ML Intro

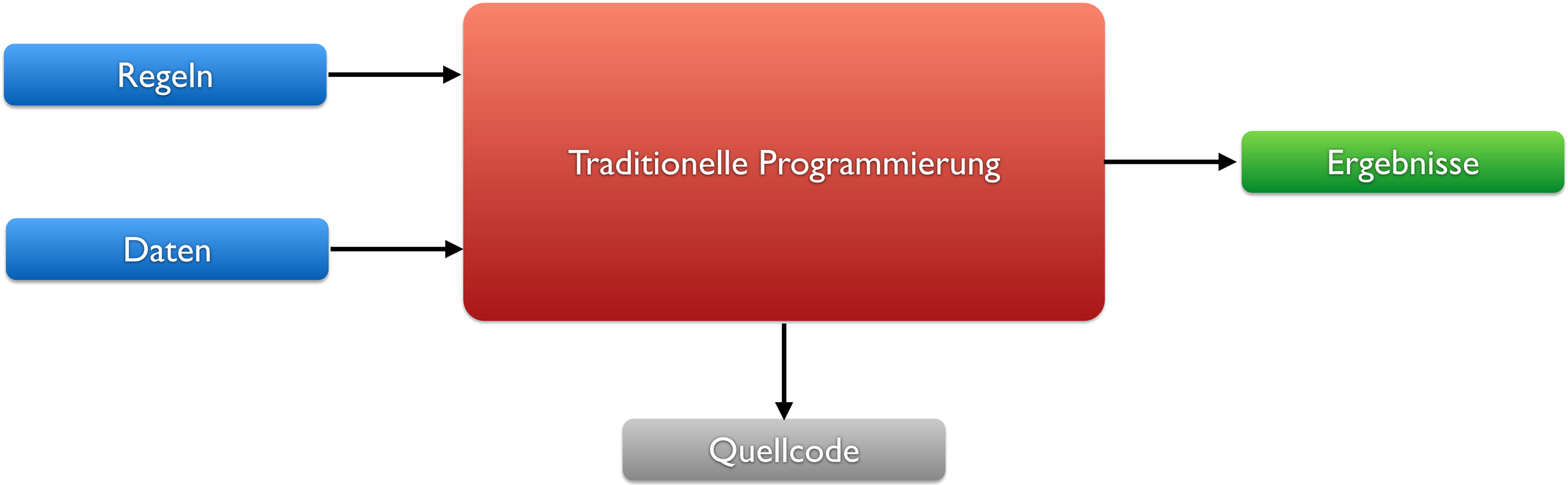
Tiefe neuronale Netze in 30 Minuten

Michal Harakal, Javaland 2024

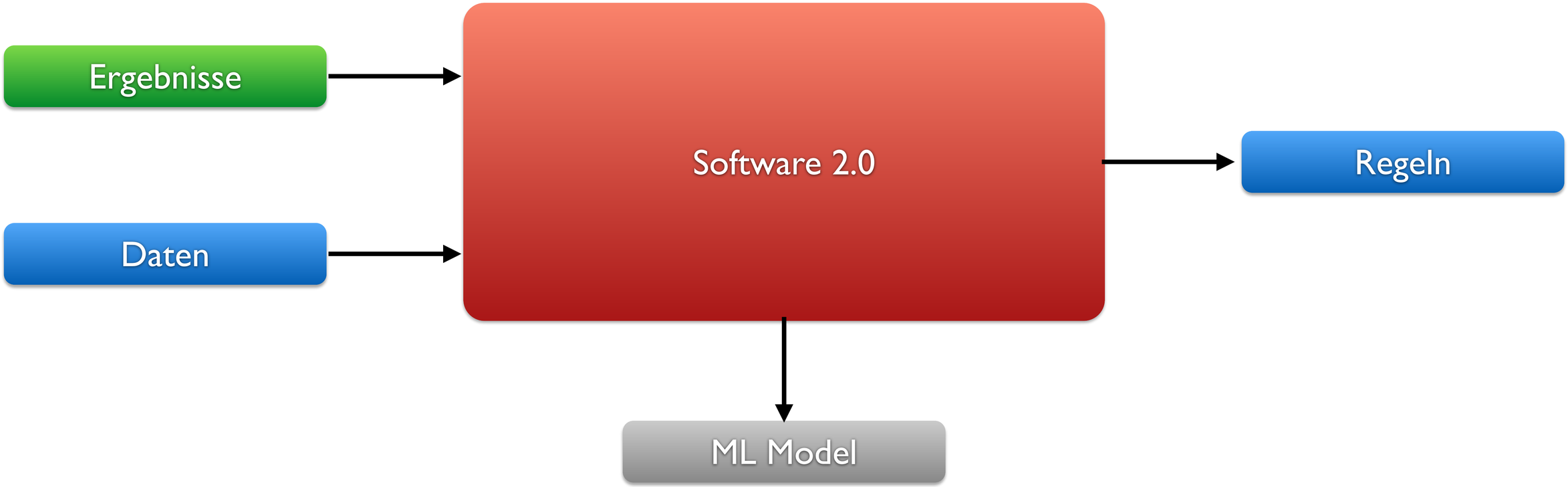
Gradient descent can write code better than you. I'm sorry.

