LEARNING DYNAMICS-INFORMED LEARNING CURVE EXTRAPOLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

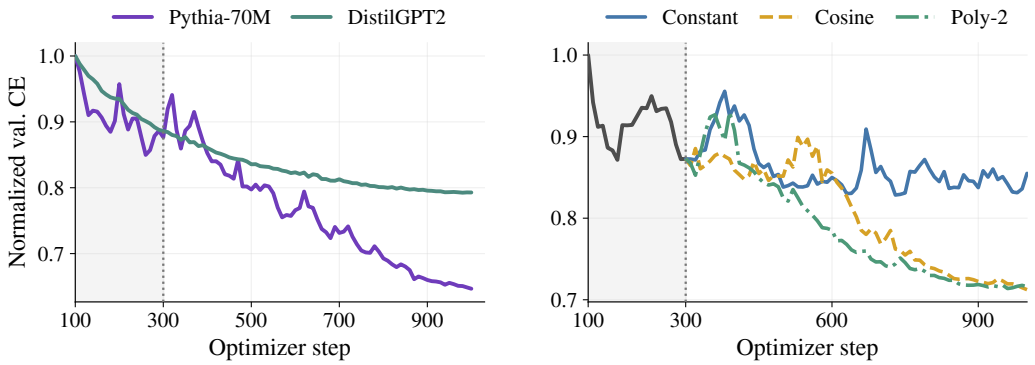
As the computational demands of machine learning training keep growing, how to obtain more high-quality models under limited resources has attracted increasing attention. Recently, Computational Resource Efficient Learning (CoRE-Learning) has been proposed to address this problem (Zhou, 2024). A key component of CoRE-Learning is learning curve extrapolation (LCE), which predicts future model performance from partially observed training trajectories to guide resource allocation. Existing LCE methods largely rely on curve families or empirical descriptors, leaving the underlying learning process implicit. Nonetheless, we observe that similar early curves can yield different future improvements under the same learning-rate schedule, while a shared checkpoint can evolve differently under different schedules. These observations motivate us to construct learning dynamics-informed generative priors for LCE. We propose Learning Dynamics-Informed PFN (LD-PFN), which builds a generative prior from a reduced spectral model of learning dynamics. A spectral surrogate represents the remaining loss as components with different decay rates, and learning-rate schedules map resource usage to effective optimization progress. We train a PFN on trajectories sampled from this prior for probabilistic extrapolation of validation cross-entropy, without task-specific updates or access to model internals at prediction time. Experiments across diverse fine-tuning scenarios and evaluation settings show that LD-PFN improves predictive accuracy and probabilistic quality over existing baselines.

1 INTRODUCTION

Machine learning under limited computational resources has attracted increasing attention, as resource constraints affect both the efficiency and effectiveness of model training. The recently proposed Computational Resource Efficient Learning (CoRE-Learning) framework (Zhou, 2024) formalizes how resource availability and scheduling influence learning performance. In such resource-constrained settings, effective allocation calls for anticipating how each training run will improve with additional computation. Learning curve extrapolation (LCE) supports these decisions by predicting future model performance from partially observed training trajectories (Mohr & van Rijn, 2024). For a single learning task, these predictions can guide the allocation of a limited budget across candidate hyperparameter configurations to find a high-performing model (Feurer & Hutter, 2019; Domhan et al., 2015). For multiple concurrent learning tasks, resource allocation instead seeks to improve training goodput or enable more tasks to meet their performance requirements before their deadlines (Wang et al., 2024; 2026). Across these settings, LCE can provide a common predictive basis for deciding which runs to continue and how much additional computation to allocate.

In practical applications, LCE predictions need to account for uncertainty: overly optimistic forecasts can waste computation, whereas overly pessimistic forecasts may prematurely discard promising runs. Classical approaches encode regularities in learning curves through parametric families, including power laws (Cortes et al., 1993; Frey & Fisher, 1999), while later Bayesian methods combine multiple basis functions (Domhan et al., 2015). Complementary approaches incorporate information about hyperparameter configurations (Swersky et al., 2014; Klein et al., 2017) or neural architectures (Ding et al., 2025). These developments provide useful structure for extrapolation, but most of them either impose overly restrictive prior assumptions or require substantial Bayesian posterior inference overhead when predictions are made sequentially (Adriaensen et al., 2023).

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.



(a) Different remaining improvement. **(b) Schedule-dependent evolution.**

Figure 1: Two factors can shape a learning curve's continuation. In (a), Pythia-70M and DistilGPT2 (OpenThoughts-114K, same cosine schedule) show similar early trajectories but different amounts of subsequent loss reduction, motivating a latent learning-state representation. In (b), starting from the same Pythia-70M checkpoint (Mixture-of-Thoughts, step 300), different future learning-rate schedules produce distinct continuations. Cross-entropy is normalized by step-100 loss; the shaded region marks the observed prefix available to the predictor.

More recently, Prior-Data Fitted Networks (PFNs) address this inference cost by learning approximate Bayesian prediction through offline pretraining (Müller et al., 2022). A PFN is trained on synthetic curves sampled from a prior and produces a posterior predictive distribution for a new curve prefix in a single forward pass. LC-PFN uses a parametric learning-curve prior (Adriaensen et al., 2023), while NSL-PFN constructs priors tailored to neural scaling laws (Lee et al., 2025). DA-LCE incorporates difficulty descriptors derived from early curve dynamics into conditional curve generation and prediction (Li & Zhao, 2026). Importantly, PFNs require the prior to support sampling rather than tractable posterior computation, leaving considerable freedom in how training curves are generated. Efficient inference, however, shifts an important part of the modeling problem to constructing an informative generative prior for the unobserved part of a learning curve.

Despite these advances, priors used by these PFN methods are primarily organized around curve templates or data-derived conditioning signals. Figure 1 highlights two factors that can shape future learning. First, under the same learning-rate schedule, similar early validation-loss trajectories can retain substantially different amounts of future improvement, which we associate with different latent *learning states*. Second, starting from the same checkpoint and training history, different future learning-rate schedules can produce distinct subsequent trajectories, indicating that the *learning schedule* can affect how optimization progress unfolds over future resources. The current learning state is latent and only partially revealed by the observed prefix, whereas the future schedule is specified by the training configuration and can therefore be used as additional information. Learning-dynamics analyses provide a natural basis for representing multi-timescale loss decay (Jacot et al., 2018; Bordelon et al., 2024; 2025), while recent work further shows that learning-rate schedules can systematically shape training-loss trajectories (Li et al., 2025; Tissue et al., 2025). We therefore investigate whether these dynamics can provide a more informative generative prior for LCE.

To this end, we propose Learning Dynamics-Informed PFN (LD-PFN). First, a reduced spectral surrogate defines a prior over plausible latent learning states by decomposing the remaining validation loss across multiple timescales. Second, learning-rate schedules shape the synthetic trajectories and are encoded as curve-level context for PFN extrapolation. Finally, we evaluate LD-PFN on real language-model fine-tuning trajectories and examine its use in a CoRE-Learning setting.

Organization. Section 2 presents preliminaries. Section 3 presents our approach. Section 4 reports experimental results of our proposed method. Section 5 concludes the paper.

2 PRELIMINARIES

In this section, we review PFNs, our prediction framework for LCE, and generalized Gauss–Newton (GGN) geometry, which we use to construct the learning-dynamics prior.

2.1 PRIOR-DATA FITTED NETWORKS

PFNs (Müller et al., 2022) approximate Bayesian prediction by training on synthetic datasets sampled from a prior over tasks. In the underlying Bayesian model, given a prior $p(f)$, observed data $\mathcal{D}$, and a query input x, prediction requires marginalizing over the posterior $p(f \mid \mathcal{D})$:

$$p(y \mid x, \mathcal{D}) = \int p(y \mid x, f)\, p(f \mid \mathcal{D})\, df. \tag{1}$$

Since this integral is unavailable in closed form, evaluating it per dataset requires costly numerical inference. PFNs avoid this repeated computation by pretraining a predictor q_ψ to directly approximate the posterior predictive distribution, minimizing the prior-data negative log-likelihood:

$$\psi^\star = \arg\min_\psi \mathbb{E}_{(\mathcal{D},x,y)\sim p_{\text{prior}}} \left[-\log q_\psi(y \mid x, \mathcal{D})\right], \tag{2}$$

where p_{prior} is the joint distribution of a context dataset $\mathcal{D}$ and a held-out pair (x, y) generated from the same prior-sampled function. After pretraining, the model remains frozen and predicts from a new dataset and query in a single forward pass, without task-specific parameter updates.

