Learning Curves for Analysis of Deep Networks

Derek Hoiem [1] **Tanmay Gupta** [2] **Zhizhong Li** [1] **Michal M. Shlapentokh-Rothman** [1]

Abstract

Learning curves model a classifier's test error as a function of the number of training samples. Prior works show that learning curves can be used to select model parameters and extrapolate performance. We investigate how to use learning curves to evaluate design choices, such as pretraining, architecture, and data augmentation. We propose a method to robustly estimate learning curves, abstract their parameters into error and data-reliance, and evaluate the effectiveness of different parameterizations. Our experiments exemplify use of learning curves for analysis and yield several interesting observations.

1. Introduction

The performance of a learning system depends strongly on the number of training samples, but standard evaluations use a fixed train/test split. While some works measure performance using subsets of training data, the lack of a systematic way to measure and report performance as a function of training size is a barrier to progress in machine learning research, particularly in areas like representation learning, data augmentation, and low-shot learning that specifically address the limited-data regime. What gets measured gets optimized, so we need better measures of learning ability to design better classifiers for the spectrum of data availability.

In this paper, we establish and demonstrate use of *learning curves* to improve evaluation of classifiers (see Fig. 1). Learning curves, which model error as a function of training set size, were introduced nearly thirty years ago by Cortes et al. (1993)) to accelerate model selection of deep networks, and recent works have demonstrated the predictability of performance improvements with more data (Hestness et al., 2017; Johnson & Nguyen, 2017; Kaplan et al., 2020; Rosenfeld et al., 2020) or more network parameters (Kaplan et al., 2020; Rosenfeld et al., 2020). But such studies aim to extrapolate rather than evaluate and have typically required large-scale experiments that are outside the computational budgets of many research groups.

Experimentally, we find that the extended power law $e_{test}(n) = \alpha + \eta n^{\gamma}$ yields a well-fitting learning curve, where e_{test} is test error and n is the number of training samples (or "training size"), but that the parameters $\{\alpha, \eta, \gamma\}$ are individually unstable under measurement perturbations. To facilitate curve comparisons, we abstract the curve into two key parameters, e_N and β_N, that have intuitive meanings and are more stable under measurement variance. e_N is the test error at $n = N$, and β_N is a measure of data-reliance, how much a classifier's error will change if the training size changes. Our experiments show that learning curves provide insights that cannot be obtained by single-point comparisons of performance. Our aim is to promote the use of learning curves as part of a standard learning system evaluation.

Our **key contributions**:

- Investigate how to best model, estimate, characterize, and display learning curves for use in classifier analysis
- Exemplify use of learning curves with analysis of impact of error and data-reliance due to network architecture, optimization, depth, width, fine-tuning, data augmentation, and pretraining.

Table 1 shows validated and rejected popular beliefs that single-point comparisons often overlook. In the following sections, we investigate how to model learning curves (Sec. 2), how to estimate them (Sec. 3), and what they can tell us about the impact of design decisions (Sec. 4), with discussion of limitations and future work in Sec. 5.

2. Modeling Learning Curves

The learning curve measures test error e_{test} as a function of the number of training samples n for a given classification model and learning method. Previous empirical observations suggest a functional form $e_{test}(n) = \alpha + \eta n^{\gamma}$, with bias-variance trade-off and generalization theories typically indicating $\gamma = -0.5$. We summarize what bias-variance trade-off and generalization theories (Sec. 2.1) and empiri-

[1]University of Illinois at Urbana-Champaign [2]PRIOR @ Allen Institute for AI. Correspondence to: Derek Hoiem <dhoiem@illinois.edu>.

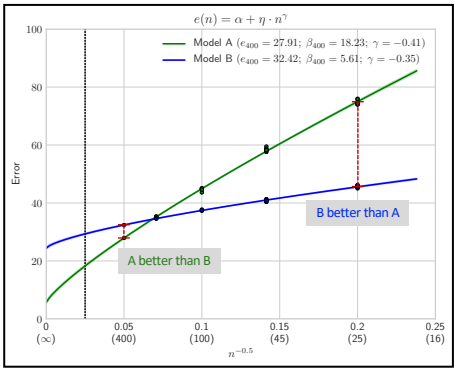
Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

Model	Error on full dataset ($n = 400$)
A	27.92
B	32.40

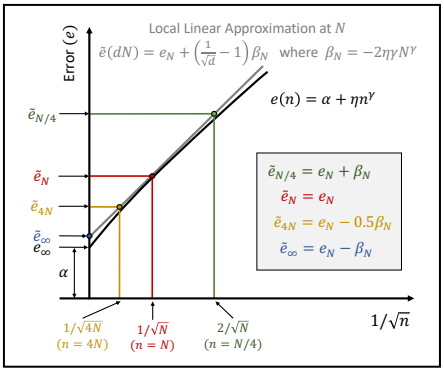
(a) Typical classifier evaluation and reporting

Model	e_{400}	β_{400}	γ
A	27.91	18.23	-0.41
B	32.42	5.61	-0.35

(b) Proposed evaluation that also characterizes change in performance with changing number of training samples as shown in (d)



(c) Model comparison via learning curves



(d) Characterization of a learning curve via error (e_N) and data-reliance (β_N)

Figure 1: **Evaluation with learning curves vs single-point comparison:** Comparing error of models trained on the full dataset, as shown in (a), is standard practice but provides an incomplete view of a classifier's performance. We propose a methodology to estimate and visualize learning curves that model a classifier's performance with varying amounts of training data (b,c). We also propose a succinct summary of a model's curve in terms of error and data-reliance (b,d).

