94 ±0.56	30.35 ±1.90	19.37 ±1.80
DM-kernel	0.050 ±0.055	16.57 ±0.91	7.72 ±0.87	4.91 ±0.73	30.32 ±1.70	18.98 ±1.90
DM-MLP	1.26 ±1.60	26.40 ±6.60	7.99 ±2.00	7.44 ±3.20	39.57 ±8.10	20.52 ±5.40
DM-TabPFN	0.055 ±0.066	18.17 ±0.93	6.99 ±0.65	4.72 ±0.52	32.46 ±2.20	18.84 ±1.70
DM-TabICLv2	1.05 ±0.25	25.28 ±1.30	13.35 ±0.81	10.12 ±0.76	44.05 ±2.10	30.10 ±2.10
MIPS	0.27 ±0.12	32.56 ±35.00	10.31 ±0.76	8.26 ±4.00	27.45 ±4.20	18.86 ±4.10
UltraFeedback						
PFN-OPE (ours)	1.120 ±0.041	**5.01** ±1.00	**9.86** ±0.42	**0.038** ±0.004	**6.22** ±0.58	**6.52** ±0.39
DM-ridge	1.55 ±0.15	9.50 ±0.92	12.11 ±0.56	0.043 ±0.043	13.76 ±1.40	19.55 ±1.10
DM-kernel	1.56 ±0.15	8.71 ±0.89	12.24 ±0.51	0.046 ±0.046	14.42 ±1.30	20.18 ±1.10
DM-MLP	1.59 ±0.67	12.95 ±2.20	13.86 ±2.70	1.66 ±1.20	18.88 ±2.90	24.08 ±4.40
DM-TabPFN	2.49 ±0.24	8.65 ±0.79	9.99 ±0.68	0.062 ±0.040	14.60 ±1.60	16.80 ±1.10
DM-TabICLv2	**0.230** ±0.066	13.05 ±1.10	17.80 ±0.69	1.11 ±0.23	20.07 ±1.80	30.16 ±1.10
MIPS	1.31 ±0.41	10.30 ±1.50	14.47 ±0.48	4.25 ±3.80	15.80 ±4.10	24.62 ±4.60

enter pretraining. Each LLM contributes $K = 8$ responses per prompt. Three seen LLMs form the pool $\mathcal{G}$ of Section 3.2: Qwen2.5-1.5B-Instruct, Qwen2.5-7B-Instruct (Yang et al., 2025), and Llama-3.2-1B-Instruct (Grattafiori et al., 2024). Two unseen LLMs serve only as evaluation targets: Llama-3.2-3B-Instruct, a larger checkpoint of a seen family, and Gemma-3-1B-IT (Gemma Team et al., 2025), an unseen family. Every evaluation policy is the pool policy of one LLM (Equation (6)), so policy shift in evaluation is a change of LLM.

Rewards. ArmoRM (Wang et al., 2024a) outputs 18 sub-rewards and the aggregated value. The prior's reward functions $R_1, \ldots, R_J$ in Equation (5) are this aggregate and 15 of ArmoRM's objectives. Four rewards are unseen. We hold out three from ArmoRM: helpfulness, honesty, and helpfulness minus 0.5 times verbosity. The fourth is GRM-Gemma2-2B (Yang et al., 2024), written as GRM, a different reward model. All rewards are standardized with pretraining statistics.

Evaluation tasks and intervals. For each setting of dataset, a behavior LLM, a target LLM, and a reward, there are 100 tasks. A task draws n evaluation prompts. At each prompt, it takes one stored behavior response with its reward and one stored target response without. This is the observation O, and every estimator receives the same one. The true target value is the mean reward of the target LLM's $1{,}024 \times 8$ stored responses. We report the MSE over the 100 tasks with a 95% interval.

Baseline estimators. PFN-OPE is one checkpoint, never updated for any reward, policy, or n. We compare it with six estimators on the same frozen embeddings. Three direct methods refit a reward regressor on every logged dataset, the per-task fitting we replace: DM-ridge, DM-kernel, and DM-MLP (Kallus et al., 2022; Wang et al., 2017). Two direct methods, DM-TabPFN (Hollmann et al., 2023) and DM-TabICLv2 (Qu et al., 2026), predict the target rewards in context with frozen tabular foundation models. They are the closest baselines: they amortize the regressor, whereas PFN-OPE amortizes the OPE. MIPS (Saito & Joachims, 2022) is marginalized importance sampling with a density ratio estimated from action embeddings.

