The effects of optimization on generalization in infinitely wide neural networks

Anastasia Borovykh [1]

Abstract

Understanding the neural network output and its behavior during training is crucial to gaining insight into the generalization capabilities of the network. It is well-known that infinite width neural networks can be represented as Gaussian processes. This however does not explain why certain networks can generalize while other perform bad out of sample. In this work we give insight into the generalization capabilities of an infinitely wide neural network through the output function evolution as governed by the neural tangent kernel (NTK). By explicitly solving for the network output dynamics, we present a theoretically grounded overview of the effects of certain hyperparameters of the optimization algorithm on generalization, in particular focusing on the effects of noise in the gradient updates.

1. Introduction

The generalization capability of any model, i.e. how well the model optimized on train data will be able to generalize to unseen test data, can be related to two factors; first of all, the model should be minimal in some sense of complexity; second of all, the model should not overfit on the noise present in the observations. These two factors result in the information bottleneck, a trade-off between the data fit and the model complexity (Tishby & Zaslavsky, 2015). Previous work has shown that the norm (Bartlett et al., 2017), (Neyshabur et al., 2015), (Li et al., 2018), the width of a minimum in weight space (Hochreiter & Schmidhuber, 1997), (Sagun et al., 2018), the input sensitivity (Novak et al., 2018) and a model's compressibility (Arora et al., 2018) can be related (either theoretically or in practice) to the model's complexity and thus its ability to perform well on unseen data. Besides model complexity, the generalization error is also related to the amount of noise present in the data and for good generalization avoiding memorization and overfitting on noise is a key challenge (Geirhos et al., 2018), (Zhang et al., 2016).

The optimization algorithm used to train the neural network can be used to bias the model into configurations that are more robust to noise and have lower model complexity, see e.g. (Arora et al., 2019), (Gunasekar et al., 2017), (Neyshabur et al., 2017a). Furthermore, it has been observed that certain parameters of SGD can be used to control the generalization error and the data fit (Jastrzkebski et al., 2017), (Seong et al., 2018), (Borovykh et al., 2019), (Chaudhari & Soatto, 2018), (Li et al., 2019). Understanding the regularization introduced by the optimization scheme in deep neural networks is crucial for understanding what and how the model learns.

In the case of an infinitely wide neural network, under the lazy training regime (Chizat & Bach, 2018), the stochastic differential equation (SDE) for the output evolution during training with gradient descent is governed by the neural tangent kernel (Jacot et al., 2018) (Yang, 2019). In particular, the neural network evolution is in this case equivalent to that of a linear model (Lee et al., 2019), and the output function can be solved for explicitly. Using this output evolution the networks convergence and generalization capabilities can be studied in order to understand how these depend on the choice of optimization algorithm. In Section 2 we give theoretical insight into the effects of the hyperparameters in the optimization on the models' generalization capabilities, in particular focussing on gradient descent and the effects of noise in the updates. We demonstrate the effects using numerical examples in Section 3.

2. Deep neural network evolution

Consider an input $x \in \mathbb{R}^{n_0}$ and a feedforward neural network consisting of L layers with n_l hidden nodes in each layer $l = 1, ..., L-1$ and a read-out layer with $n_L = 1$ output node. Each layer l in the network then computes for each $i = 1, ..., n_l$

$$a_i^l(x) = \frac{\sigma_w}{\sqrt{n_{l-1}}} \sum_{j=1}^{n_{l-1}} w_{i,j}^l z_j^{l-1} + \sigma_b b_j^l, \qquad (1)$$
$$z_i^l(x) = h(a_i^l(x)),$$

*Equal contribution [1]CWI Amsterdam, the Netherlands. Correspondence to: Anastasia Borovykh <anastasia.borovykh@cwi.nl>.

