

rethinking insurance



June 2020

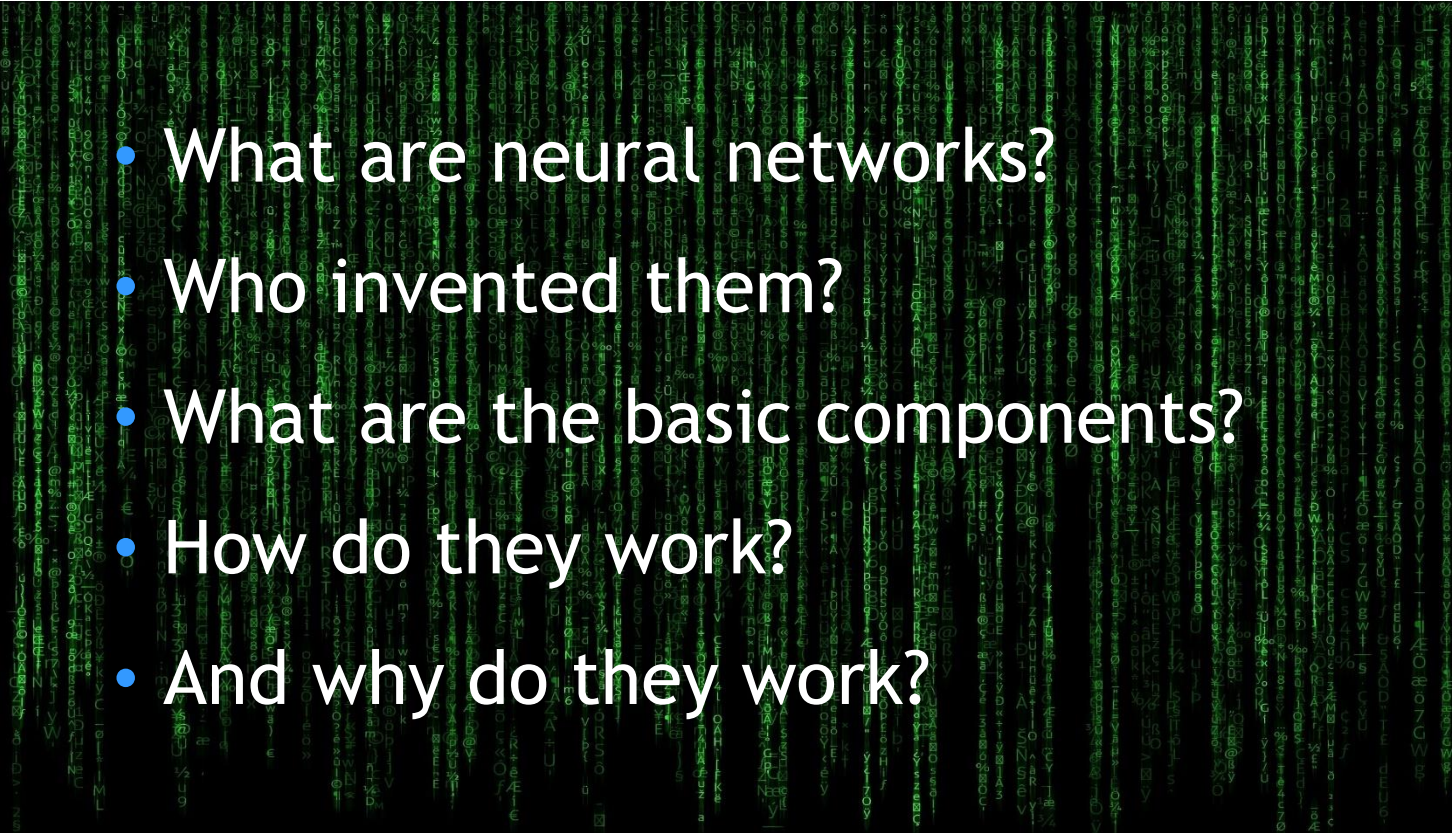
Artificial neural networks basics, part 2 - backpropagation

An elementary introduction

Dr. Stefan Nörtemann, msg life central europe

Introduction

Where do we stand?

- 
- A background image featuring a dense, vertical stream of green and white characters, resembling a digital rain or a computer terminal display, set against a dark background.
- What are neural networks?
 - Who invented them?
 - What are the basic components?
 - How do they work?
 - And why do they work?

Introduction

Where do we stand?

- What are neural networks?
- Who invented them?
- What are the basic components?
- How do they work?
- And why do they work?

Change of the weights to the unit

$$\delta = (1,5 - 0,9) \cdot 1 = 0,6$$

$$\epsilon = 0,1 \text{ (learning rate)}$$

$$\Delta = \epsilon \cdot \delta \cdot f(x)$$

$$\Delta_{HE} = 0,1 \cdot 0,6 \cdot 0,6 = 0,036$$

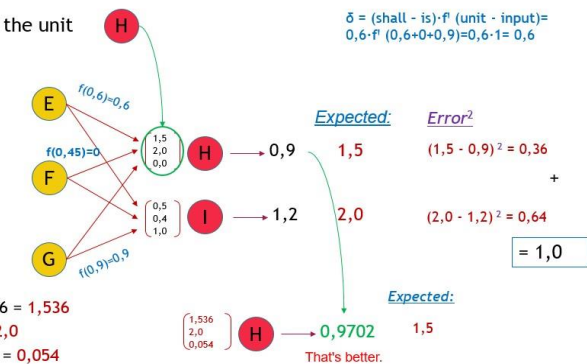
$$\Delta_{HF} = 0,1 \cdot 0,6 \cdot 0 = 0$$

$$\Delta_{HG} = 0,1 \cdot 0,6 \cdot 0,9 = 0,054$$

$$w_{HE} = w_{HE}^{alt} + \Delta_{HE} = 1,5 + 0,036 = 1,536$$

$$w_{HF} = w_{HF}^{alt} + \Delta_{HF} = 2,0 + 0 = 2,0$$

$$w_{HG} = w_{HG}^{alt} + \Delta_{HG} = 0 + 0,054 = 0,054$$



Machine learning

"A lot to learn you still have."