Popular beliefs	Your guess	Supp-orted?	Exp. figures
Pre-training on similar domains nearly always helps compared to training from scratch.	☐	Y	5a, 5b, 6
Pre-training, even on similar domains, introduces bias that would harm performance with a large enough training set.	☐	U	6
Self-/un-supervised training performs better than supervised pre-training for small datasets.	☐	N	6
Fine-tuning the entire network (vs. just the classification layer) is only helpful if the training set is large.	☐	N	5a, 5b
Increasing network depth, when fine-tuning, harms performance for small training sets, due to an overly complex model.	☐	N	7a
Increasing network depth, when fine-tuning, is more helpful for larger training sets than smaller ones.	☐	N	7a
Increasing network depth, if the backbone is frozen, is more helpful for smaller training sets than larger ones.	☐	N	7d
Increasing depth or width improves more than ensembles of smaller networks with the same number of parameters.	☐	Y	7f
Data augmentation is roughly equivalent to using a m-times larger training set for some m.	☐	Y	8

Table 1: **Deep learning quiz!** We encourage our readers to judge each claim as T (true) or F (false) and then see if our experimental results support the claim (Yes/No/Unsure). In many cases, particularly regarding fine-tuning and network depth, the results surprised the authors. Our experiments show that learning curve analysis provides a systematic way to investigate these suppositions and others.

cal studies (Sec. 2.2) can tell us about learning curves, and describe our proposed abstraction in Sec. 2.3.

2.1. Bias-variance Trade-off and Generalization Theory

The bias-variance trade-off is an intuitive and theoretically sound way to think about generalization. The "bias" is error due to inability of the classifier to encode the optimal decision function, and the "variance" is error due to limited availability of training samples for parameter estimation. This is called a trade-off because a classifier with more parameters tends to have less bias but higher variance. Geman et al. (1992) decompose mean squared regression error into bias and variance and explore the implications for neural networks, leading to the conclusion that "identifying the right preconditions is *the* substantial problem in neural modeling". This conclusion foreshadows the importance of pretraining, though Geman et al. thought the preconditions must be built in rather than learned. Domingos (2000) extends the analysis to classification. Theoretically, the mean squared error (MSE) can be modeled as $e^2_{test}(n) = bias^2 + noise^2 + var(n)$, where "noise" is irreducible error due to non-unique mapping from inputs to labels, and variance can be modeled as $var(n) = \sigma^2/n$ for n training samples.

The ηn^{γ} term in $e_{test}(n)$ appears throughout machine learning generalization theory, usually with $\gamma = -0.5$. For example, the bounds based on hypothesis VC-dimension (Vapnik & Chervonenkis, 1971) and Rademacher Complexity (Gnecco & Sanguineti, 2008) are both $O(cn^{-0.5})$ where c depends on the complexity of the classification model. More recent work also follows this form, e.g. (Neyshabur et al., 2018; Bartlett et al., 2017; Arora et al., 2018; Bousquet & Elisseeff, 2002). One caveat is that the exponential term γ can deviate from -0.5 if the classifier parameters depend directly on n. For example, (Tsybakov, 2008) shows that setting the bandwidth of a kernel density estimator based on

n causes the dominant term to be bias, rather than variance, changing γ in the bound. In our experiments, all training and model parameters are fixed for each curve, except the learning schedule, but we note that the learning curve depends on optimization methods and hyperparameters as well as the classifier model.

2.2. Empirical Studies

Some recent empirical studies (e.g. Sun et al. (2017)) claim a log-linear relationship between error and training size, but this holds only when asymptotic error is zero. Hestness et al. (2017) model error as $e_{test}(n) = \alpha + \eta n^{\gamma}$ but often find γ much smaller in magnitude than -0.5 and suggest that poor fits indicate need for better hyperparameter tuning. This empirically supports that data sensitivity depends both on the classification model and on the efficacy of the optimization algorithm and parameters. Johnson & Nguyen (2017) also find a better fit with this extended power law model than by restricting $\gamma = -0.5$ or $\alpha = 0$.

In the language domain, learning curves are used in a fascinating study by Kaplan et al. (2020). For natural language transformers, they show that a power law relationship between logistic loss, model size, compute time, and dataset size is maintained if, and only if, each is increased in tandem. We draw some similar conclusions to their study, such as that increasing model size tends to improve performance *especially* for small training sets (which surprised us). However, the studies are largely complementary, as we study convolutional nets in computer vision, classification error instead of logistic loss, and a broader range of design choices such as data augmentation, pretraining source, architecture, and optimization. Also related, Rosenfeld et al. (2020) model error as a function of both training size and number of model parameters with a five-parameter function that accounts for training size, model parameter size, and chance performance. A key difference in our work is that we focus on how to best draw insights about design choices from learning curves, rather than on extrapolation. As such, we propose methods to estimate learning curves and their variance from a relatively small number of trained models.

2.3. Proposed Characterization of Learning Curves for Evaluation

Our experiments in Sec. 4.1 show that the learning curve model $e(n) = \alpha + \eta n^{\gamma}$ results in excellent leave-one-size-out RMS error and extrapolation. However, α, η, and γ cannot be meaningfully compared across curves because the parameters have high covariance with small data perturbations, and comparing η values is not meaningful unless γ is fixed and vice-versa. This would prevent tabular comparisons and makes it harder to draw quantitative conclusions.

To overcome this problem, we propose to report error and sensitivity to training size in a way that can be derived from various learning curve models and is insensitive to data perturbations. The curve is characterized by error $e_N = \alpha + \eta N^{\gamma}$ and data-reliance β_N, and we typically choose N as the full dataset size. Noting that most learning curves are locally well approximated by a model linear in $n^{-0.5}$, we compute data-reliance as $\beta_N = N^{-0.5} \frac{\partial e}{\partial n^{-0.5}}\big|_{n=N} = -2\eta\gamma N^{\gamma}$. When the error is plotted against $n^{-0.5}$, β_N is the slope at N scaled by $N^{-0.5}$, with the scaling chosen to make the practical implications of β_N more intuitive. This yields a simple predictor for error when changing training size by a factor of d:

$$\widetilde{e}(d \cdot N) = e_N + \left(\frac{1}{\sqrt{d}} - 1\right)\beta_N. \tag{1}$$

This is a first order Taylor expansion of $e(n)$ around $n = N$ with respect to $n^{-0.5}$. By this linearized estimate, asymptotic error is $e_N - \beta_N$, a 4-fold increase in data (*e.g.* $400 \rightarrow 1600$) reduces error by $0.5\beta_N$, and using only one quarter of the dataset (*e.g.* $400 \rightarrow 100$) increases the error by β_N. For two models with similar e_N, the one with a larger β_N would outperform with more data but underperform with less. (e_N, β_N, γ) is a complete re-parameterization of the extended power law, with $\gamma + 0.5$ indicating the curvature in $n^{-0.5}$ scale. See Fig. 1d for illustration.