For learning-curve extrapolation, $\mathcal{D}$ consists of observed curve points, and query inputs specify future resource levels. Since prior-data training requires only samples from the generative model, rather than an explicit prior density or posterior computation, we can construct the training curves using a learning-dynamics surrogate, as described in Section 3.

2.2 GENERALIZED GAUSS–NEWTON GEOMETRY

Constructing a learning-dynamics prior requires a local model of how parameter changes affect the loss. A quadratic approximation provides this connection through the loss Hessian. For a nonlinear network, however, the full Hessian includes second-order output derivatives and need not be positive semidefinite. The GGN approximation (Schraudolph, 2002) simplifies this by linearizing the network output while retaining the loss curvature in output space.

Specifically, let $z_\theta(h)$ denote the network logits with parameters θ, and let $\ell(z, w)$ be the loss for target w. For the objective $L(\theta) = \mathbb{E}_{(h,w)}[\ell(z_\theta(h), w)]$, the GGN matrix is

$$G_\theta = \mathbb{E}_{(h,w)} \left[J_\theta(h)^\top H_\ell(h, w) J_\theta(h) \right], \tag{3}$$

where $J_\theta(h)$ is the Jacobian of the logits with respect to θ, and $H_\ell(h, w) \triangleq \nabla_z^2 \ell(z, w)\big|_{z=z_\theta(h)}$. Thus, G_θ is the Hessian at the expansion point of the objective obtained by replacing the network output with its first-order approximation.

For softmax cross-entropy, $H_\ell = \operatorname{diag}(\pi) - \pi\pi^\top \succeq 0$, where $\pi = \operatorname{softmax}(z_\theta(h))$, and hence $G_\theta \succeq 0$. Using G_θ as the curvature of a local quadratic surrogate therefore allows its quadratic term to be decomposed into nonnegative contributions along spectral directions. We use this property to construct the loss-dynamics representation in Section 3.2.

3 OUR APPROACH

In this section, we present LD-PFN for probabilistic extrapolation of validation loss curves. We first formulate learning-curve extrapolation as predicting a distribution over future validation losses from an observed curve prefix in Section 3.1. We then describe a spectral representation of loss decay in Section 3.2, and use this representation to construct a generative curve prior in Section 3.3. Finally, Section 3.4 introduces PFN pretraining and extrapolation from an observed curve prefix.

3.1 Problem Formulation

We consider learning-curve extrapolation for a training run with a prespecified training schedule configuration ζ. Each observation (r, L) records the validation cross-entropy L at cumulative training resource r, measured in optimization steps or processed tokens. Let r_{end} denote the planned resource budget for the complete run, and let $r_N \le r_{\mathrm{end}}$ be a target budget specified independently of future losses. We restrict attention to target budgets in the pre-overfitting regime, where the underlying validation-loss trend remains decreasing; individual observations may still fluctuate.

For ordered resource locations $r_1 < \cdots < r_N$, only the first $N_{\mathcal{C}} < N$ loss observations are available. We define the observed context and future targets as

$$\mathcal{C} = \{(r_i, L_i)\}_{i=1}^{N_{\mathcal{C}}}, \qquad \mathcal{T} = \{(r_i, L_i)\}_{i=N_{\mathcal{C}}+1}^{N}.$$

At extrapolation time, the target resource locations are known query points, while their losses remain unobserved. Our goal is to learn a predictive distribution $\widehat{q}_\psi(L \mid r, \mathcal{C}, \zeta)$ for each target resource level. We formulate this as maximizing the expected average log-likelihood of the future losses:

$$\max_{\psi} \mathbb{E}_{\mathcal{C},\mathcal{T},\zeta}\left[\frac{1}{|\mathcal{T}|}\sum_{(r,L)\in\mathcal{T}} \log \widehat{q}_\psi(L \mid r, \mathcal{C}, \zeta)\right], \tag{4}$$

where ψ denotes the predictor parameters. The following sections construct a learning-dynamics-informed prior and use it to train this predictor.

3.2 Learning-Dynamics Representation

To construct the curve prior, we use a local spectral surrogate to represent plausible latent learning states, expressing the remaining loss as components that decay at different rates.

Linearized model and GGN approximation. Let $\theta \in \mathbb{R}^P$ denote the trainable parameters, covering both full-model fine-tuning and restricted parameterizations such as LoRA (Hu et al., 2022). We write the validation cross-entropy (CE) as $L(\theta) = \mathbb{E}_{(h,w)}[-\log q_\theta(w \mid h)]$, where w is a target token and h is its context. Within this surrogate, consider a learning trajectory $\{\theta_t\}$ approaching a stationary reference $\theta^\star$ with $\nabla L(\theta^\star) = 0$, where t is the continuous time. We define the parameter residual $\Delta\theta_t = \theta_t - \theta^\star$, the current loss $L_t = L(\theta_t)$, and the reference loss $L_\infty = L(\theta^\star)$.

Following the local linearized view of neural-network training (Jacot et al., 2018; Chizat et al., 2019; Malladi et al., 2023), we approximate the local loss curvature by the generalized Gauss–Newton (GGN) matrix $G_{\theta^\star} \succeq 0$ (see Section 2.2), giving the local quadratic approximation

$$L_t - L_\infty \approx \frac{1}{2}\Delta\theta_t^\top G_{\theta^\star}\Delta\theta_t. \tag{5}$$

This approximation relates the remaining loss to the parameter residual through the local curvature.

Spectral learning dynamics and multi-time-scale loss decay. To describe how the residual evolves, we write the spectral decomposition as $G_{\theta^\star} = \sum_k \lambda_k v_k v_k^\top$, where $\lambda_k \ge 0$, and let $e_k(t) = v_k^\top \Delta\theta_t$ denote the projection of the parameter residual on mode k. For a schedule configuration ζ, let $\eta(t)$ denote the learning rate at time t. Motivated by continuous-time analyses (Jacot et al., 2018; Bordelon et al., 2024; 2025), we model the residual evolution with fixed-geometry gradient flow, which gives $\dot{e}_k(t) = -\eta(t)\lambda_k e_k(t)$. Since the validation curve is observed only from its first recorded point, we model the trajectory from the first observation at t_1. This gives

$$L_t - L_\infty \approx \sum_k u_k(t), \quad u_k(t) = u_k(t_1)\exp[-2\lambda_k S(t)], \quad S(t) = \int_{t_1}^{t} \eta(t')\,dt'. \tag{6}$$

Here, $u_k(t) = \lambda_k e_k(t)^2/2$ is the loss contribution of spectral mode k, and $\eta(t')$ is the learning rate. We interpret the reference floor L_∞, modal energies $\{u_k(t_1)\}$, and their decay rates $\{\lambda_k\}$ as a latent spectral state, while $S(t)$ controls how this state evolves under the future schedule.

To define a structured prior over such states, motivated by power-law spectral and source–capacity assumptions in previous work (Bordelon et al., 2020; Bahri et al., 2024; Bordelon et al., 2024; 2025), we parameterize the eigenvalues and modal energies at the first observation as

$$\lambda_k \propto k^{-\alpha}, \qquad u_k(t_1) \propto k^{-(1+\alpha\beta)}, \tag{7}$$

with $\alpha, \beta > 0$. Over an intermediate range of optimization progress, these assumptions yield

$$L_t - L_\infty \asymp S(t)^{-\beta}. \tag{8}$$

A derivation is provided in Appendix C. For a constant learning rate, $S(t) \propto t - t_1$, giving the power-law decay $L_t - L_\infty \asymp (t - t_1)^{-\beta}$. When the learning rate varies, the decay instead depends on the accumulated progress $S(t)$, motivating the learning-rate-dependent prior.

Since actual learning involves adaptive optimization and evolving curvature, this fixed-geometry model is used for prior construction rather than as an exact description of validation-loss dynamics.

3.3 From Dynamics to a Generative Curve Prior

We now use the representation in Section 3.2 to generate synthetic curves for the prediction setting in Section 3.1. Let r_i denote the cumulative resource at observation i, and let r_N be the target resource budget. We normalize the resource and loss as $x_i = r_i/r_N$ and $y_i = L_i/L_1$, so that $x_N = 1$ and $y_1 = 1$. During prior generation, we sample y_N as the normalized loss at this finite prediction horizon and use it to calibrate the total effective progress of the synthetic trajectory.

