Difficulty-Aware Learning Curve Extrapolation

Mengyang Li[1], **Pinlong Zhao**[2*]

[1]Tianjin Key Laboratory of Wireless Mobile Communications and Power Transmission, Tianjin Normal University, Tianjin, China, 300387
[2]School of Cyberspace, Hangzhou Dianzi University, Hangzhou, China, 310018
limengyang@tjnu.edu.cn, pinlongzhao@hdu.edu.cn

Abstract

Learning Curve Extrapolation (LCE) is a critical technique for accelerating automated machine learning by terminating unpromising training runs early. Recent state-of-the-art methods have improved predictive accuracy by incorporating contextual information, such as neural network architecture. However, these approaches, whether context-agnostic or architecture-aware, still operate under the implicit assumption of a uniform task landscape. They overlook a pivotal, complementary factor: the intrinsic difficulty of the learning task itself. This oversight leads to significant performance degradation, especially for tasks whose learning dynamics diverge from the model's priors. In this work, we argue that task difficulty is a crucial yet neglected dimension for robust LCE. We introduce Difficulty-Aware Learning Curve Extrapolation (DA-LCE), which explicitly conditions its predictions on task complexity. Our core contributions are threefold: (1) We propose a transparent, rule-based method to quantify task difficulty from early learning curve dynamics, eliminating the need for external meta-features. (2) We design a novel data generation pipeline using conditional diffusion models to create high-fidelity, difficulty-conditioned synthetic training data. (3) We introduce a Transformer-based predictor that leverages difficulty information to achieve superior accuracy across diverse benchmarks. Extensive experiments demonstrate that our approach significantly outperforms both difficulty-agnostic and architecture-aware baselines, with task difficulty emerging as a powerful conditioning signal whose impact matches or exceeds that of model architecture.

Code — https://github.com/limengyang1992/DA-LCE

Introduction

The computational cost of Automated Machine Learning (AutoML) is often prohibitive, as it requires training numerous configurations for tasks like Neural Architecture Search (NAS) (Ren et al. 2021) and Hyperparameter Optimization (HPO) (He, Zhao, and Chu 2021; Hutter, Kotthoff, and Vanschoren 2019). Learning Curve Extrapolation (LCE) is a key strategy to mitigate this cost by predicting a model's final performance from its initial training progress (Mohr and van Rijn 2024). This enables the early termination of unpromising runs, leading to significant computational savings (Zoph and Le 2017; Bergstra and Bengio 2012).

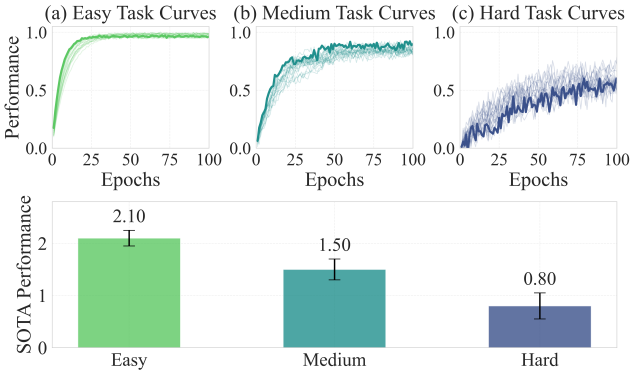

Figure 1: Learning curves exhibit diverse dynamics across task difficulties, motivating adaptive rather than uniform extrapolation strategies.

The landscape of LCE research has evolved significantly. Early approaches focused on fitting parametric models, such as power-law functions or their weighted combinations, to the observed curve segments (Domhan et al. 2015; Frey and Fisher 1999). While interpretable, these models often lack the flexibility to capture the complex, non-convex nature of modern deep learning curves (Nakkiran et al. 2021). This led to the rise of more powerful, non-parametric methods. Sequence models like LSTMs (Baker et al. 2017; Graves 2012) offered greater expressiveness, while Bayesian methods, including Bayesian Neural Networks (Klein et al. 2017), provided a principled way to quantify predictive uncertainty. Recently, the state-of-the-art has been advanced by Prior-data Fitted Networks (PFNs) (Adriaensen et al. 2023), which leverage the power of Transformers to learn an amortized approximation of Bayesian inference from a synthetic prior, achieving both high accuracy and efficiency (Vaswani et al. 2017).

Concurrently, another research thrust has focused on enriching LCE models with contextual information, recognizing that learning curves are not generated in a vacuum. A prominent example is the integration of architectural infor-

*Corresponding author.

mation, where Graph Neural Networks (GNNs) (Veličković et al. 2018) are used to encode a model's topology, which then guides the extrapolation process (Ding et al. 2025; Siems et al. 2020). These context-aware methods have compellingly demonstrated that incorporating information about the learning agent (i.e., the model architecture) improves prediction (White et al. 2023).

However, despite these parallel advancements, a critical dimension remains largely unexplored. Existing methods, whether context-agnostic or architecture-aware, implicitly assume a uniform task landscape, forcing a single model to handle fundamentally different learning dynamics. We argue that this overlooks a complementary and pivotal factor: the intrinsic difficulty of the learning task itself. To demonstrate this gap, our preliminary study (Figure 1) reveals that learning curves from tasks of varying difficulty exhibit distinct patterns, and that a state-of-the-art LCE model's performance systematically degrades as task difficulty increases. This highlights the urgent need for LCE models that can adapt to the specific challenge at hand (Peng and Pan 2023).

