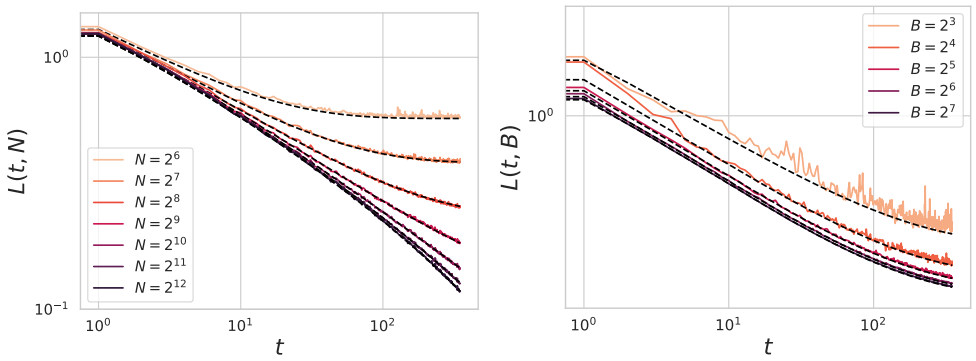
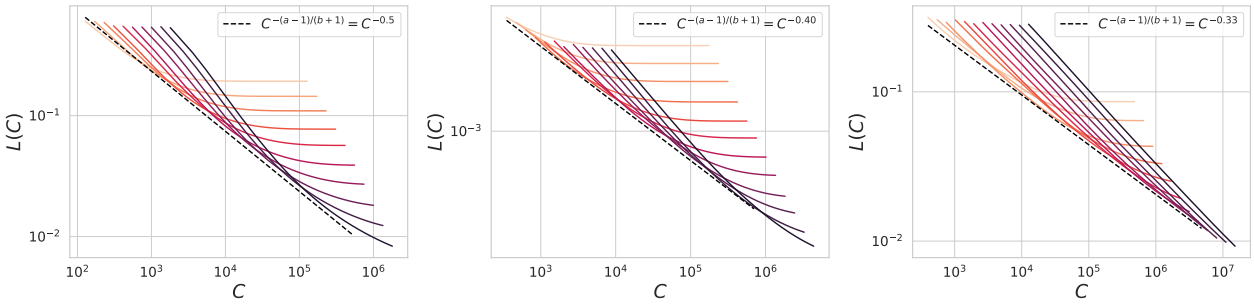


(a) SGD with batchsize $B = 32$ (b) SGD with $N = 256$ and varying batchsize B

Figure 3. Our DMFT can also capture online SGD learning including the effect of batch size fluctuations on the loss and the finite N bottleneck. (a) Power law features trained with SGD and a fixed random projection still generates asymptotes which depend on N. (b) The batchsize B impacts the loss through additional variance in the dynamics but does not lead to an asymptotic plateau.



(a) $(a, b) = (2, 1)$, $L(C) \sim C^{-0.5}$ (b) $(a, b) = (2, 1.5)$, $L(C) \sim C^{-0.4}$ (c) $(a, b) = (2, 2)$, $L(C) \sim C^{-0.33}$

Figure 4. Compute optimal scaling in our model is determined by tradeoff of time and model-size bottlenecks. Solid colored lines are simulations with power law features and in dashed black is the theoretical prediction of compute optimal scaling. Each color represents varying model sizes with $N \in [2^5, 2^{10}]$. The Pareto frontier is defined as the minimum value of L at each compute C over all possible choices of model size N. Although the final losses do not depend on the spectral decay rate b but only on the task-power exponent a, the compute optimal scaling depends does depend on b.

