Do Graph Neural Networks Follow Power Laws?

Vasil Georgiev[1]

Supervisors: Elvin Isufi[1]**, Mohamed Jebali**[1]**, Chengen Liu**[1]

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

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 21, 2026

Name of the student: Vasil Georgiev
Final project course: CSE3000 Research Project
Thesis committee: Elvin Isufi, Mohamed Jebali, Chengen Liu, Tom Viering

Abstract

Learning curves describe how model performance changes as more labeled data becomes available and can help estimate whether collecting additional labels is worthwhile. However, it remains unclear which mathematical functions best represent and extrapolate learning curves for graph neural networks. This study compares power-law and exponential models for learning curves generated by a graph neural network on node-classification datasets with different graph characteristics. The models are evaluated separately on how well they describe observed performance and how accurately they predict performance at larger, unseen labeling budgets. The results show that neither model family is universally preferable. Exponential models provide better descriptive fit on some datasets, while power-law models provide better descriptive fit on others. In the extrapolation experiments, power-law models often give more accurate predictions at larger labeled-node budgets, although the preferred model still depends on the dataset and fitting range. These findings indicate that descriptive fit and extrapolation accuracy should be treated as separate objectives. Overall, power-law behaviour appears to be a useful modelling assumption for some GNN learning curves, especially for extrapolation, but it should not be assumed to hold universally.

1 Introduction

Graph Neural Networks (GNNs) have become one of the dominant approaches for learning on graph-structured data and are widely applied to tasks such as social network analysis, recommendation systems, biological network modeling, and semi-supervised node classification [14]. In many of these applications, obtaining labeled data is expensive, time-consuming, or requires domain expertise. As a result, understanding how model performance changes as the number of labeled nodes increases is an important practical and scientific problem.

Learning curves provide a natural framework for studying this behavior. They describe how the performance of a learning algorithm changes as a function of a resource, most commonly the number of training examples. Beyond simply visualizing performance trends, learning curves are useful for estimating the value of collecting additional data, comparing methods under different labeling budgets, and predicting future model performance from limited observations [10; 12]. These properties make learning curves particularly relevant in semi-supervised node classification, where the number of labeled nodes is often limited.

The shape of learning curves has been studied extensively in the broader machine learning literature. Common parametric families include power-law and exponential models, both of which have been used to characterize performance scaling behavior [12]. More recently, research on neural scaling laws has shown that deep learning systems often exhibit approximately power-law behavior across model size, compute, and dataset scale [5; 6]. However, these findings are primarily focused on domains such as computer vision and natural language processing, while graph neural networks remain comparatively underexplored.

Knowledge gap. Recent work has begun investigating scaling behavior in graph learning settings [8], but there is still limited understanding of how GNN learning curves behave in semi-supervised node classification when the number of labeled nodes varies. Existing graph-scaling studies haven't directly compared different parametric learning-curve models and evaluating how accurately these models extrapolate future performance from small labeled-node budgets.

Contributions. This paper investigates whether power-law or exponential models better describe GNN learning curves and whether these models can reliably extrapolate performance to larger labeled-node regimes. To answer this question, learning curves are generated for multiple datasets, with varying labeling budgets. The resulting curves are then fitted using both parametric families and evaluated in terms of descriptive fit quality and extrapolation accuracy.

The main contributions of this work are as follows:

- A comparative evaluation of power-law and exponential learning-curve models for GNN node classification.
- An analysis of extrapolation accuracy when predicting performance from limited labeled-node budgets.
- An empirical study of learning-curve behavior across multiple graph datasets and experimental settings.

Outline. The remainder of this paper is organized as follows. Section 2 discusses background literature on learning curves, scaling laws, and graph neural networks. Section 3 and 4 present the methodology and experimental setup. Section 5 reports the experimental results and extrapolation analysis. Section 6 discusses the implications and limitations of the findings. Section 7 is dedicated to responsible research and Section 8 concludes the paper and outlines directions for future work.

2 Related Work

A central question in learning-curve analysis is how to model the relationship between training set size and performance. Power-law models are often used because many empirical learning curves show diminishing returns as training data increases. Exponential models are also commonly considered, especially when performance appears to approach an asymptote quickly. Prior work has therefore treated both power-law and exponential functions as plausible candidates for modeling learning curves [12]. This motivates the comparison in this work between a power-law model with offset and an exponential model with offset.

Learning curves are also useful for decision-making, not only for retrospective analysis. Mohr and van Rijn [10] survey the use of learning curves in supervised machine learning and describe applications such as data acquisition, model selection, and early stopping. In these settings, the goal is often

not only to fit observed performance values, but also to predict future performance at larger budgets. This distinction is important for the present work, since a model that fits the observed part of a curve well may still extrapolate poorly to unseen labeled-node budgets.

Learning-curve extrapolation has also been studied in the context of hyperparameter optimization and automated machine learning. For example, Gargiani et al. [4] propose probabilistic models for extrapolating learning curves across hyperparameter settings , while Adriaensen et al. [1] propose a Bayesian learning-curve extrapolation method based on prior-data fitted networks. These works show that extrapolation is a challenging problem because learning curves can be noisy, incomplete, and difficult to predict from early observations. However, they mainly focus on training progress across epochs or hyperparameter configurations, rather than on the relationship between labeled-node budget and performance in graph neural networks.

