Vorher

Nachher

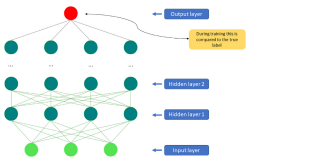

Figure 1. Artificial neural networks are built from simple linear functions followed by nonlinearities. One of the simplest class of neural network is the *multilayer perceptron*, or *feedforward neural network*, originating from the work of Rosenblatt in the 1950s [51]. It is based on simple computational units, called *neurons*, organized in *layers*. Writing i for the ith layer and j for the jth unit of that layer, the output of the jth unit at the ith layer is $z_j^{(i)} = \theta_j^{(i)T} x$. Here x consists of the outputs from the previous layer after they are fed through a simple nonlinear function called an *activation function*, typically a sigmoid function $\sigma(z) = 1/(1+e^{-z})$ or a rectified linear unit $\mathrm{ReLU}(z) = \max(0, z)$ or small variations thereof. Each layer therefore computes a weighted sum of the all the outputs from the neurons in the previous layers, followed by a nonlinearity. These are called the *layer activations*. Each layer activation is fed to the next layer in the network, which performs the same calculation, until you reach the *output layer*, where the network's predictions are produced. In the end, you obtain a hierarchical representation of the input data, where the earlier features tend to be very general, getting increasingly specific towards the output. By feeding the network training data, propagated through the layers, the network is *trained* to perform useful tasks. A training data point (or, typically, a small batch of training points) is fed to the network, the outputs and local derivatives at each node are recorded, and the difference between the output prediction and the true label is measured by an *objective function*, such as *mean absolute error* (L1), *mean squared error* (L2), *cross-entropy loss*, or *Dice loss*, depending on the application. The derivative of the objective function with respect to the output is calculated and used as a feedback signal. The discrepancy is propagated backwards through the network and all the weights are updated to reduce the error. This is achieved using *backward propagation* [52–54], which calculates the gradient of the objective function with respect to the weights in each node using the chain rule together with dynamic programming, and *gradient descent* [55], an optimization algorithm tasked with improving the weights.

2.2 Deep learning

Traditionally, machine learning models are trained to perform useful tasks based on manually designed features extracted from the raw data, or features learned by other simple machine learning models. In deep learning, the computers learn useful representations and features automatically, directly from the raw data, bypassing this manual and difficult step. By far the most common models in deep learning are various variants of artificial neural networks, but there are others. The main common characteristic of deep learning methods is their focus on *feature learning*: automatically learning representations of data. This is the primary difference between deep learning approaches and more "classical" machine learning. Discovering features and performing a task is merged into one problem, and therefore both improved during the same training process. See [38,40] for general overviews of the field.

In medical imaging the interest in deep learning is mostly triggered by *convolutional neural networks* (CNNs) [56],[14] a powerful way to learn useful representations of images and other structured data. Before it became possible to use CNNs efficiently, these features typically had to be engineered by hand, or created by less powerful machine learning models. Once it became possible to use features learned directly from the data, many of the handcrafted image features were typically left by the wayside as they turned out to be almost worthless compared to feature detectors found by CNNs.[15] There are some strong preferences embedded in CNNs based

[14] Interestingly, CNNs was applied in medical image analysis already in the early 90s, e.g. [57], but with limited success.

[15] However, combining hand-engineered features with CNN features is a very reasonable approach when low amounts of training data makes it difficult to learn good features automatically.