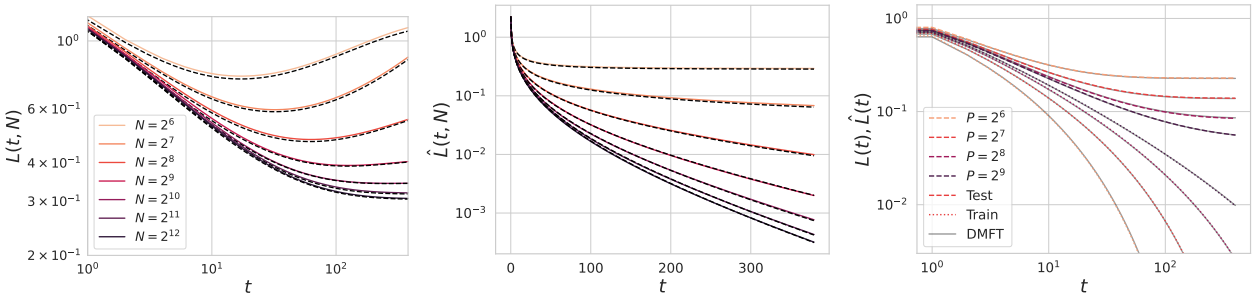
will therefore use this limiting behavior to analyze compute optimal tradeoffs of model size and training time.

Asymmetric Compute Optimal Scaling Strategy We now consider the regime where the total amount of data does not limit performance, but rather training is bottlenecked by time or model size. This could arise in the offline model with very large P or in one-pass SGD with sufficiently small learning rate or sufficiently large batch size (App. K.3). In either case, time and model size should scale differently with compute budget $C = Nt$ and $m = \min\{a - 1, 2b\}$

$$t \sim C^{\frac{bm}{a-1+bm}} \ , \ N \sim C^{\frac{a-1}{a-1+bm}} \ , \implies \mathcal{L}^\star(C) \sim C^{-\frac{(a-1)m}{a-1+bm}} . \tag{17}$$

For the regime of interest where $m = \min\{a - 1, 2b\} = a - 1$ this gives $\mathcal{L}^\star(C) \sim C^{-\frac{a-1}{1+b}}$. We obtain the above scaling by approximating the loss as a sum of the three terms in equation (14) and a constant as in (Hoffmann et al., 2022), see Appendix N. This analysis suggests that for features that have rapid decay in their eigenspectrum, it is preferable to allocate greater resources toward training time rather than model size as the compute budget increases. This is consistent with the small asymmetries observed in (Hoffmann et al., 2022) for language models and the larger asymmetries in MLPs on vision from (Bachmann et al., 2024). In the limit as $b \to 1$, the time and parameter count should be scaled linearly together. We verify this scaling rule and its b-dependence in Figure 4.

Wider is Better Requires Sufficient Data Larger models are not always better in terms of test loss for all time t, as we showed in Figure 1 (c), especially if the dataset is limited. In Figure 5, we illustrate that larger N can improve convergence to a data-bottlenecked loss for power law features. However, the loss may still be non-monotonic in training



(a) Test Loss ($P = 128$) (b) Train Loss ($P = 128$) (c) Train-Test Gaps $N = 512$

Figure 5. In a data limited regime, wider networks train faster but cannot indefinitely improve generalization by making N larger. (a) Test loss for power-law features with $a = 1.5$ and $b = 1.25$ with $P = 128$ and varying N. In this regime, there are diminishing returns to making the model size larger. (b) For $N < P$, the model is underparameterized and cannot achieve zero train loss. For $N > P$, the train loss will eventually decay at exponential rate which depends on N, despite the test loss saturating. (c) The train and test losses gradually separate at a rate which depends on P.

time, motivating regularization or early stopping.

Gradual Buildup of Overfitting Effects The exact gap between train and test losses can exactly be expressed in terms of the DMFT order parameters:

$$\mathcal{L}(t) - \hat{\mathcal{L}}(t) = -\frac{2}{P}\int_0^t dt'\, R_{0,2}(t,t')C_1(t,t') + \frac{1}{P^2}\int_0^t\int_0^t dt'ds' R_{0,2}(t,t')R_{0,2}(t,s')C_1(t',s'). \tag{18}$$

We derive this relation in the Appendix E. At early time this gap goes as $\mathcal{O}(1/P)$ (App. D, E). At late time, however, this picks up a nontrivial task-dependent scaling with P as we show in Figure 2 (e)-(f) and App. C. In Figure 5 (c) we show this gradual accumulation of finite data on the test-train loss gap. For larger datasets P it takes longer training time to begin overfitting (App. E).