In modern deep learning, power-law behavior has received particular attention through the literature on neural scaling laws. Hestness et al. [5] empirically show that generalization error often follows predictable power-law trends as dataset size and model size increase across several deep learning domains, including language modeling, machine translation, image processing, and speech recognition. This work provides motivation for considering power-law models as strong candidates for neural learning curves. However, these results are mostly based on non-graph domains, so it is not obvious that the same assumptions should hold for graph neural networks.

Graph neural networks introduce additional structure because samples are not independent in the same way as in standard supervised learning. Wu et al. [14] review GNN methods and applications, showing their importance for tasks such as node classification, link prediction, and graph classification. More recently, Liu et al. investigate neural scaling laws on graphs and argue that graph learning requires graph-specific notions of scale, since graph datasets can vary in the number of nodes, edges, and graphs [8]. Their work shows that scaling behavior in graph learning is an active research direction, but it does not directly answer which simple parametric model best describes GNN learning curves when the number of labeled nodes is varied in semi-supervised node classification.

Recent work on graph self-supervised learning further suggests that scaling behavior in graph learning may differ from the behavior observed in other deep learning domains. Ma et al. study whether graph self-supervised learning follows neural scaling laws and find that downstream performance does not necessarily improve consistently with increased scale [9]. This supports the need to empirically test scaling assumptions in graph settings rather than directly transferring conclusions from vision or language models.

Overall, previous work has studied learning-curve shapes, parametric models, extrapolation methods, deep-learning scaling laws, and graph-specific scaling behavior. However, there remains limited evidence on whether power-law or exponential models better fit and extrapolate GNN learning curves for semi-supervised node classification under varying labeled-node budgets. This work addresses that gap by comparing these two parametric model families in terms of both fit quality and extrapolation accuracy.

3 Methodology

This chapter describes the general methodology used to construct, model, and evaluate learning curves for semi-supervised node classification. The dataset-specific choices, graph neural network configuration, labeled-node budgets, and concrete fitting windows are described separately in the Experimental Setup.

3.1 Formal Problem Description

Let $G = (V, E)$ be a graph with node set V and edge set E. Each node $v_i \in V$ has a feature vector $\mathbf{x}_i \in R^d$ and, for a subset of nodes, an associated class label $y_i \in \{1, \ldots, C\}$. The task considered in this work is semi-supervised node classification, where only a limited number of labeled nodes is available for training, while model performance is evaluated on a fixed test set.

The main object of study is the learning curve of a graph neural network as a function of the labeled-node budget. Let n denote the number of labeled nodes used for training. For each budget n, a training set $\mathcal{T}_n \subset V$ is sampled such that $|\mathcal{T}_n| = n$. A graph neural network $h_{\theta,n}$ is then trained using the labels in $\mathcal{T}_n$ while using the available graph structure and node features. The resulting model is evaluated on a fixed test set, producing a performance value $m(n)$, such as accuracy, classification error, cross-entropy, or macro-F1.

The empirical learning curve is defined as the sequence

$$\mathcal{L} = \{(n_i, \bar{m}(n_i))\}_{i=1}^{K},$$

where n_i is a labeled-node budget and $\bar{m}(n_i)$ is the aggregated test performance at that budget. The objective is to determine whether a power-law model or an exponential model better describes the empirical learning curve and which model provides more accurate extrapolation to larger labeled-node budgets.

3.2 Learning-Curve Construction

For each labeled-node budget, the graph neural network is trained repeatedly using independently sampled labeled sets and different model-training random seeds. The resulting performance values are aggregated to produce one mean performance value per budget. This aggregation reduces the effects of randomness in labeled-node sampling, model initialization, and optimization.

For a metric m, the empirical learning curve is computed as

$$\bar{m}(n_i) = \frac{1}{R} \sum_{r=1}^{R} m_r(n_i),$$

where R is the number of repeated runs and $m_r(n_i)$ is the test performance at budget n_i in repetition r.

Accuracy and macro-F1 increase with performance, whereas conventional learning-curve models are commonly

formulated as decreasing error curves. Therefore, an increasing performance metric is converted to a corresponding error quantity before curve fitting when necessary. For example, accuracy is converted to classification error according to

$$e(n) = 1 - \text{accuracy}(n).$$

After fitting, predicted error values can be converted back to the original performance scale for interpretation and visualization.

3.3 Parametric Learning-Curve Models

Two parametric model families are compared: a power-law model with an offset and an exponential model with an offset. Both models contain three parameters, allowing them to be compared without introducing a difference in model complexity.

The comparison is motivated by prior learning-curve literature, where both power-law and exponential functions are commonly considered plausible models for well-behaved learning curves. Since the goal of this work is not only to describe observed performance but also to evaluate extrapolation to larger labeled-node budgets, comparing these two model families provides a direct test of whether GNN learning curves are better explained by gradual scaling behaviour or by rapid saturation [3; 12].

The power-law model is defined as

$$y(x) = ax^{-b} + c.$$

This model captures diminishing returns: the error decreases rapidly at smaller labeled-node budgets and progressively more slowly as the budget grows.

