Double Descent

A novel view on model's complexity and generalizability

Zion Sheng

Department of ECE
Duke University

November 21, 2023

Table of Contents

A Quick Review on the Classic Bias-variance Trade-off

What is the learning task?

Given some training examples $(\mathbf{x_1}, y_1), ..., (\mathbf{x_n}, y_n)$ where $\mathbf{x_i}$ is a d-dimention vector of features and y_i is a value (for regression) or a label (for classification), we want to learn a **model** $h_n : \mathbb{R}^d \to \mathbb{R}$ such that it can predict the output value or label for any input $\mathbf{x}$ as accurate as possible.

The model h_n is commonly chosen from some function class $\mathcal{H}$. For example, $\mathcal{H}$ can be a class of generalized linear model derived by empirical risk minimization (ERM). In ERM, the model is taken to be a function $h \in \mathcal{H}$ that minimizes the **empirical risk** (or **training loss**) $\frac{1}{n}\sum_{i=1}^{n} L(h_n(\mathbf{x}_i), y_i)$, where L is the loss function. Below are some example models from the same model class $\mathcal{H}$, trying to learn the pattern in the training data.

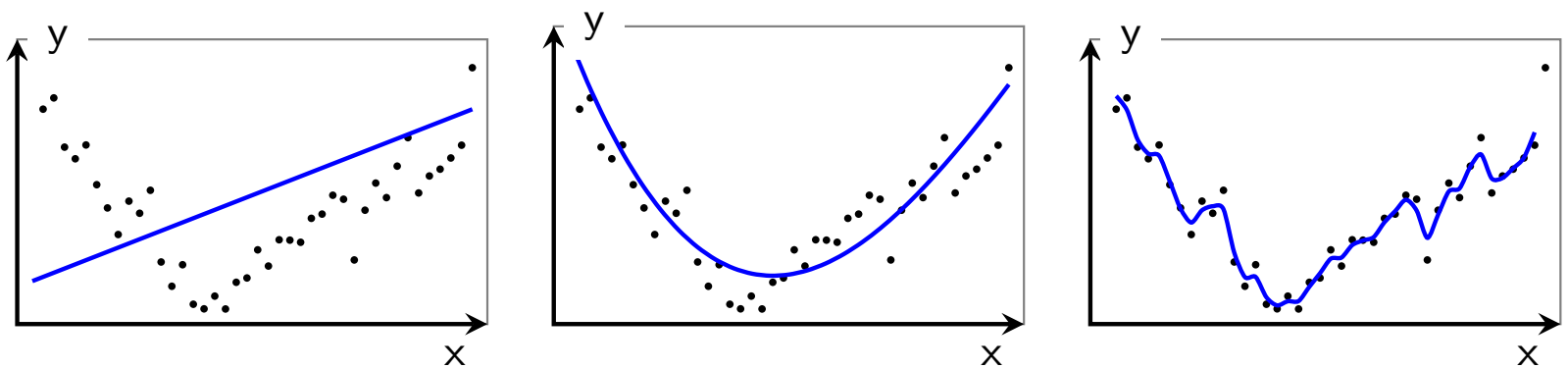


Figure 1: Models with different complexity for a simple linear regression learning

From left to right, as the model becomes more complicated, it tends to fit the training data better, i.e., a smaller training loss. But what about their inference performance on unseen data?

A Quick Review on the Classic Bias-variance Trade-off

Bias-variance trade-off

A good model should have a good **generalizability**, meaning that it can predict accurately on new data, unseen in training. To study the performance on new data, we typically assume training examples $(\mathbf{x}_i, y_i)$ are sampled randomly from a probability distribution P over $\mathbb{R}^d \times \mathbb{R}$ and test examples are also drawn independently from P.

