SQU Journal For Science, 12 (2007)121-141


# Application of Neural Networks to Data Mining

· Zlatinka Svetoslavova Kovacheva

*Higher College of Technology, Al Khuwair, Muscat, Oman,E-mail: zkovacheva@hotmail.com*

تطبيقات الشبكات العصبية في التنقيب علي البيانات

ذلاتينكا سفيتوسلافوفا كوفاتشيفا

خلاصة : التنقيب في البيانات هو علم مهم لاستخلاص المعلومات من البيانات والأرقام التي في الأغلب لا تكون مفهومة لمعظم الناس بحيث يتم استخلاص المعلومات وتقديمها بلغة سهلة ومفهومة,كما يقوم تنقيب البيانات باستخدام البيانات الحالية المتوفرة لإستنباط بيانات جديدة تستخدم لعمل التوقعات حول حقل معين. الشبكات العصبية أيضا تستخدم كأداة تحليل تكون مصممة لبناء النماذج التفسيرية وفي الوقت الحالي توجد شبكات عصبية تقوم باستخدام عمليات حسابية معقدة بشكل مباشر لتساهم في عملية بناء النماذج التفسيرية حيث أن اخر تطورات البحث في الشبكات العصبية تجعلها قريبة جدا من تعريف التنقيب في البيانات فهي أيضا تقوم باستخلاص المعلومات من البيانات الغامضة لتقدمها بمصطحات واضحة ومفهومة للجميع. الهدف الأساسي من هذه المراجعة هو مقارنة الشبكات العصبية بغيرها من أساليب تنقيب البيانات ولعرض بعض الأمثلة والتطبيقات لدور الشبكات العصبية في تنقيب البيانات على الواقع.

**ABSTRACT:** Data Mining accomplishes nontrivial extraction of implicit, previously unknown, and potentially useful information of data. The aim of data mining is to discover knowledge out of data and present it in a form that is easily comprehensible to humans. Neural Networks are analytic techniques capable of predicting new observations from other observations after executing a process of so-called learning from existing data. Neural Network techniques can also be used as a component of analyses designed to build explanatory models. Now there is neural network software that uses sophisticated algorithms directly contributing to the model building process. The latest developments in research on neural networks bring them much closer to the ideal of data mining: knowledge out of data in understandable terms. The main goal of the review is to compare neural networks with other techniques for data mining and to overview some examples of application of neural networks to data mining processes in practice.

KEYWORDS: Data mining, neural network, artificial neuron, perceptron, hopfield model, neural network software and application.

## 1. Introduction

### 1.1 Data mining

As a result of the rapid development of information technologies in the recent decades, the amount of amassed information increases at such a high rate that the most important emphasis in data processing is

laid on the methods for data extraction, analysis and understanding. Data Mining is an analytic process designed to explore data (usually large amounts of data - typically business or market related) in search of consistent patterns and/or systematic relationships between variables, and then to validate the findings by applying the detected patterns to new subsets of data. The ultimate goal of data mining is prediction - and predictive data mining is the most common type of data mining and one that has the most direct business applications. (Witten, and Frank 2000).

The term Predictive Data Mining is usually applied to identify data mining projects with the goal to design a statistical or neural network model or set of models that can be used to predict some response of interest. For example, a credit card company may want to engage in predictive data mining, to derive a model or set of models (e.g., neural networks) that can quickly identify transactions which have a high probability of being fraudulent. Other types of data mining projects (e.g., to identify cluster or segments of customers) use drill-down descriptive and exploratory methods. For more details see (Weiss *et al.*, 1997).

In the business environment, complex data mining projects may require the coordinate efforts of various experts, stakeholders, or departments throughout an entire organization. In the data mining literature, various "general frameworks" have been proposed to organize the process of gathering data, analyzing data, disseminating results, implementing results, and monitoring improvements.

In general, the process of data mining consists of three stages:

- the initial exploration,
- model building or pattern identification with validation/verification, and
- deployment (i.e., the application of the model to new data in order to generate predictions).

