A. Deriving the linear scaling rule for small batch sizes

In section 3 of the main text, we applied the central limit theorem to approximate a single SGD step by,

$$\Delta\omega_i = (\omega_{i+1} - \omega_i) \approx -\epsilon \frac{dC}{d\omega}\Big|_{\omega=\omega_i} + \sqrt{\epsilon T}\nu_i. \quad (4)$$

The temperature $T = \epsilon/B$, $\mathbb{E}(\nu_i) = 0$ and $\mathbb{E}\left(\nu_i \nu_j^\top\right) = F(\omega_i)\delta_{ij}$, where $F(\omega)$ is the empirical Fisher information matrix and δ_{ij} is the dirac delta function. Equation 4 holds so long as the gradient of each training example is an independent and uncorrelated sample from an underlying short tailed distribution. Additionally, it assumes that the training set size $N \gg B$ and the batch size $B \gg 1$. To derive the linear scaling rule, we consider the total change in the parameters over n consecutive SGD parameter updates,

$$\Delta\omega_i' = \sum_{j=0}^{n-1} \Delta\omega_{i+j} \approx -\epsilon \left(\sum_{j=0}^{n-1} \frac{dC}{d\omega}\Big|_{\omega=\omega_{i+j}} \right) + \sqrt{n\epsilon T}\xi_i. \quad (5)$$

The noise $\xi_i = (1/\sqrt{n}) \sum_{j=0}^{n-1} \nu_{i+j}$. When the product of the number of steps n and the learning rate ϵ is much smaller than the critical learning rate, $n\epsilon \ll \epsilon_{crit}$, the parameters do not move far enough for the gradients to significantly change, and therefore for all $\{j, j'\}$ greater than 0 and less than n,

$$\frac{dC}{d\omega}\Big|_{\omega=\omega_{i+j}} \approx \frac{dC}{d\omega}\Big|_{\omega=\omega_i} \quad (6)$$

$$\mathbb{E}\left(\nu_{i+j}\nu_{i+j'}\right) \approx F(\omega_i)\delta_{jj'} \quad (7)$$

Using equation 6, we can rewrite equation 5 as

$$\Delta\omega_i' \approx -n\epsilon \frac{dC}{d\omega}\Big|_{\omega=\omega_i} + \sqrt{n\epsilon T}\xi_i. \quad (8)$$

Equation 7 implies that $\mathbb{E}(\xi_i) = 0$ and $\mathbb{E}\left(\xi_i \xi_i^\top\right) \approx F(\omega_i)$. We therefore conclude that ξ and ν are both Gaussian random variables from the same distribution. Comparing equation 4 and equation 8, we conclude that n SGD updates at temperature T with learning rate ϵ is equivalent to a single SGD step at temperature T with learning rate $n\epsilon$. Since the temperature $T = \epsilon/B$, this implies that when $\epsilon \ll \epsilon_{crit}$, then simultaneously doubling both the learning rate and the batch size should draw samples from the same distribution over parameters after the same number of training epochs.

This prediction is known as the linear scaling rule (Krizhevsky, 2014; Goyal et al., 2017; Mandt et al., 2017; Smith & Le, 2017; Jastrzębski et al., 2017; Chaudhari & Soatto, 2018; McCandlish et al., 2018; Shallue et al., 2018). Since this linear scaling rule assumes that $\epsilon \ll \epsilon_{crit}$, it usually holds when the batch size is small, which appears to contradict the assumption $B \gg 1$ above. Crucially however, the distribution of ν_i does not matter in practice, since our dynamics is governed by the combined influence of noise over multiple consecutive updates, $\xi_i = (1/\sqrt{n}) \sum_{j=0}^{n-1} \nu_{i+j}$.

In other words, we do not require that equation 4 is an accurate model of an single SGD step, we only require that equation 8 is an accurate model of n SGD steps. We therefore conclude that ν_i does not need to be Gaussian, we only require that ξ_i is Gaussian. The central limit theorem predicts that, if ν_i is an independent random sample from a short-tailed distribution, ξ_i will be Gaussian if $N \gg 1$, $nB \gg 1$ and $nB \ll N$. If $\epsilon \ll \epsilon_{crit}$, then we can choose $1 \ll n \ll N$, and discard the assumption $B \gg 1$.

B. Additional experimental details

In this section, we provide additional details about the experimental setup and models considered in our study.

