## Deadlines

<table>
<thead>
<tr>
<th>Assignment</th>
<th>Date of assignment</th>
<th>Deadline (midnight 23:59)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HW1</td>
<td>Sep 6</td>
<td>Sep 19</td>
</tr>
<tr>
<td>HW2</td>
<td>Sep 20</td>
<td>Oct 3</td>
</tr>
<tr>
<td>HW3</td>
<td>Oct 4</td>
<td>Oct 17</td>
</tr>
<tr>
<td>Paper summary</td>
<td>Oct 11</td>
<td>Oct 24</td>
</tr>
<tr>
<td>HW4</td>
<td>Oct 18</td>
<td>Oct 31</td>
</tr>
<tr>
<td>HW5</td>
<td>Nov 1</td>
<td>Nov 14</td>
</tr>
<tr>
<td>HW6</td>
<td>Nov 22</td>
<td>Dec 5</td>
</tr>
<tr>
<td>Project</td>
<td>Projects are <strong>collected</strong></td>
<td>Dec 13 - 15</td>
</tr>
</tbody>
</table>

*All deadlines are subject to change, check out CampusWire and website for updates*
Machine Learning
Machine Learning
Machine Learning

Supervised Learning

Unsupervised Learning
Machine Learning

Supervised Learning

Unsupervised Learning

Reinforcement Learning
Deep Learning

- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning
This handsome man will talk to you real Deep Learning (soon)

I am here to scratch the surface :)

What is deep learning?
What is deep learning?

An adaptive **non-linear mapping** from one **space** to another

Space of things  →  Space of things  →  Space of labels
What is deep learning?

An adaptive **non-linear mapping** from one **space** to another

Space of things \[\rightarrow\] Artificial Neural Networks with many layers \[\rightarrow\] Space of labels

**DL** = Networks with many layers

Iris Setosa

Speech Thanks
Sum of inputs

$\begin{align*}
X_1 &= 5 \\
X_2 &= 5
\end{align*}$

$\rightarrow 10$
Weights

Weights are connections between nodes of the network.
Weights are connections between nodes of the network. Weights are not necessarily positive integers.
The diagram shows a 2D coordinate system with points at (5, 5) and (5, 5) connected by a line. The points are marked with + signs. The text on the right indicates that the sum of inputs is calculated as follows:

- $X_1$ is multiplied by 0.12
- $X_2$ is multiplied by -0.1

The sum of inputs is then computed and passed to a further operation (not shown in the diagram).
\[
\text{Sum of inputs} = x_1 \times 0.12 + x_2 \times -0.1 = 0.1
\]
The diagram illustrates a neural network with an input layer and a single node. The inputs $X_1$ and $X_2$ are multiplied by the weights $0.12$ and $-0.1$, respectively. The sum of the inputs is then passed through an activation function. The weight of $X_1$ is $5$ and the weight of $X_2$ is $5$. The output of the activation function is $0.1$.
Step activation function

\[ \text{Sum of inputs} \times 0.12 \]
\[ \times -0.1 \]
\[ > 0 \]

Activation function
A step activation function is used to determine the output class based on the sum of inputs. The graph shows a two-dimensional space with two input variables, $X_1$ and $X_2$. The sum of the inputs is calculated for each point as follows:

- $X_1 \times 0.12$ for the Red class
- $X_2 \times -0.1$ for the Blue class

The step activation function checks if the sum of inputs is greater than 0. If the sum is greater than 0, the output is classified as Red; otherwise, it is classified as Blue.
\[ X_1 \times 0.12 + X_2 \times -0.1 \]

**Activation function**

- **Step activation function**
  - \( > 0 \)
The figure illustrates the process of applying a step activation function to a set of inputs. The inputs are multiplied by coefficients: $X_1$ is multiplied by 0.12, and $X_2$ is multiplied by -0.1. The sum of these inputs is then compared to 0.1 to determine the class. If the sum is greater than 0.1, the class is Red; otherwise, it is Blue. The diagram shows two points on a 2D plane, with each point representing a different combination of $X_1$ and $X_2$ values.
Artificial neuron
(perceptron)

Sum of inputs

Step activation function

Red

Activation function

$X_1 \times 0.12$

$X_2 \times -0.1$

$> 0$

$0.1$
**Biological neuron**

**Artificial neuron** (perceptron)

Sum of inputs

\[ x \cdot 0.12 \]

\[ x \cdot -0.1 \]

\[ \rightarrow + \rightarrow 0.1 \]

\[ > 0 \]

Activation function

Step activation function

Red
The diagram illustrates a step activation function. The inputs $X_1$ and $X_2$ are multiplied by 0.12 and -0.1 respectively. Their sum is then calculated. If the sum is greater than 0, the activation function results in 0.1, which is indicated as 'Red'. Otherwise, the function is not activated or indicated in the diagram.
The diagram illustrates a neural network with two input variables, $X_1$ and $X_2$. The inputs are multiplied by 0.12 and -0.1 respectively, and then summed. The sum is then passed through a step activation function.

- $X_1$ is multiplied by 0.12.
- $X_2$ is multiplied by -0.1.
- The sum of the inputs is then passed through a step activation function.

The step activation function is defined as $\text{Step activation function}(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ 1 & \text{if } x > 0 \end{cases}$.
Sum of inputs x 0.12 > 0

Activation function

Step function

-0.16
The diagram illustrates a neural network layer with two input nodes, $X_1$ and $X_2$, each with different weights. The inputs are multiplied by their respective weights: $X_1$ by 0.12 and $X_2$ by -0.1. The sum of these inputs yields $-0.16$. A step activation function is applied, which is defined as $\sum_{\text{of inputs}} \times 0.12 \times -0.1 > 0$. The result, $-0.16$, is less than zero, indicating that the output of this layer is classified as 'Blue'. The diagram also includes a graph with points indicating the input values and a coordinate system for visualization.
Same procedure should be performed for all data points.
Same procedure should be performed for all data points.

$X_1 \times 0.12$

$X_2 \times -0.1$

Sum of inputs

Step activation function

> 0

Activation function
Same procedure should be performed for all data points.

Blue

Red

\( X_1 \) x 0.12

\( X_2 \) x -0.1

Sum of inputs

Step activation function

Activation function

\( > 0 \)
Activation function

Step activation function

Sum of inputs

$\begin{align*}
X_1 & \times 0.12 \\
X_2 & \times -0.1
\end{align*}$
Decision boundary separating classes is controlled by weights.

Sum of inputs

Step activation function

Activation function

$x_1 \times 0.12$

$x_2 \times -0.1$

Blue

Red

Decision boundary separating classes is controlled by weights.
By changing weights we can **increase** or **decrease** the influence of input features, hence **rotate** the decision boundary.

---

**Decision boundary** separating classes is controlled by weights.

---

By changing weights we can increase or decrease the influence of input features, hence rotate the decision boundary.
By changing weights we can **increase** or **decrease** the influence of input features, hence **rotate** the decision boundary.

Good weights are found using **backpropagation** algorithm, which we will examine further.
In this case **linear decision boundary** won’t help.

Decision boundary separating classes is controlled by weights.

Decision boundary

- **Step activation function**
- $> 0$
- Activation function

```
Sum of inputs
X_1 \times 0.12
X_2 \times -0.1
```

Decision boundary separating classes is controlled by weights.

In this case **linear decision boundary** won’t help.

```
Sum of inputs
X_1 \times 0.12
X_2 \times -0.1
```

**Decision boundary** separating classes is controlled by weights.

```
Step activation function
> 0
Activation function
```

Decision boundary separating classes is controlled by weights.

```
Sum of inputs
X_1 \times 0.12
X_2 \times -0.1
```

**Decision boundary** separating classes is controlled by weights.

```
Step activation function
> 0
Activation function
```

Decision boundary separating classes is controlled by weights.

```
Sum of inputs
X_1 \times 0.12
X_2 \times -0.1
```

**Decision boundary** separating classes is controlled by weights.

```
Step activation function
> 0
Activation function
```

Decision boundary separating classes is controlled by weights.

```
Sum of inputs
X_1 \times 0.12
X_2 \times -0.1
```

**Decision boundary** separating classes is controlled by weights.

```
Step activation function
> 0
Activation function
```

Decision boundary separating classes is controlled by weights.

```
Sum of inputs
X_1 \times 0.12
X_2 \times -0.1
```

**Decision boundary** separating classes is controlled by weights.

```
Step activation function
> 0
Activation function
```

Decision boundary separating classes is controlled by weights.

```
Sum of inputs
X_1 \times 0.12
X_2 \times -0.1
```

**Decision boundary** separating classes is controlled by weights.

```
Step activation function
> 0
Activation function
```

Decision boundary separating classes is controlled by weights.

```
Sum of inputs
X_1 \times 0.12
X_2 \times -0.1
```

**Decision boundary** separating classes is controlled by weights.

```
Step activation function
> 0
Activation function
```
In this case **linear decision boundary** won’t help.
In this case **linear decision boundary** won’t help.

Using more **neurons** and ...
In this case **linear decision boundary** won’t help.

Using more **neurons** and **non-linear activation function**
Sigmoid activation function
No matter how big your $x$ can grow, your $\text{sig}(x)$ will always be between 0 and 1.
In this case **linear decision boundary** won’t help.

Using more **neurons** and **non-linear activation function**
Rectified Linear Units (ReLU)

\[ R(x) = \text{max}(0, x) \]
In this case **linear decision boundary** won’t help.

Using more **neurons** and **non-linear activation function**
In this case **linear decision boundary** won’t help.

Using more **neurons** and **non-linear** activation function

It is possible to obtain **non-linear decision boundary**

**Sigmoid**

\[
sig(x) = \frac{1}{1 + e^{-x}}
\]

**ReLu**

\[
R(x) = \max(0, x)
\]
Consider this 2D example
Point coordinates as **input**

\[(x_1, x_2)\]
Point coordinates as input

Input layer

(x₁, x₂)

x₁

x₂
Point coordinates as \textbf{input}

$(x_1, x_2)$
Point coordinates as input

\[(x_1, x_2)\]
Point coordinates as input

(x₁,x₂)
Point coordinates as input

Hidden layer

Sigmoidal activation function

sig(x)
Point coordinates as input

(x₁, x₂)
Point coordinates as input

Output layer
Score for the first class

Class scores

out_{o1}

out_{o2}
Score for the **first** class

Score for the **second** class

Class scores

\((x_1, x_2)\)
If $\text{out}_{01} > \text{out}_{02}$ neural network predicts the first class.
Our input point belongs to the second class.