4.2 PFN-OPE GENERALIZES TO FOUR UNSEEN REWARDS

Setup. Reward shift is the case in which the direct method must refit. We test whether PFN-OPE, which never saw the new reward, can replace that refit. The policies are the six ordered pairs of seen LLMs, and only the reward changes: the seen aggregate and the four unseen rewards, at $n = 512$. The held-out ArmoRM rewards are new definitions from a reward model the prior knows. GRM is a reward model the prior never saw.

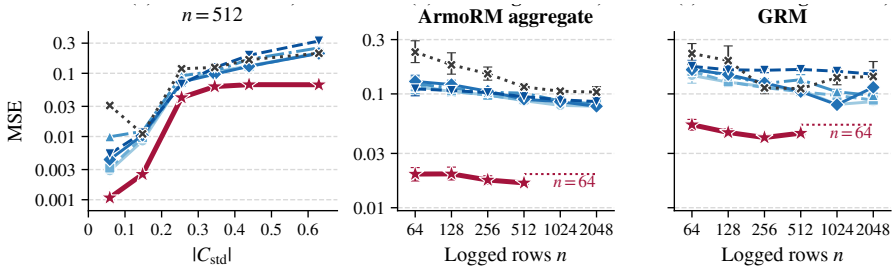


Figure 2: **The advantage grows with the correction and survives 32 times more logged data than the baselines.** Legend as in Figure 1. (a) MSE against the standardized correction $|C_{\text{std}}|$ at $n = 512$: above 0.2, every baseline's MSE rises with the correction, PFN-OPE's stays lower. (b, c) MSE against logged rows n, for seen LLMs under ArmoRM (b) and unseen LLMs under GRM (c).

Results. PFN-OPE has the lowest MSE under all five rewards on both datasets, by 2.0 to 9.3 times over the best baseline (Figure 1). The gain shrinks with distance from the prior but does not vanish. It is 3.7 to 9.3 times for the rewards built from ArmoRM's objectives, which the prior's reward combinations span, and 2.0 to 4.1 times under GRM, which lies outside them. The baselines see the new reward through the same logged rewards and refit to it. The gain is 3.7–9.3× for rewards spanned by the prior and 2.0–4.1× for out-of-prior GRM.

4.3 PFN-OPE GENERALIZES TO UNSEEN TARGET LLMS UNDER AN UNSEEN REWARD MODEL

Setup. This is the deployment case of Section 1: a candidate LLM the prior never saw, scored by a reward model the prior never saw. The target policies are Llama-3.2-3B and Gemma-3-1B, and the reward is GRM, on both datasets: 12 behavior–target pairs at $n = 512$.

Results. PFN-OPE has the lowest MSE in 11 of 12 pairs (Table 1). The gain is 3.1 to 19 times in five pairs on HelpSteer2 and 1.7 to 2.6 times in three pairs on UltraFeedback. In the other four pairs, the estimators are close. Two of these have an MSE below 0.001 for every estimator: the logged mean is already near the target value, so there is little to correct. In the remaining two, both on UltraFeedback, PFN-OPE ties DM-TabPFN from Llama-3.2-1B to Llama-3.2-3B and loses to DM-TabICLv2 by 4.9 times from Qwen2.5-7B to Llama-3.2-3B.

4.4 PFN-OPE IS ROBUST TO LARGE CORRECTION AND LESS LOGGED DATA

Setup. Figure 2a tests whether PFN-OPE performs in large correction C settings. We pool all $n = 512$ tasks of Q1 and Q2 and bin them by the correction in units of the logged reward's standard deviation, $|C_{\text{std}}|$. Figure 2b,c test the second. On the high-correction third of the logged datasets, we grow n from 64 to 2,048, for seen targets under the ArmoRM aggregate (b) and unseen targets under GRM (c). PFN-OPE is evaluated up to $n = 512$.

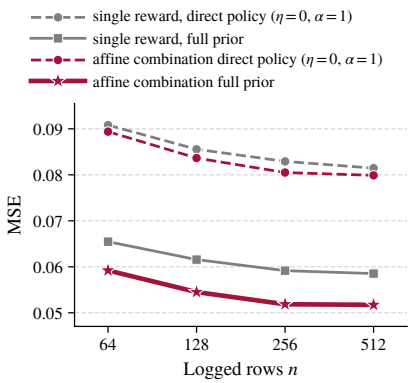


Figure 3: **Ablation of the prior.** MSE of four PFNs pretrained in a 2×2 design over the reward and the policy prior.