B.1. Our learning rate decay schedule

We illustrate our default learning rate decay schedule in figure 4. As specified in the main text, if the epoch budget is N_{epochs}, we hold the learning rate constant for $N_{epochs}/2$, before decaying the learning rate by a factor of γ every $N_{epochs}/20$. Unless specified otherwise, $\gamma = 2$. Note that in our constant step experiments, the epoch budget is proportional to the batch size, which ensures that all batch sizes decay the learning rate after the same number of steps.

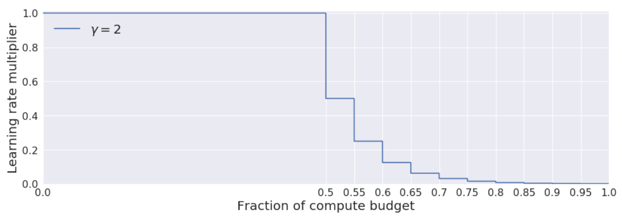


Figure 4. Our default learning rate schedule with $\gamma = 2$.

B.2. Additional models used

In addition to the 16-4 Wide-ResNet model on CIFAR-10 presented in the main paper (Zagoruyko & Komodakis, 2016), we provide additional experiments in the appendix using ResNet-50 on ImageNet (He et al., 2016), 28-10 Wide-ResNet on CIFAR-100 (Zagoruyko & Komodakis, 2016), LSTMs on Penn TreeBank (Zaremba et al., 2014) and autoencoders on MNIST (Sutskever et al., 2013).

ResNet50 on ImageNet: We follow the modified ResNet-50 implementation of Goyal et al. (2017) for training on ImageNet, and we use our default learning rate schedule without learning rate warmup (see appendix B.1). Due to the large compute budget required for these models, we train a single model for each batch size/learning rate pair.

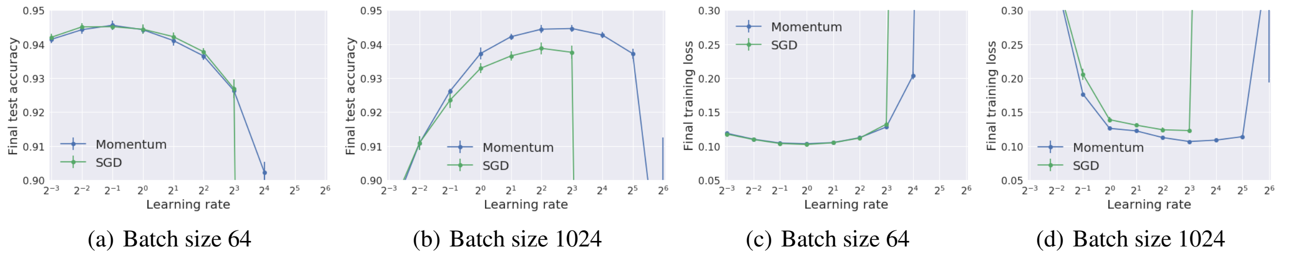


Figure 5. A 16-4 Wide-ResNet, trained with batch normalization on CIFAR-10 for 200 epochs. For completeness, we provide the performance at a range of learning rates for two batch sizes, 64 and 1024 (the performance for a range of batch sizes at the optimal learning rate is shown in figure 1). The smaller batch size is in the noise dominated regime, while the larger batch size is in the curvature dominated regime. We provide the final test accuracies in figures a and b, and the final training losses at in figures c and d. SGD and SGD with Momentum always achieve similar final performance in the small learning rate limit, while SGD performs poorly when the learning rate is large. When the batch size is small, the optimal learning rate is also small, and so both methods have similar optimal test accuracy/training loss. When the batch size is large, the optimal learning rate is large, and SGD with Momentum performs better.

