VSB TECHNICAL UNIVERSITY OF OSTRAVA | FACULTY OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE | DEPARTMENT OF COMPUTER SCIENCE

Deep Learning

Artificial Neural Networks

Jan Platoš, Radek Svoboda

March 24, 2024

Department of Computer Science
Faculty of Electrical Engineering and Computer Science
VŠB - Technical University of Ostrava

Artificial Neural Networks

Artificial Neural Networks

Human nervous system

- The system is composed of cells, called neurons.
- The neurons are connected to other neurons using synapses.
- The strength of the synapses is affected by learning (external stimuli).

Artificial Neural Networks

- The system is composed of nodes, called neurons.
- The neurons are units of computation that:
  - Receive the inputs from other neurons.
  - Processes these inputs (computes).
  - Set its output.
- The computation process is affected by the input weights and activation function.
- The weights are analogous to the strength of the synapse.
- The weights are affected by the learning process.

Artificial Neural Networks

- The neural networks ability to learn is based on the architecture of the network.
  - Single-layer neural network.
  - Multi-layer neural network.
  - Recurrent neural networks.
  - Kohonen Maps (Self Organized Maps).
  - Convolution networks.
  - Deep neural networks.
  - ...
  -
- The learning is done by presenting the test instances to the network and correction of the output according to the expected output by weight adjusting.

- The basic architecture of neural network with two layers.
  - The input layer has one node for each input attribute.
  - The input node only transmit the input value to the output node.
  - The connection between input and output nodes are weighted.
  - The output layer consist of one output neuron.
  - The output neuron computes the output value.
- The labels are from the set of $\{-1, +1\}$.

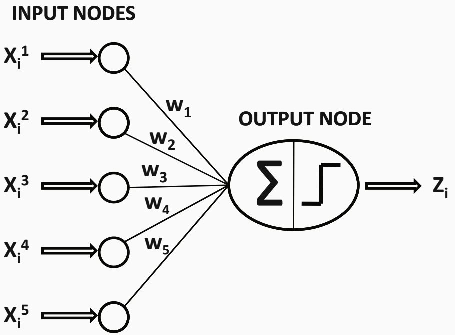


Figure 1: The Perceptron

- The weighted inputs are transformed into output value.
- The value in drawn from the set $\{-1, +1\}$.
- The value may be interpreted as the perceptron prediction of the class variable.
- The weights $W = \{w_1, \ldots, w_d\}$ are modified when the predicted output does not match expected value.

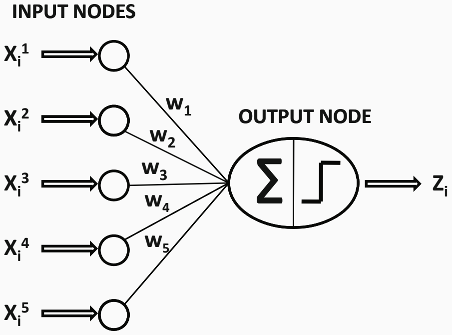


Figure 2: The Perceptron

Artificial Neural Networks - Single-layer Neural Network

- The function learned by the perceptron is referred as *activation function*.
- The function is usually signed linear function (e.g. weighted sum).
- The $W = \{w_1, \ldots, w_d\}$ are the weights for the connections of d different inputs to the output neuron.
- The d is also the dimensionality of the data.
- The b is the bias associated with the activation function.
- The output $z_i \in \{-1, +1\}$ is for the data record $\overline{X_i} = (x_i^1, \ldots, x_i^d)$ computed as follows:

$$z_i = sign\left\{\sum_{j=1}^{d} w_j x_i^j + b\right\} = sign\left\{\overline{W} \cdot \overline{X_i} + b\right\}$$

$$z_i = sign\left\{\sum_{j=1}^{d} w_j x_i^j + b\right\} = sign\left\{\overline{W} \cdot \overline{X_i} + b\right\}$$

- The difference between the prediction of the class value z_i and the real class value y_i is $(y_i - z_i) \in \{-2, 0, 2\}$.
- The result is 0 when the prediction and reality is the same.
- The weight vector $\overline{W}$ and bias b need to be updated, based on the error $(y_i - z_i)$.

$$z_i = sign\left\{\sum_{j=1}^{d} w_j x_i^j + b\right\} = sign\left\{\overline{W} \cdot \overline{X_i} + b\right\}$$

- The learning process is iterative.
- The weight update rule for i-th input point $\overline{X_i}$ in t-th iteration is as follows:

$$\overline{W}^{t+1} = \overline{W}^{t} + \eta(y_i - z_i)\overline{X_i}$$

- The η is the learning rate that regulate the learning speed of the network.
- Each cycle per input points in the learning phase is referred as an *epoch*.

