Other Approaches for Investigating Neural Network Generalization

Thomas Walker

Summer 2023

Contents

1 Stochastic Gradient Descent

The architecture of deep neural networks was proposed long before they manifested as a useful machine learning technique. This delay was due in part to the difficulty in training the large architectures in a stable and effective manner. Learning algorithms such as Stochastic Gradient Descent (SGD) have extracted remarkable properties from deep neural network architectures. Many of the properties are still mysterious to researchers, and these architectures seem to have greater potential than what was previously thought. To try and grapple with this it is important to understand the precise mechanisms of SGD as this has instantiated the networks with the majority of these properties. It has been observed that SGD is able to train these networks in the over-parameterized setting such that they converge to global minima of the loss landscape. In [3] an attempt is made to explain this using dynamic stability. This approach illustrates how the randomness induced by SGD is vital, and why regular gradient descent (GD) is not an effective learning algorithm for training neural networks. As is the case with most machine learning scenarios, one is trying to minimize the training error

$$f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x),$$

where each $f_i(x)$ can be thought of as the loss of the i^{th} example of a training set at the parameter value x. A general optimizer for this problem can be written as

$$x_{t+1} = x_t - G(x_t; \xi_t), \tag{1}$$

where ξ_t is a random variable independent of x_t and each ξ_t are $\mathrm{i.i.d.}$ Note that

- for GD, $G(x_t;\xi_t) = \eta \nabla f_{\xi_t}(x_t)$, and
- for SGD, $G(x_t;\xi_t) = \frac{\eta}{n}\sum_{i=1}^{n} \nabla f_i(x_t)$.

Definition 1.1. *Call x^* a fixed point (1) if for any ξ it follows that $G(x^*, \xi) = 0$.*

Definition 1.2. *Let x^* be a fixed point of (1). For the linearized dynamical system,*

$$\tilde{x}_{t+1} = \tilde{x}_t - A_{\xi_t}(\tilde{x}_t - x^*)$$

where $A_\xi = \nabla_x G(x^*, \xi_t)$. The fixed point x^* is linearly stable if there exists a constant C such that

$$\mathbb{E}\left(\|\tilde{x}_t\|^2\right) \leq C \|\tilde{x}_0\|^2$$

for all $t > 0$.

If it is assume that $f(x^*) = 0$, and the approximation

$$f(x) \approx \frac{1}{2n} \sum_{i=1}^{n} (x - x^*)^\top H_i (x - x^*)$$

with $H_i \nabla^2 f_i(x^*)$, is used then the linearized SGD is given by

$$x_{t+1} = x_t - \frac{\eta}{B} \sum_{j=1}^{B} H_{\xi_j}(x_t - x^*).$$

Where B is the batch size and $\xi = \{\xi_1, \ldots, \xi_B\}$ is a uniform, non-replaceable random sampling of size B on $\{1, \ldots, n\}$.

Definition 1.3. *Let $H = \frac{1}{n}\sum_{i=1}^{n} H_i$ and $\Sigma = \frac{1}{n}\sum_{i=1}^{n} H_i^2 - H^2$. Let $a = \lambda_{\max}(H)$ be the sharpness, and $s = \lambda_{\max}\left(\Sigma^{\frac{1}{2}}\right)$ be the non-uniformity.*

Theorem 1.4. *The global minimum x^* is linearly stable for SGD with learning rate η and batch size B if the following is satisfied*

$$\lambda_{\max}\left((1 - \eta H)^2 + \frac{\eta^2 (n - B)}{B(n - 1)} \Sigma\right) \leq 1.$$

For $d = 1$ this is a necessary and sufficient condition.

Remark 1.5. *A simpler necessary condition is*

$$0 \leq a \leq \frac{2}{\eta}, \text{ and } 0 \leq s \leq \frac{1}{\eta}\sqrt{\frac{B(n-1)}{n-B}}. \tag{2}$$

For a fixed learning rate η,

1. GD can converge to minima satisfying $a \leq \frac{2}{\eta}$, and
2. SGD can converge to minima satisfying (2).

Hence, SGD can filter out minima with large non-uniformity. The difference between GD and SGD is that SGD must converge to solutions that fit the data uniformly well.