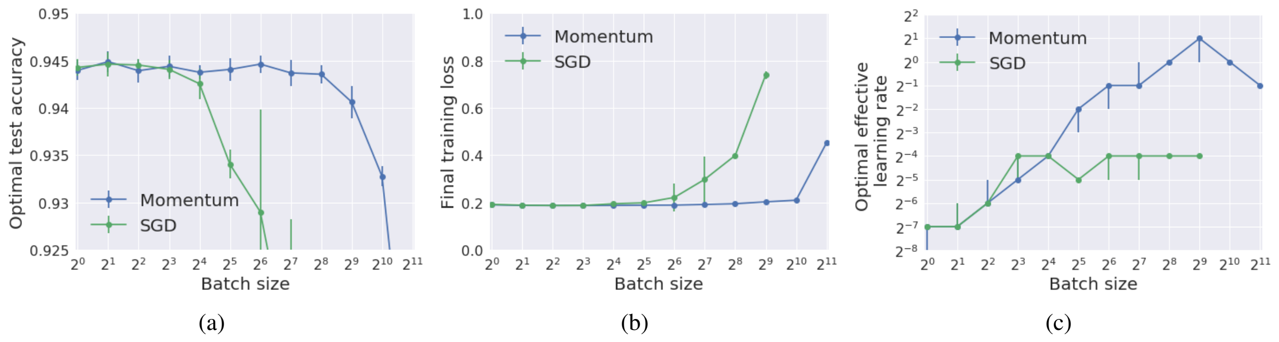


Figure 6. A 16-4 Wide-ResNet, trained without batch normalization using Regularized SkipInit (De & Smith, 2020) on CIFAR-10 for 200 epochs. We report the performance of SGD and SGD with Momentum. We perform a grid search to identify the optimal learning rate which maximizes the test accuracy, and report the mean performance of the best 12 of 15 runs. a) The test accuracy is independent of batch size when the batch size is small, but begins to fall when the batch size is sufficiently large ($B \gtrsim 8$ for SGD and $B \gtrsim 256$ for SGD with Momentum. b) The training loss at the optimal effective learning rate is independent of batch size when the batch size is small, but rises rapidly when the batch size is sufficiently large. c) The optimal effective learning rate is proportional to batch size when the batch size is small for both both SGD and SGD with Momentum, while it is independent of batch size when the batch size is sufficiently large.

28-10 Wide-ResNet on CIFAR100: We train 28-10 Wide-ResNets on CIFAR-100 (Zagoruyko & Komodakis, 2016). We use our default learning rate schedule, as described in appendix B.1, which reaches the same test set accuracy as is reported by Zagoruyko & Komodakis (2016).

LSTM on Penn TreeBank: We train a word-level LSTM language model on the Penn TreeBank dataset (PTB), following the implementation described in Zaremba et al. (2014). The LSTM model used has two layers with 650 units per layer. The parameters are initialized uniformly in $[-0.05, 0.05]$. We apply gradient clipping at 5, as well as dropout with probability 0.5 on the non-recurrent connections. We train the LSTM using an unroll step of 35, and use the learning rate decay schedule described in appendix B.1. As with the other models tested in this paper, this learning rate schedule reaches the same test perplexity performance as the original schedules reported in (Zaremba et al., 2014).

Autoencoder on MNIST: We train a fully-connected autoencoder on MNIST. Our network architecture is described by the sequence of layer widths $\{784, 1000, 500, 250, 30, 250, 500, 1000, 784\}$, where 784 denotes the input and output dimensions. For more details on this architecture, we refer to Sutskever et al. (2013). This model has often been used as an optimization benchmark (Sutskever et al., 2013; Kidambi et al., 2018). The L2 regularization parameter was set at 10^{-5}. We use our default learning rate schedule, as described in appendix B.1.

C. Additional results under constant epoch budgets

In this section, we provide additional experimental results to verify the existence of two regimes of SGD under a constant epoch budget. In all cases, we observe a transition from a

Table 2. ResNet-50, trained on ImageNet for 90 epochs. We follow the implementation of Goyal et al. (2017), however we introduce our modified learning rate schedule defined in appendix B.1. We perform a grid search to identify the optimal effective learning rate and report the performance of a single training run. The test accuracies achieved by SGD and Momentum are equal when the batch size is small, but Momentum outperforms SGD when the batch size is large. For SGD with Momentum, the optimal effective learning rate is proportional to batch size for all batch sizes considered, while this linear scaling rule breaks at large batch sizes for SGD.

	Batch size	**Optimal test accuracy (%)**	**Training loss**	**Optimal effective learning rate**
SGD	256	77.0	2.25	1.0
	1024	76.7	2.25	4.0
	4096	76.1	2.30	8.0
Momentum	256	77.0	2.25	1.0
	1024	76.8	2.25	4.0
	4096	76.8	2.25	16.0