Ensembling is Not Always Compute Optimal Ensembling a set of models means averaging their predictions over the same datasets but with different intitialization seeds. This reduces test loss by reducing the variance of the model output f due to initialization. This improvement can be predicted from an extension of our DMFT (App. H). Analogously, bagging over B datasets reduces variance due to sampling of data.

One might imagine that ensembling many finite sized models would allow one to approach the performance of an infinite sized model ($N \to \infty$). If this were possible, the compute optimal strategy could involve a tradeoff between ensemble count and model size. However, recent experiments show that there is a limited benefit from ensembling on large datasets when compared to increasing model size (Vyas et al., 2023). We illustrate this in Figure 6 (a). Our theory can explain these observations as it predicts the effect of ensembling E times on the learning dynamics as we show in App. H. The main reason to prefer increasing N rather than increasing E is that larger N has lower *bias* in the dynamics, whereas ensembling only reduces variance. The bias of the model $\mathcal{B}$ has the form

$$\mathcal{B}(t, N, P) = \sum_k \lambda_k (w_k^\star)^2 H_k(t, N, P)^2, \tag{19}$$

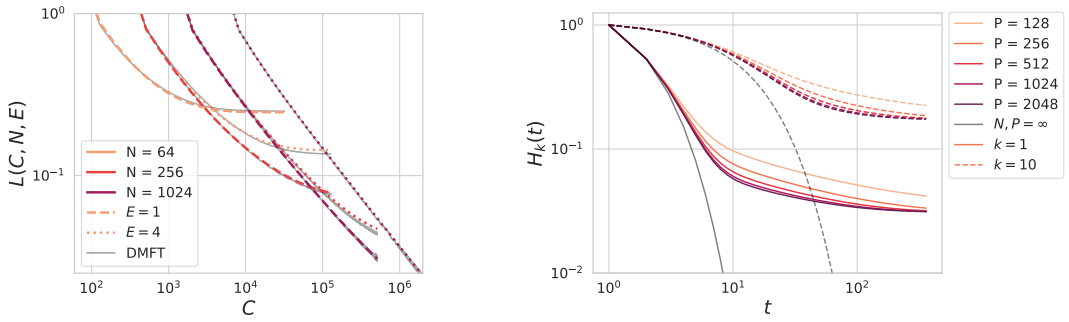
which depend on transfer function H_k that we illustrate for power-law features in Figure 6 (b). Since $H_k(t)$ depend on N, P, we see that ensembling/bagging cannot recover the learning curve of the $N, P \to \infty$ system since the bias is limited by finite N, P.

5. Tests on Realistic Networks

We now move beyond synthetic power-law datasets and consider realistic image datasets and architectures. We take the CIFAR-5M dataset introduced in (Nakkiran et al., 2021a) and consider the task of classfiying animate vs inanimate objects. We plot the spectra of the finite-width NTK at initialization across different widths for a Wide ResNet (Zagoruyko & Komodakis, 2016) in Figure 7 a). Here the width parameter corresponds to the number of channels in the hidden layers. Following (Canatar et al., 2021), we define $C(k)$ as the fraction of the task captured by the top k kernel eigenmodes:

$$C(k) \equiv \frac{\sum_{i \le k} \lambda_i (w_i^*)^2}{\sum_i \lambda_i (w_i^*)^2}. \tag{20}$$