If $\text{out}_{01} > \text{out}_{02}$ neural network predicts the first class.
Our input point belongs to the second class.

If $\text{out}_{o1} > \text{out}_{o2}$ neural network predicts the first class.

Therefore for this point we expect $\text{out}_{o2} > \text{out}_{o1}$.
Biases
What is the role of **bias** in Neural Networks?
What is the role of **bias** in Neural Networks?

**Bias** helps to **shift** the resulting curve.
What is the role of **bias** in Neural Networks?

- **Input:** \( x \)
- **Output:** \( \text{sig}(w_1 \cdot x) \)

\( w_1 = [0.5, 1.0, 2.0] \)

\[ \text{sig}(x) = \frac{1}{1 + e^{-x}} \]

Source: http://www.uta.fi/sis/tie/neuro/index/Neurocomputing2.pdf
What is the role of **bias** in Neural Networks?

**Input**  
\[ x \]  
\[ w_1 = [0.5, 1.0, 2.0] \]

**Output**  
\[ \text{sig}(w_1 \ast x) \]

**Input**  
\[ x \]

**Output**  
\[ \text{sig}(w_1 \ast x + b_1) \]

\[ b_1 = [-4.0, 0.0, 4.0] \quad \text{and} \quad w_1 = [1.0] \]

\[
\text{sig}(x) = \frac{1}{1 + e^{-(wx + b)}}
\]

Source: http://www.uta.fi/sis/tie/neuro/index/Neurocomputing2.pdf
This particular architecture is **arbitrary** and meant as an **example**

In practice, people use already existing architectures (e.g. **ResNet**, **BERT**, **U-Net** etc.)
When you create a new neural network, first you need to ** initialise** weights and biases.
When you create a new neural network, first you need to **initialise** weights and biases.

Sample **normal** distribution:

```python
import numpy as np
W = np.random.randn(2,2)
b = np.random.randn(1,2)
```
When you create a new neural network, first you need to **initialise** weights and biases.

Sample **normal** distribution:

```python
import numpy as np
W = np.random.randn(2,2)  
b = np.random.randn(1,2)
```
Let’s calculate **class scores** for each class using **feed-forward path** algorithm.
Let's calculate **class scores** for each class using **feed-forward path** algorithm.

Coordinates of the point $(0.05, 0.1)$.
Let's calculate **class scores** for each class using **feed-forward path** algorithm.
Let's calculate **class scores** for each class using **feed-forward path** algorithm.
Let's calculate class scores for each class using feed-forward path algorithm.

What is the input of the first neuron of the hidden layer $\text{in}(h_1)$?

Let's calculate class scores for each class using feed-forward path algorithm.

What is the input of the first neuron of the hidden layer $\text{in}(h_1)$?

Let's calculate class scores for each class using feed-forward path algorithm.

What is the input of the first neuron of the hidden layer $\text{in}(h_1)$?
Let's calculate class scores for each class using feed-forward path algorithm.

What is the input of the first neuron of the hidden layer $\text{in}(h_1)$?

$$\text{in}(h_1) = \ldots + \ldots + \ldots$$
Let's calculate class scores for each class using the feed-forward path algorithm.

What is the input of the first neuron of the hidden layer $\text{in}(h_1)$?

$$\text{in}(h_1) = x_1 * w_1 + x_2 * w_2 + b_{11}$$
Let's calculate class scores for each class using feed-forward path algorithm.

What is the input of the first neuron of the hidden layer $\text{in}(h_1)$?

\[
\text{in}(h_1) = x_1 \times 0.15 + x_2 \times 0.2 + 0.35
\]
Let's calculate class scores for each class using feed-forward path algorithm.

What is the input of the first neuron of the hidden layer \( \text{in}(h_1) \)?

\[
\text{in}(h_1) = 0.05 \times 0.15 + 0.1 \times 0.2 + 0.35
\]
Let's calculate class scores for each class using feed-forward path algorithm.

What is the input of the first neuron of the hidden layer $\text{in}(h_1)$?

$$\text{in}(h_1) = 0.38$$
Let's calculate class scores for each class using the feed-forward path algorithm.

What is the input of the first neuron of the hidden layer $\text{in}(h_1)$?

Let's calculate class scores for each class using the feed-forward path algorithm.

What is the output of the first neuron of the hidden layer $\text{out}(h_1)$?

\[
\text{in}(h_1) = 0.38
\]

\[
\text{out}(h_1) = \ldots
\]
Let's calculate class scores for each class using feed-forward path algorithm.

What is the input of the first neuron of the hidden layer in(h$_1$)?

in(h$_1$) = 0.38

What is the output of the first neuron of the hidden layer out(h$_1$)?

out(h$_1$) = sig(in(h$_1$))

Let's calculate class scores for each class using feed-forward path algorithm.
Let's calculate class scores for each class using the feed-forward path algorithm.

What is the input of the first neuron of the hidden layer $in(h_1)$?

$in(h_1) = 0.38$

What is the output of the first neuron of the hidden layer $out(h_1)$?

$out(h_1) = \text{sig}(0.38)$
Let's calculate class scores for each class using the feed-forward path algorithm.

What is the input of the first neuron of the hidden layer $\text{in}(h_1)$?

What is the output of the first neuron of the hidden layer $\text{out}(h_1)$?

$\text{in}(h_1) = 0.38$

$\text{out}(h_1) = \text{sig}(0.38)$

$\text{sig}(x)$
Let's calculate class scores for each class using the feed-forward path algorithm.

What is the **input** of the first neuron of the hidden layer $\text{in}(h_1)$?

$\text{in}(h_1) = 0.38$

What is the **output** of the first neuron of the hidden layer $\text{out}(h_1)$?

$\text{out}(h_1) = \text{sig}(0.38)$

The input of the first neuron of the hidden layer is 0.38.

The output of the first neuron of the hidden layer is $\text{sig}(0.38) = 0.59$. 

---

**Diagram Notes:**
- $x_1$ and $x_2$ represent input features.
- The node with input 0.05 and output 0.35 is labeled as $b_{11}$.
- The node with input 0.1 and output 0.59 is labeled as $h_1$.
- The sigmoid function $\text{sig}(x)$ is shown with input 0.38 and output 0.59.
Let's calculate class scores for each class using feed-forward path algorithm.

What is the input of the first neuron of the hidden layer $\text{in}(h_1)$?

\[ \text{in}(h_1) = 0.38 \]

What is the output of the first neuron of the hidden layer $\text{out}(h_1)$?

\[ \text{out}(h_1) = \text{sig}(0.38) = 0.59 \]
Let's calculate **class scores** for each class using feed-forward path algorithm.
Let's calculate **class scores** for each class using **feed-forward path** algorithm.

What is the **input** of the second neuron of the hidden layer in $h_2$?
Let's calculate **class scores** for each class using *feed-forward path algorithm*

Let's calculate class scores for each class using feed-forward path algorithm

\[ \text{out}(h_1) = 0.59 \]

What is the **input** of the second neuron of the hidden layer \( \text{in}(h_2) \)?

\[ \text{in}(h_2) = \ldots \]
Let's calculate **class scores** for each class using **feed-forward path** algorithm.

What is the **input** of the second neuron of the hidden layer \(\text{in}(h_2)\)?

\[
\text{in}(h_2) = x_1 w_3 + x_2 w_4 + b_{12}
\]
Let's calculate class scores for each class using feed-forward path algorithm.

What is the input of the second neuron of the hidden layer $\text{in}(h_2)$?

\[
\text{in}(h_2) = x_1 \times 0.25 + x_2 \times 0.3 + 0.35
\]
Let's calculate **class scores** for each class using **feed-forward path** algorithm.

Let's calculate class scores for each class using feed-forward path algorithm.

\[
\text{out}(h_1) = 0.59
\]

\[
\text{in}(h_2) = 0.05 \times 0.25 + 0.1 \times 0.3 + 0.35
\]

What is the **input** of the second neuron of the hidden layer \(in(h_2)\)?

\[
\text{in}(h_2) = 0.05 \times 0.25 + 0.1 \times 0.3 + 0.35
\]
Let's calculate class scores for each class using feed-forward path algorithm.

What is the input of the second neuron of the hidden layer $\text{in}(h_2)$?

$\text{in}(h_2) = 0.39$
Let's calculate **class scores** for each class using **feed-forward path** algorithm.

What is the **input** of the second neuron of the hidden layer \( \text{in}(h_2) \)?

\[ \text{in}(h_2) = 0.39 \]

What is the **output** of the second neuron of the hidden layer \( \text{out}(h_2) \)?

\[ \text{out}(h_2) = \ldots \]
Let's calculate **class scores** for each class using **feed-forward path** algorithm.

Let's calculate class scores for each class using feed-forward path algorithm.

\[
\text{out}(h_1) = 0.59
\]

\[
in(h_2) = 0.39
\]

\[
\text{out}(h_2) = \text{sig}(0.39)
\]

What is the **input** of the second neuron of the hidden layer \(in(h_2)\)?

What is the **output** of the second neuron of the hidden layer \(out(h_2)\)?
Let's calculate class scores for each class using feed-forward path algorithm.

Let's calculate class scores for each class using feed-forward path algorithm.

\[ \text{out}(h_1) = 0.59 \]

What is the input of the second neuron of the hidden layer \( \text{in}(h_2) \)?

\[ \text{in}(h_2) = 0.39 \]

What is the output of the second neuron of the hidden layer \( \text{out}(h_2) \)?

\[ \text{out}(h_2) = 0.6 \]
Let's calculate class scores for each class using feed-forward path algorithm.

\[
\begin{align*}
\text{out}(\mathbf{h}_1) &= 0.59 \\
\text{out}(\mathbf{h}_2) &= 0.6
\end{align*}
\]
Let's calculate **class scores** for each class using **feed-forward path** algorithm.

\[
\begin{align*}
\text{out}(h_1) &= 0.59 \\
\text{out}(h_2) &= 0.6
\end{align*}
\]
Let's calculate class scores for each class using feed-forward path algorithm.

out($h_1$) = 0.59

out($h_2$) = 0.6

What is the input of the first neuron of the output layer in($o_1$)?
Let's calculate class scores for each class using feed-forward path algorithm.

