MSE | MASTER OF SCIENCE IN ENGINEERING

Exam Preparation

TSM_DeLearn

General information

- Condition: 75% of handed-in homework passed
- Written exam, 120 minutes
- No electronic devices permitted
- One A4 page (front and back) of handwritten notes allowed

Exam structure

- Part A: First half of the lecture (Prof. K. Zahn)
- Part B: Second half of the lecture (Prof. A. Fischer)
- Both parts give the same amount of points

General guidelines to prepare for the exam

- Study the slides and additional material we gave (pointers, videos, etc)
- Be able to redo the PW including interpretation of obtained results
- Go through the review questions provided with each PW

Example Questions

- **Targeting theory and concept:** Focus on definitions given in class (blue frames), understanding concepts such as loss function, gradient descent, optimisation/ regularisation procedures, computational graphs, “static” graph strategy or eager execution in TensorFlow, functioning of layers composing a CNN, intuition behind deep architectures, etc.

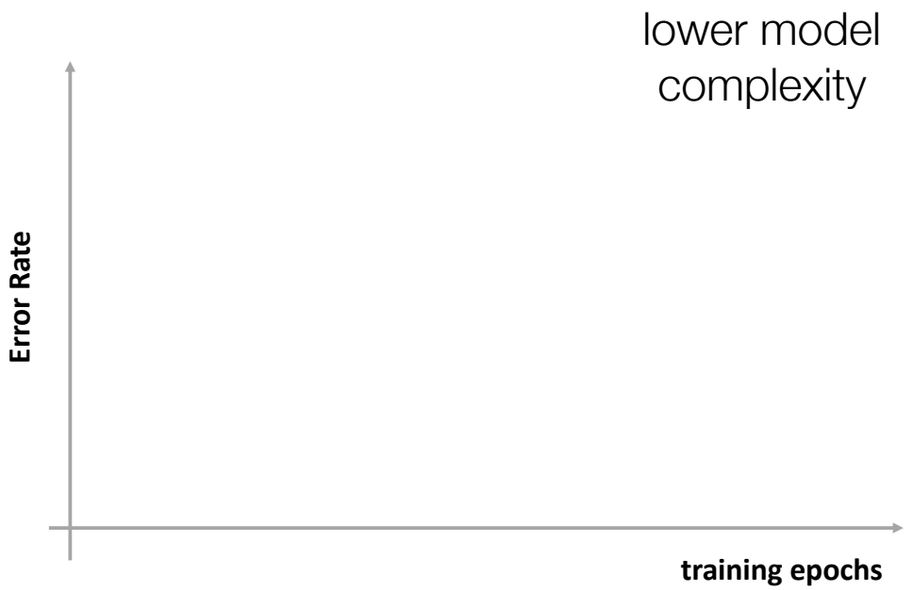
- **Numerical exercise:** E.g. computing the numerical values of gradients in a computational graph, computing the activation map on a simple example, computing the number of parameters of a given architecture.

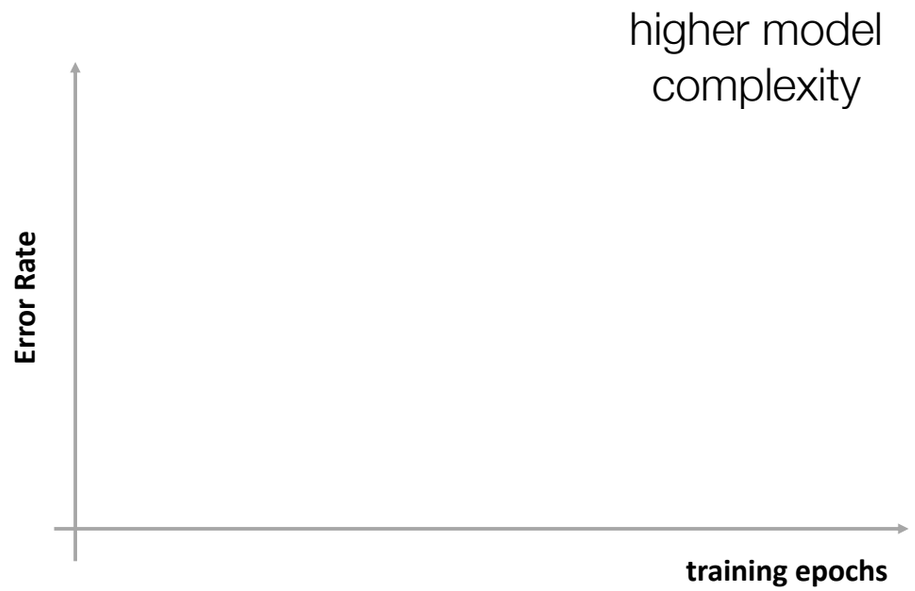
- **Code reading:** Re-explaining a piece of code that you had to produce during the practical works, eventually filling holes in a proposed code (we will not penalise syntax error, we want to see concept such as: “here I would add a Dense layer…”)

- **Use case questions:** In front of a given real-life problem described in, let’s say half a page, what would be your deep learning strategy

Learning Curves

- Plot qualitatively the train and test error rates you expect for the given change in the settings (to be compared horizontally). Express that in 1-2 sentences and explain why it changes the way you expect.





