Back-Propagation and Algorithms for Training Artificial Neural Networks with TensorFlow

Gero Kauerauf

30. October 2020

Software and Tools for Computational Engineering

RWTH AACHEN UNIVERSITY

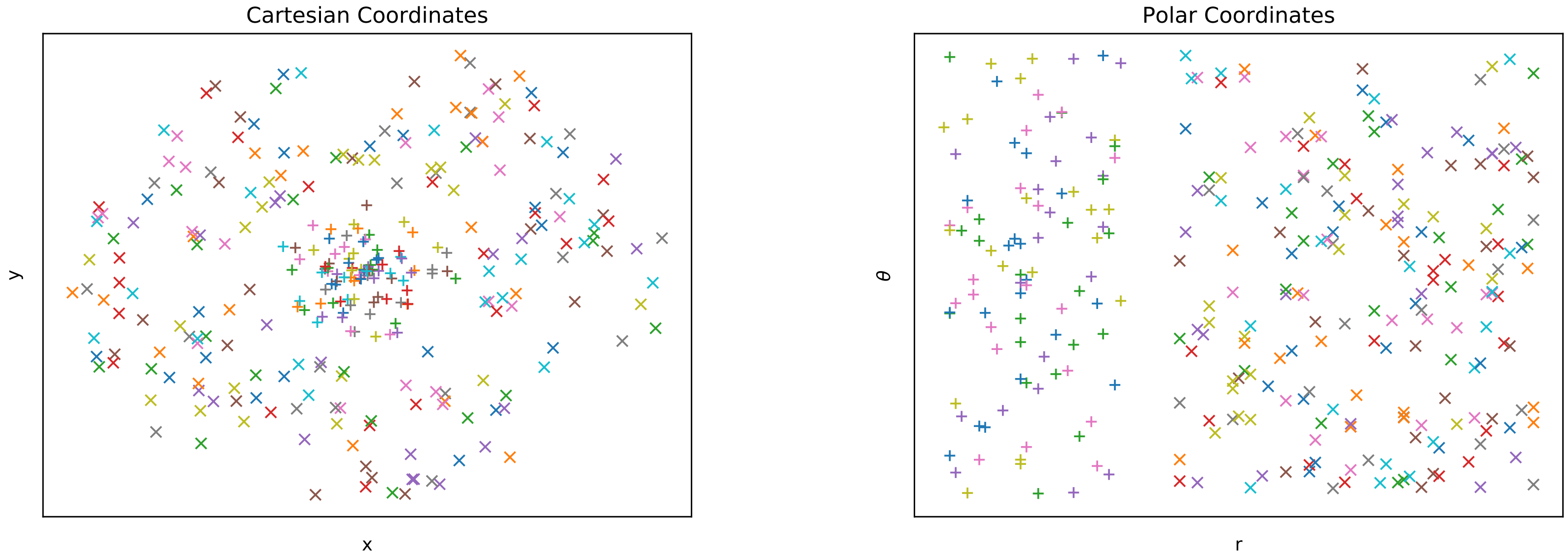
List of Contents

Introduction

- What is a complicated problem for a computer?
- A problems that is
  - hard to describe formally
  - intuitive solvable for humans
- For example: Recognizing a flower on a picture
- Machine Learning
  - Acquiring its own knowledge
  - Extracting patterns from data
- Deep Learning
  - Hierarchy of Concepts
  - Representative Graph has Layers

Data Representation

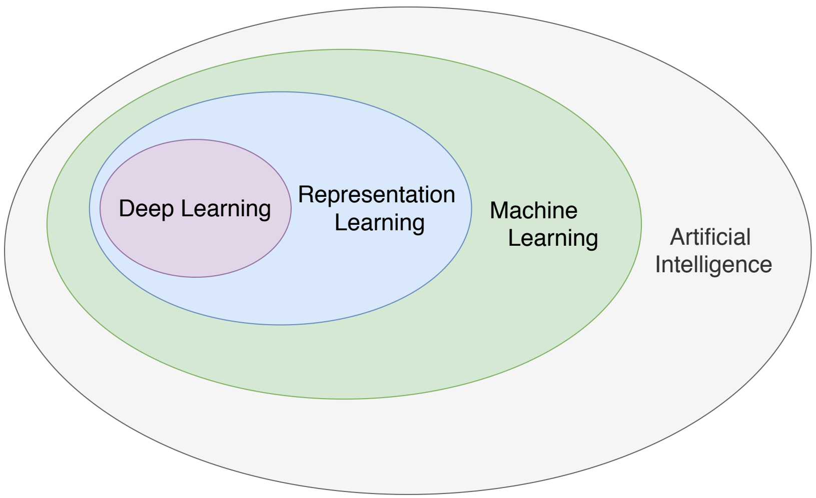
- Importance of Data Representation
  - Tasks can be impossible in one representation and easy in another