3. Estimating Learning Curves

We now describe the method for estimating the learning curve from error measurements with confidence bounds on the estimate. Let e_{ij} denote the random variable corresponding to test error when the model is trained on the j^{th} fold of n_i samples (either per class or in total). We assume $\{e_{ij}\}_{j=1}^{F_i}$ are i.i.d according to $\mathcal{N}(\mu_i, \sigma_i^2)$. We want to estimate learning curve parameters α (asymptotic error), η, and γ, such that $e_{ij} = \alpha + \eta n_i^{\gamma} + \epsilon_{ij}$ where $\epsilon_{ij} \sim \mathcal{N}(0, \sigma_i^2)$ and $\mu_{ij} = \mathbb{E}[e_{ij}] = \mu_i$. Sections 3.1 and 3.2 describe how to estimate mean and variance of α and η for a given γ, and Sec. 3.3 describes our approach for estimating γ.

3.1. Weighted Least Squares Formulation

We estimate learning curve parameters $\{\alpha, \eta\}$ by optimizing a weighted least squares objective:

$$\mathcal{G}(\gamma) = \min_{\alpha,\eta} \sum_{i=1}^{S} \sum_{j=1}^{F_i} w_{ij} \left(e_{ij} - \alpha - \eta n^{\gamma}\right)^2 \tag{2}$$

where $w_{ij} = 1/(F_i\sigma_i^2)$. F_i is the number of models trained with data size n_i and is used to normalize the weight so that the total weight for observations from each training size does not depend on F_i. The factor of σ_i^2 accounts for the variance of ϵ_{ij}. Assuming constant σ_i^2 and removing the F_i factor would yield unweighted least squares.

The variance of the estimate of σ_i^2 from F_i samples is $2\sigma_i^4/F_i$, which can lead to over- or under-weighting data for particular i if F_i is small. Recall that each sample e_{ij} requires training an entire model, so F_i is always small in our experiments. We would expect the variance to have the form $\sigma_i^2 = \sigma_0^2 + \hat{\sigma}^2/n_i$, where σ_0^2 is the variance due to random initialization and optimization and $\hat{\sigma}^2/n_i$ is the variance due to randomness in selecting n_i samples. We validated this variance model by averaging over the variance estimates for many different network models on the CIFAR-100 (Krizhevsky, 2012) dataset. We use $\sigma_0^2 = 0.02$ in all experiments and then use a least squares fit to estimate a single $\hat{\sigma}^2$ parameter from all samples $\mathbf{e}$ in a given learning curve. This attention to w_{ij} may seem fussy, but without such care we find that the learning curve often fails to account sufficiently for all the data in some cases.

3.2. Solving for Learning Curve Mean and Variance

Concatenating errors across dataset sizes (indexed by i) and folds results in an error vector $\boldsymbol{e}$ of dimension $D = \sum_{i=1}^{S} F_i$. For each $d \in \{1, \cdots, D\}$, $\boldsymbol{e}[d]$ is an observation of error at dataset size n_{i_d} that follows $\mathcal{N}(\mu_{i_d}, \sigma_{i_d}^2)$ with i_d mapping d to the corresponding i.

The weighted least squares problem can be formulated as solving a system of linear equations denoted by $W^{1/2}\boldsymbol{e} = W^{1/2}A\boldsymbol{\theta}$, where $W \in \mathbb{R}^{D\times D}$ is a diagonal matrix of weights $W_{dd} = w_d$, $A \in \mathbb{R}^{D\times 2}$ is a matrix with $A[d,:] = [1 \;\; n_d^\gamma]$, and $\boldsymbol{\theta} = [\alpha \;\; \eta]^T$ are the parameters of the learning curve, treating γ as fixed for now. The estimator for the learning curve is then given by $\hat{\boldsymbol{\theta}} = (W^{1/2}A)^+ W^{1/2}\boldsymbol{e} = M\boldsymbol{e}$, where $M \in \mathbb{R}^{2\times D}$ and $^+$ is pseudo-inverse operator.

The covariance of the estimator is given by $\Sigma_{\hat{\boldsymbol{\theta}}} = M\Sigma_{\boldsymbol{e}}M^T$, where $\Sigma_{\hat{\boldsymbol{\theta}}} \in \mathbb{R}^{2\times 2}$ and $\Sigma_{\boldsymbol{e}} \in \mathbb{R}^{D\times D}$ is the diagonal covariance of $\boldsymbol{e}$ with $\Sigma_{\boldsymbol{e}}[d,d] = \sigma_{i_d}^2$. We compute our empirical estimate of σ_i^2 as described in Sec. 3.1.

Since the estimated curve is given by $\hat{e}(n) = [1 \;\; n^\gamma]\,\hat{\boldsymbol{\theta}}$, the 95% bounds at any n can be computed as $\hat{e}(n) \pm 1.96 \times \hat{\sigma}(n)$ with

$$\hat{\sigma}^2(n) = \begin{bmatrix} 1 & n^\gamma \end{bmatrix} \Sigma_{\hat{\boldsymbol{\theta}}} \begin{bmatrix} 1 \\ n^\gamma \end{bmatrix} \tag{3}$$

For a given γ, these confidence bounds reflect the variance in the estimated curve due to variance in error measurements.

3.3. Estimating γ

We search for γ that minimizes the weighted least squares objective with an L1-prior that slightly encourages values close to 0.5. Specifically, we solve

$$\min_{\gamma\in(-1,0)} \mathcal{G}(\gamma) + \lambda|\gamma + 0.5| \tag{4}$$

by searching over $\gamma \in \{-0.99, ..., -0.01\}$ with $\lambda = 5$ (with error on 100 point scale) for our experiments.

4. Experiments

We validate our choice of learning curve model and estimation method in Sec. 4.1 and use the learning curves to explore impact of design decisions on error and data-reliance in Sec. 4.2.

