Extrapolating Learning Curves: When Do Neural Networks Outperform Parametric Models?

Adelina Andreea Cazacu[1]

Supervisors: Tom Viering[1]**, Cheng Yan**[1]**, Sayak Mukherjee**[1]

[1]**EEMCS, Delft University of Technology, The Netherlands**

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfillment of the Requirements
For the Bachelor of Computer Science and Engineering
June 22, 2025

Name of the student: Adelina Andreea Cazacu
Final project course: CSE3000 Research Project
Thesis committee: Tom Viering, Cheng Yan, Sayak Mukherjee, Matthijs Spaan

Abstract

Learning curve extrapolation helps practitioners predict model performance at larger data scales, enabling better planning for data collection and computational resource allocation. This paper investigates when neural networks outperform parametric models for this task. We conduct a comprehensive comparison of LC-PFNs (Learning Curve Prior-Fitted Networks) and three established parametric models (POW4, MMF4, WBL4) using LCDB v1.1, a large-scale dataset with learning curves generated across 265 classification tasks and 24 learners. Surprisingly, we find that parametric models — especially POW4 and MMF4 — consistently outperform LC-PFN across all generalization scenarios and most cutoff regions. However, LC-PFN demonstrates competitive performance when extrapolating from early-stage data, ranking second-best at 10%, 30%, and 50% cutoffs. This suggests LC-PFNs can be valuable when only a small fraction of the learning curve is available. LC-PFN is particularly challenged by smooth and flat curves, but shows slightly improved performance on irregular patterns such as peaking and dipping curves, though it remains outperformed by all parametric models. These trends highlight a misalignment between LC-PFN's training distribution and the real-world diversity of learning curves. Our findings emphasize the strength of parametric models under realistic conditions and suggest avenues for improving LC-PFNs through architectural flexibility and curve length variability during training.

1 Introduction

Machine learning (ML) practitioners in both academic and industry mediums often face a critical question when designing an ML application: "How much data is enough?". Learning curves, which plot model performance as a function of training set size, are a fundamental tool for answering this question by revealing how performance scales with available data. However, these curves can exhibit unexpected behaviors - often referred to as ill-behaved curves, characterized by non-monotonic patterns, plateaus, or sudden performance drops [10, 6]. Thus, accurately modeling a relationship between training set size and model performance can empower scientists and engineers to make informed decisions on whether the desired accuracy targets are achievable within practical constraints of budget, time, and environmental impact, in terms of data collection and computational requirements. This efficiency gain is particularly important in an era where environmental concerns about ML's footprint are growing [5].

Unlike the more widely studied epoch-based learning curves that track performance over training iterations, sample-size learning curves enable practitioners to devise data collection strategies before substantial investments are made, even for models like K-Nearest-Neighbors that don't involve iterative training [10, 7].

Recent advancements in this area have explored both parametric and neural network approaches to learning curve extrapolation. Parametric models, such as MMF4 and WBL4, have demonstrated strong performance across a wide range of learning scenarios [8], while neural network approaches such as Learning Curve Prior-Fitted Networks (LC-PFNs) have shown promise in epoch-based learning curve extrapolation [1]. Viering et al. [11] observed that these approaches have different strengths and weaknesses, particularly when generalizing to unseen datasets and learners, yet the specific conditions under which neural networks outperform parametric models remain poorly understood. Moreover, the aforementioned comparative study has been conducted on LCDB 1.0 [8], while the more challenging and realistic LCDB 1.1 — which reveals significantly higher rates of ill-behaved learning curves [12] — remains underexplored in comparative analyses. This research aims to address this gap by investigating the main research question:

> When do neural networks outperform parametric models in learning curve extrapolation?

To clarify the direction of the research, this paper will aim to answer the following three sub-questions:

RQ1: How do parametric models and neural networks compare in learning curve extrapolation when tested on known datasets and known learners (KDKL), unseen datasets (UD), unseen learners (UL), and simultaneously unseen datasets and learners (UDUL)?

RQ2: How does the amount of the observed learning curve (i.e., the region before the cutoff) affect the relative extrapolation performance of parametric models and neural networks?

RQ3: How does the shape of the learning curve influence the relative performance of parametric models and neural networks in learning curve extrapolation?

These questions collectively address the core factors that determine method selection in practice. The first question evaluates how both approaches handle different types of generalization challenges, which is crucial since real-world applications typically involve extrapolating to new domains or algorithms not seen during training. The second question aims to explore how the amount of observed data before extrapolation affects relative performance, offering a solid intuition about the minimum observations required for reliable predictions with each approach. The third and final question investigates whether certain learning curve characteristics inherently favor one approach over another, as different curve shapes may align better with the inductive biases of parametric versus neural methods.

The remainder of this paper is organized as follows: Section 2 reviews the relevant literature concerning the Learning Curve Database, parametric models, and neural network approaches to learning curve extrapolation. Section 3 describes the experimental methodology, including model implementations, dataset splits, and evaluation protocols used to compare LC-PFNs against parametric models. Section 4 outlines the experimental setup for both studies, detailing the evaluation of models across generalization scenarios and cutoff

percentages (Experiment 1) and the analysis of performance across different learning curve shapes (Experiment 2). Section 5 presents the results from both experiments, revealing the comparative performance of neural networks versus parametric models under varying conditions. Section 6 discusses the implications of our findings, potential explanations for observed performance differences, and limitations of current approaches. Section 7 concludes with a summary of key contributions and outlines promising directions for future research. Finally, Section 8 addresses responsible research practices including reproducibility measures and environmental impact considerations.

2 Related Work

This section reviews the relevant literature and establishes the theoretical foundation for our comparative study. We examine existing work on the Learning Curve Database, parametric modeling approaches for learning curve extrapolation, neural network methods including LC-PFNs, and precursory comparative studies in existing literature.

2.1 The Learning Curve Database

To investigate the conditions under which neural networks outperform parametric models in learning curve extrapolation, this study adopts a comprehensive experimental methodology using the Learning Curve Database (LCDB) version 1.1 [12]. This dataset provides a large-scale collection of learning curves spanning 265 OpenML classification tasks and 24 different machine learning algorithms, offering diverse scenarios for robust evaluation.

The LCDB 1.1 dataset represents a significant advancement over its predecessor, LCDB 1.0 [8], particularly in its recognition and systematic documentation of ill-behaved learning curves. While previous work often assumed learning curves to be well-behaved (monotonic and convex), LCDB 1.1 reveals that approximately 14% of learning curves exhibit significant ill-behavior—nearly twice the rate previously estimated [12]. This finding is crucial for our comparative analysis, as it provides a more realistic and challenging benchmark that better reflects the complexities encountered in practical machine learning scenarios.

The LCDB 1.1 dataset contains learning curves across 137 **anchor** points, corresponding to training set sizes ranging from 16 to 1,097,152 samples following the formula $\lceil 16 \times 2^{i/8} \rceil$ for $i \in \{0, 1, ..., 136\}$. Each learning curve represents the error rate performance trajectory of a specific algorithm on a particular dataset as a function of training set size (x-axis), where performance (y-axis) is measured on the validation set. The inclusion of ill-behaved curves, which may exhibit non-monotonic behavior, peaks, or other irregular patterns, presents new, unique challenges for both parametric and neural network approaches.

2.2 Parametric Models for Learning Curve Extrapolation

To extrapolate learning curves from partial observations, we employ established parametric models that reflect common inductive biases in machine learning performance trends. Based on empirical studies showing their superior performance [8, 2, 3], we implement three nonlinear functional forms: MMF4, WBL4, and POW4. Each of these models captures distinct asymptotic behaviors found in typical learning curves, such as early rapid gains followed by saturation or smooth power-law decay.

Given a learning curve $y(n)$, where n denotes the anchor corresponding to the training set size used, the extrapolation procedure begins by selecting a cutoff point that defines the amount of observed data. This cutoff can be chosen randomly, explicitly specified by index, or determined as a percentage of the full curve (e.g., 70%). To ensure sufficient information for reliable fitting, a minimum of ten points is always required before model estimation is attempted.

The MMF4 model is defined as:

$$f(n) = \frac{ab + cn^d}{b + n^d}$$

This function is known for its sigmoidal shape, which captures fast initial improvement that asymptotically plateaus—mirroring the typical saturation behavior of ML models.

The WBL4 model, based on the Weibull family of functions, is expressed as:

$$f(n) = c - b \cdot \exp(-a \cdot n^d)$$

This model is motivated by its exponential convergence toward an asymptote, often suitable for curves that show diminishing returns quickly after a brief rise.

Lastly, the POW4 model reflects power-law decay and is defined as:

$$f(n) = a - b \cdot (d + n)^{-c}$$

This power-law formulation captures learning curves that exhibit diminishing returns following an inverse power relationship with training set size. The model's flexibility lies in its ability to represent both steep initial improvements (when c is large) and more gradual asymptotic approaches to peak performance (when c is small).