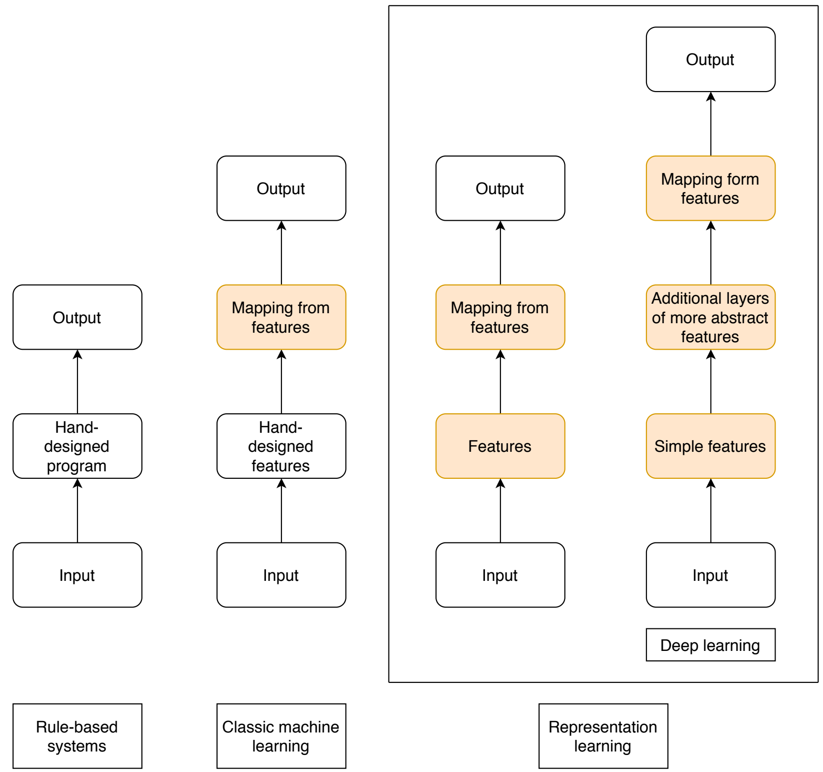


  - One solution to this is **representation learning**
    - Machine Learning now also discovers the representation itself
    - Often better Performance
    - AI can rapidly adapt to new tasks with minimal human intervention