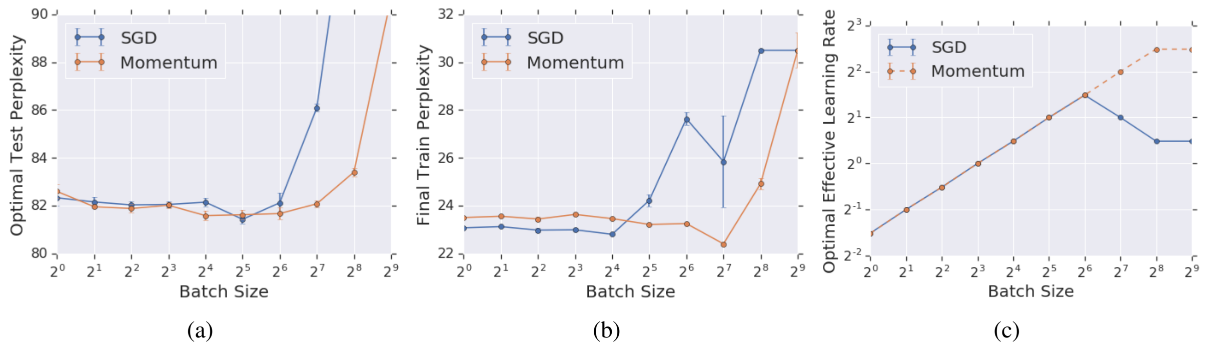


Figure 7. A word-level LSTM language model trained on PTB for 40 epochs. We report the performance of SGD and SGD with Momentum. We perform a grid search to identify the optimal learning rate which maximizes the test set perplexity, and report the mean performance of the best 5 of 7 runs. a) The test set perplexity of SGD with Momentum is independent of batch size when the batch size is small, but begins to rise when the batch size exceeds 128. The test set perplexity of vanilla SGD starts rising for batch sizes exceeding 64. b) We see similar phenomena on the training set perplexity. c) Surprisingly, the optimal effective learning rate is proportional to *square root* of the batch size when the batch size is small, while it levels off for larger batch sizes. We note that the gradients of consecutive minibatches in a language model are not independent, which violates the assumptions required to derive the linear scaling rule.

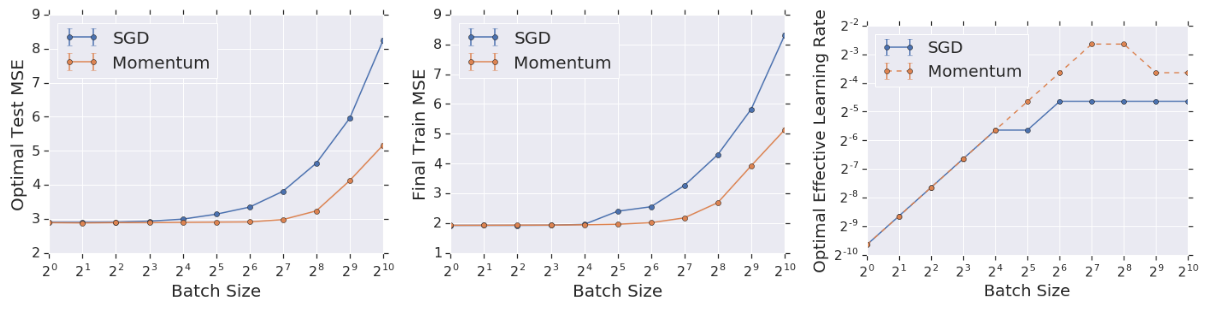
small batch regime, where the learning rate increases with the batch size and SGD with Momentum does not outperform SGD, to a large batch regime, where the learning rate is independent of the batch size and SGD with Momentum outperforms SGD. Under a constant epoch budget, both the training loss and the optimal test accuracy are independent of batch size in the noise dominated regime, but begin to degrade when one enters the curvature dominated regime.

C.1. Learning rate sweep with batch normalization for two batch sizes on CIFAR-10 Wide-ResNet model

In figure 5, we provide additional results with the 16-4 Wide-ResNet, trained with batch normalization on CIFAR-10 for 200 epochs. Here we provide the final test set accuracies and the final training set losses for a full learning rate sweep at two batch sizes, 64 and 1024. From figure 5, we see that SGD and Momentum always achieve similar final performance in the small learning rate limit. This confirms previous theoretical work showing the equivalence of SGD and Momentum in the small learning rate limit when the momentum parameter is kept fixed (Orr & Leen, 1994; Qian, 1999; Yuan et al., 2016). Meanwhile, SGD performs poorly compared to SGD with Momentum when the learning rate is large. When the batch size is small, the optimal learning rates for both methods are also small, and so the two methods have the same optimal test accuracy. However when the batch size is large, the optimal learning rate is large, and consequently SGD with Momentum outperforms vanilla SGD. These results are entirely consistent with the two regimes of SGD discussed in section 3.

