Universität des Saarlandes
Prof. Dr. Dietrich Klakow
Lehrstuhl für Signalverarbeitung
NNTI Winter Term 2024/2025

# Exercise Sheet 6

Feed Forward Neural Network

**Deadline: 04.12.2024 23:59**

**Guidelines:** You are expected to work in a group of 2-3 students. While submitting the assignments, please make sure to include the following information for all our teammates in your PDF/python script:

**Name:**

**Student ID (matriculation number):**

**Email:**

Your submissions should be zipped as **Name1_id1_Name2_id2_Name3_id3.zip** when you have multiple files. For assignments where you are submitting a single file, use the **same naming convention** without creating a zip. For any clarification, please reach out to us on the **CMS Forum**.
Note that the above instructions are mandatory. If you are not following them, tutors can decide not to correct your exercise.

**Exercise 6.1 - Activation functions** ($2 \times 0.5+2 \times 0.25+1+0.5+0.5+0.25 = 3.75$ points)

1. Show that for $x \in \mathbb{R}$ the following identities hold:

   a) $\sigma(-x) = 1 - \sigma(x)$

   b) $\tanh(x) = 2\sigma(2x) - 1$

   c) $\frac{d}{dx}\sigma(x) = (1 - \sigma(x)) \cdot \sigma(x) = \sigma(-x) \cdot \sigma(x)$

   d) $\frac{d}{dx}\tanh(x) = 4(1 - \sigma(2x)) \cdot \sigma(2x)$

2. Consider a two-layer neural network with the output function defined as:

$$y_k(\mathbf{x}, \mathbf{w}) = \sigma\left(\sum_{j=1}^{M} w_{kj}^{(2)} h\left(\sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}\right) + w_{k0}^{(2)}\right),$$

   where the hidden non-linear activation functions $h(\cdot)$ are logistic sigmoid functions $\sigma(a)$. Demonstrate that there exists another equivalent network that computes exactly the same function $y_k(\mathbf{x}, \mathbf{w})$, but with hidden unit activation functions given by $\tanh(a)$. *Hint:* Use the relationship between $\sigma(a)$ and $\tanh(a)$ to show that the network parameters differ only by linear transformations.

*Note:* The network consists of:

* **Input layer:** Takes input $\mathbf{x} = [x_1, x_2, \ldots, x_D]$, where $D$ is the number of features.
* **Hidden layer:** Applies a linear transformation $\sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}$ to the inputs (where $w_{ji}^{(1)}$ are the weights and $w_{j0}^{(1)}$ is the bias), followed by a nonlinear activation $h(\cdot)$.
* **Output layer:** Combines hidden layer outputs using a linear transformation as well, followed by an output activation $\sigma(\cdot)$.

3. Show that the following two inequalities hold for all $x \in \mathbb{R}$:

$$0 < \frac{d}{dx}\sigma(x) \leq \frac{1}{4} \qquad 0 < \frac{d}{dx}tanh(x) \leq 1$$

*Hint:* Your starting point can be the function $f : (0, 1) \to \mathbb{R}, f(t) = -t^2 + t$.
What problem do sigmoid/tanh activation functions suffer from? What activation function would prevent that?

4. Show that the softmax function is invariant with respect to the addition of constant vectors $\mathbf{c} = (c, \ldots, c)^T$, i.e.,

$$\text{softmax}(\mathbf{y} + \mathbf{c}) = \text{softmax}(\mathbf{y})$$

What numerical issues may arise when using the Softmax activation function? How can we prevent those issues using the afore-proven property of the Softmax?

5. Why are activation functions necessary? Why not just use multiple hidden layers with the identity function?

**Exercise 6.2 - Output and Loss Functions** (5×0.25 = 1.25 points)

For each scenario below, identify the appropriate output function (e.g., softmax, sigmoid, linear) and the corresponding loss function (e.g., cross-entropy, mean squared error, etc.) that should be used in a neural network model. **For the last two, briefly explain your choice**.

a) Binary Classification: A neural network predicts whether a social media post contains misinformation or not.

