1

Introduction to the Artificial Neural Networks

Andrej Krenker[1], Janez Bešter[2] and Andrej Kos[2]
[1]*Consalta d.o.o.*
[2]*Faculty of Electrical Engineering, University of Ljubljana*
Slovenia

1. Introduction

An Artificial Neural Network (ANN) is a mathematical model that tries to simulate the structure and functionalities of biological neural networks. Basic building block of every artificial neural network is artificial neuron, that is, a simple mathematical model (function). Such a model has three simple sets of rules: multiplication, summation and activation. At the entrance of artificial neuron the inputs are weighted what means that every input value is multiplied with individual weight. In the middle section of artificial neuron is sum function that sums all weighted inputs and bias. At the exit of artificial neuron the sum of previously weighted inputs and bias is passing trough activation function that is also called transfer function (Fig. 1.).

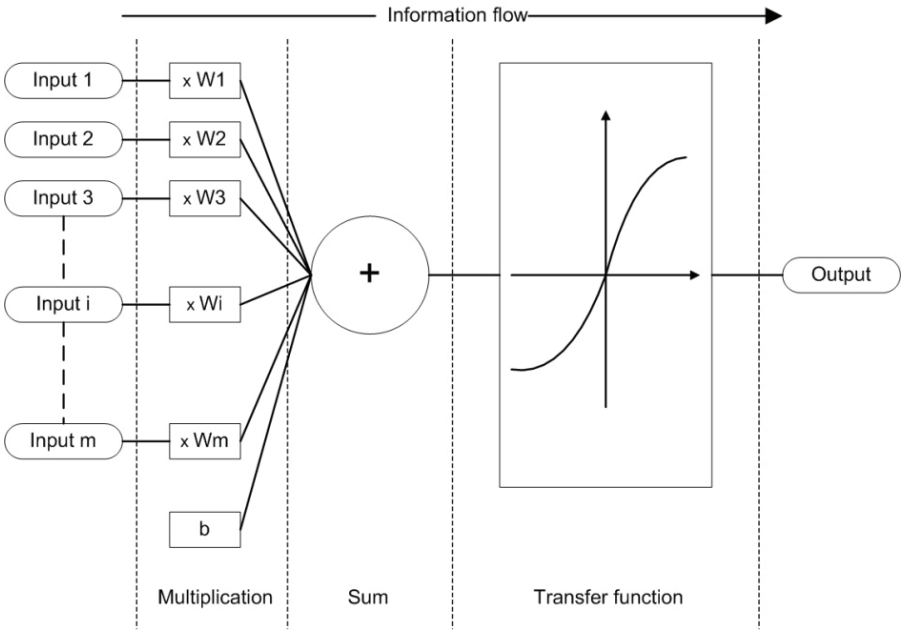

Fig. 1. Working principle of an artificial neuron.

Although the working principles and simple set of rules of artificial neuron looks like nothing special the full potential and calculation power of these models come to life when we start to interconnect them into artificial neural networks (Fig. 2.). These artificial neural networks use simple fact that complexity can grown out of merely few basic and simple rules.

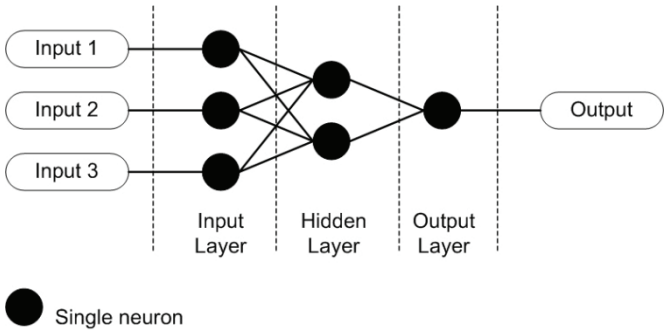

Fig. 2. Example of simple artificial neural network.

In order to fully harvest the benefits of mathematical complexity that can be achieved through interconnection of individual artificial neurons and not just making system complex and unmanageable we usually do not interconnect these artificial neurons randomly. In the past, researchers have come up with several "standardised" topographies of artificial neural networks. These predefined topographies can help us with easier, faster and more efficient problem solving. Different types of artificial neural network topographies are suited for solving different types of problems. After determining the type of given problem we need to decide for topology of artificial neural network we are going to use and then fine-tune it. We need to fine-tune the topology itself and its parameters.

Fine tuned topology of artificial neural network does not mean that we can start using our artificial neural network, it is only a precondition. Before we can use our artificial neural network we need to teach it solving the type of given problem. Just as biological neural networks can learn their behaviour/responses on the basis of inputs that they get from their environment the artificial neural networks can do the same. There are three major learning paradigms: supervised learning, unsupervised learning and reinforcement learning. We choose learning paradigm similar as we chose artificial neuron network topography - based on the problem we are trying to solve. Although learning paradigms are different in their principles they all have one thing in common; on the basis of "learning data" and "learning rules" (chosen cost function) artificial neural network is trying to achieve proper output response in accordance to input signals.

After choosing topology of an artificial neural network, fine-tuning of the topology and when artificial neural network has learn a proper behaviour we can start using it for solving given problem. Artificial neural networks have been in use for some time now and we can find them working in areas such as process control, chemistry, gaming, radar systems, automotive industry, space industry, astronomy, genetics, banking, fraud detection, etc. and solving of problems like function approximation, regression analysis, time series prediction, classification, pattern recognition, decision making, data processing, filtering, clustering, etc., naming a few.