2.3 LC-PFN for Learning Curve Extrapolation

The LC-PFN approach introduced by Adriaensen et al. [1] employs Prior-Data Fitted Networks (PFNs)—Transformer-based neural networks pre-trained on synthetic data generated from a parametric prior over learning curves. Unlike parametric methods that provide point estimates, LC-PFN outputs discretized probability distributions over 1,000 bins, enabling full uncertainty quantification. The method demonstrated remarkable computational efficiency with inference times under 0.1 seconds compared to over 100 seconds for MCMC methods—a speedup exceeding 10,000×.

Evaluation on 20,000 real learning curves revealed that LC-PFN's superiority is conditional rather than universal. Adriaensen et al. [1] shows that while consistently outperforming MCMC on PD1, NAS-Bench-201, and Taskset benchmarks, performance on LCBench was only marginal. Critically, the method failed on NAS-Bench-201 tasks containing inflection points not represented in the training prior,

demonstrating that neural approaches are heavily dependent on how well their training distribution captures real learning curve characteristics. This prior dependence contrasts with parametric methods that can adapt their functional forms during fitting.

The LC-PFN study's findings directly support the motivation for systematic comparative analysis by showing that neither approach universally dominates. The method's practical advantages emerged most clearly in dynamic decision-making contexts, achieving 2-6× speedups in early stopping applications across 45 of 53 datasets. However, the substantial upfront investment required (9 hours pre-training on the Nvidia Tesla P100 on 10M synthetic curves) versus parametric methods' zero pre-training cost highlights different computational trade-offs that may favor different approaches depending on the application context and available learning curve diversity.

2.4 Prior Comparative Studies

The most directly relevant prior work is the study by Viering et al.[11], which conducted a comparative analysis of different LC-PFN approaches using LCDB 1.0. Their research developed data-driven priors for LC-PFNs, comparing parametric prior-based LC-PFNs (using MMF4 and WBL4 curve generators) against a "Real Data LC-PFN" trained directly on learning curve data with augmentation. The Real Data LC-PFN approach outperformed the original epoch-trained LC-PFN in 78-80% of cases when extrapolating sample-size learning curves.

However, several important research gaps remain that our study aims to address:

Database limitations: Viering et al.'s analysis was conducted on LCDB 1.0, which underestimated the prevalence of ill-behaved learning curves. By utilizing LCDB 1.1, the current study aims to provide a more realistic evaluation environment that better reflects the challenges encountered in practice, where approximately 14% of learning curves exhibit significant non-monotonic behavior or other irregularities.

Incomplete generalization analysis: The prior study focused primarily on three generalization scenarios: UD, UL, and UDUL. Notably absent was analysis of the Known Learner, KDKL scenario, which represents an important baseline for understanding method performance under optimal conditions where both the algorithm and dataset characteristics have been previously observed.

Limited parametric model coverage: While Viering et al. evaluated MMF4 and WBL4 models within their LC-PFN framework, they did not include POW4, despite evidence suggesting that MMF4, WBL4, and POW4 collectively represent the most effective parametric approaches for learning curve modeling [8]. Additionally, their study did not directly compare the LC-PFN against traditional parametric fitting methods.

Narrow evaluation scope: Beyond generalization scenarios, this paper examines how different learning curve shapes and various extrapolation cutoff percentages influence comparative performance between parametric and neural approaches, providing a more comprehensive understanding of the conditions that favor each method.

3 Methodology

This section details the modeling and evaluation procedures used for learning curve extrapolation. We first describe the implementation of a neural network-based probabilistic forecasting model (LC-PFN), including its training objectives, data augmentation strategy, and hyperparameter configuration. We then outline the setup for fitting classical parametric baselines, emphasizing consistent initialization and bounded optimization to ensure comparability. Finally, we define the evaluation protocol, specifying the extrapolation scenarios and metrics used to assess performance across varying levels of observed curve data.

3.1 Model Implementation

Neural Network Approach - LC-PFN

During training, the model was optimized using the bar distribution loss, which enables the network to output full predictive distributions over learning curve values rather than point estimates. For evaluation purposes, the *median* of the predicted distribution is reported as the final extrapolated curve.

To increase generalization and robustness, a data augmentation strategy [9] was adopted: each input learning curve was synthetically varied through randomized scaling between anchor points. Specifically, pairs of anchor indices were selected uniformly at random, and the curve was stretched or compressed accordingly to simulate plausible alternative growth patterns. This approach, also adopted by Viering et al. [11], preserves the essential shape characteristics of the original curve while exposing the model to a wider variety of trajectories, thus aiming to promote better extrapolation across unseen datasets and learners.

The LC-PFN model was trained using the following hyperparameter configuration: a sequence length of 80 anchor points (`SEQ_LEN = 80`), embedding size of 128 (`EMSIZE = 128`), and a 3-layer architecture (`NLAYERS = 3`). The probabilistic output distribution was discretized into 1000 bins (`NUM_BORDERS = 1000`) to support fine-grained uncertainty modeling. Training was performed for 300 epochs (`EPOCH = 300`) using a batch size of 50 (`BATCH_SIZE = 50`) and a learning rate of 1×10^{-4} (`LR = 0.0001`). These settings were selected based on the recommendations in [1] and validated through empirical testing on held-out validation curves.

Parametric Models

The model is then fitted to the observed portion of the curve via nonlinear least squares. To prevent data leakage, the initial parameter values and bounds are not adapted based on the curve being fitted. Instead, they are fixed a priori, informed by empirical stability.

For MMF4, the initial guess $[a, b, c, d] = [0.9, 1000.0, 0.1, 1.0]$ encodes a strong asymptotic behavior with gradual curvature, while the bounds $[0.01, 1e\text{-}6, 0.0, 0.01], [1.0, \infty, 1.0, 10.0]$ are selected to reflect plausible limits for accuracy and to prevent numerical instability during optimization.

In the case of WBL4, the initial parameters are set to $[a, b, c, d] = [0.001, 0.8, 0.9, 1.0]$, reflecting slow exponential decay toward high performance. The fitting bounds ensure smooth convergence and prevent divergence, especially in the presence of steep curves or small-scale datasets.

POW4 is initialised with $[a, b, c, d] = [0.9, 0.8, 1.0, 100.0]$, with bounds $[0.01, 0.01, 0.001, 1.0]$, $[1.0, 2.0, 5.0, 10000.0]$ to ensure the model remains flexible yet stable during fitting.

Fitting is performed using `scipy.optimize.curve_fit`, prioritizing bounded optimization. In cases where this fails (e.g., due to local minima or ill-conditioning), an unbounded optimization is attempted as a fallback. If both strategies fail, the model reverts to using the initial guess to ensure the procedure remains complete for all curves.

To ensure that the predicted performance remains valid, extrapolated values are clipped to the $[0, 1]$ interval, which reflects the normalized accuracy range of the underlying tasks.

3.2 Dataset splits

For **Experiment 1**, to ensure rigorous evaluation of generalization capabilities, a systematic data partitioning strategy was implemented. Datasets were randomly split into training (80%) and testing (20%) subsets, while the 24 machine learning algorithms were similarly partitioned into training (80%) and testing (20%) subsets. This dual partitioning enables the construction of four distinct evaluation scenarios that address different aspects of generalization:

- **Known Datasets, Known Learners (KDKL)**: Both datasets and learners present during training. Thus, this set contains curves already seen during training.
- **Unseen Datasets (UD)**: New datasets with algorithms seen during training. This set also encompasses never-before-seen curves, but curves that originate from the application of ML algorithms associated with the curves seen in training.
- **Unseen Learners (UL)**: New learners with datasets seen during training. Similarly, curves originate from datasets associated with curves from the training set.
- **Unseen Datasets, Unseen Learners (UDUL)**: Both datasets and algorithms absent from training.

To analyze the impact of curve shape on model performance in **Experiment 2**, learning curves were categorized based on their characteristic behaviors identified in prior literature. The 23 learning algorithms in LCDB 1.1 (N.B. learner with index 23 - DummyClassifier, has been excluded from this analysis) were grouped into four distinct categories based on their typical learning curve patterns, according to the empirical study by Yan et al.[12], reflected in Figure 1:

- Flat learners (indices: 1, 2, 3, 14): Algorithms that typically produce learning curves with minimal performance improvement as training set size increases, often exhibiting plateau behavior early in the learning process.
- Monotonic convex learners (indices: 0, 4, 5, 20, 21, 22, 6, 7, 8, 10, 18): Algorithms that demonstrate the classic learning curve behavior with consistent performance improvements that gradually level off, following smooth convex patterns toward asymptotic performance.

