Neural Networks

Krisztian Buza

Department of Artificial Intelligence
Eötvös Loránd University
Budapest, Hungary

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Announcements

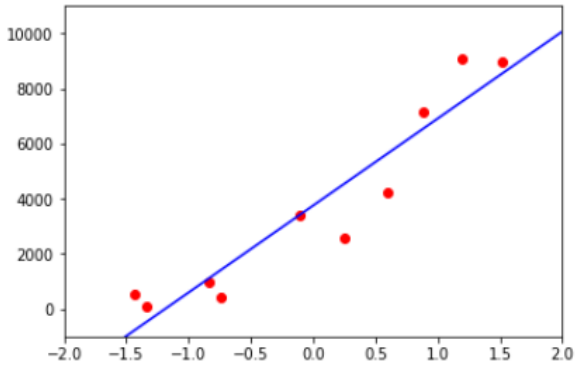
Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Announcements

- Quick Test Replacement Test (QTRT)
  - In the same session with the „big test“
  - 11th Dec: „big test“ (60 minutes) + ~~QTRT (30 minutes)~~
  - Due to multiple requests, QTRT will be organized between 16th Dec and 20th Dec
  - You may collect maximal 14 points which replace the result of the 7 worst quick tests
- PIN code for querying the results of tests
- Form groups for presentation about ethical, social, economical, technical aspect of AI

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Calculation of Root Mean Square Error



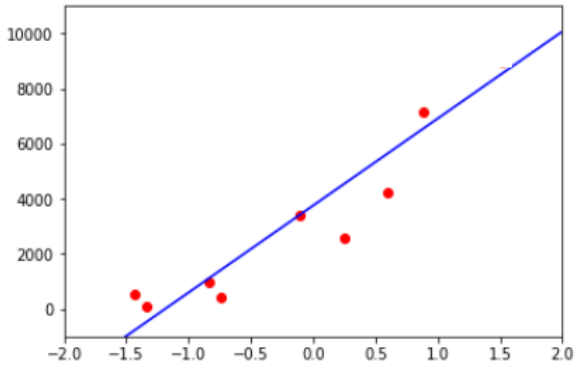
y	$\hat{y}$
549	-770
100	-470
976	1132
441	1433
3401	3435
2600	4537
4241	5638
7150	6539
9100	7541
9000	8542

n = 10

$$\frac{1}{10}((549-(-770))^2+(100-(-470))^2+$$
$$+(976-1132)^2+(441-1433)^2+$$
$$+(3401-3435)^2+(2600-4537)^2+$$
$$+(4241-5638)^2+(7150-6539)^2+$$
$$+(9100-7541)^2+(9000-8542)^2)$$

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Calculation of Root Mean Square Error



y	$\hat{y}$
549	-770
100	-470
976	1132
441	1433
3401	3435
2600	4537
4241	5638
7150	6539
~~9100~~	~~7541~~
~~9000~~	~~8542~~

$n = 8$

$$\sqrt{\frac{1}{\cancel{10}\,8}\left((549-(-770))^2+(100-(-470))^2+\right.}$$
$$+(976-1132)^2+(441-1433)^2+$$
$$+(3401-3435)^2+(2600-4537)^2+$$
$$+(4241-5638)^2+(7150-6539)^2+$$
$$\left.\cancel{+(9100-7541)^2+(9000-8542)^2}\right)$$

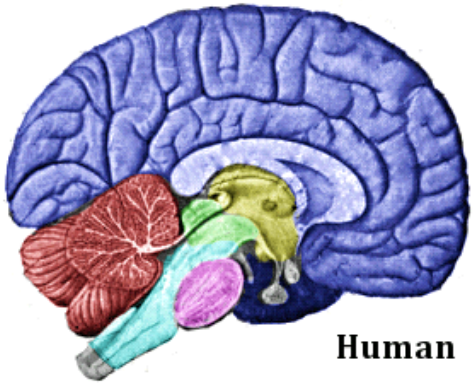
Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Artificial Neural Networks

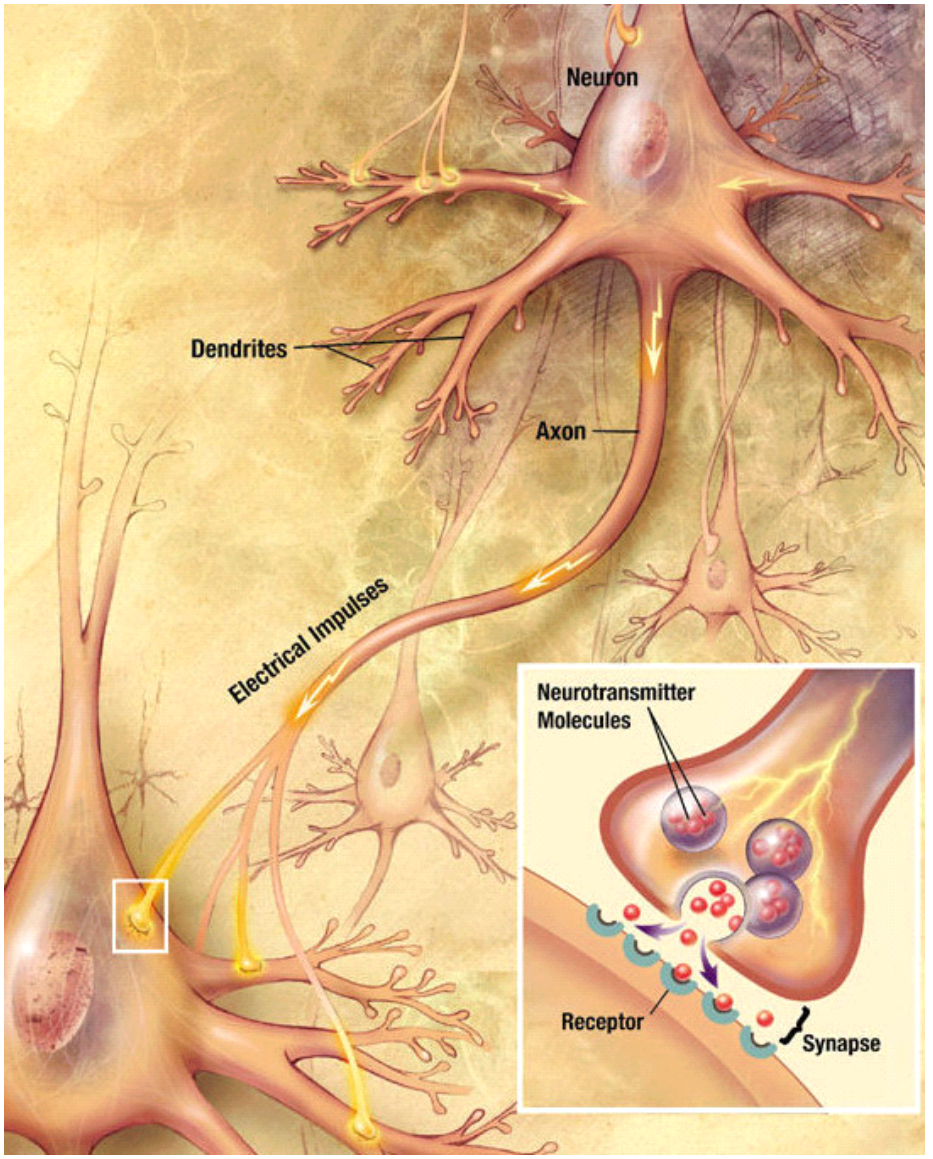
Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Motivation

- Complex classification tasks with many variables
  (e.g. image classification)
- Biological motivation
  (single learning algorithm, structure, spikes, learned features)
- State-of-the-art learning algorithms



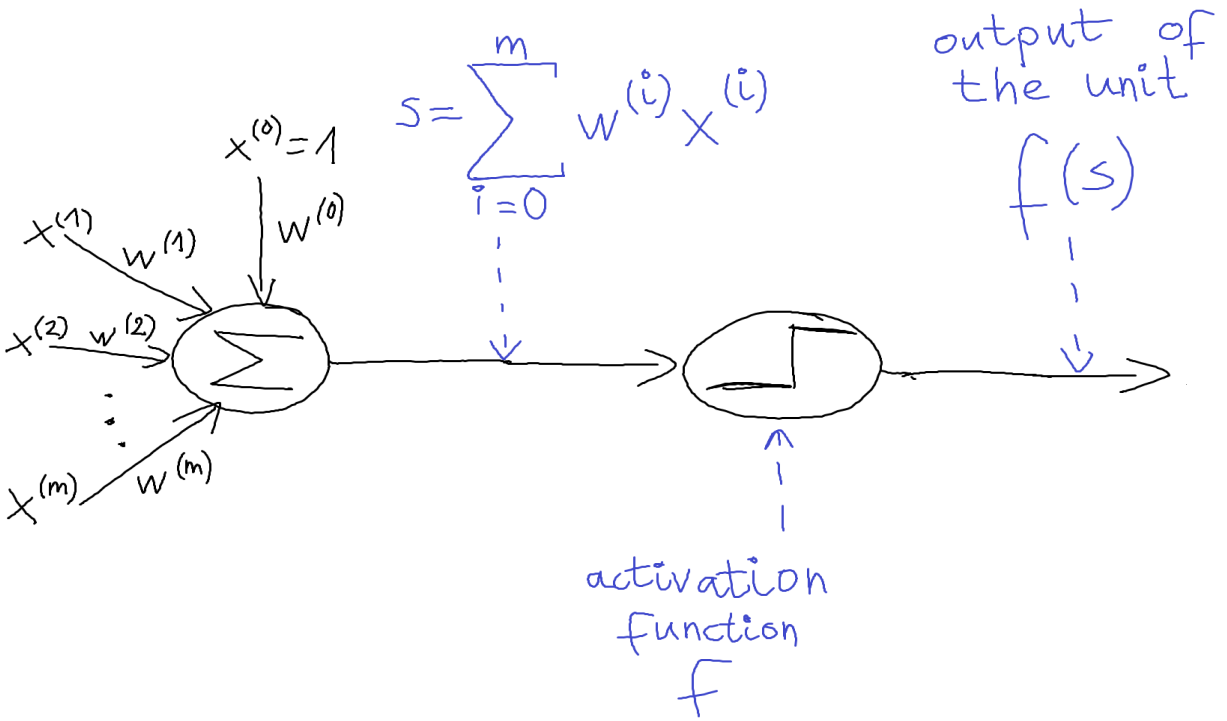
By Vertebrate-brain-regions.png: Looie496derivative work: Looie496 (Vertebrate-brain-regions.png) [Public domain], via Wikimedia Commons



By user:Looie496 created file, US National Institutes of Health, National Institute on Aging created original [Public domain], via Wikimedia Commons