Andrej Karpathy, 4.8.2017

Software 2.0

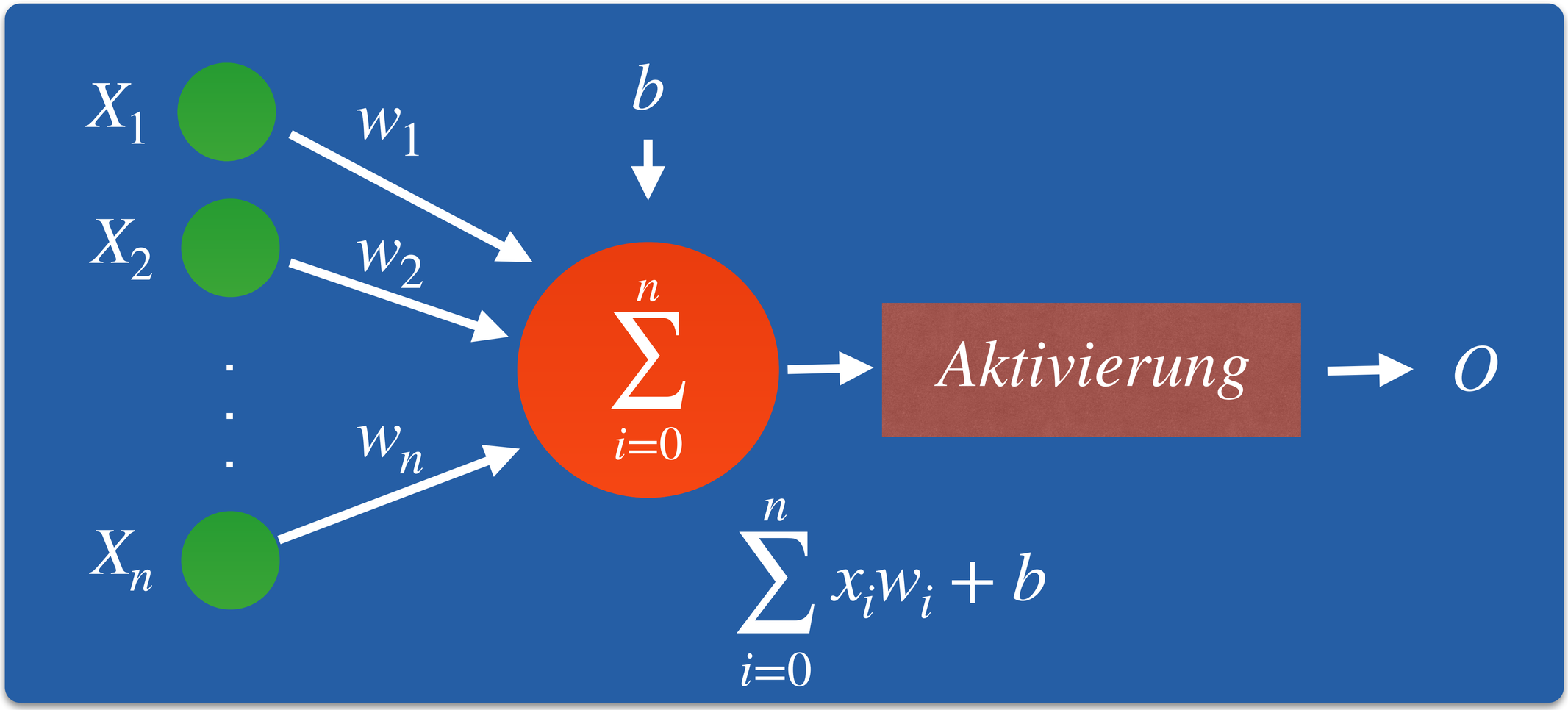


Traditionele Programmierung

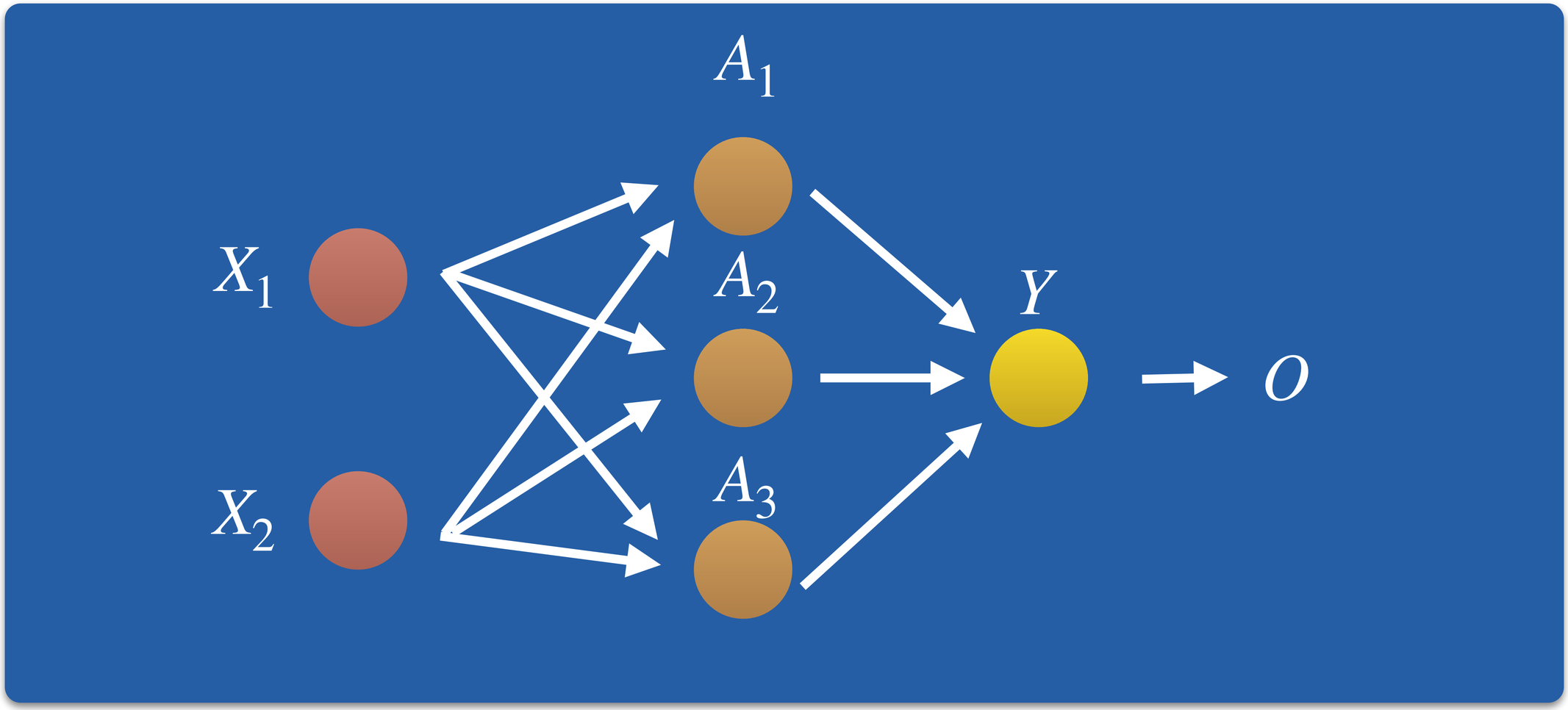


Software 2.0