As topic of artificial neural networks is complex and this chapter is only informative nature we encourage novice reader to find detail information on artificial neural networks in (Gurney, 1997; Kröse & Smagt 1996; Pavešić, 2000; Rojas 1996).

2. Artificial neuron

Artificial neuron is a basic building block of every artificial neural network. Its design and functionalities are derived from observation of a biological neuron that is basic building block of biological neural networks (systems) which includes the brain, spinal cord and peripheral ganglia. Similarities in design and functionalities can be seen in Fig.3. where the left side of a figure represents a biological neuron with its soma, dendrites and axon and where the right side of a figure represents an artificial neuron with its inputs, weights, transfer function, bias and outputs.

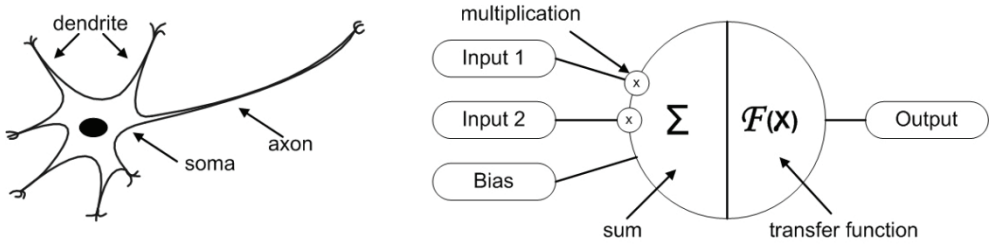

Fig. 3. Biological and artificial neuron design.

In case of biological neuron information comes into the neuron via dendrite, soma processes the information and passes it on via axon. In case of artificial neuron the information comes into the body of an artificial neuron via inputs that are weighted (each input can be individually multiplied with a weight). The body of an artificial neuron then sums the weighted inputs, bias and "processes" the sum with a transfer function. At the end an artificial neuron passes the processed information via output(s). Benefit of artificial neuron model simplicity can be seen in its mathematical description below:

$$y(k) = F\left(\sum_{i=0}^{m} w_i(k) \cdot x_i(k) + b\right) \tag{1}$$

Where:

- $x_i(k)$ is input value in discrete time k where i goes from 0 to m,
- $w_i(k)$ is weight value in discrete time k where i goes from 0 to m,
- b is bias,
- F is a transfer function,
- $y_i(k)$ is output value in discrete time k.

As seen from a model of an artificial neuron and its equation (1) the major unknown variable of our model is its transfer function. Transfer function defines the properties of artificial neuron and can be any mathematical function. We choose it on the basis of problem that artificial neuron (artificial neural network) needs to solve and in most cases we choose it from the following set of functions: *Step function*, *Linear function* and *Non-linear (Sigmoid) function*.

Step function is binary function that has only two possible output values (e.g. zero and one). That means if input value meets specific threshold the output value results in one value and if specific threshold is not meet that results in different output value. Situation can be described with equation (2).

$$y = \begin{cases} 1 \; if \; w_i x_i \geq threshold \\ 0 \; if \; w_i x_i \; < threshold \end{cases} \tag{2}$$

When this type of transfer function is used in artificial neuron we call this artificial neuron *perceptron*. Perceptron is used for solving classification problems and as such it can be most commonly found in the last layer of artificial neural networks. In case of linear transfer function artificial neuron is doing simple linear transformation over the sum of weighted inputs and bias. Such an artificial neuron is in contrast to perceptron most commonly used in the input layer of artificial neural networks. When we use non-linear function the sigmoid function is the most commonly used. Sigmoid function has easily calculated derivate, which can be important when calculating weight updates in the artificial neural network.

3. Artificial Neural Networks

When combining two or more artificial neurons we are getting an artificial neural network. If single artificial neuron has almost no usefulness in solving real-life problems the artificial neural networks have it. In fact artificial neural networks are capable of solving complex real-life problems by processing information in their basic building blocks (artificial neurons) in a non-linear, distributed, parallel and local way.

The way that individual artificial neurons are interconnected is called topology, architecture or graph of an artificial neural network. The fact that interconnection can be done in numerous ways results in numerous possible topologies that are divided into two basic classes. Fig. 4. shows these two topologies; the left side of the figure represent simple feed-forward topology (acyclic graph) where information flows from inputs to outputs in only one direction and the right side of the figure represent simple recurrent topology (semi-cyclic graph) where some of the information flows not only in one direction from input to output but also in opposite direction. While observing Fig. 4. we need to mention that for easier handling and mathematical describing of an artificial neural network we group individual neurons in layers. On Fig. 4. we can see input, hidden and output layer.