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

A Neural Unit



$x^{(1)}, x^{(2)}, \ldots, x^{(m)}$ = inputs of the unit

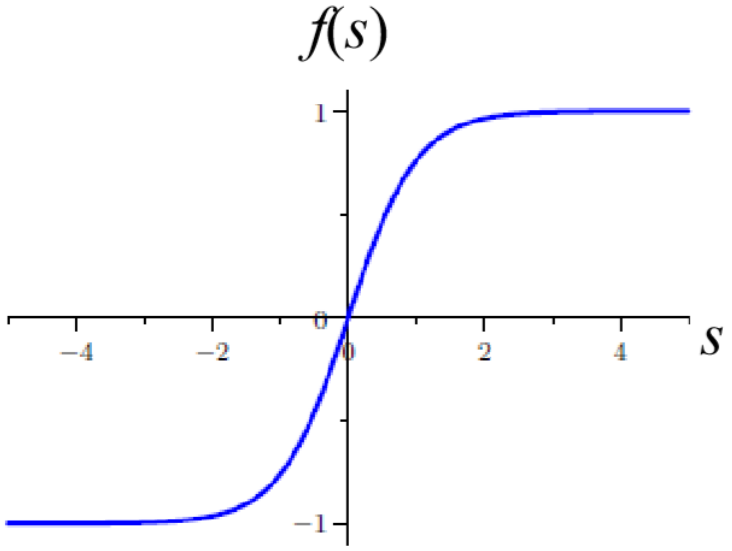
$w^{(1)}, w^{(2)}, \ldots, w^{(m)}$ = weights of $x^{(1)}, x^{(2)}, \ldots, x^{(m)}$

$w^{(0)}$ = bias weight

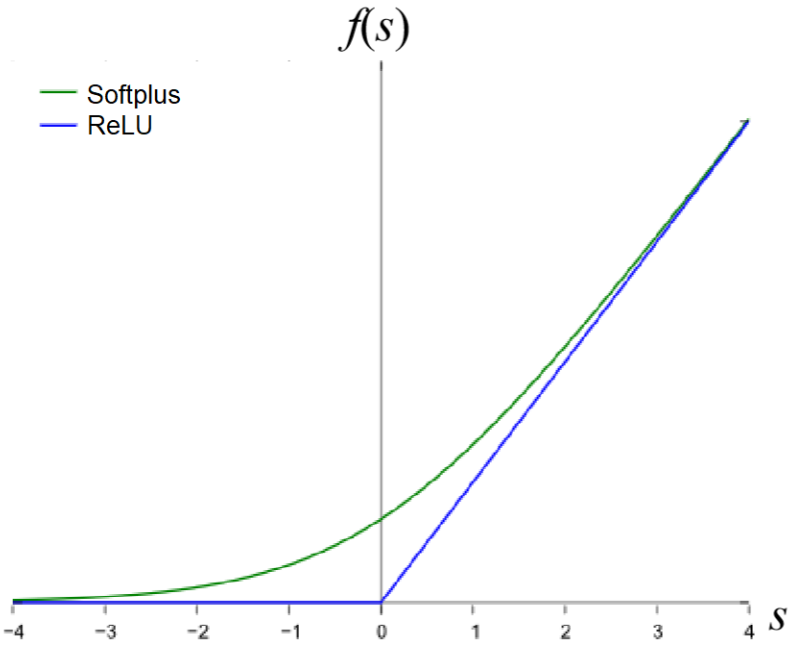
Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Activation Functions

- sigmoid → logistic regression
- ReLU → linear regression with nonnegative target

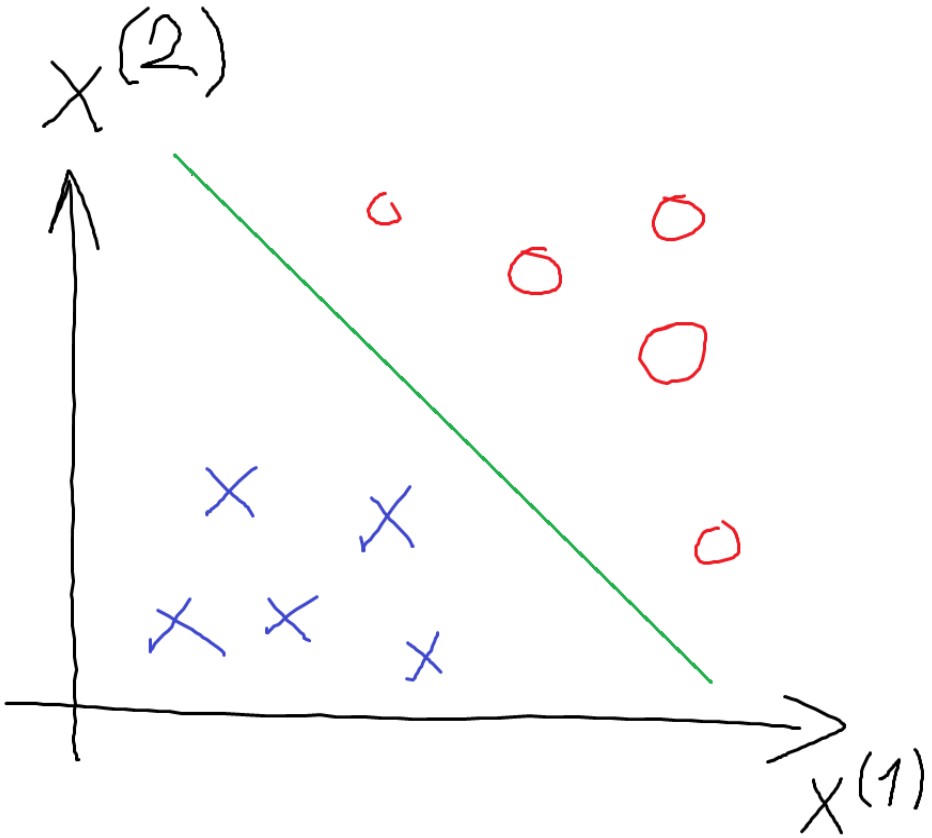


Activation Functions	
Linear	$f(s) = s$
Sigmoid	$f(s) = (1+e^{-s})^{-1}$
Hyperbolic tangent	$f(s) = \tanh(s)$
Softsign	$f(s) = s((1+\|s\|)^{-1})$
Rectifier Linear Unit (ReLU)	$f(s) = \max(0, s)$
Softplus	$f(s) = \ln(1+e^{s})$
...	...



Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Single Sigmoidal Unit – Decision Boundary



Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

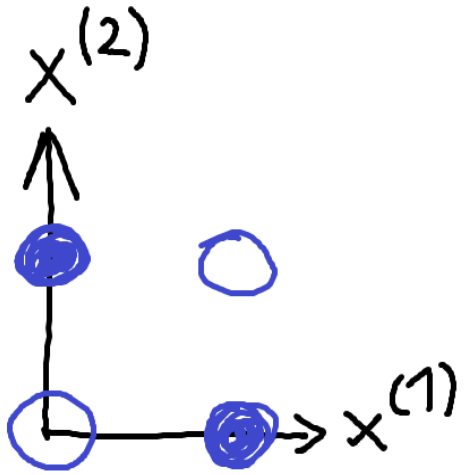
Can a Sigmoidal Unit Learn the XOR Function?

$x^{(1)}$	$x^{(2)}$	$x^{(1)}$ XOR $x^{(2)}$
0	0	0
0	1	1
1	0	1
1	1	0

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Can a Sigmoidal Unit Learn the XOR Function?

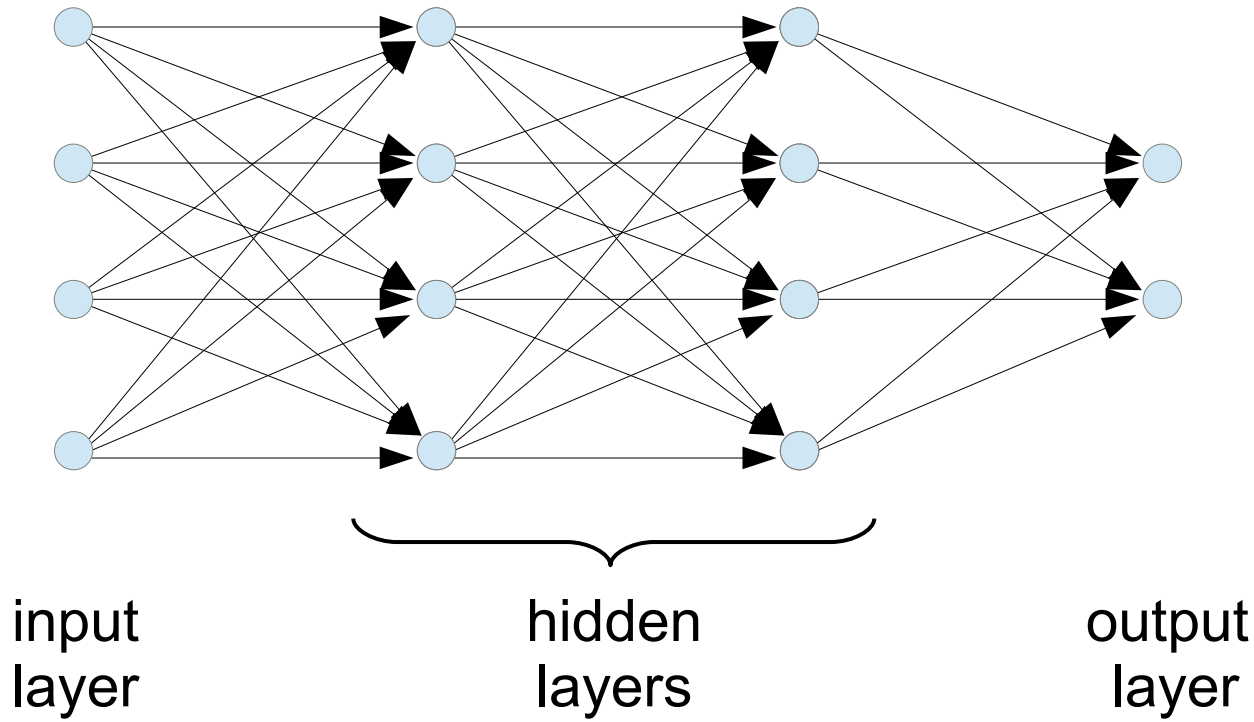
$x^{(1)}$	$x^{(2)}$	$x^{(1)}$ XOR $x^{(2)}$
0	0	0
0	1	1
1	0	1
1	1	0



Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

How does a Neural Network Work?