The exponential model is defined as

$$y(x) = ae^{-bx} + c.$$

This model also represents rapid initial improvement, but it approaches its asymptote more quickly than the power-law model.

In both models, a controls the scale of the curve, b controls the decay rate, and c represents the estimated asymptotic error level. Although both families capture diminishing improvements as the labeled-node budget increases, they make different assumptions about the rate at which the curve approaches its asymptote. The power-law model permits slower long-term decay and is commonly associated with scaling-law behaviour, whereas the exponential model represents stronger saturation.

3.4 Curve Fitting Procedure

The parameters of the power-law and exponential models are estimated using the curve_fit function from scipy.optimize [13]. This function fits a user-defined curve to observed data by choosing parameter values that minimize the squared differences between the observed learning-curve points and the values predicted by the curve.

In this work, the input to the fitting procedure is the labeled-node budget, and the target value is the corresponding mean test error. Since the reported performance metric is accuracy, accuracy is first converted to error before fitting. After the curves are fitted, the predicted errors are converted back to accuracy for visualization.

The same fitting procedure is used for both the power-law and exponential models, and for all datasets and fitting windows. This ensures that differences in fit and extrapolation performance are caused by the choice of curve model rather than by differences in the fitting method. The fitted parameters are then used either to evaluate how well the model describes the observed points or to predict performance at larger held-out labeled-node budgets

3.5 Extrapolation Evaluation

In addition to describing observed learning-curve points, the parametric models are evaluated according to their ability to extrapolate to larger labeled-node budgets.

Let $\mathcal{B}_{\text{fit}}$ denote the budgets included in a fitting window. The parameters of each curve model are estimated using only the observations associated with these budgets. The fitted model is subsequently used to predict performance at a set of held-out budgets $\mathcal{B}_{\text{ext}}$, where

$$n_j > \max \mathcal{B}_{\text{fit}} \qquad \text{for all } n_j \in \mathcal{B}_{\text{ext}}.$$

The predictions are compared with the empirical performance values obtained by training the graph neural network at the held-out budgets. This procedure tests whether observations from small and intermediate labeled-node regimes are sufficient to predict performance when more labels become available.

Descriptive fit and extrapolation are evaluated separately. A model can fit the observed points closely while nevertheless producing inaccurate predictions outside the fitting range. Conversely, a model with a slightly poorer in-window fit may better represent the long-term trend of the learning curve.

3.6 Evaluation Metrics

Curve quality is assessed in two settings. Descriptive fit quality measures how closely a parametric curve matches the observed points within the fitting window. Extrapolation quality measures how accurately the fitted curve predicts the held-out observations at larger labeled-node budgets.

For both settings, root mean squared error (RMSE) is used:

$$\text{RMSE} = \sqrt{\frac{1}{K}\sum_{i=1}^{K}(y_i - \hat{y}_i)^2},$$

Here, y_i is the observed empirical value and $\hat{y}_i$ is the value predicted by the fitted curve. For descriptive fit, the summation is taken over the budgets within the fitting window. For extrapolation, it is taken over the held-out larger budgets.

When an increasing metric is converted to an error quantity for fitting, the observed and predicted values are compared on the same scale. Predictions may subsequently be transformed back to the original metric scale for visualization. The final comparison between the power-law and exponential models considers both descriptive fit quality and extrapolation accuracy.

4 Experimental Setup

This chapter describes how the methodology is instantiated in the experiments. It introduces the datasets and split policies, the graph neural network used to construct the empirical learning curves, the labeled-node sampling protocol, and the exact fitting and extrapolation configurations.

4.1 Datasets and Split Policy

The experiments use four node-classification datasets covering both homophilic and heterophilic graph settings. Edge homophily, denoted by h, is the proportion of edges connecting nodes with the same class label; higher values therefore indicate stronger similarity between connected nodes.

Cora. Cora [7] is a homophilic citation network with 2,708 nodes, 10,556 directed edges, 1,433 node features, 7 classes, and $h \approx 0.81$. The standard PyTorch Geometric split is used, with 500 validation nodes and 1,000 test nodes. The remaining 1,208 nodes form the training candidate pool, allowing label budgets up to 640.

PubMed. PubMed [15] is a homophilic citation network with 19,717 nodes, 88,648 directed edges, 500 node features, 3 classes, and $h \approx 0.80$. Using the standard PyTorch Geometric split, 500 nodes are reserved for validation and 1,000 for testing, leaving 18,217 training candidates and a maximum label budget of 1,280.

Chameleon-filtered. Chameleon-filtered [11] is a heterophilic Wikipedia page network with 890 nodes, 17,708 directed edges, 2,325 node features, 5 classes, and $h \approx 0.23$. The filtered variant removes duplicate nodes that may inflate performance. Its 10 predefined splits leave at least 409 training candidates, permitting label budgets up to 320.

Squirrel-filtered. Squirrel-filtered [11] is a heterophilic Wikipedia page network with 2,223 nodes, 93,996 directed edges, 2,089 node features, 5 classes, and $h \approx 0.20$. The filtered dataset provides 10 predefined splits with between 1,047 and 1,085 training candidates, allowing label budgets up to 640.