C.2. Results without batch normalization on CIFAR-10 Wide-ResNet

In figure 6 we present results when training our 16-4 Wide-ResNet (Zagoruyko & Komodakis, 2016). We follow the same setup and learning rate schedule described in section 4, and we train for 200 epochs. However we remove batch normalization, and introduce the Regularized Skip-



(a) (b) (c)

Figure 8. A fully connected autoencoder, trained on MNIST for 200 epochs. We report the performance of SGD and SGD w/ Momentum. We perform a grid search to identify the optimal learning rate which maximizes the mean-squared error (MSE) on the test set, and report the mean performance of the best 5 of 7 runs. a) The test MSE of SGD w/ Momentum is initially independent of batch size, but it begins to rise when the batch size exceeds 128. The test MSE of vanilla SGD starts rising for batch sizes exceeding 16. b) We see similar phenomena on the training set MSE. c) The optimal effective learning rate is proportional to batch size when the batch size is small for both vanilla SGD and SGD w/ Momentum, while it becomes independent of batch size for larger batch sizes. The optimal effective learning rate in the curvature dominated regime is larger for SGD w/ Momentum.

Init initialization scheme proposed by De & Smith (2020). This initialization scheme enables the training of very deep networks, and it reduces the gap in test accuracy between networks trained with and without batch normalization.

We observe remarkably similar trends to those observed in section 5 of the main text, although the critical learning rate, beyond which the optimal learning rate of SGD is independent of batch size, is significantly smaller when batch normalization is not used. The performance of SGD on both the training and the test set is independent of batch size for very small batch sizes $B \lesssim 8$, while the performance of SGD with Momentum is constant for batch sizes $B \lesssim 256$. Above these thresholds, the performance of both methods degrades rapidly. These observations are explained by the optimal effective learning rates in figure 6(c). SGD with Momentum has a significantly larger maximum stable learning rate, enabling it to scale to larger batch sizes.

C.3. Results from additional models

In table 2, we provide results for ResNet-50 trained on ImageNet for 90 epochs at a small range of batch sizes. SGD with and without Momentum achieve similar test accuracies when the batch size is small, but SGD with Momentum outperforms SGD without Momentum when the batch size is large. The optimal effective learning rate is proportional to batch size for all batch sizes considered when using SGD with Momentum, but not when using vanilla SGD.

In figure 7, we present results for the LSTM on PTB trained for 40 epochs. Once again, we see that SGD and SGD with Momentum have similar performance for small batch sizes. Performance for SGD starts degrading for batch sizes exceeding 64, whereas performance for SGD with Momentum starts degrading for batch sizes exceeding 128. However, unlike our previous experiments, we notice that the optimal learning rate increases as *square root* of the batch size for small batch sizes, before leveling off at a constant value for larger batch sizes. The square root scaling observed here could be due to correlations between consecutive data samples when training the LSTM, which violate the assumptions used to derive the linear scaling rule in section 3.

In figure 8, we present results on training the fully-connected autoencoder on MNIST for 200 epochs. As before, we notice that for small batch sizes, the performance of both SGD and SGD with Momentum is independent of batch size, while performance begins to degrade when the batch size is large. On this model, the performance of SGD begins to degrade at much smaller batch sizes than we observed in normalized residual networks, and consequently SGD with Momentum starts outperforming SGD at much smaller batch sizes. This is likely due to the poor conditioning of the model due to the bottleneck structure of its architecture.

D. Additional results under constant step budgets

In this section, we provide additional results studying how the optimal test accuracy depends on the batch size under a constant step budget for a range of models. We train with SGD with Momentum. In each case, we set the number of training steps to be equal to the number of training steps taken by the largest batch size before performance starts degrading under our constant epoch budget experiments.

In table 3, we show results for training a 16-4 Wide-ResNet on CIFAR-10 without batch normalization using Regular-

Table 3. The optimal test accuracy and final training loss for a range of batch sizes under a constant step budget. For each batch size, we train a 16-4 Wide-ResNet without batch normalization on CIFAR-10 using Regularized SkipInit (De & Smith, 2020) for 156,250 updates. We perform a grid search to identify the optimal learning rate which maximizes the test accuracy, and we provide the average performance of the best 12 out of 15 training runs. The final test accuracy falls for very large batches. We note that, although the final training loss rises slightly at batch size 2048, the training loss remains lower than that achieved at batch size 128 (for which the test accuracy was maximized).