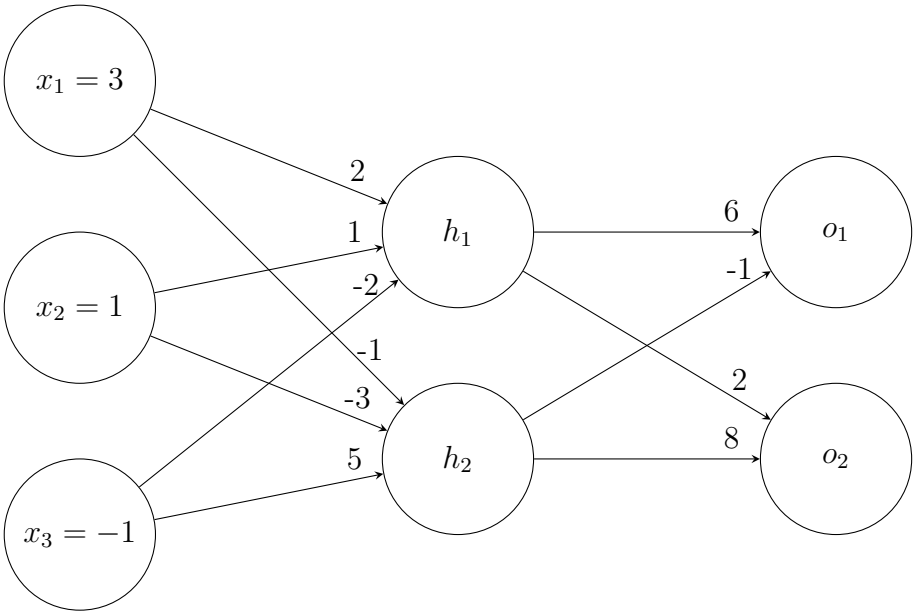
b) Multi-class Classification: A fruit sorting machine identifies a single type of fruit from an image, and each image belongs to exactly one class.

c) Regression: A self-driving car predicts the distance to the nearest obstacle based on input from its sensors.

d) Multi-label Classification: A content moderation system tags uploaded videos with relevant categories (e.g., 'sports,' 'news,' 'education'), where a video can have multiple tags.

e) Next-word Prediction: A language model predicts the next word in a sentence from a vocabulary of over 100,000 words, where the words are organized hierarchically to optimize computational efficiency.

**Exercise 6.3 - Forward Pass for a Fully-Connected NN** (0.5+0.25 points)




Given above is a simple Feed Forward Neural Netwrok (FFNN) with one hidden layer. This network takes an input of three features and produces a vector output. The hidden layer has two hidden units where the activation fucntions are ReLUs. The output function is a Softmax function. For the sake of simplicity, assume that there are **no bias** parameters in the network.

a) Perform one inference step (forward pass) with the given input and weights for the given network. State the formulae that you use throughout your computations and show all the intermediate steps, i.e $h_1 = ...$, $o_1 = ...$

$$W^{(1)} = \begin{bmatrix} 2 & -1 \\ 1 & -3 \\ -2 & 5 \end{bmatrix} \qquad W^{(2)} = \begin{bmatrix} 6 & 2 \\ -1 & 8 \end{bmatrix}$$

b) If this is a binary classification problem, what would be the predicted class label for this given input. *Hint:* Think about what the output of the Softmax implies.

**Exercise 6.4 - BatchNorm** (3×0.25 + 0.75 = 1.5 points)

Please answer the following questions briefly and justify your answers. Your answers should be **no longer than 2-3 sentences**.

a) What motivated batch Normalization?

b) After normalizing the pre-activations in a given layer of the network, BatchNorm re-scales those pre-activations of the batch to have mean $\beta$ and standard deviation $\gamma$, where $\beta$ and $\gamma$ are adaptive learnable parameters. Doesn't this seem just like un-doing the normalization? if no, explain how those are different.

c) How would the behavior of BatchNorm change if the batch size is reduced to 1? At inference time we perform a forward pass with a single sample only, how does the BatchNorm layer to handle this case?

d) How batch-normalization helps optimization? *Hint:* the main results of the paper [1]. Just skim through the paper, don't get consumed by the details.

**Exercise 6.5 - Genral Questions** ($4 \times 0.25 = 1$ points)

Please answer the following questions briefly and justify your answers. Your answers should be **no longer than 2-3 sentences**.

a) Does weight space symmetry help or hinder the optimization process? *Hint:* Think about how it affects the loss surface.

b) Does gradient descent always converge to the minimum of a convex function?

c) Give an example function where Newton's Method converges in one step to the optimum. Does gradient descent require more or less steps?

d) Why is Newton's Method hardly applicable for training Neural Networks?

**Exercise 6.6 - BatchNorm** (1.75 points)

See the accompanying jupyter notebook.

**Exercise 6.7 - Optimization Methods (Bonus)** ($3 \times 0.5 = 1.5$ points)

Read the paper [2] and answer the following:

a) What critical points are considered a major challenge in high-dimensional non-convex optimization?

b) How does gradient descent behave near those critical points?

c) What limitation of Newton's method makes it unsuitable for escaping such critical points?

# References

[1] Santurkar, S., Tsipras, D., Ilyas, A., & Madry, A. (2018, May 29). How does batch normalization help optimization? arXiv.org. https://arxiv.org/abs/1805.11604

[2] Pascanu, R., Dauphin, Y. N., Ganguli, S., & Bengio, Y. (2014, May 19). On the saddle point problem for non-convex optimization. arXiv.org. http://arxiv.org/abs/1405.4604