This paper introduces a novel framework, Difficulty-Aware Learning Curve Extrapolation (DA-LCE), to directly address this challenge. To our knowledge, this is the first approach that combines a self-contained, rule-based proxy for dynamic task difficulty with a high-fidelity, conditional data generation process. Instead of relying on external meta-features (Vanschoren et al. 2014), our method computes a difficulty proxy directly from early curve dynamics. Recognizing that the efficacy of any PFN-based approach hinges on the quality of its training prior, we address a common limitation: the discrepancy between synthetic data generated from simple parametric functions and the complex nature of real-world learning curves. To resolve this, we introduce a novel data generation process. We leverage a conditional diffusion model, trained on a diverse corpus of real data, to create a high-fidelity synthetic dataset. In this dataset, each curve's shape is explicitly conditioned on our difficulty proxy, providing a rich and structured prior that enables our core model to learn nuanced, task-adaptive extrapolation strategies. Our main contributions are threefold:

- We introduce a novel, self-contained method to quantify an Early-stage Difficulty Proxy directly from learning curve observations. This rule-based approach avoids reliance on external meta-features and provides a robust, interpretable signal for task adaptation based on initial training dynamics.
- We design a new data generation pipeline to bridge the gap between synthetic priors and real-world learning dynamics. By training a conditional diffusion model on real data, we create a high-fidelity synthetic prior where curve shapes are explicitly conditioned on our difficulty proxy.
- We propose the CD-PFN, a Transformer-based model that leverages the difficulty proxy to make task-adaptive predictions. Through extensive experiments, we demonstrate that our framework sets a new state-of-the-art, significantly outperforming both difficulty-agnostic and architecture-aware baselines, particularly on challenging and non-standard tasks.

Related Work

The Evolution of LCE Models. The goal of LCE is to forecast a model's final performance from its initial training trajectory. Early approaches fitted parametric functions (e.g., power-law, logistic) to partial curve segments (Domhan et al. 2015; Frey and Fisher 1999). While interpretable, these methods lacked the flexibility to capture the non-monotonic dynamics of modern deep learning (Nakkiran et al. 2021). To address this, non-parametric methods emerged. Sequence models like LSTMs (Baker et al. 2017; Graves 2012) enhanced expressiveness, while Bayesian Neural Networks (Klein et al. 2017; Swersky, Snoek, and Adams 2014) offered principled uncertainty estimates.

Recently, Prior-data Fitted Networks (PFNs) (Adriaensen et al. 2023) have set a new state-of-the-art by leveraging Transformers (Vaswani et al. 2017) to approximate Bayesian inference through large-scale synthetic priors (Wang et al. 2024). However, these methods typically assume a universal prior, applying the same extrapolation strategy across tasks of varying difficulty. This agnostic treatment of task complexity limits robustness, a gap our work addresses through explicit difficulty modeling.

Incorporating Context: From Architectures to Tasks. Parallel to advances in LCE model design, researchers have explored enriching models with contextual information. Architecture-aware methods, for example, encode network topology using GNNs to guide extrapolation (Ding et al. 2025; Siems et al. 2020; White et al. 2023). These methods have demonstrated that conditioning on the structure of the *learner* can significantly improve predictive accuracy (Ning et al. 2020).

However, while these methods focus on *how* a model learns, they largely overlook *what* is being learned—the intrinsic difficulty of the task itself. Prior studies on task hardness have relied on static, external dataset meta-features (Vanschoren et al. 2014), which may not generalize well to unseen tasks or be available in practical scenarios. In contrast, we propose a dynamic, self-contained difficulty proxy computed directly from early-stage learning curves. This approach, rooted in meta-learning principles (Finn, Abbeel, and Levine 2017; Peng and Pan 2023), introduces a novel and scalable form of task-aware conditioning. Unlike prior methods, our framework requires no external meta-data, enabling more practical and adaptive extrapolation.

Methodology

Our proposed framework, DA-LCE, enhances extrapolation accuracy by conditioning its predictions on the intrinsic difficulty of the learning task. The entire framework, illustrated in Figure 2, is designed to be self-contained and operates in two distinct phases.

Offline Training Phase

The offline phase encompasses all preparatory steps required to train our core extrapolation model. This computationally intensive phase, which includes learning a generative prior and training the extrapolation network, is performed only once before deployment.

