Chapter 15

Generalization in the overparameterized regime

In the previous chapter, we discussed the theory of generalization for deep neural networks trained by minimizing the empirical risk. A key conclusion was that good generalization is possible as long as we choose an architecture that has a moderate number of network parameters relative to the number of training samples. Moreover, we saw in Section 14.6 that the best performance can be expected when the neural network size is chosen to balance the generalization and approximation errors, by minimizing their sum.

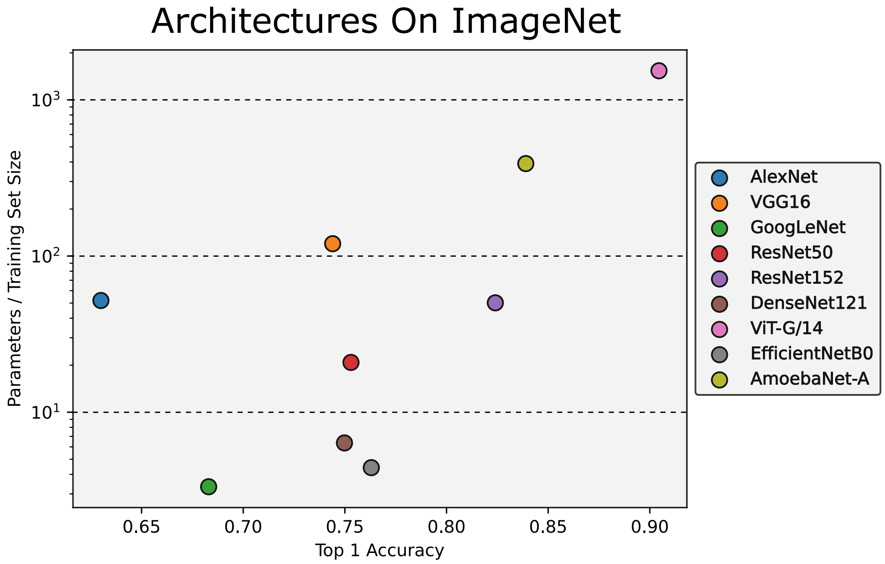


Figure 15.1: ImageNet Classification Competition: Final score on the test set in the Top 1 category vs. Parameters-to-Training-Samples Ratio. Note that all architectures have more parameters than training samples. Architectures include AlexNet [119], VGG16 [214], GoogLeNet [221], ResNet50/ResNet152 [86], DenseNet121 [95], ViT-G/14 [247], EfficientNetB0 [223], and AmoebaNet [187].

Surprisingly, successful network architectures do not necessarily follow these theoretical observations. Consider the neural network architectures in Figure 15.1. They represent some of the most renowned image classification models, and all of them participated in the ImageNet Classification Competition [49]. The training set consisted of 1.2 million images. The x-axis shows the model performance, and the y-axis displays the ratio of the number of parameters to the size of the training set; notably, all architectures have a ratio larger than one, i.e. have more parameters than training samples. For the largest model, there are by a factor 1000 more network parameters than training samples.

Given that the practical application of deep learning appears to operate in a regime significantly different from the one analyzed in Chapter 14, we must ask: Why do these methods still work effectively?

15.1 The double descent phenomenon

The success of deep learning in a regime not covered by traditional statistical learning theory puzzled researchers for some time. In [14], an intriguing set of experiments was performed. These experiments indicate that while the risk follows the upper bound from Section 14.6 for neural network architectures that do not interpolate the data, the curve does not expand to infinity in the way that Figure 14.4 suggests. Instead, after surpassing the so-called “interpolation threshold”, the risk starts to decrease again. This behavior, known as double descent, is illustrated in Figure 15.2.

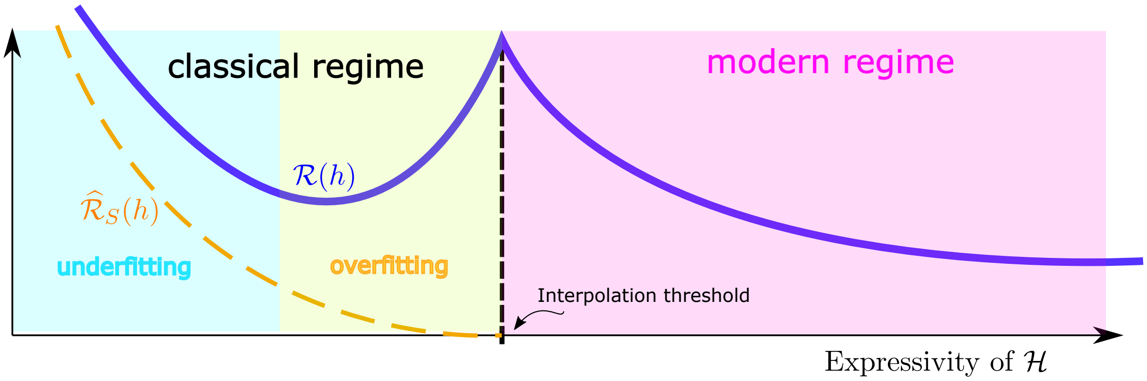


Figure 15.2: Illustration of the double descent phenomenon.