Results. Below $|C_{\text{std}}| = 0.2$ every estimator is accurate, with MSE at or below 0.012 for every direct method, because the logged mean is already close to the target value. Above 0.2 the baselines' MSE grows with the correction and reaches 0.21 to 0.32 in the largest bin, while PFN-OPE's stays between 0.04 and 0.07 (Figure 2a). The ratio to the best baseline widens from 1.6 to 3.1. A regressor fitted to the logged rewards predicts target rewards near them, so it inherits the logged mean's error C^2. An estimator trained to predict C does not. More data does not close the gap. With 64 rows, PFN-OPE is more accurate than every baseline with 2,048, by 3.9 times for seen targets and 1.6 times for unseen targets (Figure 2b,c). The baselines' curves flatten above PFN-OPE's, so their remaining error is due to extrapolation. In deployment, those rows are annotations: PFN-OPE with 64 rows is more accurate than the baselines with 32 times as many.

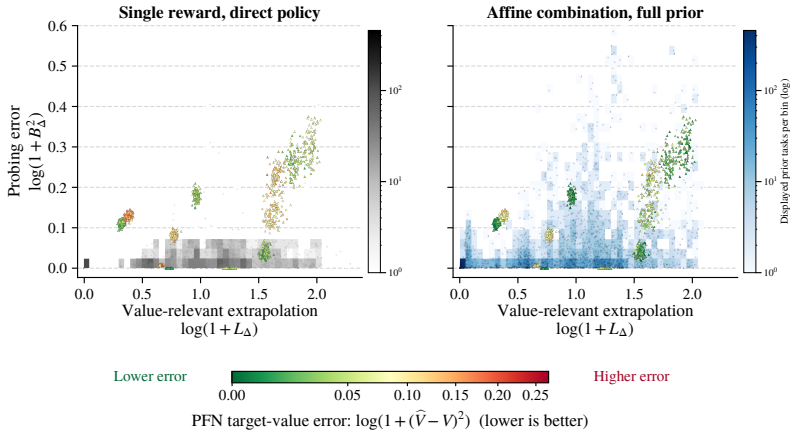


Figure 4: **The PFN fails where its prior has no tasks, and the shifted policies fill the gap.** Each panel shows the pretraining tasks. Triangles are the evaluation tasks with an unseen target LLM under GRM. Left: one reward per task, direct policies. Right: affine reward combinations, full policy prior.

4.5 Prior ablation: the policy axis of the prior carries the gain

Setup. The prior is the estimator's entire inductive bias (Section 2), and Section 3.2 widened it along two axes: reward definitions and policy shifts. To see which axis carries the gain, we pretrained four PFNs in a 2×2 design: one reward function per task or the affine combination of Equation (5), crossed with the direct policies ($\eta = 0$, $\alpha = 1$) or the full policy prior with tilted and routed policies.

Results. The full policy prior lowers MSE by 28% to 36% at every n, whether the reward prior is a single function or a combination (Figure 3). Reward combinations help little on their own, improving by 2% to 3% with pool policies and by 10% to 12% on top of the full policy prior.

4.6 Prior coverage is important in pretraining

Setup. We place every task in a plane whose two axes measure how hard the task is for a linear reward model on the frozen embeddings (definitions in Section D). The horizontal axis, L_Δ, is how far the target responses move away from the logged ones in embedding space, in directions the logged responses do not cover. The vertical axis, the probe correction error B^2_Δ, is how wrong a linear reward model fitted to the logs is about the value gap between the two policies. We plot 1,000 pretraining tasks from each of the two arms of the ablation and the 1,000 evaluation tasks with an unseen target LLM under GRM, at $n = 128$. Each evaluation task is colored by the error of the PFN pretrained on that panel's prior.

Results. The two priors cover the same region except the left corner and upper region, and that corner is where the two PFNs differ (Figure 4). The pool-only prior has no pretraining task there, and its PFN has its highest error on the evaluation tasks there. The full prior has tasks there, and its PFN's error there is low. Elsewhere, both PFNs are accurate where their priors are dense. Every evaluation task in the corner is a Llama-3.2-1B→Llama-3.2-3B task: a target from the family of the logged LLM, whose responses differ in reward more than in form. This shift is closest to a post-trained checkpoint of the logged model, and the shifted policies are what put it in the prior.