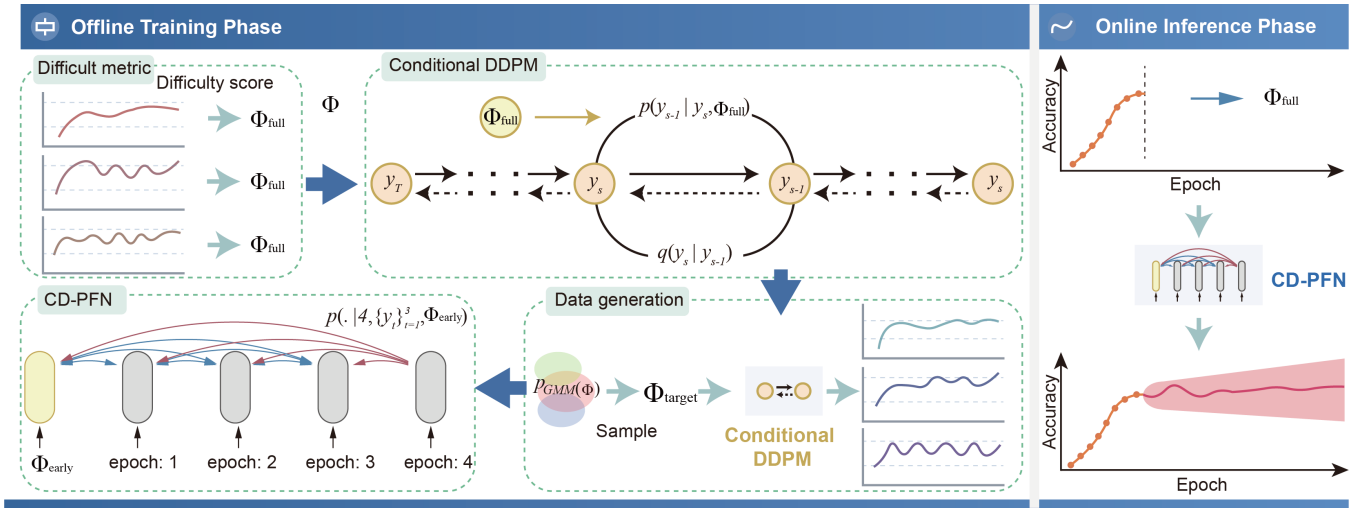

Figure 2: The DA-LCE framework, illustrating the distinct offline training and online inference phases.

Characterizing Task Difficulty via Early-stage Dynamics. A cornerstone of our framework is the ability to quantify a task's intrinsic difficulty directly from an initial segment of its performance curve. Instead of relying on external, often unavailable, dataset meta-features, we propose a more direct approach: we construct a difficulty proxy by characterizing the observable learning dynamics. We posit that a task's inherent difficulty is directly reflected in the behavior of the learning curve it produces; for instance, a difficult task might manifest as slow progress, high volatility, or complex, non-linear learning phases. This principle allows us to create a conditioning signal that is both informative and self-contained.

To operationalize this, we draw upon foundational principles from learning curve analysis (Mohr and van Rijn 2024; Frey and Fisher 1999), which suggest that curve behavior can be effectively summarized by its overall rate of progress, its structural complexity (deviation from simple patterns), and its local stability (noisiness). We instantiate these principles by designing three rule-based indicators extracted directly from the observed data. Given a partial learning curve $\{y_t\}_{t=1}^{T}$, we compute a three-dimensional dynamics vector, which serves as our difficulty proxy: $\Phi_{\text{early}} = [\phi_{\text{prog}}, \phi_{\text{nonlin}}, \phi_{\text{vol}}]$. Each component is defined as follows:

- Rate of Progress (ϕ_{prog}): This indicator captures the most fundamental aspect of learning—the speed of improvement. A higher rate of progress often signals an easier task or a more effective model configuration. We measure this as the average improvement per observed epoch, a definition that remains applicable during online inference where the total training length is unknown:

$$\phi_{\text{prog}} = \frac{y_T - y_1}{T} \tag{1}$$

- Non-Linearity (ϕ_{nonlin}): Real-world learning is rarely linear. This indicator quantifies the curve's structural complexity by measuring its deviation from a simple linear trajectory. High non-linearity can indicate complex learning phases, such as plateaus followed by sharp improvements, which are common in non-convex optimization (He, Zhao, and Chu 2021). We quantify this using the mean squared error relative to a straight line connecting the start and end points:

$$\phi_{\text{nonlin}} = \frac{1}{T}\sum_{t=1}^{T}\left(y_t - \left(y_1 + \frac{y_T - y_1}{T-1}(t-1)\right)\right)^2 \tag{2}$$

- Volatility (ϕ_{vol}): The stability of the training process is another crucial signal. High volatility, characterized by large fluctuations between consecutive epochs, can be symptomatic of issues like large learning rates, small batch sizes, or inherent stochasticity in the task itself (Baker et al. 2017). We measure this using the standard deviation of the first-order differences of the curve:

$$\phi_{\text{vol}} = \text{StdDev}(\{y_t - y_{t-1}\}_{t=2}^{T}) \tag{3}$$

To ensure these indicators are on a comparable scale and to facilitate stable model training, each component of the resulting vector is standardized using the mean and standard deviation pre-computed from a large, diverse reference set of real-world learning curves. This entire characterization process is applied consistently throughout our framework. Crucially, through comprehensive empirical analysis, we validate that this early-stage proxy, Φ_{early}, exhibits strong and stable correlations with full-trajectory dynamics, justifying its use for conditioning our generative model.

Conditional Synthetic Data Generation. The performance of PFN-based methods is fundamentally dependent on the fidelity of the synthetic prior used for training. To build a robust and generalizable prior, we first assemble a large corpus of real-world learning curves, $\mathcal{D}_{\text{real}}$, drawn from three complementary sources: HPO-Bench (Eggensperger et al. 2021) for hyperparameter optimization trajectories,

JAHS-Bench-201 (Bansal et al. 2022) for joint architecture-hyperparameter search curves, and parametric baselines following Domhan et al. (Domhan et al. 2015). These sources do not contain any of the evaluation tasks used in our benchmarks (LCBench, NAS-Bench-201, Taskset, PD1), and employ different task setups, search spaces, and model families to avoid direct leakage and ensure fair generalization assessment. To overcome the limitations of simple parametric functions, which often fail to capture the rich diversity of real-world learning curves, we introduce a novel data generation pipeline. This pipeline leverages a conditional Denoising Diffusion Probabilistic Model (DDPM) (Ho, Jain, and Abbeel 2020) to create a high-fidelity, dynamics-conditioned synthetic corpus.

Modeling the Distribution of Real-world Dynamics. To ensure our synthetic data is grounded in reality, we first model the distribution of dynamics from a large corpus of N real-world learning curves, $\mathcal{D}_{\text{real}}$. For each complete curve in this corpus, we compute its full-trajectory dynamics vector, Φ_{full}. This results in a dataset of N 3D data points.