What is the input of the first neuron of the output layer in(\(o_1\))?
Let's calculate class scores for each class using feed-forward path algorithm.

\[ \text{in}(o_1) = \text{out}(h_1)w_5 + \text{out}(h_2)w_6 + b_{21} \]

What is the input of the first neuron of the output layer in \( o_1 \)?

\[ \text{out}(h_1) = 0.59 \]

\[ \text{out}(h_2) = 0.6 \]

\[ \text{in}(o_1) = 0.59 \times 0.40 + 0.6 \times 0.45 + 0.6 \]

h1 neuron input:

\[ 0.59 \times 0.40 + 0.6 \times 0.45 + 0.6 \]

h2 neuron input:

\[ 0.6 \times 0.45 + 0.6 \]

Input of the first neuron of the output layer in \( o_1 \):

\[ 0.59 \times 0.40 + 0.6 \times 0.45 + 0.6 \]

\[ = 0.236 + 0.27 + 0.6 \]

\[ = 1.106 \]
Let's calculate class scores for each class using feed-forward path algorithm.

\[
in(o_1) = \text{out}(h_1) \times 0.4 + \text{out}(h_2) \times 0.45 + 0.6
\]

What is the input of the first neuron of the output layer, \(in(o_1)\)?

\[
in(o_1) = 0.59 \times 0.4 + 0.6 \times 0.45 + 0.6
\]

\[
\text{out}(h_1) = 0.59
\]

\[
\text{out}(h_2) = 0.6
\]

\[
\text{in}(o_1) = \text{out}(h_1) \times 0.4 + \text{out}(h_2) \times 0.45 + 0.6
\]

\[
in(o_1) = 0.59 \times 0.4 + 0.6 \times 0.45 + 0.6
\]
Let's calculate class scores for each class using feed-forward path algorithm. For the first neuron of the output layer, the input is:

\[ \text{in}(o_1) = 0.59 \times 0.4 + 0.6 \times 0.45 + 0.6 \]
What is the input of the first neuron of the output layer $in(o_1)$?

\[
in(o_1) = 1.1
\]
Let's calculate **class scores** for each class using **feed-forward path** algorithm.

Let's calculate class scores for each class using feed-forward path algorithm.

out($h_1$) = 0.59  

in($o_1$) = 1.1

out($h_2$) = 0.6
Let's calculate class scores for each class using feed-forward path algorithm.

What is the output of the first neuron of the output layer \( \text{out}(o_1) \)?

\[
\begin{align*}
\text{out}(h_1) &= 0.59 \\
\text{in}(o_1) &= 1.1 \\
\text{out}(h_2) &= 0.6
\end{align*}
\]

\( \text{out}(h_1) \) is the output of the first hidden layer.
\( \text{in}(o_1) \) is the input to the first output neuron.
\( \text{out}(h_2) \) is the output of the second hidden layer.

\[
\begin{align*}
\text{out}(o_1) &= 0.59 + 1.1 \\
&= 1.69
\end{align*}
\]

The output of the first neuron of the output layer is 1.69.
Let's calculate class scores for each class using feed-forward path algorithm.

What is the output of the first neuron of the output layer \( \text{out}(o_1) \)?

\[
\text{out}(h_1) = 0.59
\]

\[
\text{in}(o_1) = 1.1
\]

Again sigmoid
Let's calculate class scores for each class using feed-forward path algorithm.

What is the output of the first neuron of the output layer \( \text{out}(o_1) \)?

\[
\begin{align*}
\text{out}(h_1) &= 0.59 \\
\text{in}(o_1) &= 1.1 \\
\text{out}(o_1) &= \text{sig}(1.1)
\end{align*}
\]
What is the output of the first neuron of the output layer $\text{out}(o_1)$?

Let's calculate class scores for each class using feed-forward path algorithm.

\[ \text{out}(h_1) = 0.59 \]
\[ \text{out}(h_2) = 0.6 \]
\[ \text{in}(o_1) = 1.1 \]
\[ \text{out}(o_1) = 0.75 \]

Again sigmoid.
Let's calculate **class scores** for each class using **feed-forward path** algorithm.

Let's calculate class scores for each class using feed-forward path algorithm.
Let’s calculate **class scores** for each class using **feed-forward path** algorithm.
Let’s calculate **class scores** for each class using **feed-forward path** algorithm.

Often in practice, feed-forward path is performed using **matrices**.
Let's calculate **class scores** for each class using **feed-forward path** algorithm.

Often in practice, feed-forward path is performed using **matrices**
Let’s calculate **class scores** for each class using **feed-forward path** algorithm.

Often in practice, feed-forward path is performed using **matrices**.
Let's calculate **class scores** for each class using **feed-forward path** algorithm.

Often in practice, feed-forward path is performed using **matrices**.
Let's calculate **class scores** for each class using **feed-forward path** algorithm.

**Input matrix**

\[
\begin{pmatrix}
X_1 & X_2 \\
W_1 & W_3 \\
W_2 & W_4
\end{pmatrix}
\]

\[
X \ast W_1 + \begin{pmatrix} b_{11} & b_{12} \end{pmatrix}
\]

**Sigmoid**

\[
\text{Sigmoid}(h_1) = 0.59
\]

\[
\text{out}(h_1) = 0.59
\]

\[
in(o_1) = 1.1
\]

\[
\text{out}(o_1) = 0.75
\]

\[
\text{out}(o_2) = 0.77
\]

Often in practice, feed-forward path is performed using **matrices**.
Let’s calculate **class scores** for each class using **feed-forward path** algorithm

Often in practice, feed-forward path is performed using **matrices**
Let's calculate **class scores** for each class using **feed-forward path** algorithm.

Often in practice, feed-forward path is performed using **matrices**.
Let's calculate class scores for each class using feed-forward path algorithm.

Often in practice, feed-forward path is performed using matrices.
Let's calculate **class scores** for each class using **feed-forward path** algorithm.

Often in practice, feed-forward path is performed using **matrices**.
Let's calculate **class scores** for each class using **feed-forward path** algorithm.

Let's calculate class scores for each class using **feed-forward path** algorithm.

- $\text{out}(h_1) = 0.59$
- $\text{in}(o_1) = 1.1$
- $\text{out}(o_1) = 0.75$
- $\text{out}(o_2) = 0.77$
How good are the class predictions (scores)?

\[
\begin{align*}
\text{out}(h_1) &= 0.59 \\
\text{in}(o_1) &= 1.1 \\
\text{out}(h_2) &= 0.6 \\
\text{out}(o_1) &= 0.75 \\
\text{out}(o_2) &= 0.77
\end{align*}
\]
How good are the class predictions (scores)?

Our input point belongs to the second class

\[ x_1 \]
\[ x_2 \]
How good are the class predictions (scores)?

Our input point belongs to the second class.

Therefore for this point we expect:
- \( \text{out}(o_1) \) to be very close to 0
- \( \text{out}(o_2) \) to be very close to 1

\( \text{out}(o_1) = 0.75 \)
\( \text{out}(o_2) = 0.77 \)
How good are the class predictions (scores)?

Our input point belongs to the second class. Therefore for this point we expect:

- \( \text{out}(o_1) \) to be very close to 0
- \( \text{out}(o_2) \) to be very close to 1

These expected values for each input point are stored in a variable called ground truth or truth for short. These are labels in supervised learning terms.
How good are the class predictions (scores)?

Our input point belongs to the second class.

Let's calculate the error for these predictions (MSE).

out(o₁) to be very close to 0
out(o₂) to be very close to 1

out(o₁) = 0.75
out(o₂) = 0.77
How good are the class predictions (scores)?

Our input point belongs to the second class.

Let's calculate the error for these predictions (MSE):

\[ E_{total} = E_{o1} + E_{o2} \]

out(o_1) to be very close to 0

out(o_1) = 0.75

out(o_2) to be very close to 1

out(o_2) = 0.77
How good are the class predictions (scores)?

Our input point belongs to the second class.

Let’s calculate the error for these predictions (MSE)

\[
E_{total} = E_{o1} + E_{o2}
\]

\[
E_{o1} = \text{out(o1)} = 0.75
\]

\[
\text{out(o1)} \text{ to be very close to 0}
\]

\[
\text{out(o2)} \text{ to be very close to 1}
\]

\[
\text{out(o2)} = 0.77
\]
How good are the class **predictions** (scores)?

Our **input** point belongs to the **second** class.

Let's calculate the **error** for these predictions (MSE).

\[
E_{\text{total}} = E_{o_1} + E_{o_2}
\]

\[
E_{o_1} = \frac{1}{2} (\text{truth} - \text{out}_{o_1})^2
\]

out(o_1) to be very close to 0

out(o_1) = 0.75

out(o_2) to be very close to 1

out(o_2) = 0.77

out(h_1) = 0.59

in(o_1) = 1.1

out(h_2) = 0.6

in(o_2) = 1.1

out(h_3) = 0.55

in(o_3) = 1.1

out(h_4) = 0.30

in(o_4) = 1.1

out(h_5) = 0.40

in(o_5) = 1.1

out(h_6) = 0.25

in(o_6) = 1.1

out(h_7) = 0.50

in(o_7) = 1.1

out(h_8) = 0.45

in(o_8) = 1.1

out(h_9) = 0.35

in(o_9) = 1.1

out(h_{10}) = 0.20

in(o_{10}) = 1.1

out(h_{11}) = 0.15

in(o_{11}) = 1.1

out(h_{12}) = 0.10

in(o_{12}) = 1.1

out(h_{13}) = 0.05

in(o_{13}) = 1.1

out(h_{14}) = 0.00

in(o_{14}) = 1.1

x_1
x_2

(0.05, 0.1)
How good are the class predictions (scores)?

Our input point belongs to the second class.

Let's calculate the error for these predictions (MSE).

\[ E_{\text{total}} = E_{o_1} + E_{o_2} \]

\[ E_{o_1} = \frac{1}{2} (\text{truth} - \text{out}_{o_1})^2 \]
How good are the class predictions (scores)?

Our input point belongs to the second class.

Let's calculate the error for these predictions (MSE).

\[
E_{total} = E_{o1} + E_{o2}
\]

\[
E_{o1} = \frac{1}{2}(\text{truth} - 0.75)^2
\]

out(o_1) to be very close to 0

out(o_2) to be very close to 1

out(o_1) = 0.75

out(o_2) = 0.77
How good are the class predictions (scores)?

Our input point belongs to the second class.

Let's calculate the error for these predictions (MSE).

\[ E_{total} = E_{o_1} + E_{o_2} \]

\[ E_{o_1} = \frac{1}{2} (0 - 0.75)^2 \]

\[ \text{out}(o_1) = 0.75 \]

\[ \text{out}(o_2) = 0.77 \]

\[ \text{in}(o_1) = 1.1 \]

\[ \text{out}(h_1) = 0.59 \]

\[ \text{out}(o_1) \text{ to be very close to 0} \]

\[ \text{out}(o_2) \text{ to be very close to 1} \]
How good are the class predictions (scores)?