- *(general)*: Automatic generation of knowledge from experience
- *(concrete)*: methods of programming learning algorithms

ML is often classified as a subfield of *artificial intelligence*.

On a very abstract level, the following methods of machine learning are distinguished:

- *supervised* learning
 - *Semi-supervised* learning
 - *active learning* (*Active Learning*)
- *unsupervised* learning
- *reinforcement* Learning



*) Master Yoda, Star Wars, Episode V

Source: <https://mubi.com/films/modern-time>

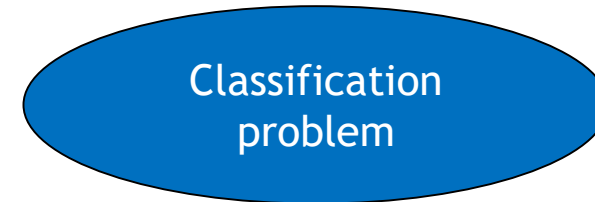
Supervised learning

"In **supervised learning**, the learning algorithm is trained on examples with known results and then used for prediction with new data".

The aim of supervised learning is to predict a target characteristic using explanatory characteristics.

The target characteristic can be either a **nominal characteristic**, for example

- the customer cancels his contract or not
- the customer chooses a capital settlement or not



... or a **numerical characteristic**, for example

- the amount of damage



*) Quoted after: Friedrich Loser, "Data Science/-Mining/-Machine Learning with R for Actuaries, 2016

Training & Test



The following basic procedure has been established for supervised machine learning:

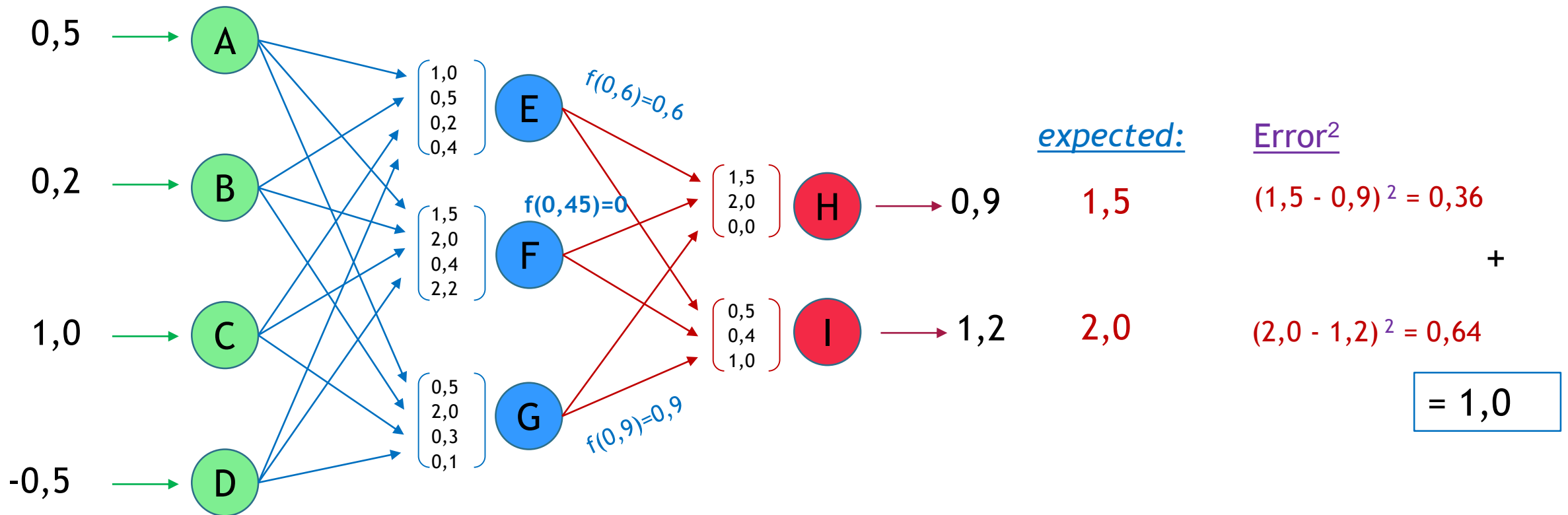
- The dataset is divided into a **training dataset** and a **test dataset**.*
- With the **training data** sets the learning procedure is trained.
- Sometimes it can be useful to train the learning procedure several times using the entire course data stock. Each of these passes is called an "**epoch**".
- With the **test data** sets the trained learning procedure is then tested.
- Only then is the learning procedure applied to **new data** sets.

*) The ratio depends again on the learning process. In practice, a ratio of 90 to 10 (or 80 to 20) training records to test records is often found.

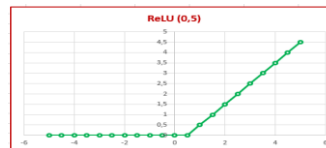
kNN - Training: supervised learning

- preliminary work:
 - Initialization of the kNN = determination of the starting weights (mostly random)
 - Determination of a learning rate
 - Determination of number of epochs
 - Selection of a cost function (also error function), e.g: Sum of squares
- then iteratively for each training data record ($i=1, \dots, n$)
 - Calculation of the output of the network for the input of the i -data set (**forward pass**)
 - comparison with the specified output ("label") of the i -data set
 - Adjustment of the weights by **backpropagation** = minimization of the cost function

Error analysis



Activation function: $f = \text{ReLU} (0.5)$



,no bias (so that the graphic does not become too confusing).