Instead of directly using these discrete, empirically observed dynamics vectors to train our generative model, we opt for a more robust strategy by fitting a continuous probability model to them. Specifically, we fit a Gaussian Mixture Model (GMM) to these points to capture the underlying distribution $p(\Phi)$. This choice is motivated by two key advantages. First, the GMM smooths and generalizes the empirical distribution, allowing us to sample new target dynamics vectors, Φ_{target}, that are statistically representative of real-world scenarios but not explicitly present in our finite corpus $\mathcal{D}_{\text{real}}$. This enhances the diversity of our synthetic prior. Second, the fitted GMM acts as a controllable generative prior. It provides a complete, parametric representation of the dynamics landscape, which could in future work enable targeted data augmentation by oversampling from specific modes of the distribution, such as those corresponding to particularly challenging tasks. During the generation phase, we sample new target dynamics vectors, $\Phi_{\text{target}} \sim p_{\text{GMM}}(\Phi)$, to guide the synthesis of our high-fidelity curves.

Training the Conditional Diffusion Model. We employ a DDPM tailored for 1D sequence data, treating a learning curve $y_0 \in \mathbb{R}^M$ as a signal. The DDPM consists of a fixed forward noising process, q, which gradually adds Gaussian noise to the data over S discrete timesteps according to a variance schedule $\{\beta_s\}_{s=1}^S$:

$$q(y_s|y_{s-1}) = \mathcal{N}(y_s; \sqrt{1-\beta_s}y_{s-1}, \beta_s\mathbf{I}) \tag{4}$$

The reverse process is parameterized by a neural network, ϵ_θ, which is trained to predict the noise ϵ that was added at timestep s, given the noisy curve y_s and a conditioning signal.

To make the generation conditional on our dynamics vector, we augment the denoising network to accept the 3D vector Φ as an additional input, denoted as $\epsilon_\theta(y_s, s, \Phi)$. We use a 1D U-Net architecture (Ronneberger, Fischer, and Brox 2015) as the backbone for ϵ_θ. The dynamics vector Φ is first projected into a high-dimensional embedding. This embedding, along with an embedding of the timestep s, is then infused into each residual block of the U-Net via adaptive normalization layers (Perez et al. 2018). This mechanism allows the dynamics vector to modulate the entire denoising process. The network is trained on pairs of real curves and their full-trajectory dynamics vectors, $(y_0, \Phi_{\text{full}}) \sim \mathcal{D}_{\text{paired}}$, by optimizing the simplified DDPM objective:

$$\mathcal{L}_{\text{diff}} = \mathbb{E}_{s,y_0,\Phi_{\text{full}},\epsilon}\left[||\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_s}y_0 + \sqrt{1-\bar{\alpha}_s}\epsilon, s, \Phi_{\text{full}})||^2\right] \tag{5}$$

where y_s is the noisy version of the clean curve y_0, $s \sim \text{Uniform}(1, ..., S)$, $\epsilon \sim \mathcal{N}(0, \mathbf{I})$, and $\bar{\alpha}_s = \prod_{i=1}^{s}(1-\beta_i)$.

Generating the High-Fidelity Synthetic Corpus. Once the conditional diffusion model ϵ_θ is trained, it becomes a powerful and controllable generator. We can synthesize a new, realistic learning curve y_{synth} for any desired dynamics profile Φ_{target} by starting with pure Gaussian noise $y_S \sim \mathcal{N}(0, \mathbf{I})$ and iteratively applying the reverse denoising step for $s = S, ..., 1$:

$$y_{s-1} = \frac{1}{\sqrt{\alpha_s}}\left(y_s - \frac{1-\alpha_s}{\sqrt{1-\bar{\alpha}_s}}\epsilon_\theta(y_s, s, \Phi_{\text{target}})\right) + \sigma_s\mathbf{z} \tag{6}$$

where $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I})$ if $s > 1$, else $\mathbf{z} = 0$, $\alpha_s = 1 - \beta_s$, and σ_s is the standard deviation of the reverse step. By repeatedly sampling Φ_{target} from our fitted GMM and running this generation process, we create a large and diverse synthetic dataset $\mathcal{D}_{\text{synth}}$. This high-fidelity corpus, where each curve's shape is explicitly tied to a realistic dynamics profile, forms the foundation for training our CD-PFN.

Conditional Difficulty-aware PFN. The final component of our offline pipeline is the Conditional Difficulty-aware Prior-data fitted Network (CD-PFN), the model responsible for the actual extrapolation. The CD-PFN is a Transformer-based architecture (Vaswani et al. 2017) trained on our high-fidelity synthetic corpus, $\mathcal{D}_{\text{synth}}$. Its goal is to perform amortized Bayesian inference, approximating the posterior predictive distribution $p(y_{t'}|\{y_t\}_{t=1}^T, \Phi_{\text{early}})$ in a single, efficient forward pass.

Rationale for Explicit Conditioning. A crucial design choice in our framework is the use of an explicit, global conditioning signal, Φ_{early}, rather than relying on the Transformer to implicitly learn these dynamics from the raw sequence of curve points. This choice is motivated by two primary advantages. First, it provides a powerful inductive bias. By explicitly feeding the model summary statistics about the curve's behavior, we guide its attention towards salient dynamic properties, making the learning process more efficient and data-driven. Second, this mechanism helps to disentangle representations. It allows the model to learn a function that maps specific dynamic characteristics to tailored extrapolation strategies, i.e., $f(\Phi_{\text{early}}) \rightarrow$ Strategy, rather than learning a single, monolithic strategy for all curve types. This is particularly beneficial for improving robustness on atypical curves whose dynamics deviate from the majority.