Different AI disciplines

- Relations between different AI disciplines



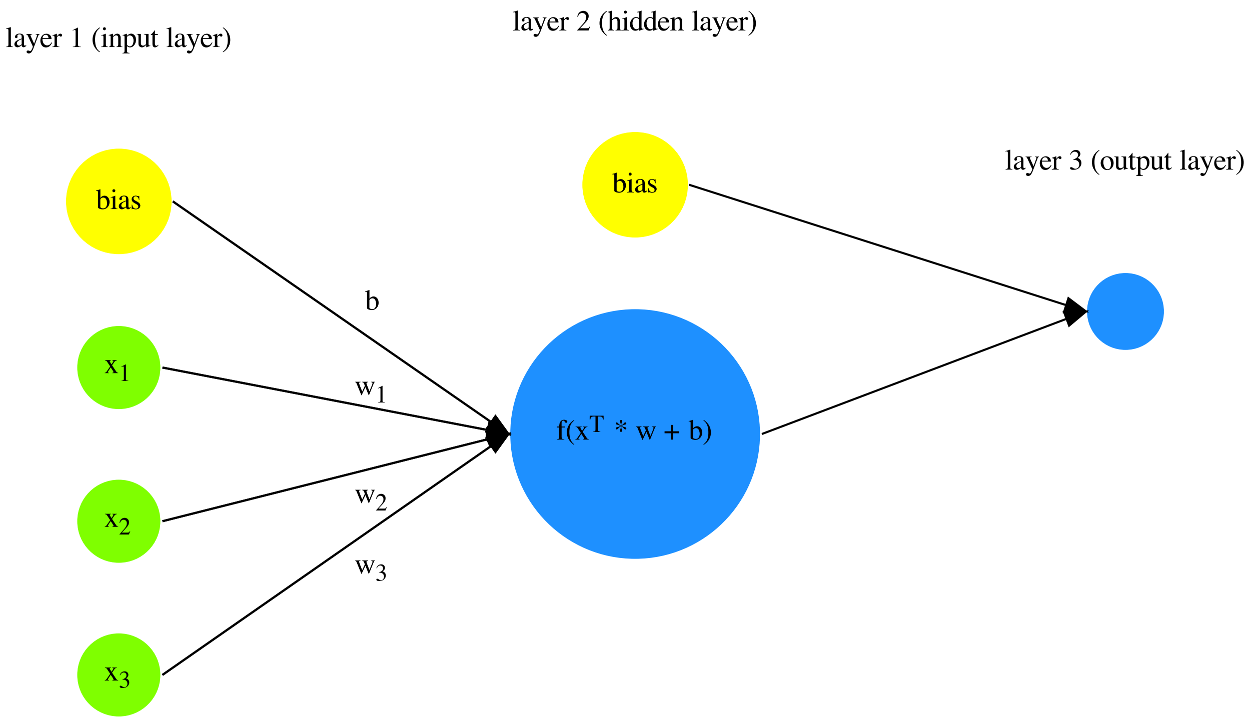


www.DeepLearningBook.org

Neural Networks

- An Neural Network consists out of nodes (vertices) and edges
- Nodes are modeling Neurons
- Edges are modeling synapses
- Nodes have an activations function (e.g. a rectifier function)
- The graph is weighted, that means each edge has a weight to it ($w \in \mathbb{R}$)
- How it works:
  - Input nodes are given an input value
  - Each node sums up the inputs that it gets and outputs the activation function value of that sum