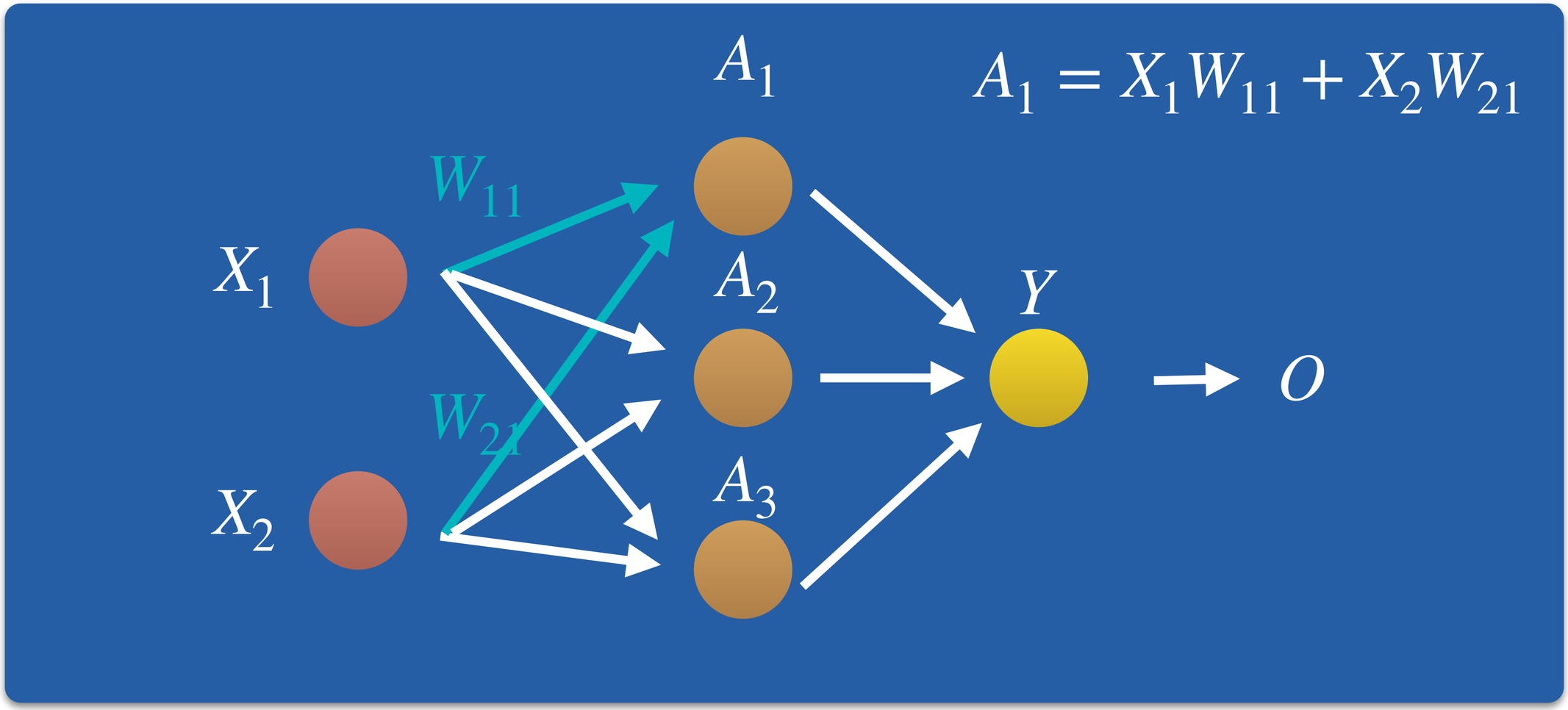
Künstliches neuronales Netz



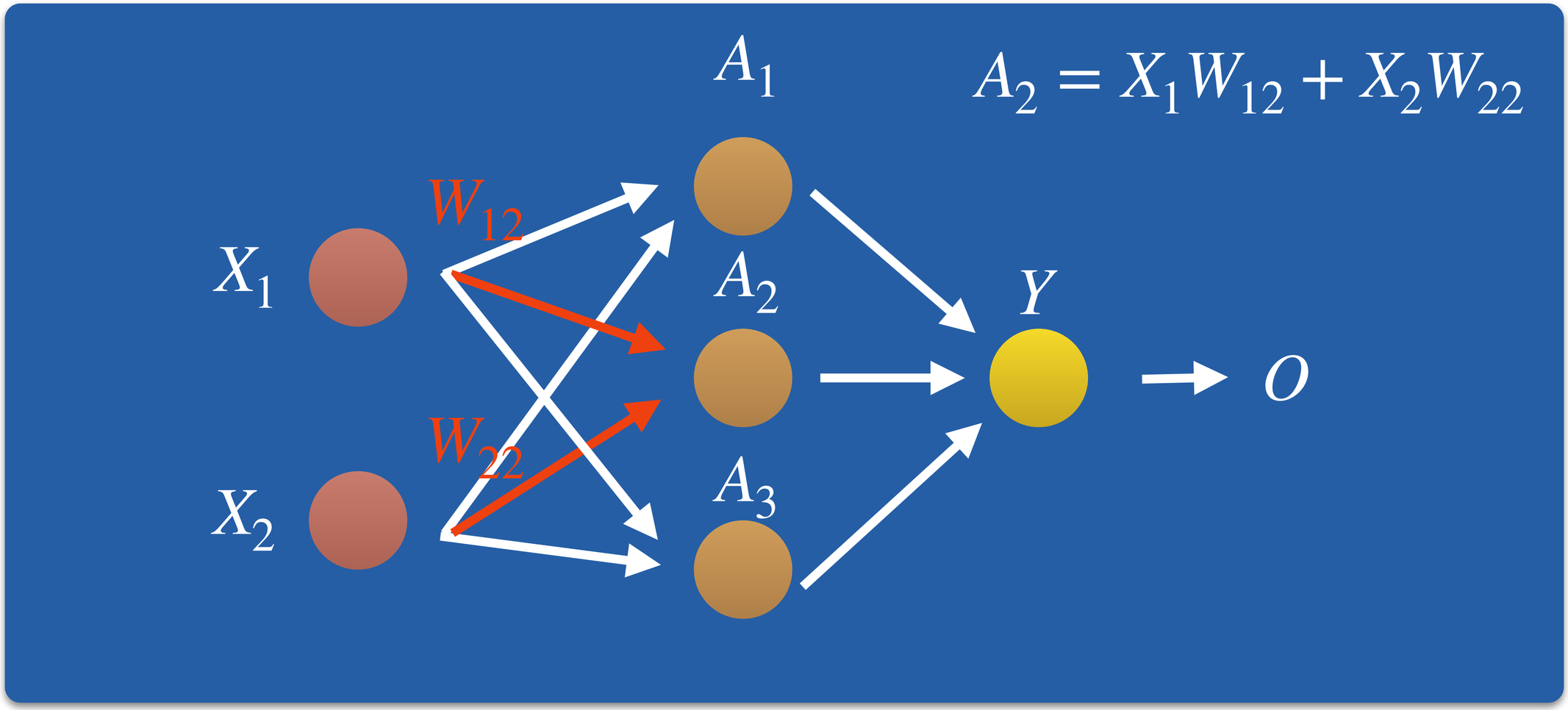
Neuron



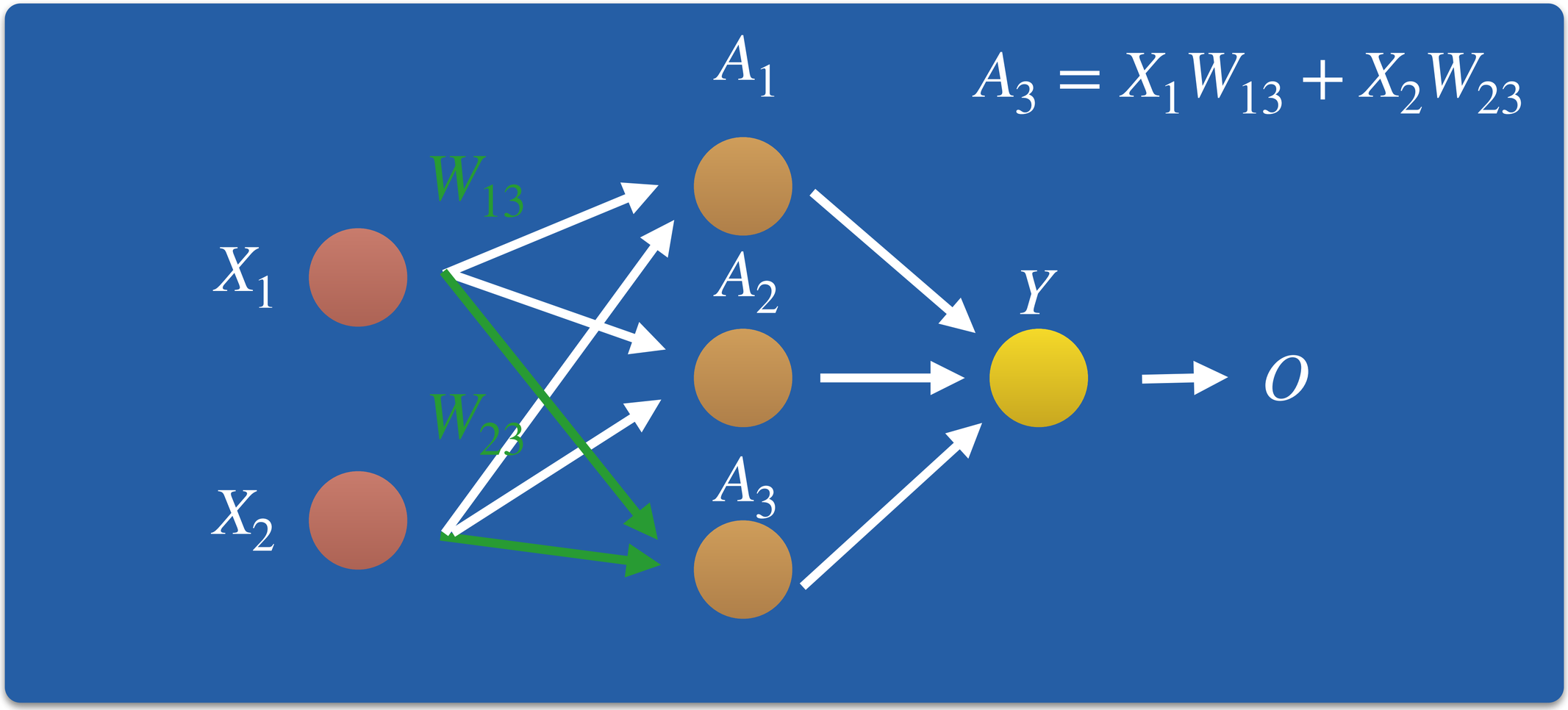
Tiefes neuronales Netz



Forward Propagation