- What change do you expect to see when the parameters (weights) are not properly initialised?

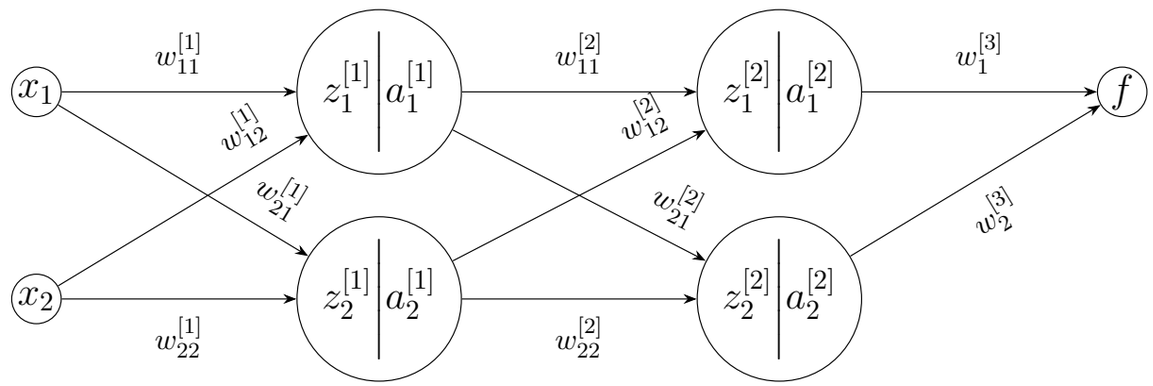
Comp Graphs, Backprop

For the network with two hidden layers as depicted on the right and with

$$f = w_1^{[3]} a_1^{[2]} + w_2^{[3]} a_2^{[2]}$$

compute the derivatives

$$\frac{\partial f}{\partial z_1^{[2]}} \text{ and } \frac{\partial f}{\partial w_2^{[2]}}$$



$$Z^{[1]} = \begin{bmatrix} z_1^{[1]} \\ z_2^{[1]} \end{bmatrix} = \begin{bmatrix} w_{11}^{[1]} & w_{12}^{[1]} \\ w_{21}^{[1]} & w_{22}^{[1]} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad , \quad A^{[1]} = \begin{bmatrix} a_1^{[1]} \\ a_2^{[1]} \end{bmatrix} = \begin{bmatrix} \sigma(z_1^{[1]}) \\ \sigma(z_2^{[1]}) \end{bmatrix}$$

$$Z^{[2]} = \begin{bmatrix} z_1^{[2]} \\ z_2^{[2]} \end{bmatrix} = \begin{bmatrix} w_{11}^{[2]} & w_{12}^{[2]} \\ w_{21}^{[2]} & w_{22}^{[2]} \end{bmatrix} \begin{bmatrix} a_1^{[1]} \\ a_2^{[1]} \end{bmatrix} \quad , \quad A^{[2]} = \begin{bmatrix} a_1^{[2]} \\ a_2^{[2]} \end{bmatrix} = \begin{bmatrix} \sigma(z_1^{[2]}) \\ \sigma(z_2^{[2]}) \end{bmatrix}$$

CNNs

- Consider a convolutional layer with k=10 filters of f=width=height=5, padding p=0 and stride s=2, ‘relu’ activation
  - Compute the shape of the output if the input has shape 28x28x3.
  - Compute the number of parameters that need to be trained.

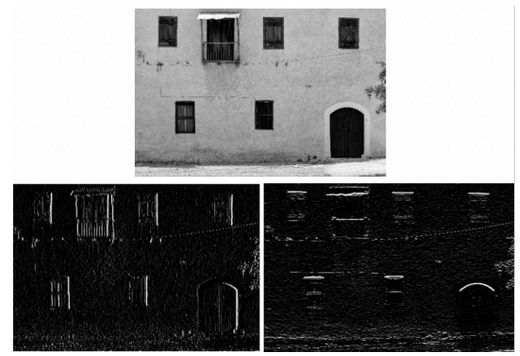
- For input shape 28x28x3 compute the shape of the output for the following network:
  - k=10 filters with f=width=height=5, padding p=1 and stride s=2, ‘relu’ activation function
  - max pooling with stride s=2
  - k=15 filters with f=width=height=3, padding p=2 and stride s=2, ‘relu’ activation
  - 20 filters with width=height=1, padding p=0 and stride s=1, ‘relu’ activation

- What padding needs to be used for a convolutional layer with 3 filters of f=height=width=5 and stride s=1 applied to input images of shape 28x28x3 so that the output and input shape are the same?

