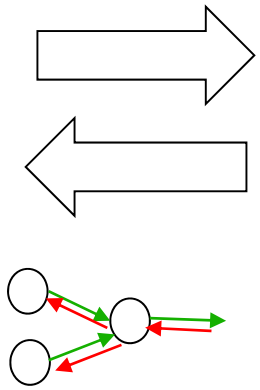
Deep Learning
Lecture 4
First Model with Tensorflow

Prof. Dr. Rainer Herrler
Phone: 09721/ 940-8710
Email: rainer.herrler@fhws.de

Pictures from Wikipedia / Pixabay
Some Pictures generated with Dall-E or Stable Diffusion

What have we done before Easter ?

- Forward Propagation
- Backward Propagation
- Theory for efficient calculation of partial derivatives
- Vectorized Implementation with numpy

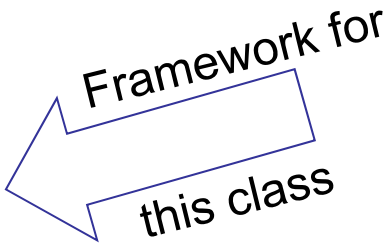


- Next we start using a specialized Framework

Tools and Frameworks for Deep Learning

Frameworks for Deep Learning

- **Tensorflow**
  - Developed by Google
  - Top framework used in Industry
  - goes hand in hand with KERAS as a high level API
- **Pytorch**
  - Developed by Facebook
  - Top framework used in research
- **Caffe 2**
  - Developed by UC Berceley
  - Very fast framework used in academic area
- **Mxnet**
  - Developed by the Apache Foundation used by Amazon
  - Fast and efficient low level Framework
  - Supports multiple programming languages



Getting in touch with tensorflow

- We start using our anaconda installation
- Create and activate a environment (e.g DL) with the following recommended packages
  - Python 3.10
  - Tensorflow 2.10
- Create a directory
- Start jupyter notebook
- Use example notebooks (see later slides)

Setup procedures

Minimal setup without GPU

```
1. conda create -n DL python=3.10 anaconda
2. conda activate DL
3. conda install tensorflow=2.10
4. jupyter notebook
```

Setups with GPUs

This is different from system to system and can be very complex and time consuming to setup. For NVIDEA cards I have the following hints that helped in April 2023 on Windows:

- Install Cuda Toolkit 11.2.2 (not newest 12.1)
- Install Cudnn 8.1.1 (not newest 8.8.1)

First contact with tensorflow

DL_004_TesorsAndVariables.ipynb
DL_005_ExerciseArray2Tensor.ipynb

Differences to Numpy (Conversion recipes)

- np.arange -> tf.range
- Assignment ->
  - var[i1,i2].assign(value) works
  - var[i1][i2].assign didn’t work
- Adjust dtype : int to tf.int64
- reshape has different syntax (tf.reshape … axis has to be specified)
- Dot Multiplication with tf.tensordot

What did we learn in DL_004 and DL_005

- Tensorflow provides tensors which are equivalent to arrays of numpy
- Numpy is fast but does not use a GPU
- Tensorflow distinguishes between constant tensors and variable tensors
- Tensorflow allows to use GPUs or TPUs
- Many operations in Tensorflow are similar or equal to numpy, translation is not too difficult.
- Vectorized operations can be performed much faster than implementations with for-loops
- We saw how to do runtime-measurements

A first network with Keras

DL_006_FashionMnist.ipynb
DL_007_FashionMnistMultinomial.ipynb

Last Slide on 14.04.2023

What did we learn in DL_006 and DL_007

- We create a simple Keras network
  - We instantiate the class “Sequential”
  - We add Layers specifying number of nodes and activation function
  - For the first layer we need the input_shape
  - We compile the model specifying the optimizer, loss function and metrics to track
- Training the network
  - We call fit with the training data X,y
  - We can specify epochs and batch_size
  - The fit-Function returns history-Information for us to plot graphs for the surveilling the training process