UNIVERSITÄT DUISBURG ESSEN

Open-Minded

AdaLinE & Multilayer Perceptron

Neuroinformatics Tutorial 8

Duc Duy Pham[1]

[1]Intelligent Systems, Faculty of Engineering,
University of Duisburg-Essen, Germany

Content

- Revision: Practical Task (Already uploaded)
- Revision: Lecture (AdaLinE)
- New Practical Tasks (Already uploaded)
- Revision: Lecture (MLP)

Content

- Revision: Practical Task (Already uploaded)
- **Revision: Lecture (AdaLinE)**
- New Practical Tasks (Already uploaded)
- Revision: Lecture (MLP)

Revision: Lecture

- Which methods were presented in the lecture to train AdaLinE?

Revision: Lecture

- Which methods were presented in the lecture to train AdaLinE?
    1. Proportional Learning Rule
    2. Loss minimization by gradient descent
    3. Loss minimization by Simplex Optimization
    4. Weight estimation by Pseudo Inverse

Revision: Lecture

- Which methods were presented in the lecture to train AdaLinE?
  1. Proportional Learning Rule
  2. Loss minimization by gradient descent
  3. Loss minimization by Simplex Optimization
  4. Weight estimation by Pseudo Inverse

A: all	B: 1,2,4
C: 1,2	D: 1,4

Revision: Lecture

- Which methods were presented in the lecture to train AdaLinE?
  1. Proportional Learning Rule
  2. Loss minimization by gradient descent
  3. Loss minimization by Simplex Optimization
  4. Weight estimation by Pseudo Inverse

A: all

B: 1,2,4

C: 1,2

D: 1,4

Revision: Lecture

- Which methods were presented in the lecture to train AdaLinE?
  1. Proportional Learning Rule
  2. Loss minimization by gradient descent
  3. Loss minimization by Simplex Optimization
  4. Weight estimation by Pseudo Inverse

A: all

B: 1,2,4

C: 1,2

D: 1,4

AdaLinE Pseudo Inverse

- Step by step:
  (Assume data set that stores samples by row)

AdaLinE Pseudo Inverse

- Step by step:
  (Assume data set that stores samples by row)

  AdaLinE calculates scalar product of each row with weight vector

AdaLinE Pseudo Inverse

- Step by step:
  (Assume data set that stores samples by row)

  AdaLinE calculates scalar product of each row with weight vector

  Output should be close to desired output

AdaLinE Pseudo Inverse

- Step by step:
  (Assume data set that stores samples by row)

  AdaLinE calculates scalar product of each row with weight vector

  Output should be close to desired output

  Basically: $\boldsymbol{X} \cdot \boldsymbol{w} = \boldsymbol{Y}$

AdaLinE Pseudo Inverse

- Step by step:
  (Assume data set that stores samples by row)

  AdaLinE calculates scalar product of each row with weight vector

  Output should be close to desired output

  Basically: $X \cdot w = Y$

$$\begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 2 \end{pmatrix} \cdot \begin{pmatrix} w_0 \\ w_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 4 \\ 4 \end{pmatrix}$$

AdaLinE Pseudo Inverse

- Step by step:
  (Assume data set that stores samples by row)

  AdaLinE calculates scalar product of each row with weight vector

  Output should be close to desired output

  Basically: $$X \cdot w = Y$$

$$\begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 2 \end{pmatrix} \cdot \begin{pmatrix} w_0 \\ w_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 4 \\ 4 \end{pmatrix}$$

This is an over determined linear equation system!
Use Pseudo Inverse to find weight vector!

AdaLinE Pseudo Inverse

- This is the Pseudo Inverse of X: $(X^T X)^{-1} X^T$
- It is the Pseudo Inverse since multiplication with X results in the identity matrix

$$(X^T X)^{-1} X^T X \cdot w = (X^T X)^{-1} X^T Y$$

$$w = (X^T X)^{-1} X^T Y$$

Content

- Revision: Practical Task (Already uploaded)
- Revision: Lecture (AdaLinE)
- New Practical Tasks (Already uploaded)
- Revision: Lecture (MLP)

Revision: Lecture

- What is a Multilayer Perceptron?