For each dataset, validation and test nodes remain fixed within a split while labeled training nodes are sampled from the corresponding candidate pool. Only the sampled labels are used for supervision, while message passing is performed over the complete graph, following the transductive semi-supervised node-classification setting.

4.2 Labeled-Node Budgets and Sampling Protocol

The nominal labeled-node budget grid is

$$n \in \{5, 10, 20, 40, 80, 160, 320, 640, 1280\}.$$

These budgets follow an approximately geometric progression, doubling at each step. This provides coverage across both low-label and higher-label regimes while keeping the total number of budget levels manageable. The spacing also allows learning-curve behaviour to be observed over multiple scales of labeled-node availability, which is particularly useful when comparing power-law and exponential growth patterns. A budget is included for a dataset only when it does not exceed the size of the corresponding training candidate pool. The resulting feasible budget sets are summarized in Table 1.

Table 1: Feasible labeled-node budgets for each dataset.

Dataset	Feasible labeled-node budgets
Cora	$5, 10, 20, 40, 80, 160, 320, 640$
PubMed	$5, 10, 20, 40, 80, 160, 320, 640, 1280$
Chameleon-filtered	$5, 10, 20, 40, 80, 160, 320$
Squirrel-filtered	$5, 10, 20, 40, 80, 160, 320, 640$

At each feasible budget, n nodes are sampled uniformly at random from the training candidate pool. A new labeled subset is sampled for each repeated run, so each budget-level estimate reflects both model-training variability and variation caused by the selection of labeled nodes.

4.3 Graph Neural Network Model

The empirical learning curves are generated using a two-layer ChebNet [2] with 64 hidden channels and Chebyshev filters of order three. Symmetric graph normalization is applied, and dropout with a rate of 0.5 is used to reduce overfitting. The model is trained with Adam using a learning rate of 0.01 and weight decay of 5×10^{-4} for at most 200 epochs. Training is stopped early when the validation cross-entropy does not improve for 50 consecutive epochs. The configuration was informed by established settings for semi-supervised graph convolutional networks [7], including the optimizer, learning rate, regularization, dropout, and training duration. These settings were kept fixed across datasets and labeled-node budgets to avoid introducing dataset-specific tuning into the learning-curve comparison. Preliminary experiments produced classification performance comparable to values reported in the literature, indicating that the configuration provides a suitable baseline for the present analysis.

4.4 Primary Fitting and Extrapolation Configuration

The extrapolation experiments are designed to test whether the early part of a learning curve can predict performance at larger labeled-node budgets. For each dataset, the curve models are fitted using a prefix of the observed budget sequence, and the remaining larger budgets are held out for evaluation.

The fitting prefixes are chosen so that each model is estimated from multiple observed points while still leaving later budgets for extrapolation. In general, the last one or two feasible budgets of each dataset are held out as extrapolation targets, depending on how many budget levels are available for that dataset. This creates a forecasting setting in which the model must predict performance beyond the range used for fitting.

Because the maximum feasible budget differs across datasets, the exact held-out budgets are dataset-specific. The same protocol is applied to both the power-law and exponential models, so differences in extrapolation error reflect the behaviour of the curve families rather than differences in the evaluation setup.

4.5 Statistical evaluation

Every combination of dataset and labeled-node budget (n) is evaluated over (R = 20) runs. For Cora and PubMed, all