Batch size	Optimal test accuracy	Final training loss	Optimal effective learning rate
16	92.6 ± 0.2	0.349 ± 0.004	2^{-4} (2^{-4} to 2^{-2})
32	93.9 ± 0.1	0.269 ± 0.004	2^{-3} (2^{-3} to 2^{-2})
64	94.4 ± 0.1	0.192 ± 0.002	2^{-2} (2^{-2} to 2^{-1})
128	94.6 ± 0.1	0.122 ± 0.000	2^{-1} (2^{-1} to 2^{-1})
256	94.4 ± 0.1	0.071 ± 0.001	2^{-1} (2^{-1} to 2^{0})
512	94.1 ± 0.1	0.043 ± 0.000	2^{-0} (2^{-1} to 2^{1})
1024	93.8 ± 0.1	0.028 ± 0.000	2^{-1} (2^{-1} to 2^{0})
2048	93.0 ± 0.6	0.054 ± 0.050	2^{-1} (2^{-1} to 2^{0})

Table 4. The optimal test accuracy and final training loss for a range of batch sizes under a constant step budget. For each batch size, we train a 28-10 Wide-ResNet with batch normalization on CIFAR-100 for 9765 updates. We perform a grid search to identify the optimal learning rate which maximizes the test accuracy, and we provide the average performance of the best 12 out of 15 training runs. The test accuracy initially rises as the batch size rises, but it falls for very large batches. The training loss also rises at the largest batch size considered of 16384 but remains comparable to that observed at batch size 2048 (for which the test accuracy was maximized).

Batch size	Optimal test accuracy (%)	Final training loss	Optimal effective learning rate
256	78.9 ± 0.1	0.609 ± 0.004	2^{2} (2^{2} to 2^{2})
512	79.9 ± 0.2	0.462 ± 0.008	2^{3} (2^{2} to 2^{3})
1024	80.1 ± 0.2	0.274 ± 0.003	2^{4} (2^{3} to 2^{4})
2048	80.2 ± 0.2	0.132 ± 0.002	2^{4} (2^{4} to 2^{4})
4096	79.6 ± 0.2	0.073 ± 0.001	2^{5} (2^{5} to 2^{5})
8192	78.1 ± 0.4	0.045 ± 0.001	2^{5} (2^{1} to 2^{5})
16384	72.2 ± 0.2	0.156 ± 0.036	2^{1} (2^{1} to 2^{2})

ized SkipInit (De & Smith, 2020) for 156250 updates. This corresponds to 200 epochs when the batch size is 64. The final training loss falls as the batch size rises, while the test accuracy drops for large batch sizes. At the largest batch size considered, both the test accuracy and the training loss exhibit a large standard deviation across different training runs (at the optimal learning rate). At this batch size, we note that the variance is much lower at lower learning rates (at which the training loss is also lower), however these smaller learning rates also achieve lower mean test accuracy.

In table 4, we train a 28-10 Wide-ResNet on CIFAR-100 with batch normalization for 9765 steps at a range of batch sizes (which corresponds to 200 epochs when the batch size is 1024), while in table 5 we train a 28-10 Wide-ResNet on CIFAR-100 without batch normalization (using Regularized SkipInit) for 156,250 steps (which corresponds to 200 epochs when the batch size is 64). In both cases the test accuracy drops significantly when the batch size is very large. In table 4 the training loss falls as the batch size rises, while in 5 the training loss rises slightly for very large batches.

In table 6, we train the word-level LSTM on the Penn Tree-Bank (PTB) dataset (Zaremba et al., 2014) for 16560 updates. This corresponds to 40 epochs at batch size 64. We described this model in appendix B, and we train using the learning rate schedule defined in appendix B.1 using SGD with Momentum. The test perplexity increases as the batch size increases, while the training perplexity falls.

In table 7, we train a fully connected auto-encoder on MNIST for 156,250 updates (Sutskever et al., 2013). This corresponds to 200 epochs when the batch size is 64. We described this model in appendix B, and we train using the learning rate schedule defined in appendix B.1 using SGD with Momentum. The test set MSE increases slightly as the batch size increases, while the training set MSE falls as the batch size rises. Although the training set MSE does appear to rise slightly for a batch size of 4096, we note that the training loss in this case is similar to that achieved with a batch size of 64, while the test set MSE in this case is worse than that at batch size 64. The optimal effective learning rate is independent of the batch size, suggesting that the learning rate may be close to curvature dominated regime. We note that the benefits of noise appear to be significantly smaller in this architecture than for the Wide-ResNet or LSTM.

Table 5. The optimal test accuracy and final training loss for a range of batch sizes under a constant step budget. For each batch size, we train a 28-10 Wide-ResNet on CIFAR-100 without batch normalization using Regularized SkipInit for 156,250 updates. We perform a grid search to identify the optimal learning rate which maximizes the test accuracy, and we provide the average performance of the best 12 out of 15 training runs. The final test accuracy falls for very large batches, while surprisingly the training loss also rises slightly.