ICML 2019 Workshop on Understanding and Improving Generalization in Deep Learning, Long Beach, California, 2019.

where σ_w^2, σ_b^2 are the weight and bias variances, respectively, $h(\cdot)$ is the non-linear activation function, $w^l \in \mathbb{R}^{n_{l-1} \times n_l}$ and $b^l \in \mathbb{R}^{n_l}$. This parametrisation differs slightly from the standard one due to the scaling of the weights in both the forward and backward pass (see also Appendix E in (Lee et al., 2019)). Note that in the first layer we compute the linear combination using the input, i.e. $z^0 = x$ with n_0 as the input layer size. The output is given by $\hat{y}(x) = a^{L+1}(x)$. Let $X \in \mathbb{R}^{N \times n_0}$, $Y \in \mathbb{R}^{N \times 1}$ be the training set. Working in the regression setting we assume here that the loss is given by the mean-squared error: $\mathcal{L}(X, \hat{y}_t, Y) = \frac{1}{2N}||\hat{y}(X) - Y||_2^2$.

2.1. Gradient descent

At initialisation $t = 0$, the output $\hat{y}_0$, is a random function with the limiting distribution being a Gaussian process (Neal, 2012), (Lee et al., 2018), (Matthews et al., 2018), i.e. as $n_1, ..., n_l \to \infty$ it holds that $a_i^l \sim \mathcal{GP}(0, k^l(x^p, x^q))$, with $k^l(x, x') = \mathbb{E}\left[a_i^l(x) a_i^l(x')\right]$.

Consider a FNN trained with gradient descent. Set $\theta^l = [w_{i,j}^l, b_j^l]_{i,j}$ to be the collection of parameters mapping to the l-th layer such that $\theta^l \in \mathbb{R}^{(n_{l-1}+1) \times n_l}$ and let $\theta \in \mathbb{R}^d$ where $d = (n_0 + 1)n_1 + ...(n_l + 1)n_L$ be the vectorised form of all parameters. If the number of neurons in an FNN sequentially goes to infinity $n_1, ..., n_{L-1} \to \infty$ (note that (Yang, 2019) extends this to a simultaneous limit) and

$$\int_0^T ||\nabla_{\hat{y}} \mathcal{L}(\hat{y}_t(X))||_2 dt < \infty,$$

the evolution of the deep neural network is similar to that of a linear network, i.e. $\hat{y}_t = \hat{y}_0 + \nabla_\theta \hat{y}_0(\theta_t - \theta_0)$, and the parameter and output evolution during training are given by

$$\partial_t \theta_t = -\eta (\nabla_\theta \hat{y}_0)^T \nabla_{\hat{y}} \mathcal{L}(\hat{y}_t),\ \partial_t \hat{y}_t = -\eta \Theta_0 \nabla_{\hat{y}} \mathcal{L}(\hat{y}_t),$$

which can be solved to give

$$\theta_t = \theta_0 - \nabla_\theta \hat{y}_0(X)^T \Theta_0^{-1} (I - e^{-\frac{\eta}{N}\Theta_0 t})(\hat{y}_0(X) - Y),$$
$$\hat{y}_t(X) = \left(I - e^{-\frac{\eta}{N}\Theta_0 t}\right) Y + e^{-\frac{\eta}{N}\Theta_0 t} \hat{y}_0,$$

where we have used that as $n_1, ..., n_{L-1} \to \infty$ $\nabla_\theta \hat{y}_0(X)(\nabla_\theta \hat{y}_0(X))^T \to \Theta_0$. We will refer to Θ_0 as the neural tangent kernel (NTK) and the scaling in (1) as the NTK scaling.

In previous work, e.g. (Gunasekar et al., 2017), (Neyshabur et al., 2017b), it was shown that gradient descent performs some form of implicit regularization. Due to this, the solution obtained by gradient descent generalizes well, since it can be shown to be the lowest-complexity solution in some notion of complexity. A similar result holds in our setting.

Lemma 1 (Minimum norm solution)**.** *Consider an n^* such that $||\hat{y}^{lin} - \hat{y}|| < \epsilon$ for some small enough ϵ if $n_1, ..., n_{L-1} > n^*$. Gradient descent in deep and wide non-linear networks converges to the minimum norm solution, i.e.*

$$\theta_t \to \arg\min_{\hat{y}_0 + \nabla_\theta \hat{y}_0(\theta - \theta_0) = Y} ||\theta - \theta_0||_2.$$

In other words, the weights that the network converges to when trained with gradient descent are such that their distance to the initial weights is the smallest among all weights that satisfy $\lim_{t\to\infty} \hat{y}_t = Y$. However, since the solution with L_2-norm fits the training data with zero error, if significant amount of noise is present in the target points $y_1, ..., y_N$, the solution will be overly complex and generalize badly.