Setup: Each learning curve is fit to test errors measured after training the classifier on various subsets of the training data. When less than the full training set is used, we train multiple classifiers using different partitions of data (e.g. four classifiers on four quarters of the data). We avoid use of benchmark test sets, since the curves are used for analysis and ablation. Instead, training data are split 80/20, with 80% of data used for training and validation and remaining 20% for testing. For each curve, all models are trained using an initial learning rate that is selected using one subset of training data, and the learning rate schedule is set for each n based on validation. Unless otherwise noted, models are pretrained on ImageNet. Other hyperparameters are fixed for all experiments. Unless otherwise noted, we use the Ranger optimizer (Wright, 2019), which combines Rectified Adam (Liu et al., 2020), Look Ahead (Zhang et al., 2019), and Gradient Centralization (Yong et al., 2020), as preliminary experiments showed its effectiveness. For "linear", we train only the final classification layer with the other weights frozen to initialized values. All weights are trained when "fine-tuning". Tests are on Cifar100 (Krizhevsky, 2012), Cifar10, Places365 (Zhou et al., 2017), or Caltech-101 (L. Fei-Fei; Fergus, 2006). See supplemental materials (Appendix A) for more implementation details.

4.1. Evaluation of Learning Curves Model and Fitting

We validate our learning curve model using leave-one-size-out prediction error, e.g. predicting empirical mean performance with 400 samples per class based on observing error from models trained on 25, 50, 100, and 200 samples.

Weighting Schemes. In the Fig. 2 table, learning curve prediction error is averaged for 16 Cifar100 classifiers with varying design decisions. We compare three weighting schemes (w's in Eq. 2): $w_{ij} = 1$ is unweighted; $w_{ij} = 1/\sigma_i^2$ is weighted by estimated size-dependent standard deviation; $w_{ij} = 1/(F_i\sigma_i^2)$ makes the total weight for a given dataset size invariant to the number of folds. On average our proposed weighting performs best with high significance compared to unweighted. The p-value is paired t-test of difference of means calculated across all dataset sizes.

Model Choice. We consider other parameterizations that are special cases of $e(n) = \alpha + \eta n^\gamma + \delta n^{2\gamma}$. Setting $\delta = 0$ (top row of table) yields the model described in Sec. 2.3

Params	Weights	R^2	RMSE 25	50	100	200	400	avg	p-value
α, η, γ	$\frac{1}{\sigma_i^2 F_i}$	0.998	2.40	0.86	**0.54**	0.57	**0.85**	**1.04**	-
	$\frac{1}{\sigma_i^2}$	0.999	**2.38**	0.83	0.69	0.54	1.08	1.10	0.06
	1	0.998	2.66	0.86	0.79	**0.50**	1.26	1.21	0.008
α, η	$\frac{1}{\sigma_i^2 F_i}$	0.988	3.41	1.09	0.69	0.72	1.21	1.42	<0.001
α, η, δ	$\frac{1}{\sigma_i^2 F_i}$	0.999	2.89	**0.74**	0.68	0.56	0.94	1.16	0.05
$\alpha, \eta, \delta, \gamma$	$\frac{1}{\sigma_i^2 F_i}$	0.999	3.46	**0.74**	0.70	0.59	1.00	1.30	0.02

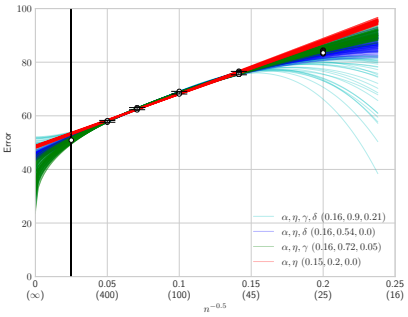


Figure 2: **Learning curve model and weights validation.** See Sec. 4.1 for explanation.

and used in our experiments. The table in Fig. 2 shows that our model outperforms the others, in most cases with high significance, and achieves a very good fit with R^2 of 0.998.

Model Stability. Each data point requires training a classifier, so we want to verify whether the curves can be estimated from few points. We test stability and sample requirements by repeatedly fitting curves to four resampled data points for a model (Resnet-18, no pretraining, fine-tuned, tested on Places365). Based on estimates of mean and standard deviation, one point each at $n = \{50, 100, 200, 400\}$ is sampled and used to fit a curve, repeated 100 times. Parentheses in legend show standard deviation of estimates of e_N, β_N, and γ. Our preferred model extrapolates best to $n = 1600$ and $n = 25$ (plotted as white circles) while retaining stable estimates of of e_N and β_N, but predicted asymptotic error α varies widely. Supplemental material (Appendix D) shows similar estimates of e_N and β_N by fixing $\gamma = -0.5$ and fitting only α and η on the three largest sizes (typically $n = \{100, 200, 400\}$), indicating that a lightweight approach of training a few models can yield similar conclusions.

4.2. Learning Curve Comparisons

We explore a broad range of design decisions, aiming to exemplify the use of learning curves and demonstrate that such analysis leads interesting observations. Most of these design decisions warrant more complete investigation in separate papers to draw more general conclusions.

Figures: We plot the fitted learning curves and confidence bounds, with observed test errors as circles. The legend displays γ, **error** e_N, and **data reliance** β_N with $N = 400$ for Cifar100 and Places365 and $N = 4000$ for Cifar10. The x-axis is in scale $n^{-0.5}$ (n in parentheses is the number of samples per class), but γ is fit for each curve. A vertical bar indicates $n = 1600$, which we consider the limit of accurate extrapolation from curves fit to $n \leq 400$ samples. All points are used for fitting, except in Fig. 5b $n = 1600$ is held out to test extrapolation. *Best viewed in color.*

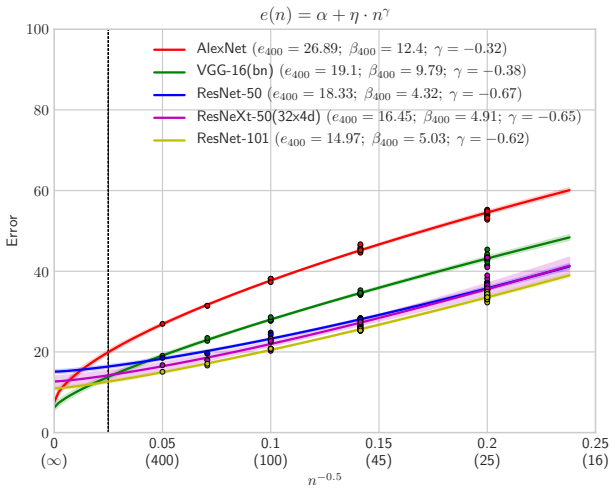


Figure 3: **Architecture** (Cifar100 w/ finetuning)

Network architecture: Advances in CNN architectures have reduced number of parameters while also reducing error over the range of training sizes. On Cifar100, AlexNet has 61M parameters; VGG-16, 138M; ResNet-50, 26M; ResNeXt-50, 25M; and ResNet-101, 45M. The landmark architecture papers do not examine effect of training data size, so it is interesting to see in Fig. 3 that each major advance through ResNet reduces both data reliance and e_{400}. ResNeXt appears to slightly reduce e_{400} without change to data reliance.