Batch size	Optimal test accuracy	Final training loss	Optimal effective learning rate
32	77.9 ± 0.2	0.0512 ± 0.0611	0.025 (0.025 to 0.050)
64	79.0 ± 0.2	0.0141 ± 0.0281	0.050 (0.050 to 0.050)
128	79.4 ± 0.1	0.0011 ± 0.0004	0.100 (0.100 to 0.100)
256	78.9 ± 0.1	0.0191 ± 0.0221	0.200 (0.200 to 0.200)
512	77.7 ± 0.1	0.0022 ± 0.0035	0.200 (0.200 to 0.200)
1024	76.0 ± 0.1	0.0009 ± 0.0012	0.200 (0.200 to 0.200)
2048	74.2 ± 0.3	0.0029 ± 0.0025	0.200 (0.200 to 0.200)

Table 6. The optimal test set perplexity and final training set perplexity for a range of batch sizes under a constant step budget. For each batch size, we train a word-level LSTM on PTB for 16560 updates. We perform a grid search to identify the optimal learning rate which maximizes the test set perplexity, and we provide the average performance of the best 5 out of 7 training runs. The optimal test perplexity increases as the batch size rises, while the training perplexity falls.

Batch size	Optimal test perplexity	Final training perplexity	Optimal effective learning rate
16	88.06 ± 0.26	35.85 ± 0.06	1.4 (1.4 to 1.4)
32	82.50 ± 0.13	28.99 ± 0.03	2.0 (2.0 to 2.0)
64	81.67 ± 0.26	23.25 ± 0.05	2.8 (2.8 to 2.8)
128	86.04 ± 0.49	21.16 ± 0.06	5.6 (5.6 to 5.6)
256	92.19 ± 0.26	15.74 ± 0.03	5.6 (5.6 to 5.6)

E. Additional results with a fixed batch size and variable epoch budget

We now provide additional experimental results to accompany those provided in section 7, where we study whether the optimal training temperature is independent of the epoch budget. We use SGD with Momentum with the momentum parameter $m = 0.9$ for all our experiments in this section.

In figure 9, we present results on a word-level LSTM on the PTB dataset for a batch size of 64 and for varying epoch budgets. Note that the original LSTM model in Zaremba et al. (2014) was trained for 39 epochs. The results in figure 9 are remarkably similar to those presented in figure 2. As the epoch budget rises, the test set perplexity first falls but then begins to increase. The training set perplexity falls monotonically as the epoch budget increases. Finally, the optimal learning rate which minimizes the test set perplexity is independent of the epoch budget once this epoch budget is not too small, while the optimal learning rate which minimizes the training set perplexity falls.

In figure 10, we present results on a fully connected autoencoder trained on MNIST for a batch size of 32 and for a range of epoch budgets. Note that the autoencoder results presented in section C were trained for 200 epochs. Figures 10(a) and 10(b) are similar to figures 2(a) and 2(b) in the main text. Initially the test set MSE falls as the epoch budget increases, but then it starts increasing. The training set MSE falls monotonically as the epoch budget rises. In figure 10(c) however, we notice that the learning rate that minimizes the test set MSE decreases as the epoch budget rises. This is the opposite of what we observed in figures 2 and 9. To further investigate this, in figure 10(d) we plot the mean test set MSE during training for an epoch budget of 800 for learning rates $\epsilon = 0.004$ and $\epsilon = 0.002$. We notice that for the larger learning rate $\epsilon = 0.004$, the model overfits faster on the training set, causing the test set MSE to rise by the time of the first learning rate drop at 400 epochs. This is consistently the case for all epoch budgets over 200 epochs. To avoid the test set MSE from rising, the optimal learning rate for the test MSE drops to slow down training sufficiently such that there is no overfitting before the first learning rate decay. Meanwhile the optimal learning rate to minimize the training loss is more or less constant. This suggests that early stopping is particularly important in this architecture and dataset, and that it has more influence on the final test performance than stochastic gradient noise.

Table 7. The optimal test set MSE and final training set MSE for a range of batch sizes under a constant step budget. For each batch size, we train a fully connected autoencoder on MNIST for 156,250 updates. We perform a grid search to identify the optimal learning rate which maximizes the test set MSE, and we provide the average performance of the best 5 out of 7 training runs. The final test MSE falls for large batch sizes, although this effect is rather weak in this model.