Our input point belongs to the second class.

Let’s calculate the error for these predictions (MSE):

\[ E_{total} = E_{o1} + E_{o2} \]

\[ E_{o1} = 0.28 \]

\[ \text{out}(o_1) = 0.75, \text{out}(o_2) = 0.77 \]

\[ \text{out}(h_1) = 0.59, \text{in}(o_1) = 1.1 \]

\[ \text{out}(o_1) \text{ to be very close to 0, out}(o_2) \text{ to be very close to 1} \]

\[ (0.05,0.1) \]
How good are the class **predictions** (scores)?

Our **input** point belongs to the **second** class.

Let’s calculate the **error** for these predictions (MSE)

\[ E_{total} = 0.28 + E_{o_2} \]

\[ out(o_1) = 0.75 \]

\[ out(o_1) \text{ to be very close to } 0 \]

\[ out(o_2) = 0.77 \]

\[ out(o_2) \text{ to be very close to } 1 \]
How good are the class predictions (scores)?

Our input point belongs to the second class.

Let's calculate the error for these predictions (MSE).

$$E_{\text{total}} = 0.28 + E_{o_2}$$

Error on the first output neuron.

out($o_1$) to be very close to 0

out($o_1$) to be 0.75

out($o_2$) to be very close to 1

out($o_2$) to be 0.77
How good are the class predictions (scores)?

Our input point belongs to the second class.

Let's calculate the error for these predictions (MSE).

\[ E_{total} = 0.28 + E_{o2} \]

out(o1) to be very close to 0.
out(o1) = 0.75
in(o1) = 1.1
out(h1) = 0.59

out(o2) to be very close to 1.
out(o2) = 0.77

How good are the class predictions (scores)?

Our input point belongs to the second class.

Let's calculate the error for these predictions (MSE)

\[ E_{total} = 0.28 + E_{o2} \]

\[ E_{o2} = \frac{1}{2}(truth - out_{o2})^2 \]
How good are the class predictions (scores)?

Our input point belongs to the second class.

Let’s calculate the error for these predictions (MSE).

\[ E_{\text{total}} = 0.28 + E_{o_2} \]

\[ E_{o_2} = \frac{1}{2} (1 - 0.77)^2 \]
How good are the class **predictions** (scores)?

Our input point belongs to the second class.

Let's calculate the error for these predictions (MSE).

$$E_{\text{total}} = 0.28 + E_{o2}$$

$$E_{o2} = 0.025$$
How good are the class predictions (scores)?

Our input point belongs to the second class.

Let's calculate the error for these predictions (MSE):

\[ E_{total} = 0.28 + 0.025 \]

out(o₁) to be very close to 0
out(o₂) to be very close to 1
How good are the class **predictions** (scores)?

Our input point belongs to the **second** class.

Let's calculate the **error** for these predictions (**MSE**)

\[ E_{total} = 0.28 + 0.025 \]

Prediction: 0.75, error: 0.28, truth = 0
Prediction: 0.77, error: 0.025, truth = 1
How good are the class **predictions** (scores)?

Our **input** point belongs to the **second** class.

Let's calculate the **error** for these predictions (**MSE**).

\[ E_{total} = 0.305 \]

<table>
<thead>
<tr>
<th>Prediction</th>
<th>Error</th>
<th>Truth</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.75</td>
<td>0.28</td>
<td>0</td>
</tr>
<tr>
<td>0.77</td>
<td>0.025</td>
<td>1</td>
</tr>
</tbody>
</table>
How good are the class **predictions** (scores)?

Our input point belongs to the **second** class.

Let’s calculate the **error** for these predictions (**MSE**).

\[ E_{total} = 0.305 \]

What can we do to make the error **smaller**?
What can we do to make the error **smaller**?

- $x_1$: 0.05
- $x_2$: 0.1

### Prediction and Error

- **prediction**
  - $h_1$: 0.59
  - $h_2$: 0.6
  - $o_1$: 1.1

- **error**
  - $h_1$: 0.28
  - $h_2$: 0.025

### Truth

- $o_1$: 0
- $o_1$: 1
What can we do to make the error smaller?

We need to make the predictions match the truth.

<table>
<thead>
<tr>
<th>Prediction</th>
<th>Error</th>
<th>Truth</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.75</td>
<td>0.28</td>
<td>0</td>
</tr>
<tr>
<td>0.77</td>
<td>0.025</td>
<td>1</td>
</tr>
</tbody>
</table>
What can we do to make the error **smaller**?

\[
\begin{align*}
\text{out}(h_1) &= 0.59 \\
\text{in}(o_1) &= 1.1 \\
\text{prediction} &= 0.75 \\
\text{error} &= 0.28 \\
\text{truth} &= 0 \\
\text{prediction} &= 0.77 \\
\text{error} &= 0.025 \\
\text{truth} &= 1
\end{align*}
\]
What can we do to make the error **smaller**?

Can we change the **inputs**?

<table>
<thead>
<tr>
<th>Prediction</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.75</td>
<td>0.28</td>
</tr>
<tr>
<td>0.77</td>
<td>0.025</td>
</tr>
</tbody>
</table>

prediction | error

truth = 0

truth = 1
What can we do to make the error **smaller**?

Can we change the **inputs**?

Yes, but this will compromise the data.

<table>
<thead>
<tr>
<th>prediction</th>
<th>error</th>
<th>truth</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.75</td>
<td>0.28</td>
<td>0</td>
</tr>
<tr>
<td>0.77</td>
<td>0.025</td>
<td>1</td>
</tr>
</tbody>
</table>
What can we do to make the error **smaller**?

The only way is to change **weights** and **biases**.
What can we do to make the error **smaller**?

Let's try to reduce total **error** by changing \( w_5 \)

<table>
<thead>
<tr>
<th>Prediction</th>
<th>Error</th>
<th>Truth</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.75</td>
<td>0.28</td>
<td>0</td>
</tr>
<tr>
<td>0.77</td>
<td>0.025</td>
<td>1</td>
</tr>
</tbody>
</table>
What can we do to make the error **smaller**?

Let's try to reduce **total** error by changing $w_5$.
What can we do to make the error smaller?

Let's try to reduce total error by changing $w_5$

$$E_{total} = 0.28 + 0.025$$
What can we do to make the error **smaller**?

Let’s try to reduce **total error** by changing \( w_5 \)

\[
E_{total} = 0.28 + 0.025
\]
What can we do to make the error smaller?

Let's try to reduce total error by changing $w_5$

$$E_{total} = 0.29 + 0.025$$
What can we do to make the error **smaller**?

Let's try to reduce **total** error by changing $w_5$

\[
E_{total} = 0.28 + 0.025
\]

prediction | error
--- | ---
0.75 | 0.28
truth = 0

prediction | error
--- | ---
0.77 | 0.025
truth = 1
What can we do to make the error **smaller**?

Let's try to reduce **total error** by changing $w_5$

$$E_{total} = 0.278 + 0.025$$
What can we do to make the error **smaller**?

Let's try to reduce **total error** by changing $w_5$

\[
E_{total} = 0.273 + 0.025
\]
What can we do to make the error **smaller**?

Let’s try to reduce total **error** by changing $W_5$

$$E_{total} = 0.273 + 0.025$$

Every time we change a weight, have to recompute the **feed-forward path**
What can we do to make the error **smaller**?

Every time we change a weight, have to recompute the **feed-forward path**
We want a way to **efficiently update all our weights** so that **total error decreases** most substantially.
We want a way to **efficiently update all** our **weights** so that **total error decreases** most substantially.

We want to compute the **gradient**.
We want a way to **efficiently update all our weights** so that **total error decreases** most substantially.

\[ \frac{\partial E}{\partial w} \]

Gradient shows the direction of the **greatest increase** of the function.
We want a way to **efficiently update all our weights** so that **total error decreases** most substantially.

Gradient shows the direction of the greatest increase of the function.
Backpropagation algorithm

Let's try to reduce **total error** by changing $w_5$.
Backpropagation algorithm

Let's try to reduce **total error** by changing $w_5$

We need to find a gradient $E_{total}$ with respect to $w_5$

$$\frac{\partial E_{total}}{\partial w_5} = \ldots$$
Backpropagation algorithm

Let’s try to reduce total error by changing $w_5$

We need to find a gradient $E_{\text{total}}$ with respect to $w_5$

$$\frac{\partial E_{\text{total}}}{\partial w_5} = \ldots$$

Will show the direction of the increase of the function
Backpropagation algorithm

Let’s try to reduce total error by changing $w_5$

We need to find a gradient $E_{total}$ with respect to $w_5$

$$\frac{\partial E_{total}}{\partial w_5} = \ldots$$
Let’s try to reduce the total error by changing $w_5$.

We need to find a gradient $\frac{\partial E_{\text{total}}}{\partial w_5}$ with respect to $w_5$.

$$\frac{\partial E_{\text{total}}}{\partial w_5} = \frac{\partial E_{\text{total}}}{\partial E_{o_1}} \ast \ldots$$
**Backpropagation algorithm**

Let's try to reduce total error by changing $w_5$

We need to find a gradient $E_{\text{total}}$ with respect to $w_5$

$$\frac{\partial E_{\text{total}}}{\partial w_5} = \frac{\partial E_{\text{total}}}{\partial E_{o_1}} * \ldots$$

$$E_{\text{total}} = E_{o_1} + E_{o_2}$$
Backpropagation algorithm

Let's try to reduce total error by changing $w_5$

We need to find a gradient $E_{\text{total}}$ with respect to $w_5$

$$\frac{\partial E_{\text{total}}}{\partial w_5} = \frac{\partial E_{\text{total}}}{\partial E_{o_1}} \ast \frac{\partial E_{o_1}}{\partial \text{out}_{o_1}} \ast \ldots$$
Backpropagation algorithm

Let's try to reduce **total error** by changing $w_5$

We need to find a **gradient** $E_{total}$ with respect to $w_5$

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial E_{o1}} \times \frac{\partial E_{o1}}{\partial out_{o1}} \times \frac{\partial out_{o1}}{\partial in_{o1}} \times \ldots$$
Let's try to reduce total error by changing $w_5$

We need to find a gradient $E_{total}$ with respect to $w_5$

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial E_{o1}} \ast \frac{\partial E_{o1}}{\partial out_{o1}} \ast \frac{\partial out_{o1}}{\partial in_{o1}} \ast \frac{\partial in_{o1}}{\partial w_5}$$

Backpropagation algorithm
Backpropagation algorithm

Let’s try to reduce total error by changing $w_5$

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial E_{o1}} \ast \frac{\partial E_{o1}}{\partial out_{o1}} \ast \frac{\partial out_{o1}}{\partial in_{o1}} \ast \frac{\partial in_{o1}}{\partial w_5}$$

$$E_{total} = E_{o1} + E_{o2}$$
Let's try to reduce total error by changing $w_5$.