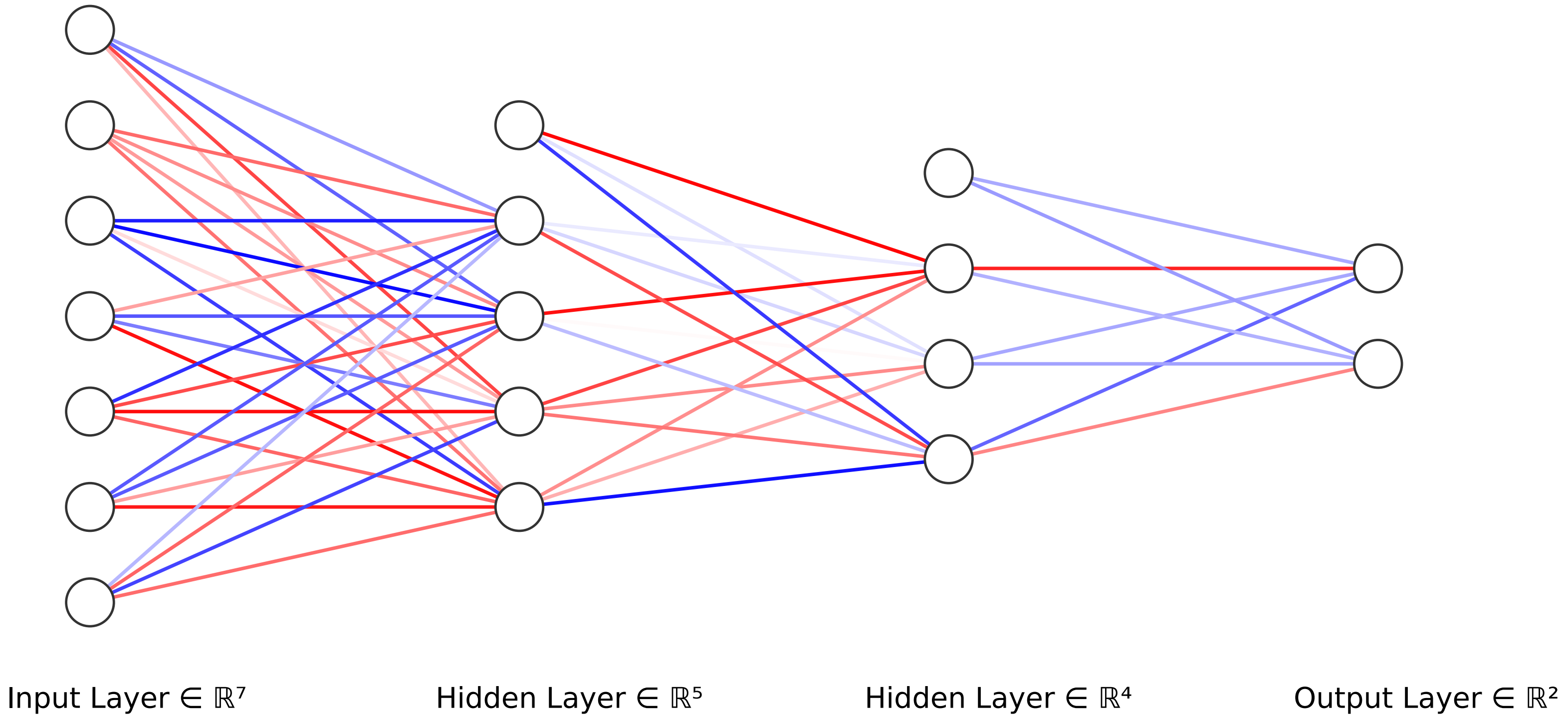
Neural Networks



- Where $x^T * w + b$ is nothing but $\sum_{i=1}^{3}(x_i * w_i) + b$

Neural Networks

- A deep neural network consists out of
  - an input layer
  - multiple hidden layers
  - an output layer

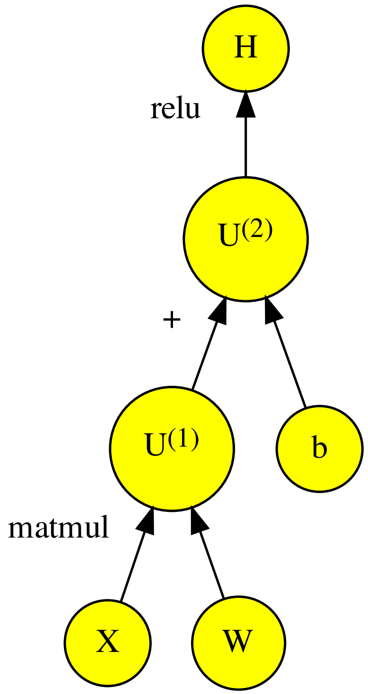


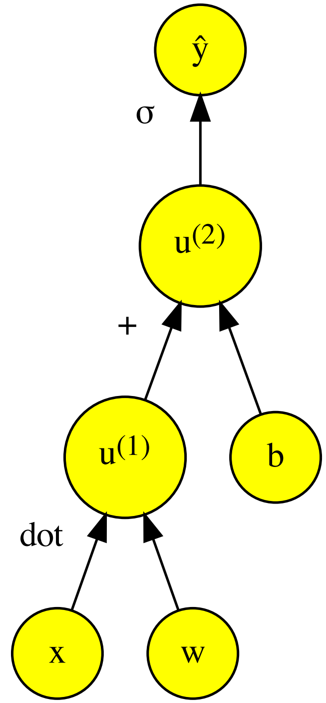
Linear Algebra

- Nothing but linear algebra
- All operations can easily be described with vectors and matrices

- We can describe the weights of each layer of the DNN with a weight matrix
- The input can be written as a vector, same goes for the biases
- Thus propagating through the network is simply a matrix-vector multiplication plus the corresponding bias foreach layer