15.1.1 Least-squares regression revisited

To gain further insight, we consider least-squares (kernel) regression as introduced in Section 11.2. Consider a data sample $(\boldsymbol{x}_j, y_j)_{j=1}^m \subseteq \mathbb{R}^d \times \mathbb{R}$ generated by some ground-truth function f, i.e.

$$y_j = f(\boldsymbol{x}_j) \qquad \text{for } j = 1, \dots, m. \tag{15.1.1}$$

Let $\phi_j : \mathbb{R}^d \to \mathbb{R}$, $j \in \mathbb{N}$, be a sequence of *ansatz functions*. For $n \in \mathbb{N}$, we wish to fit a function $\boldsymbol{x} \mapsto \sum_{i=1}^n w_i \phi_i(\boldsymbol{x})$ to the data using linear least-squares. To this end, we introduce the feature map

$$\mathbb{R}^d \ni \boldsymbol{x} \mapsto \phi(\boldsymbol{x}) := (\phi_1(\boldsymbol{x}), \dots, \phi_n(\boldsymbol{x}))^\top \in \mathbb{R}^n.$$

The goal is to determine coefficients $\boldsymbol{w} \in \mathbb{R}^n$ minimizing the empirical risk

$$\widehat{\mathcal{R}}_S(\boldsymbol{w}) = \frac{1}{m}\sum_{j=1}^{m}\Big(\sum_{i=1}^{n} w_i\phi_i(\boldsymbol{x}_j) - y_j\Big)^2 = \frac{1}{m}\sum_{j=1}^{m}(\langle\phi(\boldsymbol{x}_j), \boldsymbol{w}\rangle - y_j)^2.$$

With

$$\boldsymbol{A}_n := \begin{pmatrix} \phi_1(\boldsymbol{x}_1) & \dots & \phi_n(\boldsymbol{x}_1) \\ \vdots & \ddots & \vdots \\ \phi_1(\boldsymbol{x}_m) & \dots & \phi_n(\boldsymbol{x}_m) \end{pmatrix} = \begin{pmatrix} \phi(\boldsymbol{x}_1)^\top \\ \vdots \\ \phi(\boldsymbol{x}_m)^\top \end{pmatrix} \in \mathbb{R}^{m\times n} \tag{15.1.2}$$

and $\boldsymbol{y} = (y_1, \dots, y_m)^\top$ it holds

$$\widehat{\mathcal{R}}_S(\boldsymbol{w}) = \frac{1}{m}\|\boldsymbol{A}_n\boldsymbol{w} - \boldsymbol{y}\|^2. \tag{15.1.3}$$

As discussed in Sections 11.1-11.2, a unique minimizer of (15.1.3) only exists if $\boldsymbol{A}_n$ has rank n. For a minimizer $\boldsymbol{w}_n$, the fitted function reads

$$f_n(x) := \sum_{j=1}^{n} w_{n,j}\phi_j(x). \tag{15.1.4}$$

We are interested in the behavior of the f_n as a function of n (the number of ansatz functions/parameters of our model), and distinguish between two cases:

- *Underparameterized*: If $n < m$ we have fewer parameters n than training points m. For the least squares problem of minimizing $\widehat{\mathcal{R}}_S$, this means that there are more conditions m than free parameters n. Thus, in general, we cannot interpolate the data, and we have $\min_{\boldsymbol{w}\in\mathbb{R}^n} \widehat{\mathcal{R}}_S(\boldsymbol{w}) > 0$.

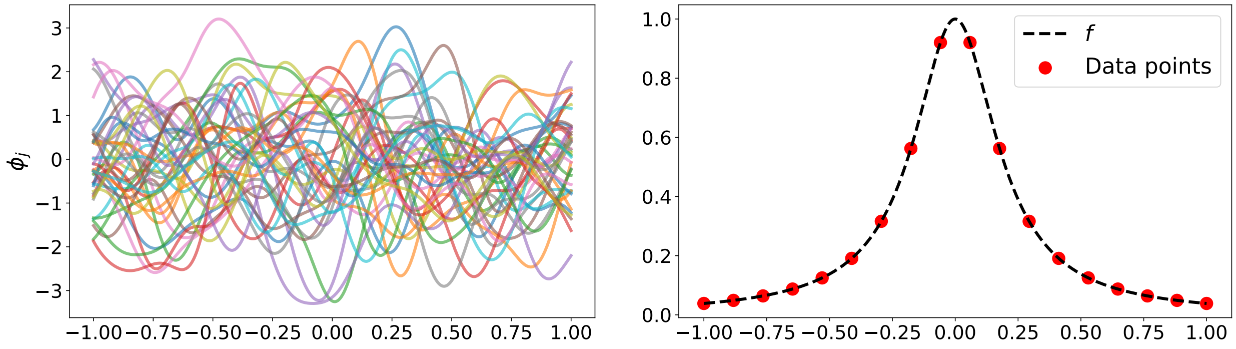
- *Overparameterized*: If $n \geq m$, then we have at least as many parameters n as training points m. If the $\boldsymbol{x}_j$ and the ϕ_j are such that $\boldsymbol{A}_n \in \mathbb{R}^{m\times n}$ has full rank m, then there exists $\boldsymbol{w}$ such that $\widehat{\mathcal{R}}_S(\boldsymbol{w}) = 0$. If $n > m$, then $\boldsymbol{A}_n$ necessarily has a nontrivial kernel, and there exist infinitely many parameters choices $\boldsymbol{w}$ that yield zero empirical risk $\widehat{\mathcal{R}}_S$. Some of them lead to better, and some lead to worse prediction functions f_n in (15.1.4).