Architecture and Training Protocol. The input to the CD-PFN is a sequence of embeddings representing the observed epoch-performance pairs, $\{(t_i, y_i)\}_{i=1}^T$. Our key architectural modification is the introduction of a global condi-

Model	LCBench		NAS-Bench-201		Taskset		PD1	
	MAPE ↓	NLL ↓	MAPE ↓	NLL ↓	MAPE ↓	NLL ↓	MAPE ↓	NLL ↓
LSTM	0.218	-	0.387	-	0.412	-	0.445	-
NODE	0.108	0.228	0.205	0.341	0.318	0.452	0.398	0.562
NSDE	0.104	0.221	0.199	0.335	0.312	0.445	0.391	0.553
LC-GODE	<u>0.094</u>	<u>0.198</u>	<u>0.176</u>	<u>0.295</u>	<u>0.282</u>	<u>0.401</u>	<u>0.371</u>	<u>0.508</u>
LC-PFN	0.101	0.209	0.192	0.312	0.297	0.418	0.385	0.524
DA-LCE (Ours)	**0.073**	**0.158**	**0.141**	**0.239**	**0.224**	**0.329**	**0.304**	**0.421**

Table 1: Overall performance comparison on all benchmarks.

tioning mechanism. The 3D dynamics vector, Φ_{early}, is projected into a high-dimensional embedding $\mathbf{e}_\Phi$ using a dedicated MLP. This embedding is then prepended to the input sequence as a special `[DYNAMICS]` token. Through the self-attention mechanism, this global context vector can be accessed by all subsequent tokens, allowing it to modulate the representation of the entire learning curve. The final output of the CD-PFN, p_θ, is a categorical distribution over a set of K discretized performance bins.

The model is trained on a vast number of simulated extrapolation tasks, making it robust to the uncertainty of early-stage observations. For each training instance, we sample a full synthetic curve from our diffusion-based prior, select a random cutoff point T, and a random future target epoch $t' > T$. The model is then tasked with predicting the performance $y_{t'}$ given the partial curve $\{y_i\}_{i=1}^{T}$ and its corresponding difficulty proxy Φ_{early}, which is computed on-the-fly from this partial input. The training objective is to minimize the batch-averaged cross-entropy loss between the model's predicted distribution and the true, discretized value of $y_{t'}$. By repeatedly executing this protocol, the CD-PFN learns a robust mapping from early-stage indicators to future performance, effectively internalizing how to extrapolate from partial and uncertain dynamic signals.

Experiments

We conduct a comprehensive set of experiments to validate the effectiveness of our proposed Difficulty-Aware Learning Curve Extrapolation (DA-LCE) framework.

Experimental Setup

Benchmarks. To ensure a robust and comprehensive evaluation, we test our framework across a diverse suite of four large-scale learning curve benchmarks, encompassing a wide range of model architectures, task types, and data modalities. Specifically, we use: (1) LCBench (Zimmer, Lindauer, and Hutter 2021), which contains learning curves for MLPs on various tabular classification tasks; (2) NAS-Bench-201 (Dong and Yang 2020), a standard benchmark for CNN-based architectures on image classification; (3) Taskset (Metz et al. 2020), which provides curves from RNN models on text classification, known for exhibiting more complex and volatile dynamics; and (4) PD1 (Wang et al. 2024), a recent benchmark featuring Transformer models on protein sequence and translation tasks. This diverse selection allows us to rigorously assess the generalization capabilities of all methods, particularly on tasks of varying intrinsic difficulty.

Baselines. We compare our proposed DA-LCE against a comprehensive suite of baselines to rigorously test its advantages. Our main competitors are the state-of-the-art difficulty-agnostic model, LC-PFN (Adriaensen et al. 2023), and the leading architecture-aware model, LC-GODE (Ding et al. 2025). To ensure a thorough comparison, we also include representative methods based on Neural ODEs (NODE, NSDE) and classic sequence models (LSTM).

Evaluation Metrics. To provide a multifaceted assessment, we evaluate all methods on three key aspects of performance. We measure point prediction accuracy using the Mean Absolute Percentage Error (MAPE), assess probabilistic forecast quality with the Negative Log-Likelihood (NLL), and quantify practical utility for early stopping via the Anytime Regret. This combination of metrics allows for a comprehensive evaluation of both predictive accuracy and real-world efficiency.

Main Result

We begin by presenting the overall performance comparison of DA-LCE against all baseline methods across our four diverse benchmarks. The results, summarized in Table 1, demonstrate a clear and consistent advantage for our difficulty-aware approach. DA-LCE achieves the best performance on nearly every benchmark across all metrics, particularly in the more challenging NLL, which measures probabilistic accuracy. This indicates that our model not only produces more accurate point estimates but also generates more reliable and well-calibrated predictive distributions. Notably, while the architecture-aware LC-GODE shows strong performance, it is consistently surpassed by our method, suggesting that understanding task difficulty provides a more powerful signal for extrapolation than architectural information alone.