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial E_{o1}} \ast \frac{\partial E_{o1}}{\partial out_{o1}} \ast \frac{\partial out_{o1}}{\partial in_{o1}} \ast \frac{\partial in_{o1}}{\partial w_5}$$

Chain rule
Let's try to reduce total error by changing $w_5$

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial E_{o1}} \ast \frac{\partial E_{o1}}{\partial out_{o1}} \ast \frac{\partial out_{o1}}{\partial in_{o1}} \ast \frac{\partial in_{o1}}{\partial w_5}$$

Backpropagation algorithm
Backpropagation algorithm

Let’s try to reduce total error by changing $w_5$

$$E_{total} = E_{o_1} + E_{o_2}$$

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial E_{o_1}} \cdot \frac{\partial E_{o_1}}{\partial out_{o_1}} \cdot \frac{\partial out_{o_1}}{\partial in_{o_1}} \cdot \frac{\partial in_{o_1}}{\partial w_5}$$

$$E_{total} = E_{o_1} + E_{o_2}$$
Backpropagation algorithm

Let’s try to reduce total error by changing $w_5$

$$E_{total} = E_{o_1} + E_{o_2}$$

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial E_{o_1}} \ast \frac{\partial E_{o_1}}{\partial out_{o_1}} \ast \frac{\partial out_{o_1}}{\partial in_{o_1}} \ast \frac{\partial in_{o_1}}{\partial w_5}$$

$$E_{total} = E_{o_1} + E_{o_2}$$

$$\frac{\partial E_{total}}{\partial E_{o_1}} = \frac{\partial}{\partial E_{o_1}}(E_{o_1} + E_{o_2})$$
Let's try to reduce total error by changing $w_5$

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial E_{o1}} \cdot \frac{\partial E_{o1}}{\partial y} \cdot \frac{\partial y}{\partial x} \cdot \frac{\partial y}{\partial w_5}$$
Backpropagation algorithm

Let’s try to reduce total error by changing $w_5$

$$E_{total} = E_{o_1} + E_{o_2}$$

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial E_{o_1}} \ast \frac{\partial E_{o_1}}{\partial out_{o_1}} \ast \frac{\partial out_{o_1}}{\partial in_{o_1}} \ast \frac{\partial in_{o_1}}{\partial w_5}$$

$$f(x, y) = x + y$$

$$\frac{\partial E_{total}}{\partial E_{o_1}} = \frac{\partial}{\partial E_{o_1}} (E_{o_1} + E_{o_2})$$
Backpropagation algorithm

Let’s try to reduce **total error** by changing \( w_5 \)

\[
\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial E_{o_1}} \ast \frac{\partial E_{o_1}}{\partial out_{o_1}} \ast \frac{\partial out_{o_1}}{\partial in_{o_1}} \ast \frac{\partial in_{o_1}}{\partial w_5}
\]

\[
f(x, y) = x + y
\]

\[
\frac{\partial f(x, y)}{\partial x} = \frac{\partial}{\partial x} (x + y)
\]
Backpropagation algorithm

Let’s try to reduce total error by changing $w_5$

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial E_{o1}} \ast \frac{\partial E_{o1}}{\partial out_{o1}} \ast \frac{\partial out_{o1}}{\partial in_{o1}} \ast \frac{\partial in_{o1}}{\partial w_5}$$

$$f(x, y) = x + y$$

$$\frac{\partial f(x, y)}{\partial x} = \frac{\partial}{\partial x} x + \frac{\partial}{\partial x} y$$
Let's try to reduce total error by changing $w_5$

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial E_{o1}} \ast \frac{\partial E_{o1}}{\partial out_{o1}} \ast \frac{\partial out_{o1}}{\partial in_{o1}} \ast \frac{\partial in_{o1}}{\partial w_5}$$

$$f(x, y) = x + y$$

$$\frac{\partial f(x, y)}{\partial x} = \frac{\partial}{\partial x} x + 0$$
Backpropagation algorithm

Let’s try to reduce total error by changing $w_5$

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial E_{o1}} \times \frac{\partial E_{o1}}{\partial out_{o1}} \times \frac{\partial out_{o1}}{\partial in_{o1}} \times \frac{\partial in_{o1}}{\partial w_5}$$

$$f(x, y) = x + y$$

$$\frac{\partial f(x, y)}{\partial x} = 1$$
Backpropagation algorithm

Let’s try to reduce total error by changing $w_5$

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial E_{o1}} \times \frac{\partial E_{o1}}{\partial out_{o1}} \times \frac{\partial out_{o1}}{\partial in_{o1}} \times \frac{\partial in_{o1}}{\partial w_5}$$

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial E_{o1}} = 1$$
Backpropagation algorithm

Let's try to reduce total error by changing $w_5$

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial E_{o_1}} \ast \frac{\partial E_{o_1}}{\partial out_{o_1}} \ast \frac{\partial out_{o_1}}{\partial in_{o_1}} \ast \frac{\partial in_{o_1}}{\partial w_5}$$

What does this actually say?

$$E_{total} = E_{o_1} + E_{o_2}$$

$$\frac{\partial E_{total}}{\partial E_{o_1}} = 1$$
If we increase $E_{o1}$ by 1

What does this actually say?

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial E_{o1}} = 1$$
If we increase $E_{o1}$ by 1

$$E_{total} = (E_{o1} + 1) + E_{o2}$$

$$\frac{\partial E_{total}}{\partial E_{o1}} = 1$$
If we increase $E_{o1}$ by 1

$$E_{total} = (E_{o1} + E_{o2}) + 1$$

$$\frac{\partial E_{total}}{\partial E_{o1}} = 1$$

What does this actually say?
What does this actually say?

If we increase $E_{o1}$ by 1

$$E_{total} = E_{total} + 1$$

$E_{total}$ would also increase by 1

$$\frac{\partial E_{total}}{\partial E_{o1}} = 1$$
Backpropagation algorithm

Let’s try to reduce total error by changing $w_5$

\[
\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial E_{o1}} \times \frac{\partial E_{o1}}{\partial out_{o1}} \times \frac{\partial out_{o1}}{\partial in_{o1}} \times \frac{\partial in_{o1}}{\partial w_5}
\]

$E_{total} = E_{o1} + E_{o2}$

$\text{out}(h_1) = 0.59$

$\text{in}(o_1) = 1.1$

$\text{out}_{o1} = 0.75$

$\text{truth} = 0$

$E_{o1}$

$0.40 (w_5)$

$0.75$

$0.28$
Let's try to reduce total error by changing $w_5$.

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial E_{o1}} \ast \frac{\partial E_{o1}}{\partial out_{o1}} \ast \frac{\partial out_{o1}}{\partial in_{o1}} \ast \frac{\partial in_{o1}}{\partial w_5}$$

We know this value.
Let's try to reduce total error by changing $w_5$.

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial w_5} = 1 \times \frac{\partial E_{o1}}{\partial out_{o1}} \times \frac{\partial out_{o1}}{\partial in_{o1}} \times \frac{\partial in_{o1}}{\partial w_5}$$
**Backpropagation algorithm**

Let’s try to reduce total error by changing $w_5$

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial w_5} = 1 \times \frac{\partial E_{o1}}{\partial out_{o1}} \times \frac{\partial out_{o1}}{\partial in_{o1}} \times \frac{\partial in_{o1}}{\partial w_5}$$
Backpropagation algorithm

Let's try to reduce **total error** by changing $w_5$

\[
\frac{\partial E_{\text{total}}}{\partial w_5} = 1 \times \frac{\partial E_{o_1}}{\partial \text{out}_{o_1}} \times \frac{\partial \text{out}_{o_1}}{\partial \text{in}_{o_1}} \times \frac{\partial \text{in}_{o_1}}{\partial w_5}
\]

\[
E_{o_1} = \frac{1}{2} (\text{truth} - \text{out}_{o_1})^2
\]

\[
E_{\text{total}} = E_{o_1} + E_{o_2}
\]

\[
in(o_1) = 1.1
\]

\[
\text{out}_{o_1} = 0.75
\]

\[
\text{truth} = 0
\]

\[
0.40 (w_5)
\]

\[
0.28
\]
Backpropagation algorithm

Let’s try to reduce total error by changing $w_5$

\[
\frac{\partial E_{total}}{\partial w_5} = 1 \times \frac{\partial E_{o1}}{\partial out_{o1}} \times \frac{\partial out_{o1}}{\partial in_{o1}} \times \frac{\partial in_{o1}}{\partial w_5}
\]

\[
E_{o1} = \frac{1}{2} (truth - out_{o1})^2
\]

\[
\frac{\partial E_{o1}}{\partial out_{o1}} = \frac{\partial}{\partial out_{o1}} \left( \frac{1}{2} (truth - out_{o1})^2 \right)
\]
Let’s try to reduce **total error** by changing $w_5$

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial w_5} = 1 * \frac{\partial E_{o1}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial in_{o1}} * \frac{\partial in_{o1}}{\partial w_5}$$

$$E_{o1} = \frac{1}{2} (x - y)^2$$

$$\frac{\partial E_{o1}}{\partial y} = \frac{1}{2} \frac{\partial}{\partial y} (\frac{1}{2} (x - y)^2)$$
Backpropagation algorithm

Let’s try to reduce total error by changing $w_5$

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial w_5} = 1 * \frac{\partial E_{o1}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial in_{o1}} * \frac{\partial in_{o1}}{\partial w_5}$$

$$E_{o1} = \frac{1}{2}(x - y)^2$$

$$\frac{\partial E_{o1}}{\partial y} = (\frac{1}{2} * 2(x - y)) * \frac{\partial}{\partial y}(x - y)$$
Backpropagation algorithm

Let's try to reduce total error by changing $w_5$

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial w_5} = 1 \ast \frac{\partial E_{o1}}{\partial out_{o1}} \ast \frac{\partial out_{o1}}{\partial in_{o1}} \ast \frac{\partial in_{o1}}{\partial w_5}$$

$$E_{o1} = \frac{1}{2} (x - y)^2$$

$$\frac{\partial E_{o1}}{\partial y} = \left( \frac{1}{2} \ast 2(x - y) \right) \ast (-1)$$
Let’s try to reduce total error by changing $w_5$.