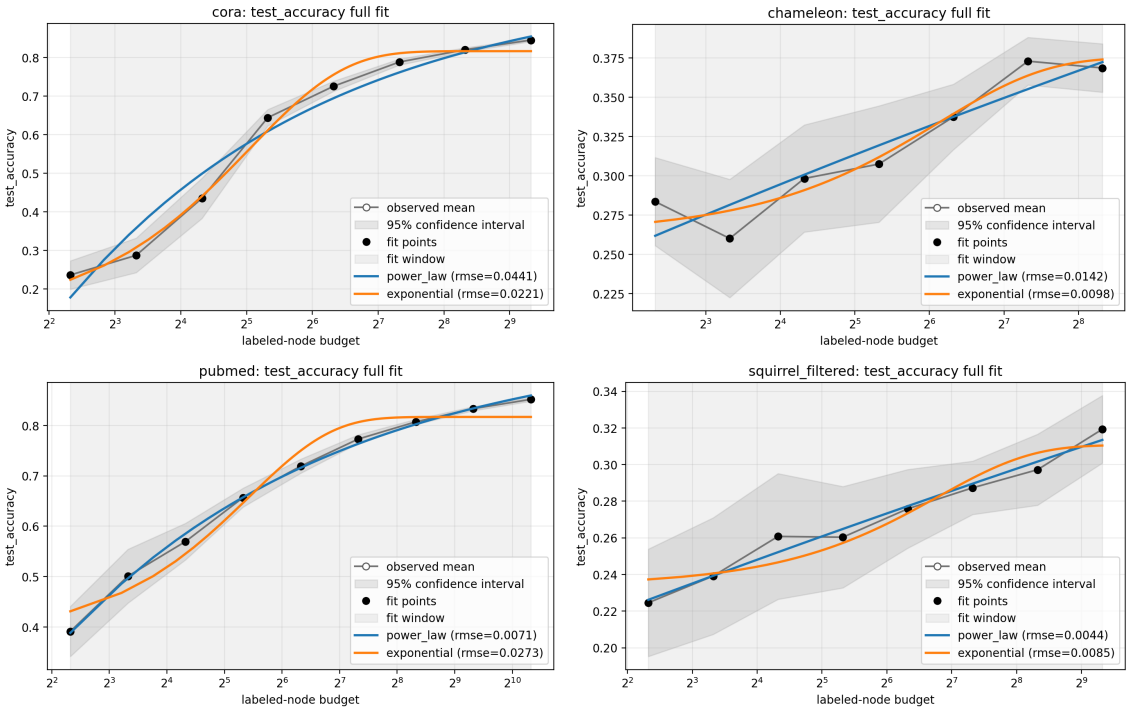

Figure 1: Power-law and exponential models fitted to the complete observed learning curves for Cora, Chameleon-filtered, PubMed, and Squirrel-filtered. Each panel shows the mean test accuracy across repeated runs as a function of the labeled-node budget, together with the fitted power-law and exponential curves.

runs use the standard dataset split and differ only in their random seed. For Chameleon-filtered and Squirrel-filtered, the 20 runs are distributed across the 10 predefined splits, with two seeds evaluated per split. Consequently, the variability observed on these two datasets captures both stochastic variation between runs and differences between the predefined data partitions. Results are summarized using the mean test accuracy over all runs, accompanied by 95% bootstrap confidence intervals.

5 Results

This section compares the descriptive-fit and extrapolation performance of the power-law and exponential learning-curve models. The analysis is organized by evaluation setting rather than by dataset. First, the models are evaluated according to how well they describe the complete observed learning curves. Second, they are evaluated according to their ability to extrapolate from smaller labeled-node budgets to larger held-out budgets. Results are reported for Cora, PubMed, Chameleon-filtered, and Squirrel-filtered using test accuracy as the performance metric.

5.1 Full-Curve Descriptive Fit

The first analysis evaluates how well each curve model describes the complete observed learning curve for each dataset. Both models are fitted using all feasible labeled-node budgets available for the corresponding dataset.

Figure 1 shows the full-curve fits for Cora, PubMed, Chameleon-filtered, and Squirrel-filtered. The observed points represent the mean test accuracy across repeated runs, with uncertainty shown by the error bars. Table 2 reports the RMSE of each fitted curve; lower values indicate a closer descriptive fit to the observed learning curve.

Table 2: Descriptive fit over the complete observed learning curves. Entries report RMSE; lower values indicate better fit.

	Curve model	
Dataset	Power-law	Exponential
Cora	0.044	**0.022**
PubMed	**0.007**	0.027
Chameleon-filtered	0.014	**0.010**
Squirrel-filtered	**0.004**	0.008

The descriptive-fit results do not show a universal preference for either curve family. The exponential model achieves lower RMSE on Cora and Chameleon-filtered, whereas the power-law model achieves lower RMSE on PubMed and Squirrel-filtered. This indicates that the best descriptive model depends on the dataset. The pattern is not explained solely by graph homophily, since the two homophilic citation datasets, Cora and PubMed, prefer different curve families, and the two heterophilic Wikipedia datasets, Chameleon-filtered and Squirrel-filtered, also prefer different curve families.

5.2 Extrapolation Results

The extrapolation analysis evaluates whether the fitted curve models can predict performance at larger labeled-node budgets that were not used during fitting. For each extrapolation experiment, the models are fitted using all observations up to a selected maximum budget. The remaining larger budgets are held out and used only for evaluation.

Figure 2 shows representative extrapolation experiments for Cora, PubMed, Chameleon-filtered, and Squirrel-filtered. The corresponding RMSE values across the evaluated fitting cutoffs are reported in Tables 3–6. Each table reports both the in-window fitting RMSE and the held-out extrapolation RMSE, allowing descriptive fit and extrapolation accuracy to be compared directly.

Table 3: Cora extrapolation results. PL denotes the power-law model and Exp. denotes the exponential model. Lower RMSE values indicate better performance.

	Fit		Ext.	
Through	PL	Exp.	PL	Exp.
40	0.054	**0.011**	**0.051**	0.162
80	0.054	**0.020**	**0.023**	0.051
160	0.051	**0.018**	**0.008**	0.048

Table 4: Chameleon extrapolation results. PL denotes the power-law model and Exp. denotes the exponential model. Entries report RMSE; lower values indicate better performance.

	Fit		Ext.	
Through	PL	Exp.	PL	Exp.
40	0.013	**0.012**	**0.037**	0.075
80	0.013	**0.010**	0.023	**0.022**

Table 5: Pubmed extrapolation results. PL denotes the power-law model and Exp. denotes the exponential model. Lower RMSE values indicate better performance.