The challenge comes from the mis-match between the goals of minimizing the empirical risk and minimizing the **true risk** (or **test error**) $\mathbb{E}_{(\mathbf{x},y)\sim P}[L(h(\mathbf{x}), y)]$. Suppose the true model is g and the random noise is ϵ. Recall that the test error can be decomposed into:

$$\mathbb{E}_{(\mathbf{x},y)\sim P}[L(h(\mathbf{x}), y)] = (\mathbb{E}(h(\mathbf{x})) - g(\mathbf{x}))^2 + \mathrm{Var}(h(\mathbf{x})) + \mathrm{Var}(\epsilon) \tag{1}$$

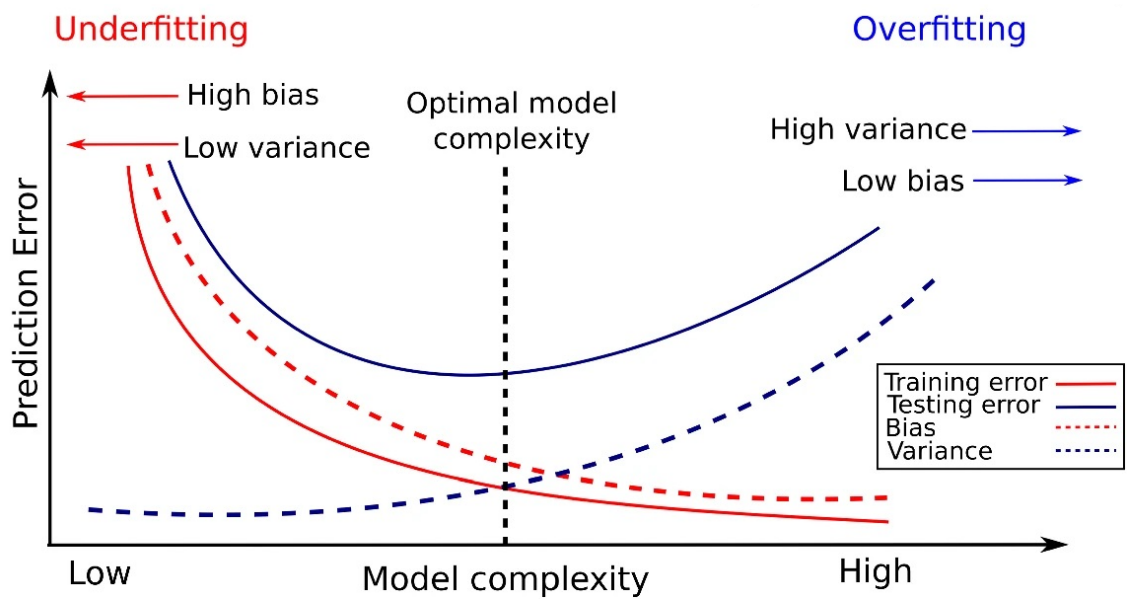


Figure 2: The visualization of bias-variance trade-off (Source: Tharwat, 2019 [7])

Introduction to Double Descent

Background

In modern machine learning, large models, such as deep neural networks and other non-nonlinear models, have become mainstream. These models usually have a huge amount of parameters, and sometimes even more than the number of samples. We call this **overparameterization**. Despite these models tend to achieve near-zero training loss, it turns out that they can still give very accurate predictions during testing.

Double descent

Double descent is the phenomenon that: as the model complexity increases, the test risk of trained models first decreases and then increases (the standard U-shape), and then decreases again. The peak in test risk occurs in the "under-parameterized stage" when the models are just barely able to fit the training set. The second descent occurs in the "overparameterized stage" when the model capacity is large enough to **interpolate** the training data.