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial w_5} = 1 \ast \frac{\partial E_{o1}}{\partial out_{o1}} \ast \frac{\partial out_{o1}}{\partial in_{o1}} \ast \frac{\partial in_{o1}}{\partial w_5}$$

$$E_{o1} = \frac{1}{2} (x - y)^2$$

$$\frac{\partial E_{o1}}{\partial y} = (x - y) \ast (-1)$$
Let's try to reduce total error by changing $w_5$.

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial w_5} = 1 \ast \frac{\partial E_{o1}}{\partial out_{o1}} \ast \frac{\partial out_{o1}}{\partial in_{o1}} \ast \frac{\partial in_{o1}}{\partial w_5}$$

$$E_{o1} = \frac{1}{2} (x - y)^2$$

$$\frac{\partial E_{o1}}{\partial y} = y - x$$
Backpropagation algorithm

Let's try to reduce total error by changing $w_5$

$$\frac{\partial E_{\text{total}}}{\partial w_5} = 1 \ast \frac{\partial E_{o_1}}{\partial \text{out}_{o_1}} \ast \frac{\partial \text{out}_{o_1}}{\partial \text{in}_{o_1}} \ast \frac{\partial \text{in}_{o_1}}{\partial w_5}$$

$$E_{o_1} = \frac{1}{2} (\text{truth} - \text{out}_{o_1})^2$$

$$\frac{\partial E_{o_1}}{\partial y} = \text{out}_{o_1} - \text{truth}$$
Let’s try to reduce total error by changing $w_5$

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial w_5} = 1 \ast \frac{\partial E_{o1}}{\partial out_{o1}} \ast \frac{\partial out_{o1}}{\partial in_{o1}} \ast \frac{\partial in_{o1}}{\partial w_5}$$

$$E_{o1} = \frac{1}{2} (\text{truth} - out_{o1})^2$$

$$\frac{\partial E_{o1}}{\partial y} = 0.75 - 0$$

**Backpropagation algorithm**
Let's try to reduce **total error** by changing $w_5$

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial w_5} = 1 \times \frac{\partial E_{o1}}{\partial out_{o1}} \times \frac{\partial out_{o1}}{\partial in_{o1}} \times \frac{\partial in_{o1}}{\partial w_5}$$

$$E_{o1} = \frac{1}{2}(truth - out_{o1})^2$$

$$\frac{\partial E_{o1}}{\partial y} = 0.75$$
Let’s try to reduce total error by changing $w_5$

\[
\frac{\partial E_{total}}{\partial w_5} = 1 \times \frac{\partial E_{o1}}{\partial out_{o1}} \times \frac{\partial out_{o1}}{\partial in_{o1}} \times \frac{\partial in_{o1}}{\partial w_5}
\]

We know this value
Backpropagation algorithm

Let’s try to reduce total error by changing $w_5$

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial w_5} = 1 \times 0.75 \times \frac{\partial out_{o1}}{\partial in_{o1}} \times \frac{\partial in_{o1}}{\partial w_5}$$

We know this value
Let's try to reduce the total error by changing $w_5$.

Backpropagation algorithm

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial w_5} = 1 \times 0.75 \times \frac{\partial out_{o1}}{\partial in_{o1}} \times \frac{\partial in_{o1}}{\partial w_5}$$
The Backpropagation algorithm

Let’s try to reduce total error by changing $w_5$

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial w_5} = 1 \times 0.75 \times \frac{\partial out_{o1}}{\partial in_{o1}} \times \frac{\partial in_{o1}}{\partial w_5}$$

$$out_{o1} = \frac{1}{1 + e^{-in_{o1}}}$$

$$\frac{\partial out_{o1}}{\partial in_{o1}} = \frac{\partial}{\partial in_{o1}} \left( \frac{1}{1 + e^{-in_{o1}}} \right)$$
Backpropagation algorithm

Let's try to reduce total error by changing $w_5$

$$\frac{\partial E_{\text{total}}}{\partial w_5} = 1 \cdot 0.75 \cdot \frac{\partial \text{out}_o}{\partial \text{in}_o} \cdot \frac{\partial \text{in}_o}{\partial w_5}$$

$$\text{out}_o = \frac{1}{1 + e^{-\text{in}_o}}$$

There is a shortcut

$$\frac{\partial \text{out}_o}{\partial \text{in}_o} = \frac{\partial}{\partial \text{in}_o} \left( \frac{1}{1 + e^{-\text{in}_o}} \right)$$
Let’s try to reduce total error by changing $w_5$.

$$\frac{\partial E_{total}}{\partial w_5} = 1 \times 0.75 \times \frac{\partial \text{out}_{o1}}{\partial \text{in}_{o1}} \times \frac{\partial \text{in}_{o1}}{\partial w_5}$$

There is a shortcut:

$$\frac{\partial \text{out}_{o1}}{\partial \text{in}_{o1}} = \text{out}_{o1} \times (1 - \text{out}_{o1})$$

The Backpropagation algorithm is shown in the diagram.
**Backpropagation algorithm**

Let’s try to reduce **total error** by changing $w_5$

\[
\frac{\partial E_{total}}{\partial w_5} = 1 \times 0.75 \times \frac{\partial out_{o1}}{\partial in_{o1}} \times \frac{\partial in_{o1}}{\partial w_5}
\]

\[
out_{o1} = \frac{1}{1 + e^{-in_{o1}}}
\]

There is a **shortcut**

\[
\frac{\partial out_{o1}}{\partial in_{o1}} = 0.75 \times (1 - 0.75)
\]
Let’s try to reduce total error by changing $w_5$

$$\frac{\partial E_{total}}{\partial w_5} = 1 \times 0.75 \times \frac{\partial out_{o1}}{\partial in_{o1}} \times \frac{\partial in_{o1}}{\partial w_5}$$

$$out_{o1} = \frac{1}{1 + e^{-in_{o1}}}$$

There is a shortcut

$$\frac{\partial out_{o1}}{\partial in_{o1}} = 0.186$$
Let's try to reduce total error by changing $w_5$.

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial w_5} = 1 \times 0.75 \times 0.186 \times \frac{\partial in_{o1}}{\partial w_5}$$
Let's try to reduce total error by changing $w_5$

$$E_{total} = E_{o_1} + E_{o_2}$$

$$\frac{\partial E_{total}}{\partial w_5} = 1 * 0.75 * 0.186 * \frac{\partial in_{o_1}}{\partial w_5}$$
Let’s try to reduce total error by changing $w_5$

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial w_5} = 1 \times 0.75 \times 0.186 \times \frac{\partial in_{o1}}{\partial w_5}$$
Let’s try to reduce total error by changing $w_5$

$$\frac{\partial E_{total}}{\partial w_5} = 1 \times 0.75 \times 0.186 \times \frac{\partial in_{o1}}{\partial w_5}$$

$$in_{o1} = out_{h1} \times w_5 + out_{h2} \times w_6 + b_{21}$$
Let's try to reduce total error by changing $w_5$.
Let's try to reduce **total error** by changing $w_5$

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial w_5} = 1 \times 0.75 \times 0.186 \times \frac{\partial in_{o1}}{\partial w_5}$$

$$in_{o1} = out_{h1} \times w_5 + out_{h2} \times w_6 + b_{21}$$

$$\frac{\partial in_{o1}}{\partial w_5} = \frac{\partial}{\partial w_5} (out_{h1} \times w_5 + out_{h2} \times w_6 + b_{21})$$
Backpropagation algorithm

Let's try to reduce total error by changing $w_5$

\[
\frac{\partial E_{total}}{\partial w_5} = 1 * 0.75 * 0.186 * \frac{\partial in_{o1}}{\partial w_5}
\]

\[
in_{o1} = \text{out}_{h1} * w_5 + \text{out}_{h2} * w_6 + b_{21}
\]

\[
\frac{\partial in_{o1}}{\partial w_5} = \frac{\partial}{\partial w_5}(\text{out}_{h1} * w_5) + \frac{\partial}{\partial w_5}(\text{out}_{h2} * w_6) + \frac{\partial}{\partial w_5}(b_{21})
\]
Backpropagation algorithm

Let’s try to reduce total error by changing $w_5$

$$\frac{\partial E_{\text{total}}}{\partial w_5} = 1 \times 0.75 \times 0.186 \times \frac{\partial in_{o_1}}{\partial w_5}$$

$$in_{o_1} = out_{h_1} \times w_5 + out_{h_2} \times w_6 + b_{21}$$

$$\frac{\partial in_{o_1}}{\partial w_5} = \frac{\partial}{\partial w_5} (out_{h_1} \times w_5) + \frac{\partial}{\partial w_5} (out_{h_2} \times w_6) + 0$$
Let’s try to reduce total error by changing $w_5$

$$E_{total} = E_{o1} + E_{o2}$$

$$\frac{\partial E_{total}}{\partial w_5} = 1 \cdot 0.75 \cdot 0.186 \cdot \frac{\partial in_{o1}}{\partial w_5}$$

$$in_{o1} = out_{h_1} \cdot w_5 + out_{h_2} \cdot w_6 + b_{21}$$

$$\frac{\partial in_{o1}}{\partial w_5} = \frac{\partial}{\partial w_5} (out_{h_1} \cdot w_5) + 0 + 0$$
Let’s try to reduce total error by changing \( w_5 \)

\[
\frac{\partial E_{total}}{\partial w_5} = 1 \times 0.75 \times 0.186 \times \frac{\partial in_{o_1}}{\partial w_5}
\]

\[
in_{o_1} = out_{h_1} \times w_5 + out_{h_2} \times w_6 + b_{21}
\]

\[
\frac{\partial in_{o_1}}{\partial w_5} = out_{h_1} + 0 + 0
\]
Backpropagation algorithm

Let’s try to reduce total error by changing $w_5$

$$\frac{\partial E_{total}}{\partial w_5} = 1 \times 0.75 \times 0.186 \times \frac{\partial in_{o1}}{\partial w_5}$$

$$in_{o1} = out_{h1} \times w_5 + out_{h2} \times w_6 + b_{21}$$

$$\frac{\partial in_{o1}}{\partial w_5} = 0.59$$
Backpropagation algorithm

Let’s try to reduce total error by changing $w_5$

$$\frac{\partial E_{total}}{\partial w_5} = 1 \times 0.75 \times 0.186 \times 0.59$$
Backpropagation algorithm

Let’s try to reduce total error by changing $w_5$

$$\frac{\partial E_{total}}{\partial w_5} = 1 \times 0.75 \times 0.186 \times 0.59$$
Backpropagation algorithm

Let's try to reduce total error by changing $w_5$

$$\frac{\partial E_{total}}{\partial w_5} = 1 \times 0.75 \times 0.186 \times 0.59 = 0.082$$
Backpropagation algorithm

Let's try to reduce total error by changing $w_5$

What does this actually say?

$$\frac{\partial E_{\text{total}}}{\partial w_5} = 1 \times 0.75 \times 0.186 \times 0.59 = 0.082$$
Backpropagation algorithm

Let's try to reduce total error by changing $w_5$

What does this **actually** say?

$$\frac{\partial E_{total}}{\partial w_5} = 1 \times 0.75 \times 0.186 \times 0.59 = 0.082$$

If we **increase** $w_5$, $E_{total}$ will **increase**
Let's try to reduce total error by changing $w_5$.