*Stage* 1: *exploration*

This stage usually starts with data preparation which may involve cleaning data, data transformations, selecting subsets of records and - in case of data sets with large numbers of variables ("fields") - performing some preliminary feature selection operations to bring the number of variables to a manageable range (depending on the statistical methods which are being considered).

Data preparation and cleaning is an often neglected but extremely important step in the data mining process. It is particularly applicable to the typical data mining projects where large data sets collected via some automatic methods (e.g., via the Web) serve as the input into the analyses. Often, the method by which the data was gathered was not tightly controlled, and so the data may contain out-of-range values (e.g., Income: -100), impossible data combinations (e.g., Gender: Male, Pregnant: Yes), and the like. Analyzing data that has not been carefully screened for such problems can produce highly misleading results, in particular in predictive data mining.

Then, depending on the nature of the analytic problem, this first stage of the process of data mining may involve anywhere between a simple choice of straightforward predictors for a regression model, to elaborate exploratory analyses using a wide variety of graphical and statistical methods.

Data reduction in the context of data mining is usually applied to projects where the goal is to aggregate or amalgamate the information contained in large datasets into manageable (smaller) information nuggets. Data reduction methods can include simple tabulation, aggregation (computing descriptive statistics) or more sophisticated techniques like clustering, principal components analysis, etc.

The concept of drill-down analysis applies to the area of data mining, to denote the interactive exploration of data, in particular of large databases. The process of drill-down analyses begins by considering some simple break-downs of the data by a few variables of interest (e.g., gender, geographic region, etc.). Various statistics, tables, histograms, and other graphical summaries can be computed for each group. Next one may want to "drill-down" to expose and further analyze the data "underneath" one of the categorizations, for example, one might want to further review the data for males from one region. Again, various statistical and graphical summaries can be computed for those cases only, which might suggest further break-downs by other variables (e.g., income, age, etc.). At the lowest ("bottom") level are the raw data: For example, you may want to review the addresses of

male customers from one region, for a certain income group, etc., and to offer to those customers some particular services of particular utility to that group.

*Stage* 2: *model building and validation*

This stage involves considering various models and choosing the best one based on their predictive performance (i.e., explaining the variability in question and producing stable results across samples). This may sound like a simple operation, but in fact, it sometimes involves a very elaborate process. There are a variety of techniques developed to achieve that goal - many of which are based on so-called "competitive evaluation of models", that is, applying different models to the same data set and then comparing their performance to choose the best. These techniques, which are often considered the core of predictive data mining, include: Bagging (Voting, Averaging), Boosting, Stacking (Stacked Generalizations), and Meta-Learning.

The concept of bagging (voting for classification, averaging for regression-type problems with continuous dependent variables of interest) applies to the area of predictive data mining, to combine the predicted classifications (prediction) from multiple models, or from the same type of model for different learning data. It is also used to address the inherent instability of results when applying complex models to relatively small data sets. Suppose your data mining task is to build a model for predictive classification, and the dataset from which to train the model (learning data set, which contains observed classifications) is relatively small. You could repeatedly sub-sample (with replacement) from the dataset, and apply, for example, a tree classifier to the successive samples. In practice, very different trees will often be grown for the different samples, illustrating the instability of models often evident with small datasets. One method of deriving a single prediction (for new observations) is to use all trees found in the different samples, and to apply some simple voting: The final classification is the one most often predicted by the different trees. Note that some weighted combination of predictions (weighted vote, weighted average) is also possible, and commonly used. A sophisticated (machine learning) algorithm for generating weights for weighted prediction or voting is the Boosting procedure.

The concept of boosting applies to the area of predictive data mining, to generate multiple models or classifiers (for prediction or classification), and to derive weights to combine the predictions from those models into a single prediction or predicted classification.

A simple algorithm for boosting works like this: Assign greater weight to those observations that were difficult to classify (where the misclassification rate was high), and lower weights to those that were easy to classify (where the misclassification rate was low). Then apply the classifier again to the weighted data and continue with the next iteration (application of the analysis method for classification to the re-weighted data).

Boosting will generate a sequence of classifiers, where each consecutive classifier in the sequence is an "expert" in classifying observations that were not well classified by those preceding it. During deployment (for prediction or classification of new cases), the predictions from the different classifiers can then be combined (e.g., via voting, or some weighted voting procedure) to derive a single best prediction or classification.