CNNs contd.

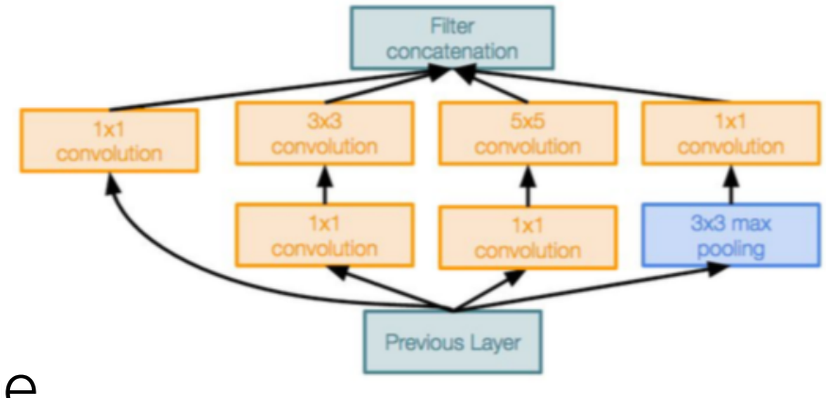
- Which of the following is true about max-pooling?
  - It allows a neuron in a network to have information about features in a larger part of the image, compared to a neuron at the same depth in a network without max pooling.
  - It increases the number of parameters when compared to a similar network without max pooling.
  - It increases the sensitivity of the network towards the position of features within an image.

- Look at the grayscale image at the top:
  - Deduce what type of convolutional filter was used to get each of the lower images.
  - Explain briefly and include typical values of these filters. The filters have a shape of (3,3).



Deep Network Architectures

- Explain two key innovations introduced by GoogLeNet?
- Describe the structure of the new type of layer introduced with GoogLeNet and explain the ideas behind.
- Compute the number of (trainable) parameters involved when applying an inception module (as depicted on the right) to a layer with 10 filters. Furthermore, compute the output dimension of the layer by assuming that the input shape is 10x30x30.
- VGG and GoogLeNet increased the depth significantly as compared with previous models. What is GoogLeNet's solution to deal with the problems with backprop applied in deep architectures?



Inception module

Auto-Encoders

- Design a NN model for de-noising images.
  What would be the steps to train such a model?
- List typical applications for auto-encoders and describe why and how auto-encoders can be used in these applications.

Recurrent Nets

- List 5 typical applications for RNNs and explain why for these applications RNNs may help.
- In what sense are recurrent layers different from convolutional layers? How are parameters shared in RNNs and in CNNs?
- Assume you have designed a RNN for classifying sequences. What are typical data preparation steps?
- Calculate the number of trainable parameters for the model specified with the given code snippet (in `keras`).

```
model = Sequential()
model.add(SimpleRNN(units=64, return_sequences=False, \
                    input_shape=(maxlen,len_vocab)))
model.add(Dense(2, activation='softmax'))

model.compile(loss='categorical_crossentropy', \
              optimizer='adam',metrics=['accuracy'])
model.summary()
```

- Describe the problems typically encountered when training SimpleRNNs and how these can be encountered / solved.
- Why do you observe the vanishing/exploding gradient problem more prominently in RNNs? Why is it more pronounced in SimpleRNNs as opposed to LSTMs?

Bird recognition

- You are approached by the Swiss national ornithology association: SNOA. They would like to build a system to identify birds in natural settings.
  - They want to use 3 cameras positioned each on three different trees pointing to a scene on the ground. The cameras are triggered at the same time by a movement sensor, i.e. you have for each detection 3 pictures of the bird with different angles.
  - The SNOA wants to identify 52 different bird types for which they provided the labels on the dataset. Images may contain other types of birds (out of the 52, labelled as UNKNOWN) or contain nothing if the movement sensor triggers unexpectedly.
  - The dataset contains approximately 1500 images per bird type.
- Explain your strategy for building a bird recognition system.
  - Would you use deep learning, if yes how?
  - What type of architecture would you use?
  - How would you prepare the data?
  - How would you evaluate the system?