Gradient descent method

Problem: The network does not deliver the correct / expected output for the first training data set

Solution:

- the error is a function of all weights, so:
- improve your net, i.e. reduce the error by changing the weights
- Concretely: Improve the weights by descending the error function in small steps!
- Direction of the steepest ascent or descent
- $\nabla E = \left(\frac{\partial E}{\partial w_{l,k}} \right)_{l,k}$ for weights $w_{l,k}$

Backpropagation - Procedure

- Formula for adjusting the weights

$$\Delta = \varepsilon \cdot \delta \cdot f(x)$$

- In practice one wants to optimize the weights for all training data sets, therefore it has proved to be a good idea not to let the adjustment mechanism "fully" pass through, but to rein it in with a so-called **learning rate** ε .
- $f(x)$ denotes the input in the unit (after the activation function has been applied).
- To determine the delta δ we distinguish between output and hidden layer.
- For an output unit, δ is calculated from the so-called **delta rule**: $\delta = (\text{should-is}) \cdot f'(\text{unit-input})$
- For a unit in a hidden layer it is a bit more complicated: Here the deltas of the subsequent layer are weighted and summed up, all in all **δ results as** follows:

$$\delta_i = \begin{cases} f'(\text{unit input}) \cdot (\text{shall} - \text{is}) & ; \text{ if output-layer} \\ f'(\text{unit input}) \cdot \sum_L \delta_i \cdot w_{li} & ; \text{ if hidden-layer} \end{cases}$$

Backpropagation - History

- Backpropagation = generalization of the delta rule to multi-layer networks.
- For output units, the error is simply the difference between output and expected output (label). With hidden units, the error of the output layer is divided proportionally to the size of the connected link weights. Then you summarize the respective shares of each hidden unit.
- The problem was first recognized in the so-called control theory and solved in the 1960s. There are many fathers here: **Stuart Dreyfus** was able to make a simple derivation for the gradient descent method using the chain rule.
- **Paul Werbos** (*1947, USA) recognized the possibility of applying this to artificial neural networks and introduced the current form of *backpropagation* in his dissertation in 1974. It then took a few more years to show that this is a viable approach. The discovery triggered a (renewed) boom in the use of artificial neural networks in the 1980s.

Zoom into the unit **H** - Backpropagation

Change of the weights to the unit

$$\delta = (1,5 - 0,9) \cdot 1 = 0,6$$

$\varepsilon = 0,1$ (learning rate)

$$\Delta = \varepsilon \cdot \delta \cdot f(x)$$

$$\Delta_{HE} = 0,1 \cdot 0,6 \cdot 0,6 = 0,036$$

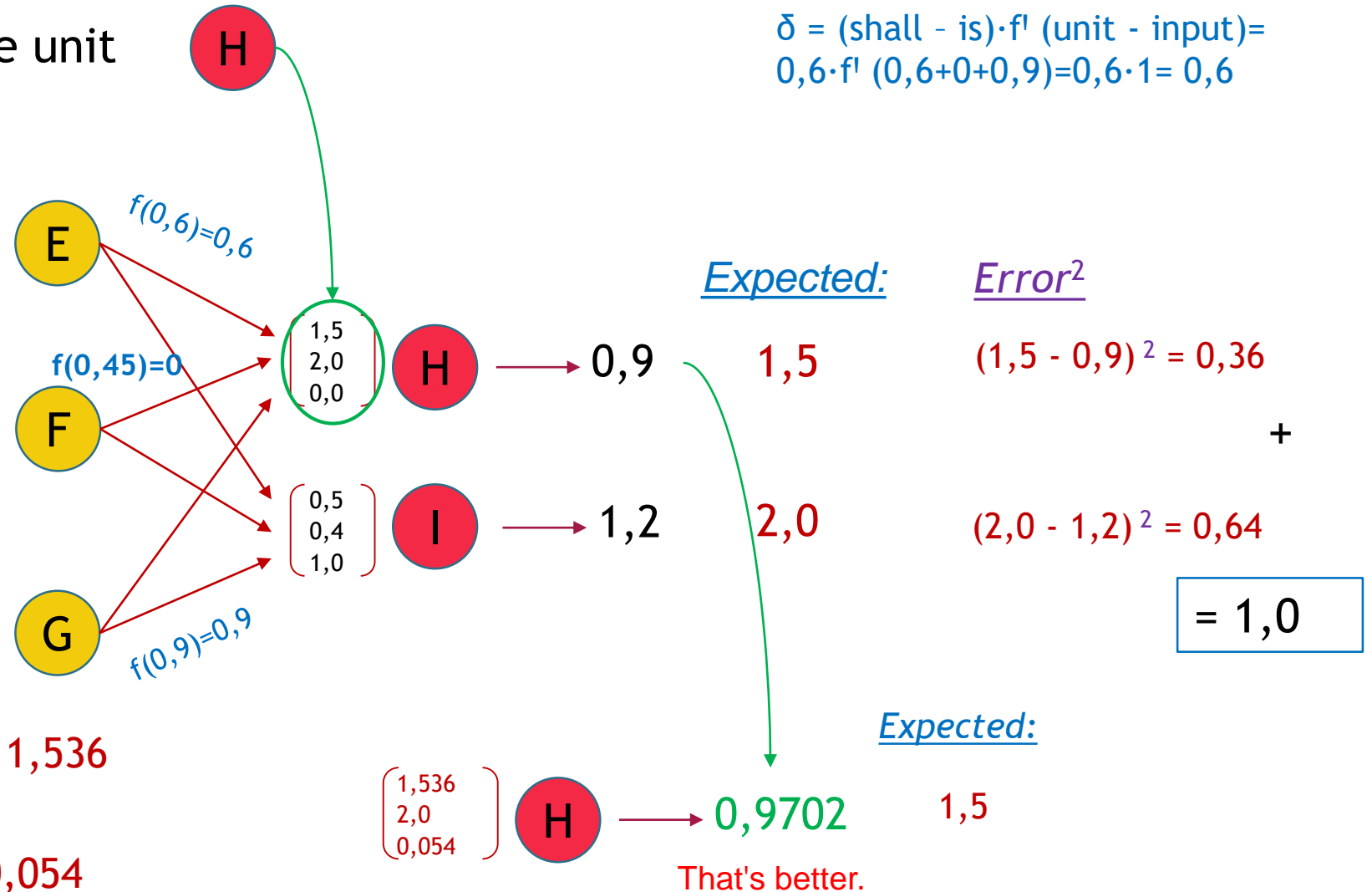
$$\Delta_{HF} = 0,1 \cdot 0,6 \cdot 0 = 0$$