		Flat	Mono & Conv	Peaking	Dipping
0	*SVM_Linear*	3.4	91.7	1.51	2.64
1	*SVM_Poly*	17.36	78.49	0.38	1.89
2	*SVM_RBF*	18.49	73.21	0	0.38
3	*SVM_Sigmoid*	19.25	32.45	23.4	42.26
4	*Decision Tree*	4.53	94.34	0.38	0.75
5	*ExtraTree*	3.77	94.72	0	0.38
6	*LogisticRegression*	6.79	88.68	1.13	1.51
7	*PassiveAggressive*	5.66	85.66	1.89	3.4
8	*Perceptron*	3.02	93.96	0.75	1.51
9	*RidgeClassifier*	7.17	76.23	10.94	4.91
10	*SGDClassifier*	2.26	95.09	0.38	2.26
11	*MLP*	4.91	67.55	9.81	3.77
12	*LDA*	3.77	58.87	24.53	6.42
13	*QDA*	3.77	51.32	19.62	26.79
14	*BernoulliNB*	26.42	60.38	4.91	5.66
15	*MultinomialNB*	9.06	54.72	3.02	4.53
16	*ComplementNB*	8.3	54.34	4.15	6.79
17	*GaussianNB*	4.53	71.32	12.83	24.53
18	*KNN*	10.94	86.79	0.75	2.26
19	*NearestCentroid*	10.94	81.13	4.15	11.32
20	*ens.ExtraTrees*	9.06	87.92	0.75	1.89
21	*ens.RandomForest*	9.06	88.3	0.38	1.13
22	*ens.GradientBoosting*	3.4	95.09	0	0.75

Figure 1: Shape statistics for each learner in LCDB 1.1. The green borders represent the groups selected for each column (e.g. the green bordered values on column 'Flat' indicate the learners whose learning curves best exhibit this shape characteristic).

- Peaking learners (indices: 3, 9, 11, 12, 13, 17): Algorithms whose learning curves exhibit maximum performance at intermediate training set sizes, followed by performance degradation.
- Dipping learners (indices: 3, 19, 13, 12, 17): Algorithms that show temporary performance drops during the learning process before potentially recovering, creating non-monotonic patterns that violate common learning curve assumptions.

A similar 80-20 train-test split is applied for each shape category set.

3.3 Evaluation Protocol

Performance Metrics

Model performance was evaluated using four complementary metrics, each addressing specific challenges inherent in learning curve extrapolation:

- **Symmetric Mean Absolute Percentage Error (SMAPE)**:

$$\text{SMAPE} = \frac{100}{n} \sum_{i=1}^{n} \frac{|y_i - \hat{y}_i|}{|y_i| + |\hat{y}_i|}$$

SMAPE was chosen as the primary metric because learning curve extrapolation involves comparing prediction quality across algorithm-dataset combinations with vastly different performance scales and saturation levels. SMAPE's scale-independence enables fair comparison of extrapolation accuracy regardless of whether the

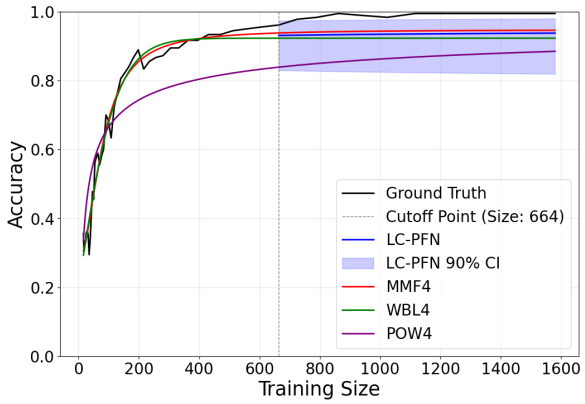

Figure 2: Example of learning curve extrapolation. The vertical dotted line represents the cutoff point. Data points on the left-hand side the cutoff are observed by the model, and then used to extrapolate the data points on the right-hand side.

curve operates in low-performance or high-performance regimes, focusing on relative prediction errors rather than absolute deviations. Kreinovich et al. [4] moves SMAPE from being an empirically successful but theoretically unmotivated metric to having a solid mathematical foundation based on approximation theory.

- **Mean Squared Error (MSE)** and **Mean Absolute Error (MAE)** were included as widely used regression metrics to facilitate comparison with other studies, with MSE providing sensitivity to outliers and MAE offering robust, easily interpretable average prediction errors.

Extrapolation Procedure

For each learning curve evaluation, a cutoff point was established to simulate real-world scenarios where only partial learning curves are available. The curve was divided into observed (training) and unobserved (testing) portions, with models fitted on the observed portion and evaluated on their ability to extrapolate to the unobserved region. The cutoff point is determined by finding the anchor size closest to the target percentage of the range between minimum and maximum training sizes in each curve.

Figure 2 shows a visual example where extrapolation is performed by all four models.

4 Experimental Setup

This section describes the two complementary experiments designed to evaluate model performance across different generalization scenarios, extrapolation cutoffs, and learning curve shape characteristics.

4.1 Experiment 1: Model Comparison Across Domain Scenarios and Cutoffs

This experiment aims to simultaneously address the first two research questions presented in Section 1. It investigates how parametric models and neural networks compare across diverse generalization scenarios (**RQ1**), and further examines how their relative extrapolation performance varies as the proportion of observed data is increased (**RQ2**).

The first step involves training the LC-PFN on a comprehensive dataset of 1.5 million learning curves generated through curve augmentation. The specific training configuration and dataset details for the LC-PFN are provided in Section 3.

Performance comparisons across the four generalization scenarios (**KDKL**, **UD**, **UL**, and **UDUL**) and five cutoff percentages (**10%**, **30%**, **50%**, **70%**, and **90%**) were conducted using 1000 randomly sampled learning curves for each performance metric per scenario, to ensure computational feasibility while maintaining statistical power.

Model performance was assessed using the three aforementioned complementary metrics: **SMAPE**, **MAE**, and **MSE**. These metrics provide different perspectives on prediction accuracy, with SMAPE offering scale-invariant comparison, MAE providing intuitive absolute error interpretation, and MSE emphasizing larger prediction errors.

The primary analytical approach centers on ranking comparisons to provide a comprehensive view of relative model performance across different experimental conditions. This ranking-based evaluation enables direct comparison of models across the diverse scenarios and cutoff percentages, through identifying consistent performance patterns and trade-offs. Statistical significance testing was conducted to quantify the reliability of observed performance differences between models. Detailed performance distributions are captured through boxplot visualizations, which are provided in the Appendix (Section A) to illustrate the full range of performance variability and outliers across the sampled learning curves.

4.2 Experiment 2: Model Comparison Across Learning Curve Shapes and Cutoffs

This experiment directly addresses the third research question by investigating how the shape of learning curves influences the relative performance of parametric models and neural networks in learning curve extrapolation (**RQ3**). The analysis aims to identify specific learning curve morphologies where LC-PFN demonstrates superior performance over the three parametric models (Power Law, Exponential, and Logarithmic), and conversely, where parametric approaches maintain their advantage.

The experimental design leverages a similarly trained LC-PFN model as the one in Experiment 1. Learning curves were systematically categorized into distinct shape classes based characteristics observed in prior studies [12] (see Figure 1).

Performance comparisons were conducted across five cutoff percentages (10%, 30%, 50%, 70%, and 90%). While cutoffs are not central to answering **RQ3**, sampling equally across the same fixed cutoffs as Experiment 1 reduces variability in results, as cutoff percentage can significantly impact model performance as demonstrated previously. For each shape category and cutoff combination, 200 randomly sampled learning curves were evaluated.

Model performance was assessed using the same three complementary metrics: SMAPE, MAE, and MSE. The analytical framework centers on shape-specific ranking analysis to identify performance patterns unique to different learning curve morphologies. Detailed performance distributions are

captured through boxplot visualizations provided in the Appendix (Section B).

5 Results

This section presents the findings from two experiments examining model performance across the experimental conditions discussed above.

5.1 Results of Experiment 1: Model Comparison Across Domain Scenarios and Cutoffs

The comprehensive evaluation of the four models (LC-PFN, MMF4, WBL4, and POW4) across different generalization scenarios and cutoff percentages reveals several key findings regarding their relative performance and extrapolation capabilities.

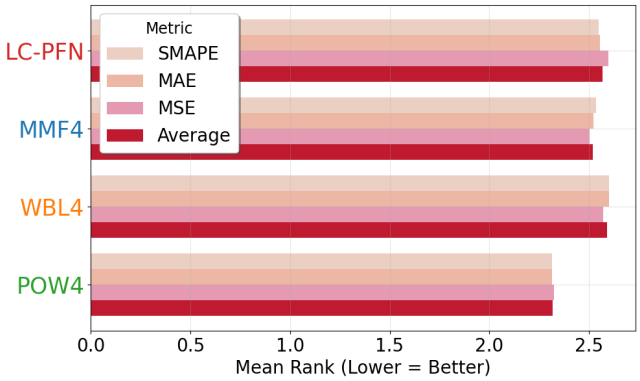

Figure 3: Mean model rankings across three evaluation metrics (SMAPE, MAE, MSE). Lower is better. All metrics yield consistent rankings, with POW4 performing best overall, followed by MMF4, LC-PFN, and WBL4.

Figure 3 shows that all three evaluation metrics produced comparable rankings across the four models, strengthening the reliability of the chosen metrics in the context of this experimental setup. The average ranking across all metrics indicates that POW4 achieved the best overall performance, followed by MMF4, LC-PFN, and WBL4, respectively. While POW4 demonstrated superior performance, the margins between models were relatively modest.