2 Rademacher Complexity

It is generally accepted that having a more straightforward function that correctly classifies a dataset is more likely to generalize well to unseen data. The generalization bounds we will discuss will often reward simpler models, where the definition of simple may vary in different contexts. A lot of work in this field tends to be empirical as heuristics are derived for complexity that can be monitored during training. Sometimes theoretically motivated complexity measures are proposed that can be used to derive explicit generalization bounds. An important complexity measure is Rademacher complexity, which forms the basis of one of the main results in generalization theory. Suppose we have a hypothesis class $\mathcal{H}$ (i.e. the possible neural networks defined by a particular set of hyper-parameters), and a training set S. For a loss function l, let $l \circ \mathcal{H} := \{l \circ h : h \in \mathcal{H}\}$.

Definition 2.1 ([1]). *Let $\mathcal{G}$ be a family of functions mapping from $\mathcal{Z}$ to $[a,b]$ and $S=(z_1,\ldots,z_m)$ a fixed sample of size m with elements in $\mathcal{Z}$. Then, the empirical Rademacher complexity of $\mathcal{G}$ with respect to the sample S is defined as:*

$$\mathfrak{R}_S(\mathcal{G}) = \frac{1}{m}\mathbb{E}_{\xi\in\{\pm 1\}^m}\left(\sup_{f\in\mathcal{G}}\sum_{i=1}^{m}\xi_i f(x_i)\right).$$

Definition 2.2 ([1]). *Let $\mathcal{D}$ be a distribution from which samples are drawn. For any integer $m\geq 1$, the Rademacher complexity of a family of functions $\mathcal{G}$ is the expectation of the empirical Rademacher complexity over all samples of size m drawn from $\mathcal{D}$. That is,*

$$\mathfrak{R}_m(\mathcal{G}) = \mathbb{E}_{S\sim\mathcal{D}^m}\left(\hat{\mathfrak{R}}_S(\mathcal{G})\right).$$

Theorem 2.3 ([1]). *Let $\mathcal{G}$ be a family of functions mapping from $\mathcal{Z}$ to $[0,1]$. Then for any $\delta>0$ with probability $1-\delta$ over the draw of an* i.i.d *sample S of size m then,*

$$\mathbb{E}(g(z)) \leq \frac{1}{m}\sum_{i=1}^{m} g(z_i) + 2\mathfrak{R}_m(\mathcal{G}) + \sqrt{\frac{\log\left(\frac{1}{\delta}\right)}{2m}}, \text{ and}$$

$$\mathbb{E}(g(z)) \leq \frac{1}{m}\sum_{i=1}^{m} g(z_i) + 2\hat{\mathfrak{R}}_m(\mathcal{G}) + 3\sqrt{\frac{\log\left(\frac{2}{\delta}\right)}{2m}}.$$

holds for all $g\in\mathcal{G}$.

3 Unit-Wise Capacity Measures

The work of [2] looks at two-layer fully connected ReLU networks. The inputs have dimension d, outputs have dimension c, and layers have h hidden units. The function of the network is represented as $f_{\mathbf{V},\mathbf{U}}(\mathbf{x}) = \mathbf{V}(\mathbf{U}\mathbf{x})_+$ where $\mathbf{x}\in\mathbb{R}^d, \mathbf{U}\in\mathbb{R}^{h\times d}$ and $\mathbf{V}\in\mathbb{R}^{c\times h}$. With $\mathbf{u}_i$ denoting the incoming weights to hidden unit i and $\mathbf{v}_i$, the outgoing weights to hidden unit i. The network is initialized with weights $\mathbf{U}_0$ and $\mathbf{V}_0$ where $\mathbf{u}_i^0$ and $\mathbf{v}_i^0$ denote the corresponding weights as defined above. The network will be used to perform c-class classification, where the maximum output of a score function gives the predicted label. Define this score function to be the margin operator $\mu:\mathbb{R}^c\times[c]\to\mathbb{R}$ which scores an output $f(\mathbf{x})$ for each label $y\in[c]$ according to $\mu(f(\mathbf{x}),y) = f(\mathbf{x})[y] - \max_{i\neq y} f(\mathbf{x})[i]$. To train the network we use the ramp loss,

$$l_\gamma(f(\mathbf{x}),y) = \begin{cases} 0 & \mu(f(\mathbf{x}),y) > \gamma \\ \frac{\mu(f(\mathbf{x}),y)}{\gamma} & \mu(f(\mathbf{x}),y)\in[0,\gamma] \\ 1 & \mu(f(\mathbf{x}),y) < 0. \end{cases}$$