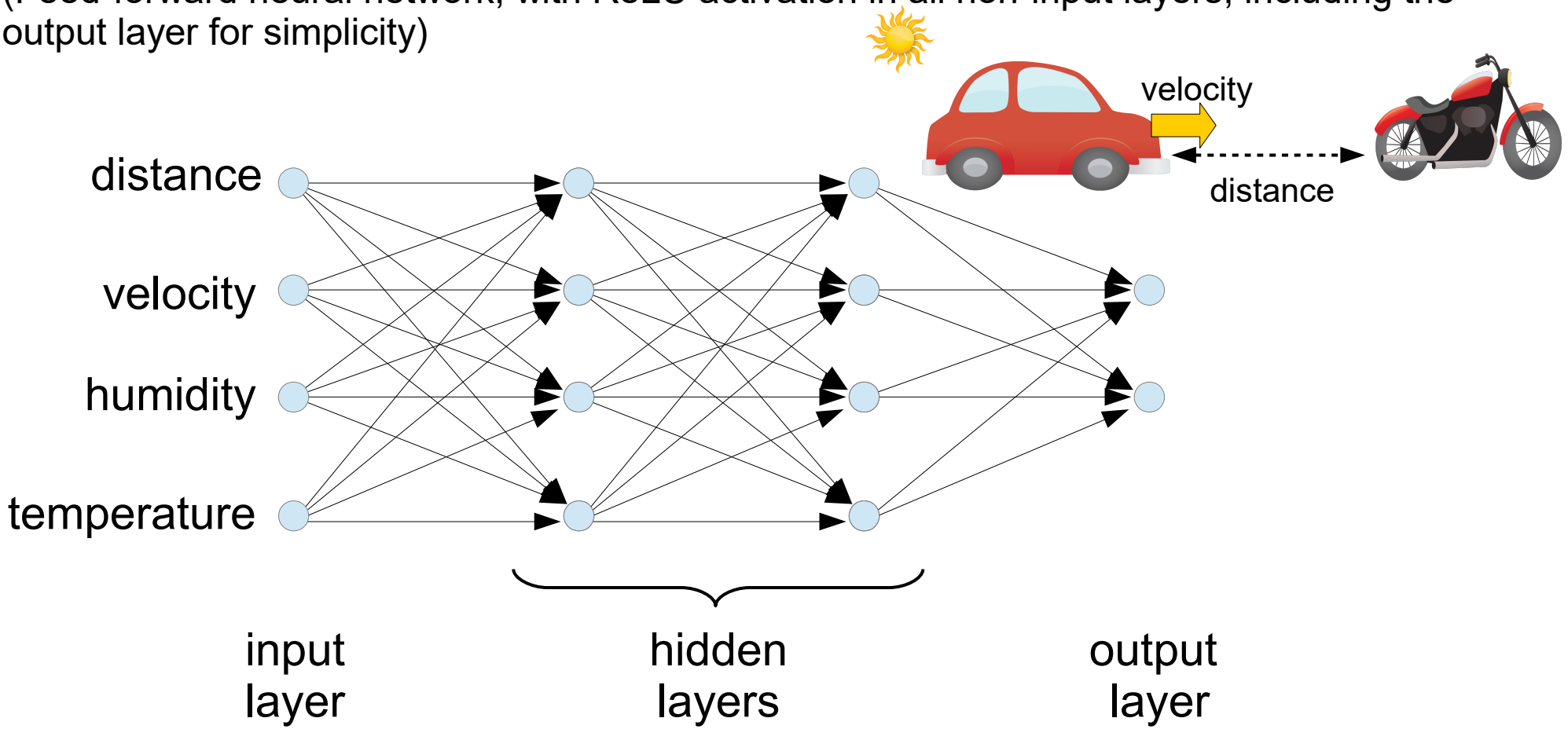
(Feed-forward neural network, with ReLU activation in all non-input layers, including the output layer for simplicity)



Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

How does a Neural Network Work?

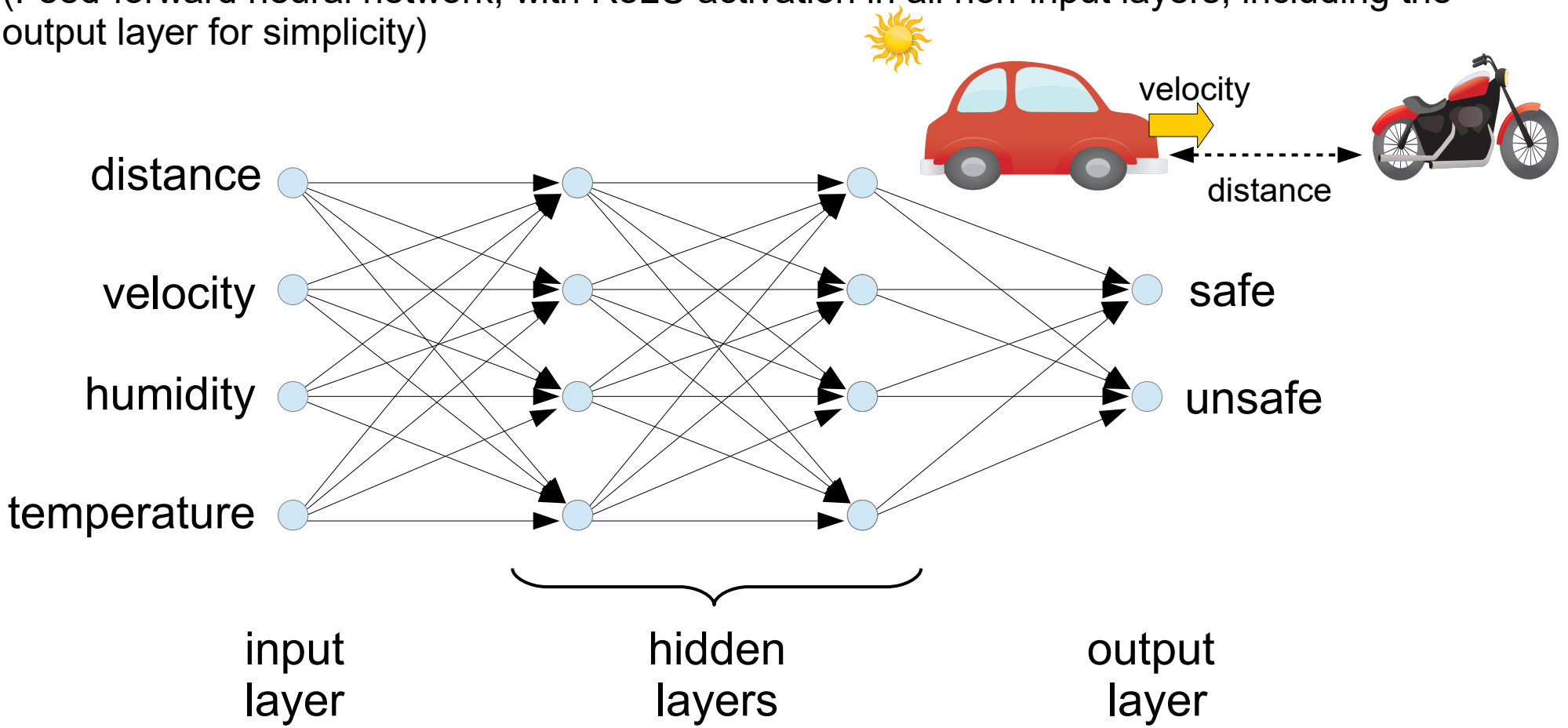
(Feed-forward neural network, with ReLU activation in all non-input layers, including the output layer for simplicity)



Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

How does a Neural Network Work?

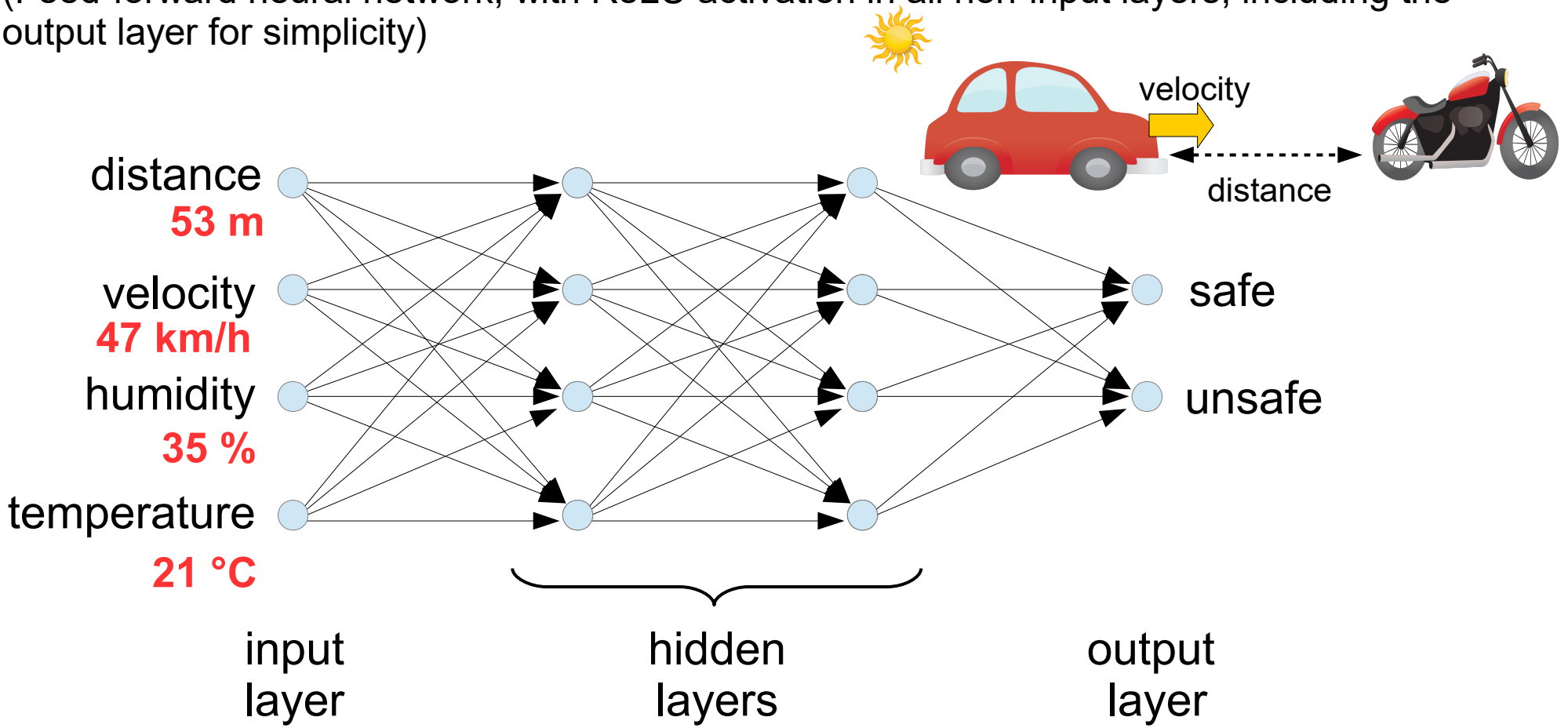
(Feed-forward neural network, with ReLU activation in all non-input layers, including the output layer for simplicity)



Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

How does a Neural Network Work?