Latent learning state. We instantiate this state using the reference floor y_∞, modal energies $\{u_k(t_1)\}$, and modal decay rates $\{\lambda_k\}$. We sample a floor ratio ρ and set $y_\infty = \rho y_N$, then sample α and β and construct K spectral modes following Eq. 7. The modal energies $\{u_k(t_1)\}$ are rescaled to sum to $1 - y_\infty$, giving $y_1 = 1$ at the first observation. We then choose S_N so the trajectory reaches y_N, thereby calibrating the total effective progress over the modeled horizon.

Schedule-dependent evolution. Since absolute learning-rate magnitudes are task-dependent, we normalize the learning rate as $\bar{\eta}(s) = \eta(s)/\eta_{\text{peak}}$, where $s \in [0, 1]$ represents the position. We use it to model the shape of future learning trajectories, and define the cumulative exposure as

$$A(s) = \frac{\int_0^s \bar{\eta}(a)\, da}{\int_0^1 \bar{\eta}(a)\, da}, \tag{9}$$

Let r_{end} denote the budget at which the complete schedule ends, and let $\gamma = r_N/r_{\text{end}} \in (0, 1]$ be the fraction covered by the horizon. The resource coordinate x_i then corresponds to position γx_i on the schedule. We rescale the cumulative exposure over the modeled horizon as

$$S_i = S_N \frac{A(\gamma x_i) - A(\gamma x_1)}{A(\gamma) - A(\gamma x_1)}, \tag{10}$$

so $S_1 = 0$ and $S_i = S_N$ at $x_i = 1$. The future schedule therefore determines how the total progress S_N is distributed over the resource horizon. Substituting this into Eq. 6 gives the synthetic curve

$$y_i = y_\infty + \sum_{k=1}^{K} u_k(t_1) \exp[-2\lambda_k S_i]\,. \tag{11}$$

The prior samples linear, cosine, and constant schedules, together with their terminal-to-peak learning-rate ratios. Before PFN pretraining, we perturb each generated curve and renormalize it so that its first value remains one. The sampling distributions of the prior are provided in Appendix B.4.

3.4 PFN Probabilistic Extrapolation

We pretrain a PFN on the synthetic curves for probabilistic LCE, with schedule information derived from the prespecified schedule configuration ζ provided as additional context. The PFN takes observed resource–loss pairs, query resource locations, and the resulting schedule token as input.

Algorithm 1: Prior-data pretraining of LD-PFN

Input: Prior distributions in Appendix B.4; PFN $\widehat{q}_\psi$; batch size B; number of updates U.
Output: Pretrained PFN parameters ψ.
Initialize ψ;
for $u \leftarrow 1$ **to** U **do**
  Sample curve length N and context size $N_{\mathcal{C}}$;
  for $b \leftarrow 1$ **to** B **do**
    Sample ρ, y_N and set $y_\infty = \rho y_N$;
    Sample $\alpha, \beta, \{\varepsilon_k\}_{k=1}^K$ and construct $\{\lambda_k, u_k(t_1)\}_{k=1}^K$ using Eq. 7; rescale the modal energies and choose S_N to match y_N;
    Sample $m, \gamma, \bar{\eta}_{\text{end}}$ and construct $\phi(m, \bar{\eta}_{\text{end}}, \gamma)$ using Eq. 12;
    Set $x_i = i/N$ and compute $\{S_i\}_{i=1}^N$ using Eqs. 9–10;
    Generate $\{y_i\}_{i=1}^N$ using Eq. 11, add noise with sampled σ and renormalize to $y_1 = 1$;
    Prepend $\phi(m, \bar{\eta}_{\text{end}}, \gamma)$ as a curve-level context token;
  Compute $\frac{1}{B}\sum_{b=1}^B \mathcal{L}_{\text{PFN}}^{(b)}(\psi)$ using Eq. 13;
  Update ψ by AdamW;
return ψ;

Prior-data pretraining. For each synthetic curve, we sample a cutoff $N_{\mathcal{C}}$ with $2 \le N_{\mathcal{C}} < N$, and use the first $N_{\mathcal{C}}$ points as the observed prefix $\mathcal{C}$. We encode the schedule family m, terminal-to-peak ratio $\bar{\eta}_{\text{end}}$, and target-horizon progress γ into a curve-level token,

$$\phi(m, \bar{\eta}_{\text{end}}, \gamma) = E_{\text{sched}}(m) + W_\eta \bar{\eta}_{\text{end}} + W_\gamma \gamma + \mathbf{e}_{\text{LR}}. \tag{12}$$

Here, E_{sched} is a categorical embedding, W_η and W_γ are linear projections, and $\mathbf{e}_{\text{LR}}$ is a learned token-type embedding. The token is prepended to the PFN input as a curve-level context token.

Conditioned on $\mathcal{C}$ and the schedule token ϕ, the PFN predicts a distribution at each curve position. Let $\widehat{q}_\psi$ denote the predictive density parameterized by the PFN. The PFN was trained by minimizing the average negative log-likelihood over all curve positions, which aims to provide denser supervision and encourage consistency with the observed curve prefix:

$$\mathcal{L}_{\text{PFN}}(\psi) = -\frac{1}{N}\sum_{i=1}^{N} \log \widehat{q}_\psi(y_i \mid x_i, \mathcal{C}, \phi)\,. \tag{13}$$

At each pretraining step, we sample a batch of curves and update ψ using the average loss with AdamW optimizer. Algorithm 1 summarizes the procedure.

Learning-schedule-conditioned extrapolation. At inference time, we obtain m and $\bar{\eta}_{\text{end}}$ from the training configuration and γ from the target horizon, construct the same schedule token ϕ as in pretraining, and prepend it to the PFN input. The observed resources and losses are normalized by r_N and L_1 respectively, and the predictions are mapped back to the original loss scale using L_1.

4 Experiments

We evaluate LD-PFN on complete-suffix learning-curve extrapolation and its downstream use in CoRE-Learning resource allocation. We first describe the experimental setup in Section 4.1 and compare LD-PFN with existing baselines in point accuracy and predictive-distribution quality in Section 4.2. We then examine the contributions of spectral structure and learning schedule modeling in Section 4.3, followed by a CoRE-Learning resource-allocation study in Section 4.4.

4.1 Experimental Setup

Datasets. We use two collections of validation cross-entropy trajectories, covering LoRA-based SFT and heterogeneous fine-tuning settings.

Table 1: Complete-suffix extrapolation performance averaged over seven cutoffs (20%–80%). Best results are bold and second-best results are underlined; "–" denotes unavailable probabilistic metrics.

	LoRA-Train			OpenAssistant (OA)			**All-Curve Average**		
Method	**RMSLE** ↓	**LL** ↑	**CRPS** ↓	**RMSLE** ↓	**LL** ↑	**CRPS** ↓	**RMSLE** ↓	**LL** ↑	**CRPS** ↓
M1	.02198	–	–	.02138	–	–	.02166	–	–
M2	<u>.00597</u>	–	–	.01921	–	–	.01285	–	–
M3	.01149	–	–	.01883	–	–	.01531	–	–
M4	.00783	–	–	.01731	–	–	.01276	–	–
LC-PFN	.00778	3.258	<u>.00461</u>	.01458	2.818	**.00951**	<u>.01131</u>	3.029	<u>.00716</u>
NSL-PFN	.01041	<u>3.370</u>	.00554	<u>.01439</u>	<u>2.908</u>	.01159	.01248	<u>3.130</u>	.00868
CD-PFN	.02393	2.297	.01215	.02936	1.916	.02434	.02676	2.099	.01849
LD-PFN	**.00304**	**4.593**	**.00159**	**.01362**	**3.049**	<u>.00966</u>	**.00854**	**3.790**	**.00579**

(i) *LoRA-Train* contains 108 LoRA supervised fine-tuning runs across three base models—Gemma-2-2B, Llama-3.2-3B, and Qwen2.5-3B—and three instruction corpora: OpenHermes-2.5 (Teknium, 2023), UltraChat-200K (Ding et al., 2023), and OpenThoughts-114K (Guha et al., 2026). We use cumulative `seen_tokens` as the resource coordinate.

(ii) *OpenAssistant (OA)* (Köpf et al., 2023) contains 117 validation-loss trajectories from 16 public fine-tuning runs collected from Weights & Biases logs. We restrict the trajectories to the pre-overfitting regime and use the logged training step as the resource coordinate. Further dataset details are provided in Appendix B.1.

Baselines. We compare with three PFN-based extrapolators—LC-PFN (Adriaensen et al., 2023), NSL-PFN (Lee et al., 2025), and CD-PFN (Li & Zhao, 2026)—and the parametric M1–M4 baselines considered by Lee et al. (2025). PFN checkpoints are frozen, whereas M1–M4 are fitted to each observed prefix. All methods use only the observed prefix and future query coordinates; LD-PFN additionally receives its prespecified schedule configuration and target-horizon progress. Implementation-specific adaptations are detailed in Appendix B.3.