Hence, the following definitions of error emerge,

- $L_\gamma(f) = \mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}}(l_\gamma(f(\mathbf{x},y))$, the expected margin loss of a predictor $f(\cdot)$, for distribution $\mathcal{D}$ and margin $\gamma>0$.
- $\hat{L}_\gamma(f)$, the empirical margin loss.
- $L_0(f)$, the expected risk.
- $\hat{L}_0(f)$, the expected training error.

For a hypothesis class $\mathcal{H}$, a training set S and with $l_\gamma\circ\mathcal{H} := \{l_\gamma\circ h : h\in\mathcal{H}\}$ Theorem 2.3 can be applied to obtain.

Theorem 3.1. *With probability* $1-\delta$ *over a training set of size* m*,*

$$L_0(f) \leq \hat{L}_\gamma(f) + 2\mathfrak{R}_S(l_\gamma \circ \mathcal{H}) + 3\sqrt{\frac{\log\left(\frac{2}{\delta}\right)}{2m}}$$

holds for any $f \in \mathcal{H}$*.*

Definition 3.2. *The unique capacity of a hidden unit* i *is* $\beta_i = \left\|\mathbf{u}_i - \mathbf{u}_i^0\right\|_2$*.*

Definition 3.3. *The unit impact of a hidden unit* i *is* $\alpha_i = \|\mathbf{v}_i\|_2$*.*

Definition 3.4. *Let* $\mathcal{W}$ *be the restricted set of parameters*

$$\mathcal{W} = \left\{(\mathbf{V}, \mathbf{U}) : \mathbf{V} \in \mathbb{R}^{c\times h}, \mathbf{U} \in \mathbb{R}^{h\times d}, \|\mathbf{v}_i\| \leq \alpha_i, \left\|\mathbf{u}_i - \mathbf{u}_i^0\right\|_2 \leq \beta_i\right\}$$

and let $\mathcal{F}_\mathcal{W}$ *be the corresponding class of neural networks*

$$\mathcal{F}_\mathcal{W} = \{f(\mathbf{x}) = \mathbf{V}(\mathbf{U}\mathbf{x})_+ : (\mathbf{V}, \mathbf{U}) \in \mathcal{W}\}$$

Theorem 3.5. *Given a training set* $S = \{\mathbf{x}_i\}_{i=1}^m$ *and* $\gamma > 0$*, then*

$$\begin{aligned}\mathfrak{R}_S(l_\gamma \circ \mathcal{F}_\mathcal{W}) &\leq \frac{2\sqrt{2c}+2}{\gamma m}\sum_{j=1}^h \alpha_j\left(\beta_j\|\mathbf{X}\|_F + \left\|\mathbf{u}_j^0\mathbf{X}\right\|_2\right)\\ &\leq \frac{2\sqrt{2c}+2}{\gamma m}\|\alpha\|_2\left(\|\beta\|_2\sqrt{\frac{1}{m}\sum_{i=1}^m\|\mathbf{x}_i\|_2^2} + \sqrt{\frac{1}{m}\sum_{i=1}^m\|\mathbf{U}^0\mathbf{x}_i\|_2^2}\right).\end{aligned}$$