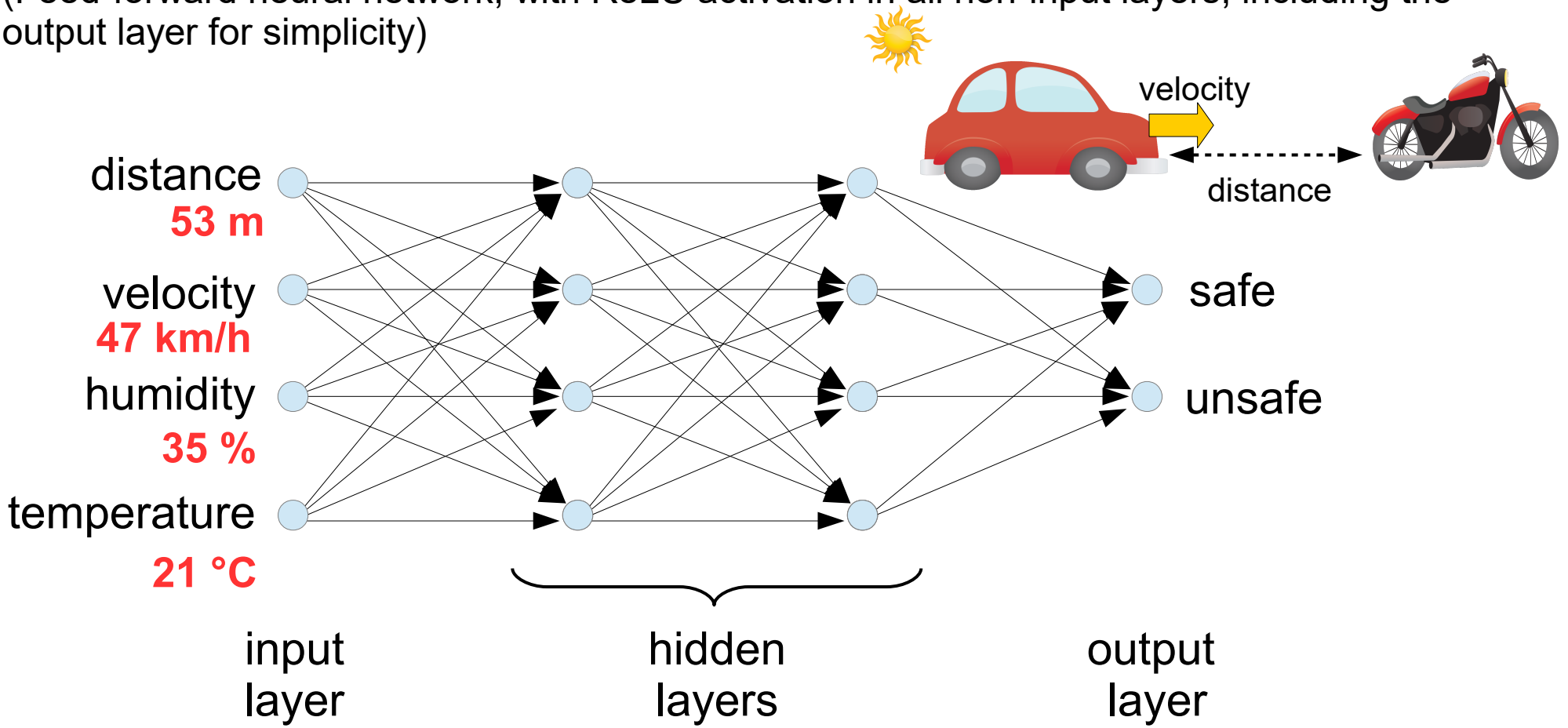
(Feed-forward neural network, with ReLU activation in all non-input layers, including the output layer for simplicity)



Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

How does a Neural Network Work?

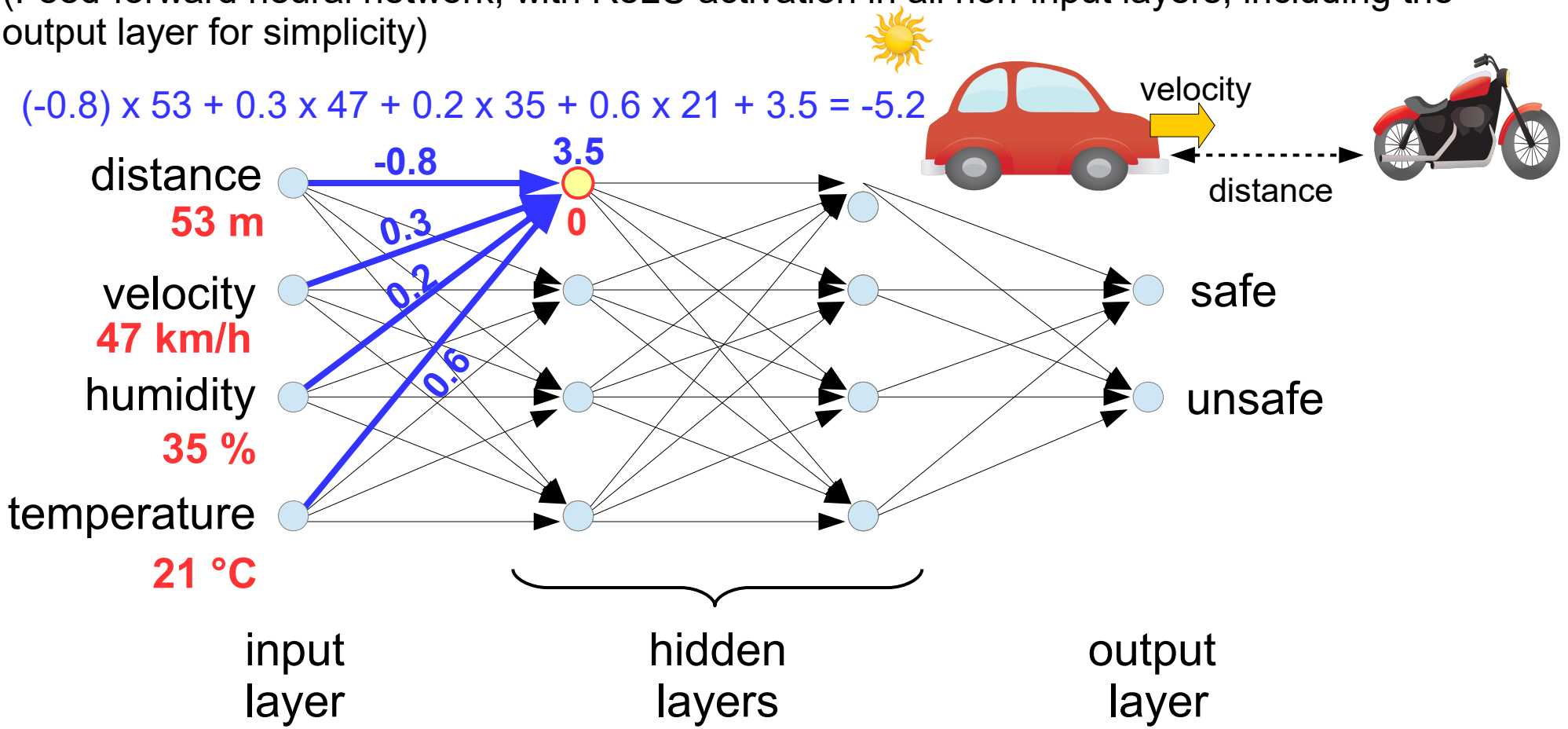
(Feed-forward neural network, with ReLU activation in all non-input layers, including the output layer for simplicity)



Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

How does a Neural Network Work?

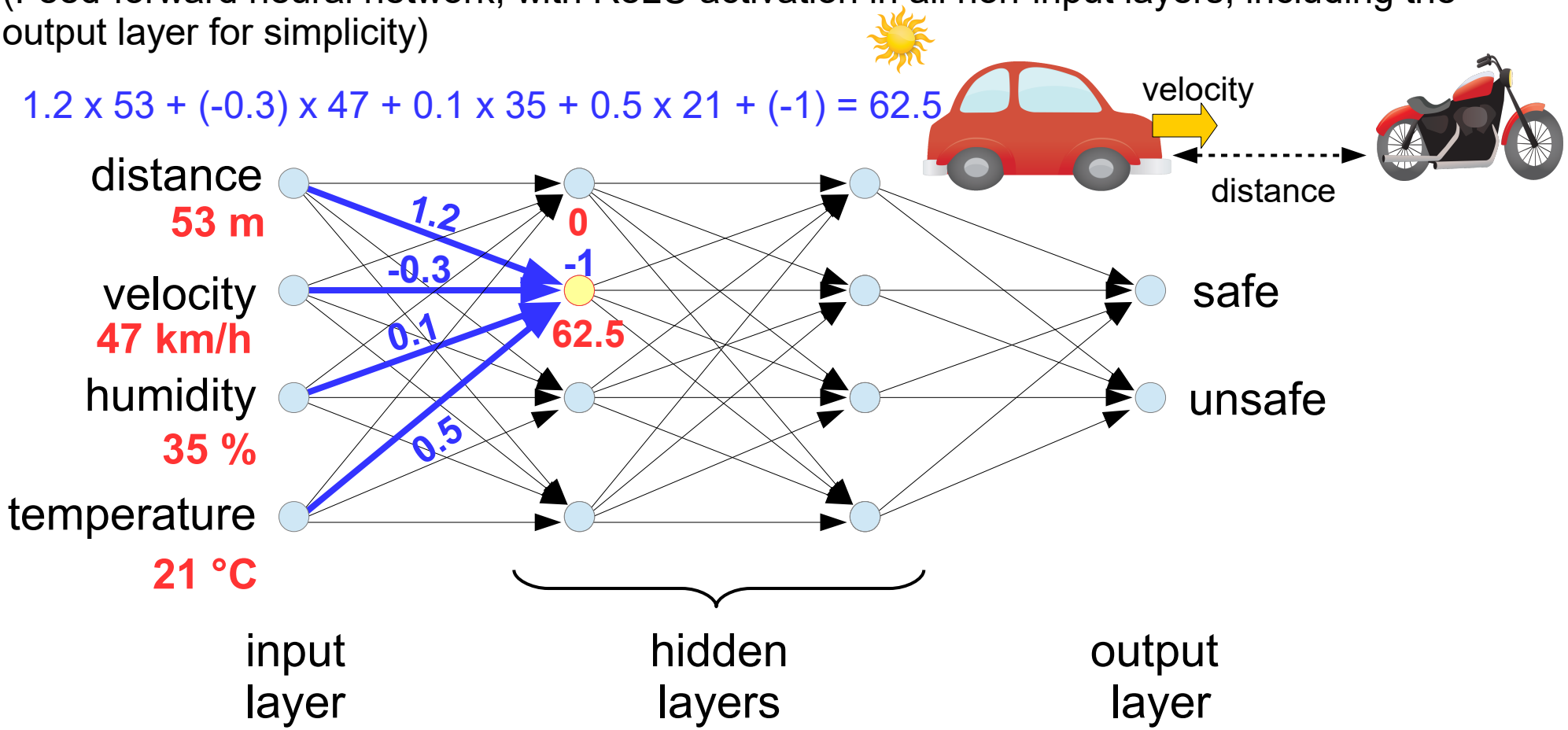
(Feed-forward neural network, with ReLU activation in all non-input layers, including the output layer for simplicity)



Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

How does a Neural Network Work?