$$\Delta_{HG} = 0,1 \cdot 0,6 \cdot 0,9 = 0,054$$

$$W_{HE} = W_{HE}^{alt} + \Delta_{HE} = 1,5 + 0,036 = 1,536$$

$$W_{HF} = W_{HF}^{alt} + \Delta_{HF} = 2,0 + 0 = 2,0$$

$$W_{HG} = W_{HG}^{alt} + \Delta_{HG} = 0 + 0,054 = 0,054$$



Zoom into the unit **I** - Backpropagation

Change of the weights to the unit

$$\delta = (2,0 - 1,2) \cdot 1 = 0,8$$

$$\varepsilon = 0,1 \text{ (learning rate)}$$

$$\Delta = \varepsilon \cdot \delta \cdot f(x)$$

$$\Delta_{IE} = 0,1 \cdot 0,8 \cdot 0,6 = 0,048$$

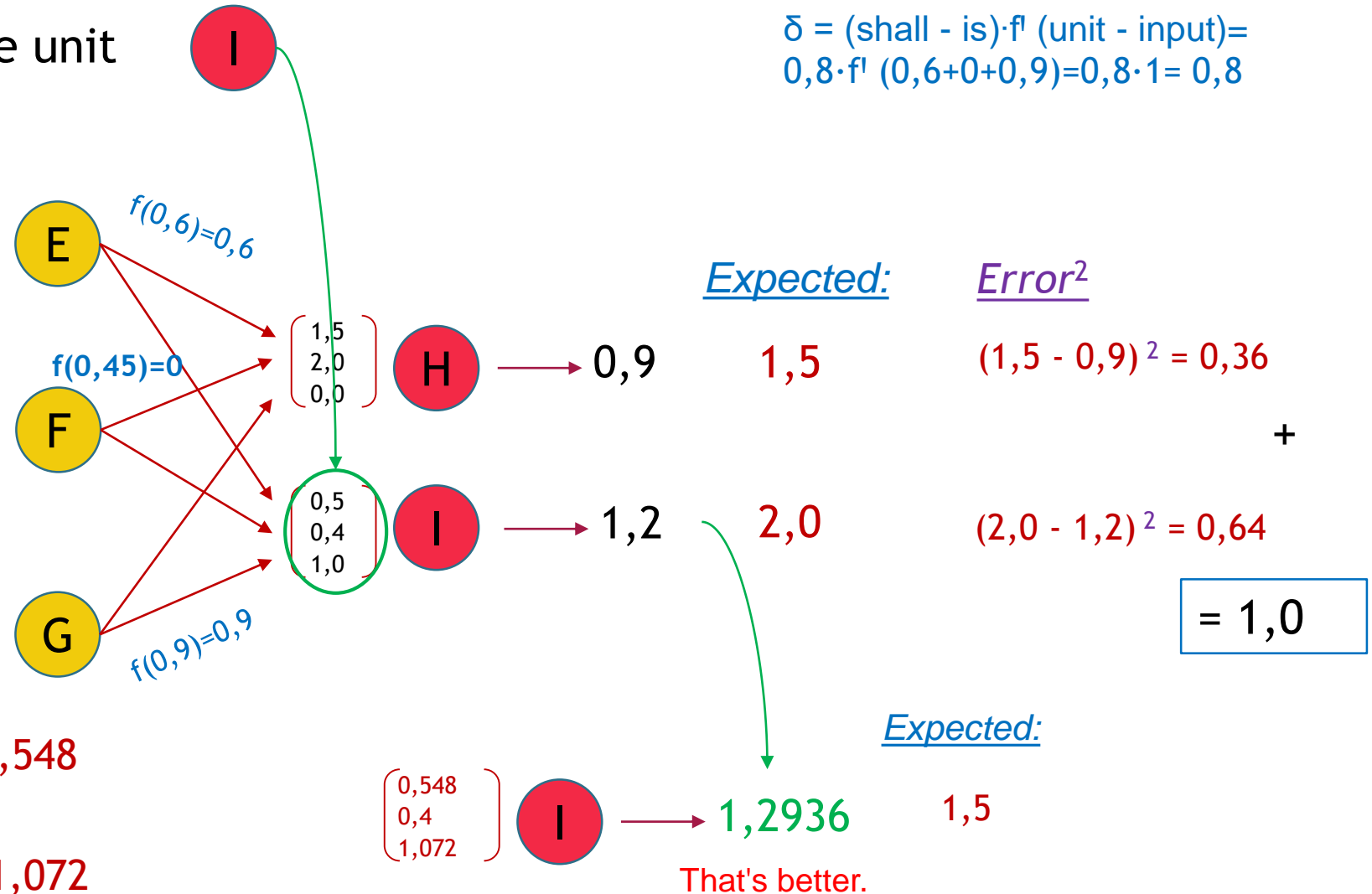
$$\Delta_{IF} = 0,1 \cdot 0,8 \cdot 0 = 0$$

