Fundamentals of Neural Networks

Mathias Jackermeier

June 6, 2018

Technische Universität München

Introduction

Figure 1: A self-driving car.
Credit: Marc van der Chijs / CC BY-ND 2.0

Introduction

Figure 2: A digital assistant.
Credit: Kārlis Dambrāns / CC BY 2.0

Introduction

Figure 3: Object detection in images.
Credit: Lu et. al[1]

[1] "1-HKUST: Object Detection in ILSVRC 2014", *CoRR*, vol. abs/1409.6155, 2014

Outline

Outline

The Perceptron

Example Task

- Predict whether an input image of a handwritten digit shows a zero or another digit

MNIST Data Sample

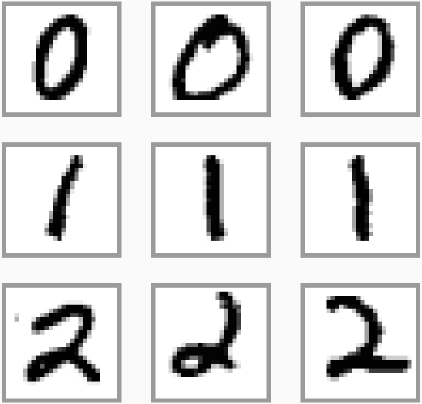

Figure 4: Examples from the MNIST database.
Credit: Josef Steppan / CC BY-SA 4.0

Example Task

- Predict whether an input image of a handwritten digit shows a zero or another digit
- The image is represented as a flattened vector of pixel intensities $\mathbf{x} \in \mathbb{R}^{784}$

Example Task

- Predict whether an input image of a handwritten digit shows a zero or another digit
- The image is represented as a flattened vector of pixel intensities $\mathbf{x} \in \mathbb{R}^{784}$
- The output should be 1 if the image shows a zero, otherwise it should be -1

Example Task

- Predict whether an input image of a handwritten digit shows a zero or another digit
- The image is represented as a flattened vector of pixel intensities $\mathbf{x} \in \mathbb{R}^{784}$
- The output should be 1 if the image shows a zero, otherwise it should be -1
- **Idea**: Assign a weight to every input pixel

Model Specification

The perceptron accepts n input values and computes an output value $\hat{y}$:

$$\begin{aligned} \hat{y} &= \operatorname{sign}\left(\sum_{i=1}^{n} w_i x_i\right) \\ \equiv \hat{y} &= \operatorname{sign}\left(\mathbf{w}^\top \mathbf{x}\right) \end{aligned} \tag{1}$$

Visual Representation

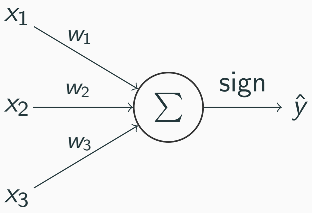

Figure 5: A visual representation of the perceptron model.

Shortcomings of the Perceptron

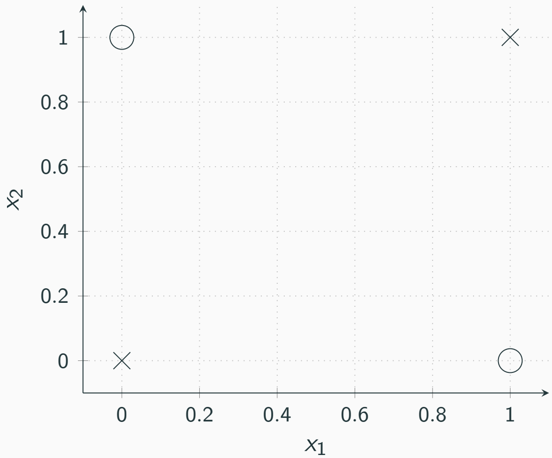

Figure 6: The perceptron cannot learn the XOR function since the data is not linearly separable.