5 Conclusion

We introduced PFN-OPE, a prior-data fitted network that amortizes off-policy evaluation of LLMs. We pretrained it once on tasks built from a response pool, where every target value is known by construction. At test-time, it maps a logged dataset and sampled target responses to a value estimate in one forward pass, with no per-task fitting. On HelpSteer2 and UltraFeedback, PFN-OPE has 2.0 to 9.3 times lower MSE than the best baseline. It has the lowest MSE in 11 of 12 behavior–target pairs when the target LLM and the reward model are both unseen. As post-training objectives and feedback signals continue to shift, we see PFN-OPE as a step toward evaluation that keeps pace by reusing one pretraining run across the stream of target policies that deployment produces.

AI USE STATEMENT

In this work, we used generative AI tools to assist with the design of our research methodology and experiments (providing feedback on experiments), to implement methods, to reformat datasets, and to support qualitative data analysis. We have not used generative AI tools to generate synthetic datasets, develop theoretical models or conceptual frameworks, propose or refine hypotheses, or interpret results, and formulating or proving mathematical claims, writing proofs, and translation are not applicable to this work. Additionally, we used generative AI tools to suggest experimental parameters, create and edit software code, draft parts of the paper, and edit the paper to improve readability. We have reviewed all AI-assisted work. AI-generated code was reviewed and tested for correctness before being used in any experiment. AI-assisted qualitative analyses were independently verified by 2 authors. All AI-drafted or AI-edited text was checked by the authors for accuracy and revised as needed. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Vahid Balazadeh, Hamidreza Kamkari, Valentin Thomas, Junwei Ma, Bingru Li, Jesse C. Cresswell, and Rahul Krishnan. CausalPFN: Amortized causal effect estimation via in-context learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=RblaNJGx8C.

Matej Cief, Jacek Golebiowski, Philipp Schmidt, Ziawasch Abedjan, and Artur Bekasov. Learning action embeddings for off-policy evaluation. In *Advances in Information Retrieval*, 2024.

Kyle Cranmer, Johann Brehmer, and Gilles Louppe. The frontier of simulation-based inference. *Proceedings of the National Academy of Sciences*, 117(48):30055–30062, 2020. doi: 10.1073/pnas.1912789117. URL https://www.pnas.org/doi/abs/10.1073/pnas.1912789117.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Bingxiang He, Wei Zhu, Yuan Ni, Guotong Xie, Ruobing Xie, Yankai Lin, Zhiyuan Liu, and Maosong Sun. ULTRAFEEDBACK: Boosting language models with scaled AI feedback. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 9722–9744. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/cui24f.html.

Miroslav Dudík, John Langford, and Lihong Li. Doubly robust policy evaluation and learning. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pp. 1097–1104, Madison, WI, USA, 2011. Omnipress. ISBN 9781450306195.

Mehrdad Farajtabar, Yinlam Chow, and Mohammad Ghavamzadeh. More robust doubly robust off-policy evaluation. In *International Conference on Machine Learning*, pp. 1447–1456. PMLR, 2018.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Kai Helli, David Schnurr, Noah Hollmann, Samuel Müller, and Frank Hutter. Drift-resilient tabPFN: In-context learning temporal distribution shifts on tabular data. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=p3tSEFMwpG.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: A transformer that solves small tabular classification problems in a second. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=cp5PvcI6w8_.

Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637(8045):319–326, 2025. doi: 10.1038/s41586-024-08328-6. URL https://doi.org/10.1038/s41586-024-08328-6.

Nan Jiang and Lihong Li. Doubly robust off-policy value evaluation for reinforcement learning. In *International conference on machine learning*, pp. 652–661. PMLR, 2016.

Nathan Kallus, Xiaojie Mao, Kaiwen Wang, and Zhengyuan Zhou. Doubly robust distributionally robust off-policy evaluation and learning. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 10598–10632. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/kallus22a.html.

Roger Koenker and Gilbert Bassett. Regression quantiles. *Econometrica*, 46(1):33–50, 1978.

Michael Laskin, Luyu Wang, Junhyuk Oh, Emilio Parisotto, Stephen Spencer, Richie Steigerwald, DJ Strouse, Steven Stenberg Hansen, Angelos Filos, Ethan Brooks, Maxime Gazeau, Himanshu Sahni, Satinder Singh, and Volodymyr Mnih. In-context reinforcement learning with algorithm distillation. In *NeurIPS 2022 Foundation Models for Decision Making Workshop*, 2022. URL https://openreview.net/forum?id=OridE7C5BP2.