Performance Across Transfer Scenarios

Analysis of model performance across transfer scenarios reveals distinct patterns for each model (see Figure 4). POW4 consistently demonstrated the strongest performance across all four transfer scenarios (KDKL, UD, UDUL, and UL). Contrary to expectations, LC-PFN ranked worst in both the KDKL and UD scenarios, where it theoretically should have performed better, given its exposure to similar curves and learners during training. However, LC-PFN showed improved performance in the UL and UDUL scenarios, exceeding both MMF4 and WBL4, suggesting reasonable generalization capability to unseen curve data, though still trailing behind POW4. The three parametric models (MMF4, POW4, WBL4) exhibited some variability across transfer scenarios, though statistical significance testing (Figure 6) confirms these differences are not statistically meaningful.

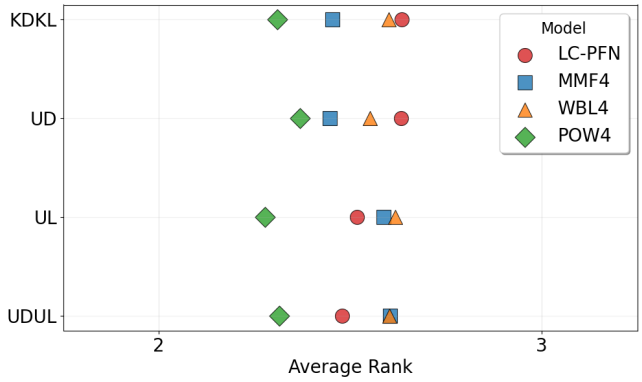

Figure 4: Average model rankings across the four transfer scenarios. Lower is better. POW4 consistently outperforms other models across all scenarios, while LC-PFN shows unexpected weakness in familiar scenarios (KDKL, UD) but slighly stronger generalization to completely unseen data (UL, UDUL).

Performance Across Cutoff Percentages

The analysis of performance across different cutoff percentages (Figure 5) reveals that POW4 maintained the best performance across all cutoff points except at 90%, where MMF4 slightly exceeded it. POW4 demonstrated remarkable stability across all cutoff percentages, while both MMF4 and WBL4 showed improved performance as more of the learning curve became available for fitting. Most notably, LC-PFN exhibited a distinctive pattern: it ranked second-best at the 10%, 30%, and 50% cutoffs but showed degraded performance as the cutoff percentage increased. This finding suggests that LC-PFN demonstrates particular strength in extrapolating from limited early-stage learning curve data.

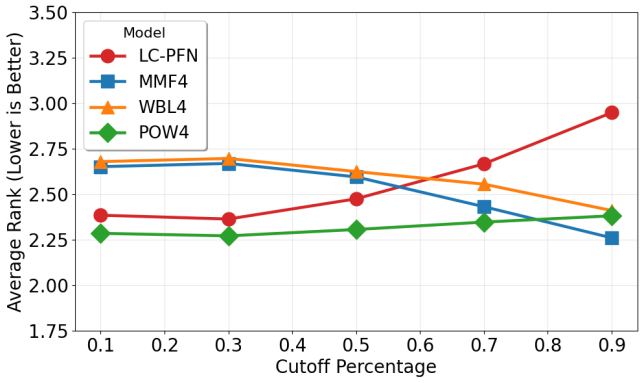

Figure 5: Model ranking across different learning curve cutoff percentages. LC-PFN demonstrates competitive performance at early cutoffs (10-50%) but degrades as more data becomes available, while POW4 maintains consistently strong performance across all cutoffs.

Statistical Significance Analysis

The statistical significance analysis (Figure 6) reveals that LC-PFN differs significantly from all parametric models, with very low p-values (ranging from $2.8e-8$ to $5.6e-10$). These unusually low p-values can also be attributed to the large sample size ($n = 1000$) used in the analysis. In contrast, comparisons between the parametric models (MMF4, POW4, WBL4) yielded higher p-values (ranging from 0.27 to 0.87), indicating that the performance differences between these models are not statistically significant.

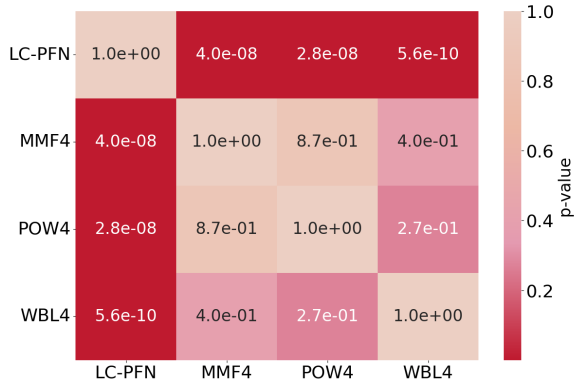

Figure 6: Statistical significance heatmap showing p-values for pairwise model comparisons using MAE. LC-PFN shows statistically significant differences from all parametric models, while comparisons between parametric models show higher p-values indicating non-significant differences.

5.2 Results of Experiment 2: Model Comparison Across Learning Curve Shapes and Cutoffs

This shape-specific evaluation of model performance provides some insights into the specialized strengths and weaknesses of both parametric and neural network approaches.

Figure 7 shows the average ranking performance of each model across the four learning curve shapes. POW4 and MMF4 emerge as the two strongest competitors, with their relative performance varying depending on learning curve shape. For Flat and MonoConv shapes, POW4 demonstrates a slight advantage over MMF4, while for Peaking and Dipping curves, MMF4 outperforms POW4. WBL4 consistently ranks third across all shape categories, while LC-PFN shows the weakest performance, particularly struggling with Dipping curves where it ranks substantially behind all parametric models.

The radar chart analysis (Figure 8) provides an alternative visualization of these performance patterns, displaying each model's relative strength profile across the four learning curve shapes. The plotted values represent normalized performance scores, calculated by transforming average ranks into "goodness" scores where higher values indicate better performance: $score = (max\_rank - model\_rank + 1)/max\_rank$. This transformation allows direct comparison of model strengths, with values near 1.0 indicating excellent performance and values near 0.0 indicating poor performance for that particular shape.

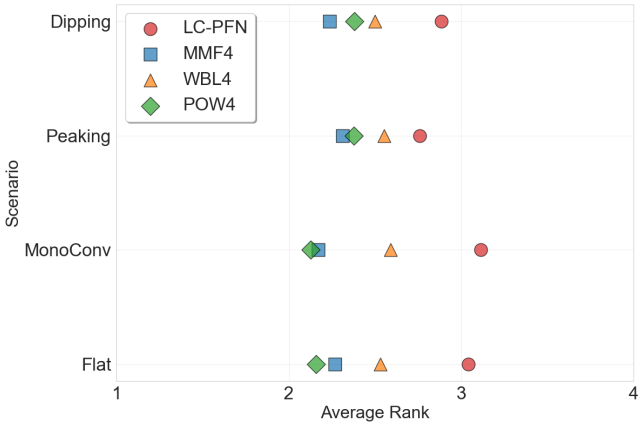

Figure 7: Average model rankings across four learning curve shape categories. POW4 and MMF4 compete closely, with POW4 showing advantages for Flat and MonoConv shapes while MMF4 excels at Peaking and Dipping curves. LC-PFN ranks worst.

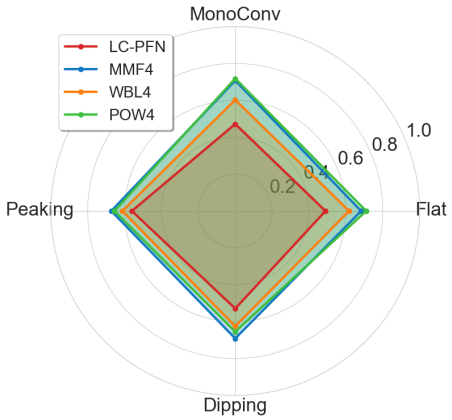

Figure 8: Radar chart showing normalized performance scores (0-1 scale) for each model across learning curve shapes. POW4 demonstrates balanced performance, MMF4 shows particular strength on complex curve morphologies.

Figure 9 synthesizes these findings into practical model selection guidance based on learning curve shape and cutoff percentage. The recommendations are determined by identifying the model with the lowest average rank for each shape-cutoff combination, while the color intensity represents recommendation confidence calculated from the performance gap between the best and second-best models. A rank difference of 0.5 or greater between the top two models results in maximum confidence (darkest color), indicating a clear performance advantage.

The matrix reveals that POW4 dominates recommendations for early-stage extrapolation (10-30% cutoffs) across all curve shapes, while MMF4 increasingly becomes the preferred choice at higher cutoff percentages (50-90%), particularly for Peaking and Dipping curves. This pattern suggests that MMF4's additional mathematical complexity provides advantages when sufficient data is available to prop-

erly estimate its parameters, especially for curves with more complex morphological characteristics. The confidence levels indicate that model selection becomes particularly critical for certain shape-cutoff combinations, with the highest confidence recommendations occurring where clear performance differences exist between competing models.