Evaluation protocol and metrics. We evaluate seven observation cutoffs from 20% to 80%. At each cutoff, the observed prefix is used to predict the complete remaining suffix, with no future observations revealed or appended to the context. We report RMSLE for point accuracy. Since M1–M4 are deterministic point estimators, LL and CRPS (Matheson & Winkler, 1976; Gneiting & Raftery, 2007) are reported only for the probabilistic PFN predictors, on the original validation-loss scale. Metrics are computed over each curve's suffix, averaged equally across curves at each cutoff, and then equally across cutoffs. For the All-Curve Average, curves from both datasets are weighted equally at each cutoff before averaging equally across cutoffs. Lower RMSLE and CRPS and higher LL are better; full metric definitions are provided in Appendix B.2.

4.2 Extrapolation Results

We evaluate complete-suffix extrapolation from partial observations against existing LCE baselines. Table 1 reports performance averaged over the seven observation cutoffs, while Figure 2 shows the cutoff-wise results. LD-PFN achieves the best average RMSLE and LL on both datasets and the best All-Curve Average for all three metrics.

On LoRA-Train, LD-PFN achieves the best RMSLE, LL, and CRPS at every observation cutoff. Averaged across cutoffs, it reduces RMSLE by approximately 49% and CRPS by 65% relative to the best competing baseline for each metric. On OA, the gains are smaller: LD-PFN achieves the lowest mean RMSLE and highest mean LL, with the lowest RMSLE at 60%–80% and CRPS at 50%–80%. The smaller margin on OA is consistent with its greater heterogeneity in training settings and learning trajectories relative to LoRA-Train. Its mean OA CRPS remains close to LC-PFN.

4.3 Ablation Studies

We ablate LD-PFN to examine the contributions of spectral structure and learning schedule modeling. *LD-PFN no spectral* retains the same multi-exponential loss representation but samples the

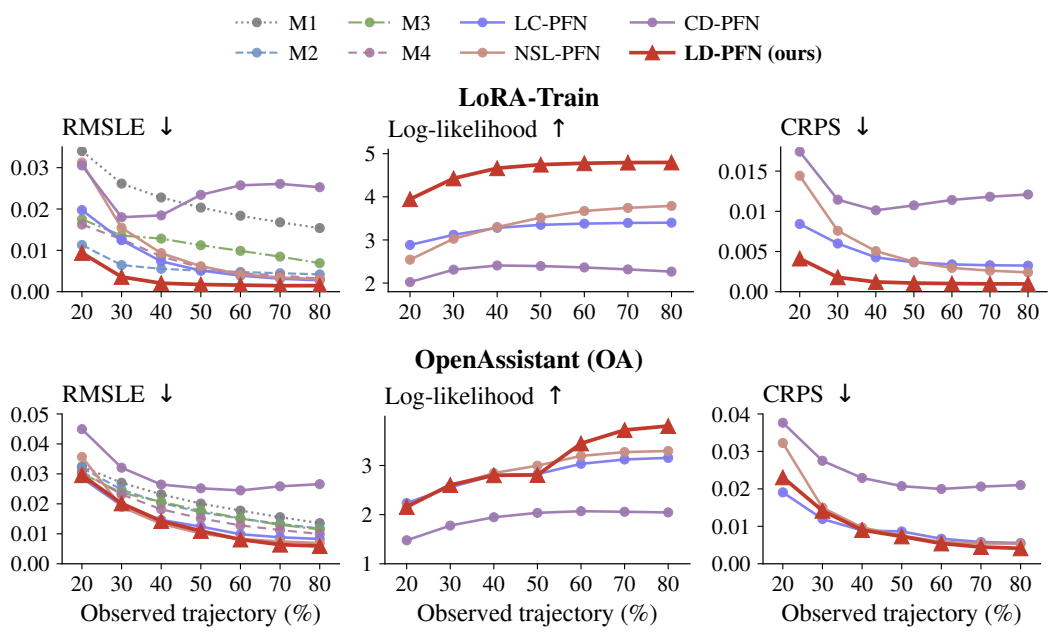


Figure 2: Cutoff-wise extrapolation performance on LoRA-Train and OpenAssistant.

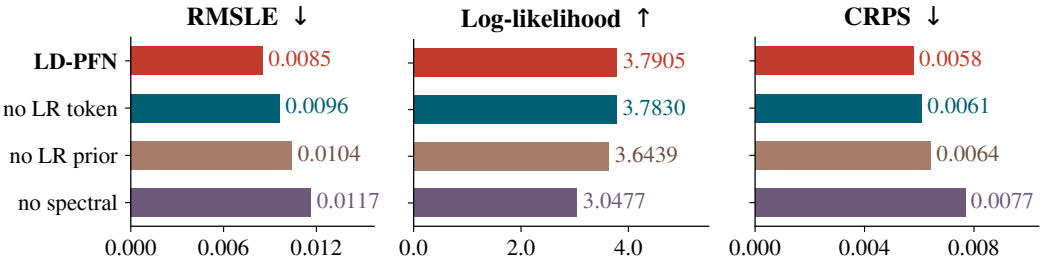


Figure 3: Ablation performance averaged over all curves and observation cutoffs.

modal decay rates and initial energies independently, removing the power-law relations in Eq. 7; this comparison tests whether the spectral structure provides an inductive bias beyond the expressiveness of a generic multi-timescale exponential mixture. *LD-PFN no LR token* removes explicit schedule conditioning while preserving the LR-dependent prior, and *LD-PFN no LR prior* further removes the LR-dependent optimization clock from the prior. All variants follow the same evaluation protocol.

Figure 3 shows that the complete LD-PFN achieves the best aggregate performance across all three metrics. Removing the spectral structure causes the largest degradation, supporting its importance in prior construction. Removing the conditioning schedule token degrades performance, with a further drop when LR-dependent prior dynamics are also removed. These results support the contributions of both spectral structure and learning schedule modeling.

4.4 Application to CoRE-Learning Resource Allocation

We further examine whether improved extrapolation can support better resource-allocation outcomes in CoRE-Learning (Zhou, 2024), where multiple deadline-constrained learning tasks compete for limited computational resources. LaiRA (Wang et al., 2026) uses a look-ahead feasibility check to conservatively detect whether each task can still reach its target loss before the deadline. We keep its allocation strategy fixed, while replacing only the predictor used in the look-ahead feasibility check with LD-PFN, LC-PFN, NSL-PFN, or DKF, thereby examining the effect of extrapolation quality within the same scheduling procedure.

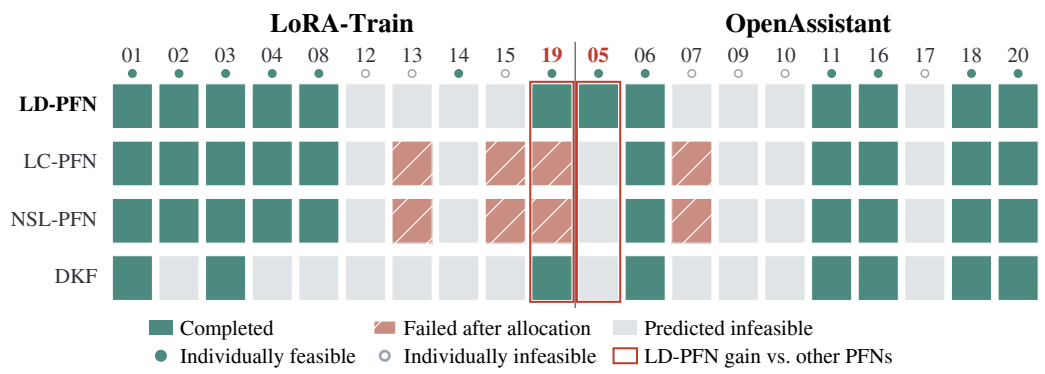


Figure 4: Per-task outcomes in the fixed CoRE-Learning scenario under the same LaiRA policy. Red outlines indicate tasks successful under LD-PFN but not under LC-PFN or NSL-PFN.