	Fit		Ext.	
Through	PL	Exp.	PL	Exp.
40	**0.007**	0.012	**0.009**	0.134
80	**0.007**	0.014	**0.009**	0.100
160	**0.007**	0.018	**0.010**	0.076
320	**0.006**	0.019	**0.013**	0.060

The extrapolation results show a stronger preference for the power-law model than the full-curve descriptive-fit results. On Cora, the exponential model obtains lower fitting RMSE at each cutoff, but the power-law model extrapolates more accurately at all reported cutoffs. This illustrates that a closer fit to the observed lower-budget points does not necessarily imply better prediction at larger budgets.

A similar pattern appears on PubMed, where the power-law model achieves both lower fitting RMSE and substantially lower extrapolation RMSE across all evaluated cutoffs. On Squirrel-filtered, the power-law model also gives lower extrapolation RMSE at all reported cutoffs, even when the fitting RMSE values are very close. Chameleon-filtered is the least consistent case: the power-law model extrapolates better when fitting through 40, while the exponential model is slightly better when fitting through 80. Overall, the extrapolation analysis suggests that the power-law model is often more reliable for predicting larger-budget performance, but the preferred model can still depend on the dataset and the available fitting range.

Table 6: Squirrel extrapolation results. PL denotes the power-law model and Exp. denotes the exponential model. Lower RMSE values indicate better performance.

	Fit		Ext.	
Through	PL	Exp.	PL	Exp.
40	0.004	**0.003**	**0.028**	0.036
80	**0.004**	0.004	**0.019**	0.032
160	**0.004**	0.006	**0.013**	0.027

6 Discussion

Overall, the results show that descriptive fit and extrapolation accuracy should be treated as separate objectives. A model that closely describes the observed learning curve does not necessarily provide the most accurate predictions at larger held-out budgets. The results also show that neither the power-law model nor the exponential model is uniformly best across all datasets and evaluation settings.

6.1 Descriptive Fit of Power-Law and Exponential Models

The first research aim was to compare power-law and exponential models as descriptive models of GNN learning curves. The full-curve results show that neither model family provides the best descriptive fit on every dataset. The exponential model achieves lower RMSE on Cora and Chameleon-filtered, while the power-law model achieves lower RMSE on PubMed and Squirrel-filtered. Therefore, the descriptive-fit results do not support a universal preference for either model family.

This result is important because both candidate models represent plausible forms of diminishing returns. The exponential model represents faster saturation, while the power-law model allows slower continued improvement at larger budgets. The fact that both models are preferred on different datasets suggests that the observed learning-curve shape depends on dataset-specific factors rather than being determined solely by the choice of GNN architecture or by the general expectation that learning curves should follow a particular functional form.

The descriptive-fit results also indicate that graph homophily alone does not explain the preferred curve family. Cora and PubMed are both homophilic citation networks, but they favour different descriptive models. Similarly, Chameleon-filtered and Squirrel-filtered are both het-

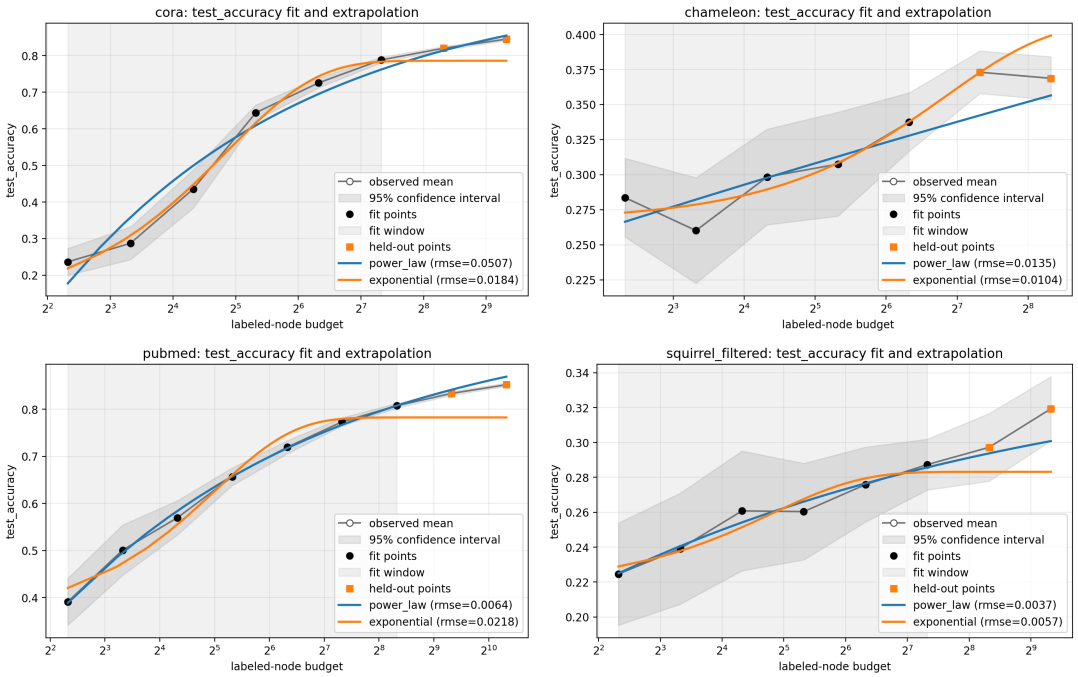

Figure 2: Representative extrapolation experiments for Cora, Chameleon-filtered, PubMed, and Squirrel-filtered. Each panel shows the observed lower-budget points used for fitting, the held-out larger-budget points used for extrapolation evaluation, and the fitted power-law and exponential curves.

erophilic Wikipedia datasets, but they also differ in their best descriptive fit. This suggests that other factors, such as dataset size, class distribution, split policy, graph structure, and the suitability of ChebNet for the dataset, may influence the shape of the observed learning curve.