Theorem 3.6. *For any* $h \geq 2, \gamma > 0, \delta \in (0,1)$ *and* $\mathbf{U}^0 \in \mathbb{R}^{h\times d}$ *with probability* $1-\delta$ *over the training set* $S = \{\mathbf{x}_i\}_{i=1}^m \subset \mathbb{R}^d$*, for any function* $f(\mathbf{x}) = \mathbf{V}(\mathbf{U}\mathbf{x})_+$ *such that* $\mathbf{V} \in \mathbb{R}^{c\times h}$ *and* $\mathbf{U} \in \mathbb{R}^{h\times d}$*,*

$$\begin{aligned}L_0(f) &\leq \hat{L}_\gamma(f) + \tilde{O}\left(\frac{\sqrt{c}\|\mathbf{V}\|_F\left(\left\|\mathbf{U}-\mathbf{U}^0\right\|_F\|\mathbf{X}\|_F + \left\|\mathbf{U}^0\mathbf{X}\right\|_F\right)}{\gamma m} + \sqrt{\frac{h}{m}}\right)\\ &\leq \hat{L}_\gamma(f) + \tilde{O}\left(\frac{\sqrt{c}\|\mathbf{V}\|_F\left(\left\|\mathbf{U}-\mathbf{U}^0\right\|_F + \left\|\mathbf{U}^0\right\|_2\right)\sqrt{\frac{1}{m}\sum_{i=1}^m\|\mathbf{x}_i\|_2^2}}{\gamma\sqrt{m}} + \sqrt{\frac{h}{m}}\right)\end{aligned}$$

Therefore, the term $\|\mathbf{V}\|_F\left(\left\|\mathbf{U}-\mathbf{U}^0\right\|_F + \left\|\mathbf{U}^0\right\|_2\right) + \sqrt{h}$ can be used as a heuristic for complexity.

4 Validation Paradigm

The work of [4] looks into how the training-validation paradigm leads to deep neural networks that generalize well. In this paradigm, a validation set of data is held out to optimize the model architecture and hyper-parameters. Giving rise to the hypothesis that *deep neural networks can obtain good generalization error by performing a model search on the validation set*.
Consider an input $x \in \mathcal{X}$ and a label $y \in \mathcal{Y}$. The loss function is denoted $\mathcal{L}$ and $\mathcal{R}[f] = \mathbb{E}_{x,y\sim\mathbb{P}_{(\mathcal{X},\mathcal{Y})}}\left(\mathcal{L}(f(x),y)\right)$ is the expected risk of a function f for $\mathbb{P}_{(\mathcal{X},\mathcal{Y})}$ being the true distribution. Let $f_{\mathcal{A}(S)} : \mathcal{X} \to \mathcal{Y}$ denote the function learnt by a learning algorithm $\mathcal{A}$ on training set $S := \{(x_1,y_1),\ldots,(x_m,y_m)\}$. The set of possible learned functions is characterized by the hypothesis space $\mathcal{F}$. Associated with this space we have the family of loss functions $\mathcal{L}_{\mathcal{F}} := \{g : g \in \mathcal{F}, g(x,y) = \mathcal{L}(f(x),y)\}$. Machine learning aims to minimize $\mathcal{R}\left(f_{\mathcal{A}(S)}\right)$. However, this is non-computable as $\mathbb{P}_{(\mathcal{X},\mathcal{Y})}$ is unknown. Therefore, one minimizes the empirical risk

$$\mathcal{R}_S\left(f_{\mathcal{A}(S)}\right) = \frac{1}{|S|}\sum_{(x,y)\in S}\mathcal{L}(f_{\mathcal{A}(S)}(x),y),$$

where the generalization gap is defined to be $\mathcal{R}\left(f_{\mathcal{A}(S)}\right) - \mathcal{R}_S\left(f_{\mathcal{A}(S)}\right)$.

Proposition 4.1. *Let $S^{(\mathrm{val})}_{m_{\mathrm{val}}}$ be a held-out validation set, where $\left|S^{(\mathrm{val})}_{m_{\mathrm{val}}}\right| = m_{\mathrm{val}}$. Assume that m_{val} is an* i.i.d *sample from $\mathbb{P}_{(X,Y)}$. Let $\kappa_{f,i} = \mathcal{R}(f) - \mathcal{L}(f(x_i),y_i)$ for $(x_i,y_i) \in S^{(\mathrm{val})}_{m_{\mathrm{val}}}$. Suppose that $\mathbb{E}\left(\kappa^2_{f,i}\right) \le \gamma^2$ and $|\kappa_{f,i}| \le C$ almost surely for all $(f,i) \in \mathcal{F}_{\mathrm{val}} \times \{1,\ldots,m_{\mathrm{val}}\}$. Then, for $\delta \in (0,1]$, with probability $1-\delta$*