In the overparameterized case, there exist many minimizers of $\widehat{\mathcal{R}}_S$. The training algorithm we use to compute a minimizer determines the type of prediction function f_n we obtain. To observe double descent, i.e. to achieve good generalization for large n, we need to choose the minimizer carefully. In the following, we consider the unique minimal 2-norm minimizer, which is defined as

$$\boldsymbol{w}_{n,*} = \Big(\operatorname{argmin}_{\{\boldsymbol{w}\in\mathbb{R}^n \,|\, \widehat{\mathcal{R}}_S(\boldsymbol{w}) \leq \widehat{\mathcal{R}}_S(\boldsymbol{v})\ \forall \boldsymbol{v}\in\mathbb{R}^n\}} \|\boldsymbol{w}\|\Big) \in \mathbb{R}^n. \tag{15.1.5}$$

15.1.2 An example

Now let us consider a concrete example. In Figure 15.3 we plot a set of 40 ansatz functions $\phi_1, \dots, \phi_{40}$, which are drawn from a Gaussian process. Additionally, the figure shows a plot of the Runge function f, and $m = 18$ equispaced points which are used as the training data points. We then fit a function in $\operatorname{span}\{\phi_1, \dots, \phi_n\}$ via (15.1.5) and (15.1.4). The result is displayed in Figure 15.4:



(a) ansatz functions ϕ_j

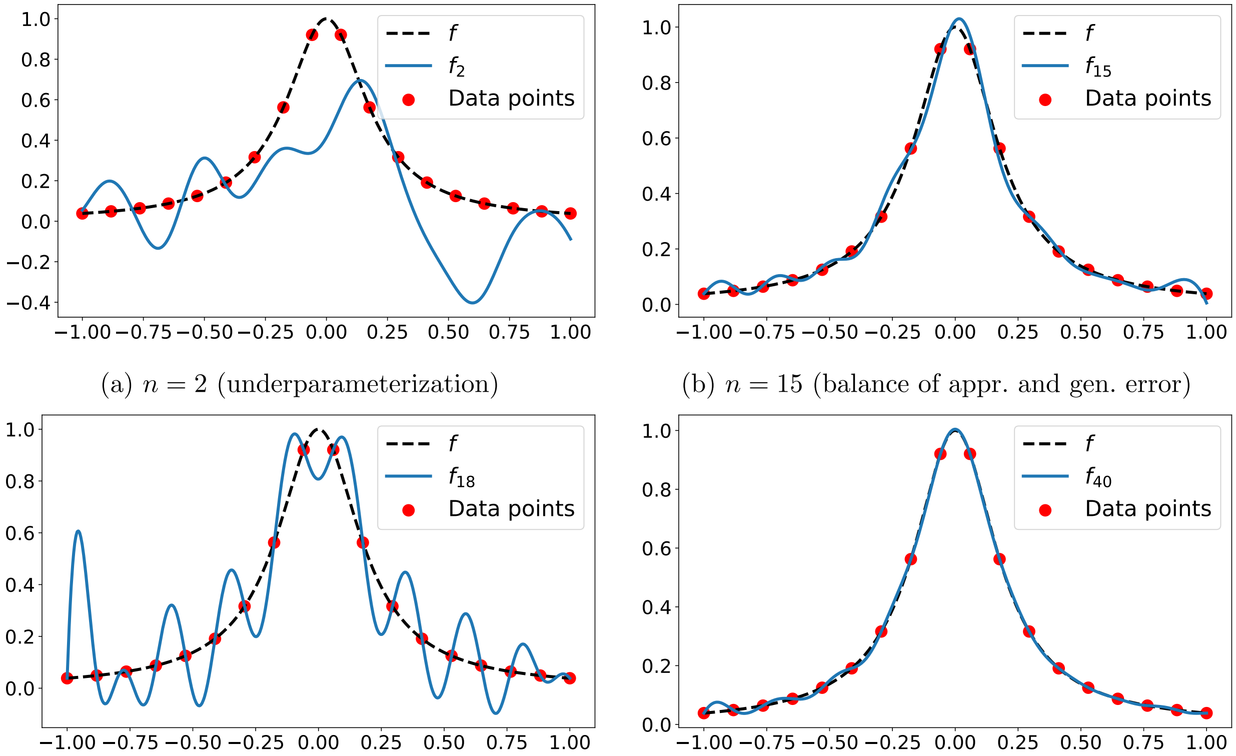
(b) Runge function f and data points

Figure 15.3: Ansatz functions $\phi_1, \dots, \phi_{40}$ drawn from a Gaussian process, along with the Runge function and 18 equispaced data points.

- $n = 2$: The model can only represent functions in $\operatorname{span}\{\phi_1, \phi_2\}$. It is not yet expressive enough to give a meaningful approximation of f.
- $n = 15$: The model has sufficient expressivity to capture the main characteristics of f. Since $n = 15 < 18 = m$, it is not yet able to interpolate the data. Thus it allows to strike a good balanced between the approximation and generalization error, which corresponds to the scenario discussed in Chapter 14.
- $n = 18$: We are at the interpolation threshold. The model is capable of interpolating the data, and there is a unique $\boldsymbol{w}$ such that $\widehat{\mathcal{R}}_S(\boldsymbol{w}) = 0$. Yet, in between data points the behavior of the predictor f_{18} seems erratic, and displays strong oscillations. This is referred to as **overfitting**, and is to be expected due to our analysis in Chapter 14; while the approximation error at the data points has improved compared to the case $n = 15$, the generalization error has gotten worse.
- $n = 40$: This is the overparameterized regime, where we have significantly more parameters than data points. Our prediction f_{40} interpolates the data and appears to be the best overall approximation to f so far, due to a "good" choice of minimizer of $\widehat{\mathcal{R}}_S$, namely (15.1.5). We also note that, while quite good, the fit is not perfect. We cannot expect significant improvement in performance by further increasing n, since at this point the main limiting factor is the amount of available data. Also see Figure 15.5 (a).