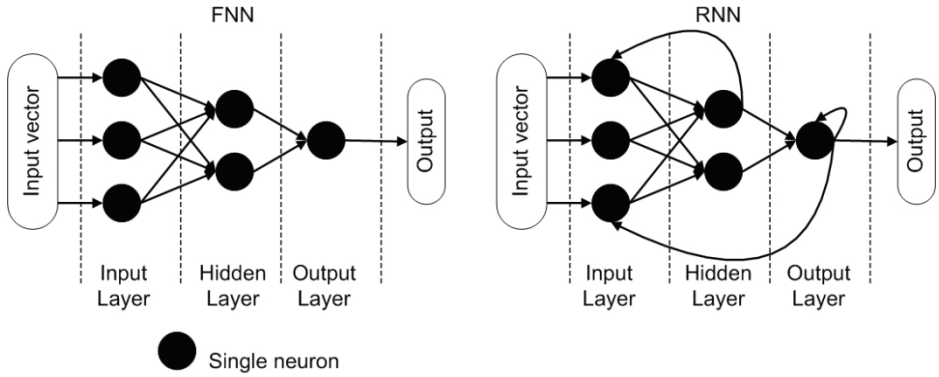

Fig. 4. Feed-forward (FNN) and recurrent (RNN) topology of an artificial neural network.

When we choose and build topology of our artificial neural network we only finished half of the task before we can use this artificial neural network for solving given problem. Just as biological neural networks need to learn their proper responses to the given inputs from the environment the artificial neural networks need to do the same. So the next step is to learn proper response of an artificial neural network and this can be achieved through learning (supervised, un-supervised or reinforcement learning). No matter which method we use, the task of learning is to set the values of weight and biases on basis of learning data to minimize the chosen cost function.

3.1 Feed-forward Artificial Neural Networks

Artificial neural network with feed-forward topology is called *Feed-Forward* artificial neural network and as such has only one condition: information must flow from input to output in only one direction with no back-loops. There are no limitations on number of layers, type of transfer function used in individual artificial neuron or number of connections between individual artificial neurons. The simplest feed-forward artificial neural network is a single perceptron that is only capable of learning linear separable problems. Simple multi-layer feed-forward artificial neural network for purpose of analytical description (sets of equations (3), (4) and (5)) is shown on Fig. 5.

$$\begin{aligned} n_1 &= F_1(w_1 x_1 + b_1) \\ n_2 &= F_2(w_2 x_2 + b_2) \\ n_3 &= F_2(w_2 x_2 + b_2) \\ n_4 &= F_3(w_3 x_3 + b_3) \end{aligned} \tag{3}$$

$$\begin{aligned} m_1 &= F_4(q_1 n_1 + q_2 n_2 + b_4) \\ m_2 &= F_5(q_3 n_3 + q_4 n_4 + b_5) \end{aligned} \tag{4}$$

$$y = F_6(r_1 m_1 + r_2 m_2 + b_6)$$

$$y = F_6\begin{bmatrix} r_1\big(F_4\big[q_1 F_1[w_1 x_1 + b_1] + q_2 F_2[w_2 x_2 + b_2]\big] + b_4\big) + \cdots \\ \ldots + r_2(F_5[q_3 F_2[w_2 x_2 + b_2] + q_4 F_3[w_3 x_3 + b_3] + b_5]) + b_6 \end{bmatrix} \tag{5}$$

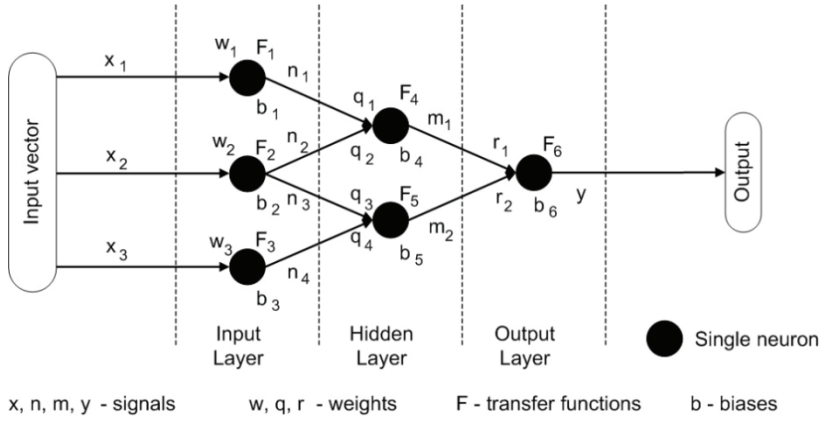

Fig. 5. Feed-forward artificial neural network.

As seen on Fig. 5 and corresponding analytical description with sets of equations (3), (4) and (5) the simple feed-forward artificial neural network can led to relatively long mathematical descriptions where artificial neural networks' parameters optimization problem solving by hand is impractical. Although analytical description can be used on any complex artificial neural network in practise we use computers and specialised software that can help us build, mathematically describe and optimise any type of artificial neural network.

3.2 Recurrent Artificial Neural Networks

Artificial neural network with the recurrent topology is called *Recurrent* artificial neural network. It is similar to feed-forward neural network with no limitations regarding back-loops. In these cases information is no longer transmitted only in one direction but it is also transmitted backwards. This creates an internal state of the network which allows it to exhibit dynamic temporal behaviour. Recurrent artificial neural networks can use their internal memory to process any sequence of inputs. Fig. 6. shows small *Fully Recurrent* artificial neural network and complexity of its artificial neuron interconnections.

The most basic topology of recurrent artificial neural network is fully recurrent artificial network where every basic building block (artificial neuron) is directly connected to every other basic building block in all direction. Other recurrent artificial neural networks such as Hopfield, Elman, Jordan, bi-directional and other networks are just special cases of recurrent artificial neural networks.