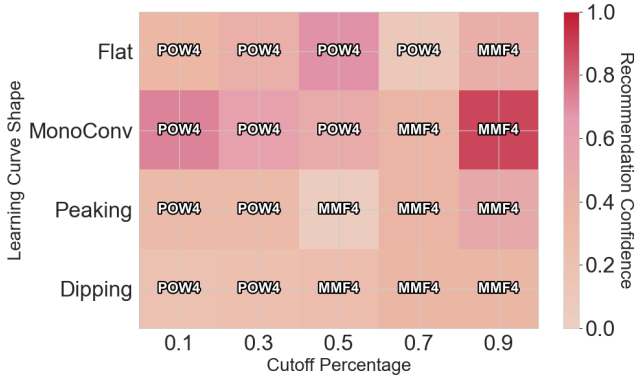

Figure 9: Model recommendation matrix based on lowest average rank performance, with color intensity indicating recommendation confidence. POW4 dominates early cutoffs while MMF4 becomes preferred at higher cutoffs for complex, non-traditional shapes (i.e. Peaking and Dipping).

6 Discussion

Our comprehensive evaluation across LCDB 1.1 reveals counterintuitive findings that challenge expectations about neural network superiority in learning curve extrapolation. Parametric models, particularly POW4 and MMF4, consistently outperformed LC-PFN across most scenarios, contradicting the assumption that neural networks should excel at capturing complex patterns in learning curve data.

However, a crucial finding emerges from the cutoff analysis, offering insight into **RQ2**: LC-PFN ranked second-best at 10%, 30%, and 50% cutoffs but showed degraded performance as cutoff percentages increased. This pattern reveals that LC-PFN demonstrates particular strength in extrapolating from limited early-stage learning curve data, suggesting real potential for neural approaches when practitioners have access to only small portions of learning curves.

Beyond this, a nuanced trend emerges when investigating **RQ3** considering the shape of the learning curve. LC-PFN consistently lagged behind all parametric models, but its relative performance varied depending on the curve's morphology. It showed slightly improved rankings on more irregular curves—particularly those with peaking or dipping behaviors—suggesting some adaptability to non-monotonic patterns, even though it still trailed behind parametric baselines like MMF4 and POW4. On smoother or flatter curves, where more predictable shape patterns dominate, LC-PFN performed comparatively worse. This indicates that while the neural model struggles broadly, it may be somewhat more sensitive to complex curve dynamics, albeit not robust enough to consistently compete with the more tailored inductive structures of parametric approaches.

Important Disclaimer: The LC-PFN architecture used in this study was trained exclusively on curves of fixed length (`SEQ_LEN = 80`), which limits interpretation of these results. This constraint means the model never encountered curves of varying lengths during training, fundamentally undermining its ability to generalize across the diverse curve lengths in LCDB 1.1. As a result, the KDKL set, having fewer fixed-length curves, is disproportionately reduced when applying this length filter, potentially misrepresenting LC-PFN's performance in this scenario. This limitation particularly impacts transfer scenario analysis when addressing **RQ1** and may help explain why LC-PFN underperforms even in settings where strong generalization should not be necessary.

Several other factors may contribute to LC-PFN's suboptimal performance beyond early cutoffs. The model could be overfitting to its synthetic training distribution, which may inadequately capture the 14% prevalence of ill-behaved curves in LCDB 1.1. Additionally, the modest training configuration, imposed by the computational and time constraints of this study, may lack sufficient representational capacity for the diverse patterns across learning curves.

Conversely, parametric models' consistent superiority stems from their mathematical inductive biases that align well with common learning curve behaviors. Unlike neural networks, parametric models require no training—only fitting to each individual curve—making them computationally efficient and robust to distribution shifts. These built-in advantages give them an edge in extrapolation, particularly when computational budgets are limited or when curve-by-curve adaptability is prioritized.

Practitioners should consider not only predictive performance but also computational commitments and flexibility when choosing between LC-PFNs and parametric models.

7 Conclusions and Future Work

This research demonstrates that parametric models, particularly POW4 and MMF4, consistently outperform LC-PFNs for learning curve extrapolation across most scenarios in LCDB 1.1. However, the critical finding that LC-PFN performs second-best at early cutoff percentages (10-50%) reveals significant potential for neural approaches when extrapolating from limited data, precisely the scenario most valuable to practitioners seeking early performance predictions.

The identified architectural limitation requiring fixed-length sequences severely constrains current LC-PFN applicability and likely explains much of its underperformance. Future work should prioritize developing LC-PFN architectures that train on curves of varying lengths rather than fixed sequences, which we expect to considerably improve neural network performance. Additionally, exploring larger, more expressive architectures and ensemble strategies combining parametric and neural approaches could leverage the complementary strengths revealed by this analysis.

These findings suggest that while parametric models remain the more reliable general choice, neural networks show particular promise for early-stage extrapolation scenarios, warranting continued development of improved architectures that address current limitations.

8 Responsible Research

To ensure full transparency and reproducibility, all code used in this study is publicly available on GitHub, including data processing methods, model training, and evaluation notebooks. The repository includes detailed documentation with setup instructions, and dependency specifications (`requirements.txt`). Random seeds were fixed across all experiments (`seed=42`) to guarantee consistent results across runs. Additionally, the pre-trained weights of the LC-PFN model is contained within the same codebase, allowing others to replicate the findings without retraining from scratch.

All experimental configurations, hyperparameters, and model architectures are documented in the codebase. This enables future researchers to verify results, build upon the work, or benchmark against the same experimental conditions with minimal setup overhead.

This research uses the publicly available Learning Curve Database (LCDB) v1.1 and OpenML datasets - all freely accessible for research purposes. No personally identifiable or sensitive information was involved in this study, as all learning curves represent aggregated performance metrics from ML algorithms. We acknowledge the original contributors to LCDB and OpenML, and encourage users of our code to maintain proper attribution to these foundational resources.

In the interest of transparency, we acknowledge the use of large language models (LLMs) to assist with specific aspects of this research. Artificial Intelligence (AI) tools were employed solely for language refinement—polishing sentence structure and improving text flow while preserving all original research content, ideas, and findings as our own intellectual contribution. Additionally, LLMs were used to generate initial code templates and structural frameworks for data visualization, serving as starting points that were then extensively modified and refined. All final visualizations, analyses, interpretations, and scientific conclusions represent our independent work, with AI assistance limited to enhancing presentation quality and maintaining aesthetic consistency across figures. For a more detailed breakdown of AI usage, consult Section C from the Appendix.

Given the growing concern about the environmental impact of ML research, we report the computational resources used in this study. The LC-PFN training required approximately 1 GPU-hour on NVIDIA Tesla P100, consuming $0.25kWh$ of electricity. The estimated carbon footprint is $0.125kgCO_2$ equivalent, calculated using the methodology from Strubell et al. [5] with global average electricity carbon intensity factors.

This research contributes to the fundamental understanding of when different approaches are most effective for learning curve extrapolation. Positive practical implications include:

- Helping practitioners make more informed decisions about data collection strategies
- Reducing computational waste through better model selection guidance
- Contributing to more efficient ML workflows

We encourage readers to view these results as contributing to an ongoing scientific dialogue rather than definitive answers, and to conduct domain-specific validation before making significant methodological commitments.

References

[1] Steven Adriaensen et al. "Efficient Bayesian Learning Curve Extrapolation using Prior-Data Fitted Networks". In: *NeurIPS*. 2023.

[2] Baohua Gu, Feifang Hu, and Huan Liu. "Modelling Classification Performance for Large Data Sets". In: *Advances in Web-Age Information Management*. Ed. by X. Sean Wang, Ge Yu, and Hongjun Lu. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 317–328. ISBN: 978-3-540-47714-3.

[3] Prasanth Kolachina et al. "Prediction of Learning Curves in Machine Translation". In: *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Haizhou Li et al. Jeju Island, Korea: Association for Computational Linguistics, July 2012, pp. 22–30. URL: https://aclanthology.org/P12-1003/.

[4] Vladik Kreinovich, Hung T. Nguyen, and Rujira Ouncharoen. *How to Estimate Forecasting Quality: A System-Motivated Derivation of Symmetric Mean Absolute Percentage Error (SMAPE) and Other Similar Characteristics*. Technical Report UTEP-CS-14-53. University of Texas at El Paso, Department of Computer Science, July 2014. URL: https://scholarworks.utep.edu/cs_techrep/865.

[5] Alexandre Lacoste et al. *Quantifying the Carbon Emissions of Machine Learning*. 2019. arXiv: 1910.09700 [cs.CY]. URL: https://arxiv.org/abs/1910.09700.

[6] Marco Loog and Tom Viering. *A Survey of Learning Curves with Bad Behavior: or How More Data Need Not Lead to Better Performance*. 2022. arXiv: 2211.14061 [cs.LG]. URL: https://arxiv.org/abs/2211.14061.