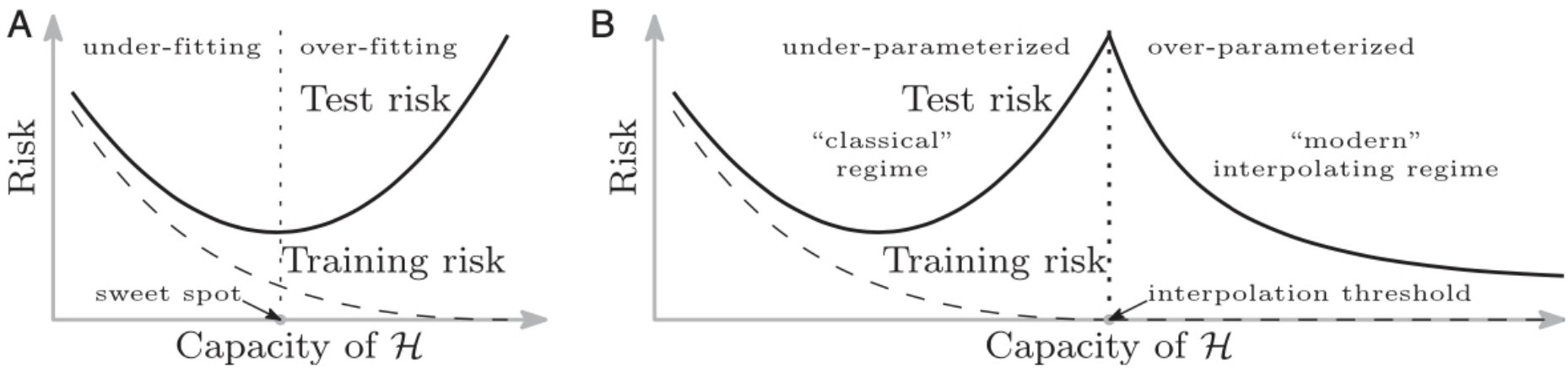


Figure 3: The visualization of double descent (Source: Belkin et al., 2019 [2])

Introduction to Double Descent

Why does double descent happen?

Intuitively, ...

- Belkin et al., 2019 [2]
  They believe that, the capacity of the function class does not necessarily reflect how well the predictor matches the inductive bias for the problem at hand. Here, the inductive bias that seems appropriate is the **regularity** or **smoothness** of a function, which can be measured by a certain function space norm (we don't know, but it exists). By considering larger function classes containing more compatible candidate models, we are able to find interpolating functions that have smaller norm. Thus, increasing function class capacity improves performance of models.
- Nakkiran et al., 2021 [6]
  They define a new complexity measure called the **effective model complexity** and conjecture a generalized double descent with respect to this measure.

Alternative explanations proposed by some physicists, ...

- Baldassi et al. (2021) [1] & Canatar et al. (2021) [3]
  They use methods in statistical physics to investigate the occurrence of double descents in DNN learning. Baldassi et al. posit that the structure of solution space is changed before and after the interpolation threshold. Canatar et al. argue that there could be multiple descents and the model is learning different patterns with different sets of features in each stage. Fundamentally, the switch between these stages is analogous to the **phase transition** in physics.

A Simple Experiment to See Double Descent

Experiment setup

- Ground-truth distribution P is $(\mathbf{x}, y) \in \mathbb{R}^d \times \mathbb{R}$ with covariates $\mathbf{x} \sim \mathcal{N}(0, I_d)$ and response $y = \mathbf{x}^\intercal \beta + \epsilon$ where the noise term $\epsilon \in \mathcal{N}(0, \sigma^2)$.
- We are given n samples $(\mathbf{x_i}, y_i)$ from the distribution P, and we want to learn a linear model $h_n = \mathbf{x}^\intercal \hat{\beta}$ for estimating y given x, which can minimize the test error $R(\hat{\beta})$. Since the features $\mathbf{x}$ is **isotropic**, the test error can be expressed as:

$$R(\hat{\beta}) = \mathbb{E}_{(\mathbf{x},y)\sim P}[(\mathbf{x}^\intercal \hat{\beta} - y)^2] = ||\hat{\beta} - \beta||^2 + \sigma^2 \tag{2}$$

- To train this model, we will start by zero-initialization and run full-batch graident descent on following learning objective:

$$\min_{\hat{\beta}} \frac{1}{n} ||\mathbf{X}\hat{\beta} - \mathbf{y}||^2 \tag{3}$$

The solution found by gradient descent is $\hat{\beta} = \mathbf{X}^\dagger \mathbf{y}$, where $\mathbf{X}^\dagger$ denotes the pseudoinverse. $\mathbf{X}^\dagger$ has different values depending on the ratio n/d. When $n \geq d$, we are in the "under-parameterized" regime and there is a unique minimizer of the objective in Equation (3). When $n < d$, we are "overparameterized" and there are many minimizers of Equation (3). In this regime, gradient descent finds the minimum with smallest $\mathcal{L}_2$ norm $||\beta||^2$. That is, the solution can be written as:

$$\hat{\beta} = \mathbf{X}^\dagger \mathbf{y} = \begin{cases} \arg\min_{\hat{\beta}} ||\hat{\beta}||^2 \text{ s.t. } \mathbf{X}\hat{\hat{\beta}} = \mathbf{y} & \text{when } n < d \\ \arg\min_{\hat{\beta}} ||\mathbf{X}\hat{\beta} - \mathbf{y}||^2 & \text{when } n \geq d \end{cases} \tag{4}$$