Haanvid Lee, Tri Wahyu Guntara, Jongmin Lee, Yung-Kyun Noh, and Kee-Eung Kim. Kernel metric learning for in-sample off-policy evaluation of deterministic RL policies. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=plebgsdiiV.

Samuel Müller, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka, and Frank Hutter. Transformers can do bayesian inference. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=KSugKcbNf9.

Samuel Müller, Matthias Feurer, Noah Hollmann, and Frank Hutter. PFNs4BO: In-context learning for Bayesian optimization. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 25444–25470. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/muller23a.html.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744. Curran Associates, Inc., 2022. doi: 10.52202/068431-2011. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/b1efde53be364a73914f58805a001731-Paper-Conference.pdf.

Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. TabICLv2: A better, faster, scalable, and open tabular foundation model. 2026.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=HPuSIXJaa9.

Rafael Rafailov, Yaswanth Chittepu, Ryan Park, Harshit Sikchi, Joey Hejna, W. Bradley Knox, Chelsea Finn, and Scott Niekum. Scaling laws for reward model overoptimization in direct alignment algorithms. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 126207–126242. Curran Associates, Inc., 2024. doi: 10.52202/079017-4009. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/e45caa3d5273d105b8d045e748636957-Paper-Conference.pdf.

Noveen Sachdeva, Lequn Wang, Dawen Liang, Nathan Kallus, and Julian McAuley. Off-policy evaluation for large action spaces via policy convolution. In *Proceedings of the ACM Web Conference 2024*, WWW '24, pp. 3576–3585, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400701719. doi: 10.1145/3589334.3645501. URL https://doi.org/10.1145/3589334.3645501.

Yuta Saito and Thorsten Joachims. Off-policy evaluation for large action spaces via embeddings. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 19089–19122. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/saito22a.html.

Yuta Saito, Shunsuke Aihara, Megumi Matsutani, and Yusuke Narita. Open bandit dataset and pipeline: Towards realistic and reproducible off-policy evaluation, 2021. URL https://openreview.net/forum?id=GfWRFRuO51B.

Alex Strehl, John Langford, Lihong Li, and Sham M Kakade. Learning from logged implicit exploration data. In J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta (eds.), *Advances in Neural Information Processing Systems*, volume 23. Curran Associates, Inc., 2010. URL https://proceedings.neurips.cc/paper_files/paper/2010/file/c0f168ce8900fa56e57789e2a2f2c9d0-Paper.pdf.

Haoxiang Wang, Wei Xiong, Tengyang Xie, Han Zhao, and Tong Zhang. Interpretable preferences via multi-objective reward modeling and mixture-of-experts. In *Findings of Empirical Methods in Natural Language Processing*, 2024a.

Yu-Xiang Wang, Alekh Agarwal, and Miroslav Dudík. Optimal and adaptive off-policy evaluation in contextual bandits. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3589–3597. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/wang17a.html.

Zhilin Wang, Yi Dong, Olivier Delalleau, Jiaqi Zeng, Gerald Shen, Daniel Egert, Jimmy J. Zhang, Makesh Narsimhan Sreedhar, and Oleksii Kuchaiev. Helpsteer 2: Open-source dataset for training top-performing reward models. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024b. URL https://openreview.net/forum?id=PvVKUFhaNy.

Mengdi Xu, Yikang Shen, Shun Zhang, Yuchen Lu, Ding Zhao, Joshua Tenenbaum, and Chuang Gan. Prompting decision transformer for few-shot policy generalization. In *International Conference on Machine Learning*, pp. 24631–24645. PMLR, 2022.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Rui Yang, Ruomeng Ding, Yong Lin, Huan Zhang, and Tong Zhang. Regularizing hidden states enables learning generalizable reward model for llms. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 62279–62309. Curran Associates, Inc., 2024. doi: 10.52202/

079017-1990. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/71f7154547c748c8041505521ca433ab-Paper-Conference.pdf.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 46595–46623. Curran Associates, Inc., 2023. doi: 10.52202/075280-2020. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/91f18a1287b398d378ef22505bf41832-Paper-Datasets_and_Benchmarks.pdf.