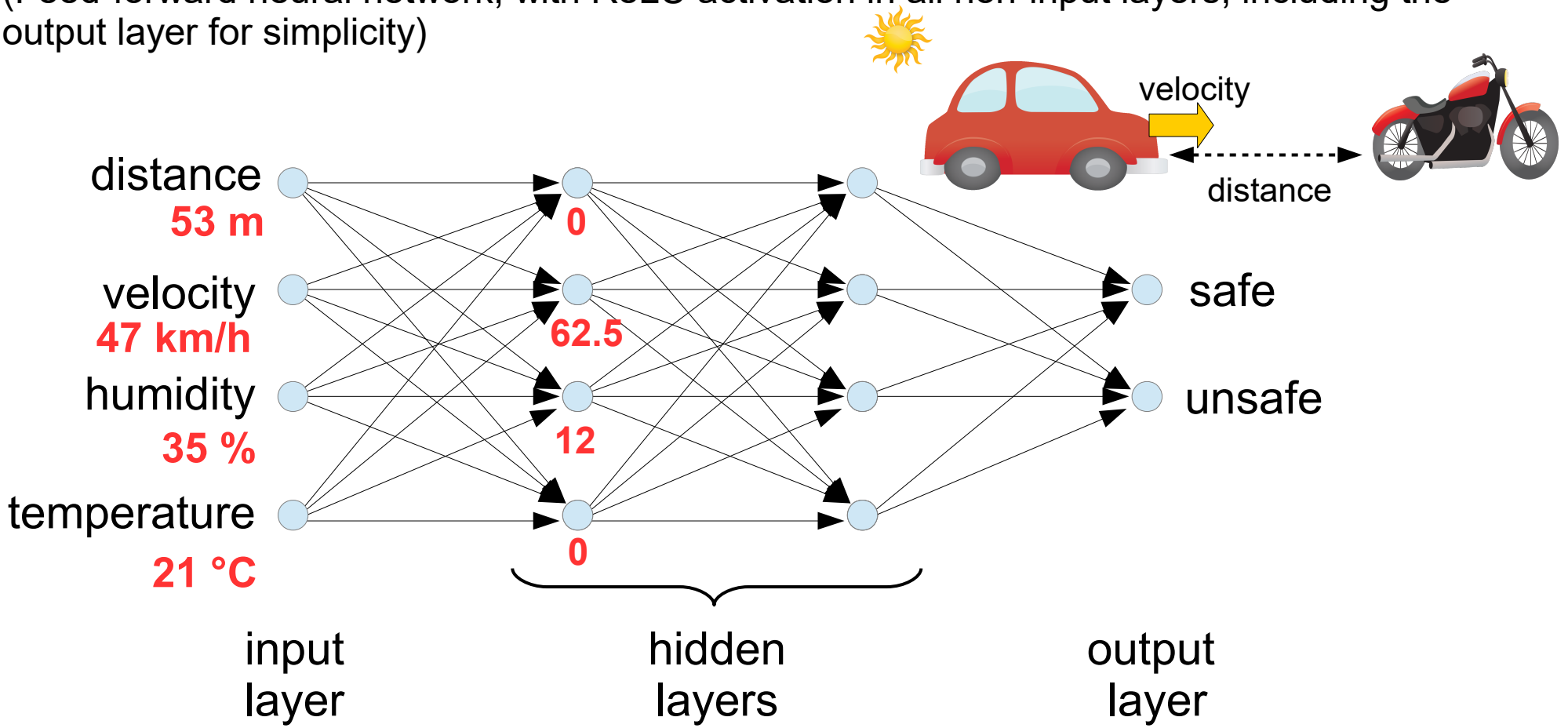
(Feed-forward neural network, with ReLU activation in all non-input layers, including the output layer for simplicity)



Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

How does a Neural Network Work?

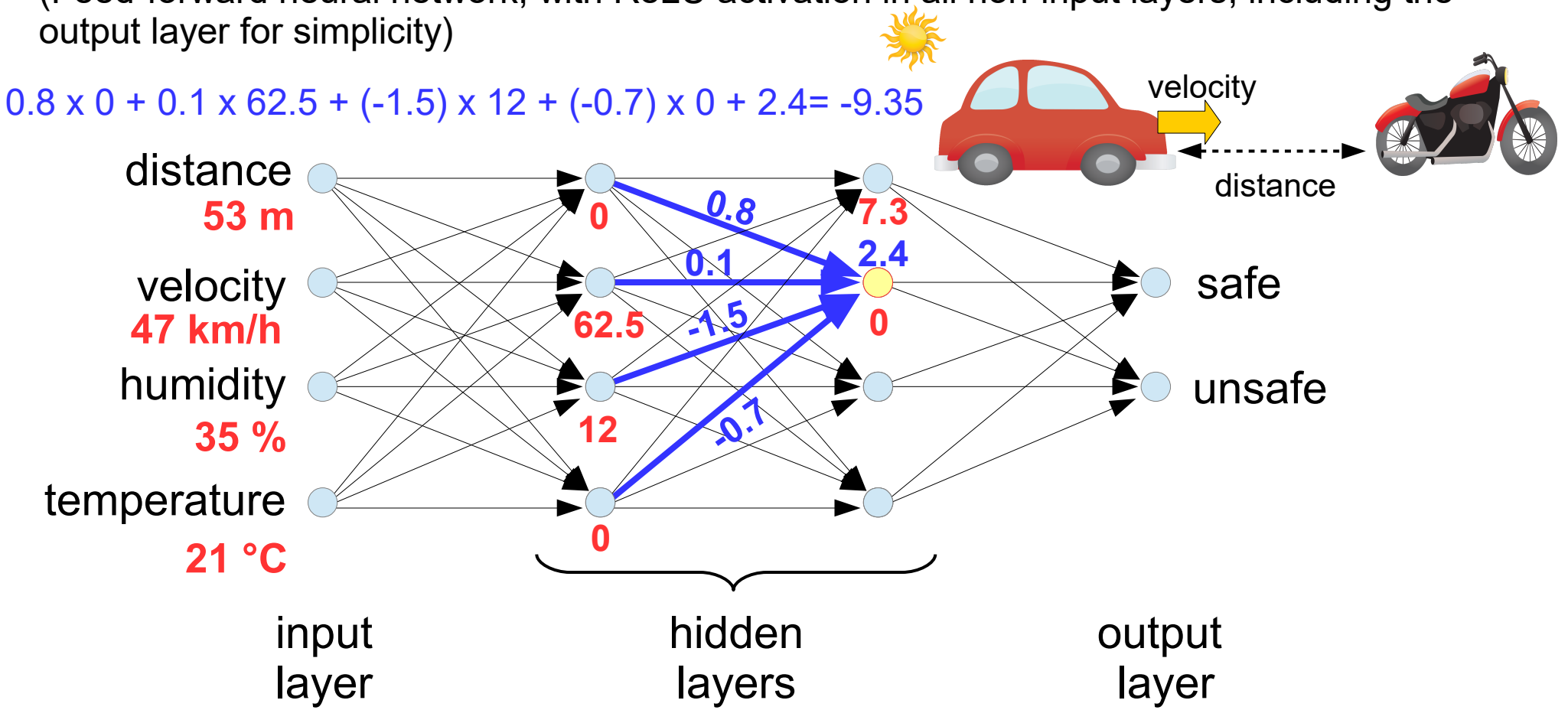
(Feed-forward neural network, with ReLU activation in all non-input layers, including the output layer for simplicity)



Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

How does a Neural Network Work?

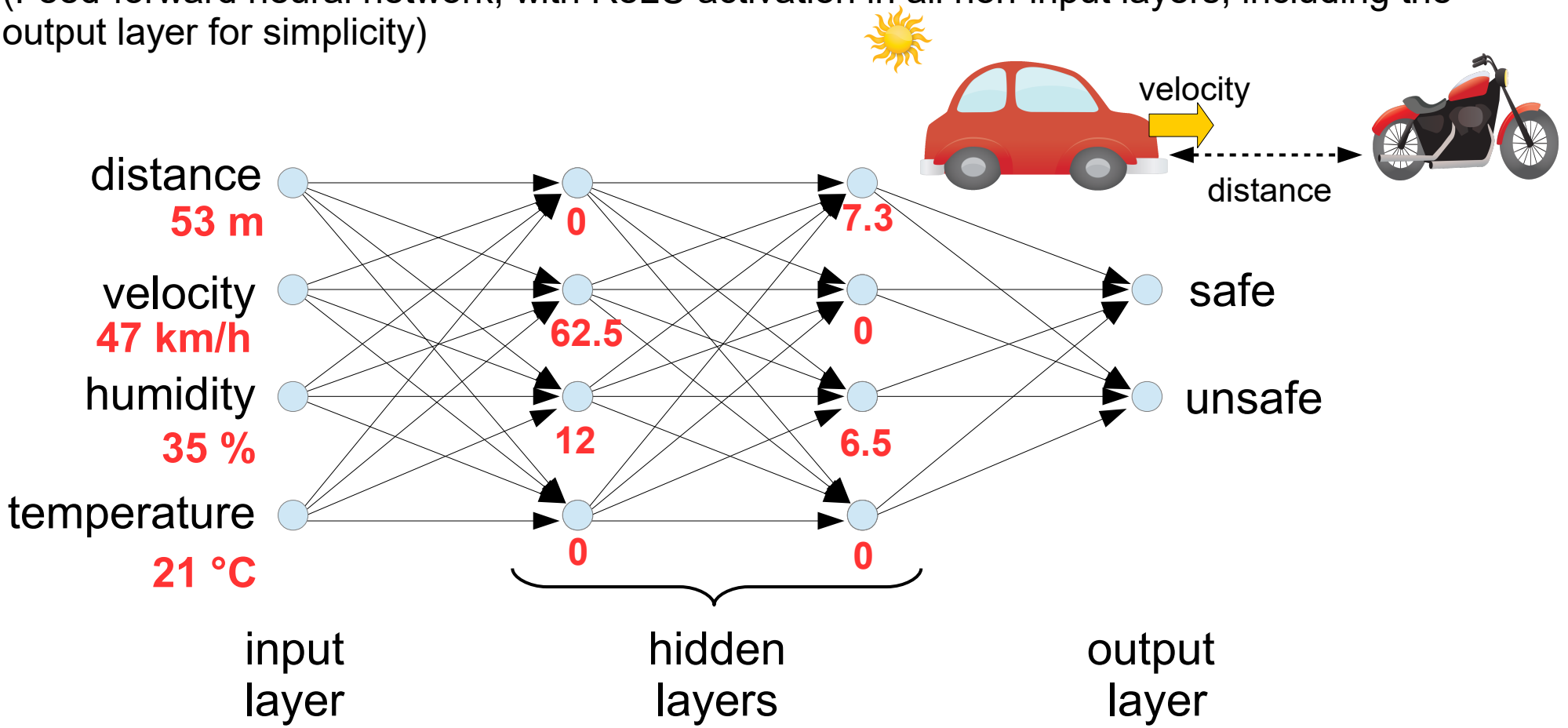
(Feed-forward neural network, with ReLU activation in all non-input layers, including the output layer for simplicity)



Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

How does a Neural Network Work?