Revision: Lecture

- What is a Multilayer Perceptron?
    1. Network of Perceptrons
    2. Perceptrons organized in multiple layers
    3. One Perceptron with multiple inner hidden layers
    4. Basically a multilayer Turing Machine

Revision: Lecture

- What is a Multilayer Perceptron?
  1. Network of Perceptrons
  2. Perceptrons organized in multiple layers
  3. One Perceptron with multiple inner hidden layers
  4. Basically a multilayer Turing Machine

A: all

B: 1,2,4

C: 1,2

D: 1,2,3

Revision: Lecture

- What is a Multilayer Perceptron?
  1. Network of Perceptrons
  2. Perceptrons organized in multiple layers
  3. One Perceptron with multiple inner hidden layers
  4. Basically a multilayer Turing Machine

A: all

B: 1,2,4

C: 1,2

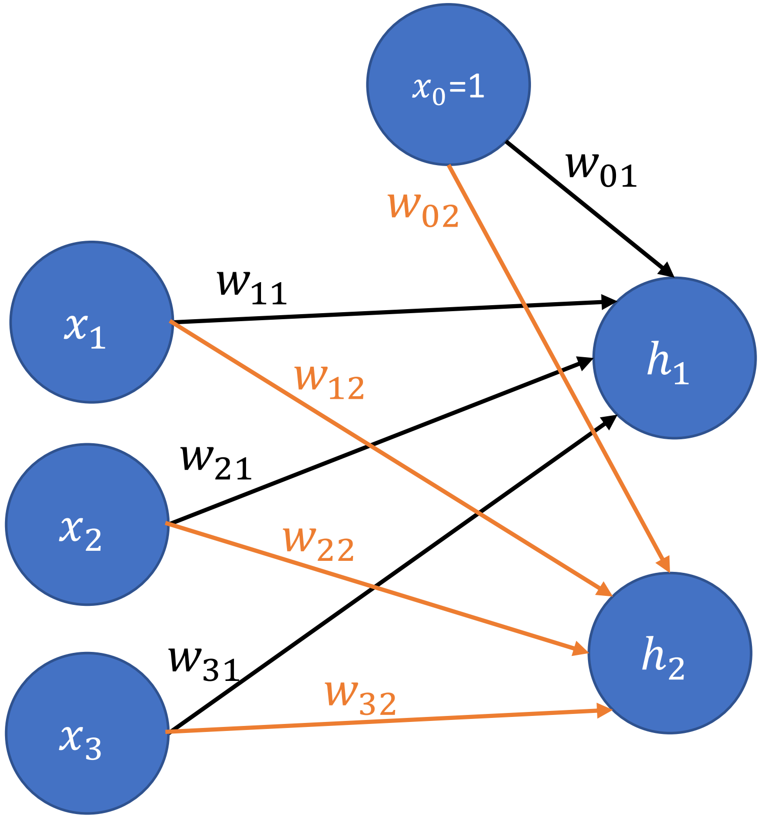
D: 1,2,3

MLP - Definition

- Network of Perceptrons
- Perceptrons organized in multiple layers
- Weighted connection between each Perceptron of one layer to each perceptron of the next layer
- Propagation function of perceptrons: Linear Associator
- Activation function (usually) defined per layer

- 1 Input layer (your usual input vector)
- (Multiple hidden layers)
- 1 Output layer