While aggregate metrics demonstrate the overall superiority of DA-LCE, they do not fully reveal the source of its robustness. Our core hypothesis is that performance on LCE is strongly dependent on the intrinsic difficulty of the task. To investigate this, we stratify the test curves from all benchmarks into three categories—*Easy*, *Medium*, and *Hard*—based on a K-Means clustering of their full-trajectory dynamics vectors Φ_{full}. Figure 3 plots the MAPE of each model

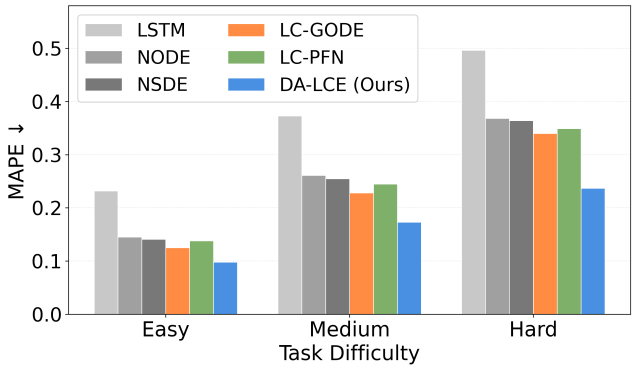

Figure 3: MAPE on tasks of varying difficulty. DA-LCE maintains low error, while baselines degrade on harder tasks.

across these difficulty strata.

The results provide a striking confirmation of our hypothesis. On *Easy* tasks, most models perform reasonably well, with our DA-LCE holding a modest edge. However, as task difficulty increases, the performance of all baseline models degrades significantly. The difficulty-agnostic LC-PFN and even the architecture-aware LC-GODE struggle to adapt, exhibiting a sharp rise in error on *Hard* tasks. In stark contrast, our DA-LCE maintains a much more stable and lower error rate across all strata. Its ability to explicitly condition on the difficulty proxy allows it to apply a more robust extrapolation strategy precisely when it is needed most. This result compellingly illustrates that difficulty awareness is not just beneficial, but essential for building truly robust LCE systems.

Ablation Studies

Having established the superior performance of our full DA-LCE framework, we now conduct a series of controlled ablation studies to rigorously dissect the sources of this improvement. Our goal is to quantify the individual contributions of our two primary innovations: the high-fidelity, diffusion-based prior and the explicit difficulty-aware conditioning. To this end, we compare our DA-LCE (Full Model) against three key variants: (1) the standard LC-PFN, which serves as a baseline with neither component; (2) a model using our diffusion prior but w/o Difficulty conditioning, to isolate the prior's effect; and (3) a model with difficulty conditioning but w/o the Diffusion Prior, to isolate the conditioning's effect. This setup allows us to precisely attribute performance gains to each of our core contributions.

Table 2 presents the results of this study, showing the average MAPE across all benchmarks for each model variant. The findings provide clear and compelling evidence for our design choices. Firstly, comparing the *Baseline* with *Ours (w/o Difficulty)* reveals a significant performance gain. This demonstrates that simply replacing the simple parametric prior with our high-fidelity, diffusion-generated corpus substantially improves the model's ability to capture real-world learning dynamics. The DDPM-based prior provides a much richer and more realistic foundation for learning extrapolation. Secondly, the comparison between the *Baseline* and *Ours (w/o Diffusion Prior)* also shows a notable reduction in error. This confirms that introducing the difficulty-aware conditioning mechanism, even with a simple prior, allows the model to make more adaptive and accurate predictions. The model effectively learns to tailor its extrapolation strategy based on the explicit difficulty signal.

Model Variant	**MAPE ↓**
Baseline (LC-PFN)	0.244
Ours (w/o Difficulty)	0.208
Ours (w/o Diffusion Prior)	0.193
DA-LCE (Full Model)	**0.186**

Table 2: Ablation study on DA-LCE components (average MAPE across all benchmarks).

Most importantly, our *DA-LCE (Full Model)* achieves the lowest error, significantly outperforming all other variants. This result highlights a crucial synergistic effect: the best performance is not achieved by either innovation in isolation, but by their combination. The high-fidelity prior provides the model with a rich understanding of *what* realistic curves look like, while the difficulty conditioning provides the crucial context of *which* specific type of curve dynamics to expect. This synergy validates the integrity of our proposed framework and confirms that both components are essential for achieving a new state-of-the-art in robust learning curve extrapolation.

Application

While metrics like MAPE and NLL are crucial for evaluating predictive accuracy, the ultimate goal of LCE is to accelerate AutoML processes. To demonstrate the practical utility of DA-LCE, we evaluate its effectiveness in a realistic model selection scenario using predictive early stopping. In this setup, we simulate the process of sequentially training a set of candidate models (e.g., different architectures or hyperparameter configurations) and use the LCE model to decide whether to terminate a run prematurely if it is predicted to be suboptimal.

We compare the efficiency of early stopping strategies guided by our DA-LCE, the best-performing baseline (LC-GODE), and the difficulty-agnostic LC-PFN. We also include a *No-Stop* baseline, which trains every configuration for its full duration and represents the standard, non-accelerated approach. The performance is measured by the *Anytime Regret*, which plots the performance gap to the true best model as a function of the total computational budget consumed (measured in cumulative epochs).

Figure 4 illustrates the results, averaged across all tasks in our benchmarks. The *No-Stop* baseline shows a linear decrease in regret, as expected. Both LC-PFN and LC-GODE offer significant speedups over this baseline, successfully identifying and pruning unpromising runs. However, our DA-LCE-guided strategy demonstrates markedly superior efficiency. Its curve drops more steeply and converges to the optimal performance level using a fraction of the computa-

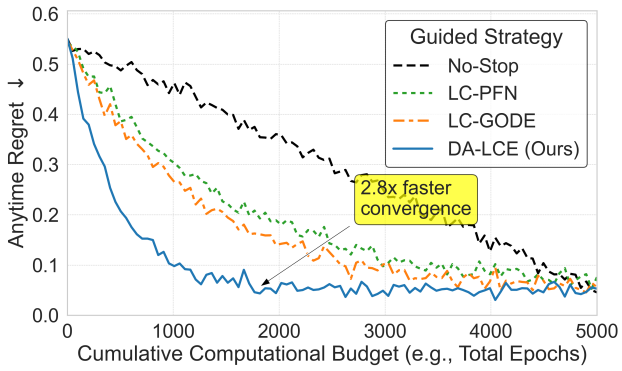

Figure 4: Anytime regret in model selection. DA-LCE finds the best model with significantly less computation.

tional budget required by the others.