Optimization method: In Fig. 4, we show results on Cifar10 when training ResNet-18 using four different optimization methods: Ranger (Wright, 2019), Adam (Kingma & Ba, 2015), stochastic gradient descent (SGD) w/ momentum, and SGD w/o momentum. With pretraining, all methods perform similarly, but when training from scratch Ranger outperforms with lower e_{4000} and β_{4000}. SGD without momentum performs the worst and is least consistent across folds. Optimization papers routinely show error as a function of training iterations, but not the relationship to training size, so it is interesting to see empirically how the optimizer matters most with small data sizes and/or no pretraining, likely because better optimizers reduce variance of parameter estimation.

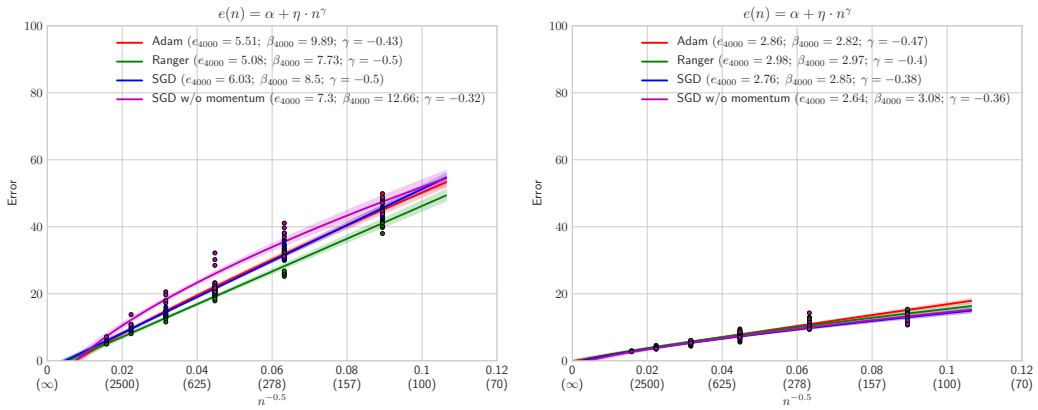


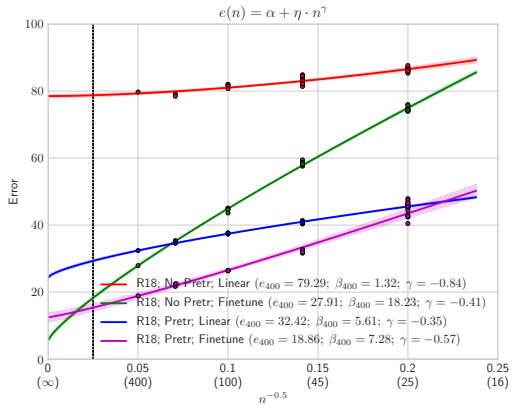
Figure 4: **Optimization** on Cifar10 with ResNet-18. Without pretraining on left; with ImageNet pretraining on right.

Pretraining and fine-tuning: Some prior works examine the effectiveness of pretraining. Kornblith et al. (2019) show that fine-tuned pretrained models outperform randomly initialized models across many architectures and datasets, but the gap is often small and narrows as training size grows. For object detection, He et al. (2019) find that, with long learning schedules, randomly initialized networks approach the performance of networks pretrained for classification, even on smaller training sizes, and Zoph et al. (2020) show that pretraining can sometimes harm performance when strong data augmentation is used.

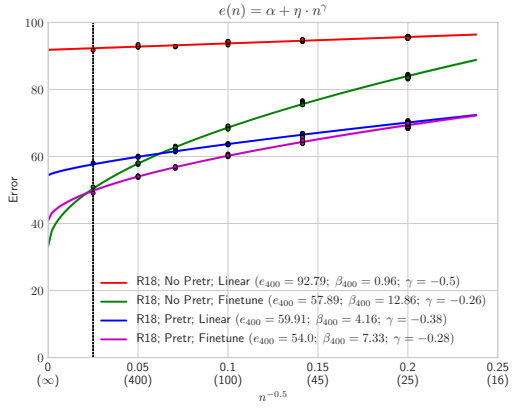
Compared to prior works, our use of learning curve models enables extrapolation beyond available data and numerical comparison of data reliance. In Fig. 5 we see that, without fine-tuning ("linear"), pretraining leads to a huge improvement in e_{400} for all training sizes. When fine-tuning, the pretraining greatly reduces data-reliance β_{400} and also reduces e_{400}, providing strong advantages with smaller training sizes that may disappear with enough data.

Pretraining data sources: In Fig. 6, we test on Cifar100 and Caltech-101 to compare different pretrained models: randomly initialized, supervised on ImageNet or Places365 (Zhou et al., 2017), and self-supervised on ImageNet (MOCO by He et al. (2020)). The strongest impact is on data-reliance, which leads to consistent orderings of models across training sizes for both datasets. Supervised ImageNet pretraining has lowest e_N and β_N, then self-supervised MOCO, then supervised Places365, with random initialization trailing far behind all pretrained models. The original papers excluded either analysis of training size (MOCO) or fine-tuning (Places365), so ours is the first analysis on image benchmarks to compare finetuned models from different pretraining sources as a function of training size. Newly proposed methods for representation learning would benefit from further learning curve analysis.