APPENDIX CONTENTS

A Related Work **15**

B Algorithms **15**

C Experimental details **15**

C.1 Pool 15

C.2 Evaluation protocol 15

C.3 Estimators 16

D Prior-Coverage Diagnostics **16**

E Limitations **17**

Algorithm 1 PFN-OPE pretraining

Require: Response pool: prompts $\mathcal{X}_0$, pool policies $\{\widehat{g}\}_{g\in\mathcal{G}}$, stored rewards $R_1, \ldots, R_J$.

1: **for** each pretraining example **do**
2: Sample a task τ: prompt distribution $p(x \mid \tau)$, weights $\lambda_\tau, \omega_\tau$, LLMs g, g', and (η, α); form R_τ (Equation (5)), $\mu = \widehat{g}$, and π (Equation (7)).
3: Compute $V_\tau(\pi)$ over the pool (Equation (8)).
4: Draw n prompts $x_i \sim p(\cdot \mid \tau)$ and, for each, $a_i^\mu \sim \mu(\cdot \mid x_i)$, $a_i^\pi \sim \pi(\cdot \mid x_i)$, and $r_i^\mu = R_\tau(x_i, a_i^\mu)$; form $O = (\mathcal{D}_\mu, \mathcal{D}_\pi)$.
5: Set $C = V_\tau(\pi) - \bar{r}$ (Equation (2)).
6: Update θ on the loss of Equation (3) at (O, C).
7: **end for**

A Related Work

Prior-fitted networks. Prior-fitted networks (PFN) train a transformer on datasets sampled from a synthetic prior so that a single forward pass approximates posterior-predictive inference in context (Müller et al., 2022). PFNs have been successfully applied for tabular prediction (Hollmann et al., 2023; 2025; Helli et al., 2024), Bayesian optimization (Müller et al., 2023) and causal effect estimation (Balazadeh et al., 2025). In our work, we extend the idea to off-policy evaluation of LLMs.

Off-policy evaluation from text embeddings. Classical OPE estimators such as IPS are unable to handle continuous or large action space. Kernel-based IPS methods relax target policy distributions in IPS computation to a kernel (Lee et al., 2024; Sachdeva et al., 2024). However, it is non-trivial to extend these kernel-based IPS to high dimensional text embedding. MIPS (Saito & Joachims, 2022) applies IPS in a given embedding space, marginalizing over action variations that share them embedding. Cief et al. (2024) proposed using reward prediction loss to train the embedding for MIPS. However, these MIPS-based methods still need the per-dataset fit.

In-context bandits and reinforcement learning. In bandits and reinforcement learning settings, in-context learning has demonstrated efficient policy generalization (Xu et al., 2022) and posterior-sampling behavior (Laskin et al., 2022). These methods condition transformer-based policy models on trajectory data or interaction history to generalize to new tasks. Our work applies this in-context paradigm to off-policy evaluation: we train PFN-OPE to perform in-context value estimation, eliminating per-task fitting while enabling generalization to unseen policies and reward functions.

B Algorithms

Algorithm 2 gives every step of the procedure summarized in Algorithm 1.

C Experimental details

This appendix gives the evaluation protocol and the estimator settings behind Section 4.

C.1 Pool

Prompts and splits. Each dataset's prompts are split at random into a pretraining set and 1,024 evaluation prompts. Every evaluation task in Section 4 draws its prompts from the evaluation set, so no evaluation prompt, behavior response, or target response appears in pretraining.

Response generation and scoring. For each prompt and each LLM, seen or unseen, we sample $K = 8$ responses by nucleus sampling with temperature 0.7 and top-p 0.9. Every response is scored against its prompt by ArmoRM and GRM.

C.2 Evaluation protocol

Correction and rank groups. For a task with logged rewards $r_1, \ldots, r_n$, the correction is the prediction target $C = V_\tau(\pi) - \bar{r}$ of Equation (2). The standardized correction of Section 4.4 is

Algorithm 2 Generating pretraining data for PFN-OPE

Require: Prompts $\mathcal{X}_0$; LLMs $\mathcal{G}$; reward functions $R_1, \dots, R_J$; responses per prompt K.
Ensure: Pretraining examples (O, C), with the rewards of the target responses hidden from O.

Stage 1: build the response pool once

for each prompt $x \in \mathcal{X}_0$ and LLM $g \in \mathcal{G}$ **do**
  Sample K responses $a_g^{(1)}(x), \dots, a_g^{(K)}(x) \sim g(\cdot \mid x)$ and store them as $\mathcal{A}_g(x)$.
  Compute $R_j(x, a)$ for every stored response $a \in \mathcal{A}_g(x)$ and every $j = 1, \dots, J$.
end for
Standardize each reward over the pool, $\tilde{R}_j = (R_j - m_j)/s_j$.
Define the pool policy $\widehat{g}(\cdot \mid x)$ of each LLM as the uniform distribution on $\mathcal{A}_g(x)$ (Equation (6)).