Figure 15.5 (a) displays the error $\|f - f_n\|_{L^2([-1,1])}$ over n. We observe the characteristic double descent curve, where the error initially decreases, after peaking at the interpolation threshold, which is marked by the dashed red line. Afterwards, in the overparameterized regime, it starts to decrease again. Figure 15.5 (b) displays $\|\boldsymbol{w}_{n,*}\|$. Note how the Euclidean norm of the coefficient vector also peaks at the interpolation threshold.

We emphasize that the precise nature of the convergence curves depends strongly on various factors, such as the distribution and number of training points m, the ground truth f, and the choice of ansatz functions ϕ_j (e.g., the specific kernel used to generate the ϕ_j in Figure 15.3 (a)). In the present setting we achieve a good approximation of f for $n = 15 < 18 = m$ corresponding to the regime where the approximation and interpolation errors are balanced. However, as Figure 15.5



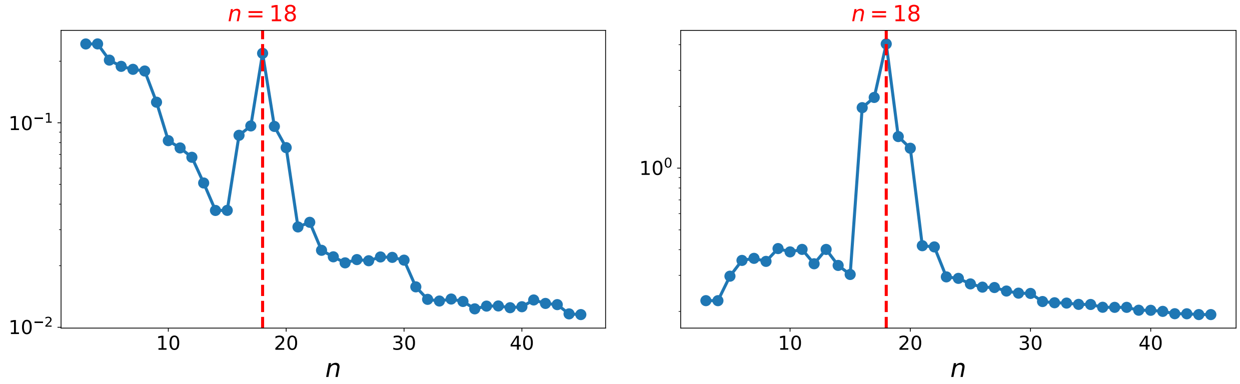
(a) $n = 2$ (underparameterization)

(b) $n = 15$ (balance of appr. and gen. error)

(c) $n = 18$ (interpolation threshold)

(d) $n = 40$ (overparameterization)

Figure 15.4: Fit of the $m = 18$ red data points using the ansatz functions $\phi_1, \dots, \phi_n$ from Figure 15.3, employing equations (15.1.5) and (15.1.4) for different numbers of ansatz functions n.



(a) $\|f - f_n\|_{L^2([-1,1])}$

(b) $\|\boldsymbol{w}_{n,*}\|$

Figure 15.5: The L^2-error for the fitted functions in Figure 15.4, and the ℓ^2-norm of the corresponding coefficient vector $\boldsymbol{w}_{n,*}$ defined in (15.1.5).

(a) shows, it can be difficult to determine a suitable value of $n < m$ a priori, and the acceptable range of n values can be quite narrow. For overparametrization ($n \gg m$), the precise choice of n is less critical, potentially making the algorithm more stable in this regime. We encourage the reader to conduct similar experiments and explore different settings to get a better feeling for the double descent phenomenon.

15.2 Size of weights

In Figure 15.5, we observed that the norm of the coefficients $\|\boldsymbol{w}_{n,*}\|$ exhibits similar behavior to the L^2-error, peaking at the interpolation threshold $n = 18$. In machine learning, large weights are usually undesirable, as they are associated with large derivatives or oscillatory behavior. This is evident in the example shown in Figure 15.4 for $n = 18$. Assuming that the data in (15.1.1) was generated by a "smooth" function f, e.g. a function with moderate Lipschitz constant, these large derivatives of the prediction function may lead to poor generalization. It is important to note that such a smoothness assumption about f may or may not be satisfied. However, if f is not smooth, there is little hope of accurately recovering f from limited data (see the discussion in Section 9.2).

The next result gives an explanation for the observed behavior of $\|\boldsymbol{w}_{n,*}\|$.

Proposition 15.1. *Assume that $\boldsymbol{x}_1, \dots, \boldsymbol{x}_m$ and the $(\phi_j)_{j\in\mathbb{N}}$ are such that $\boldsymbol{A}_n$ in* (15.1.2) *has full rank n for all $n \le m$. Given $\boldsymbol{y} \in \mathbb{R}^m$, denote by $\boldsymbol{w}_{n,*}(\boldsymbol{y})$ the vector in* (15.1.5)*. Then*

$$n \mapsto \sup_{\|\boldsymbol{y}\|=1} \|\boldsymbol{w}_{n,*}(\boldsymbol{y})\| \quad \textit{is monotonically} \quad \begin{cases} \textit{increasing} & \textit{for } n < m, \\ \textit{decreasing} & \textit{for } n \ge m. \end{cases}$$

Proof. We start with the case $n \ge m$. By assumption $\boldsymbol{A}_m$ has full rank m, and thus $\boldsymbol{A}_n$ has rank m for all $n \ge m$, see (15.1.2). In particular, there exists $\boldsymbol{w}_n \in \mathbb{R}^n$ such that $\boldsymbol{A}_n \boldsymbol{w}_n = \boldsymbol{y}$. Now fix