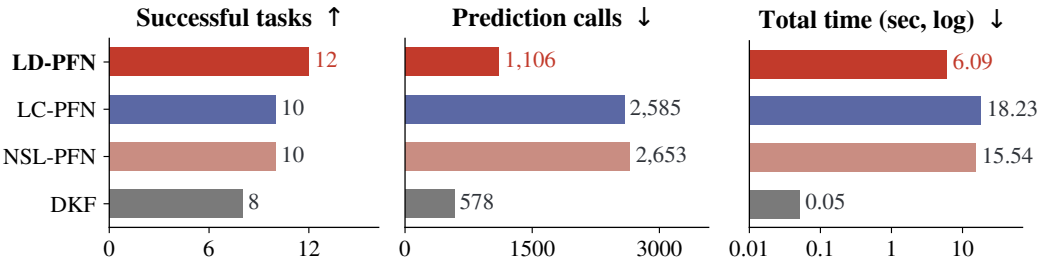


Figure 5: Successful tasks and look-ahead prediction overhead under the LaiRA policy. Prediction time is shown on a logarithmic scale; call counts depend on the scheduler trajectory.

We replay a fixed 20-task scenario with 10 LoRA-Train and 10 OpenAssistant trajectories, revealing the first 20% of each curve before scheduling. All variants use the same tasks, target losses, deadlines, and resource rates. For PFN predictors, LaiRA uses the 5th predictive percentile as an optimistic look-ahead feasibility estimate, while DKF uses its point prediction. Scenario construction and replay details are provided in Appendix B.5.

Figures 4 and 5 show that LD-PFN leads to more successful tasks while requiring less prediction overhead than the other PFN predictors. Under the fixed LaiRA policy, this suggests that improved extrapolation can translate into better allocation outcomes, which highlights the practical importance of reliable LCE. Note that this trace-driven evaluation is constructed from recorded learning curves and does not capture the full complexity of real-world scheduling; we therefore view it as evidence of downstream applicability rather than end-to-end system validation.

5 Conclusion

This paper focuses on Learning Curve Extrapolation (LCE), which aims to predict future loss trajectories from partially observed training curves. We introduced Learning Dynamics-Informed PFN (LD-PFN), a probabilistic framework for validation cross-entropy. Our method represents latent learning state uncertainty with a GGN-based spectral prior and incorporates the prescribed learning schedule in both prior generation and PFN conditioned extrapolation. Empirical results demonstrate the effectiveness of LD-PFN for LCE, and further show its utility in CoRE-Learning procedure.

In LD-PFN, the spectral learning-dynamics representation provides the modeling basis for PFN prior construction. An interesting direction for future research is to further study the spectral characteristics of realistic progress validation dynamics across different settings. Such empirical investigations may deepen our understanding of the connection between spectral learning dynamics and learning-curve evolution, and provide further insights for learning-dynamics-informed prior design.

AI USE STATEMENT

In this work, we used generative AI tools to help develop theoretical models or conceptual frameworks, formulate mathematical claims, assist in the writing of proofs, design or provide feedback on research methodology or experiments, implement methods, assist with translation, clean and reformat dataset, support qualitative and thematic data analysis, interpret results. We have not used generative AI tools to provide critical ingredients for proving mathematical claims or to propose or refine hypotheses; generating synthetic data sets is not applicable to this work. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide the implementation and evaluation details in Appendix B, which describes dataset construction, evaluation metrics, baseline adaptations, the LD-PFN prior configuration, and the CoRE-Learning setup. Appendix C provides the derivation of the asymptotic result used in our method.

REFERENCES

Steven Adriaensen, Herilalaina Rakotoarison, Samuel Müller, and Frank Hutter. Efficient Bayesian learning curve extrapolation using prior-data fitted networks. In *Advances in Neural Information Processing Systems 36 (NeurIPS)*, pp. 19858–19886, 2023.

Zahra Rahimi Afzal, Tara Esmaeilbeig, Mojtaba Soltanalian, and Mesrob I. Ohannessian. Linearization explains fine-tuning in large language models. In *Advances in Neural Information Processing Systems 38 (NeurIPS)*, pp. 146033–146062, 2025.

Theodoros Athanasiadis, Steven Adriaensen, Samuel Müller, and Frank Hutter. Tune my Adam, please! In *Proceedings of the Fourth International Conference on Automated Machine Learning (AutoML), Non-Archival Track*, 2025.

Yasaman Bahri, Ethan Dyer, Jared Kaplan, Jaehoon Lee, and Utkarsh Sharma. Explaining neural scaling laws. *Proceedings of the National Academy of Sciences*, 121(27):e2311878121, 2024.

Bowen Baker, Otkrist Gupta, Ramesh Raskar, and Nikhil Naik. Accelerating neural architecture search using performance prediction. In *ICLR 2018 Workshop Track*, 2018.

Blake Bordelon and Cengiz Pehlevan. Learning curves for SGD on structured features. In *Proceedings of the 10th International Conference on Learning Representations (ICLR)*, 2022.

Blake Bordelon, Abdulkadir Canatar, and Cengiz Pehlevan. Spectrum dependent learning curves in kernel regression and wide neural networks. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, pp. 1024–1034, 2020.

Blake Bordelon, Alexander Atanasov, and Cengiz Pehlevan. A dynamical model of neural scaling laws. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, pp. 4345–4382, 2024.

Blake Bordelon, Alexander Atanasov, and Cengiz Pehlevan. How feature learning can improve neural scaling laws. In *Proceedings of the 13th International Conference on Learning Representations (ICLR)*, 2025.

Abdulkadir Canatar, Blake Bordelon, and Cengiz Pehlevan. Spectral bias and task-model alignment explain generalization in kernel regression and infinitely wide neural networks. *Nature Communications*, 12(1):2914, 2021.

Lénaïc Chizat, Edouard Oyallon, and Francis Bach. On lazy training in differentiable programming. In *Advances in Neural Information Processing Systems 32 (NeurIPS)*, pp. 2933–2943, 2019.

Corinna Cortes, L. D. Jackel, Sara A. Solla, Vladimir Vapnik, and John S. Denker. Learning curves: Asymptotic values and rate of convergence. In *Advances in Neural Information Processing Systems 6 (NeurIPS)*, pp. 327–334, 1993.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. Enhancing chat language models by scaling high-quality instructional conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 3029–3051, 2023.

Yanna Ding, Zijie Huang, Xiao Shou, Yihang Guo, Yizhou Sun, and Jianxi Gao. Architecture-aware learning curve extrapolation via graph ordinary differential equation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 16289–16297, 2025.

Tobias Domhan, Jost Tobias Springenberg, and Frank Hutter. Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 3460–3468, 2015.

Samuel Dooley, Gurnoor Singh Khurana, Chirag Mohapatra, Siddartha V. Naidu, and Colin White. ForecastPFN: Synthetically-trained zero-shot forecasting. In *Advances in Neural Information Processing Systems 36 (NeurIPS)*, pp. 2403–2426, 2023.

Matthias Feurer and Frank Hutter. Hyperparameter optimization. In *Automated Machine Learning: Methods, Systems, Challenges*, pp. 3–33. Springer, 2019.

Lewis J. Frey and Douglas H. Fisher. Modeling decision tree performance with the power law. In *Proceedings of the Seventh International Workshop on Artificial Intelligence and Statistics (AISTATS)*, 1999.

Matilde Gargiani, Aaron Klein, Stefan Falkner, and Frank Hutter. Probabilistic rollouts for learning curve extrapolation across hyperparameter settings. In *6th ICML Workshop on Automated Machine Learning (AutoML)*, 2019.

Tilmann Gneiting and Adrian E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.

Etash Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Sprague, Ashima Suvarna, Benjamin Feuer, Leon Liangyu Chen, Zaid Khan, Eric Frankel, Sachin Grover, Caroline Choi, Niklas Muennighoff, Shiye Su, Wanjia Zhao, John Yang, Shreyas Pimpalgaonkar, Kartik Sharma, Charlie Ji, Yichuan Deng, Sarah Pratt, Vivek Ramanujan, Jon Saad-Falcon, Stutee Acharya, Jeffrey Li, Achal Dave, Alon Albalak, Kushal Arora, Blake Wulfe, Chinmay Hegde, Greg Durrett, Sewoong Oh, Mohit Bansal, Saadia Gabriel, Aditya Grover, Kai-Wei Chang, Vaishaal Shankar, Aaron Gokaslan, Mike Merrill, Tatsunori Hashimoto, Yejin Choi, Jenia Jitsev, Reinhard Heckel, Maheswaran Sathiamoorthy, Alex Dimakis, and Ludwig Schmidt. OpenThoughts: Data recipes for reasoning models. In *Proceedings of the 14th International Conference on Learning Representations (ICLR)*, 2026.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: A transformer that solves small tabular classification problems in a second. In *Proceedings of the 11th International Conference on Learning Representations (ICLR)*, 2023.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *Proceedings of the 10th International Conference on Learning Representations (ICLR)*, 2022.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural Tangent Kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems 31 (NeurIPS)*, pp. 8580–8589, 2018.