Calculation of propagated value

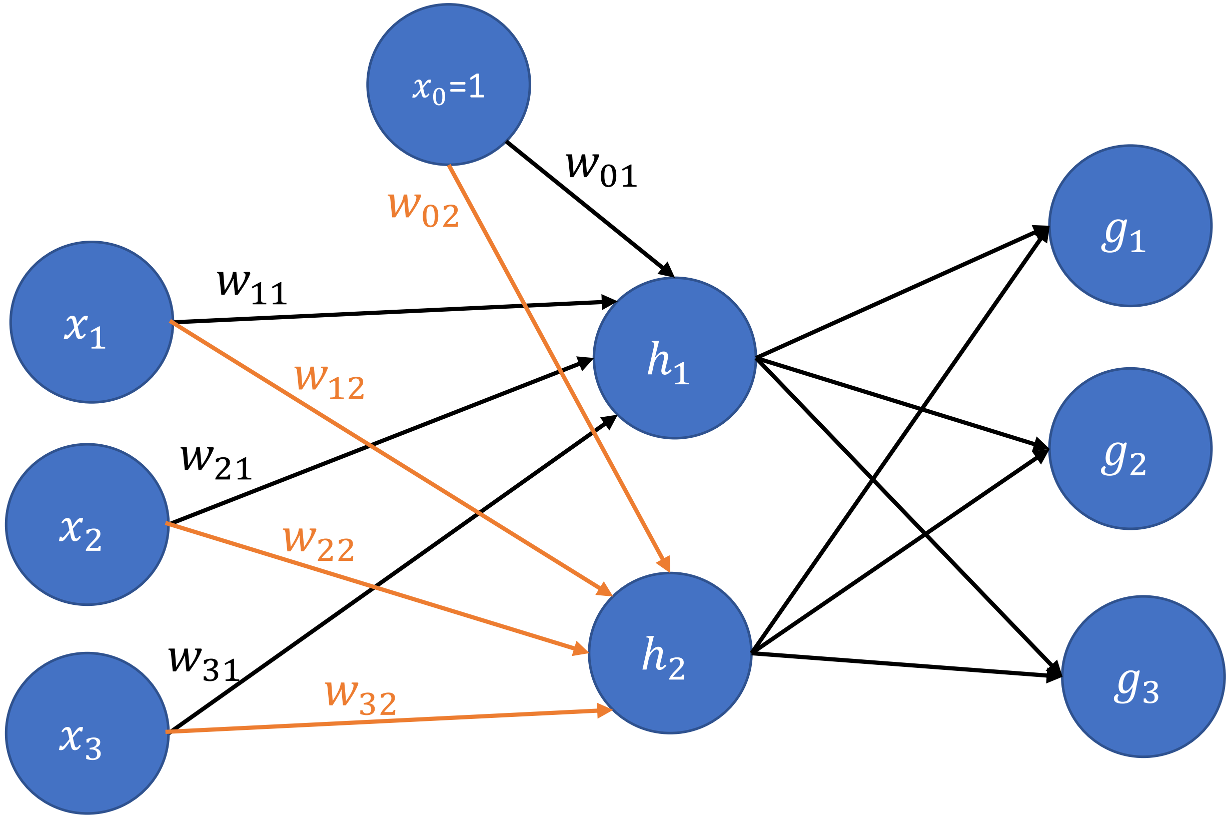


$$h_1 = \sum_{i=0}^{3} w_{i1} x_i$$

$$h_2 = \sum_{i=0}^{3} w_{i2} x_i$$

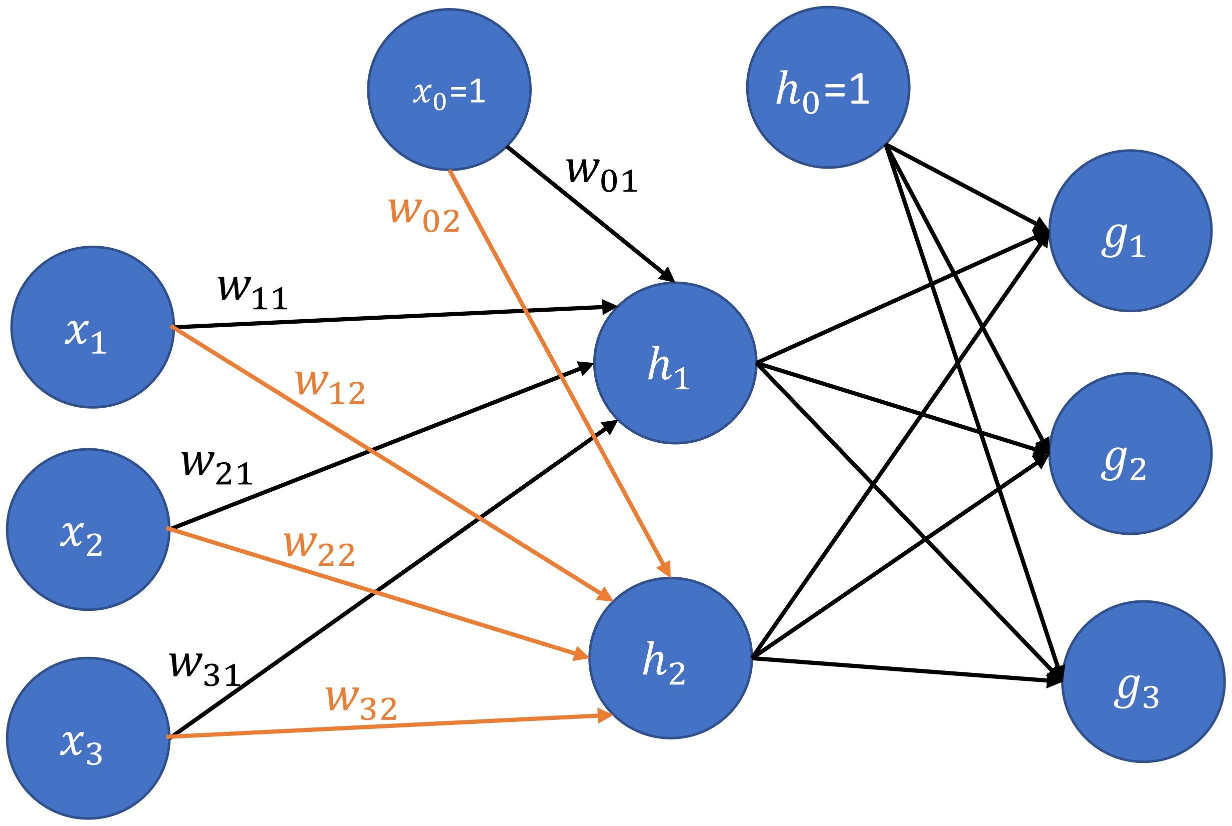
$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \qquad W = \begin{bmatrix} w_{01} & w_{02} \\ w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \end{bmatrix} \qquad \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} = W^T \cdot x$$