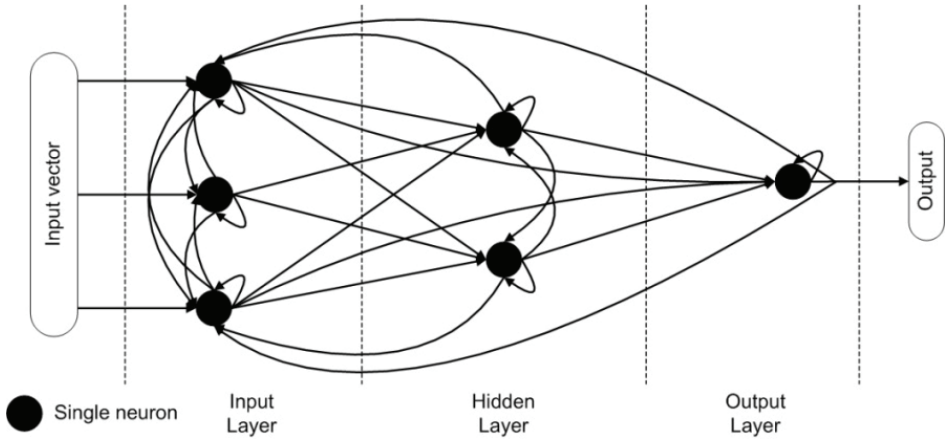

Fig. 6. Fully recurrent artificial neural network.

3.3 Hopfield Artificial Neural Network

A *Hopfield* artificial neural network is a type of recurrent artificial neural network that is used to store one or more stable target vectors. These stable vectors can be viewed as memories that the network recalls when provided with similar vectors that act as a cue to the network memory. These binary units only take two different values for their states that are determined by whether or not the units' input exceeds their threshold. Binary units can take either values of 1 or -1, or values of 1 or 0. Consequently there are two possible definitions for binary unit activation a_i (equation (6) and (7)):

$$a_i = \begin{cases} -1 & if \sum_j w_{ij} s_j > \theta_i, \\ 1 & otherwise. \end{cases} \quad (6)$$

$$a_i = \begin{cases} 0 & if \sum_j w_{ij} s_j > \theta_i, \\ 1 & otherwise. \end{cases} \quad (7)$$

Where:

- w_{ij} is the strength of the connection weight from unit j to unit i,
- s_j is the state of unit j,
- θ_i is the threshold of unit i.

While talking about connections w_{ij} we need to mention that there are typical two restrictions: no unit has a connection with itself (w_{ii}) and that connections are symmetric $w_{ij} = w_{ji}$.

The requirement that weights must be symmetric is typically used, as it will guarantee that the energy function decreases monotonically while following the activation rules. If non-symmetric weights are used the network may exhibit some periodic or chaotic behaviour. Training a Hopfield artificial neural network (Fig. 7.) involves lowering the energy of states that the artificial neural network should remember.

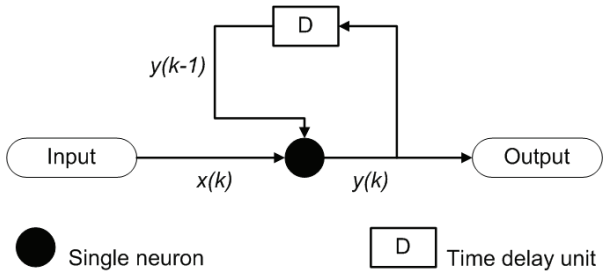

Fig. 7. Simple "one neuron" Hopfield artificial neural network.

3.4 Elman and Jordan Artificial Neural Networks

Elman network also referred as *Simple Recurrent Network* is special case of recurrent artificial neural networks. It differs from conventional two-layer networks in that the first layer has a recurrent connection. It is a simple three-layer artificial neural network that has back-loop from hidden layer to input layer trough so called context unit (Fig. 8.). This type of artificial neural network has memory that allowing it to both detect and generate time-varying patterns.

The Elman artificial neural network has typically sigmoid artificial neurons in its hidden layer, and linear artificial neurons in its output layer. This combination of artificial neurons transfer functions can approximate any function with arbitrary accuracy if only there is enough artificial neurons in hidden layer. Being able to store information Elman artificial neural network is capable of generating temporal patterns as well as spatial patterns and

responding on them. Jordan network (Fig. 9.) is similar to Elman network. The only difference is that context units are fed from the output layer instead of the hidden layer.

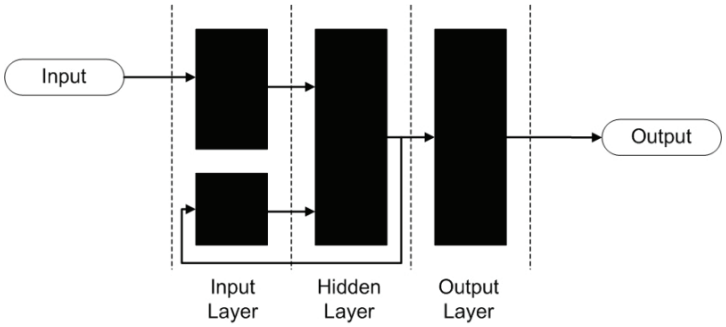

Fig. 8. Elman artificial neural network.