Aaron Klein, Stefan Falkner, Jost Tobias Springenberg, and Frank Hutter. Learning curve prediction with Bayesian neural networks. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*, 2017.

Andreas Köpf, Yannic Kilcher, Dimitri von Rütte, Sotiris Anagnostidis, Zhi Rui Tam, Keith Stevens, Abdullah Barhoum, Duc Nguyen, Oliver Stanley, Richárd Nagyfi, Shahul ES, Sameer Suri, David Glushkov, Arnav Dantuluri, Andrew Maguire, Christoph Schuhmann, Huu Nguyen, and Alexander Mattick. OpenAssistant conversations - democratizing large language model alignment. In *Advances in Neural Information Processing Systems 36 (NeurIPS)*, pp. 47669–47681, 2023.

Dongwoo Lee, Dong Bok Lee, Steven Adriaensen, Juho Lee, Sung Ju Hwang, Frank Hutter, Seon Joo Kim, and Hae Beom Lee. Bayesian neural scaling law extrapolation with prior-data fitted networks. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, pp. 33243–33275, 2025.

Jaehoon Lee, Lechao Xiao, Samuel S. Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. In *Advances in Neural Information Processing Systems 32 (NeurIPS)*, pp. 8570–8581, 2019.

Binghui Li, Fengling Chen, Zixun Huang, Lean Wang, and Lei Wu. Functional scaling laws in kernel regression: Loss dynamics and learning rate schedules. In *Advances in Neural Information Processing Systems 38 (NeurIPS)*, pp. 112258–112316, 2025.

Mengyang Li and Pinlong Zhao. Difficulty-aware learning curve extrapolation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 23021–23029, 2026.

Jihao Andreas Lin, Sebastian Ament, Maximilian Balandat, David Eriksson, José Miguel Hernández-Lobato, and Eytan Bakshy. Scalable Gaussian processes with latent Kronecker structure. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, pp. 37730–37744, 2025.

Kairong Luo, Haodong Wen, Shengding Hu, Zhenbo Sun, Maosong Sun, Zhiyuan Liu, Kaifeng Lyu, and Wenguang Chen. A multi-power law for loss curve prediction across learning rate schedules. In *Proceedings of the 13th International Conference on Learning Representations (ICLR)*, 2025.

Sadhika Malladi, Alexander Wettig, Dingli Yu, Danqi Chen, and Sanjeev Arora. A kernel-based view of language model fine-tuning. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, pp. 23610–23641, 2023.

James E. Matheson and Robert L. Winkler. Scoring rules for continuous probability distributions. *Management Science*, 22(10):1087–1096, 1976.

Felix Mohr and Jan N. van Rijn. Learning curves for decision making in supervised machine learning: A survey. *Machine Learning*, 113:8371–8425, 2024.

Samuel Müller, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka, and Frank Hutter. Transformers can do Bayesian inference. In *Proceedings of the 10th International Conference on Learning Representations (ICLR)*, 2022.

Samuel Müller, Matthias Feurer, Noah Hollmann, and Frank Hutter. PFNs4BO: In-context learning for Bayesian optimization. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, pp. 25444–25470, 2023.

Herilalaina Rakotoarison, Steven Adriaensen, Neeratyoy Mallik, Samir Garibov, Eddie Bergman, and Frank Hutter. In-context freeze-thaw Bayesian optimization for hyperparameter optimization. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, pp. 41982–42008, 2024.

Yi Ren and Danica J. Sutherland. Learning dynamics of LLM finetuning. In *Proceedings of the 13th International Conference on Learning Representations (ICLR)*, 2025.

Nicol N. Schraudolph. Fast curvature matrix-vector products for second-order gradient descent. *Neural Computation*, 14(7):1723–1738, 2002.

Kevin Swersky, Jasper Snoek, and Ryan Prescott Adams. Freeze-thaw Bayesian optimization. *ArXiv preprint arXiv:1406.3896*, 2014.

Teknium. OpenHermes 2.5: An open dataset of synthetic data for generalist LLM assistants. Hugging Face dataset, 2023.

Howe Tissue, Venus Wang, and Lu Wang. Scaling law with learning rate annealing. In *Advances in Neural Information Processing Systems 38 (NeurIPS)*, pp. 101068–101104, 2025.

Stijn Verdenius, Andrea Zerio, and Roy L. M. Wang. LaT-PFN: A joint embedding predictive architecture for in-context time-series forecasting. *ArXiv preprint arXiv:2405.10093*, 2024.

Tom Julian Viering, Steven Adriaensen, Herilalaina Rakotoarison, and Frank Hutter. From epoch to sample size: Developing new data-driven priors for learning curve prior-fitted networks. In *Proceedings of the Third International Conference on Automated Machine Learning (AutoML 2024), Workshop Track*, 2024.

Jing Wang, Miao Yu, Peng Zhao, and Zhi-Hua Zhou. Learning with adaptive resource allocation. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, pp. 52099–52116, 2024.

Jing Wang, Xi-Tong Liu, and Zhi-Hua Zhou. CoRE-learning with look-ahead and immediate resource allocation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 26294–26301, 2026.

Xingjin Wang, Howe Tissue, Lu Wang, Linjing Li, and Daniel Dajun Zeng. Learning dynamics in continual pre-training for large language models. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, pp. 64525–64548, 2025.

Martin Wistuba and Tejaswini Pedapati. Learning to rank learning curves. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, pp. 10303–10312, 2020.

Martin Wistuba, Arlind Kadra, and Josif Grabocka. Supervising the multi-fidelity race of hyperparameter configurations. In *Advances in Neural Information Processing Systems 35 (NeurIPS)*, pp. 13470–13484, 2022.

Cheng Yan, Steven Adriaensen, and Tom Julian Viering. Context-aware learning curve extrapolation with prior-data fitted networks. In *Proceedings of the 2nd ICML Workshop on Foundation Models for Structured Data (FMSD)*, 2026.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, and Zheyan Luo. LlamaFactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL), System Demonstrations*, pp. 400–410, 2024.

Zhi-Hua Zhou. Learnability with time-sharing computational resource concerns. *National Science Review*, 11(10):nwae204, 2024.

A Related Work

In this section, we review related work on learning curve extrapolation, prior-data fitted networks and learning dynamics for understanding how training trajectories evolve.

Learning curve extrapolation. Learning curve extrapolation (LCE) has been widely used in hyperparameter optimization (HPO), neural architecture search (NAS), model selection, and resource-constrained learning (Mohr & van Rijn, 2024). In HPO, partial learning curves are used to terminate poor configurations or decide which configuration should receive the next training or fidelity increment (Swersky et al., 2014; Domhan et al., 2015; Klein et al., 2017; Gargiani et al., 2019; Wistuba et al., 2022; Lin et al., 2025); these predictions are typically used to support early stopping, configuration selection, or budget allocation. In NAS and model selection, learning-curve predictors estimate the final performance of partially trained architectures (Baker et al., 2018), directly learn their relative ranking (Wistuba & Pedapati, 2020), or incorporate architecture information to extrapolate curves for model ranking (Ding et al., 2025); their primary use is cross-candidate selection, for which final-performance prediction or relative ordering is often sufficient. In CoRE-Learning, multiple deadline-constrained learning tasks compete for shared limited computational resources, where LARA and LaiRA use learning-curve extrapolation or look-ahead feasibility estimation as predictive components and further perform resource allocation across tasks (Zhou, 2024; Wang et al., 2024; 2026). Our work instead focuses on probabilistic within-run extrapolation, predicting the complete remaining validation-loss trajectory up to a prespecified future resource budget.

Prior-data Fitted Networks. Prior-data Fitted Networks (PFNs) are Transformer-based models for in-context Bayesian prediction, learning to perform probabilistic inference from synthetic data sampled from a predefined prior (Müller et al., 2022). Beyond curve extrapolation, PFNs have been applied to tabular prediction (Hollmann et al., 2023), Bayesian optimization (Müller et al., 2023) and time-series forecasting (Dooley et al., 2023; Verdenius et al., 2024), illustrating their use as a general framework for amortized Bayesian inference. For extrapolation, LC-PFN uses a parametric learning-curve prior (Adriaensen et al., 2023), FT-PFN additionally conditions on hyperparameter configurations for freeze-thaw optimization (Rakotoarison et al., 2024), and NSL-PFN constructs specialized functional priors for neural scaling-law (Lee et al., 2025). Subsequent variants further incorporate optimizer configurations or learner identity as contextual information (Athanasiadis et al., 2025; Yan et al., 2026), while data-driven extensions construct priors from empirical learning curves (Viering et al., 2024). These studies highlight the importance of both prior-data construction and contextual information for PFN extrapolation. LD-PFN instead couples a learning-dynamics-derived prior with explicit conditioning on the prescribed learning-rate schedule.

