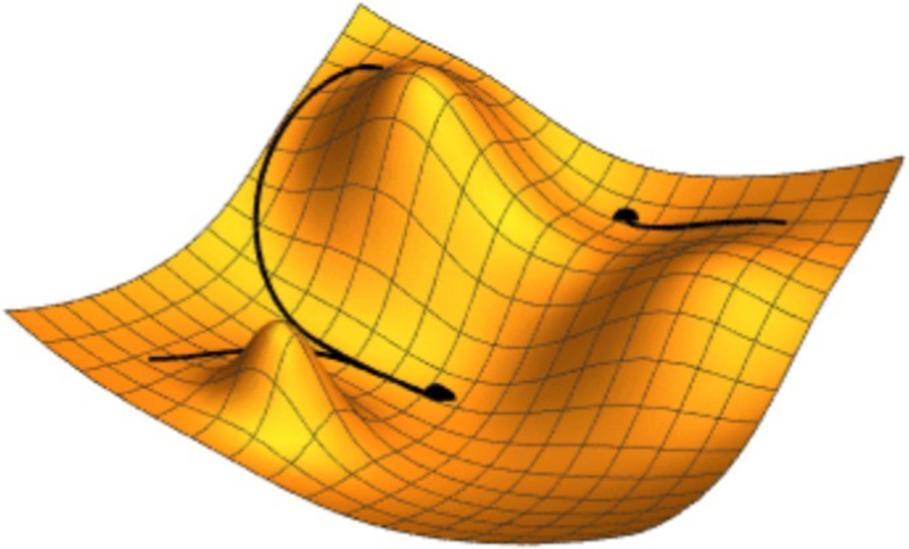
RUHR-UNIVERSITÄT BOCHUM

Review and Outlook

Sen Cheng
Institute for Neural Computation

FUNCTION APPROXIMATION AS OPTIMIZATION PROBLEM

1) Data
2) Model class
3) Loss function
4) Optimization algorithm



MACHINE LEARNING - CLASSES OF ALGORITHMS

- Paradigm: supervised, (unsupervised, reinforcement)
- Task: regression, classification
- Mode: batch, incremental/ online
- Representation: model-based, (instance-based)

LECTURES OFFERED BY THE INI

- Machine Learning: Supervised Methods
- Machine Learning: Unsupervised Methods
- Machine Learning: Reinforcement Learning [offered in future]
- Machine Learning: Evolutionary Algorithms
- Computersehen: Einführung [BSc]
- Computer Vision: Deep Learning Methods

LIMITATIONS OF ANN

- Training ANN
  - requires massive amounts of training data,
  - consumes a lot of energy, and
  - generates CO_2 emissions (training one model > car in its lifetime)
- Network architecture and hyperparameters have to be fine-tuned

LECTURES OFFERED BY THE INI

- Machine Learning: Supervised Methods
- Machine Learning: Unsupervised Methods
- Machine Learning: Reinforcement Learning [offered in future]
- Machine Learning: Evolutionary Algorithms
- Computersehen: Einführung [BSc]
- Computer Vision: Deep Learning Methods
- Autonomous Robotics: Action, Perception, and Cognition
- Computational Neuroscience: Introduction [BSc, SS 2021]
- Computational Neuroscience: Neural Dynamics
- Computational Neuroscience: Vision & Memory

FINAL EXAM

- Written final exam, 120 min., in English
- Date/Time: Friday, 21.02.2020, 12-2 pm
- Place: HZO 10
- Only registered students will be allowed to take the exam
  - AI, ETIT: FlexNow by 10.01.2020
  - other RUB students: eCampus by 31.01.2020
  - non-RUB students: Moodle by 31.01.2020
  - last day to withdraw from registration: 07.02.2020
  - no-shows automatically receive a grade of 0%
- Bring your student id and one A4-sized sheet with notes

CONTENTS OF FINAL

- The final exam will cover everything discussed in the lectures and tutorials, except for some excluded topics (see next slide).
- Even if proof/derivations is excluded, you still need to know the conclusions.

TOPICS

- 01 - Introduction
- 02 - Optimization
- 03 - Regression
- 04 - Model selection
- 05 - Logistic regression
- ~~06 - Biological neural networks~~
- 07 - Artificial neural networks
- 08 – Perceptron
  - ~~convergence proof~~
- 09 - Multi-layer networks
- 10 - Deep neural networks
- 11 - Deep neural networks 2
- 12 - Recurrent neural networks
  - ~~GRU~~
- 13 - Hopfield network
  - ~~convergence proof~~
  - ~~capacity derivation~~
- 14 - Boltzmann machine
  - ~~derivation of learning rule~~

SAMPLE QUESTION 1

- In the standard gradient descent algorithm, choosing the initial value is an important step. Plot a loss function where you can show, given a small enough learning rate and a large enough number of iterations, the chosen initial value prevents gradient descent from finding the global minimum. You do not need to write down any mathematical expressions.

SAMPLE QUESTION 2

- What is the advantage of the update rule in Newton's method

$$\theta_{n+1} = \theta_n - \frac{L'(\theta, X, Y)}{L''(\theta, X, Y)}$$

over gradient descent. Why can this advantage often be problematic?