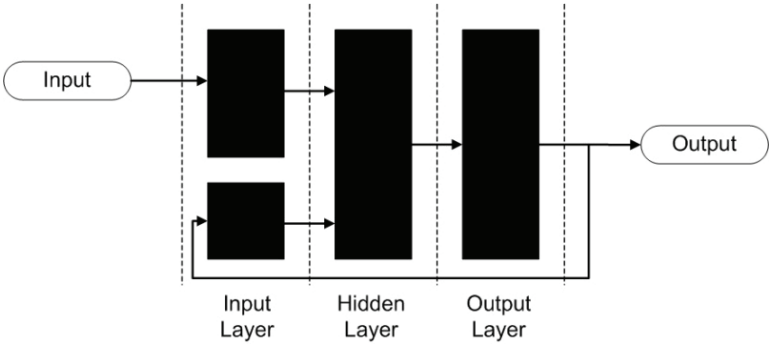

Fig. 9. Jordan artificial neural network.

3.5 Long Short Term Memory

Long Short Term Memory is one of the recurrent artificial neural networks topologies. In contrast with basic recurrent artificial neural networks it can learn from its experience to process, classify and predict time series with very long time lags of unknown size between important events. This makes Long Short Term Memory to outperform other recurrent artificial neural networks, Hidden Markov Models and other sequence learning methods.

Long Short Term Memory artificial neural network is build from Long Short Term Memory blocks that are capable of remembering value for any length of time. This is achieved with gates that determine when the input is significant enough remembering it, when continue to remembering or forgetting it, and when to output the value.

Architecture of Long Short Term Memory block is shown in Fig. 10 where input layer consists of sigmoid units. Top neuron in the input layer process input value that might be

sent to a memory unit depends on computed value of second neuron from the top in the input layer. The third neuron from the top in the input layer decide how long will memory unit hold (remember) its value and the bottom most neuron determines when value from memory should be released to the output. Neurons in first hidden layer and in output layer are doing simple multiplication of their inputs and a neuron in the second hidden layer computes simple linear function of its inputs. Output of the second hidden layer is fed back into input and first hidden layer in order to help making decisions.

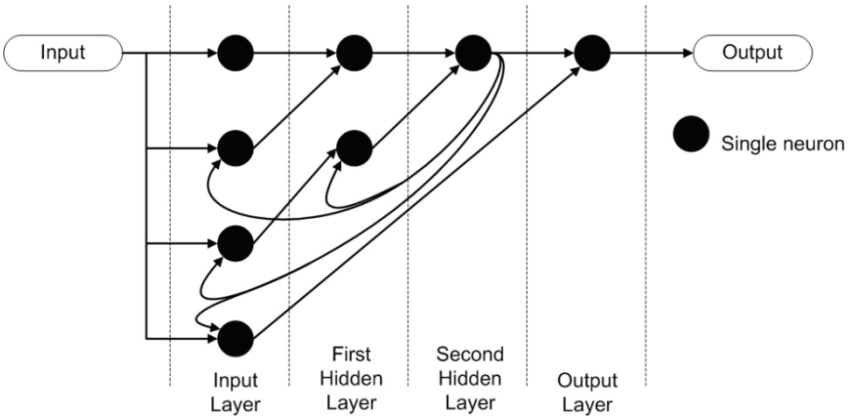

Fig. 10. Simple Long Short Term Memory artificial neural network (block).

3.6 Bi-directional Artificial Neural Networks (Bi-ANN)

Bi-directional artificial neural networks (Fig. 11.) are designed to predict complex time series. They consist of two individual interconnected artificial neural (sub) networks that performs direct and inverse (bidirectional) transformation. Interconnection of artificial neural sub networks is done through two dynamic artificial neurons that are capable of remembering their internal states. This type of interconnection between future and past values of the processed signals increase time series prediction capabilities. As such these artificial neural networks not only predict future values of input data but also past values. That brings need for two phase learning; in first phase we teach one artificial neural sub network for predicting future and in the second phase we teach a second artificial neural sub network for predicting past.

3.7 Self-Organizing Map (SOM)

Self-organizing map is an artificial neural network that is related to feed-forward networks but it needs to be told that this type of architecture is fundamentally different in arrangement of neurons and motivation. Common arrangement of neurons is in a hexagonal or rectangular grid (Fig. 12.). Self-organizing map is different in comparison to other artificial neural networks in the sense that they use a neighbourhood function to preserve the topological properties of the input space. They uses unsupervised learning paradigm to

produce a low-dimensional, discrete representation of the input space of the training samples, called a map what makes them especially useful for visualizing low-dimensional views of high-dimensional data. Such networks can learn to detect regularities and correlations in their input and adapt their future responses to that input accordingly.