[7] Felix Mohr and Jan N. van Rijn. "Fast and informative model selection using learning curve cross-validation". In: *IEEE TPAMI* (2023).

[8] Felix Mohr et al. "LCDB 1.0: An extensive learning curves database for classification tasks". In: *ECML PKDD*. Springer, 2022, pp. 3–19.

[9] Connor Shorten and Taghi M. Khoshgoftaar. "A survey on Image Data Augmentation for Deep Learning". In: *Journal of Big Data* 6.1 (July 2019), p. 60. ISSN: 2196-1115. DOI: 10.1186/s40537-019-0197-0. URL: https://doi.org/10.1186/s40537-019-0197-0.

[10] Tom Viering and Marco Loog. "The shape of learning curves: a review". In: *IEEE TPAMI* 45.6 (2022), pp. 7799–7819.

[11] Tom Julian Viering et al. "From Epoch to Sample Size: Developing New Data-driven Priors for Learning Curve Prior-Fitted Networks". In: *AutoML Conference 2024 (Workshop Track)*. 2024. URL: https://openreview.net/forum?id=neEKHQDTHV.

[12] Cheng Yan, Felix Mohr, and Tom Viering. *LCDB 1.1: A Database Illustrating Learning Curves Are More Ill-Behaved Than Previously Thought*. 2025. arXiv: 2505.15657 [cs.LG]. URL: https://arxiv.org/abs/2505.15657.

A Experiment 1: Additional Resulting Plots

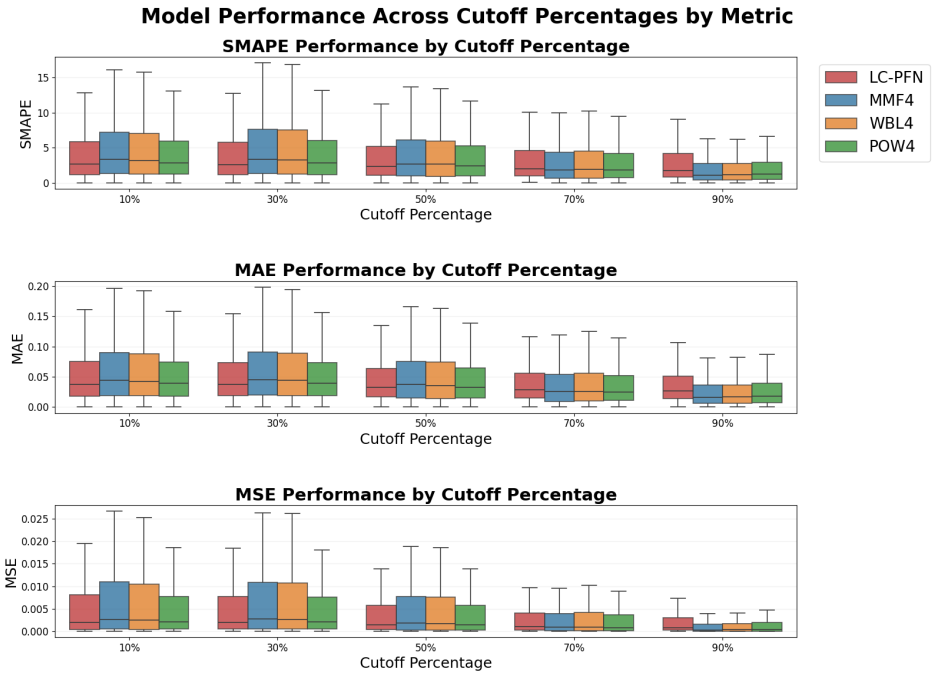

Figure 10: Model performance across different cutoff percentages (10%–90%) for SMAPE, MAE, and MSE. Lower is better.

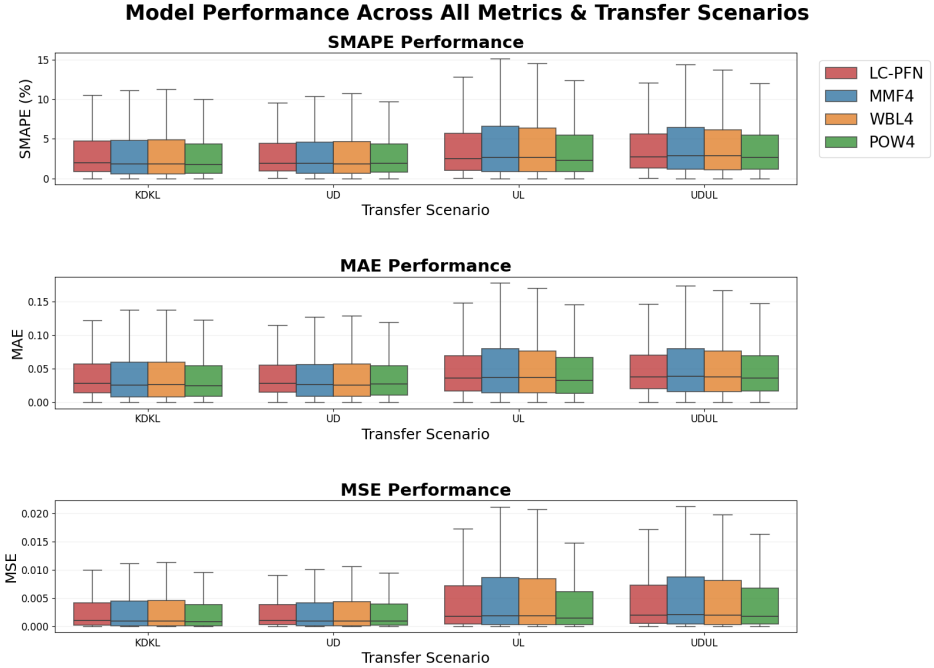

Figure 11: Model performance across four transfer scenarios: Known Dataset Known Learner (KDKL), Unseen Dataset (UD), Unseen Learner (UL), and Unseen Dataset Unseen Learner (UDUL). Lower is better.

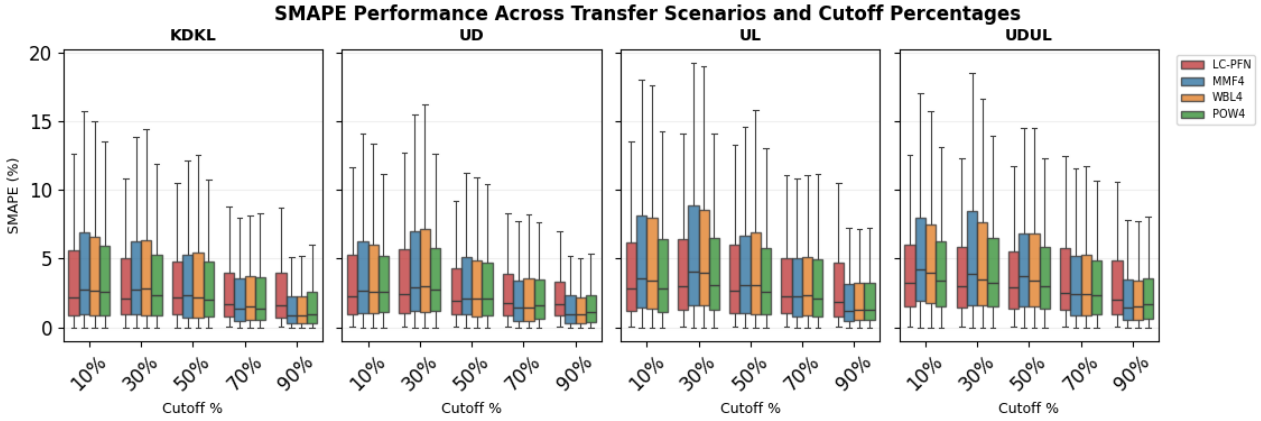

Figure 12: Performance indicated by SMAPE across all transfer scenarios and cutoffs. 4 transfer scenarios x 5 cutoff points = 20 experimental conditions. Lower is better.

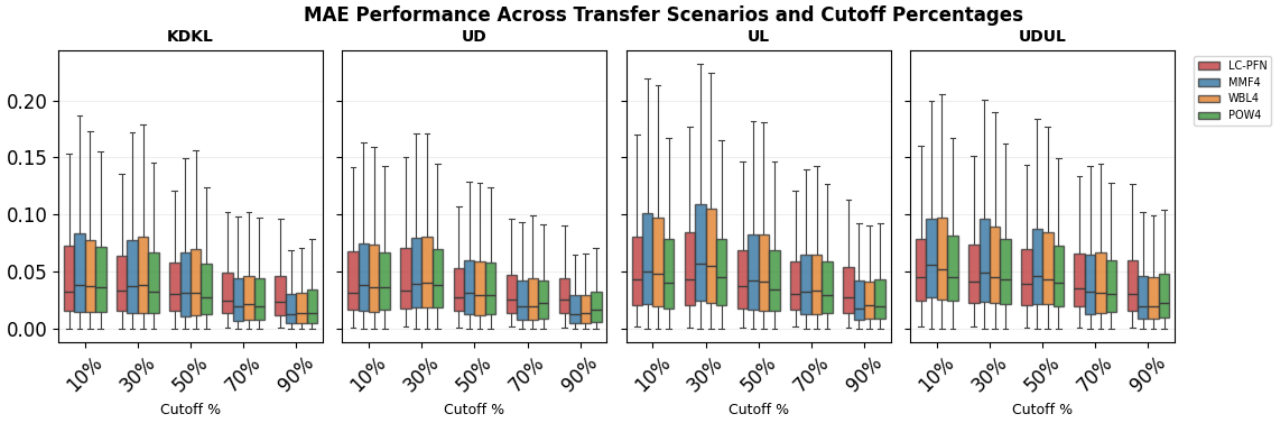

Figure 13: Performance indicated by MAE across all transfer scenarios and cutoffs. 4 transfer scenarios x 5 cutoff points = 20 experimental conditions. Lower is better.