What does this **actually** say?

$$\frac{\partial E_{\text{total}}}{\partial w_5} = 1 \times 0.75 \times 0.186 \times 0.59 = 0.082$$

If we **increase** $w_5$, $E_{\text{total}}$ will **increase**.

So we need to **decrease** it!
\[ w_5 = w_5 - \frac{\partial E_{\text{total}}}{\partial w_5} \]

So we need to decrease it!
If update by a lot, we can miss the optimum.

\[ w_5 = w_5 - \frac{\partial E_{\text{total}}}{\partial w_5} \]

So we need to **decrease** it!
\[ w_5 = w_5 - \frac{\partial E_{total}}{\partial w_5} \]

Thus, we make a bit smaller step

So we need to \textbf{decrease} it!
\[ w_5 = w_5 - \eta^* \frac{\partial E_{total}}{\partial w_5} \]

So we need to decrease it!

Thus, we make a bit smaller step
\[ w_5 = w_5 - \eta^* 0.082 \]

Thus, we make a bit smaller step

So we need to decrease it!
$w_5 = 0.4 - \eta^* 0.082$

Thus, we make a bit smaller step

So we need to decrease it!
$w_5 = 0.4 - 0.5 \times 0.082$

Thus, we make a bit **smaller step**

So we need to **decrease** it!
$w_5 = 0.4 - 0.5 * 0.082$

Step size is also called **learning rate**, and it can be really small.

Thus, we make a bit **smaller step**

So we need to **decrease** it!
\( w_5 = 0.3585 \)

Thus, we make a bit smaller step

So we need to decrease it!
What can we do to make the error **smaller**?

\[ E_{total} = 0.28 + 0.025 = 0.305 \]
What can we do to make the error \textbf{smaller}?

$$w_5 = 0.3585$$

$$E_{\text{total}} = 0.28 + 0.025 = 0.305$$
What can we do to make the error \textit{smaller}?

\[ w_5 = 0.3585 \]

\[ E_{total} = 0.28 + 0.025 = 0.305 \]
What can we do to make the error **smaller**?

\[ w_5 = 0.3585 \]

\[ E_{total} = 0.28 + 0.025 = 0.303 \]
What can we do to make the error **smaller**?

This looks fine for one weight ($w_5$) how about all the rest?

$$w_5 = 0.3585$$

$E_{total} = 0.28 + 0.025 = 0.303$
out(h₁) = 0.59
out(h₂) = 0.6

in(o₁) = 1.1

prediction error
truth = 0
0.77 0.025
prediction error
truth = 1
0.745 0.278
After updating all remaining weights total error = \textbf{0.131}
After updating all remaining weights total error = 0.131
Repeating 1000 times decreases it to 0.001
In this example we were dealing with only one weight ($w_5$)

$$w_5 = 0.3585$$

This is not how it is usually done in practice.

Usually updates for weights in the same layer are computed at the same time using matrix operations.

Let’s see how backpropagation works using **matrix** notation.
\[ X \ast W + b \]
(x_1, x_2)

\[ X \ast W + b = P \]

Truth (T)

Predictions (P)

\[ E_{total} \]
How does $W$ influence $E_{total}$?
How does $W$ influence $E_{total}$?

We use the **chain rule** again!
How does $P$ influence $E_{total}$?

We use the **chain rule** again!

$X \ast W + b = P \quad T \rightarrow E_{total}$

Predictions ($P$)

Truth ($T$)

$E_{total}$
How does $P$ influence $E_{total}$?

We use the **chain rule** again!

$$
\frac{\partial E_{total}}{\partial P} = \frac{1}{2} \frac{\partial (T - P)^2}{\partial P}
$$
How does $P$ influence $E_{total}$?

We use the **chain rule** again!

$$
\frac{\partial E_{total}}{\partial P} = \frac{1}{2} \times 2 \times (-1) \times (T - P)
$$

$X \ast W + b = P \rightarrow E_{total}$
How does $P$ influence $E_{total}$?

We use the chain rule again!

$X \ast W + b = P + P_1 + P_2 = E_T$

$\frac{\partial E_{total}}{\partial P} = P - T$

Truth ($T$) | Predictions ($P$) | $E_{total}$
How does \( \mathbf{W} \) influence \( \mathbf{P} \)?

We use the **chain rule** again!

\[
\begin{align*}
\mathbf{X} \ast \mathbf{W} + \mathbf{b} &= \mathbf{P} \\
\frac{\partial E_{\text{total}}}{\partial \mathbf{P}} &= \mathbf{P} - \mathbf{T}
\end{align*}
\]
How does $W$ influence $P$?

We use the **chain rule** again!

$$
\frac{\partial P}{\partial W} = \frac{\partial (X*W + b)}{\partial W} = \frac{\partial E_{total}}{\partial P} = P - T
$$
How does $W$ influence $P$?

We use the **chain rule** again!

$$\frac{\partial P}{\partial W} = X^T$$

$$\frac{\partial E_{total}}{\partial P} = P - T$$

$X \ast W + b = P \rightarrow E_{total}$

$P_1$ $P_2$
How does $W$ influence $E_{total}$?

We use the **chain rule** again!

$$ \frac{\partial E_{total}}{\partial W} = ? $$

$$ \frac{\partial P}{\partial W} = X^T \quad \frac{\partial E_{total}}{\partial P} = P - T $$

$$ X \ast W + b = P \quad T \rightarrow E_{total} $$

$$ \begin{array}{cc}
  x_1 & w_1 \\
  x_2 & w_2 \\
  w_3 & w_4 \\
  b_1 & b_2 \\
  P_1 & P_2 \\
  E_{T} \\
\end{array} $$
How does $W$ influence $E_{total}$?

We use the **chain rule** again!

\[
\frac{\partial E_{total}}{\partial W} = \frac{\partial E_{total}}{\partial P} \ast \frac{\partial P}{\partial W}
\]

\[
\frac{\partial P}{\partial W} = X^T \quad \frac{\partial E_{total}}{\partial P} = P - T
\]

\[
X \ast W + b = P \quad T \quad E_{total}
\]
How does $W$ influence $E_{total}$?

What is the problem here?

$$\frac{\partial E_{total}}{\partial W} = \frac{\partial E_{total}}{\partial P} \ast \frac{\partial P}{\partial W}$$

$$\frac{\partial P}{\partial W} = X^T$$

$$\frac{\partial E_{total}}{\partial P} = P - T$$

$$X \ast W + b = P$$

$$P \rightarrow E_{total}$$
How does $W$ influence $E_{total}$?

What is the problem here?


$$\frac{\partial E_{total}}{\partial W} = \frac{\partial E_{total}}{\partial P} * \frac{\partial P}{\partial W}$$

$$\frac{\partial P}{\partial W} = X^T \quad \frac{\partial E_{total}}{\partial P} = P - T$$

$$X \ast W + b = P \quad E_{total}$$

Truth ($T$)

Predictions ($P$)

$E_{total}$
What is the **problem** here?

\[
\begin{bmatrix} \mathbf{w}_1 & \mathbf{w}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{w}_1 & \mathbf{w}_3 \end{bmatrix} \begin{bmatrix} 2; N \end{bmatrix}
\]

\[
\frac{\partial E_{\text{total}}}{\partial \mathbf{W}} = \frac{\partial E_{\text{total}}}{\partial \mathbf{P}} \times \frac{\partial \mathbf{P}}{\partial \mathbf{W}}
\]

\[
\frac{\partial \mathbf{P}}{\partial \mathbf{W}} = \mathbf{X}^T \quad \frac{\partial E_{\text{total}}}{\partial \mathbf{P}} = \mathbf{P} - \mathbf{T}
\]

\[
\mathbf{X} \times \mathbf{W} + \mathbf{b} = \mathbf{P}
\]

\[
\mathbf{P} \times \mathbf{E}_{\text{total}}
\]
How does $W$ influence $E_{total}$?

What is the problem here?

$[?] \times [?] = [?] \times [2; N]$

$$\frac{\partial E_{total}}{\partial W} = \frac{\partial E_{total}}{\partial P} \times \frac{\partial P}{\partial W}$$

$$\frac{\partial P}{\partial W} = X^T$$

$$\frac{\partial E_{total}}{\partial P} = P - T$$

$$X \times W + b = P$$

$P$ $\rightarrow$ $E_{total}$
How does $W$ influence $E_{total}$?

What is the problem here?

\[
[?;?] = [?;2] \times [2; N]
\]

\[
\frac{\partial E_{total}}{\partial W} = \frac{\partial E_{total}}{\partial P} \times \frac{\partial P}{\partial W}
\]

\[
\frac{\partial P}{\partial W} = X^T \quad \frac{\partial E_{total}}{\partial P} = P - T
\]

\[
X \times W + b = P \quad E_{total}
\]
How does $W$ influence $E_{total}$?

What is the problem here?

$[?;?] = [N;2][2;N]$

$\frac{\partial E_{total}}{\partial W} = \frac{\partial E_{total}}{\partial P} \ast \frac{\partial P}{\partial W}$

$\frac{\partial P}{\partial W} = X^T \quad \frac{\partial E_{total}}{\partial P} = P - T$

$X \ast W + b = P \rightarrow E_{total}$
How does $W$ influence $E_{total}$?

What is the problem here?

$[2; 2] = [N; 2]*[2; N]$}

$$\frac{\partial E_{total}}{\partial W} = \frac{\partial E_{total}}{\partial P} \ast \frac{\partial P}{\partial W}$$

$$\frac{\partial P}{\partial W} = X^T \ast \frac{\partial E_{total}}{\partial P} = P - T$$

$$X \ast W + b = P \rightarrow E_{total}$$
We finally know how $W$ influence $E_{total}$!