Network depth, width, and ensembles: The classical view is that smaller datasets need simpler models to avoid



(a) Transfer: ImageNet to Cifar100



(b) Transfer: ImageNet to Places365

Figure 5: **Pretraining and fine-tuning** with ResNet-18.

overfitting. In Figs. 7a, 7d, we show that, not only do deeper networks have better potential at higher data sizes, their data reliance does not increase (nearly parallel and drops a little for fine-tuning), making deeper networks perfectly suitable for smaller datasets. For linear classifiers (Fig. 7d), the deeper networks provide better features, leading to consistent drop in e_{400}. The small jump in data reliance between

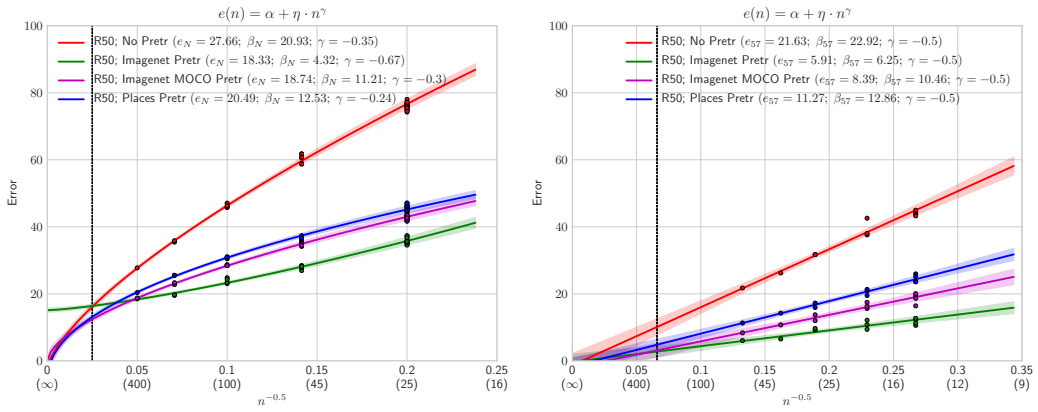


Figure 6: **Pretraining sources**: test on Cifar100 (left) and Caltech-101 (right).

Resnet-34 and Resnet-50 may be due to the increased last layer input size from 512 to 2048 nodes. When increasing width, the fine-tuned networks (Fig. 7b) have reduced e_{400} without much change to data-reliance. With linear classifiers (Fig. 7e), increasing the width leads to little change or even increase in e_{400} with slight decrease in data-reliance.

An alternative to using a deeper or wider network is forming ensembles. Figure 7c shows that, while an ensemble of six ResNet-18's (each 11.7M parameters) improves over a single model, it has higher e_{400} and data-reliance than ResNet-101 (44.5M), Wide-ResNet-50 (68.9M), and Wide-ResNet-101 (126.9M). Three ResNet-50's (each 25.6M) underperforms Wide-ResNet-50 on e_{400} but outperforms for small amounts of data due to lower data reliance. Fig. 7f tabulates data reliance and error to simplify comparison.

Rosenfeld et al. (2020) show that error can be modeled as a function of either training size, model size, or both. Modeling both jointly can provide additional capabilities such as selecting model size based on data size, but requires many more experiments to fit the curve. Our experiments show more clearly the effect on data reliance due to different ways of changing model size.

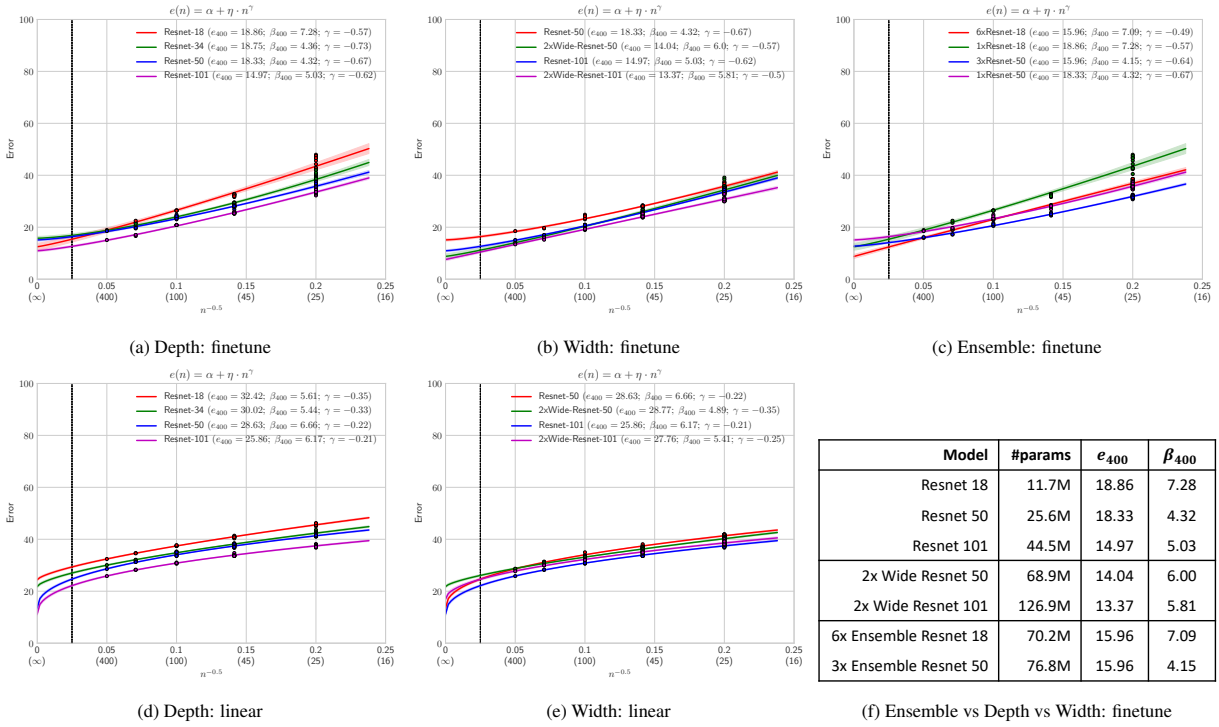
Data Augmentation: We are not aware of previous studies on interaction between data augmentation and training size. For example, a large survey (Shorten & Khoshgoftaar, 2019) compares different augmentation methods only on full training size. One may expect that data augmentation acts as a regularizer with reduced effect for large training sizes, or even possibly negative effect due to introducing bias. However, Fig. 8 shows that data augmentation on Places365 reduces error for all training sizes with little or no change to data-reliance when fine-tuning. $e(n)$ with augmentation roughly equals $e(1.8n)$ without it, supporting the view that augmentation acts as a multiplier on the value of an example. For the linear classifier, data augmentation has little apparent effect due to low data-reliance, but the results are still consistent with this multiplier.