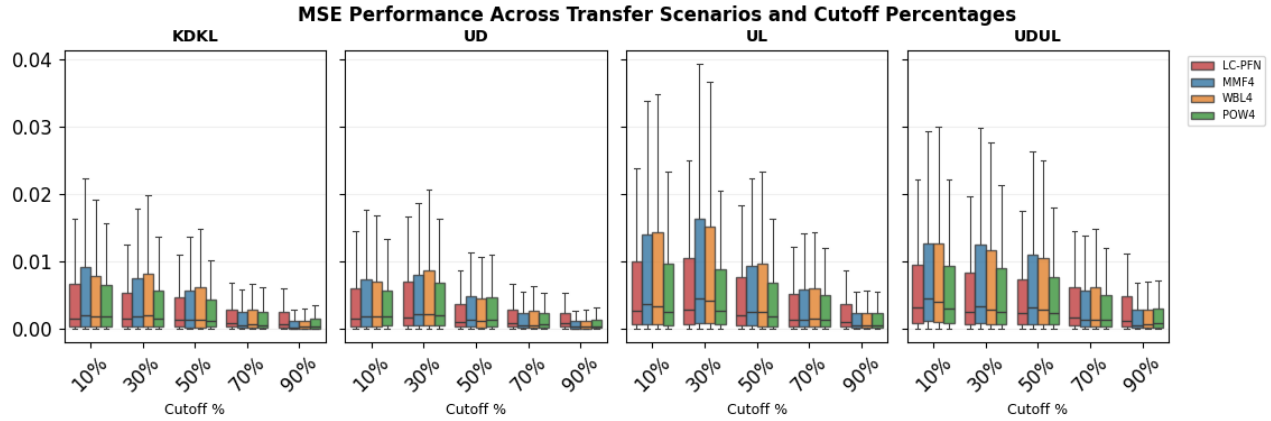

Figure 14: Performance indicated by MSE across all transfer scenarios and cutoffs. 4 transfer scenarios x 5 cutoff points = 20 experimental conditions. Lower is better.

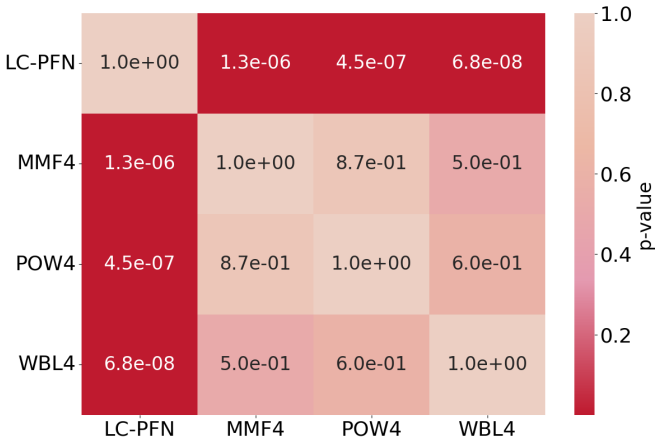

Figure 15: Statistical significance heatmap showing p-values for pairwise model comparisons using SMAPE.

B Experiment 2: Additional Resulting Plots

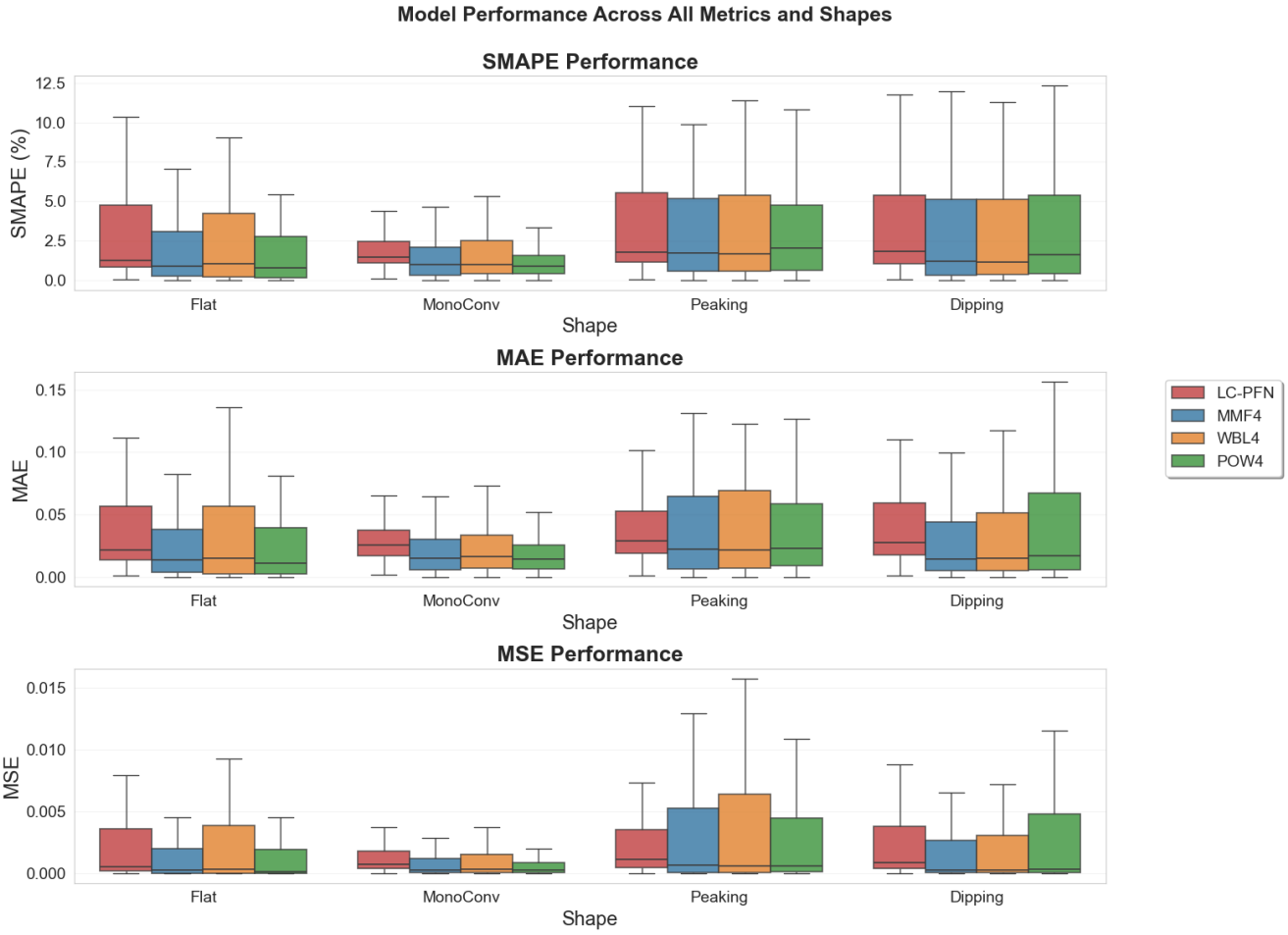

Figure 16: Model performance across All Metrics (SMAPE, MAE, MSE) and Shapes (Flat, Monotone & Convex, Peaking, Dipping). Lower is better.