Computational Graphs

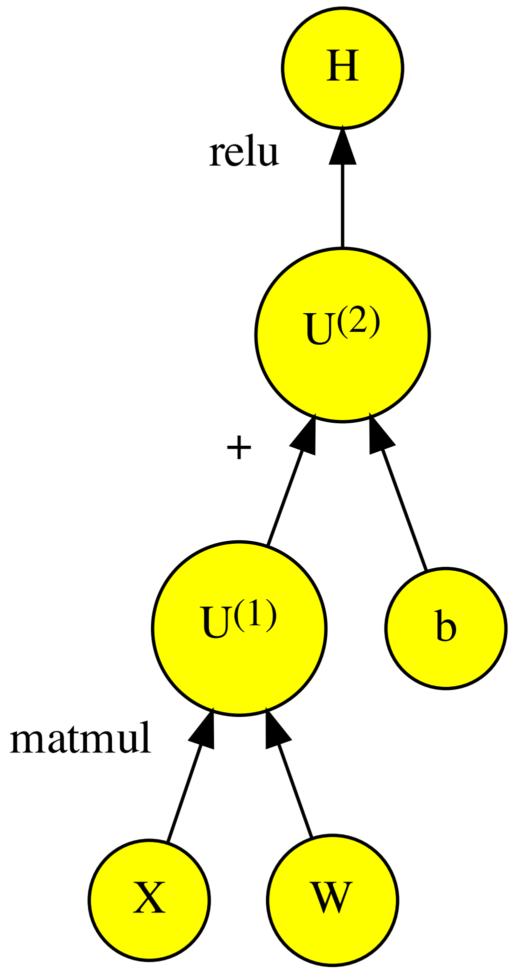




www.DeepLearningBook.org

- A computational graph is used to describe a mathematical expression as a graph
- This allows us to apply *graph algorithms* on it

Computational Graphs

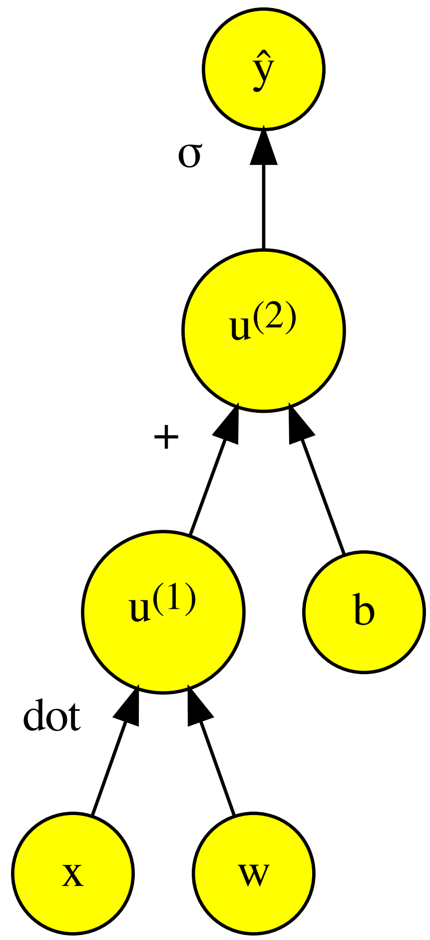


- First we compute XW, that we store in $U^{(1)}$
- Then we calculate $U^{(2)} = U^{(1)} + b \Leftrightarrow U^{(2)} = XW + b$
- Finally ReLU refers to *rectifier linear unit*
- $\rightarrow f(x) = max\{0, x\} = |x|$

- This computational graph computes $H = max\{0, XW + b\}$

Computational Graphs

- This one here computes
  $\hat{y} = \sigma(x^T w + b)$



Back-Propagation

- Back-Propagation is a “graph algorithm” that computes the the gradient of such graph
- Backpropagation is the recursive usage of the *Chain-Rule* to obtain the gradient

- We can now define a cost function for a DNN
- For that cost function we can find the respective computational graph
- We can use the Back-Propagation to find the gradient of the cost function
- And then find the minimum of the gradient of the cost function via *Stochastic Gradient Descent* (SGD) an extension of the normal *Gradient Descent*, which is an iterative algorithm for finding a local minimum

Sources

- Sources:
  - **Deep Learning** by Ian Goodfellow and Yoshua Bengio and Aaron Courville
    www.deeplearningbook.org
  - **TensorFlow**
    www.tensorflow.org
  - And of course wikipedia for quick look ups

- Tools used:
  - **Graphviz** for plotting computational graphs
    www.graphviz.org
  - **matplotlib** for more plots
    www.matplotlib.org

Workplan

- Topics as discussed:
  - Back-Propagation
    - How does Back-Propagation calculate the gradient
    - Cost functions of Neural Networks
  - (Stochastic-) Gradient Descent
    - Gradient Descent
    - Especially Stochastic Gradient Descent
  - Neural Networks in TensorFlow
    - Different models in TensorFlow (Keras)