$$\Delta_{IG} = 0,1 \cdot 0,8 \cdot 0,9 = 0,072$$

$$W_{IE} = W_{IE}^{alt} + \Delta_{IE} = 0,5 + 0,048 = 0,548$$

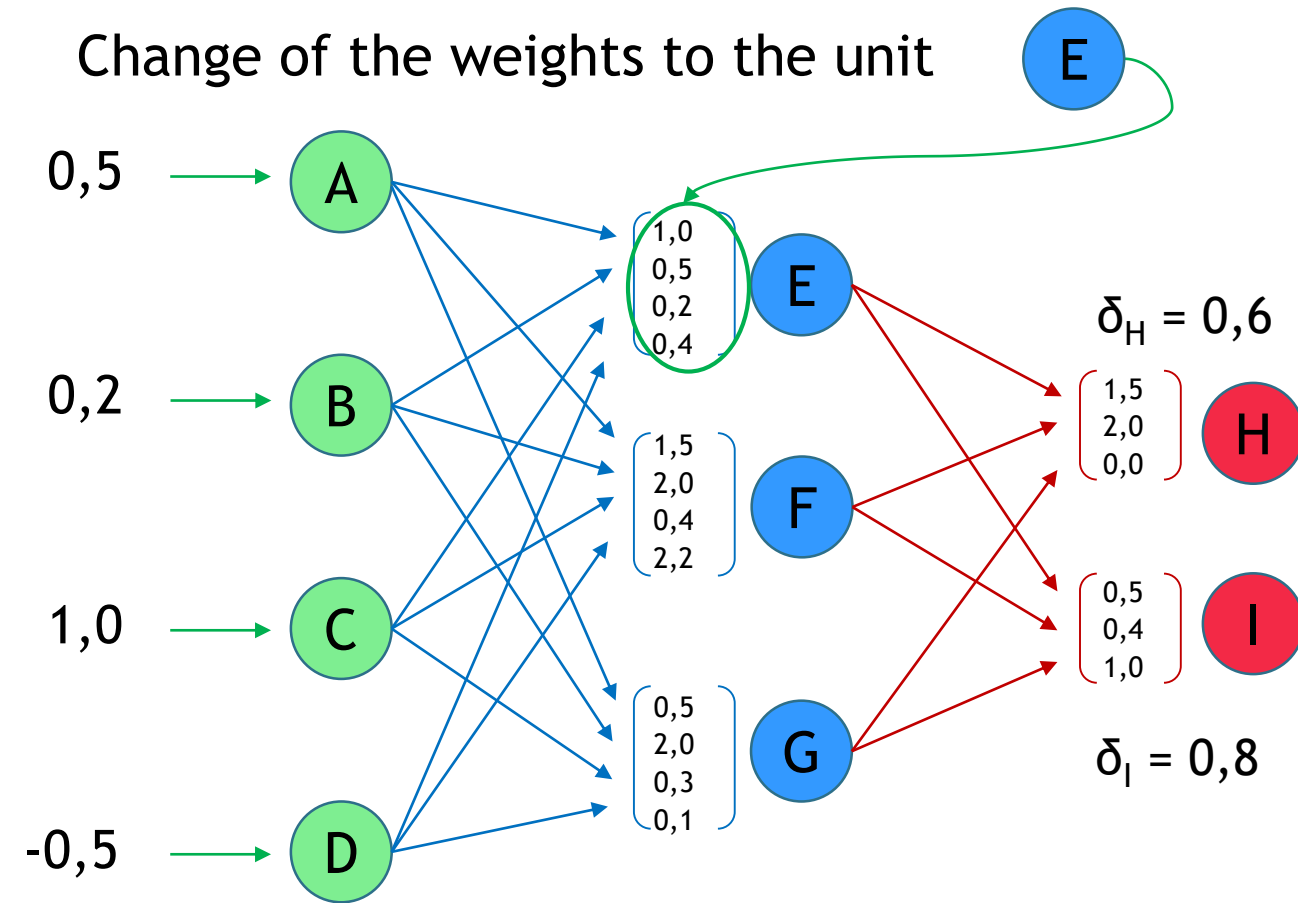
$$W_{IF} = W_{IF}^{alt} + \Delta_{IF} = 0,4 + 0 = 0,4$$

$$W_{IG} = W_{IG}^{alt} + \Delta_{IG} = 1,0 + 0,072 = 1,072$$



Zoom into the unit **E** - Backpropagation

Change of the weights to the unit **E**



$$\delta = f'(\text{unit-input}) \cdot \sum_L \delta_i \cdot w_{li} \quad \Delta = \epsilon \cdot \delta \cdot f(x)$$