Quantitatively, to reach the same performance level as the *No-Stop* baseline, DA-LCE requires approximately 2.8x less computational resources. This substantial speedup is a direct consequence of its enhanced predictive robustness; by making more reliable forecasts, especially for difficult or non-standard tasks that often confound other methods, DA-LCE can make more confident and correct termination decisions. This result confirms that the superior accuracy of our difficulty-aware framework translates directly into tangible and significant efficiency gains, making it a more reliable and powerful tool for building intelligent and cost-effective AutoML systems.

Qualitative Analysis and Insights

To provide deeper insight into the inner workings of our framework, we conduct a qualitative analysis of its key components. Our goal is to visually verify that (a) our conditional diffusion model can generate realistic and dynamically-controlled learning curves, and (b) our DA-LCE model effectively utilizes the difficulty-aware conditioning signal to adapt its predictions.

Validating the Conditional Generative Prior. First, we examine the capabilities of our DDPM-based generator. We sample two distinct target dynamics vectors, Φ_{target}, from our fitted GMM: one representing an *Easy* task (high progress, low non-linearity, low volatility) and another representing a *Hard* task (low progress, high non-linearity, high volatility). We then use the trained diffusion model to generate several learning curve samples conditioned on each of these vectors. As shown in Figures 5(a) and (b), the results are compelling. The curves generated for the *Easy* condition exhibit smooth, rapid convergence, closely mirroring idealized learning trajectories. In contrast, the curves conditioned on the *Hard* vector display the intended characteristics: they learn much slower, exhibit significant plateaus and fluctuations, and yet remain realistic. This visually confirms that our generative prior is not only capable of producing high-fidelity curves but is also highly controllable, providing a valid and powerful foundation for training our extrapolation model.

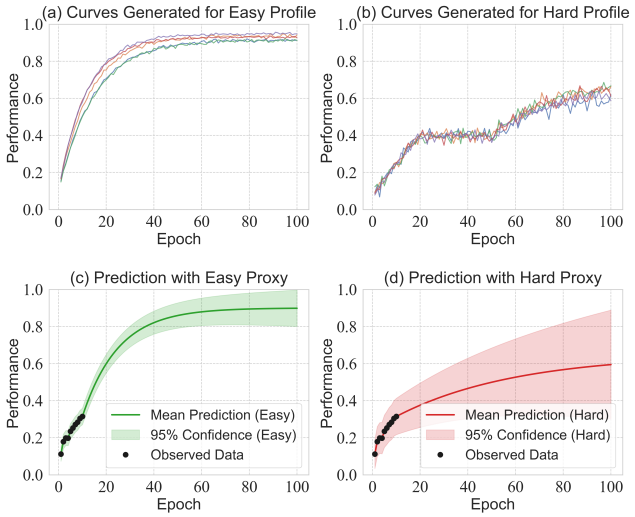

Figure 5: Qualitative analysis of the DA-LCE framework.

Visualizing Difficulty-Aware Adaptation. Next, we investigate how the DA-LCE model leverages the difficulty proxy during inference. We take a single, fixed partial learning curve from a real-world task (the first 10 epochs). We then feed this *same* partial curve into our DA-LCE model twice, but each time with a different, manually-specified difficulty proxy Φ_{early}. The first prediction is conditioned on an *Easy* proxy, while the second is conditioned on a *Hard* proxy.

The resulting extrapolations, shown in Figures 5(c) and (d), starkly illustrate the adaptive nature of our model. When conditioned on the *Easy* proxy, the model predicts a confident, optimistic trajectory with rapid convergence to a high performance level. However, when the *exact same* historical data is presented with the *Hard* proxy, the model's prediction changes dramatically. It becomes more conservative, forecasting slower progress, a lower final performance, and a wider uncertainty band, reflecting the increased difficulty. This experiment provides direct visual evidence that our model has learned to interpret the difficulty signal and uses it to intelligently modulate its predictions, moving beyond simple pattern matching to a more nuanced, context-aware form of reasoning.

Conclusion

In this work, we introduced DA-LCE, a novel framework that addresses a critical oversight in Learning Curve Extrapolation: the neglect of task difficulty. By proposing a self-contained difficulty proxy and leveraging a conditional diffusion model to create a high-fidelity prior, our CD-PFN model achieves state-of-the-art performance, significantly outperforming both difficulty-agnostic and architecture-aware baselines. Our findings establish task difficulty as a powerful and complementary source of contextual information, proving its impact is as significant as model architecture for building more robust and efficient AutoML systems.

References

Adriaensen, S.; Rakotoarison, H.; Müller, S.; and Hutter, F. 2023. Efficient bayesian learning curve extrapolation using prior-data fitted networks. *Advances in Neural Information Processing Systems*, 36: 19858–19886.

Baker, B.; Gupta, O.; Raskar, R.; and Naik, N. 2017. Accelerating Neural Architecture Search using Performance Prediction. In *International Conference on Learning Representations Workshop*.