A metric that is used in deep neural networks to study the generalization and robustness to input noise is the input Jacobian (e.g (Novak et al., 2018)) defined as $J_x = \partial_x \hat{y}_t$. This Jacobian is particularly useful for understanding the generalization capabilities when the network is trained on noisy data. Computing the input Jacobian with respect to a point x^*, we obtain

$$J_x = \hat{y}_0'(x^*) - \Theta_0'(x^*, X)\Theta_0^{-1}(I - e^{-\eta\Theta_0 t})(\hat{y}_0(X) - Y). \quad (2)$$

The two hyperparameters related to the optimization algorithm, η and t, influence the size of this Jacobian, i.e. if t and η are small, the input Jacobian is small resulting in a smoother, more robust solution.

2.2. Gradient descent with stochasticity

As has been mentioned in previous work, the noise in SGD can benefit the generalization capabilities of neural networks. In particular, as observed in e.g. (Jastrzebski et al., 2018), (Jastrzkebski et al., 2017), (Smith & Le, 2018), (Chaudhari & Soatto, 2018), (Park et al.) a relationship exists between the test error and the learning rate and batch size used in the SGD updating scheme. Here we consider the effects of gradient descent with stochasticity for a network in which only the output layer weights are trained and for a deep neural network where the stochasticity is assumed to be in the function space.

2.2.1. Noisy gradient descent for a linear network

Consider a setting in which we only train the parameters of the final layer, i.e. SGD is applied only to $\theta^L \in \mathbb{R}^{n_{L-1}+1}$, a column vector, and all the other parameters $\theta^1, ..., \theta^{L-1}$ are frozen after initialization. The network output is given by $\hat{y}_t(x) = \bar{z}(x)\theta_t^L$, where we have defined for ease of notation $\bar{z}(x) = \left[\frac{\sigma_w z^{L-1}(x)}{\sqrt{n_{L-1}}}, \sigma_b\right]$ such that $\bar{z}(X) \in \mathbb{R}^{N \times (n_{L-1}+1)}$. Note that this model is thus linear in the weights but non-linear in the inputs.

Under the assumption of an isotropic covariance function, the continuous limit of SGD can be written as (Mandt et al., 2017), (Li et al., 2017)

$$d\theta_t^L = -\frac{\eta}{N}\bar{z}(X)^T\left(\bar{z}(X)\theta_t^L - Y\right)dt + \sigma\frac{\eta}{\sqrt{M}}dW_t,$$

where $\sigma \in \mathbb{R}$ and W_t is a n_{L-1}-dimensional Brownian motion. Denote $\Theta_0^{lin} := \bar{z}(X)\bar{z}(X)^T$. The update rule for the output function is then given by

$$d\hat{y}_t = -\frac{\eta}{N}\Theta_0^{lin}\left(\hat{y}_t - Y\right)dt + \bar{z}(X)\sigma\frac{\eta}{\sqrt{M}}dW_t.$$

Note that this is an SDE similar to a multi-variate Ornstein-Uhlenbeck process. This SDE can be solved explicitly to give, for $t > s$,

$$\hat{y}_t(X) = \left(I - e^{-\frac{\eta}{N}\Theta_0^{lin}t}\right)Y + e^{-\frac{\eta}{N}\Theta_0^{lin}t}\hat{y}_0(X) + \int_0^t e^{-\frac{\eta}{N}\Theta_0^{lin}(t-s)}\bar{z}(X)\sigma\frac{\eta}{\sqrt{M}}dW_s.$$

Note that $\mathbb{E}[\hat{y}(X)] = Y$ as $t \to \infty$. Therefore, the addition of isotropic noise does not seem to regularize or smooth the linear model output; the variance however becomes larger as σ increases.

