Freie Universität Berlin
FB Mathematik und Informatik
Prof. Dr. Frank Noé

SoSe 2018
15.06.2018

Deep Learning (19238501)

Mock Exam

Name, given name: ______________________________

Matriculation number: ___________________________

Exam time frame: 45 min

Supporting material allowed to be brought along: one sheet of paper (DIN-A4) with your own handwritten notes.

Problem **solutions** must be given in **readable handwriting** together with **all side calculations and explanations needed to understand them** on the provided **DIN A4**-sheets; additional sheets are available upon request. Use a black or blue ballpen or inkwriter (any notes written by pencil or in red ink **will not count**).

All sheets that are to count to your exam must be tagged by your **Name** and **matriculation number**. Make sure all notes are clearly assigned the relevant **number of the examination task**!

Do not hesitate to ask the supervising personnel should you have any exam-related questions whatsoever!

Question:	I	II	III	Total
Points:	5	2	3	10
Score:				

Grade:

Examiner:

I. Deep Learning Theory

(A) What is the **test** set used for? (1 pt)

- ☐ Training the model.
- ☐ Hyperparameter optimization.
- ☐ Model validation after hyperparameter optimization.

(B) We want to apply a two-dimensional convolution of size 5×6 with 32 filters to an image with size 400×300 and 4 color channels. Without bias, how many trainable parameters do we have for this convolution? (1 pt)

- ☐ 5×6.
- ☐ $4 \times 5 \times 6 \times 32$.
- ☐ $400 \times 300 \times 32$.

(C) What does the universal representation theorem (URT) state? (1 pt)

(D) Which kind of functions can be approximated up to arbitrary accuracy by neural networks with only linear activations? Provide a proof for your answer. (2 pts)

II. Shallow Learning (2 pts)

Prove that the principal component analysis (PCA) projections are statistically uncorrelated assuming that all eigenvalues are strictly positive and have multiplicity one (pairwise different).

III. Coding Exercise (3 pts)

We want to approximate an unknown function

$$f : \mathbb{R}^{10} \to \mathbb{R}^2,$$

for which we have N_{samples} observations

$$\mathbf{y}_n = f(\mathbf{x}_n) + \epsilon_n,$$

where $\mathbf{y}_n \in \mathbb{R}^2$, $\mathbf{x}_n \in \mathbb{R}^{10}$, and $n = 1, \ldots, N_{\text{samples}}$; the ϵ_n are small, random errors which are also unkown.

Given the minimal API on page 5, implement a neural network with 3 hidden layers with 32, 16, and 8 nodes; each hidden layer must have a rectified linear unit (ReLU) activation:

$$\text{ReLU}(x) = \max\{0, x\}.$$

Resource: deep learning API

Every `Tensor` can be used with operations `*, +, -, /, **` acting element-wise. Single or multiple elements can be accessed through indexing and slicing. Additionally there are functions

```
nn.sum(x)  # summation over all elements
nn.sqrt(x) # element-wise square root
nn.pow(x, exponent)  # element-wise power
```

```
nn.linear(inputs, units)
```

Implements the operation `out = Wx + b` .

Arguments:

- `inputs` : input tensor
- `units` : dimensionality of output space

Returns a tensor of shape `(*(inputs.shape[:-1]), units)` .

```
nn.clamp(inputs, minval=None, maxval=None)
```

Restricts the values contained in `inputs` to `[minval, maxval]` if not `None` .

Arguments:

- `minval` : Lower bound for `inputs` if not `None`
- `maxval` : Upper bound for `inputs` if not `None`

Returns a tensor of shape `inputs.shape` .