Institut for Theoretical Physics
University Leipzig

Prof. Dr. B. Rosenow
A. Afanah , M. Staats

# Statistical Mechanics of Deep Learning - Problem set 13

*Winter Term 2023/24*

**Hand in:** Friday, 26.01 at 10:00 am, you can upload your solutions to the course webpage on Moodle platform.

## 26. Gaussian Process Regression *9 Points*

In the lecture, we have established the equivalence between noisy gradient descent training with $L_2$-regularization of an infinitely wide neural network and a Gaussian process. Since noisy gradient descent is challenging to simulate numerically, in this problem we replace it by adding a noise term to the training data, i.e. the training data outputs $y_i$ are replaced by $y_i + \zeta_i$, with $\langle \zeta_i \rangle = 0$ and $\langle \zeta_i \zeta_j \rangle = \delta_{ij}\sigma^2$. The purpose of this problem is to numerically demonstrate the equivalence between training a wide neural network and predicting the test data with a Gaussian process.

(a) We consider a two-layer neural network defined by

$$f_{\mathbf{w}}(\mathbf{x}) \;=\; \sum_{i=1}^{N} w_i^{(2)}\, g\left(\sum_{j=1}^{d} w_{ij}^{(1)}\, x_j\right) \quad ,$$

with ReLu transfer function $g(z) = \theta(z)\, z$ and input $\mathbf{x} \in \mathbb{R}^d$. The corresponding Gaussian process is defined by the kernel

$$K(\mathbf{x}, \mathbf{x}') = \langle f_{\mathbf{w}}(\mathbf{x}) f_{\mathbf{w}}(\mathbf{x}') \rangle_w$$

where the average is performed over the set of weights $\{w_i^{(2)}, w_{ij}^{(1)}\}$ with variances $\sigma_1^2 = 1/d$ and $\sigma_2^2 = 1/N$. Numerically, the average can be performed by initializing $M = 400$ different networks using Pytorch, and then computing

$$K(\mathbf{x}, \mathbf{x}') \;=\; \frac{1}{M} \sum_{i=1}^{M} f_{\mathbf{w}_i}(\mathbf{x}) f_{\mathbf{w}_j}(\mathbf{x}') \quad .$$

Perform the average for $N = 2000$ hidden units, and for 200 pairs of vectors $\mathbf{x}, \mathbf{x}' \in \mathbb{S}^d$ for $d = 10$. Compare the numerically averaged kernel with the analytic result

$$K_{\text{analytic}}(\mathbf{x}, \mathbf{x}') \;=\; \sigma_1^2 \sigma_2^2 N \frac{1}{2\pi} \left(\sin\theta + (\pi - \theta)\cos\theta\right) \qquad \text{where } \cos\theta = (\mathbf{x}, \mathbf{x}')$$

by plotting the set of 200 pairs of data points $\{K(\mathbf{x}, \mathbf{x}'), \arccos(\mathbf{x}, \mathbf{x}')\}$ together with the theoretical prediction $K_{\text{analytic}}(\theta)$.

(b) Draw $p = 40$ training data points $\mathbf{x}_i \in \mathbb{S}^d$ and generate labels $y_i$ via the target function $h(\mathbf{x}) = \sqrt{d} \cdot |\mathbf{x} \cdot \mathbf{T}|$ with a fixed $T \in \mathbb{S}^d$ which has randomly chosen components and is then normalized to unity. Add noise with variance $\sigma^2$ to the labels. Now $K(\mathbf{x}_i, \mathbf{x}_j) \equiv K_{ij}$ defines a $p \times p$ matrix. Compute $K_{ij}$ for $N = 100$ and $M = 100$. The Gaussian kernel predictor based on this training data is then given by:

$$g^*(\mathbf{x}_{\text{test}}) = \sum_{n,m=1}^{p} K(\mathbf{x}_{\text{test}}, \mathbf{x}_n) \left(\underline{\underline{K}} + \sigma^2 \mathbb{I}\right)^{-1}_{n,m} y_n \quad .$$

Here, $\mathbb{I}$ denotes the identity matrix, $\sigma^2 = 0.1$ is the variance of the noise added to the labels $y_i$, and $(\underline{\underline{K}} + \sigma^2 \mathbb{I})^{-1}_{n,m}$ must be understood as inverting the matrix $(\underline{\underline{K}} + \sigma^2 \mathbb{I})$ and evaluating the inverse for indices $n, m$. A good check for a correct implementation is that for $\sigma = 0$ you must find $g^*(\mathbf{x}_i) = y_i$.
Next, draw 40 test vectors $\mathbf{x}_{i,\text{test}} \in \mathbb{S}^d$, and compute the mean squared error (loss) as an average

$$\frac{1}{40} \sum_{i=1}^{40} \left[g^*(\mathbf{x}_{i,\text{test}}) - h(\mathbf{x}_{i,\text{test}})\right]^2$$

over the test vectors.

(c) Build a neural network as described in (a) for N=100 and train it on the data set prepared in (b). Use standard gradient descent with mean squared error as loss function, and $L_2$-regularization with strength $\lambda = \sigma^2/\sigma_1^2$. What is the loss of the network on the test data set prepared in (b)? Remember to add label noise to the training labels but not to the test labels!

Hint: Make sure that $M > p$ for all experiments. Otherwise $\text{rank}(\underline{\underline{K}}) < p$ and $\underline{\underline{K}}$ will not be invertible.