5. Discussion

Evaluation methodology is the foundation of research, impacting how we choose problems and rank solutions. Large train and test sets now serve as the fuel and crucible to refine machine learning methods. We now discuss the need for learning curves, limitations of our experiments, and directions for future work.

Case for learning curve models. Compared to individual runs, learning curves often provide more accurate or confident conclusions, which is crucial, as any misleading/partial conclusions can block progress. For example, in Fig. 5b, comparison at $N = 1600$ may indicate that pretraining has little impact on error when fine-tuning, but the curves show large differences in data-reliance leading to a large gap when less data is available. In Fig. 6 (left), comparison at $N = 400$ may indicate that MOCO and supervised ImageNet pretraining are equally effective, but the curves show that the supervised ImageNet pretrained classifier has lower data-reliance ($\beta = 4.32$ vs. $\beta = 11.21$) and thus outperforms with less data. Generally, individual runs cannot reveal the sample efficiency of a learner. One curve may dominate another because it has less bias (leading to uniformly lower error, as with "linear" classifiers of increasing depth in Fig. 7) or less variance (leading to different curve slopes, as in different pretrained models in Fig. 6), and distinguishing helps understand and improve.

Compared to a piecewise affine fit (i.e., plotting a line through observed error points), our modeling and characterization in terms of error@N and data-reliance has important advantages. First, our two-parameter characterization can be tabulated (Fig. 7f) facilitating comparison within and across papers. We can learn from object detection research, where the practice of reporting PR/ROC curves gave way to reporting AP, making it easier to average and compare within and across works when working with multiple datasets. Second, a piecewise affine fit is less stable under measurement error, leading to significantly poorer extrapolations. Finally,



(a) Depth: finetune (b) Width: finetune (c) Ensemble: finetune

(d) Depth: linear (e) Width: linear

Model	#params	e_{400}	β_{400}
Resnet 18	11.7M	18.86	7.28
Resnet 50	25.6M	18.33	4.32
Resnet 101	44.5M	14.97	5.03
2x Wide Resnet 50	68.9M	14.04	6.00
2x Wide Resnet 101	126.9M	13.37	5.81
6x Ensemble Resnet 18	70.2M	15.96	7.09
3x Ensemble Resnet 50	76.8M	15.96	4.15

(f) Ensemble vs Depth vs Width: finetune

Figure 7: **Depth, width, and ensembles** on Cifar100. The table (f) compactly compares several curves.

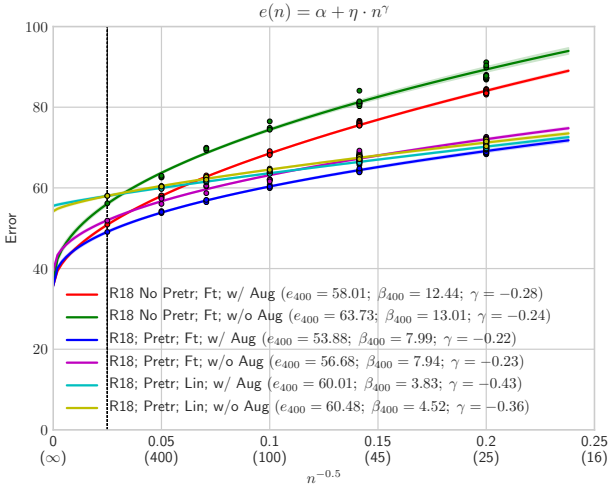


Figure 8: **Data augmentation** on Places365.

the parametric model helps to identify poor hyperparameter selection, indicated by poor model fit or extreme values of γ.

Cause and impact of γ: We speculate that γ is largely determined by hyperparameters and optimization rather than model design. This presents an opportunity to identify poor training regimes and improve them. Intuitively, one would expect that more negative γ values are better (i.e. $\gamma = -1$ preferable to $\gamma = -0.5$), since the error is $O(n^{\gamma})$, but we find the high-magnitude γ tends to come with high asymptotic error, indicating that the efficiency comes at cost of over-commitment to initial conditions. We speculate (but with some disagreement among authors) that $\gamma \approx -0.5$ is an indication of a well-trained curve and will generally outperform curves with higher or lower γ, given the same classification model.

Small training sets: Error is bounded and classifier performance with small training sets may be modeled as transitioning from random guess to informed prediction, as shown by Rosenfeld et al. (2020). For simplicity, we do not model performance with very small training size, but studying the small data regime could be interesting, particularly to determine whether design decisions have an impact at the small size that is not apparent at larger sizes.

Losses and Prediction types: We analyze multiclass classification error, but the same analysis could likely be extended to other prediction types. For example, Kaplan et al. (2020) analyze learning manifolds of cross-entropy loss of language model transformers. Learning curve models of cross-entropy loss could also be used for problems like object detection or semantic segmentation that typically use more complex aggregate evaluations.

More design parameters and interactions: The interaction between data scale, model scale, and performance is well-explored by Kaplan et al. (2020) and Rosenfeld et al. (2020), but it could also be interesting to explore interactions, e.g. between class of architecture (e.g. VGG, ResNet, EfficientNet (Tan & Le, 2019)) and some design parameters,

to see the impact of ideas such as skip-connections, residual layers and bottlenecks. More extensive evaluation of data augmentation, representation learning, optimization, and regularization would also be interesting.

Unbalanced class distributions: In most of our experiments, we use equal number of samples per class. Our experiments on Caltech-101 in Fig. 6 (right) and Pets and Sun397 (Fig. 10 in supplemental) use the original imbalanced distributions, demonstrating that the same learning curve model applies, but further experiments are needed to examine the impact of class imbalance.

Limits to extrapolation: Our experiments indicate good extrapolation up to 4x the observed training size. However, we caution against drawing conclusions about asymptotic error, since estimation of α is highly sensitive to data perturbations. $e_N - \beta_N$ provides a more stable indicator of large-sample performance.