2.2.2. The regularization effects of gradient descent

Consider gradient descent over the MSE loss for the linear network with a regularization term, i.e. $\mathcal{L}(X, \hat{y}_t, Y) = \frac{1}{2N}||\hat{y}(X) - Y||_2^2 + \frac{\lambda}{2}||\theta^L||_2^2$. As observed in the previous section, gradient descent with noise does not regularize the linear network. In this section we want to understand when and how gradient descent *does* result in a smoothed and regularized solution. Applying GD to the loss function we obtain,

$$\theta_{t+1}^L = \theta_t^L - \frac{\eta}{N}\bar{z}(X)^T(\hat{y}_t - Y) - \frac{\eta}{N}\lambda\theta_t^L,$$
$$\hat{y}_{t+1} = \hat{y}_t - \frac{\eta}{N}\bar{z}(X)\bar{z}(X)^T(\hat{y}_t - Y) - \frac{\eta}{N}\lambda\hat{y}_t.$$

Remark 2 (Similarity to noisy training)**.** Note that this is similar to adding $\mathcal{N}(0, \sqrt{\lambda})$-noise to the inputs X during training, since by a derivation similar to (Reed et al., 1992) the noise added to the inputs results in a loss surface where the additional regularization term is given by $\lambda||\theta^L||_2^2$.

Solving the continuous forms of these expressions for $\hat{y}_t(X)$ we obtain,

$$\hat{y}_t(X) = e^{-\frac{\eta}{N}(\Theta_0^{lin}+\lambda)t}\hat{y}_0(X) + \Theta_0^{lin}Y\left(\Theta_0^{lin} + \lambda\right)^{-1}\left(I - e^{-\frac{\eta}{N}(\Theta_0^{lin}+\lambda)t}\right).$$

The convergence is thus slowed down by the regularization coefficient, so that early stopping leads to smoother solutions than the ones without regularization; and at the same time, as $t \to \infty$, the solution does not converge to Y, but to solution with more smoothness (as observed by a smaller Jacobian when λ increases).

2.2.3. Noisy gradient descent for a deep network

Consider the following continuous evolution of the weights: $d\theta_t = -\eta\nabla_\theta\hat{y}_t^T\nabla_{\hat{y}}\mathcal{L}(\hat{y}_t) + \sigma\frac{\eta}{\sqrt{M}}dW_t$. The evolution of the output function, by Itô's lemma is then given by,

$$d\hat{y}_t = -\frac{\eta}{N}\nabla_\theta\hat{y}_t(\nabla_\theta\hat{y}_t)^T(\hat{y}_t - Y)dt + \frac{1}{2}\sigma^2\frac{\eta^2}{M}Tr(\Delta_\theta\hat{y}_t(x))_{x\in\mathcal{X}}dt + \frac{\eta}{\sqrt{M}}\sigma\nabla_\theta\hat{y}_t dW_t.$$

Under certain assumptions on the SGD training dynamics, the neural network output can be approximated by its first-order approximation,$\hat{y}_t \approx \hat{y}_t^{lin} = \hat{y}_0 + \nabla_\theta\hat{y}_0(\theta_t - \theta_0)$. Informally, this holds if the evolution of the original network under SGD does not deviate from the evolution of the linearized network under SGD, which in turn holds if the noise and/or the Hessian are/is sufficiently small. This in turn, by arguments similar to Appendix F in (Lee et al., 2019), holds if

$$\sup_{t\in[0,T]}\left|\left|\Theta_t + \frac{1}{2}\sigma^2\frac{\eta^2}{M}Tr(\Delta_\theta\hat{y}_t(x))_{x\in\mathcal{X}} - \Theta_0\right|\right|_{op} \to 0.$$

Assuming that this convergence holds and using the fact that for the linear model approximation $\Delta_\theta\hat{y}_t = 0$, we obtain using the limiting behavior of the kernels,

$$\hat{y}_t(X) = \left(I - e^{-\frac{\eta}{N}\Theta_0 t}\right)Y + e^{-\frac{\eta}{N}\Theta_0 t}\hat{y}_0(X) - \sigma\frac{\eta}{N}\int_0^t e^{-\frac{\eta}{N}\Theta_0(t-s)}\Theta_0 dW_s.$$