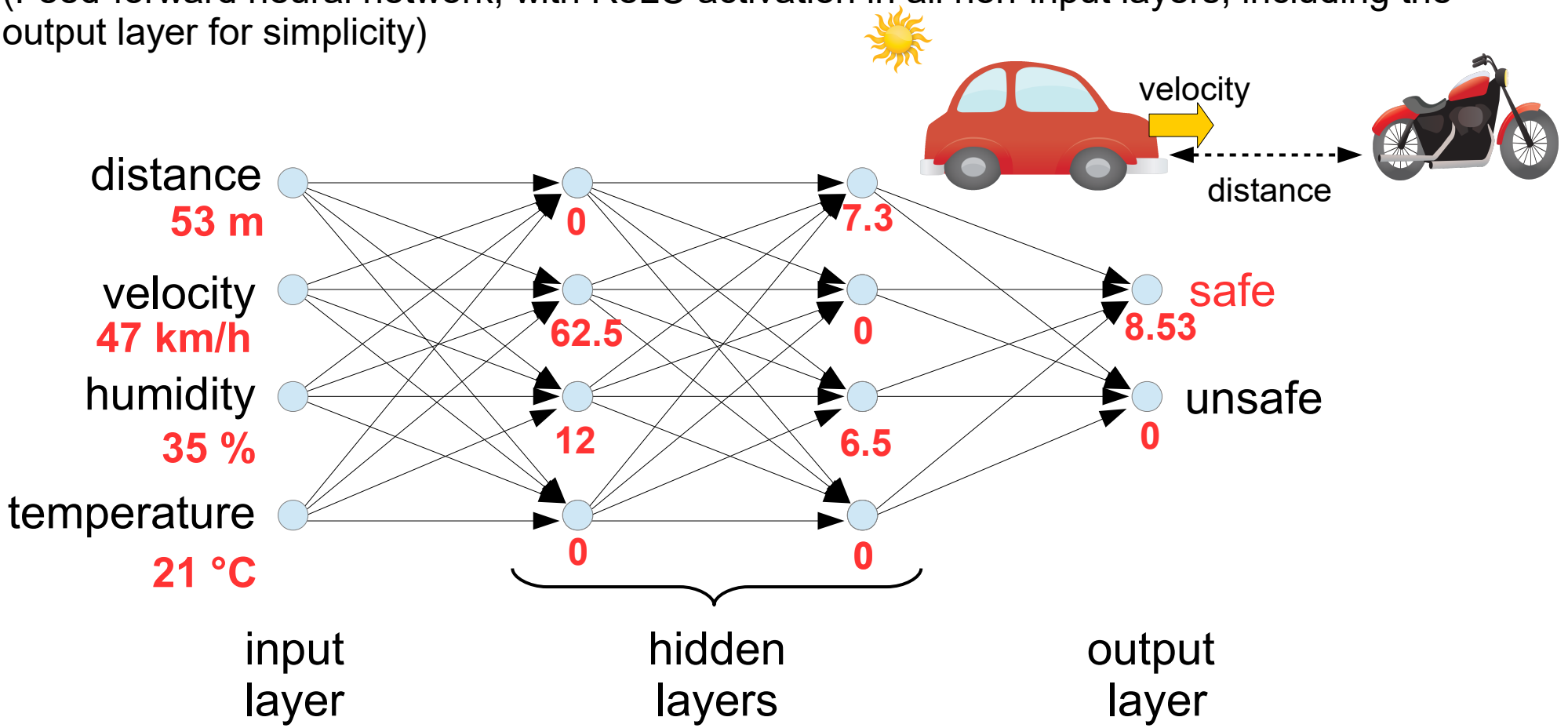
(Feed-forward neural network, with ReLU activation in all non-input layers, including the output layer for simplicity)



Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

How does a Neural Network Work?

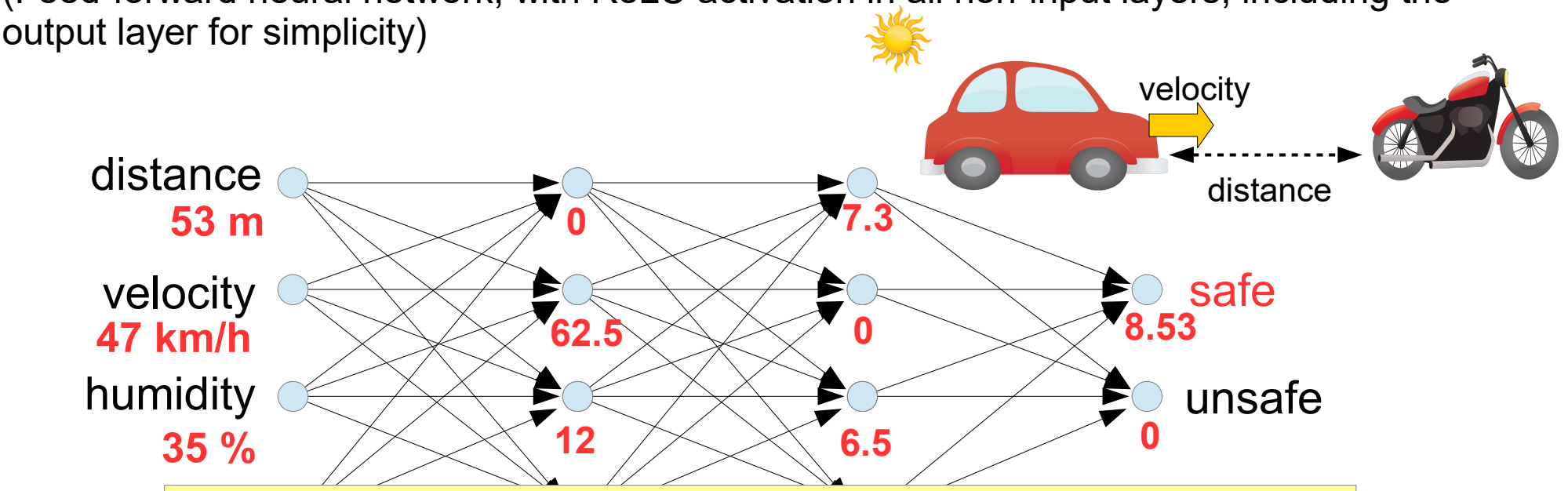
(Feed-forward neural network, with ReLU activation in all non-input layers, including the output layer for simplicity)



Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

How does a Neural Network Work?

(Feed-forward neural network, with ReLU activation in all non-input layers, including the output layer for simplicity)



The weights (parameters) of the network are determined ~~by the learning algorithm, NOT by a human expert~~ by an optimization algorithm, such as batch gradient descent!

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

How to Find Appropriate Weights?

- The error is a function of all the weights
- Calculate partial derivatives
- Use a gradient-based optimization technique (e.g. batch gradient descent)
- Backpropagtion
  - Technique to calculate the gradients
  - Repeated application of the chain rule

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Neural Networks with One Hidden Layer

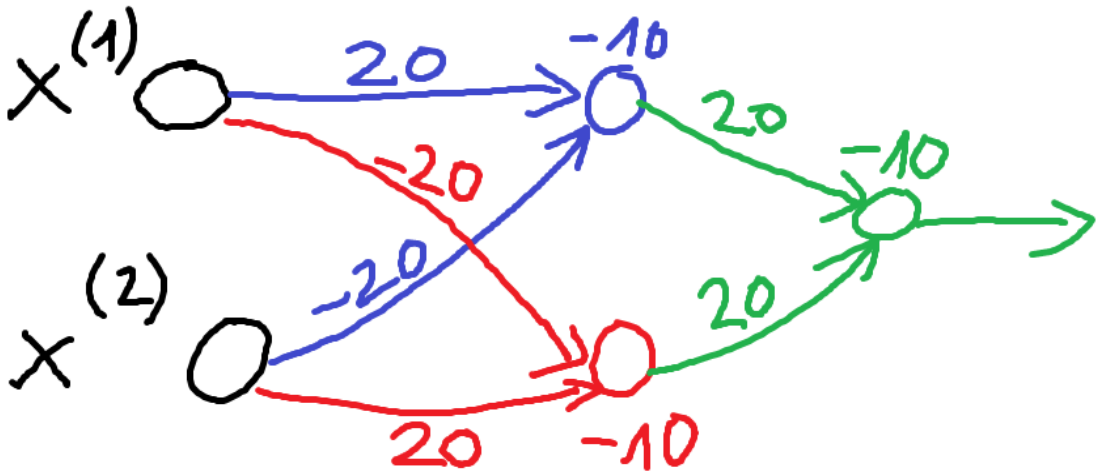
- Neural networks with one hidden layer with nonlinear activation function are **universal function approximators**.

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Neural Networks with One Hidden Layer

- Neural networks with one hidden layer with nonlinear activation function are **universal function approximators**.
- Illustration for the case of binary data: XOR

$x^{(1)}$	$x^{(2)}$	$x^{(1)}$ XOR $x^{(2)}$
0	0	0
0	1	1
1	0	1
1	1	0



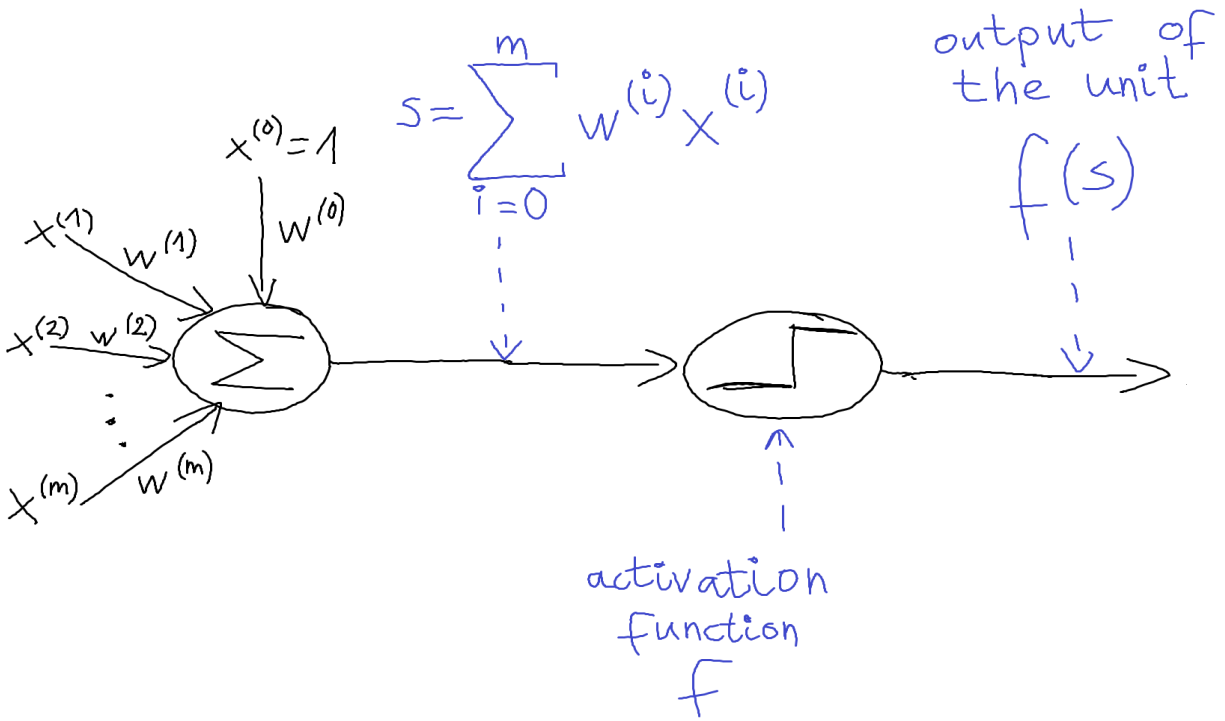
Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Do We Need More Than One Hidden Layer?

- Neural networks with a single hidden layer are universal function approximators.
- But: in order to have a good approximation, we may need extremly large number of units if we only have one hidden layer.