Feedforward Neural Networks

Networks of perceptron-like units

- **Idea**: A combination of multiple perceptrons could make much better predictions

Networks of perceptron-like units

- **Idea**: A combination of multiple perceptrons could make much better predictions
- We arrange the perceptrons in layers

Networks of perceptron-like units

- **Idea**: A combination of multiple perceptrons could make much better predictions
- We arrange the perceptrons in layers
- The input of a layer is the output of the previous layer

Networks of perceptron-like units

- **Idea**: A combination of multiple perceptrons could make much better predictions
- We arrange the perceptrons in layers
- The input of a layer is the output of the previous layer
- This network model is called *feedforward neural network* or *multilayer perceptron*

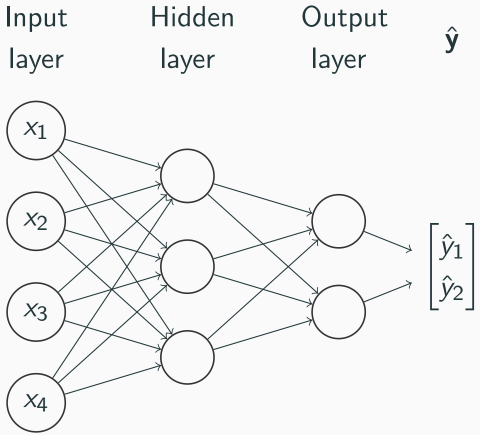

Figure 7: A three-layer feedforward neural network.

Neurons

- In practice, we use a modified version of the perceptron as computing units

Neurons

- In practice, we use a modified version of the perceptron as computing units
- We add a scalar bias value to the output computation:

$$\hat{y} = \text{sign}\left(\mathbf{w}^\top \mathbf{x} + b\right) \tag{2}$$

Neurons

- In practice, we use a modified version of the perceptron as computing units
- We add a scalar bias value to the output computation:

$$\hat{y} = \operatorname{sign}\left(\mathbf{w}^\top \mathbf{x} + b\right) \tag{2}$$

- We replace the sign function with a generic function f:

$$\hat{y} = f\left(\mathbf{w}^\top \mathbf{x} + b\right) \tag{3}$$

Neurons

- In practice, we use a modified version of the perceptron as computing units
- We add a scalar bias value to the output computation:

$$\hat{y} = \text{sign}\left(\mathbf{w}^\top \mathbf{x} + b\right) \tag{2}$$

- We replace the sign function with a generic function f:

$$\hat{y} = f\left(\mathbf{w}^\top \mathbf{x} + b\right) \tag{3}$$

- These modified perceptrons are often called *neurons* or simply *units*

Mathematical Formulation

- We can specify a single neuron with a weight vector $\mathbf{w}$ and a bias value b

Mathematical Formulation

- We can specify a single neuron with a weight vector $\mathbf{w}$ and a bias value b
- Since a neural network consists of multiple neurons in a layer, we need weight *matrices* $\mathbf{W}^{(l)}$ and bias *vectors* $\mathbf{b}^{(l)}$ to specify the parameters of a layer l

Mathematical Formulation

- We can specify a single neuron with a weight vector $\mathbf{w}$ and a bias value b
- Since a neural network consists of multiple neurons in a layer, we need weight *matrices* $\mathbf{W}^{(l)}$ and bias *vectors* $\mathbf{b}^{(l)}$ to specify the parameters of a layer l
- The weight $w_{ij}^{(l)}$ is the weight from the xx neuron in the xx layer to the xx neuron in the xx layer

Mathematical Formulation

- We can specify a single neuron with a weight vector $\mathbf{w}$ and a bias value b
- Since a neural network consists of multiple neurons in a layer, we need weight *matrices* $\mathbf{W}^{(l)}$ and bias *vectors* $\mathbf{b}^{(l)}$ to specify the parameters of a layer l
- The weight $w_{ij}^{(l)}$ is the weight from the xx neuron in the xx layer to the xx neuron in the xx layer
- The bias $b_i^{(l)}$ is the bias of the xx neuron in the xx layer

Mathematical Formulation

- The output at layer l is then given by

$$\mathbf{a}^{(l)} = f^{(l)}\left(\mathbf{W}^{(l)\top}\mathbf{a}^{(l-1)} + \mathbf{b}^{(l)}\right) \tag{4}$$

Training Feedforward Neural Networks

Extensions

Thank you!