We have the following result,

Lemma 3 (Expected MSE for noisy training)**.** *Consider $n_1, ..., n_{L-1} \to \infty$, so that the deep neural network evolution is governed by the NTK Θ_0. It holds that,*

$$\mathbb{E}\left[||\hat{y}_t - Y||_2^2\right] = \mathbb{E}\left[||e^{-\frac{\eta}{N}\Theta_0 t}(\hat{y}_0 - Y)||_2^2\right] + \sigma^2\frac{\eta^2}{N^2}\int_0^t\sum_{i=1}^N\sum_{j=1}^N\left(\left[e^{-\frac{\eta}{N}\Theta_0(t-s)}\Theta_0\right]_{ij}\right)^2 dt.$$

Proof. The statement follows from using the multi-dimensional Itô Isometry and using the fact that $\hat{y}_0$ and W_t are uncorrelated and $\mathbb{E}[\hat{y}_0] = 0$ and that the expectation of an Itô integral is zero we obtain the statement. □

From Lemma 3 we observe that the stochasticity in noisy gradient descent can result in slower convergence and even in the limit $t \to \infty$ the MSE may not fully converge on the train data. In particular, the larger the noise coefficient σ, the larger the training error will be.

3. Empirical evidence

In this section we present numerical results that validate the theoretical observations made in the previous sections empirically. We consider the function $y_i = \sin(0.1t_i) + c\epsilon_i$, with $c = 0.3$ and $t_i \in \{0, 1, ..., 100\}$ and $\epsilon_i \sim \mathcal{N}(0, 1)$. The network is trained using $t-4, ..., t$ to predict $t+1$ using the MSE loss. The network consists of $L = 5$ layers, with $n_l = 100$ nodes per layer and uses the rectified linear activation function; unless otherwise mentioned we use $n_{its} = 10.000$ training iterations, $\eta = 1$ as a learning rate, $\beta = 0.1$ and $n_b = 100$ as the batch size.

The convergence speed From the input Jacobian in (2), besides the structure of the kernel and the data itself, the training hyperparameters influencing the smoothness of the solution are the learning rate η and the number of iterations t. In Figure 1 we show the effects of these hyperparameters on the noisy sine function. As expected, the function is smoother and generalizes better if t and η are small.

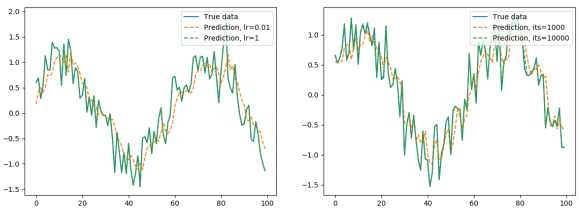

Figure 1. The neural network outputs for different learning rates and numbers of iterations. Using a smaller λ or fewer n_{its} results in a *smoother* solution. (R) with $\eta = 0.01$, the train and test MSEs are 0.13 and 0.13, resp.; using $\eta = 1$ they are $6e10^{-6}$ and 0.20. (L) with $n_{its} = 1000$, the train and test MSEs are 0.09 and 0.14, resp.; using $n_{its} = 10.000$ they are $9e10^{-4}$ and 0.22.

The effects of noise Table 1 shows the train and test MSE for different values of the noise σ. As expected, noise can cause the solution to not fully converge, resulting in slightly better out of sample performance; however too much noise can cause the network to underfit. In Table 2 we show the results of SGD averaged over 20 runs for the NTK scaling (lazy training regime) and regular scaling (here, He initialization) with $\eta = 0.01$. Also here more noise in the *lazy* training regime results in better generalization due to the model not fully converging on the train data. Nevertheless, we observe that for neural networks in the *regular* training regime, i.e. ones in which we do not scale the weights with the NTK scaling, the noise in the SGD has much more regularizing effect.

Table 1. **The performance of gradient descent with added noise.** The results are averaged over 20 runs; the higher the noise, the higher the training error, as expected from Lemma 3. Note that the noise of size $\sigma = 0.5$ has the best performance on the test set.