$[2; 2] = [2; N] * [N; 2]$

$$\frac{\partial E_{total}}{\partial W} = \frac{\partial P}{\partial W} * \frac{\partial E_{total}}{\partial P}$$

$$\frac{\partial P}{\partial W} = X^T$$

$$\frac{\partial E_{total}}{\partial P} = P - T$$

$$X * W + b = P$$

$$T \rightarrow E_{total}$$
What else is there?

$$X \ast W + b = P$$

Truth ($T$)

Predictions ($P$)

$E_{total}$
What else is there? **Biases!**

\[ X \ast W + b = P \]

**Predictions (P)**

**Truth (T)**

\[ E_{total} \]

\[ \Sigma \]

\[ \Sigma \]

\[ b_1 \quad b_2 \]

\[ E_T \]

\[ (x_1, x_2) \]

\[ (x_1, x_2) \]
In order to estimate how much $b$ influence $E_{total}$ we again need to know how much $E_{total}$ is influenced by $P$.
In order to estimate how much \( b \) influence \( E_{total} \) we again need to know how much \( E_{total} \) is influenced by \( P \).

We already know this!

\[
\frac{\partial E_{total}}{\partial P} = P - T
\]
How does $b$ influence $P$?

$$\frac{\partial P}{\partial b} = \frac{\partial (X \ast W + b)}{\partial b}$$

$$X \ast W + b = P$$

Predictions $(P)$

$$\frac{\partial E_{total}}{\partial P} = P - T$$

Truth $(T)$

$E_{total}$
How does $b$ influence $P$?

$$\frac{\partial P}{\partial b} = 1$$

$$\frac{\partial E_{\text{total}}}{\partial P} = P - T$$
Estimate how much $b$ influence $E_{total}$

$$\frac{\partial P}{\partial b} = 1$$

$$\frac{\partial E_{total}}{\partial P} = P - T$$
Estimate how much \( b \) influence \( E_{total} \)

\[
[?;?] = [?;?] \ast [?;?]
\]

\[
\frac{\partial E_{total}}{\partial b} = \frac{\partial P}{\partial b} \ast \frac{\partial E_{total}}{\partial P}
\]

\[
\frac{\partial P}{\partial b} = 1
\]

\[
\frac{\partial E_{total}}{\partial P} = P - T
\]
Estimate how much $b$ influence $E_{total}$

\[
[1; 2] = [?; ?] \times [N; 2]
\]

\[
\frac{\partial E_{total}}{\partial b} = \frac{\partial P}{\partial b} \times \frac{\partial E_{total}}{\partial P}
\]

\[
\frac{\partial P}{\partial b} = 1
\]

\[
\frac{\partial E_{total}}{\partial P} = P - T
\]

\[
P \rightarrow T \rightarrow E_{total}
\]

\[
b_1 \quad b_2
\]

\[
P_1 \quad P_2
\]

\[
E_T
\]
Estimate how much \( b \) influence \( E_{\text{total}} \)

\[
[1; 2] = [1; 1] \times [N; 2]
\]

\[
\frac{\partial E_{\text{total}}}{\partial b} = \frac{\partial P}{\partial b} \times \frac{\partial E_{\text{total}}}{\partial P}
\]

\[
\frac{\partial P}{\partial b} = 1
\]

\[
\frac{\partial E_{\text{total}}}{\partial P} = P - T
\]

\[
P \rightarrow E_{\text{total}}
\]

\[
T
\]
Estimate how much $b$ influence $E_{total}$.

$[1; 2] = [1; 1] \ast [N; 2]$  

$$\frac{\partial E_{total}}{\partial b} = \frac{\partial P}{\partial b} \ast \frac{\partial E_{total}}{\partial P}$$ 

How is this possible?

$$\frac{\partial P}{\partial b} = 1 \quad \Rightarrow \quad \frac{\partial E_{total}}{\partial P} = P - T$$

$$X \ast W + b = P \quad \Rightarrow \quad T \quad \Rightarrow \quad E_{total}$$

$[1; 2] \ast [N; 2]$
Estimate how much $b$ influence $E_{total}$

$[1; 2] = [1; 1] \times [N; 2]$

How is this possible?

It is not.

$$\frac{\partial E_{total}}{\partial b} = \frac{\partial P}{\partial b} \times \frac{\partial E_{total}}{\partial P}$$
Estimate how much $b$ influence $E_{total}$

$$[1; 2] = [1; 1] \times [N; 2]$$

$$\frac{\partial E_{total}}{\partial b} = \frac{\partial P}{\partial b} \times \frac{\partial E_{total}}{\partial P}$$

How is this possible?

It is not.

How can we make it possible?

$$\frac{\partial P}{\partial b} = 1$$

$$\frac{\partial E_{total}}{\partial P} = P - T$$

$$X \times W + b = P \rightarrow E_{total}$$
Estimate how much $b$ influence $E_{total}$

\[
[1; 2] = \sum [1; 1] \ast [N; 2]
\]

\[
\frac{\partial E_{total}}{\partial b} = \left( \sum \left( \frac{\partial P}{\partial b} \ast \frac{\partial E_{total}}{\partial P} \right) \right)
\]

**Sum across rows**

\[
\frac{\partial P}{\partial b} = 1
\]

\[
\frac{\partial E_{total}}{\partial P} = P - T
\]

\[
X \ast W + b = P \rightarrow E_{total}
\]

Truth ($T$) | Predictions ($P$) | $E_{total}$
Estimate how much $b$ influence $E_{total}$

$$[1; 2] = \sum [1; 1] * [N; 2] = [1; 2]$$

$$\frac{\partial E_{total}}{\partial b} = \sum \left( \frac{\partial P}{\partial b} * \frac{\partial E_{total}}{\partial P} \right)$$

Sum across rows

$$\frac{\partial P}{\partial b} = 1$$

$$\frac{\partial E_{total}}{\partial P} = T - P$$

$$X * W + b = P$$

$$T \rightarrow E_{total}$$
Now that we finally know $db$ and $dW$ we can update original weights and biases.

$W$

\[
\begin{array}{cc}
W_1 & W_3 \\
W_2 & W_4 \\
\end{array}
\]

Updated
Now that we finally know $db$ and $dW$ we can update original weights and biases

\[
\begin{pmatrix}
  W_1 & W_3 \\
  W_2 & W_4 \\
\end{pmatrix} =
\begin{pmatrix}
  W_1 & W_3 \\
  W_2 & W_4 \\
\end{pmatrix} -
\end{equation}
Now that we finally know $db$ and $dW$ we can update original weights and biases.

\[
\text{Updated } W = W - \frac{\partial E_{\text{total}}}{\partial W} * \\
\begin{array}{cc}
W_1 & W_3 \\
W_2 & W_4 \\
\end{array} = \\
\begin{array}{cc}
W_1 & W_3 \\
W_2 & W_4 \\
\end{array} - \\
\begin{array}{cc}
dW_1 & dW_3 \\
dW_2 & dW_4 \\
\end{array}
\]
Now that we finally know $db$ and $dW$ we can update original weights and biases.

$$\text{Updated } W = W - \eta \times \frac{\partial E_{\text{total}}}{\partial W}$$

Learning rate
Now that we finally know $db$ and $dW$ we can update original weights and biases.

$$
\text{Updated} \quad W \quad = \quad W \quad - \quad \eta \quad * \quad \frac{\partial E_{\text{total}}}{\partial W} \\
\text{Updated} \quad b \quad = \quad b \quad - \quad \eta \quad * \quad \frac{\partial E_{\text{total}}}{\partial b}
$$
What would change if we had an activation function? (e.g. ReLu)
What would change if we had an activation function? (e.g. ReLu)

Truth ($T$)  

$E_{total}$

Predictions ($P$)
What would change if we had an activation function? (e.g. ReLu)
What would change if we had an activation function? (e.g. ReLu)

All negative predictions will turn to 0

\[ \text{ReLu} \left( X \ast W + b \right) = P \xrightarrow{T} E_{\text{total}} \]
What would change if we had an activation function? (e.g. ReLu)

All negative predictions will turn to 0

ReLu \( (X * W + b) = P \)

Backpropagation will also change
We finally know how $W$ influence $E_{total}$!

\[
\frac{\partial E_{total}}{\partial W} = \frac{\partial P}{\partial W} \ast \frac{\partial E_{total}}{\partial P} = P - T
\]

\[
X \ast W + b = P \rightarrow E_{total}
\]
How does $W$ influence $P$?

$$\frac{\partial P}{\partial W} =$$

$$X \ast W + b = P \rightarrow E_{total}$$

$$T \rightarrow E_{total}$$

Predictions ($P$)  
Truth ($T$)  

$X$  
$W$  
$b$  
$P$  
$E_{total}$  

$X_1$  $X_2$  
$W_1$ $W_3$  
$W_2$ $W_4$  
$b_1$ $b_2$  
$P_1$ $P_2$  
$E_T$
How does $W$ influence $P$?

$$\frac{\partial P}{\partial W} = \frac{\partial (\max(0, (X \ast W + b)))}{\partial W}$$

$$X \ast W + b = P \rightarrow E_{total}$$
How does $W$ influence $P$?

\[
\frac{\partial P}{\partial W} = \begin{cases} 
\frac{\partial (X \ast W + b)}{\partial W} & \text{for } X > 0 \\
0 & \text{for } X \leq 0 
\end{cases}
\]

$X \ast W + b = P$
How does $W$ influence $P$?

$$\frac{\partial P}{\partial W} = \begin{cases} X^T & \text{for } X > 0 \\ 0 & \text{for } X \leq 0 \end{cases}$$

$$X \ast W + b = P$$

Truth ($T$) $\rightarrow$ Predictions ($P$) $\rightarrow$ $E_{total}$
How does $W$ influence $P$?

This would be $X$ with 0s instead of negative values.

$$\frac{\partial P}{\partial W} = \begin{cases} X^T & \text{for } X > 0 \\ 0 & \text{for } X \leq 0 \end{cases}$$

$$X \ast W + b = P \rightarrow E_{total}$$
Training Neural Networks
(part I)

http://playground.tensorflow.org/
Resources:

Brandon Rohrer’s youtube video: How Deep Neural Networks Work (https://youtu.be/ILsA4nyG7I0)

Stanford University CS231n Convolutional Neural Networks for Visual Recognition (github.io version): http://cs231n.github.io/


Andrej Karpathy’s blog post: Yes you should understand backprop (https://medium.com/@karpathy/yes-you-should-understand-backprop-e2f06eab496b)

Raul Vicente’s lecture: From brain to Deep Learning and back (https://www.uttv.ee/naita?id=23585)

That's all Folks!