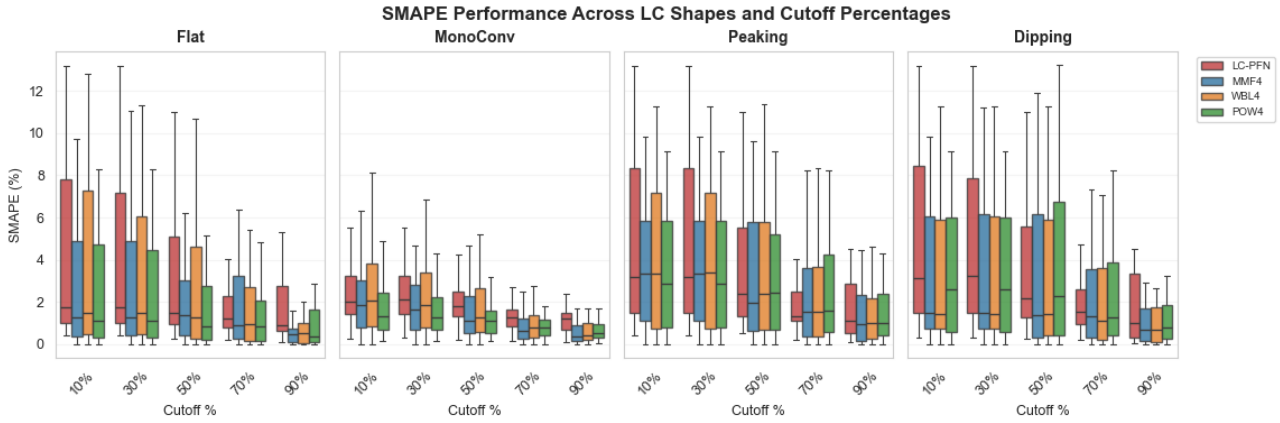

Figure 17: Performance indicated by SMAPE across all shapes and cutoffs. 4 shapes x 5 cutoff points = 20 experimental conditions. Lower is better.

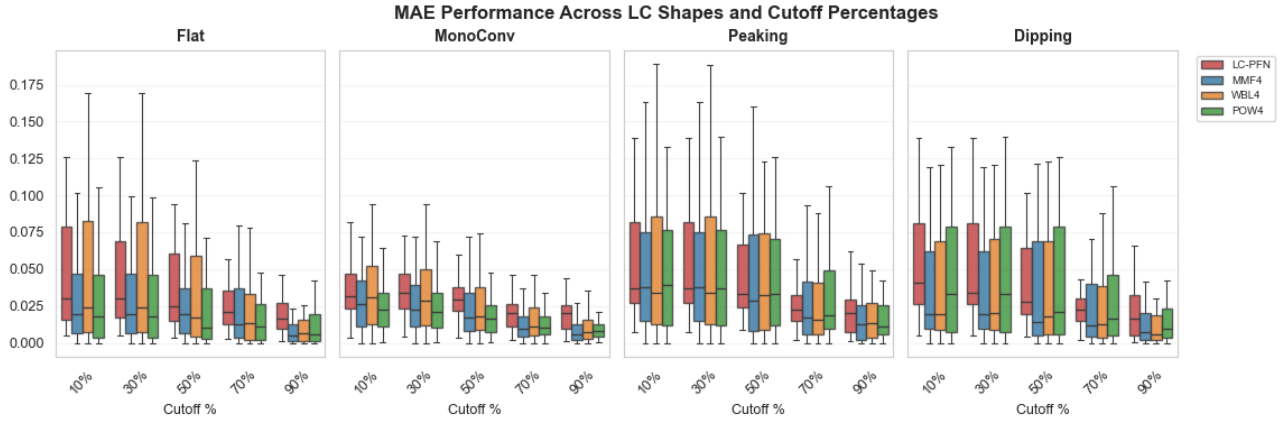

Figure 18: Performance indicated by MAE across all shapes and cutoffs. 4 shapes x 5 cutoff points = 20 experimental conditions. Lower is better.

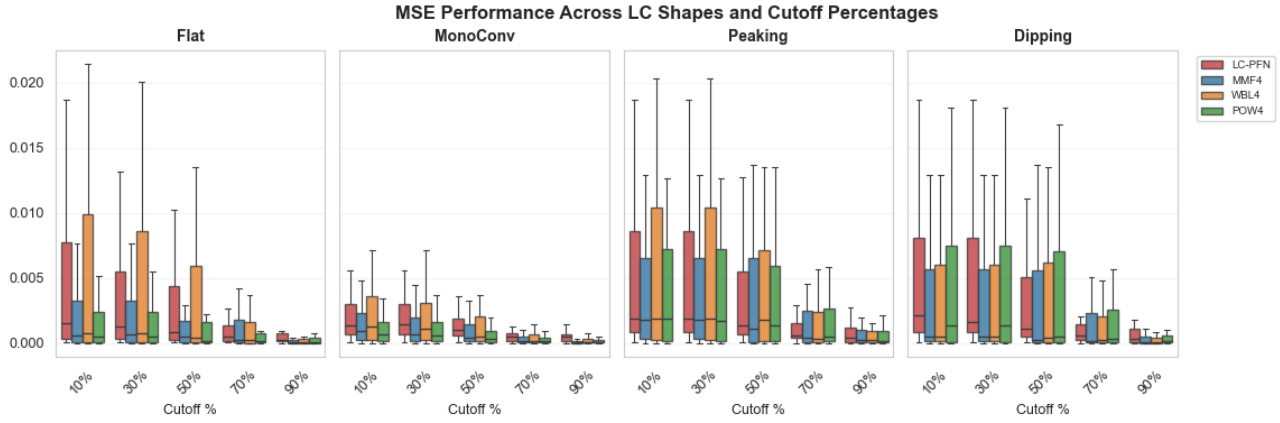

Figure 19: Performance indicated by MSE across all shapes and cutoffs. 4 shapes x 5 cutoff points = 20 experimental conditions. Lower is better.

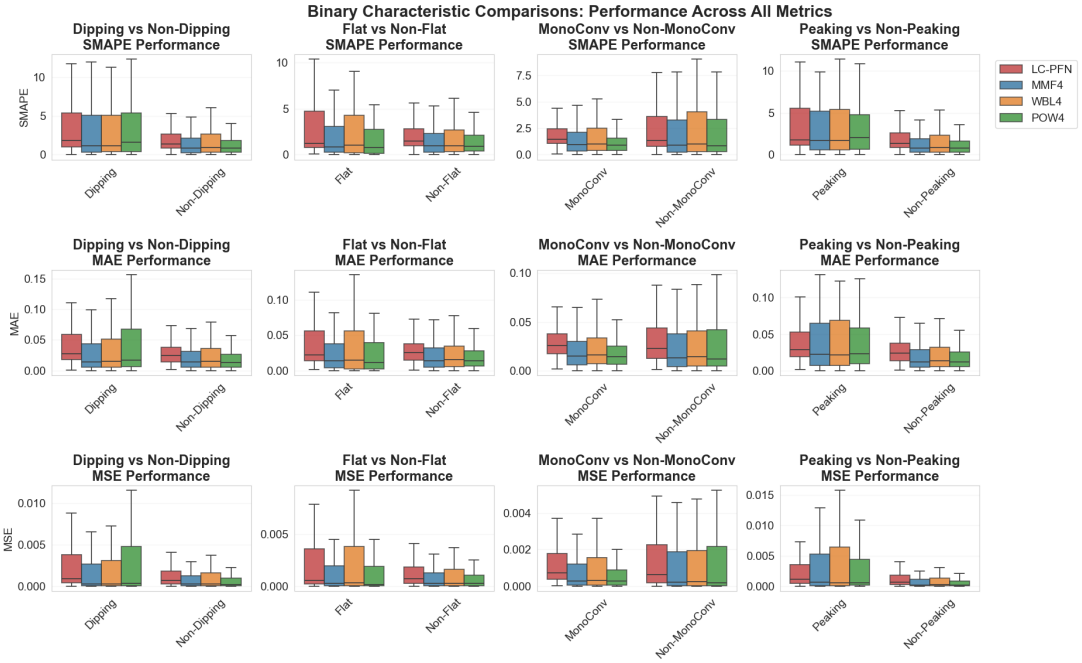

Figure 20: Binary Shape Comparisons across All Metrics. Eg. 'Dipping' is the set of samples with curves displaying this shape, 'Non-Dipping' is the set of samples containing curves with less dipping characteristics. Lower is better.

C Artificial Intelligence (AI) Usage Disclosure

AI Assistance Category	Used	Not Used
Language refinement and syntax improvement	✓	
Initial code structure for visualization/plot templates	✓	
Aesthetic consistency across figures and plots	✓	
Research conceptualization and hypothesis formulation		✓
Data collection and experimental design		✓
Statistical analysis and result interpretation		✓
Scientific conclusions and discussion of findings		✓
Literature review and citation selection		✓

Table 1: Scope of AI tool utilization in research workflow.

AI assistance was applied selectively after independent development of all core research content. Language enhancement tools were used to improve paragraph clarity while preserving technical accuracy, and initial visualization frameworks were generated and subsequently customized to meet specific analytical requirements. All AI-assisted outputs underwent careful review to maintain consistency with research objectives.
To illustrate the scope of AI assistance, some typical requests for language enhancement were structured as:

> *"Improve the readability of this technical paragraph while maintaining all specific terminology and quantitative details. Focus on sentence structure and transitions without altering the scientific content or conclusions."*

> *"Create a function that takes data X and processes it by [detailed description of processing method], then displays the results as a [detailed description of the type of plot/visualisation]."*

D Supporting Code

The supporting code can be found at the following GitHub address: https://github.com/adelinacazacu/Extrapolating-Learning-Curves-When-Do-Neural-Networks-Outperform-Parametric-Models-.