Batch size	Optimal test set MSE	Final training set MSE	Optimal effective learning rate
64	2.91 ± 0.01	2.017 ± 0.003	0.08 $(0.08$ to $0.08)$
256	2.95 ± 0.01	2.010 ± 0.005	0.08 $(0.08$ to $0.08)$
1024	2.96 ± 0.01	2.005 ± 0.011	0.08 $(0.08$ to $0.08)$
4096	2.98 ± 0.01	2.018 ± 0.008	0.08 $(0.08$ to $0.08)$

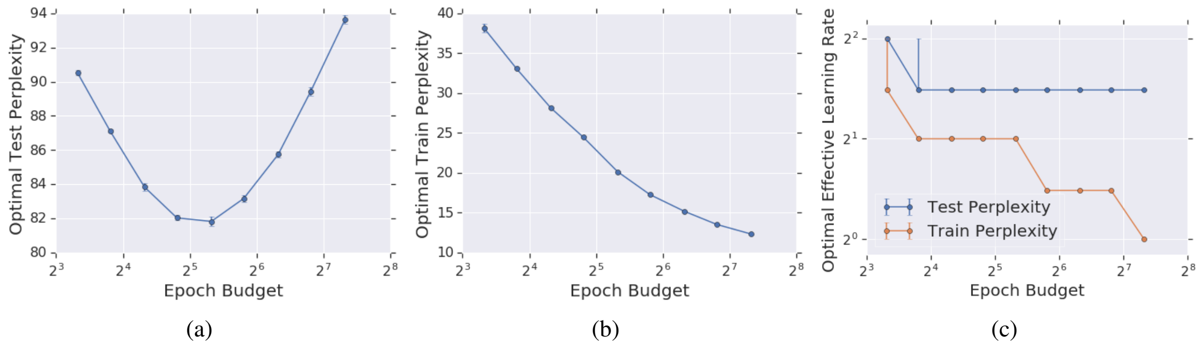


Figure 9. The performance of a word-level LSTM language model trained on the Penn TreeBank dataset using SGD with Momentum and a batch size of 64 at a range of epoch budgets. We identify both the optimal effective learning rate which minimizes the test set perplexity and the optimal effective learning rate which minimizes the training set perplexity, and we present the mean performance of the best 5 out of 7 runs. a) Initially the test set perplexity falls as the epoch budget increases, however it begins to rise beyond 56 training epochs. b) The training set perplexity falls monotonically as the epoch budget rises. c) The learning rate that minimizes the training set perplexity falls as the epoch budget rises, while the learning rate that minimizes the test set perplexity only varies by a factor of 2 when the epoch budget rises over two orders of magnitude.

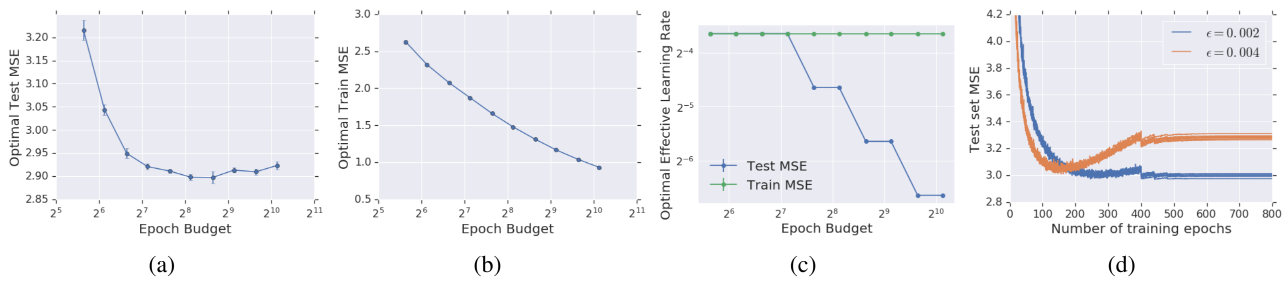


Figure 10. The performance of a fully connected autoencoder on MNIST using SGD with Momentum and a batch size of 32 at a range of epoch budgets. We identify both the optimal effective learning rate which minimizes the test set MSE and the optimal effective learning rate which minimizes the training set MSE, and we present the mean performance of the best 5 out of 7 runs. a) Initially the test set MSE falls as the epoch budget increases, before rising slightly for large epoch budgets. b) The train set MSE falls monotonically as the the epoch budget rises. c) The learning rate that minimizes the test set MSE decreases, while the learning rate that minimizes the train set MSE remains constant as the epoch budget rises. This is contrary to what we observe in figures 2 and 9. The reason for this is apparent from figure d), where we plot the test set MSE during training for all 7 runs for an epoch budget of 800 for learning rate $\epsilon = 0.004$ and $\epsilon = 0.002$. We notice that for a larger learning rate, the model overfits on the training set faster, causing the test set MSE to rise by the time of the first learning rate drop at 400 epochs. This suggests that early stopping has more influence on the final test performance in this architecture than stochastic gradient noise.