$$\delta_H = 1,5 - 0,9 = 0,6$$

$$\delta_I = 2,0 - 1,2 = 0,8$$

$$w_{HE} = 1,5$$

$$w_{IE} = 0,5$$

$$\delta_E = 1 \cdot (0,6 \cdot 1,5 + 0,8 \cdot 0,5) = 1,3$$

$$\epsilon = 0,1 \text{ (learning rate)}$$

$$\Delta_{EA} = 0,1 \cdot 1,3 \cdot 0,5 = 0,065$$

$$\Delta_{EB} = 0,1 \cdot 1,3 \cdot 0,2 = 0,026$$

$$\Delta_{EC} = 0,1 \cdot 1,3 \cdot 1,0 = 0,13$$

$$\Delta_{ED} = 0,1 \cdot 1,3 \cdot (-0,5) = -0,065$$

$$w_{EA} = w_{EA}^{\text{alt}} + \Delta_{EA} = 1,0 + 0,065 = \mathbf{1,065}$$

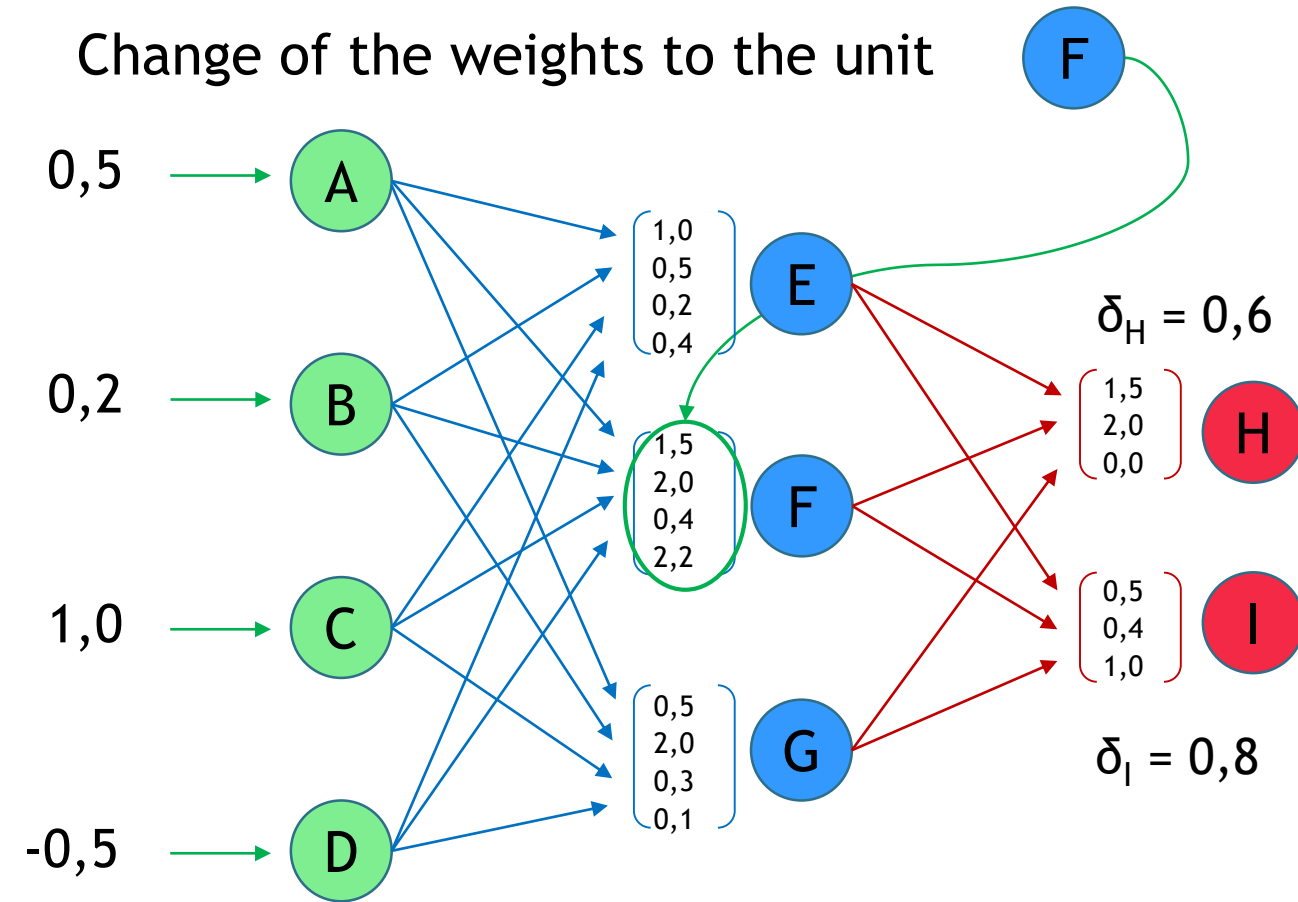
$$w_{EC} = w_{EC}^{\text{alt}} + \Delta_{EC} = 0,2 + 0,130 = \mathbf{0,330}$$

$$w_{EB} = w_{EB}^{\text{alt}} + \Delta_{EB} = 0,5 + 0,026 = \mathbf{0,526}$$

$$w_{ED} = w_{ED}^{\text{alt}} + \Delta_{ED} = 0,4 - 0,065 = \mathbf{0,335}$$

Zoom into the unit **F** - Backpropagation

Change of the weights to the unit



$$w_{FA} = w_{FA}^{\text{alt}} + \Delta_{FA} = 1,5 + 0,0 = 1,5$$

$$w_{FC} = w_{FC}^{\text{alt}} + \Delta_{FC} = 0,4 + 0,0 = 0,4$$

$$w_{FB} = w_{FB}^{\text{alt}} + \Delta_{FB} = 2,0 + 0,0 = 2,0$$

$$w_{FD} = w_{FD}^{\text{alt}} + \Delta_{FD} = 2,2 - 0,0 = 2,2$$

$$\delta = f'(\text{unit-input}) \cdot \sum_L \delta_i \cdot w_{li} \quad \Delta = \epsilon \cdot \delta \cdot f(x)$$