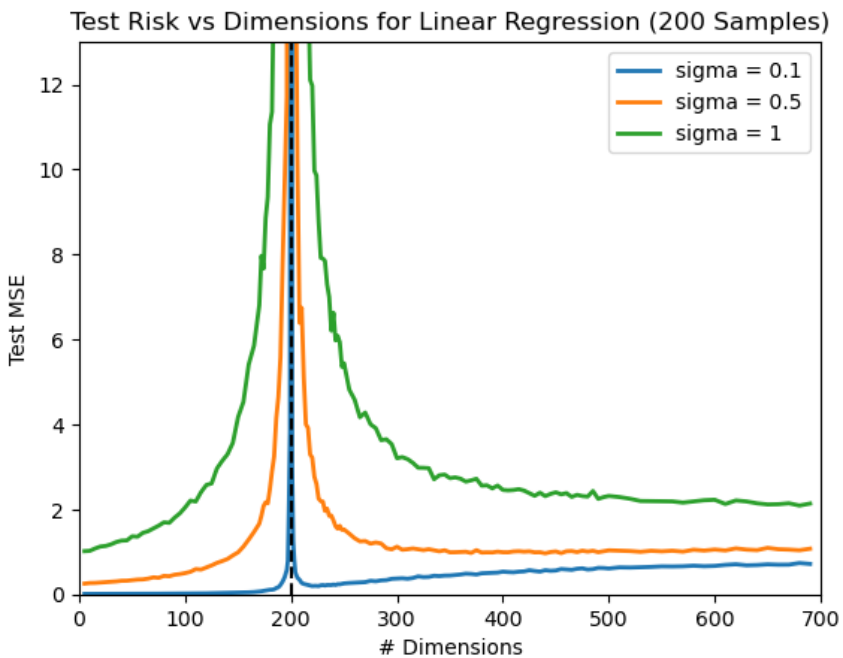
A Simple Experiment to See Double Descent

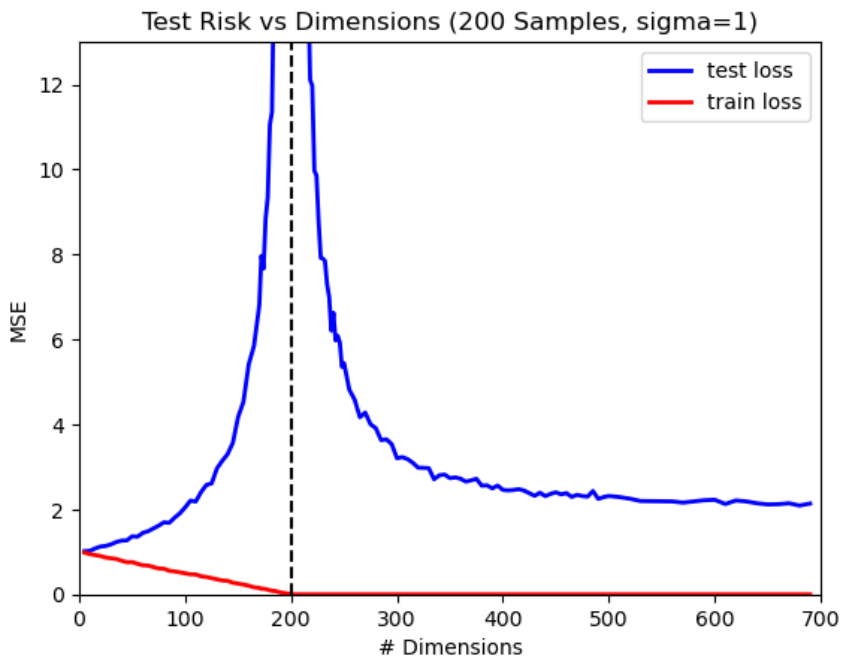
Analytical results

According to Nakkiran (2019) [5] and Hastie et al. (2022) [4], let's denote the ratio of n/d as γ, and the test error in underparameterized and overparameterized regime can be expressed as:

$$R(\gamma) = \begin{cases} \sigma^2 \dfrac{\gamma}{1-\gamma} & \text{when } \gamma < 1 \\ r^2(1 - \dfrac{1}{\gamma}) + \sigma^2 \dfrac{1}{1-\gamma} & \text{when } \gamma > 1 \end{cases} \tag{5}$$

Results on synthetic data





Why Does This Matter?