$\boldsymbol{y} \in \mathbb{R}^m$ and let $\boldsymbol{w}_n$ be any such vector. Then $\boldsymbol{w}_{n+1} := (\boldsymbol{w}_n, 0) \in \mathbb{R}^{n+1}$ satisfies $\boldsymbol{A}_{n+1}\boldsymbol{w}_{n+1} = \boldsymbol{y}$ and $\|\boldsymbol{w}_{n+1}\| = \|\boldsymbol{w}_n\|$. Thus necessarily $\|\boldsymbol{w}_{n+1,*}\| \leq \|\boldsymbol{w}_{n,*}\|$ for the minimal norm solutions defined in (15.1.5). Since this holds for every $\boldsymbol{y}$, we obtain the statement for $n \geq m$.

Now let $n < m$. Recall that the minimal norm solution can be written through the pseudo inverse

$$\boldsymbol{w}_{n,*}(\boldsymbol{y}) = \boldsymbol{A}_n^\dagger \boldsymbol{y},$$

see for instance Exercise 11.32. Here,

$$\boldsymbol{A}_n^\dagger = \boldsymbol{V}_n \begin{pmatrix} \sigma_{n,1}^{-1} & & & 0 & \\ & \ddots & & & \ddots & \\ & & \sigma_{n,n}^{-1} & & & 0 \end{pmatrix} \boldsymbol{U}_n^\top \in \mathbb{R}^{n \times m}$$

where $\boldsymbol{A}_n = \boldsymbol{U}_n \boldsymbol{\Sigma}_n \boldsymbol{V}_n^\top$ is the singular value decomposition of $\boldsymbol{A}_n$, and

$$\boldsymbol{\Sigma}_n = \begin{pmatrix} \sigma_{n,1} & & \\ & \ddots & \\ & & \sigma_{n,n} \\ 0 & & \\ & \ddots & \\ & & 0 \end{pmatrix} \in \mathbb{R}^{m \times n}$$

contains the singular values $\sigma_{n,1} \geq \cdots \geq \sigma_{n,n} > 0$ of $\boldsymbol{A}_n \in \mathbb{R}^{m \times n}$ ordered by decreasing size. Since $\boldsymbol{V}_n \in \mathbb{R}^{n \times n}$ and $\boldsymbol{U}_n \in \mathbb{R}^{m \times m}$ are orthogonal matrices, we have

$$\sup_{\|\boldsymbol{y}\|=1} \|\boldsymbol{w}_{n,*}(\boldsymbol{y})\| = \sup_{\|\boldsymbol{y}\|=1} \|\boldsymbol{A}_n^\dagger \boldsymbol{y}\| = \sigma_{n,n}^{-1}.$$

Finally, since the minimal singular value $\sigma_{n,n}$ of $\boldsymbol{A}_n$ can be written as

$$\sigma_{n,n} = \inf_{\substack{\boldsymbol{x} \in \mathbb{R}^n \\ \|\boldsymbol{x}\|=1}} \|\boldsymbol{A}_n \boldsymbol{x}\| \geq \inf_{\substack{\boldsymbol{x} \in \mathbb{R}^{n+1} \\ \|\boldsymbol{x}\|=1}} \|\boldsymbol{A}_{n+1} \boldsymbol{x}\| = \sigma_{n+1,n+1},$$

we observe that $n \mapsto \sigma_{n,n}$ is monotonically decreasing for $n \leq m$. This concludes the proof. □

15.3 Theoretical justification

Let us now examine one possible explanation of the double descent phenomenon for neural networks. While there are many alternative arguments available in the literature (see the bibliography section), the explanation presented here is based on a simplification of the ideas in [12].

The key assumption underlying our analysis is that large overparameterized neural networks tend to be Lipschitz continuous with a Lipschitz constant independent of the size. This is a consequence of neural networks typically having relatively small weights. To motivate this, let us consider the class of neural networks $\mathcal{N}(\sigma; \mathcal{A}, B)$ for an architecture $\mathcal{A}$ of depth $d \in \mathbb{N}$ and width $L \in \mathbb{N}$. If σ is C_σ-Lipschitz continuous such that $B \leq c_B \cdot (dC_\sigma)^{-1}$ for some $c_B > 0$, then by Lemma 13.2

$$\mathcal{N}(\sigma; \mathcal{A}, B) \subseteq \mathrm{Lip}(c_B^L). \tag{15.3.1}$$

An assumption of the type $B \leq c_B \cdot (dC_\sigma)^{-1}$, i.e. a scaling of the weights by the reciprocal $1/d$ of the width, is not unreasonable in practice: Standard initialization schemes such as LeCun [127] or He [85] initialization, use random weights with variance scaled inverse proportional to the input dimension of each layer. Moreover, as we saw in Chapter 11, for very wide neural networks, the weights do not move significantly from their initialization during training. Additionally, many training routines use regularization terms on the weights, thereby encouraging them the optimization routine to find small weights.

We study the generalization capacity of Lipschitz functions through the covering-number-based learning results of Chapter 14. The set of C-Lipschitz functions on a compact d-dimensional Euclidean domain $\mathrm{Lip}(C)$ has covering numbers bounded according to

$$\log(\mathcal{G}(\mathrm{Lip}(C), \varepsilon, L^\infty)) \leq C_{\mathrm{cov}} \cdot \left(\frac{C}{\varepsilon}\right)^d \qquad \text{for all } \varepsilon > 0 \tag{15.3.2}$$

for some constant C_{cov} independent of $\varepsilon > 0$. A proof can be found in [74, Lemma 7], see also [229].