Calculation of propagated value



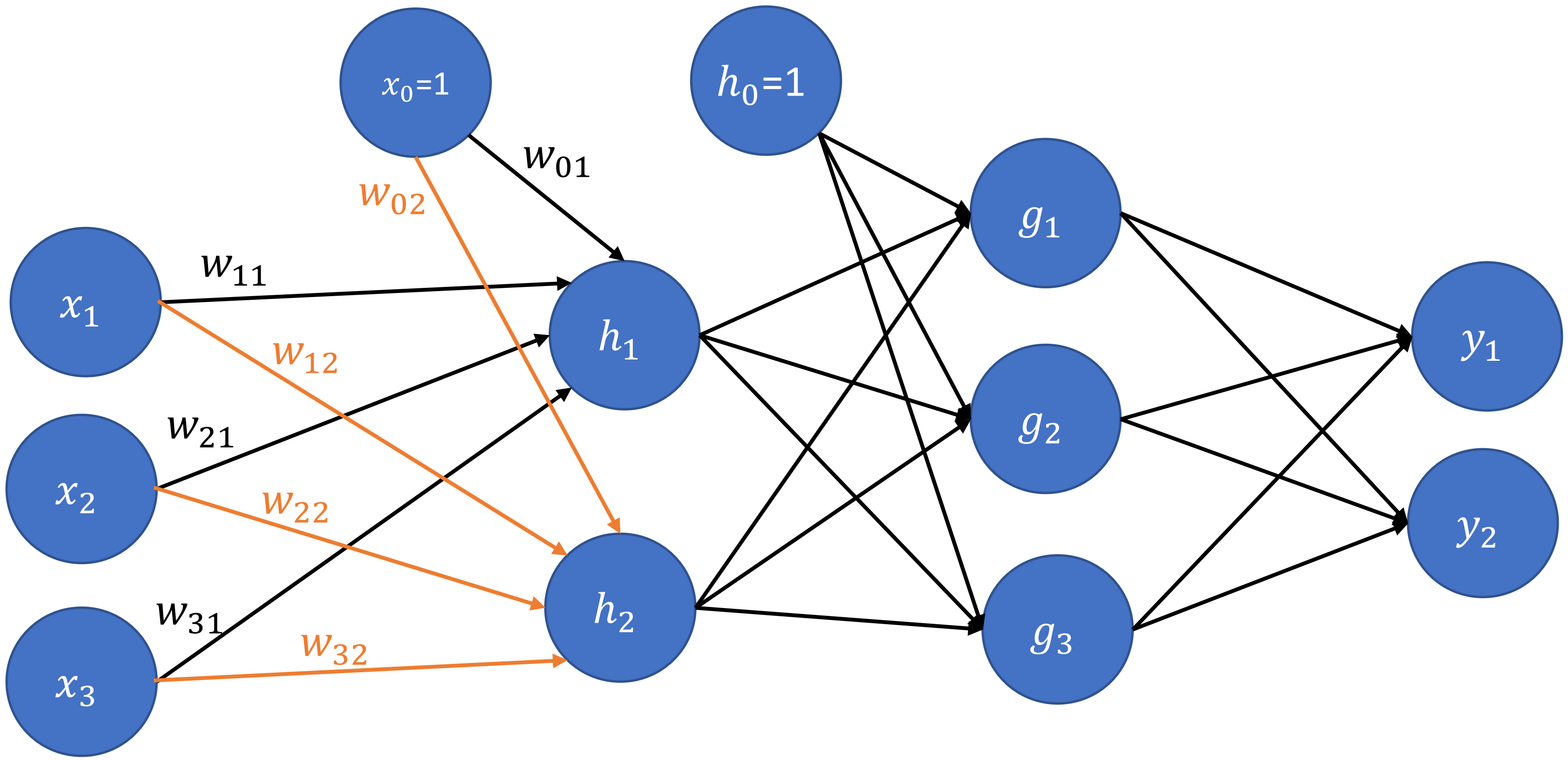
$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \qquad W = \begin{bmatrix} w_{01} & w_{02} \\ w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \end{bmatrix} \qquad \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} = W^T \cdot x$$