Since we have seen that double descent can take place even on linear regression learning tasks, it shouldn't be a big surprise to reproduce it in more complicated models such as DNN, kernel machines, and transformers. Now the question is: why does this matter?

- Theoratically justify the development of complicated models, as it suggests that increasing model complexity may not always be detrimental, and there can be a regime where larger models can perform better.
- Double descent has spurred research into understanding the theoretical underpinnings of this phenomenon. It has led to new insights into the geometry of high-dimensional spaces and the optimization landscape of overparameterized models.

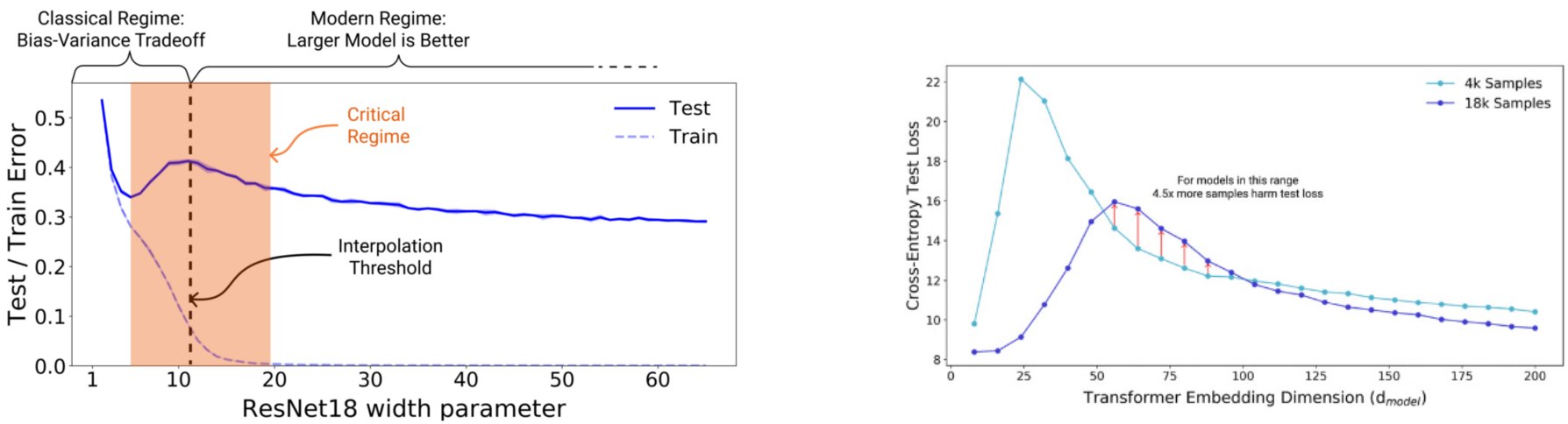
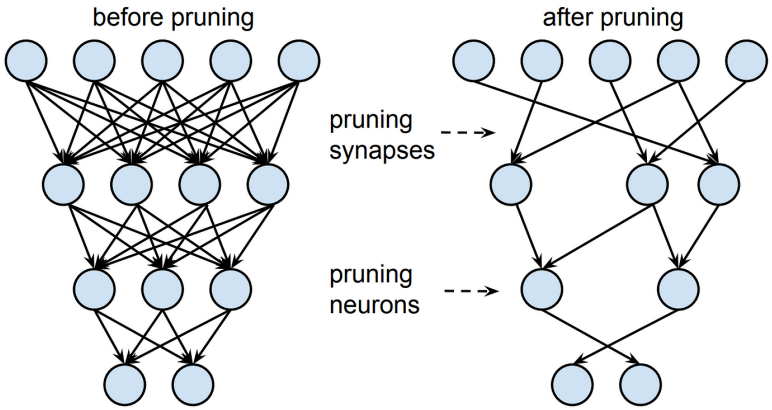