As a result of these considerations, we can identify two regimes:

- *Standard regime:* For small network size $n_\mathcal{A}$, we consider neural networks as a set parameterized by $n_\mathcal{A}$ parameters. As we have seen before, this yields a bound on the generalization error that scales linearly with $n_\mathcal{A}$. As long as $n_\mathcal{A}$ is small in comparison to the number of samples, we can expect good generalization by Theorem 14.15.
- *Overparameterized regime:* For large network size $n_\mathcal{A}$ but small weights as described before, we consider neural networks as a subset of $\mathrm{Lip}(C)$ for a constant $C > 0$. This set has a covering number bound that is independent of the number of parameters $n_\mathcal{A}$.

Choosing the better of the two generalization bounds for each regime yields the following result. Recall that $\mathcal{N}^*(\sigma; \mathcal{A}, B)$ denotes all networks in $\mathcal{N}(\sigma; \mathcal{A}, B)$ with a range contained in $[-1, 1]$ (see (14.5.1)).

Theorem 15.2. *Let C, $C_\mathcal{L} > 0$ and let $\mathcal{L} \colon [-1,1] \times [-1,1] \to \mathbb{R}$ be $C_\mathcal{L}$-Lipschitz. Further, let $\mathcal{A} = (d_0, d_1, \dots, d_{L+1}) \in \mathbb{N}^{L+2}$, let $\sigma \colon \mathbb{R} \to \mathbb{R}$ be C_σ-Lipschitz continuous with $C_\sigma \geq 1$, and $|\sigma(x)| \leq C_\sigma |x|$ for all $x \in \mathbb{R}$, and let $B > 0$.*

Then, there exist c_1, $c_2 > 0$, such that for every $m \in \mathbb{N}$ and every distribution $\mathcal{D}$ on $[-1,1]^{d_0} \times [-1,1]$ it holds with probability at least $1 - \delta$ over $S \sim \mathcal{D}^m$ that for all $\Phi \in \mathcal{N}^*(\sigma; \mathcal{A}, B) \cap \mathrm{Lip}(C)$

$$|\mathcal{R}(\Phi) - \widehat{\mathcal{R}}_S(\Phi)| \leq g(\mathcal{A}, C_\sigma, B, m) + 4C_\mathcal{L}\sqrt{\frac{\log(4/\delta)}{m}}, \tag{15.3.3}$$

where

$$g(\mathcal{A}, C_\sigma, B, m) = \min\left\{c_1\sqrt{\frac{n_\mathcal{A}\log(n_\mathcal{A}\lceil\sqrt{m}\rceil) + Ln_\mathcal{A}\log(d_{\max})}{m}}, c_2 m^{-\frac{1}{2+d_0}}\right\}.$$

Proof. Applying Theorem 14.11 with $\alpha = 1/(2+d_0)$ and (15.3.2), we obtain that with probability at least $1-\delta/2$ it holds for all $\Phi \in \mathrm{Lip}(C)$

$$\begin{aligned}
|\mathcal{R}(\Phi) - \widehat{\mathcal{R}}_S(\Phi)| &\le 4C_{\mathcal{L}}\sqrt{\frac{C_{\mathrm{cov}}(m^\alpha C)^{d_0} + \log(4/\delta)}{m}} + \frac{2C_{\mathcal{L}}}{m^\alpha} \\
&\le 4C_{\mathcal{L}}\sqrt{C_{\mathrm{cov}}C^{d_0}(m^{d_0/(d_0+2)-1})} + \frac{2C_{\mathcal{L}}}{m^\alpha} + 4C_{\mathcal{L}}\sqrt{\frac{\log(4/\delta)}{m}} \\
&= 4C_{\mathcal{L}}\sqrt{C_{\mathrm{cov}}C^{d_0}(m^{-2/(d_0+2)})} + \frac{2C_{\mathcal{L}}}{m^\alpha} + 4C_{\mathcal{L}}\sqrt{\frac{\log(4/\delta)}{m}} \\
&= \frac{(4C_{\mathcal{L}}\sqrt{C_{\mathrm{cov}}C^{d_0}} + 2C_{\mathcal{L}})}{m^\alpha} + 4C_{\mathcal{L}}\sqrt{\frac{\log(4/\delta)}{m}},
\end{aligned}$$

where we used in the second inequality that $\sqrt{x+y} \le \sqrt{x} + \sqrt{y}$ for all x, $y \ge 0$.

In addition, Theorem 14.15 yields that with probability at least $1-\delta/2$ it holds for all $\Phi \in \mathcal{N}^*(\sigma; \mathcal{A}, B)$

$$\begin{aligned}
|\mathcal{R}(\Phi) - \widehat{\mathcal{R}}_S(\Phi)| &\le 4C_{\mathcal{L}}\sqrt{\frac{n_{\mathcal{A}}\log(\lceil n_{\mathcal{A}}\sqrt{m}\rceil) + Ln_{\mathcal{A}}\log(\lceil 2C_\sigma B d_{\max}\rceil) + \log(4/\delta)}{m}} \\
&\quad + \frac{2C_{\mathcal{L}}}{\sqrt{m}} \\
&\le 6C_{\mathcal{L}}\sqrt{\frac{n_{\mathcal{A}}\log(\lceil n_{\mathcal{A}}\sqrt{m}\rceil) + Ln_{\mathcal{A}}\log(\lceil 2C_\sigma B d_{\max}\rceil)}{m}} \\
&\quad + 4C_{\mathcal{L}}\sqrt{\frac{\log(4/\delta))}{m}}.
\end{aligned}$$

Then, for $\Phi \in \mathcal{N}^*(\sigma; \mathcal{A}, B) \cap \mathrm{Lip}(C)$ the minimum of both upper bounds holds with probability at least $1-\delta$. □

The two regimes in Theorem 15.2 correspond to the two terms comprising the minimum in the definition of $g(\mathcal{A}, C_\sigma, B, m)$. The first term increases with $n_{\mathcal{A}}$ while the second is constant. In the first regime, where the first term is smaller, the generalization gap $|\mathcal{R}(\Phi) - \widehat{\mathcal{R}}_S(\Phi)|$ increases with $n_{\mathcal{A}}$.