$$\delta_H = 1,5 - 0,9 = 0,6$$

$$\delta_I = 2,0 - 1,2 = 0,8$$

$$w_{HF} = 2,0$$

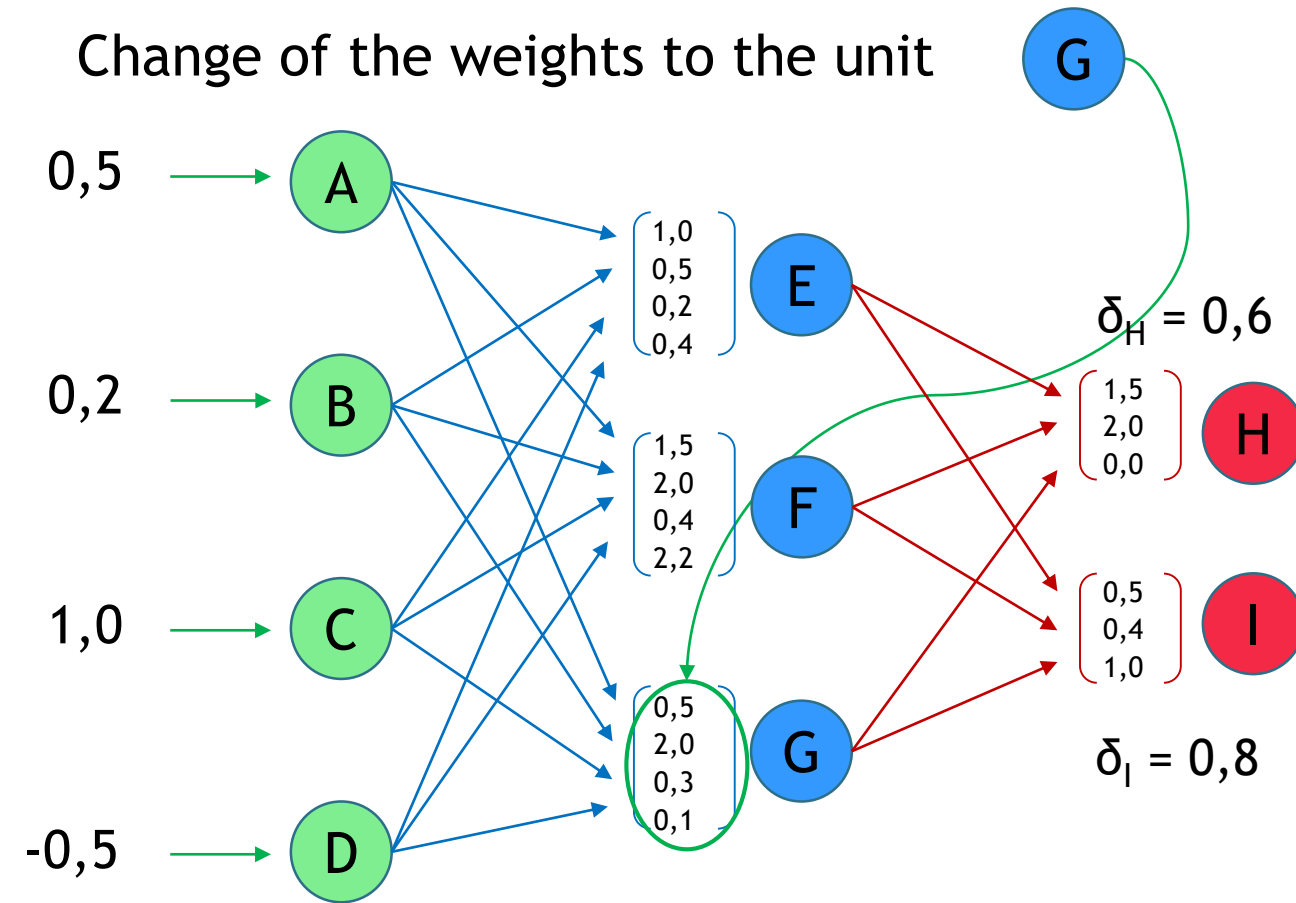
$$w_{IF} = 0,4$$

$$\delta_F = 0 \cdot (0,6 \cdot 2,0 + 0,8 \cdot 0,4) = 0,0$$

$$\epsilon = 0,1 \text{ (learning rate)}$$

Zoom into the unit **G** - Backpropagation

Change of the weights to the unit



$$\delta = f'(\text{unit-input}) \cdot \sum_L \delta_i \cdot w_{li} \quad \Delta = \varepsilon \cdot \delta \cdot f(x)$$

