Extra:
Neural network training with backward propagation

Neural network calculation algorithms

There are two types of computing involved in neural network technology:

- **Forward propagation:** The actual neural network calculation, which is used by a trained neural network to calculate the results of the input data given to the neural network.

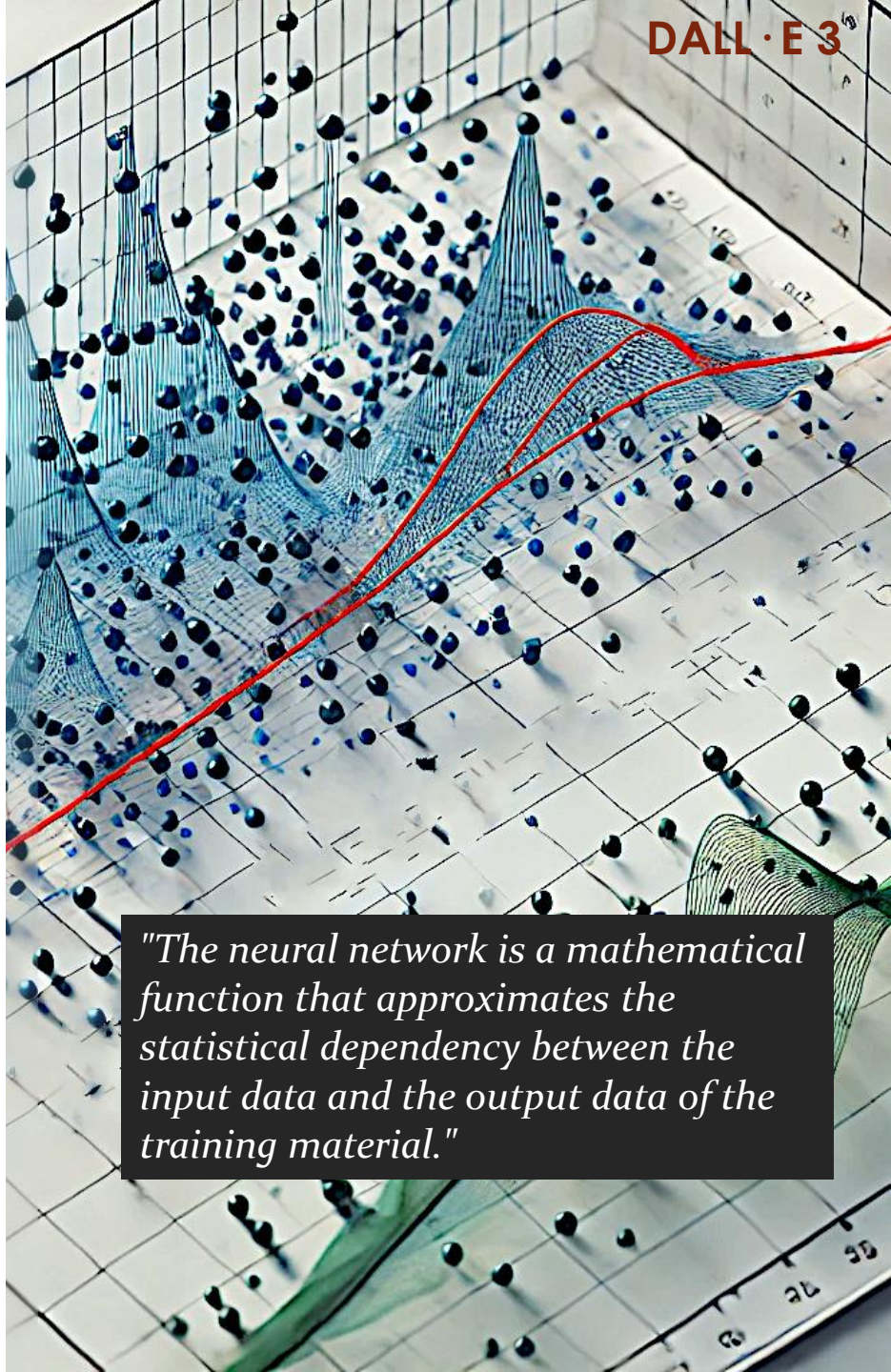
  Forward propagation was presented in detail on the preceding slides.

- **Backward propagation**: The calculation used to calculate optimal values for network parameters.

  The principle of the backward propagation algorithm was briefly explained at the end of the presentation of the forward propagation. The following slides explain the backward propagation algorithm in more detail.

Selection and preparation of the neural network training material

- **The source material** of a neural network is the material that is processed by it. The **task** of the neural network is to calculate the most reliable results possible from the source material.
- The backpropagation algorithm is based on **training material,** which contains a set of paired **input data** and **known target results**.
- **The trainability of a neural network is based on the assumption that there is a statistical dependency between the input data and the target results,** i.e. the values of the input data can be used to predict the values of the target results. N.B., that statistical dependency does not tell us anything about a possible cause-and-effect relationship.
- In other words, the training material is a model according to which a neural network is built. Therefore, **good training material is a good representation of the source material**, i.e. it is assumed to behave like the source material.



About the mathematics of the backpropagation algorithm

- The backpropagation algorithm calculates optimal values for the network parameters. The main steps of the algorithm are as follows:
  - Calculate the partial derivatives of the loss function with respect to all the calculation parameters, that is, weights and biases.
  - Correct the values of the calculation parameters by a small step in the opposite direction to the partial derivatives, i.e. "downhill".
  - Perform the forward calculation again with the new slightly corrected calculation parameter values.
  - Repeat these steps until the loss function has an acceptably low value or some other ending criterion is met.
- As a result, the neural network produces good results even from new and unknown source material, provided that it behaves like the training material.

EU

$$\delta_n^l = a_n^l * (1-a_n^l) * (a_n^l - \hat{a}_n)$$

$$\delta_m^{l-1} = a_m^{l-1} * (1-a_m^{l-1}) * \sum_{n=1}^{k} (w_{mn}^l * \delta_n^l)$$

$$z_n^l = \sum_{m=1}^{k} (a_m^{l-1} * w_{mn}^l) + 1 * b^l$$

$$\frac{\partial C}{\partial w_{nx}^2} = h_n^{out} * y_x^{out} * \left(1-y_x^{out}\right) * \left(y_x^{out} - t_x\right)$$

$$\frac{\partial C}{\partial b^2} = \sum_{x=1}^{2} \left(1 * y_x^{out} * \left(1-y_x^{out}\right) * \left(y_x^{out} - t_x\right)\right)$$

Detailed descriptions of the forward propagation and the backward propagation are available in my article "Neural network calculation algorithms".