	Train MSE	Test MSE
$\sigma = 0$	0.0012	0.25
$\sigma = 0.1$	0.0056	0.24
$\sigma = 0.5$	0.025	0.20
$\sigma = 0.9$	0.040	0.22

Table 2. **The performance of stochastic gradient descent with different noise as controlled through the batch size n_b for the lazy and regular neural network scaling.** The results are averaged over 20 runs and are denoted as (train error)-(test error). SGD regularizes the network output so that more noise results in better test performance and a smaller generalization gap. The regularization effects are more significant for the non-lazy training regime.

	$n_b = 1$		$n_b = 100$	
n_l	LAZY	REGULAR	LAZY	REGULAR
2^4	0.08-0.20	0.09-0.18	$1.7e^{-3}$-0.29	0.044-0.21
2^6	0.019-0.23	0.064-0.18	$6.6e^{-4}$-0.26	0.018-0.24
2^8	0.014-0.23	0.056-0.21	$6.0e^{-4}$-0.26	0.012-0.23

4. Conclusion and discussion

In this work we showed results on the generalization capabilities of a neural network using the explicit function evolution as governed by the NTK. We showed how the hyperparameters of the optimization algorithm, here the number of iterations and learning rate, can be adapted to result in outputs that are smoother and generalize better. In previous work it was observed that SGD implicitly regularizes the function output. In the setting studied in this work, under the lazy training regime (obtained using the scaling in (1)), we theoretically derived that noise does not regularize the solution, but can keep the error from fully converging on the train dataset. We showed empirically that SGD in the lazy regime provides some form of regularization, however based on the obtained theoretical result this effect is mostly due to the slower convergence. SGD in the regular setting provides more regularization. It would thus seem SGD has a regularizing effect in the setting in which training is *non-lazy*, i.e. when the output function is *non-linear* in the weights and the weights move significantly compared to their initialized values. It will be of interest to gain more insight into this effect, and understand whether it is indeed the case that regularization effects due to noise in certain variables arise when the network function is *non-linear* in those variables.

References

Arora, S., Ge, R., Neyshabur, B., and Zhang, Y. Stronger generalization bounds for deep nets via a compression approach. *arXiv preprint arXiv:1802.05296*, 2018.

Arora, S., Du, S. S., Hu, W., Li, Z., and Wang, R. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. *arXiv preprint arXiv:1901.08584*, 2019.

Bartlett, P. L., Foster, D. J., and Telgarsky, M. J. Spectrally-normalized margin bounds for neural networks. In *Advances in Neural Information Processing Systems*, pp. 6240–6249, 2017.

Borovykh, A., Oosterlee, C. W., and Bohte, S. M. Generalisation in fully-connected neural networks for time series forecasting. *arXiv preprint arXiv:1902.05312*, 2019.

Chaudhari, P. and Soatto, S. Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks. In *2018 Information Theory and Applications Workshop (ITA)*, pp. 1–10. IEEE, 2018.

Chizat, L. and Bach, F. A note on lazy training in supervised differentiable programming. *arXiv preprint arXiv:1812.07956*, 2018.

Geirhos, R., Temme, C. R., Rauber, J., Schütt, H. H., Bethge, M., and Wichmann, F. A. Generalisation in humans and deep neural networks. In *Advances in Neural Information Processing Systems*, pp. 7549–7561, 2018.

Gunasekar, S., Woodworth, B. E., Bhojanapalli, S., Neyshabur, B., and Srebro, N. Implicit regularization in matrix factorization. In *Advances in Neural Information Processing Systems*, pp. 6151–6159, 2017.

Gunasekar, S., Lee, J. D., Soudry, D., and Srebro, N. Implicit bias of gradient descent on linear convolutional networks. In *Advances in Neural Information Processing Systems*, pp. 9482–9491, 2018.

Hochreiter, S. and Schmidhuber, J. Flat minima. *Neural Computation*, 9(1):1–42, 1997.

Hoffer, E., Hubara, I., and Soudry, D. Train longer, generalize better: closing the generalization gap in large batch training of neural networks. In *Advances in Neural Information Processing Systems*, pp. 1731–1741, 2017.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, pp. 8580–8589, 2018.