Bansal, A.; Stoll, D.; Janowski, M.; Zela, A.; and Hutter, F. 2022. Jahs-bench-201: A foundation for research on joint architecture and hyperparameter search. *Advances in Neural Information Processing Systems*, 35: 38788–38802.

Bergstra, J.; and Bengio, Y. 2012. Random search for hyper-parameter optimization. *The journal of machine learning research*, 13(1): 281–305.

Ding, Y.; Huang, Z.; Shou, X.; Guo, Y.; Sun, Y.; and Gao, J. 2025. Architecture-aware learning curve extrapolation via graph ordinary differential equation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 16289–16297.

Domhan, T.; Springenberg, J. T.; Hutter, F.; et al. 2015. Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves. In *IJCAI*, volume 15, 3460–8.

Dong, X.; and Yang, Y. 2020. NAS-Bench-201: Extending the Scope of Reproducible Neural Architecture Search. In *International Conference on Learning Representations*.

Eggensperger, K.; Müller, P.; Mallik, N.; Feurer, M.; Sass, R.; Klein, A.; Janzing, D.; and Hutter, F. 2021. HPOBench: A Large-Scale Reproducible Benchmark for Black-Box HPO Based on OpenML. In *NeurIPS Datasets and Benchmarks Track*.

Finn, C.; Abbeel, P.; and Levine, S. 2017. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In *International Conference on Machine Learning*, 1126–1135.

Frey, L. J.; and Fisher, D. H. 1999. Modeling decision tree performance with the power law. In *Seventh international workshop on artificial intelligence and statistics*. PMLR.

Graves, A. 2012. Long short-term memory. *Supervised sequence labelling with recurrent neural networks*, 37–45.

He, X.; Zhao, K.; and Chu, X. 2021. AutoML: A survey of the state-of-the-art. *Knowledge-based systems*, 212: 106622.

Ho, J.; Jain, A.; and Abbeel, P. 2020. Denoising Diffusion Probabilistic Models. *Advances in Neural Information Processing Systems*, 33: 6840–6851.

Hutter, F.; Kotthoff, L.; and Vanschoren, J. 2019. Automated Machine Learning: Methods, Systems, Challenges. *Springer Nature*.

Klein, A.; Falkner, S.; Springenberg, J. T.; and Hutter, F. 2017. Learning curve prediction with Bayesian neural networks. In *International conference on learning representations*.

Metz, L.; Maheswaranathan, N.; Sun, R.; Freeman, C. D.; Poole, B.; and Sohl-Dickstein, J. 2020. Using a thousand optimization tasks to learn hyperparameter search strategies. *arXiv preprint arXiv:2002.11887*.

Mohr, F.; and van Rijn, J. N. 2024. Learning curves for decision making in supervised machine learning: a survey. *Machine Learning*, 113(11): 8371–8425.

Nakkiran, P.; Kaplun, G.; Bansal, Y.; Yang, T.; Barak, B.; and Sutskever, I. 2021. Deep double descent: Where bigger models and more data hurt. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(12): 124003.

Ning, X.; Zheng, Y.; Zhao, T.; Wang, Y.; and Yang, H. 2020. A generic graph-based neural architecture encoding scheme for predictor-based nas. In *European Conference on Computer Vision*, 189–204. Springer.

Peng, D.; and Pan, S. J. 2023. Clustered task-aware meta-learning by learning from learning paths. *IEEE transactions on pattern analysis and machine intelligence*, 45(8): 9426–9438.

Perez, E.; Strub, F.; De Vries, H.; Dumoulin, V.; and Courville, A. 2018. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Ren, P.; Xiao, Y.; Chang, X.; Huang, P.-Y.; Li, Z.; Chen, X.; and Wang, X. 2021. A comprehensive survey of neural architecture search: Challenges and solutions. *ACM Computing Surveys (CSUR)*, 54(4): 1–34.

Ronneberger, O.; Fischer, P.; and Brox, T. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 234–241.

Siems, J.; Zimmer, L.; Zela, A.; Lukasik, J.; Keuper, M.; and Hutter, F. 2020. NAS-Bench-301 and the Case for Surrogate Benchmarks for Neural Architecture Search. *arXiv preprint arXiv:2008.09777*.

Swersky, K.; Snoek, J.; and Adams, R. P. 2014. Freeze-Thaw Bayesian Optimization. *arXiv preprint arXiv:1406.3896*.

Vanschoren, J.; Van Rijn, J. N.; Bischl, B.; and Torgo, L. 2014. OpenML: Networked Science in Machine Learning. *ACM SIGKDD Explorations Newsletter*, 15(2): 49–60.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is All You Need. *Advances in Neural Information Processing Systems*, 30.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks. *International Conference on Learning Representations*.

Wang, Z.; Dahl, G. E.; Swersky, K.; Lee, C.; Nado, Z.; Gilmer, J.; Snoek, J.; and Ghahramani, Z. 2024. Pre-trained Gaussian Processes for Bayesian Optimization. *Journal of Machine Learning Research*, 25(212): 1–83.

White, C.; Safari, M.; Sukthanker, R.; Ru, B.; Elsken, T.; Zela, A.; Dey, D.; and Hutter, F. 2023. Neural architecture search: Insights from 1000 papers. *arXiv preprint arXiv:2301.08727*.

Zimmer, L.; Lindauer, M.; and Hutter, F. 2021. Auto-pytorch: Multi-fidelity metalearning for efficient and robust

autodl. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(9): 3079–3090.

Zoph, B.; and Le, Q. V. 2017. Neural Architecture Search with Reinforcement Learning. *International Conference on Learning Representations*.