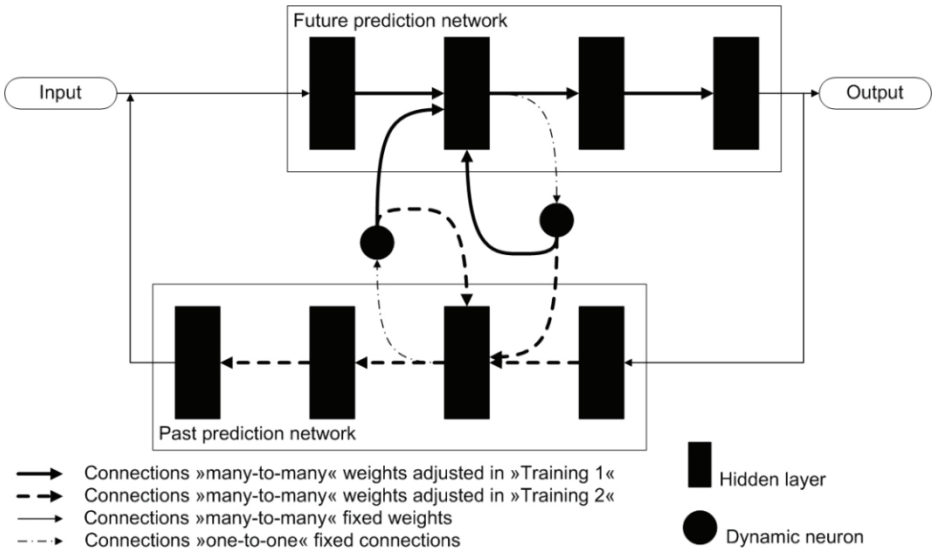

Fig. 11. Bi-directional artificial neural network.

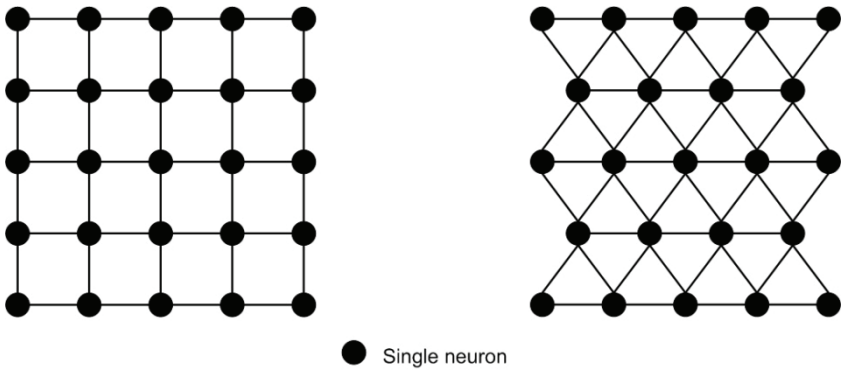

Fig. 12. Self-organizing Map in rectangular (left) and hexagonal (right) grid.

Just as others artificial neural networks need learning before they can be used the same goes for self-organizing map; where the goal of learning is to cause different parts of the artificial neural network to respond similarly to certain input patterns. While adjusting the weights of the neurons in the process of learning they are initialized either to small random values or sampled evenly from the subspace spanned by the two largest principal component eigenvectors. After initialization artificial neural network needs to be fed with large number of example vectors. At that time Euclidean distance to all weight vectors is computed and the neuron with weight vector most similar to the input is called the best matching unit. The weights of the best matching unit and neurons close to it are adjusted towards the input vector. This process is repeated for each input vector for a number of cycles. After learning phase we do so-called mapping (usage of artificial neural network) and during this phase the only one neuron whose weight vector lies closest to the input vector will be winning neuron. Distance between input and weight vector is again determined by calculating the Euclidean distance between them.

3.8 Stochastic Artificial Neural Network

Stochastic artificial neural networks are a type of an artificial intelligence tool. They are built by introducing random variations into the network, either by giving the network's neurons stochastic transfer functions, or by giving them stochastic weights. This makes them useful tools for optimization problems, since the random fluctuations help it escape from local minima. Stochastic neural networks that are built by using stochastic transfer functions are often called Boltzmann machine.

3.9 Physical Artificial Neural Network

Most of the artificial neural networks today are software-based but that does not exclude the possibility to create them with physical elements which base on adjustable electrical current resistance materials. History of physical artificial neural networks goes back in 1960's when first physical artificial neural networks were created with memory transistors called memistors. Memistors emulate synapses of artificial neurons. Although these artificial neural networks were commercialized they did not last for long due to their incapability for scalability. After this attempt several others followed such as attempt to create physical artificial neural network based on nanotechnology or phase change material.

4. Learning

There are three major learning paradigms; supervised learning, unsupervised learning and reinforcement learning. Usually they can be employed by any given type of artificial neural network architecture. Each learning paradigm has many training algorithms.

4.1 Supervised learning

Supervised learning is a machine learning technique that sets parameters of an artificial neural network from training data. The task of the learning artificial neural network is to set the value of its parameters for any valid input value after having seen output value. The training data consist of pairs of input and desired output values that are traditionally represented in data vectors. Supervised learning can also be referred as classification, where we have a wide range of classifiers, each with its strengths and weaknesses. Choosing a

suitable classifier (Multilayer perceptron, Support Vector Machines, k-nearest neighbour algorithm, Gaussian mixture model, Gaussian, naive Bayes, decision tree, radial basis function classifiers,…) for a given problem is however still more an art than a science.
In order to solve a given problem of supervised learning various steps has to be considered. In the first step we have to determine the type of training examples. In the second step we need to gather a training data set that satisfactory describe a given problem. In the third step we need to describe gathered training data set in form understandable to a chosen artificial neural network. In the fourth step we do the learning and after the learning we can test the performance of learned artificial neural network with the test (validation) data set. Test data set consist of data that has not been introduced to artificial neural network while learning.