$$\overline{W}^{t+1} = \overline{W}^{t} + \eta(y_i - z_i)\overline{X_i}$$

- The incremental term $(y_i - z_i)\overline{X_i}$ is the approximation of the negative of the gradient of the least=squares prediction error $(y_i - z_i)^2 = (y_i - sign(\overline{W} \cdot \overline{X_i} - b))^2$
- The update is performed on a tuple-by-tuple basis not a global over whole dataset.
- The perceptron may be considered a modified version of a gradient descent method that minimizes the squared error of prediction.

$$\overline{W}^{t+1} = \overline{W}^{t} + \eta(y_i - z_i)\overline{X_i}$$

- The size of the η affect the speed of the convergence and the quality of the solution.
    - The higher value of η means faster convergence, but suboptimal solution may be found.
    - Lower values of η results in higher-quality solutions with slow convergence.
- In practice, η is decreased systematically with increasing number of epochs performed.
- Higher values at the beginning allows bigger jumps in weight space and lower values later allows precise setting of the weights.

- The perceptron, with only one computational neuron produces only a linear model.
- Multi-layer perceptron adds a hidden layer beside the input and output layer.
- The hidden layer itself may consist of different type of topologies (e.g. several layers).

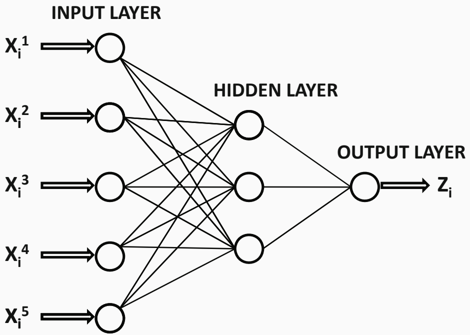


Figure 3: Multi-layer neural network

Artificial Neural Networks - Multi-layer Neural Network

- The output of nodes in one layer feed the inputs of the nodes in the next layer - this behavior is called *feed-forward network*.
- The nodes in one layer are fully connected to the neurons in the previous layer.

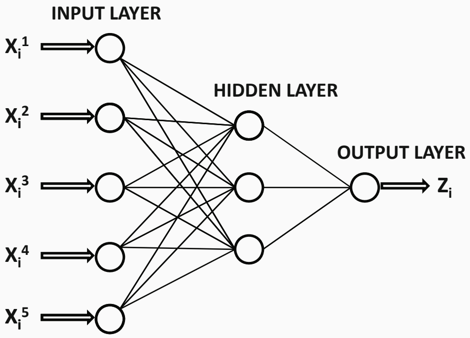


Figure 4: Multi-layer neural network

- The topology of the multi-layer feed-forward network is determined automatically.
- The perceptron may be considered as a single-layer feed-forward neural network.
- The number of layers and the number of nodes in each layer have to be determined manually.
- Standard multi-layer network uses only one hidden layer, i.e. this is considered as a two-layer feed forward neural network.
- The activation function is not limited to linear signed weighted sum, other functions such as logistic, sigmoid or hyperbolic tangents are allowed.

Sigmoid/Logistic function	$\sigma(x) = \frac{1}{1+e^{-x}}$
TanH	$\tanh(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})}$
ReLU (Rectified linear unit)	$f(x) = \begin{cases} 0 & for x \leq 0 \\ x & for x \geq 0 \end{cases}$
Sinc	$f(x) = \begin{cases} 1 & for x = 0 \\ \frac{sin(x)}{x} & for x \neq 0 \end{cases}$
Gaussian	$f(x) = e^{x^2}$
Softmax	$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$

Learning algorithm

- The learning phase is more complicated than the one in perceptron.
- The biggest problem is the get the error in the hidden layer, because the direct class label is not defined on this level.
- Some kind of *feedback* is required from the nodes in the forward layer to the nodes in earlier layers about the *expected* outputs and corresponding errors.
- This principle is realized in the *back-propagation* algorithm.

Back-propagation algorithm

- *Forward phase:*
  - The input is fed into input neurons.
  - The computed values propagates using current weights to the next layers.
  - The final predicted output is compared with the class and the error is determined.
- *Backward phase:*
  - The main goal is to learn weights in the backward direction by providing the error estimation from later layers to the earlier layers.
  - The estimation in the hidden layer is computed as a function of the error estimate and weight is the layers ahead.
  - The error is estimated again using the gradient method.
  - The process complicates the usage of non-linear functions in the inner nodes.