SAMPLE QUESTION 3

- The following Python code uses stochastic gradient descent to find the minimum of a l2-norm loss in a multiple linear regression problem. The code is incomplete because it is missing a line. What is the function of the missing line? Where should it be inserted?

```
import numpy as np

def sgd_update(X,Y,lr=0.001,runs=1):
#   X : X matrix of predictors
#   Y : observed values of dependent variable
#   lr : learning rate
#   runs : number of iterations of stochastic gradient descent

     theta = np.zeros(len(X[0,:]))
    for j in range(runs):
        for i in range(len(X)) :
            theta = theta - lr*(np.dot(X[i], theta)-Y[i])*X[i]
    return theta
```

SAMPLE QUESTION 4

- When is it preferrable to use the cross-validation method and when is it preferrable to use the holdout method? Discuss the advantages and disadvantages of each method.

SAMPLE QUESTION 5

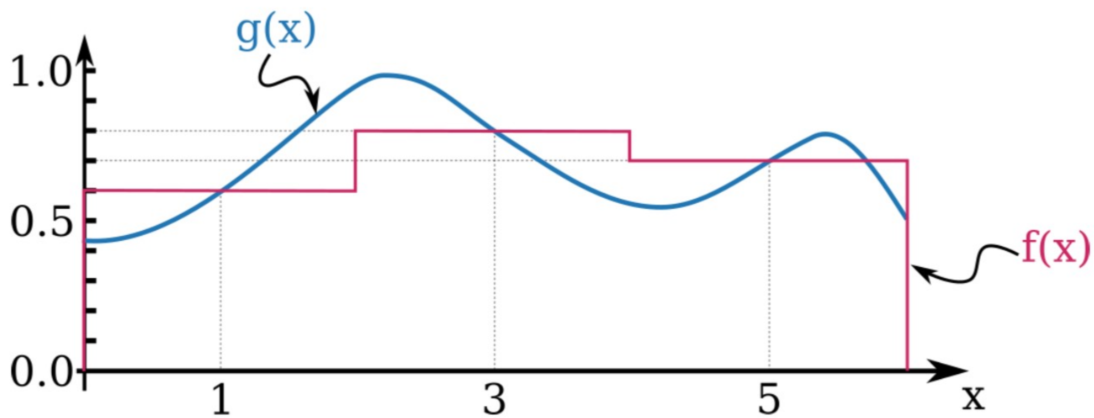
- Why does it not make sense to have values below the line $y=x$ in an ROC curve?

SAMPLE QUESTION 6

The function approximation theorem states that, under mild assumptions, a neural network $f(x)$, where

$$f(x) = \sum_{i=1}^{N} v_i \phi\left(w_i^T x + b_i\right) \tag{2}$$

and $v_i, w_i, b_i \in R$ can approximate any function $g : R \to R$. Given the function $g(x)$ in the figure below and using the logistic function as activation function, which parameters v_i, w_i and b_i produce the shown approximation $f(x)$?



SAMPLE QUESTION 7

- Suppose that two classes are not linearly separable, but you still want to use a single layer perceptron to classify them. How would you modify the basic algorithm in order to try to find a good solution?

SAMPLE QUESTION 8

- The following code was meant to predict temperature from visual and date information using an artificial neural network. In the dataset, the inputs are represented as a 64-dimensional vector and temperature labels as scalar float values. The code contains seven errors (they are not syntax errors). Find the errors, explain what is wrong and write down the correct code.

incorrect

```
# temperature network
model = Sequential ()
model.add(Dense(units=128, activation =' linear ' ))
model.add(Dense(units=128, activation =' linear ' ))
model.add(Dense(units=10, activation ='tanh' ))
# compile
model.compile(optimizer=Adam(lr=10), loss=' cross_entropy ' )
```

SAMPLE QUESTION 8

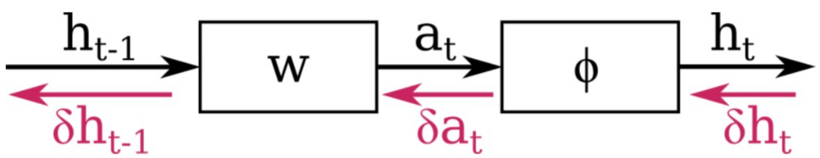
- The following code was meant to predict temperature from visual and date information using an artificial neural network. In the dataset, the inputs are represented as a 64-dimensional vector and temperature labels as scalar float values. The code contains seven errors (they are not syntax errors). Find the errors, explain what is wrong and write down the correct code.

correct

```
# temperature network
model = Sequential()
model.add(Dense(units=128, input_dim=64, activation='relu')) # or input_shape=(64,)
model.add(Dense(units=128, activation='relu'))
model.add(Dense(units=1, activation='linear'))
# compile
model.compile(optimizer=Adam(lr=0.01), loss='mse')
```

SAMPLE QUESTION 9

Write down the forward and backward pass for the variables shown in the following diagram, where w represents a matrix multiplication and ϕ an activation function:



Forward pass:

$$a_t = wh_{t-1}$$
$$h_t = \phi(a_t)$$

Backward pass:

$$\delta h_t = \frac{\partial L}{\partial h_t}$$
$$\delta a_t = \frac{\partial L}{\partial a_t} = \frac{\partial L}{\partial h_t}\frac{\partial h_t}{\partial a_t} = \delta h_t \phi'(a_t)$$
$$\delta h_{t-1} = \frac{\partial L}{\partial h_{t-1}} = \frac{\partial L}{\partial a_t}\frac{\partial a_t}{\partial h_{t-1}} = w^T \delta a_t$$