In the second regime, where the second term is smaller, the generalization gap is constant with $n_{\mathcal{A}}$. Moreover, it is reasonable to assume that the empirical risk $\widehat{\mathcal{R}}_S$ will decrease with increasing number of parameters $n_{\mathcal{A}}$.

By (15.3.3) we can bound the risk by

$$\mathcal{R}(\Phi) \le \widehat{\mathcal{R}}_S + g(\mathcal{A}, C_\sigma, B, m) + 4C_{\mathcal{L}}\sqrt{\frac{\log(4/\delta)}{m}}.$$

In the second regime, this upper bound is monotonically decreasing. In the first regime it may both decrease and increase. In some cases, this behavior can lead to an upper bound on the risk resembling the curve of Figure 15.2. The following section describes a specific scenario where this is the case.

Remark 15.3. Theorem 15.2 assumes C-Lipschitz continuity of the neural networks. As we saw in Sections 15.1.2 and 15.2, this assumption may not hold near the interpolation threshold. Hence, Theorem 15.2 likely gives a too optimistic upper bound near the interpolation threshold.

15.4 Double descent for neural network learning

Now let us understand the double descent phenomenon in the context of Theorem 15.2. We make a couple of simplifying assumptions to obtain a formula for an upper bound on the risk. First, we assume that the data $S = (\boldsymbol{x}_i, y_i)_{i=1}^m \in \mathbb{R}^{d_0} \times \mathbb{R}$ stem from a C_M-Lipschitz continuous function. In addition, we fix a depth $L \in \mathbb{N}$ and consider, for $d \in \mathbb{N}$, architectures of the form $(\sigma_{\mathrm{ReLU}}; \mathcal{A}_d)$, where

$$\mathcal{A}_d = (d_0, d, \dots, d, 1).$$

For this architecture the number of parameters is bounded by

$$n_{\mathcal{A}_d} = (d_0 + 1)d + (L-1)(d+1)d + d + 1.$$

To derive an upper bound on the risk, we start by upper bounding the empirical risk and then applying Theorem 15.2 to establish an upper bound on the generalization gap. In combination, these estimates provide an upper bound on the risk. We will then observe that this upper bound follows the double descent curve in Figure 15.2.

15.4.1 Upper bound on empirical risk

We establish an upper bound on $\widehat{\mathcal{R}}_S(\Phi)$ for $\Phi \in \mathcal{N}^*(\sigma_{\mathrm{ReLU}}; \mathcal{A}_d, B) \cap \mathrm{Lip}(C_M)$. For $B \geq C_M$, we can apply Theorem 9.6, and conclude that with a neural network of sufficient depth we can interpolate m points from a C_M-Lipschitz function with a neural network in $\mathrm{Lip}(C_M)$, if $n_{\mathcal{A}} \geq c_{\mathrm{int}} \log(m) d_0 m$. To simplify the exposition, we assume $c_{\mathrm{int}} = 1$ in the following.

Thus, $\widehat{\mathcal{R}}_S(\Phi) = 0$ as soon as $n_{\mathcal{A}} \geq \log(m) d_0 m$.

In addition, depending on smoothness properties of the data, the interpolation error may decay with some rate, by one of the results in Chapters 5, 7, or 8. For simplicity, we choose that $\widehat{\mathcal{R}}_S(\Phi) = O(n_{\mathcal{A}}^{-1})$ for $n_{\mathcal{A}}$ significantly smaller than $\log(m) d_0 m$. If we combine these two assumptions, we can make the following Ansatz for the empirical risk of $\Phi_{\mathcal{A}_d} \in \mathcal{N}^*(\sigma_{\mathrm{ReLU}}; \mathcal{A}_d, B) \cap \mathrm{Lip}(C_M)$:

$$\widehat{\mathcal{R}}_S(\Phi_{\mathcal{A}_d}) \leq \widetilde{\mathcal{R}}_S(\Phi_{\mathcal{A}_d}) \coloneqq C_{\mathrm{approx}} \max\left\{0, n_{\mathcal{A}_d}^{-1} - (\log(m) d_0 m)^{-1}\right\} \tag{15.4.1}$$

for a constant $C_{\mathrm{approx}} > 0$. Note that, we can interpolate the sample S already with $d_0 m$ parameters by Theorem 9.3. However, it is not guaranteed that this can be done using C_M-Lipschitz neural networks.