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

How to Describe the Computations?




```
s = 0
for i = 0..m:
  s+=x[i]*w[i]

if s < 0:
  s = 0
```

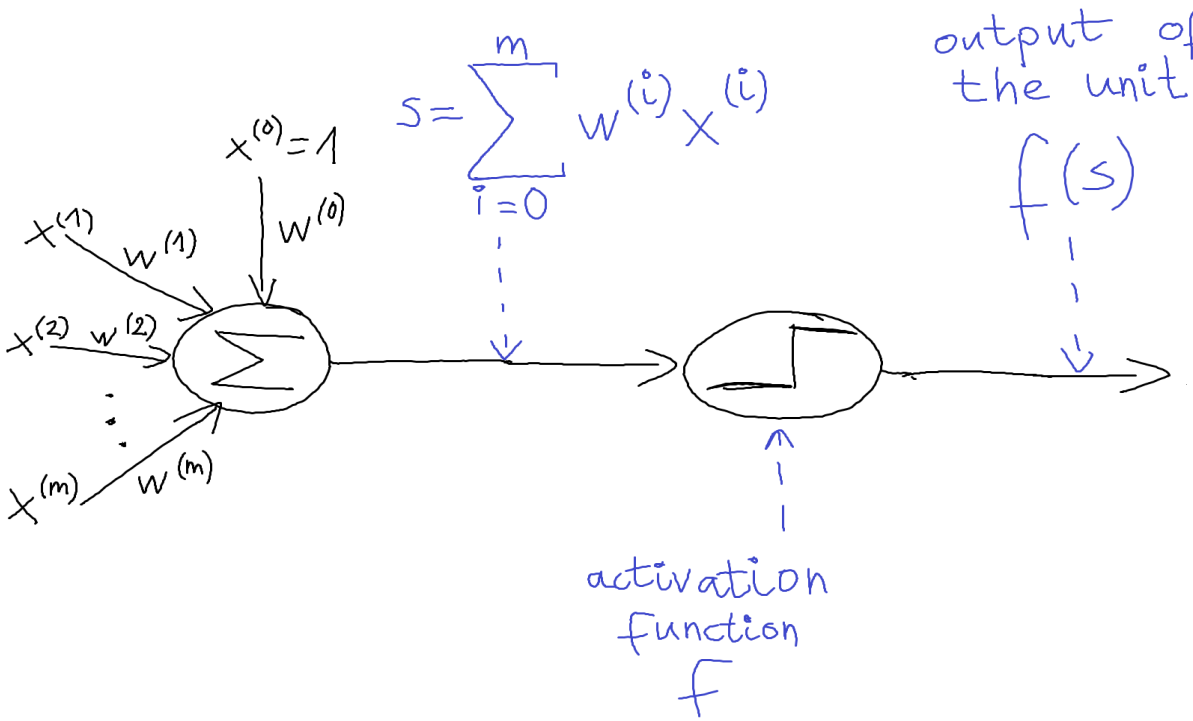
$x^{(1)}, x^{(2)}, \ldots, x^{(m)}$ = inputs of the unit

$w^{(1)}, w^{(2)}, \ldots, w^{(m)}$ = weights of $x^{(1)}, x^{(2)}, \ldots, x^{(m)}$

$w^{(0)}$ = bias weight

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Compact Description of a Single Unit




```
s = np.dot(x,w)

if s < 0:
   s = 0
```

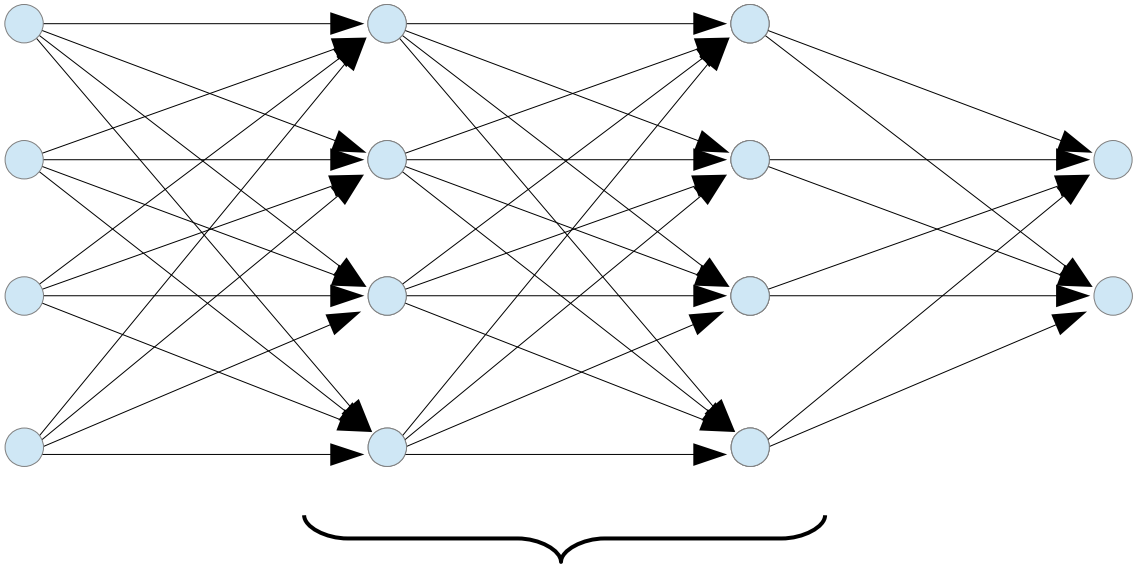
$x^{(1)}, x^{(2)}, \ldots, x^{(m)}$ = inputs of the unit

$w^{(1)}, w^{(2)}, \ldots, w^{(m)}$ = weights of $x^{(1)}, x^{(2)}, \ldots, x^{(m)}$

$w^{(0)}$ = bias weight

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

How to Describe the Computations?



input layer

hidden layers

output layer

dot product of two vectors

```
a1 = np.dot(x,w1)
a2 = np.dot(x,w2)
...

if a1 < 0:
  a1 = 0
if a2 < 0:
  a2 = 0
...

b1 = np.dot(
 [a1,a2,a3,a4],
 w1_b )
...
```

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Multiplication of Matrices

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Example:
Multiplication of 2x2 matrices

$$\begin{pmatrix} 3 & 8 \\ 2 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & 2 \\ 5 & 7 \end{pmatrix}$$

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

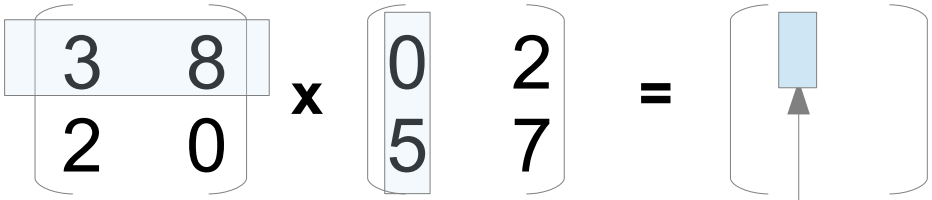
Example:
Multiplication of 2x2 matrices

$$\begin{pmatrix} 3 & 8 \\ 2 & 0 \end{pmatrix} \times \begin{pmatrix} 0 & 2 \\ 5 & 7 \end{pmatrix} = \begin{pmatrix} & \\ & \end{pmatrix}$$

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Example: Multiplication of 2x2 matrices

„row times column“

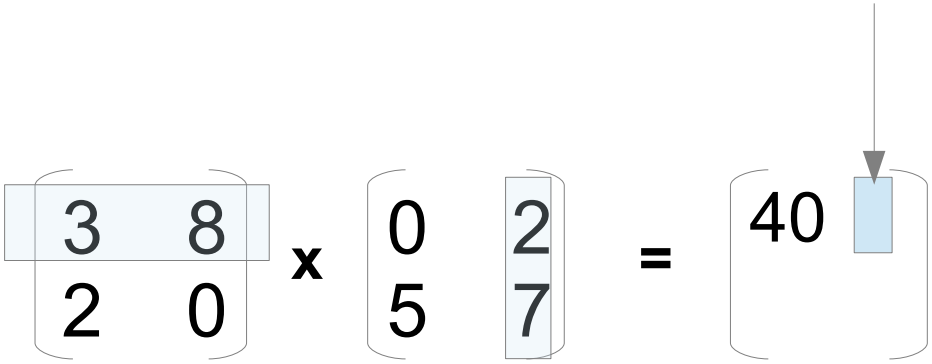


(3 x 0) + (8 x 5) = 0 + 40 = 40

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Example: Multiplication of 2x2 matrices

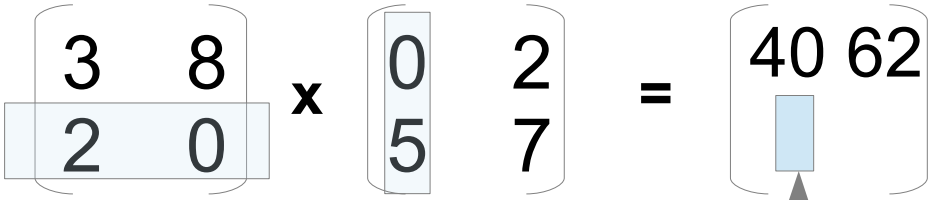
„row times column“ (3 x 2) + (8 x 7) = 6 + 56 = 62



Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Example: Multiplication of 2x2 matrices

„row times column“



(2 x 0) + (0 x 5) = 0 + 0 = 0

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Example: Multiplication of 2x2 matrices

„row times column“

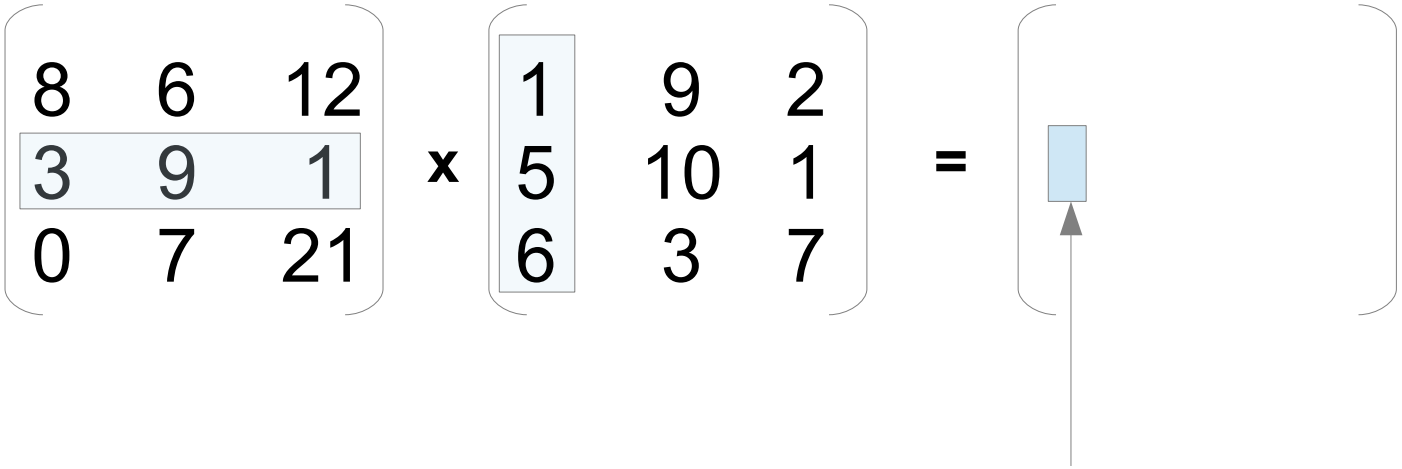
$$\begin{pmatrix} 3 & 8 \\ 2 & 0 \end{pmatrix} \times \begin{pmatrix} 0 & 2 \\ 5 & 7 \end{pmatrix} = \begin{pmatrix} 40 & 62 \\ 0 & \square \end{pmatrix}$$

$$(2 \times 2) + (0 \times 7) = 4 + 0 = 4$$

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Example: Multiplication of 3x3 matrices