6.2 Extrapolation from Limited Labeled-Node Budgets

The second research aim was to evaluate whether learning-curve models can predict performance at larger labeled-node budgets from smaller-budget observations. The extrapolation results show a clearer advantage for the power-law model than the descriptive-fit results. On Cora, the exponential model obtains lower fitting RMSE at each reported cutoff, but the power-law model achieves lower extrapolation RMSE after fitting through budgets 40, 80, and 160. This demonstrates that a closer fit to the observed lower-budget points does not necessarily imply better prediction of larger-budget performance.

The same distinction is visible on PubMed. In this case, the power-law model achieves both lower fitting RMSE and lower extrapolation RMSE across all evaluated cutoffs. The difference in extrapolation performance is substantial, with the exponential model producing much larger held-out errors despite being fitted to the same observed points. This suggests that, for PubMed, the exponential model saturates too quickly and underestimates continued improvements at larger labeled-node budgets.

Squirrel-filtered shows a similar extrapolation pattern. Although the fitting RMSE values of the two models are close, the power-law model achieves lower extrapolation RMSE at all reported cutoffs. This indicates that even when the two models describe the observed portion of the curve similarly well, their assumptions about long-term behaviour can lead to different predictions outside the fitting range.

Chameleon-filtered is the least consistent case. The power-law model extrapolates better after fitting through 40, while the exponential model is slightly better after fitting through 80. This suggests that the preferred extrapolation model can depend on the amount of the learning curve available during fitting. More generally, the results show that extrapolation performance should be evaluated directly using held-out larger budgets rather than inferred from descriptive fit alone.

6.3 Effect of the Available Budget Range

Extrapolation results should also be compared cautiously across fitting cutoffs, because later cutoffs provide more observations and leave fewer, closer held-out budgets. Thus, improved extrapolation at later cutoffs may reflect both better parameter estimation and an easier prediction task.

6.4 Learning-Curve Behaviour Across Datasets

The third research aim was to study learning-curve behaviour across multiple graph datasets. The results show that the preferred model depends on the dataset and that the pattern is not reducible to a simple homophilic-versus-heterophilic distinction. Cora and PubMed are both homophilic citation datasets, yet Cora favours the exponential model descriptively while PubMed favours the power-law model. Similarly, Chameleon-filtered and Squirrel-filtered are both heterophilic

Wikipedia datasets, but their extrapolation behaviour differs: Squirrel-filtered consistently favours the power-law model, whereas Chameleon-filtered shows a mixed result across fitting cutoffs.

These differences suggest that learning-curve shape is influenced by more than edge homophily alone. Dataset size may play a role, since PubMed contains substantially more training candidates and allows extrapolation to larger budgets than Cora. The number of classes, feature structure, and split policy may also affect how quickly performance saturates as labels are added. For the heterophilic datasets, the interaction between the graph structure and the ChebNet architecture is also relevant. Since ChebNet performs message passing based on graph connectivity, its learning behaviour may differ when neighboring nodes often belong to different classes.

The dataset-dependent results indicate that claims about GNN learning-curve shape should be made cautiously. The experiments do not show that one curve family universally describes all GNN learning curves. Instead, they suggest that power-law and exponential models capture different behaviours that may appear under different datasets and fitting ranges.

6.5 Do ChebNet Learning Curves Follow Power Laws?

The central research question of this study is whether GNN learning curves follow power laws. The results do not support a simple universal answer. Power-law models are not always the best descriptive fit to the complete observed learning curves: exponential models fit Cora and Chameleon-filtered more closely. However, the power-law model performs strongly in extrapolation, achieving better held-out predictions on Cora, PubMed, and Squirrel-filtered across the reported fitting cutoffs, with Chameleon-filtered showing a mixed result.

Therefore, the evidence supports a weaker but more useful conclusion: power-law behaviour is a plausible and often effective modelling assumption for ChebNet learning curves, especially when the goal is extrapolation to larger labeled-node budgets. However, power-law models should not be assumed to be universally correct, and descriptive fit alone is not sufficient to determine whether a learning curve follows a power law. The results instead support evaluating multiple candidate curve families and distinguishing between descriptive modelling and forecasting performance.

6.6 Limitations

A central limitation of this study is that it uses a single GNN architecture. The empirical learning curves are generated using ChebNet with a fixed training configuration. As a result, the findings describe the behaviour of this architecture under the chosen setup and should not be interpreted as a general conclusion about all GNNs. Other architectures, such as graph attention networks, message-passing neural networks, or models designed specifically for heterophilic graphs, may produce different learning-curve shapes.