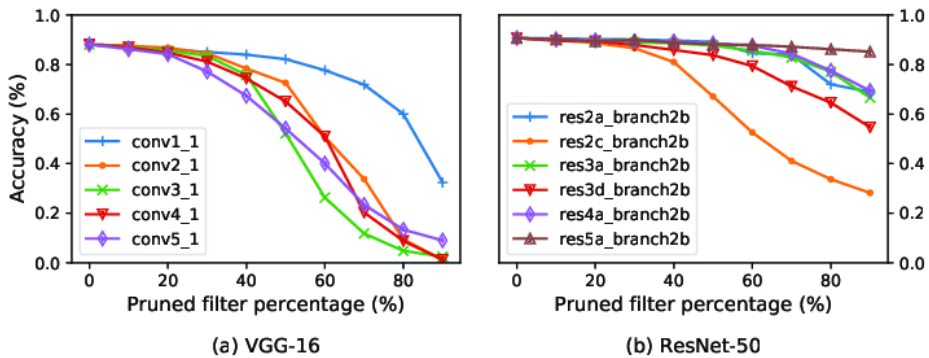
Figure 4: Double descent on ResNet and Transformer (Source: Nakkiran et al. (2021) [6])

Part 5. Plan for the Next

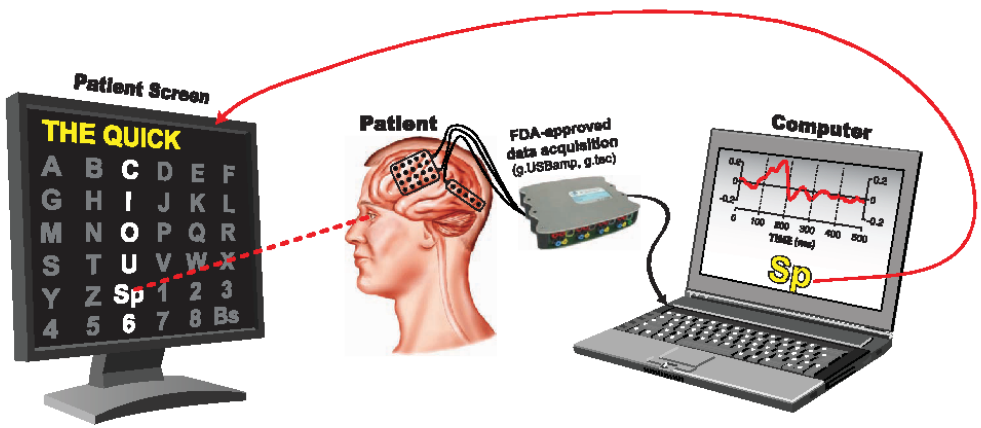
Investigate model pruning technique in a wider landscape

Model pruning in neural networks involves removing certain connections or parameters from the network to make it smaller and more computationally efficient. Despite the reduction in size, effective pruning methods aim to retain or even improve the model's performance by preserving critical connections.





Build a better classifier for P300 brain-computer interface



References

Carlo Baldassi, Clarissa Lauditi, Enrico M Malatesta, Gabriele Perugini, and Riccardo Zecchina.

Unveiling the structure of wide flat minima in neural networks.

Physical Review Letters, 127(27):278301, 2021.

Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal.

Reconciling modern machine-learning practice and the classical bias–variance trade-off.

Proceedings of the National Academy of Sciences, 116(32):15849–15854, 2019.

Abdulkadir Canatar, Blake Bordelon, and Cengiz Pehlevan.

Spectral bias and task-model alignment explain generalization in kernel regression and infinitely wide neural networks.

Nature communications, 12(1):2914, 2021.

Trevor Hastie, Andrea Montanari, Saharon Rosset, and Ryan J Tibshirani.

Surprises in high-dimensional ridgeless least squares interpolation.

Annals of statistics, 50(2):949, 2022.

Preetum Nakkiran.

More data can hurt for linear regression: Sample-wise double descent.

arXiv preprint arXiv:1912.07242, 2019.

Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever.

Deep double descent: Where bigger models and more data hurt.

Journal of Statistical Mechanics: Theory and Experiment, 2021(12):124003, 2021.

Alaa Tharwat.

Parameter investigation of support vector machine classifier with kernel functions.

Knowledge and Information Systems, 61:1269–1302, 2019.