Jastrzebski, S., Kenton, Z., Ballas, N., Fischer, A., Bengio, Y., and Storkey, A. DNN's sharpest directions along the sgd trajectory. *arXiv preprint arXiv:1807.05031*, 2018.

Jastrzkebski, S., Kenton, Z., Arpit, D., Ballas, N., Fischer, A., Bengio, Y., and Storkey, A. Three factors influencing minima in sgd. *arXiv preprint arXiv:1711.04623*, 2017.

Lee, J., Bahri, Y., Novak, R., Schoenholz, S. S., Pennington, J., and Sohl-Dickstein, J. Deep neural networks as Gaussian processes. *Submitted to ICML*, 2018.

Lee, J., Xiao, L., Schoenholz, S. S., Bahri, Y., Sohl-Dickstein, J., and Pennington, J. Wide neural networks of any depth evolve as linear models under gradient descent. *arXiv preprint arXiv:1902.06720*, 2019.

Li, J., Luo, X., and Qiao, M. On generalization error bounds of noisy gradient methods for non-convex learning. *arXiv preprint arXiv:1902.00621*, 2019.

Li, Q., Tai, C., et al. Stochastic modified equations and adaptive stochastic gradient algorithms. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 2101–2110. JMLR. org, 2017.

Li, X., Lu, J., Wang, Z., Haupt, J., and Zhao, T. On tighter generalization bound for deep neural networks: Cnns, resnets, and beyond. *arXiv preprint arXiv:1806.05159*, 2018.

Mandt, S., Hoffman, M. D., and Blei, D. M. Stochastic gradient descent as approximate Bayesian inference. *arXiv preprint arXiv:1704.04289*, 2017.

Matthews, A. G., Hron, J., Rowland, M., Turner, R. E., and Ghahramani, Z. Gaussian process behaviour in wide deep neural networks. *ICML*, 2018.

Neal, R. M. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

Neyshabur, B., Tomioka, R., and Srebro, N. Norm-based capacity control in neural networks. In *Conference on Learning Theory*, pp. 1376–1401, 2015.

Neyshabur, B., Bhojanapalli, S., McAllester, D., and Srebro, N. Exploring generalization in deep learning. In *Advances in Neural Information Processing Systems*, pp. 5947–5956, 2017a.

Neyshabur, B., Tomioka, R., Salakhutdinov, R., and Srebro, N. Geometry of optimization and implicit regularization in deep learning. *arXiv preprint arXiv:1705.03071*, 2017b.

Novak, R., Bahri, Y., Abolafia, D. A., Pennington, J., and Sohl-Dickstein, J. Sensitivity and generalization in neural networks: an empirical study. *arXiv preprint arXiv:1802.08760*, 2018.

Park, D. S., Smith, S. L., Sohl-dickstein, J., and Le, Q. V. Optimal sgd hyperparameters for fully connected networks.

Reed, R., Oh, S., and Marks, R. Regularization using jittered training data. In *Neural Networks, 1992. IJCNN., International Joint Conference on*, volume 3, pp. 147–152. IEEE, 1992.

Rotskoff, G. M. and Vanden-Eijnden, E. Neural networks as interacting particle systems: Asymptotic convexity of the loss landscape and universal scaling of the approximation error. *arXiv preprint arXiv:1805.00915*, 2018.

Sagun, L., Evci, U., Guney, V. U., Dauphin, Y., and Bottou, L. Empirical analysis of the hessian of over-parametrized neural networks. *ICML Workshop Track*, 2018.

Seong, S., Lee, Y., Kee, Y., Han, D., and Kim, J. Towards flatter loss surface via nonmonotonic learning rate scheduling. *UAI*, 2018.

Smith, S. L. and Le, Q. V. A Bayesian perspective on generalization and stochastic gradient descent. 2018.

Tishby, N. and Zaslavsky, N. Deep learning and the information bottleneck principle. In *Information Theory Workshop (ITW), 2015 IEEE*, pp. 1–5. IEEE, 2015.

Yang, G. Scaling limits of wide neural networks with weight sharing: Gaussian process behavior, gradient independence, and neural tangent kernel derivation. *arXiv preprint arXiv:1902.04760*, 2019.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.