4.2 Unsupervised learning

Unsupervised learning is a machine learning technique that sets parameters of an artificial neural network based on given data and a cost function which is to be minimized. Cost function can be any function and it is determined by the task formulation. Unsupervised learning is mostly used in applications that fall within the domain of estimation problems such as statistical modelling, compression, filtering, blind source separation and clustering.
In unsupervised learning we seek to determine how the data is organized. It differs from supervised learning and reinforcement learning in that the artificial neural network is given only unlabeled examples. One common form of unsupervised learning is clustering where we try to categorize data in different clusters by their similarity. Among above described artificial neural network models, the Self-organizing maps are the ones that the most commonly use unsupervised learning algorithms.

4.3 Reinforcement learning

Reinforcement learning is a machine learning technique that sets parameters of an artificial neural network, where data is usually not given, but generated by interactions with the environment. Reinforcement learning is concerned with how an artificial neural network ought to take actions in an environment so as to maximize some notion of long-term reward. Reinforcement learning is frequently used as a part of artificial neural network's overall learning algorithm.
After return function that needs to be maximized is defined, reinforcement learning uses several algorithms to find the policy which produces the maximum return. Naive brute force algorithm in first step calculates return function for each possible policy and chooses the policy with the largest return. Obvious weakness of this algorithm is in case of extremely large or even infinite number of possible policies. This weakness can be overcome by *value function approaches* or *direct policy estimation*. Value function approaches attempt to find a policy that maximizes the return by maintaining a set of estimates of expected returns for one policy; usually either the current or the optimal estimates. These methods converge to the correct estimates for a fixed policy and can also be used to find the optimal policy. Similar as value function approaches the direct policy estimation can also find the optimal policy. It can find it by searching it directly in policy space what greatly increases the computational cost.
Reinforcement learning is particularly suited to problems which include a long-term versus short-term reward trade-off. It has been applied successfully to various problems, including

robot control, telecommunications, and games such as chess and other sequential decision making tasks.

5. Usage of Artificial Neural Networks

One of the greatest advantages of artificial neural networks is their capability to learn from their environment. Learning from the environment comes useful in applications where complexity of the environment (data or task) make implementations of other type of solutions impractical. As such artificial neural networks can be used for variety of tasks like classification, function approximation, data processing, filtering, clustering, compression, robotics, regulations, decision making, etc. Choosing the right artificial neural network topology depends on the type of the application and data representation of a given problem. When choosing and using artificial neural networks we need to be familiar with theory of artificial neural network models and learning algorithms. Complexity of the chosen model is crucial; using to simple model for specific task usually results in poor or wrong results and over complex model for a specific task can lead to problems in the process of learning. Complex model and simple task results in memorizing and not learning. There are many learning algorithms with numerous tradeoffs between them and almost all are suitable for any type of artificial neural network model. Choosing the right learning algorithm for a given task takes a lot of experiences and experimentation on given problem and data set. When artificial neural network model and learning algorithm is properly selected we get robust tool for solving given problem.

5.1 Example: Using bi-directional artificial neural network for ICT fraud detection

Spread of Information and Communication Technologies results in not only benefits for individuals and society but also in threats and increase of Information and Communication Technology frauds. One of the main tasks for Information and Communication Technology developers is to prevent potential fraudulent misuse of new products and services. If protection against fraud fails there is a vital need to detect frauds as soon as possible. Information and Communication Technology frauds detection is based on numerous principles. One of such principle is use of artificial neural networks in the detection algorithms. Below is an example of how to use bi-directional artificial neural network for detecting mobile-phone fraud.

First task is to represent problem of detecting our fraud in the way that can be easily understand by humans and machines (computers). Each individual user or group of users behave in specific way while using mobile phone. By learning their behaviour we can teach our system to recognize and predict users' future behaviour to a certain degree of accuracy. Later comparison between predicted and real-life behaviour and potential discrepancy between them can indicate a potential fraudulent behaviour. It was shown that mobile-phone usage behaviour can be represented in the form of time series suitable for further analysis with artificial neural networks (Krenker et al., 2009). With this representation we transform the behaviour prediction task in time series prediction task. Time series prediction task can be realized with several different types of artificial neural networks but as mentioned in earlier chapters some are more suitable then others. Because we expect long and short time periods between important events in our data representation of users' behaviour the most obvious artificial neural networks to use are Long Short Term Memory and bi-directional

artificial neural networks. On the basis of others researchers' favourable results in time series prediction with bi-directional artificial neural network (Wakuya & Shida, 2001) we decided to use this artificial neural network topology for predicting our time series.

After we choose artificial neural network architecture we choose the type of learning paradigm; we choose supervised learning where we gather real life data form telecommunication system. Gathered data was divided into two sub-sets; training sub-set and validation subset. With training data sub-set artificial neural network learn to predict future and past time series and with validation data sub-set we simulate and validate the prediction capabilities of designed and fine-tuned bi-directional artificial neural networks. Validation was done with calculation of the *Average Relative Variance* that represents a measure of similarity between predicted and expected time series.

Only after we gathered information about mobile-phone fraud and after choosing representation of our problem and basic approaches for solving it we could start building the overall model for detecting mobile-phone fraud (Fig. 13.).