- Lets have an example multi-layer neural network with single output neuron.
- In each iteration do take the i-th input vector.
- Pass it through the networks using the forward pass.
- Compare the i-th output o_i to the expected value y_i.
- Compute the error and update the weight using the learning rate η.
- The goal is to optimize the weights w_i to minimize the error function of the differences between y_i and o_i.

- The error function E over whole dataset of size n may be defined as follows:

$$E = \frac{1}{2}\sum_{i=0}^{n}(y_i - o_i)^2$$

- The weights of the neurons must be adapted according to the error produced by the neuron weight.

$$w_{i+1} = -\eta\frac{\partial E}{\partial w_i} + \mu w_i$$

- The partial derivation may be computed using so called chain rule.

$$\frac{\partial E}{\partial w_i} = \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial z} \cdot \frac{\partial z}{\partial w_i}$$

- where

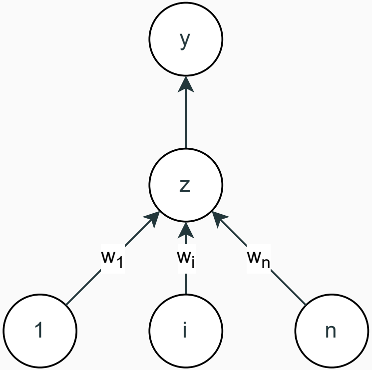
$$y = \frac{1}{1 + e^{-\lambda z}} \qquad z = \sum_{i=0}^{m} w_i x_i$$

- therefore

$$\frac{\partial z}{\partial w_i} = x_i \qquad \frac{\partial y}{\partial z} = y \cdot (1 - y)\lambda$$

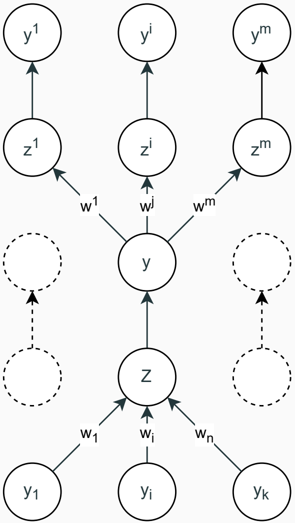
- The first partial derivation computation differs for neuron from output and hidden layer.
- The solution for the output layer and i-th output is as follows:

$$\frac{\partial E}{\partial y} = (y_i - o_i)$$



- The solution for the hidden layer and *i*-th output is as follows:

$$\frac{\partial E}{\partial y} = \sum_{j=0}^{m} \frac{\partial E}{\partial z^j} \cdot \frac{\partial z^j}{\partial y} = \sum_{j=0}^{m} \frac{\partial E}{\partial z^j} \cdot w^j$$



Artificial Neural Networks - Multi-layer Neural Network

- It has ability not only to capture decision boundaries of arbitrary shapes, but also non-contiguous class distribution with different decision boundaries in different regions.
- With increasing number of nodes and layers, virtually any function may be approximated.
- **The neural networks are universal function approximate**.

Artificial Neural Networks - Multi-layer Neural Network

- This generality brings several challenges that have to be dealt with:
  - The design of the topology presents many trade=off challenges for the analyst.
  - Higher number of nodes and layers provides greater generality but also the risk of over-fitting.
  - There is very little guidance provided from the data.
  - The neural network has poor interpretability associated with the classification process.
  - The learning process is very slow and sensitive to the noise.
  - Larger networks has very slow learning process.

Other learning algorithms:

- Gradient descent
- Stochastic Gradient Descent
  - Momentum
  - Averaging
  - AdaGrad
  - RMSProp
  - Adam
- Newton's method
- Conjugate gradient
- Quasi-Newton method
- Levenberg-Marquardt algorithm

Artificial Neural Networks - Multi-layer Neural Network

- The multi-layer neural network is more powerful than kernel SVM in its ability to capture arbitrary functions.
- It has ability not only to capture decision boundaries of arbitrary shapes, but also non-contiguous class distribution with different decision boundaries in different regions.
- With increasing number of nodes and layers, virtually any function may be approximated.
- **The neural networks are universal function approximators**.

- This generality brings several challenges that have to be dealt with:
    - The design of the topology presents many trade=off challenges for the analyst.
    - Higher number of nodes and layers provides greater generality but also the risk of over-fitting.
    - There is very little guidance provided from the data.
    - The neural network has poor interpretability associated with the classification process.
    - The learning process is very slow and sensitive to the noise.
    - Larger networks has very slow learning process.

Questions?