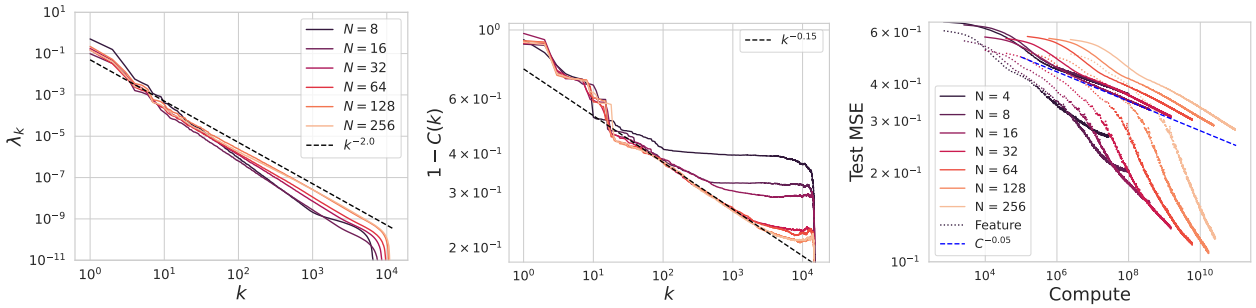
Then $1 - C(k)$ is the portion of the task left unexplained. We plot this for the initial NTKs across widths in Figure 7 b). We extract the spectral decay exponent b and the the task power exponent a from these two curves. Together, these give the learning scaling laws of the linearized neural



(a) E vs N allocation of compute

(b) Transfer Functions $H_k(t)$ for $N = 128$

Figure 6. Ensembling E models of size N improves performance by reducing initialization variance by a factor of E (see Appendix H.2) (a) However, at fixed compute $C = NEt$, increasing the model size N is preferable, since the bias is also reduced. (b) The transfer functions $H_k(t)$ computed from the DMFT determine the error as $E \to \infty$ depend on N, P and saturate in performance at long times, while the $N, P \to \infty$ curves decay exponentially.



(a) NTK Spectra for Varying Widths

(b) Task-Power Decay

(c) Lazy vs Feature Compute Scaling

Figure 7. Our theory predicts time and compute scalings for linearized networks on realistic datasets. (a) The initial NTK spectra and (b) task-power distributions for ResNets of width N on CIFAR-5M are well described by powerlaws $\lambda_k \sim k^{-2.0}$ and $k^{-0.15}$ for large k. (c) The predicted compute optimal scaling of for the ResNet obeys $\mathcal{L}_\star(C) \sim C^{-0.05}$. However, for networks outside of the kernel regime (dashed lines), feature learning can substantially alter the observed scaling laws and improve the loss curves as a function of compute.

network model on this dataset. We plot the compute optimal scaling laws of these linearized models in Figure 7 c). We also plot the predicted scaling law $C^{-(a-1)/(1+b)}$ in blue and find excellent agreement.

5.1. The Role of Feature Learning

We also compare these scalings to those of the compute optimal learning curves for feature-learning networks. We train several networks with different widths and initialization seeds for 64 epochs through the dataset. We observe substantially different compute-optimal scaling exponents in the dotted curves of Figure 7 c). This means that although our random feature model does capture the correct linearized scaling trends, which have all of the qualities observed in realistic scaling laws, more is needed to capture the acceleration of scaling induced by feature learning. Further analyses of the after-kernels of feature learning networks are performed in Appendix L. We see that the kernels continue to evolve substantially throughout training. This indicates that a full explanation of the compute optimal scaling exponents will require something resembling a mechanistic theory of kernel evolution (Long, 2021; Fort et al., 2020; Atanasov et al., 2022; Bordelon & Pehlevan, 2022b).

6. Conclusion

We have presented a model that recovers a wide variety of phenomena observed in more realistic deep learning settings. Our theory includes not just model size and dataset size as parameters but also explicitly treats the temporal dynamics of training. We observe different scaling exponents for performance in terms of model size and number of time steps. Future work to incorporate kernel evolution into this model could further shed insight into the improved scaling laws in the feature-learning regime. Overall, our results provide a theoretical interpretation of compute-optimal scaling as a competition between the training dynamics of the infinite width/infinite data limit and finite model-size bottleneck.