On Fig. 13. we can see that mobile-phone fraud detection model is build out of three modules; input module, artificial neural network module and comparison module. *Input Module* gathers users' information about usage of mobile-phone from telecommunication system in three parts. In first part it is used for gathering learning data from which *Artificial Neural Network Module* learn it-self. In second part I*nput Module* gathers users' data for purpose of validating the *Artificial Neural Network Module* and in the third part it collects users' data in real time for purpose of using deployed mobile-phone fraud system. *Artificial Neural Network Module* is bi-directional artificial neural network that is learning from gathered data and later when the mobile-phone fraud detection system is deployed continuously predicts time series that represents users' behaviour. *Comparison module* is used for validation of *Artificial Neural Network Module* in the process of learning and later when the mobile-phone fraud detection system is deployed it is used for triggering alarms in case of discrepancies between predicted and real-life gathered information about users' behaviour.

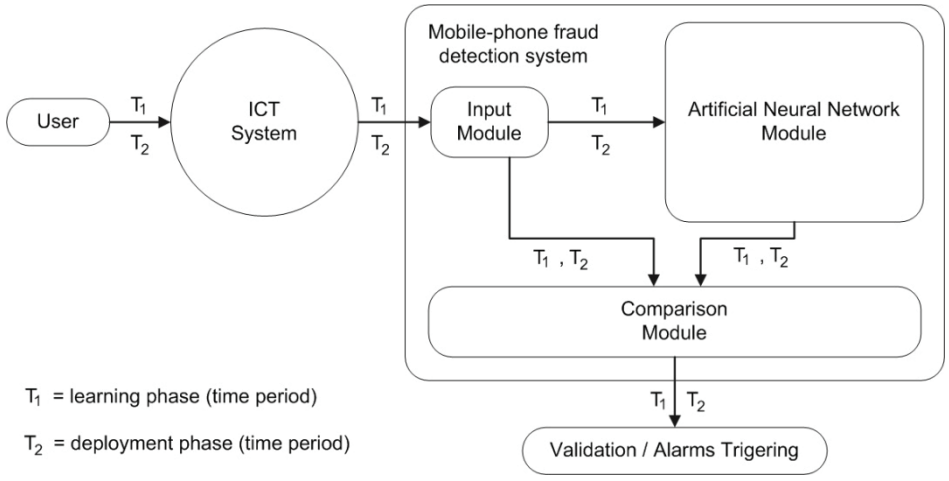

Fig. 13. Mobile-phone fraud detection model.

Although mobile-phone fraud detection system described above is simple and straight forward reader needs to realize that majority of work is not in creating and later implementing desired systems but in fine-tuning of data representation and artificial neural network architecture and its parameters that is strongly dependant on type of input data.

6. Conclusions

Artificial neural networks are widely spread and used in everyday services, products and applications. Although modern software products enable relatively easy handling with artificial neural networks, their creation, optimisation and usage in real-life situations it is necessary to understand theory that stands behind them. This chapter of the book introduces artificial neural networks to novice reader and serves as a stepping stone for all of those who would like to get more involved in the area of artificial neural networks.
In the *Introduction* in order to lighten the area of artificial neural networks we briefly described basic building blocks (artificial neuron) of artificial neural networks and their "transformation" from single artificial neuron to complete artificial neural network. In the chapter *Artificial Neuron* we present basic and important information about artificial neuron and where researchers borrowed the idea to create one. We show the similarities between biological and artificial neuron their composition and inner workings. In the chapter *Artificial Neural Networks* we describe basic information about different, most commonly used artificial neural networks topologies. We described *Feed-forward*, *Recurrent*, *Hopfield*, *Elman*, *Jordan*, *Long Short Term Memory*, *Bi-directional*, *Self Organizing Maps*, *Stochastic* and *Physical* artificial neural networks. After describing various types of artificial neural networks architectures we describe how to make them useful by learning. We describe different learning paradigms (supervised, unsupervised and reinforcement learning) in chapter *Learning*. In the last chapter *Usage of Artificial Neural Networks* we describe how to handle artificial neural networks in order to make them capable of solving certain problems. In order to show what artificial neural networks are capable of, we gave a short example how to use bi-directional artificial neural network in mobile-phone fraud detection system.

7. References

Gurney, K. (1997). *An Introduction to Neural Networks*, Routledge, ISBN 1-85728-673-1 London

Krenker A.; Volk M.; Sedlar U.; Bešter J.; Kos A. (2009). Bidirectional artificial neural networks for mobile-phone fraud detection. *ETRI Jurnal*., vol. 31, no. 1, Feb. 2009, pp. 92-94, COBISS.SI-ID 6951764

Kröse B.; Smagt P. (1996). *An Introduction to Neural Networks*, The University of Amsterdam, Amsterdam.

Pavešić N. (2000). *Razpoznavanje vzorcev: uvod v analizo in razumevanje vidnih in slušnih signalov,* Fakulteta za elektrotehniko, ISBN 961-6210-81-5, Ljubljana

Rojas R. (1996). *Neural Networks: A Systematic Introduction*, Springer, ISBN 3-540-60505-3, Germany.

Wakuya H.; Shida K.. (2001). Bi-directionalization of neural computing architecture for time series prediction. III. Application to laser intensity time record "Data Set A". *Proceedings of International Joint Conference on Neural Networks,* pp. 2098 – 2103, ISBN 0-7803-7044-9, Washington DC, 2001, Washington DC.