Forward Propagation



Forward Propagation

$$A_{11} = X_1 W_{11} + X_2 W_{21}$$

$$A_{12} = X_1 W_{12} + X_2 W_{22}$$

$$A_{13} = X_1 W_{13} + X_2 W_{23}$$

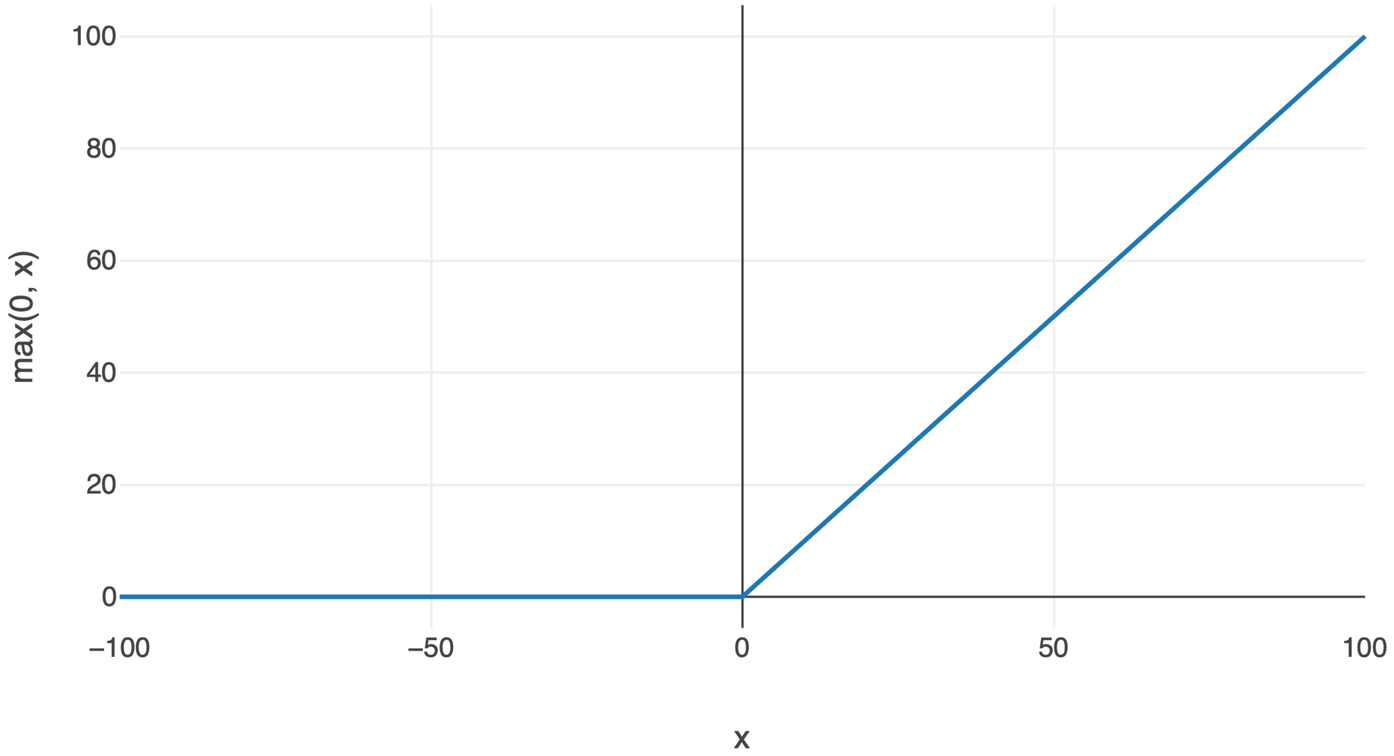
$$W = \begin{bmatrix} W_{11} & W_{12} & W_{13} \\ W_{21} & W_{22} & W_{23} \end{bmatrix}$$

$$X = \begin{bmatrix} X_1 & X_2 \end{bmatrix} \qquad A = \begin{bmatrix} A_{11} & A_{12} & A_{12} \end{bmatrix}$$

Matrizen Multiplikation

$$A = X \,.\, W$$

ReLU



Aktivierungsfunktion - Gleichricher

```
class Linear() {
   val weights:Tensor
   val bias:Tensor

   fun forward(input: Tensor): Tensor {
      val output = input.matmul(weight.value.t()) +
      bias.value
      return output
   }
}
```

Forward Propagation

Künstliches neuronales Netz tranieren