The **supplemental material** contains implementation details (Appendix A); a user guide to fitting, displaying, and using learning curves (Appendix B); experiments on additional datasets (Appendix C); and a table of learning curve parameters for all experiments, also comparing e_N and β_N produced by two learning curve models (Appendix D). Code is currently available at prior.allenai.org/projects/lcurve.

6. Conclusion

Performance depends strongly on training size, so size-variant analysis more fully shows the impact of innovations. A reluctant researcher may protest that such analysis is too complicated, too computationally expensive, or requires too much space to display. We show that learning curve models can be easily fit (Sec. 3) from a few trials (Fig. 2) and compactly summarized (Fig. 7f), leaving our evasive experimenter no excuse. Our experiments serve as examples that learning curve analysis can yield interesting observations. Learning curves can further inform training methodology, continual learning, and representation learning, among other problems, providing a better understanding of contributions and ultimately leading to faster progress in machine learning.

Acknowledgements: This research is supported in part by ONR MURI Award N000141612007. We thank reviewers for prompting further discussion.

References

Arora, S., Ge, R., Neyshabur, B., and Zhang, Y. Stronger generalization bounds for deep nets via a compression approach. In *International Conference on Machine Learning (ICML)*, 2018.

Bartlett, P. L., Foster, D. J., and Telgarsky, M. Spectrally-normalized margin bounds for neural networks. In *Conference on Neural Information Processing Systems (NIPS)*, 2017.

Bousquet, O. and Elisseeff, A. Stability and generalization. *Journal of Machine Learning Research*, 2:499–526, 2002.

Cortes, C., Jackel, L. D., Solla, S. A., Vapnik, V., and Denker, J. S. Learning curves: Asymptotic values and rate of convergence. In *Advances in Neural Information Processing Systems (NIPS)*, 1993.

Domingos, P. A unified bias-variance decomposition for zero-one and squared loss. In *National Conference on Artificial Intelligence and Conference on Innovative Applications of Artificial Intelligence*, 2000.

Falcon, W. Pytorch lightning. *GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning*, 3, 2019.

Geman, S., Bienenstock, E., and Doursat, R. Neural networks and the bias/variance dilemma. *Neural Computation*, 4(1):1–58, January 1992.

Gnecco, G. and Sanguineti, M. Approximation error bounds via rademacher's complexity. *Applied Mathematical Sciences*, 2(4), 2008.

He, K., Zhang, X., Ren, S., and Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *International Conference on Computer Vision (ICCV)*, 2015.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

He, K., Girshick, R., and Dollar, P. Rethinking imagenet pre-training. In *International Conference on Computer Vision (ICCV)*, 2019.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum contrast for unsupervised visual representation learning. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Hestness, J., Narang, S., Ardalani, N., Diamos, G. F., Jun, H., Kianinejad, H., Patwary, M. M. A., Yang, Y., and Zhou, Y. Deep learning scaling is predictable, empirically. *CoRR*, 2017.

Johnson, M. and Nguyen, D. Q. How much data is enough? Predicting how accuracy varies with training data size. Technical report, 2017.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. Scaling laws for neural language models. *ArXiv preprint*, 2020.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

Kornblith, S., Shlens, J., and Le, Q. V. Do better imagenet models transfer better? In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Krizhevsky, A. Learning multiple layers of features from tiny images. *University of Toronto*, 05 2012.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Conference on Neural Information Processing Systems (NIPS)*, 2012.

L. Fei-Fei; Fergus, R. P. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis Machine Intelligence (PAMI)*, 28:594–611, April 2006.

Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., and Han, J. On the variance of the adaptive learning rate and beyond. In *International Conference on Learning Representations (ICLR)*, 2020.

Neyshabur, B., Bhojanapalli, S., and Srebro, N. A PAC-bayesian approach to spectrally-normalized margin bounds for neural networks. In *International Conference on Learning Representations (ICLR)*, 2018.

Rosenfeld, J. S., Rosenfeld, A., Belinkov, Y., and Shavit, N. A constructive prediction of the generalization error across scales. In *International Conference on Learning Representations (ICLR)*, 2020.

Shorten, C. and Khoshgoftaar, T. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6, 07 2019. doi: 10.1186/s40537-019-0197-0.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2015.

Sun, C., Shrivastava, A., Singh, S., and Gupta, A. Revisiting unreasonable effectiveness of data in deep learning era. In *International Conference on Computer Vision (ICCV)*, 2017.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. Going deeper with convolutions. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

Tan, M. and Le, Q. V. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning (ICML)*, 2019.

Tsybakov, A. B. *Introduction to Nonparametric Estimation*. Springer Publishing Company, Incorporated, 1st edition, 2008. ISBN 0387790519.

Vapnik, V. N. and Chervonenkis, A. Y. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–280, 1971. doi: 10.1137/1116025.

Wright, L. New deep learning optimizer, ranger: Synergistic combination of radam + lookahead for the best of both. *Github https://github.com/lessw2020/Ranger-Deep-Learning-Optimizer*, 08 2019.

Xie, S., Girshick, R., Dollár, P., Tu, Z., and He, K. Aggregated residual transformations for deep neural networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Yong, H., Huang, J., Hua, X., and Zhang, L. Gradient centralization: A new optimization technique for deep neural networks. In *European Conference on Computer Vision (ECCV)*, 2020.

Zagoruyko, S. and Komodakis, N. Wide residual networks. In *British Machine Vision Conference (BMVC)*, 2016.

Zhai, X., Puigcerver, J., Kolesnikov, A., Ruyssen, P., Riquelme, C., Lucic, M., Djolonga, J., Pinto, A. S., Neumann, M., Dosovitskiy, A., Beyer, L., Bachem, O., Tschannen, M., Michalski, M., Bousquet, O., Gelly, S., and Houlsby, N. A large-scale study of representation learning with the visual task adaptation benchmark. *arXiv preprint arXiv:1910.04867*, 2020.

Zhang, M., Lucas, J., Ba, J., and Hinton, G. E. Lookahead optimizer: k steps forward, 1 step back. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., and Torralba, A. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

Zoph, B., Ghiasi, G., Lin, T., Cui, Y., Liu, H., Cubuk, E. D., and Le, Q. Rethinking pre-training and self-training. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.