15.4.2 Upper bound on generalization gap

We complement the bound on the empirical risk by an upper bound on the risk. Invoking the notation of Theorem 15.2, we have that,

$$g(\mathcal{A}_d, C_{\sigma_{\mathrm{ReLU}}}, B, m) = \min\left\{\kappa_{\mathrm{NN}}(\mathcal{A}_d, m; c_1), \kappa_{\mathrm{Lip}}(\mathcal{A}_d, m; c_2)\right\},$$

where

$$\begin{aligned} \kappa_{\mathrm{NN}}(\mathcal{A}_d, m; c_1) &\coloneqq c_1 \sqrt{\frac{n_{\mathcal{A}_d} \log(\lceil n_{\mathcal{A}} \sqrt{m} \rceil) + L n_{\mathcal{A}_d} \log(d)}{m}}, \\ \kappa_{\mathrm{Lip}}(\mathcal{A}_d, m; c_2) &\coloneqq c_2 m^{-\frac{1}{2+d_0}} \end{aligned} \tag{15.4.2}$$

for some constants c_1, $c_2 > 0$.

15.4.3 Upper bound on risk

Next, we combine (15.4.1) and (15.4.2) to obtain an upper bound on the risk $\mathcal{R}(\Phi_{\mathcal{A}_d})$. Specifically, we define

$$\begin{aligned} \widetilde{\mathcal{R}}(\Phi_{\mathcal{A}_d}) := \widetilde{\mathcal{R}}_S(\Phi_{\mathcal{A}_d}) + \min\{\kappa_{\mathrm{NN}}(\mathcal{A}_d, m; c_1), \kappa_{\mathrm{Lip}}(\mathcal{A}_d, m; c_2)\} \\ + 4C_{\mathcal{L}}\sqrt{\frac{\log(4/\delta)}{m}}. \end{aligned} \tag{15.4.3}$$

We depict in Figure 15.6 the upper bound on the risk given by (15.4.3) (excluding the terms that do not depend on the architecture). The upper bound clearly resembles the double descent phenomenon of Figure 15.2. Note that the Lipschitz interpolation point is slightly behind this threshold, which is when we assume our empirical risk to be 0. To produce the plot, we chose $L = 5$, $c_1 = 1.2 \cdot 10^{-4}$, $c_2 = 6.5 \cdot 10^{-3}$, $m = 10.000$, $d_0 = 6$, $C_{\mathrm{approx}} = 30$. We mention that the double descent phenomenon is not visible for all choices of parameters. Moreover, in our model, the fact that the peak coincides with the interpolation threshold is due to the choice of constants and does not emerge from the model. Other models of double descent explain the location of the peak more accurately [141, 82]. We note that, as observed in Remark 15.3, the peak close to the interpolation threshold that we see in Figure 15.6 would likely be more pronounced in practical scenarios.

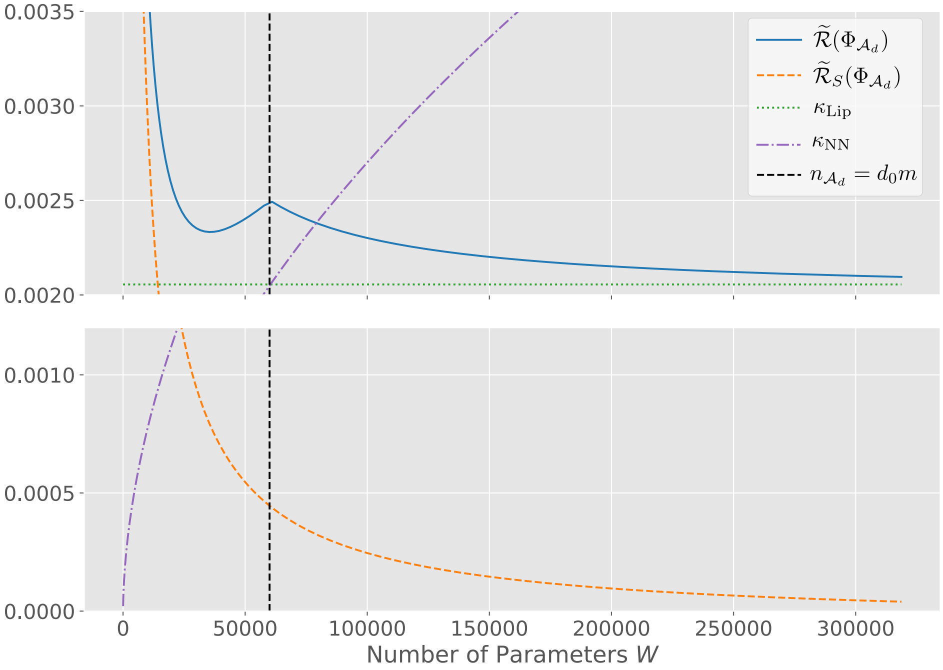


Figure 15.6: Upper bound on $\mathcal{R}(\Phi_{\mathcal{A}_d})$ derived in (15.4.3). For better visibility the part corresponding to y-values between 0.0012 and 0.0022 is not shown. The vertical dashed line indicates the interpolation threshold according to Theorem 9.3.

Bibliography and further reading

The discussion on kernel regression and the effect of the number of parameters on the norm of the weights was already given in [14]. Similar analyses, with more complex ansatz systems and more precise asymptotic estimates, are found in [141, 82]. Our results in Section 15.3 are inspired by [12]; see also [159].

For a detailed account of further arguments justifying the surprisingly good generalization capabilities of overparameterized neural networks, we refer to [19, Section 2]. Here, we only briefly mention two additional directions of inquiry. First, if the learning algorithm introduces a form of robustness, this can be leveraged to yield generalization bounds [6, 243, 24, 177]. Second, for very overparameterized neural networks, it was stipulated in [105] that neural networks become linear kernel interpolators based on the neural tangent kernel of Section 11.5.2. Thus, for large neural networks, generalization can be studied through kernel regression [105, 129, 15, 133].