$$\mathcal{R}(f) \le \mathcal{R}_{S^{(\mathrm{val})}_{m_{\mathrm{val}}}}(f) + \frac{2C\log\left(\frac{|\mathcal{F}_{\mathrm{val}}|}{\delta}\right)}{3m_{\mathrm{val}}} + \sqrt{\frac{2\gamma^2\log\left(\frac{|\mathcal{F}_{\mathrm{val}}|}{\delta}\right)}{m_{\mathrm{val}}}}$$

holds for all $f \in \mathcal{F}_{\mathrm{val}}$.

Remark 4.2.

- *$\mathcal{F}_{\mathrm{val}}$ is independent of $S^{(\mathrm{val})}_{m_{\mathrm{val}}}$.*
- *The bound is only dependent on the validation error on $S^{(\mathrm{val})}_{m_{\mathrm{val}}}$.*

The dependence on $|\mathcal{F}_{\mathrm{val}}|$ can be alleviated in the following corollary of Theorem 2.3.

Corollary 4.3. *Assume $S^{(\mathrm{val})}_{m_{\mathrm{val}}}$ is an* i.i.d *sample from $\mathbb{P}_{(\mathcal{X},\mathcal{Y})}$. Let $\mathcal{L}_{\mathcal{F}_{\mathrm{val}}} = \{g : f \in \mathcal{F}_{\mathrm{val}}, g(x,y) := \mathcal{L}(f(x),y)\}$. Then when $\mathcal{L}$ has co-domain $[0,1]$ it follows that,*

$$\mathcal{R}(f) \le \mathcal{R}_{S^{(\mathrm{val})}_{m_{\mathrm{val}}}}(f) + 2\mathfrak{R}_m\left(\mathcal{L}_{\mathcal{F}_{\mathrm{val}}}\right) + \sqrt{\frac{\log\left(\frac{1}{\delta}\right)}{m_{\mathrm{val}}}}.$$

Some code for implementing this paradigm can be found here.

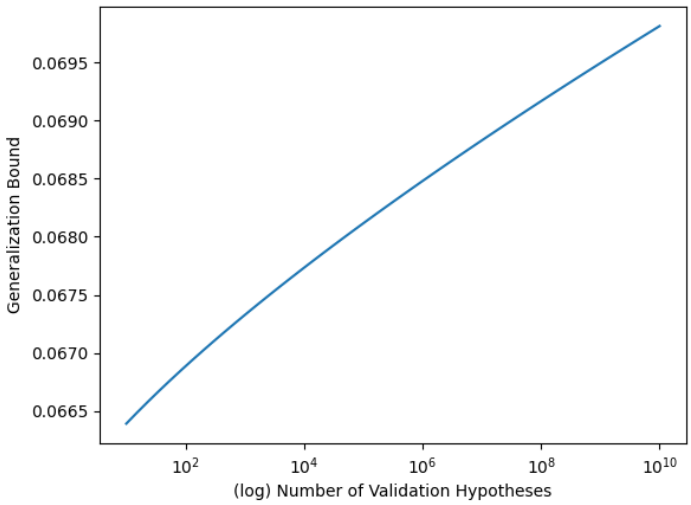

Figure 1: Scaling of generalization error bound with the number of models used for validation. Models used for validation are those that achieve low training loss.

References

[1] M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of Machine Learning*. Adaptive Computation and Machine Learning. MIT Press, 2018.

[2] Behnam Neyshabur, Zhiyuan Li, Srinadh Bhojanapalli, Yann LeCun, and Nathan Srebro. "Towards Understanding the Role of Over-Parametrization in Generalization of Neural Networks". In: *CoRR* (2018).

[3] Lei Wu, Chao Ma, and Weinan E. "How SGD Selects the Global Minima in Over-parameterized Learning: A Dynamical Stability Perspective". In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett. Vol. 31. Curran Associates, Inc., 2018.

[4] K. Kawaguchi, Y. Bengio, and L. Kaelbling. "Generalization in Deep Learning". In: *Mathematical Aspects of Deep Learning*. Cambridge University Press, 2022, pp. 112–148.