Learning dynamics. Learning-dynamics theory studies how neural-network predictions and losses evolve under optimization, often through tractable local or large-width approximations. The generalized Gauss–Newton (GGN) matrix provides an early local-geometry view through a positive-semidefinite curvature representation of neural-network objectives, including cross-entropy losses (Schraudolph, 2002). Neural tangent kernel (NTK) theory later characterizes gradient-based training in function space and shows that different kernel eigendirections can evolve at different rates (Jacot et al., 2018; Lee et al., 2019); related analyses, including linearized views, have further studied pretrained-model fine-tuning and parameter-efficient adaptation (Malladi et al., 2023; Ren & Sutherland, 2025; Afzal et al., 2025). Separately, spectral analyses relate the eigenspectrum and target alignment to mode-wise learning timescales and aggregate learning curves (Bordelon et al., 2020; Canatar et al., 2021; Bordelon & Pehlevan, 2022), while more recent studies connect spectral structure to loss scaling with training time, model size, and data size, and further investigate how feature learning changes these dynamics (Bahri et al., 2024; Bordelon et al., 2024; 2025). Recent work further models how learning-rate schedules shape complete loss trajectories through cumulative or effective optimization progress and annealing-aware dynamics (Luo et al., 2025; Wang et al., 2025; Li et al., 2025; Tissue et al., 2025). Compared with these works, which primarily characterize or predict learning dynamics themselves, LD-PFN uses a reduced spectral-and-schedule dynamics model to construct PFN prior-data and condition probabilistic extrapolation.

B Implementation Details

B.1 Dataset Processing and Construction

(i) *LoRA-Train*. The 108 runs cover three base models, three instruction corpora, LoRA ranks $\{2, 4, 8, 16\}$, and three training-data budgets per corpus. After filtering, each corpus is globally shuffled with seed 42 and split into fixed validation, test, and training sets. Training subsets are nested and defined by supervised assistant-label-token budgets using a common tokenizer, while each run is trained with LLaMA-Factory (Zheng et al., 2024) using its target model's tokenizer and chat template. We use maximum sequence lengths of 4,096 for OpenHermes and UltraChat and 8,192 for OpenThoughts; for OpenThoughts, overlength examples are removed and a fixed 512-example validation subset is used during training. The resulting curves contain 49–107 validation observations. We use cumulative `seen_tokens` as the resource coordinate, which is distinct from the supervised-token budget used to construct the training subset.

(ii) *OpenAssistant*. We construct the OA collection from public OpenAssistant fine-tuning runs on Weights & Biases. For each run, we collect validation histories whose metric name is either `lm loss validation` or matches `eval/*_loss`, using `train/global_step` as the resource coordinate when available and the W&B logging step otherwise. We remove nonnumeric or non-finite observations, sort the remaining points by resource, and retain the last downloaded value at duplicated steps. We exclude predominantly rising trajectories and retain curves with at least 15 observations. No smoothing, interpolation, resampling, or loss-scale transformation is applied. This procedure yields 117 curves from 16 runs used in our experiments.

B.2 Evaluation Metrics

For a curve with N observations and cutoff $c \in \{0.2, 0.3, \ldots, 0.8\}$, we use $N_C = \min\{\max(4, \lfloor cN \rfloor), N-1\}$ observations as context. Let $\mathcal{T}$ denote the suffix and $\widehat{L}_i$ the prediction at $(r_i, L_i) \in \mathcal{T}$. Future points are predicted from the same observed context; ground-truth values in $\mathcal{T}$ are not revealed sequentially. We measure point-prediction accuracy by

$$\mathrm{RMSLE} = \sqrt{\frac{1}{|\mathcal{T}|} \sum_{(r_i, L_i) \in \mathcal{T}} \left(\log \widehat{L}_i - \log L_i\right)^2}. \tag{14}$$

For RMSLE, LD-PFN, LC-PFN, and NSL-PFN use predictive medians; CD-PFN uses its inverse-transformed mean utility; and M1–M4 use deterministic fits.

For probabilistic predictors, let p_i and F_i denote the predictive density and cumulative distribution, respectively, at r_i. We report the mean pointwise predictive log-likelihood

$$\mathrm{LL} = \frac{1}{|\mathcal{T}|} \sum_{(r_i, L_i) \in \mathcal{T}} \log p_i(L_i), \tag{15}$$

and the continuous ranked probability score (CRPS)

$$\mathrm{CRPS} = \frac{1}{|\mathcal{T}|} \sum_{(r_i, L_i) \in \mathcal{T}} \int_{-\infty}^{\infty} \left[F_i(z) - \mathbb{I}\{z \geq L_i\}\right]^2 dz. \tag{16}$$

Both probabilistic metrics are evaluated after mapping the prediction back to the original validation-loss space, where LL is a mean of marginal predictive log-densities instead of the joint likelihood.

For each dataset and cutoff, we first compute the corresponding metric over the future suffix of each curve and then average equally across curves. Dataset-level aggregates are then averaged equally across the evaluated cutoffs. For the All-Curve Average, all curves from both datasets are weighted equally at each cutoff before averaging equally across cutoffs.

B.3 Baseline Adaptation and Implementation

All methods receive the same observed prefix and future resource queries, without access to future losses. PFN weights are frozen during extrapolation.

Table 2: Principal parameters of the LD-PFN prior.
LogitN, LogU, and Cat denote the logit-normal, log-uniform, and categorical distributions, respectively.

Parameter	Sym.	Dist. / value	Parameter	Sym.	Dist. / value
Endpoint	y_N	$\mathrm{LogitN}(2.10, 1.05^2)$	Noise scale	σ	$\mathrm{LogU}(10^{-3}, 3\times10^{-2})$
Floor ratio	ρ	$\mathrm{LogitN}(2.30, 1.70^2)$	Sched. coverage	γ	$\mathcal{U}(0,1)$
Mode number	K	128	Sched. family	m	Cat(cosine : 0.4, linear : 0.3, constant : 0.3)
Spectral decay	α	$\mathcal{U}(0.1, 3.0)$			
Energy decay	β	$\mathrm{LogU}(0.005, 0.25)$	Terminal ratio	$\bar{\eta}_{\mathrm{end}}$	0.1 $\mathrm{Beta}(1,9)$ (decay); 1 (constant)
Modal log-noise	ε_k	$\mathcal{N}(0, 0.8^2)$, i.i.d.	Seq. length	N	$\mathrm{round}(\mathrm{LogU}(45, 450))$

(i) *LC-PFN*. We use the official LC-PFN implementation and checkpoint (Adriaensen et al., 2023). Following its original inference procedure, the resource input is mapped to its native scale as

$$x_i = 100\,\frac{r_i}{r_N},$$

while the original LC-PFN loss normalization is retained for y_i. For LL and CRPS, we additionally renormalize its predictive density to the validation-loss range.

(ii) *NSL-PFN*. We follow the official NSL-PFN inference procedure (Lee et al., 2025), using the same input scaling as during its pretraining:

$$x_i = 0.01 + 0.99\frac{r_i - r_1}{r_N - r_1}, \qquad y_i = \frac{L_i}{\max_{k\in\mathcal{C}} L_k}.$$

(iii) *CD-PFN*. For CD-PFN (Li & Zhao, 2026), we use a frozen local reproduction and retain its original inference preprocessing. The curve inputs are constructed as

$$x_i = 1 + 44\frac{r_i - r_1}{r_N - r_1}, \qquad y_i = \frac{1}{1 + L_i},$$

where the loss transformation maps validation loss to the performance scale used by CD-PFN. The difficulty descriptor is computed from the observed prefix as an additional conditioning signal. We keep its native categorical-mean point prediction unchanged. To additionally report LL and CRPS, we convert its categorical output into a continuous predictive density.

(iv) *Parametric baselines*. M1–M4 follow the functional forms used by NSL-PFN (Lee et al., 2025), with numerical stabilization for degenerate cases in M3 and M4. Losses are normalized by the maximum observed prefix loss. As deterministic predictors, they are evaluated only with RMSLE.