Calculation of propagated value



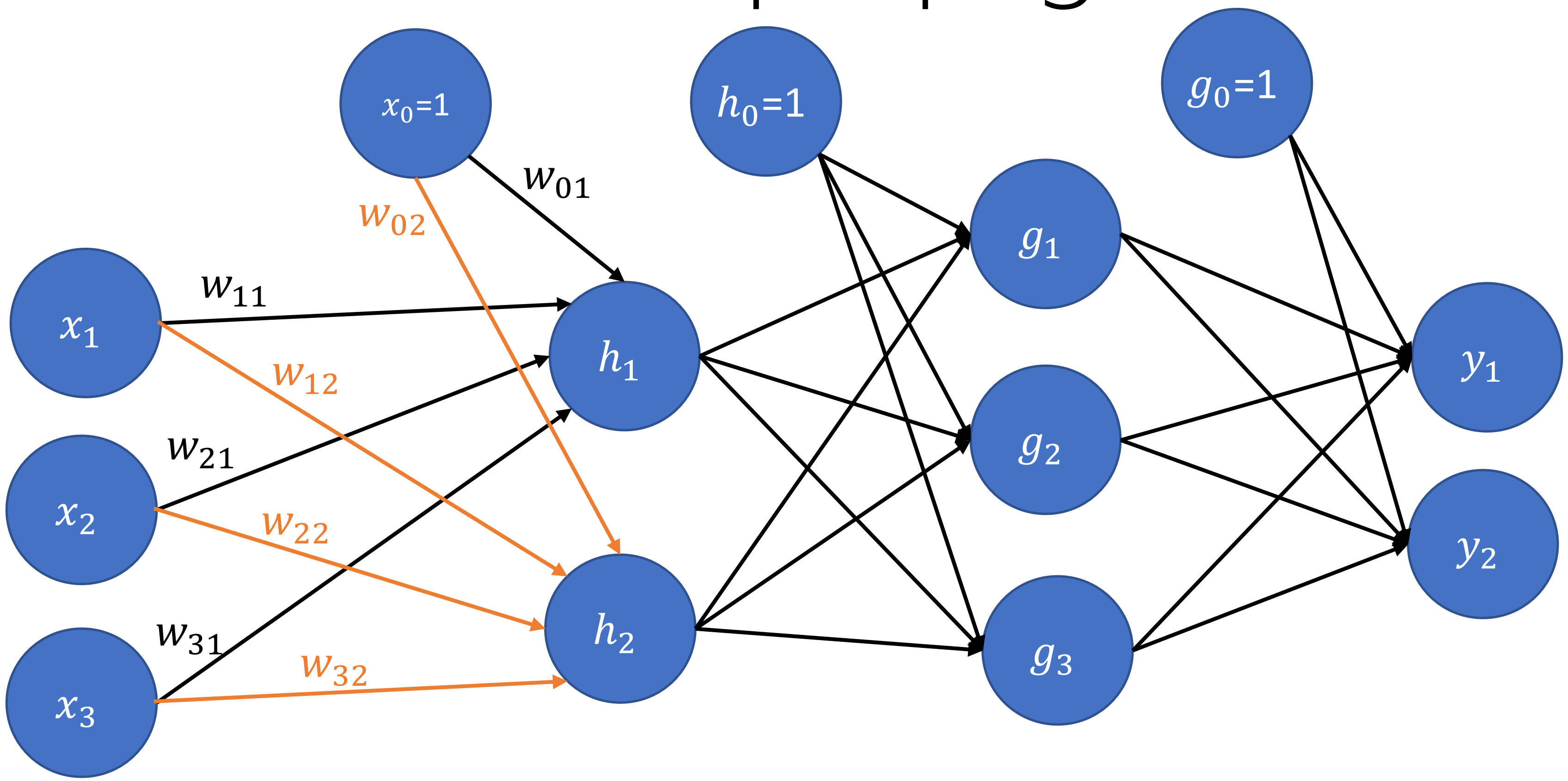
$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \qquad W = \begin{bmatrix} w_{01} & w_{02} \\ w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \end{bmatrix} \qquad \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} = W^T \cdot x$$

Calculation of propagated value



$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \qquad W = \begin{bmatrix} w_{01} & w_{02} \\ w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \end{bmatrix} \qquad \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} = W^T \cdot x$$

Calculation of propagated value



$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \qquad W = \begin{bmatrix} w_{01} & w_{02} \\ w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \end{bmatrix} \qquad \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} = W^T \cdot x$$

MLP – Caution with number of layers

- Various conventions about number of layers
- Sometimes including/excluding input layer
- In scope of this course: Excluding input layer

- 3 layer Perceptron:
  - 1 input layer
  - 2 hidden layers
  - 1 output layer

Revision: Lecture

- What are the hyperparameters of a Multilayer Perceptron?

Revision: Lecture

- What are the hyperparameters of a Multilayer Perceptron?
    1. Number of layers
    2. Number of neurons per layer
    3. Weights
    4. Activation/Propagation functions

Revision: Lecture

- What are the hyperparameters of a Multilayer Perceptron?
  1. Number of layers
  2. Number of neurons per layer
  3. Weights
  4. Activation/Propagation functions

A: all

B: 1,2,4

C: 1,2

D: 1,2,3

Revision: Lecture

- What are the hyperparameters of a Multilayer Perceptron?
  1. Number of layers
  2. Number of neurons per layer
  3. Weights
  4. Activation/Propagation functions

A: all

B: 1,2,4

C: 1,2

D: 1,2,3