$$\delta_H = 1,5 - 0,9 = 0,6$$

$$\delta_I = 2,0 - 1,2 = 0,8$$

$$w_{HG} = 0,0$$

$$w_{IG} = 1,0$$

$$\delta_G = 1 \cdot (0,6 \cdot 0 + 0,8 \cdot 1,0) = 0,8$$

$$\varepsilon = 0,1 \text{ (learning rate)}$$

$$\Delta_{GA} = 0,1 \cdot 0,8 \cdot 0,5 = 0,04$$

$$\Delta_{GB} = 0,1 \cdot 0,8 \cdot 0,2 = 0,016$$

$$\Delta_{GC} = 0,1 \cdot 0,8 \cdot 1,0 = 0,08$$

$$\Delta_{GD} = 0,1 \cdot 0,8 \cdot (-0,5) = -0,04$$

$$w_{GA} = w_{GA}^{\text{alt}} + \Delta_{GA} = 0,5 + 0,04 = 0,54$$

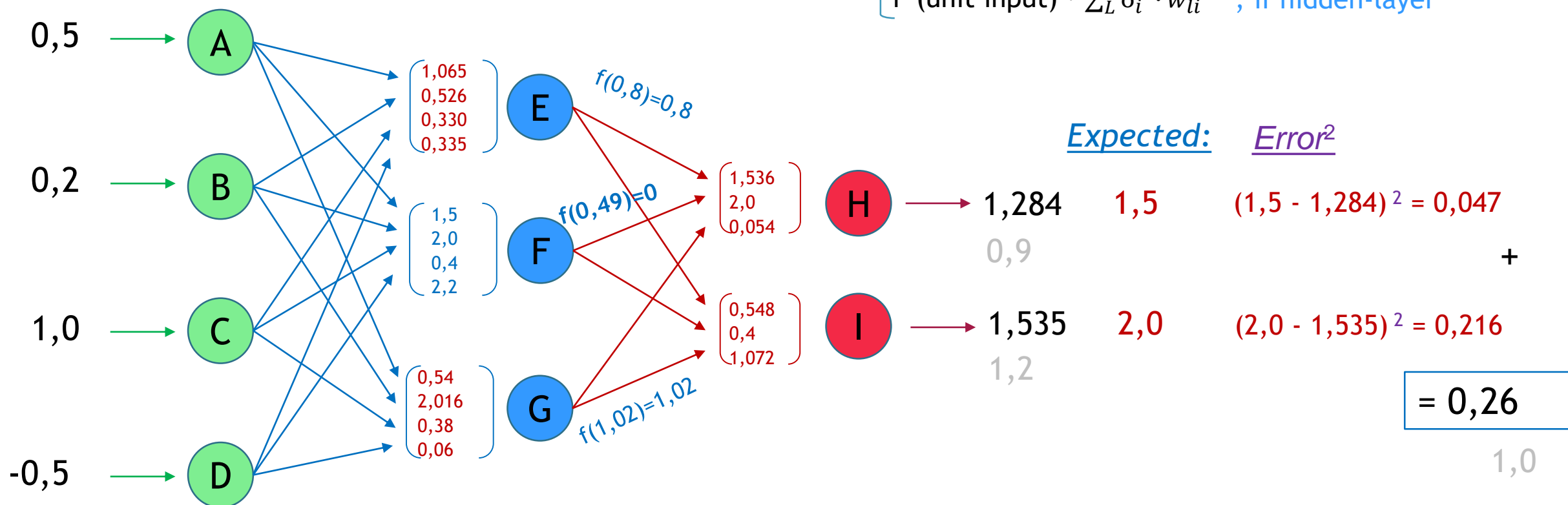
$$w_{GC} = w_{GC}^{\text{alt}} + \Delta_{GC} = 0,3 + 0,08 = 0,38$$

$$w_{GB} = w_{GB}^{\text{alt}} + \Delta_{GB} = 2,0 + 0,016 = 2,016$$

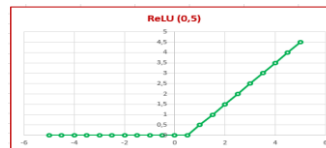
$$w_{GD} = w_{GD}^{\text{alt}} + \Delta_{GD} = 0,1 - 0,04 = 0,06$$

Adjustment of all weights - Backpropagation

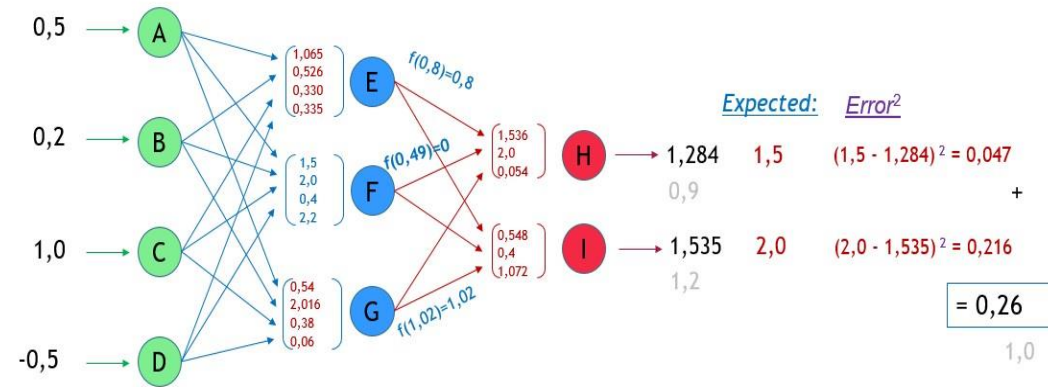
$$\delta_i = \begin{cases} f'(\text{unit input}) \cdot (\text{shall} - \text{is}) & ; \text{ if output-layer} \\ f'(\text{unit input}) \cdot \sum_L \delta_i \cdot w_{li} & ; \text{ if hidden-layer} \end{cases}$$



Activation function: $f = \text{ReLU} (0.5)$



Conclusion



- We will now repeat this process for each record in the **training dataset** ...
- ... and keep adjusting the weights.
- We then check the predictive capability of the neural network using the **test dataset**.
- If we have done everything correctly, the neural network provides us with good predictions for new and unknown data sets.

Thank you for
your attention

Let us talk to one another



Dr. Stefan Noertemann

(Aktuar DAV)

Stefan.Noertemann@msg-life.com

Tel.: +49 (0)711 949581201

www.msg-life.com