„row times column“

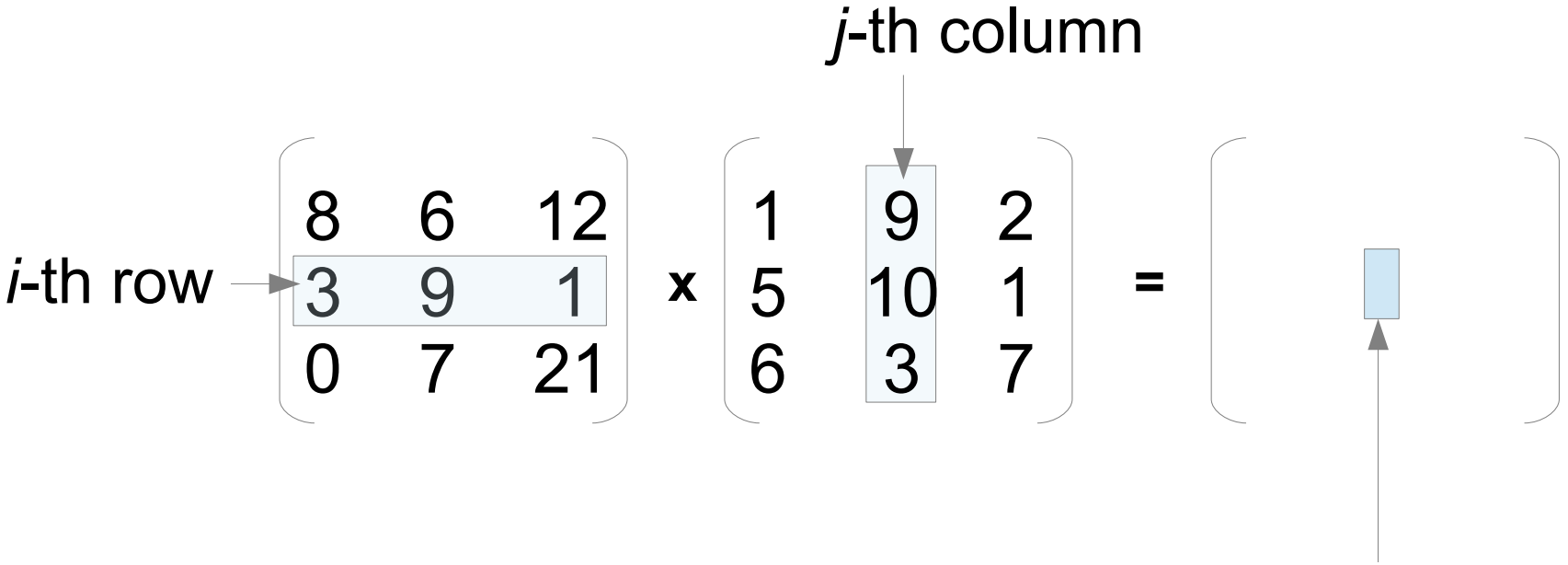


(3x1)+(9x5)+(1x6)=3+45+6=54

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Example: Multiplication of 3x3 matrices

„row times column“



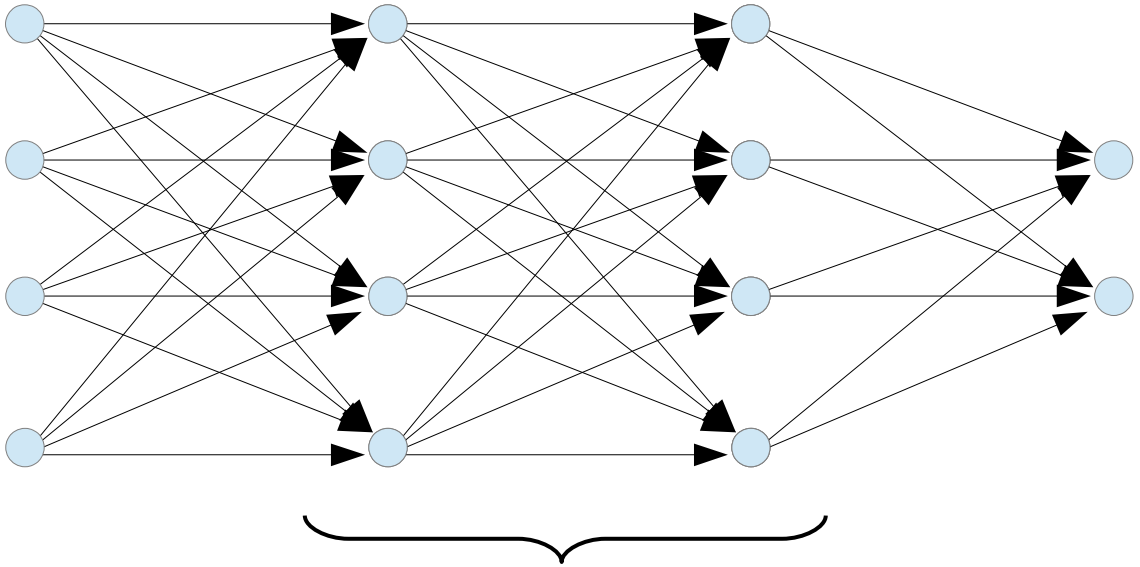
Element in the *i*-th row and *j*-th column:
product of the *i*-th row of the first matrix
and the *j*-th column of the second matrix

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Compact Description of the Computations with Matrix Multiplication

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

How to Describe the Computations?



input layer | hidden layers | output layer

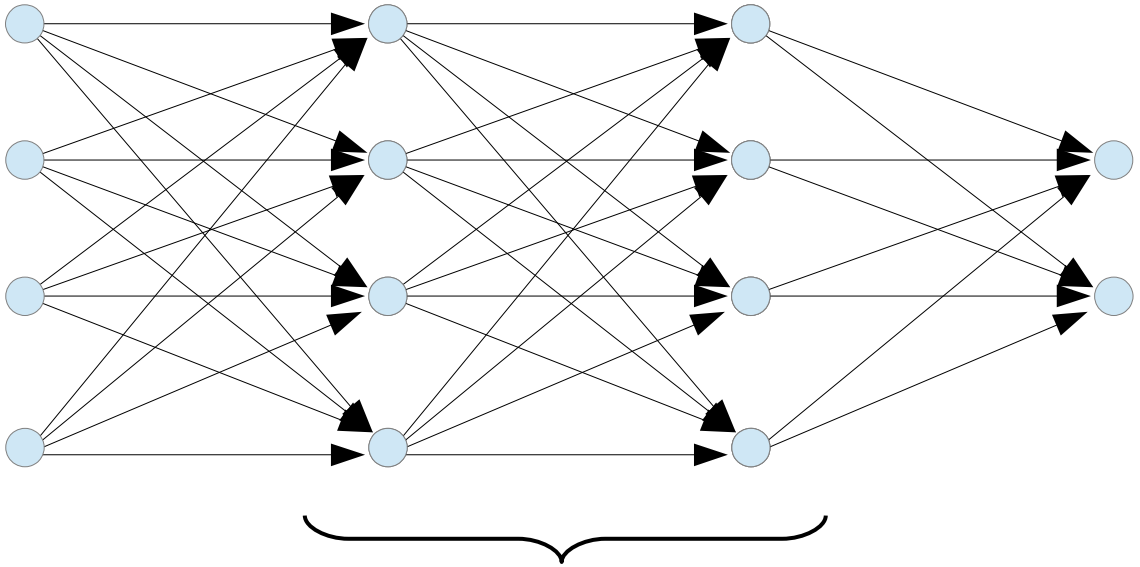
(row vector) x (matrix)

```
a = np.dot(x,W1)
a[a<0]=0

b = np.dot(a,W2)
b[b<0]=0

c = np.dot(b,W3)
c[c<0]=0
```

`x, a, b, c` are row vectors, not scalars!
`W1, W2` and **`W3`** are matrices containing all the weights between (W1) input layer and first hidden layer, (W2) first hidden layer and second hidden layer, and (W3) second hidden layer and output layer, respectively.
Each column of **`W1, W2`** and **`W3`** contains all the weights of the incoming connections associated with the same hidden unit, e.g., first column of W1 contains the weights of the first unit in the first hidden layer.

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

How to Describe the Computations?



input layer

hidden layers

output layer

(matrix) x (matrix)

```
A = np.dot(X,W1)
A[A<0]=0

B = np.dot(A,W2)
B[B<0]=0

C = np.dot(B,W3)
C[C<0]=0
```

X, A, B, C are matrices, neither vectors, nor scalars!
W1, W2 and **W3** are matrices containing all the weights between (W1) input layer and first hidden layer, (W2) first hidden layer and second hidden layer, and (W3) second hidden layer and output layer, respectively. Each column of **W1, W2** and **W3** contains all the weights of the incoming connections associated with the same hidden unit, e.g., first column of W1 contains the weights of the first unit in the first hidden layer.

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Deep Learning in a Nutshell

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

„What was wrong with backpropagation in 1986?“

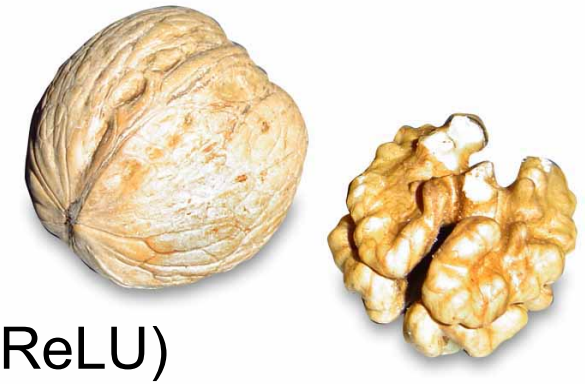
(Geoff Hinton, „Deep Learning“, May 22, 2015)

- Our labeled datasets were thousands of times too small.
- Our computers were millions of times too slow.
- We initialized the weights in a stupid way.
- We used the wrong type of non-linearity.

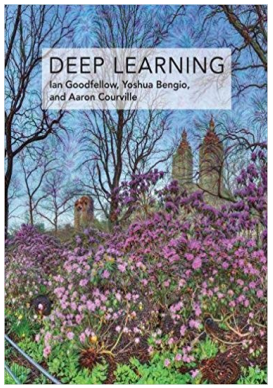
Image: https://upload.wikimedia.org/wikipedia/commons/3/34/Geoffrey_Hinton_at_UBC.jpg
Eviatar Bach [CC BY-SA 3.0 (https://creativecommons.org/licenses/by-sa/3.0)]

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Deep Learning in a Nutshell



- Size and structure of the network:
  few layers → many layers
- Activation function: sigmoid → rectified linear unit (ReLU)
- Loss function: quadratic loss → cross-entropy
- Initialization of weights: random → (unsupervised) pre-training
- Size of training data, much more memory,
  distributed computation (HPC), GPUs, TPUs…
- New regularization techniques:
  “sparsity-enforcing” regularisation terms,
  drop-out, early stop,
  parameter sharing (convolutional networks)
- New optimization techniques (e.g. ADAM)



http://www.deeplearningbook.org

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Summary

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Summary

- Mathematical operation: Matrix multiplication
- Neural Unit → neural network
- Neural networks with one hidden layer with nonlinear activations are universal function approximators
- Overview of the techniques known under the umbrella of „deep learning“

Krisztian Buza, buza@inf.elte.hu
http://biointelligence.hu

Essential Concepts

- Decision boundary
- Neural Network, Feed-forward neural network
- Neural Unit
- Activation Function – Sigmoid, ReLU
- Matrix multiplication
- GPU, TPU („AI chip“)