Stage 2: sample one task and one observation per pretraining example

for each pretraining example **do**
  Draw a prompt distribution $p(\cdot \mid \tau)$ on $\mathcal{X}_0$ and the number of logged rows n.
  Draw evaluation weights $\lambda_\tau \sim \mathcal{N}\big(w, \mathrm{diag}(w) - ww^\top\big)$ and set $R_\tau(x, a) = \sum_{j=1}^{J} \lambda_{\tau j} \tilde{R}_j(x, a)$ (Equation (5)).
  Draw selection weights $\omega_\tau \in \Delta^{J-1}$, independently of λ_τ, and set $S_\tau(x, a) = \sum_{j=1}^{J} \omega_{\tau j} \tilde{R}_j(x, a)$.
  Draw a behavior LLM $g \in \mathcal{G}$ and a target LLM $g' \in \mathcal{G}$, and set $\mu = \widehat{g}$.
  Draw the policy mechanism: pool policy ($\eta = 0$, $\alpha = 1$) with probability 0.4, selected ($\eta > 0$, $\alpha = 1$) with probability 0.4, routed ($\alpha < 1$) with probability 0.2; draw η and α accordingly.
  Form the target policy (Equation (7)),

$$\nu_\eta(a \mid x) = \frac{\widehat{g}'(a \mid x)\, e^{\eta S_\tau(x,a)}}{\sum_{a'} \widehat{g}'(a' \mid x)\, e^{\eta S_\tau(x,a')}}, \qquad \pi = (1 - \alpha)\, \mu + \alpha\, \nu_\eta.$$

  Compute the target value over the pool (Equation (8)),

$$V_\tau(\pi) = \sum_{x \in \mathcal{X}_0} p(x \mid \tau) \sum_{a} \pi(a \mid x)\, R_\tau(x, a).$$

  Draw n prompts $x_1, \dots, x_n \sim p(\cdot \mid \tau)$.
  for $i = 1, \dots, n$ **do**
    Draw $a_i^\mu \sim \mu(\cdot \mid x_i)$ and $a_i^\pi \sim \pi(\cdot \mid x_i)$ independently.
    Set the logged reward $r_i^\mu = R_\tau(x_i, a_i^\mu)$.
  end for
  Form $\mathcal{D}_\mu = \{(x_i, a_i^\mu, r_i^\mu)\}_{i=1}^n$, $\mathcal{D}_\pi = \{(x_i, a_i^\pi)\}_{i=1}^n$, and $O = (\mathcal{D}_\mu, \mathcal{D}_\pi)$.
  Set $\bar{r} = \frac{1}{n} \sum_{i=1}^n r_i^\mu$ and $C = V_\tau(\pi) - \bar{r}$ (Equation (2)).
  Emit (O, C); O holds prompts, responses, and logged rewards only, and no task parameter.
end for

$C_{\mathrm{std}} = C/s_\mu$, where s_μ is the population standard deviation of $r_1, \dots, r_n$, floored at 10^{-6}. C is not the value gap $V_\tau(\pi) - V_\tau(\mu)$: it also contains the sampling deviation of $\bar{r}$, so it varies across tasks of one setting while $V_\tau(\pi)$ stays fixed. Within each setting (dataset, reward, behavior LLM, target LLM, and n), 100 candidate tasks were generated, ranked by $|C_{\mathrm{std}}|$.

C.3 Estimators

All estimators receive the same observation $O = (\mathcal{D}_\mu, \mathcal{D}_\pi)$; target rewards are used only to compute $V_\tau(\pi)$. Hyperparameters are fixed across all settings. DM-ridge, DM-kernel, and DM-MLP are refitted on every logged dataset. DM-TabPFN and DM-TabICLv2 use frozen pretrained weights. The TabPFN adapter projects the response embeddings ϕ_a of Section 3.3 to 128 dimensions with a fixed projection fitted on the response pool. MIPS estimates the density ratio between target and logged responses from their embeddings with prompt-grouped cross-fitting, and never sees a target reward. PFN-OPE's estimate is $\widehat{V}_\theta(\pi; O)$ of Equation (4): the logged mean plus the average of the 99 predicted quantiles of C.