A second limitation is the limited number of labeled-node budgets. Both candidate curve models contain three parameters, but the number of observed budget points is relatively small, especially in the extrapolation experiments. This makes curve fitting sensitive to the available budget range and can make it difficult to distinguish genuine long-term trends from short-range fitting behaviour.

The extrapolation range also differs across datasets because the maximum feasible labeled-node budget depends on the size of the training candidate pool. PubMed permits extrapolation to larger budgets than the other datasets, while Chameleon-filtered provides fewer held-out extrapolation points. This makes the extrapolation tasks not perfectly comparable across datasets.

Another limitation is that the main analysis is based on test accuracy. Accuracy is directly interpretable for node classification, but other metrics such as macro-F1 or cross-entropy may produce different curve shapes. Macro-F1 may be more informative when class imbalance is important, while cross-entropy captures confidence-sensitive prediction quality. Therefore, the conclusions should be understood as applying primarily to accuracy-based learning curves.

Finally, performance on the heterophilic datasets may reflect both dataset difficulty and architecture mismatch. Since ChebNet relies on graph-based message passing, it may be less well suited to graphs where connected nodes frequently have different labels. The learning curves for Chameleon-filtered and Squirrel-filtered may therefore describe the scaling behaviour of this particular architecture rather than the best achievable performance on these datasets.

7 Responsible Research

This study uses publicly available benchmark datasets and does not involve personal or sensitive data. The main responsible research concern is therefore reproducibility. Since the experiments rely on random labeled-node sampling, model initialization, and predefined data splits, each budget-level experiment is repeated across multiple seeds and the reported learning-curve values are based on aggregate test performance.

Reproducibility. For Cora and PubMed, the experiments use seeds $s \in \{0, 1, \ldots, 19\}$. For Chameleon-filtered and Squirrel-filtered, seeds $s \in \{0, 1\}$ are applied separately to each of the 10 predefined splits. All stochastic components of the experiment, including labeled-node selection and parameter initialization are controlled deterministically by the corresponding seed.

Exact reproduction of the numerical results may still depend on the software and hardware environment. Different versions of PyTorch, PyTorch Geometric, CUDA, or sparse-graph operations can lead to small differences in training behaviour and final accuracy values, even when the same seeds are used. Consequently, the reported RMSE values should be interpreted as empirical estimates obtained under a specific implementation environment.

Use of AI tools. AI tools were used to support the writing process, primarily for grammar, wording, and LaTeX formatting improvements. The research design, experiments, analysis, and interpretation of the results remain the author's own responsibility.

8 Conclusions and Future Work

This study compared power-law and exponential models for learning curves generated by ChebNet [2] on semi-supervised node-classification datasets. The results show that neither model family is universally preferable. Exponential models provide better descriptive fit on some datasets, but power-law models often give more accurate extrapolation to larger labeled-node budgets. This indicates that fitting the observed part of a learning curve and forecasting future performance should be treated as separate objectives.

Overall, the results suggest that power-law behaviour is a useful modelling assumption for some GNN learning curves, especially for extrapolation, but it should not be assumed to hold universally. Future work should evaluate additional GNN architectures, and more datasets. It would also be useful to test other curve families or non-parametric extrapolation methods that can better capture dataset-dependent learning behaviour.

References

[1] Steven Adriaensen, Herilalaina Rakotoarison, Samuel Müller, and Frank Hutter. Efficient bayesian learning curve extrapolation using prior-data fitted networks. *arXiv preprint arXiv:2310.20447*, 2023.

[2] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

[3] Tobias Domhan, Jost Tobias Springenberg, and Frank Hutter. Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, IJCAI'15, pages 3460–3468. AAAI Press, 2015.

[4] Matilde Gargiani, Aaron Klein, Stefan Falkner, and Frank Hutter. Probabilistic rollouts for learning curve extrapolation across hyperparameter settings. *arXiv preprint arXiv:1910.04522*, 2019.

[5] Joel Hestness, Sharan Narang, Newsha Ardalani, et al. Deep learning scaling is predictable, empirically. *arXiv preprint arXiv:1712.00409*, 2017.

[6] Jared Kaplan, Sam McCandlish, Tom Henighan, et al. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[7] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

[8] Jingzhe Liu et al. Towards neural scaling laws on graphs. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2025.

[9] Qian Ma, Haitao Mao, Jingzhe Liu, Zhehua Zhang, Chunlin Feng, Yu Song, Yihan Shao, and Yao Ma. Do neural scaling laws exist on graph self-supervised learning? *arXiv preprint arXiv:2408.11243*, 2024.

[10] Felix Mohr et al. Learning curves for decision making in supervised machine learning: A survey. *Artificial Intelligence Review*, 2024.

[11] Oleg Platonov, Denis Kuznedelev, Michael Diskin, Artem Babenko, and Liudmila Prokhorenkova. Characterizing graph datasets for node classification: Homophily–heterophily dichotomy and beyond. In *Advances in Neural Information Processing Systems*, 2023.

[12] Tom Viering and Marco Loog. The shape of learning curves: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(6):7799–7819, 2022.

[13] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. Scipy 1.0: Fundamental algorithms for scientific computing in python. *Nature Methods*, (3):261–272, 2020.

[14] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):4–24, 2020.

[15] Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 40–48, 2016.