B.4 Details of the Generative Curve Prior

This section provides the sampling distributions and implementation details of the generative curve prior introduced in Section 3.3. Table 2 summarizes the prior configuration used for PFN pretraining.

Spectral sampling. For $k = 1, \ldots, K$, we set $\lambda_k = k^{-\alpha}$ and sample the modal energies as

$$\widetilde{u}_k = k^{-(1+\alpha\beta)}\exp(\varepsilon_k), \qquad \varepsilon_k \sim \mathcal{N}(0, 0.8^2). \tag{17}$$

After setting $y_\infty = \rho y_N$, we normalize the modal energies as $u_k(t_1) = (1 - y_\infty)\widetilde{u}_k / \sum_j \widetilde{u}_j$. The terminal effective progress S_N is then computed to match y_N as described in Section 3.3.

Schedule profiles. For simplicity, we omit the effect of warmup and write the normalized learning-rate profiles as

$$\bar{\eta}_m(s) = \begin{cases} 1, & m = \text{constant}, \\ 1 - (1 - \bar{\eta}_{\mathrm{end}})s, & m = \text{linear}, \\ \bar{\eta}_{\mathrm{end}} + \dfrac{1 - \bar{\eta}_{\mathrm{end}}}{2}\big(1 + \cos(\pi s)\big), & m = \text{cosine}, \end{cases} \tag{18}$$

for $s \in [0, 1]$. These profiles define $A(s)$ in Eq. 9.

Table 3: Frozen task configuration used in the CoRE-Learning replay.

LoRA-Train

Task	Rate (10^8 tokens/time)	DDL	Target
01	1.556	8.94	1.01
02	1.498	8.89	0.78
03	1.541	8.95	1.01
04	1.553	8.91	0.74
08	1.481	8.87	0.72
12	1.556	6.22	0.61
13	1.480	8.89	0.99
14	1.470	4.56	0.75
15	1.547	8.92	0.66
19	1.476	8.93	0.70

OpenAssistant (OA)

Task	Rate (10^4 steps/time)	DDL	Target
05	1.057	8.89	0.74
06	1.081	8.91	0.71
07	1.094	8.90	0.61
09	1.036	8.89	2.97
10	1.079	8.95	1.80
11	1.097	8.87	0.90
16	1.025	8.94	0.91
17	1.073	8.90	0.57
18	1.100	8.95	1.31
20	1.012	8.95	0.71

Sequence sampling and perturbation. For a sampled sequence length N, we use $x_i = i/N$. For each pretraining batch, we sample $c \sim \mathcal{U}(0.2, 0.8)$ and set $N_{\mathcal{C}} = \max\{2, \min(\lfloor cN \rfloor, N-1)\}$. After generating the curve, we add independent Gaussian observation noise with standard deviation σ, clip losses to remain positive, and divide by the first perturbed value so that $y_1 = 1$.

B.5 CoRE-Learning Scenario Setup

We evaluate the CoRE-Learning application through a fixed replay scenario constructed from the learning curves used in our experiments.

LaiRA resource allocation. LaiRA (Wang et al., 2026) combines a look-ahead feasibility check with immediate performance evaluation for resource allocation in CoRE-Learning. After an initial exploration stage, each scheduling round first uses a look-ahead predictor to remove tasks that are unlikely to reach their target loss before the deadline. It then applies its UCB-based weighted least-squares immediate evaluation to rank the remaining tasks, allocates the current time slice to the highest-ranked task, and adapts the next time-slice length based on recent prediction errors. After the allocated task advances, its new validation-loss observation is obtained, and the same look-ahead, immediate-evaluation, and allocation procedure is repeated in the next scheduling round.

PFN-enhanced LaiRA. We replace only LaiRA's look-ahead predictor while keeping all other components fixed. At scheduling time, LaiRA determines r_j, the maximum resource reachable by task j within its deadline . Each PFN is queried once at this resource and returns the lower bound of its central 90% predictive interval on the original loss scale,

$$\widehat{L}_j = Q_{0.05}(L \mid r_j, \mathcal{C}_j)\,, \tag{19}$$

where $\mathcal{C}_j$ denotes the observations currently available for task j, and $Q_{0.05}$ denotes the predictive 5th percentile, giving an optimistic feasibility estimate. If $\widehat{L}_j$ still exceeds the task's target loss, LaiRA marks the task as infeasible. LC-PFN and NSL-PFN use the same quantile interface, whereas the original DKF look-ahead uses its point prediction. LD-PFN additionally conditions this prediction on the prescribed schedule configuration ζ_j.

Configuration and implementation. We use a fixed 20-task scenario with 10 trajectories from LoRA-Train and 10 from OA, randomly selected using a fixed seed of 42. All tasks arrive at time zero, and the first 20% of each trajectory is used for exploration, which is available before scheduling. Table 3 reports the fixed resource rates, deadlines (DDL), and target validation losses used in the experiment. Note that LoRA-Train rates are measured in seen tokens per unit time, whereas OA rates are measured in optimization steps per unit time. PFN predictors run on GPU and DKF on CPU; reported prediction time excludes model initialization.

C Analysis of Asymptotic Power-Law Behavior

We now make precise the asymptotic argument underlying Eq. 8. Following the modal dynamics in Section 3.2, the excess validation loss is approximated by

$$L_t - L_\infty \approx \sum_{k\geq 1} u_k(t) = \sum_{k\geq 1} u_k(t_1)\exp[-2\lambda_k S(t)]\,, \tag{20}$$

where $S(t) = \int_{t_1}^{t} \eta(t')\,dt'$ and $S(t_1) = 0$. For the asymptotic calculation, we consider the unperturbed power-law backbone of the prior,

$$\lambda_k \sim c_\lambda k^{-\alpha}, \qquad u_k(t_1) \sim c_u k^{-(1+\alpha\beta)}, \qquad \alpha, \beta > 0, \tag{21}$$

for constants $c_\lambda, c_u > 0$.

Substituting Eq. 21 into Eq. 20 gives, up to multiplicative constants independent of t,

$$L_t - L_\infty \asymp \sum_{k=1}^{\infty} k^{-(1+\alpha\beta)} \exp\left[-2c_\lambda S(t) k^{-\alpha}\right]. \tag{22}$$

For large $S(t)$, we approximate the spectral sum by

$$L_t - L_\infty \asymp \int_1^\infty k^{-(1+\alpha\beta)} \exp\left[-2c_\lambda S(t) k^{-\alpha}\right] dk. \tag{23}$$

With the change of variables $a = 2c_\lambda S(t) k^{-\alpha}$, we obtain

$$L_t - L_\infty \asymp [S(t)]^{-\beta} \int_0^{2c_\lambda S(t)} a^{\beta-1} e^{-a}\, da. \tag{24}$$

For $\beta > 0$, the integral converges to $\Gamma(\beta)$ as $S(t) \to \infty$. Therefore,

$$L_t - L_\infty \asymp [S(t)]^{-\beta}\,. \tag{25}$$

The same scaling can be seen from the transition mode $k_\star$ satisfying

$$\lambda_{k_\star} S(t) \asymp 1. \tag{26}$$

Since $\lambda_k \propto k^{-\alpha}$,

$$k_\star \asymp [S(t)]^{1/\alpha}\,. \tag{27}$$

Using $u_k(t_1) \propto k^{-(1+\alpha\beta)}$, the unresolved spectral tail satisfies

$$L_t - L_\infty \approx \sum_{k>k_\star} u_k(t_1) \asymp \int_{k_\star}^\infty k^{-(1+\alpha\beta)} dk \asymp k_\star^{-\alpha\beta} \asymp [S(t)]^{-\beta}\,, \tag{28}$$

which recovers Eq. 25.

The above derivation considers an idealized infinite spectrum. In the implemented prior, only K modes are retained, so the power-law regime applies while $1 \ll k_\star \ll K$. At sufficiently late progress, the slowest retained mode dominates,

$$L_t - L_\infty \sim u_K(t_1)\exp[-2\lambda_K S(t)]\,. \tag{29}$$

Thus, a finite spectral mixture exhibits an intermediate power-law regime followed by an exponential cutoff. The implemented prior uses a finite S_N calibrated to the sampled endpoint y_N, so the asymptotic result characterizes its scaling structure without requiring every sampled trajectory to reach the asymptotic regime.

Finally, the learning-rate schedule enters the surrogate through $S(t)$. For a constant learning rate, $S(t) \propto t - t_1$, giving $L_t - L_\infty \asymp (t-t_1)^{-\beta}$. A time-varying schedule instead changes the mapping from training time to accumulated optimization progress.