Institut for Theoretical Physics
University Leipzig

Prof. Dr. B. Rosenow
M. Kühn

Statistical Mechanics of Deep Learning - Problem set 8

Winter Term 2024/25

Hand in Python code: Before **Monday 09.12.2024, 9:15**, only submit the Python code you have written. Share a Google Colab Notebook with your code and send the link via email to itpleipzig@gmail.com.

16. The Perceptron rule

6 Points

Consider the update of coupling vectors according to the Perceptron rule (see section 3.2 in the book of A. Engel and C. Van den Broeck)

$$\mathbf{J}^{(\text{new})} = \begin{cases} \mathbf{J}^{(\text{old})} + \frac{1}{\sqrt{N}} \boldsymbol{\xi}^\mu \sigma_T^\mu & , \text{ if } \mathbf{J}^{(\text{old})} \cdot \boldsymbol{\xi}^\mu \sigma_T^\mu < 0 \\ \mathbf{J}^{(\text{old})} & , \text{ otherwise.} \end{cases}$$

An interesting aspect of the perceptron rule is that it may easily be modified for the numerical analysis of Gibbs learning. Modify the couplings in order to reproduce the Gibbs learning generalization error, Fig. (1.4) of A. Engel and C. Van den Broeck, and compare to its theoretical asympototic behavior ($\varepsilon \sim \frac{0.625}{\alpha}$). This can be done by adding a random contribution with zero mean with each update of the couplings.

Hints: Making the random contribution smaller than in the book increases the speed of the algorithm. Choose a Gaussian distribution with variance $\simeq 10/\sqrt{N}$. Choose $N \leq 100$ for computational speed and average over 3 learning runs to obtain a smooth curve. Drawing random vectors from the N-sphere is important for this exercise. The following code might help:

```
# function that returns an N-dim vector from the N sphere
def N_sphere_vector( N ):
    vector = np.random.normal(0,1, N)
    vector /= np.linalg.norm( vector)
    vector *= np.sqrt(N)
    return vector
```

17. The Adatron rule

6 Points

The Adatron algorithm combines the the selectivity of the perceptron rule with the adaptive step size of adaline rule (see section 3.5 of A. Engel and C. Van den Broeck), it uses the updates

$$\delta x^\mu = \begin{cases} \gamma(1 - \Delta^\mu) & , \text{ If } \Delta^\mu < 1 \\ -x^\mu & , \text{ otherwise.} \end{cases}$$

where x^μ denotes the embedding strengths, γ is the learning rate, and Δ^μ is the stability or aligning field as defined in the lecture. Implement the adatron learning rule numerically with different training set size of $\alpha = p/N \leq 5$ and compare the generalization error with its theoretical asymptotic behavior ($\varepsilon \sim \frac{0.5005}{\alpha}$). Choose an appropriate upper limit for the maximum number of iterations.

Hint: First compute the correlation matrix $C_{\mu\nu} = \frac{1}{N} \boldsymbol{\xi}^\mu \boldsymbol{\xi}^\nu \sigma_T^\mu \sigma_T^\nu$ once and then use it to determine the stabilities for the algorithm directly from the embedding strengths. Initialize the embedding strengths as zero.