D Prior-Coverage Diagnostics

This appendix defines the two axes of Figure 4. Both describe one fixed linear-ridge reward probe on the frozen embeddings of Section 3.3; neither is a PFN-OPE prediction error.

Probe and features. For a task with logged data $\mathcal{D}_\mu$ of size $n = 128$, the feature of a prompt–response pair is the concatenation of the unit-normalized prompt and response embeddings, $\phi(x, a) := \frac{1}{\sqrt{2}} \big[\phi_x(x)/\|\phi_x(x)\|;\ \phi_a(a)/\|\phi_a(a)\|\big]$. A ridge regressor is fitted to the logged rewards in $F = 4$

prompt-grouped folds, fold b using the 96 logged rows outside it, giving $\widehat{f}_b(x,a) = \widehat{\beta}_b^\top \phi(x,a) + \widehat{\gamma}_b$; the probe is the fold average $\bar{f} \coloneqq \frac{1}{F}\sum_{b=1}^{F} \widehat{f}_b$. The expectations $\mathbb{E}_\mu[\cdot]$ and $\mathbb{E}_\pi[\cdot]$ below are over the task's prompt distribution and its pool policies, computed exactly on the stored responses.

Probe correction error (vertical axis). The probe's estimate of the value gap is $\widehat{\Delta} \coloneqq \mathbb{E}_\pi[\bar{f}] - \mathbb{E}_\mu[\bar{f}]$ and the true gap is $\Delta \coloneqq V_\tau(\pi) - V_\tau(\mu)$. The diagnostic is the squared error of the estimate in units of the logged reward variance $\sigma_\mu^2 \coloneqq \mathrm{Var}_\mu(R_\tau)$,

$$B_\Delta^2 \coloneqq \frac{(\widehat{\Delta} - \Delta)^2}{\sigma_\mu^2}. \tag{10}$$

Writing $e \coloneqq \bar{f} - R_\tau$ for the probe's pointwise error, $B_\Delta^2 = (\mathbb{E}_\pi[e] - \mathbb{E}_\mu[e])^2/\sigma_\mu^2$: an error common to logged and target responses cancels, so B_Δ^2 is zero when the probe misjudges both values by the same amount and large only when its error differs between the two response sets. It mixes finite-sample error, regularization, and misspecification of the linear probe, and it is not the probe's error on the target value itself.

Value-relevant extrapolation (horizontal axis). Let $d \coloneqq \mathbb{E}_\pi[\phi] - \mathbb{E}_\mu[\phi]$ be the mean feature shift and $\widehat{\Sigma}_b$ the covariance of the features of fold b's 96 logged rows. The diagnostic is the ridge-regularized leverage of the shift, averaged over folds,

$$L_\Delta \coloneqq \frac{1}{F}\sum_{b=1}^{F} d^\top \left(\widehat{\Sigma}_b + \lambda_b I\right)^{-1} d, \tag{11}$$

with λ_b the ridge penalty of fold b. In the eigenbasis $\{(s_k, u_k)\}$ of $\widehat{\Sigma}_b$, $d^\top(\widehat{\Sigma}_b + \lambda_b I)^{-1} d = \sum_k (u_k^\top d)^2/(s_k + \lambda_b)$, so a shift along a direction in which the logged features barely vary (small s_k) counts more than the same shift along a direction they cover. The shift d is what a linear probe's value correction depends on, $\widehat{\Delta} = \left(\frac{1}{F}\sum_b \widehat{\beta}_b\right)^\top d$, which is why we call the leverage value-relevant.

E LIMITATIONS

Evaluation targets are other LLMs, Every evaluation target is the pool policy of one LLM, so the policy shifts we test are changes of LLM. The prior-coverage analysis (Section 4.6) shows that the tilted policies reach the kind of task a post-trained checkpoint would pose.

Rewards come from reward models. Every reward in pretraining and evaluation is the output of ArmoRM or GRM, so the setting is semi-synthetic: reward definitions are heads and combinations of reward models rather than human rubrics, and any bias of those models enters the pool. Human-rated logs would test the reward shift the introduction describes; we did not have them at the scale the pool requires.

The prior is built from three LLMs. Transfer to an unseen checkpoint of a seen family and to one unseen family (Section 4.3) is the evidence that the estimator generalizes beyond the pool's LLMs, and one unseen family is limited evidence. Widening the pool to more families and to actual post-training checkpoints is the natural extension, at the one-time cost of generating and scoring their responses.