Boosting can also be applied to learning methods that do not explicitly support weights or misclassification costs. In that case, random sub-sampling can be applied to the learning data in the successive steps of the iterative boosting procedure, where the probability for selection of an observation into the subsample is inversely proportional to the accuracy of the prediction for that observation in the previous iteration (in the sequence of iterations of the boosting procedure).

*Stage* 3: *deployment*

That final stage involves using the model selected as best in the previous stage and applying it to new data in order to generate predictions or estimates of the expected outcome.

The concept of deployment in predictive data mining refers to the application of a model for prediction or classification to new data. After a satisfactory model or set of models has been identified (trained) for a particular application, one usually wants to deploy those models so that predictions or predicted classifications can quickly be obtained for new data. For example, a credit card company may want to deploy a trained model

or set of models (e.g., neural networks) to quickly identify transactions which have a high probability of being fraudulent.

The concept of Data Mining is becoming increasingly popular as a business information management tool where it is expected to reveal knowledge structures that can guide decisions in conditions of limited certainty. Recently, there has been increased interest in developing new analytic techniques specifically designed to address the issues relevant to business *Data Mining* (e.g., *Classification Trees*), but Data Mining is still based on the conceptual principles of statistics including the traditional *Exploratory Data Analysis* and modelling and it shares with them both some components of its general approaches and specific techniques. (Pregibon 1997 and Hastie *et al.* 2001).

However, an important general difference in the focus and purpose between *Data Mining* and the traditional *Exploratory Data Analysis* (*EDA*) is that *Data Mining* is more oriented towards applications than the basic nature of the underlying phenomena. In other words, *Data Mining* is relatively less concerned with identifying the specific relations between the involved variables. For example, uncovering the nature of the underlying functions or the specific types of interactive, multivariate dependencies between variables are not the main goal of *Data Mining*. Instead, the focus is on producing a solution that can generate useful predictions. Therefore, *Data Mining* accepts among others a "black box" approach to data exploration or knowledge discovery and uses not only the traditional *Exploratory Data Analysis (EDA)* techniques, but also such techniques as *Neural Networks* which can generate valid predictions but are not capable of identifying the specific nature of the interrelations

Due to its applied importance, Data Mining emerges as a rapidly growing and major area (also in statistics) where important theoretical advances are being made. For more details see (Edelstein, 1999, Berry and Linoff, 2000, Han and Kamber 2000).

## 2. Neural networks

Neural networks is one of the Data Mining techniques.They are analytic techniques modelled after the (hypothesized) processes of learning in the cognitive system and the neurological functions of the brain and capable of predicting new observations (on specific variables) from other observations (on the same or other variables) after executing a process of so-called learning from existing data. As the prediction is usually an activity of the human brain, to automate this process, it is necessary to understand "How the human brain learns?"

### 2.1 Biological neurons

The brain is a collection of about 10 billion interconnected neurons. Each neuron is a cell that uses biochemical reactions to receive, process and transmit information. A neuron's dendritic tree is connected to a thousand neighbouring neurons. When one of those neurons fire, a positive or negative charge is received by one of the dendrites. The strengths of all the received charges are added together through the processes of spatial and temporal summation. Spatial summation occurs when several weak signals are converted into a single large one, while temporal summation converts a rapid series of weak pulses from one source into one large signal. The aggregate input is then passed to the soma (cell body). The soma and the enclosed nucleus don't play a significant role in the processing of incoming and outgoing data. Their primary function is to perform the continuous maintenance required to keep the neuron functional. The part of the soma that does concern itself with the signal is the axon hillock. If the aggregate input is greater than the axon hillock's threshold value, then the neuron *fires*, and an output signal is transmitted down the axon. The strength of the output is constant, regardless of whether the input was just above the threshold, or a hundred times as great. The output strength is unaffected by the many divisions in the axon; it reaches each terminal button with the same intensity it had at the axon hillock. This uniformity is critical in an analogue device such as a brain where small errors can snowball, and where error correction is more difficult than in a digital system. (See Figure 1).

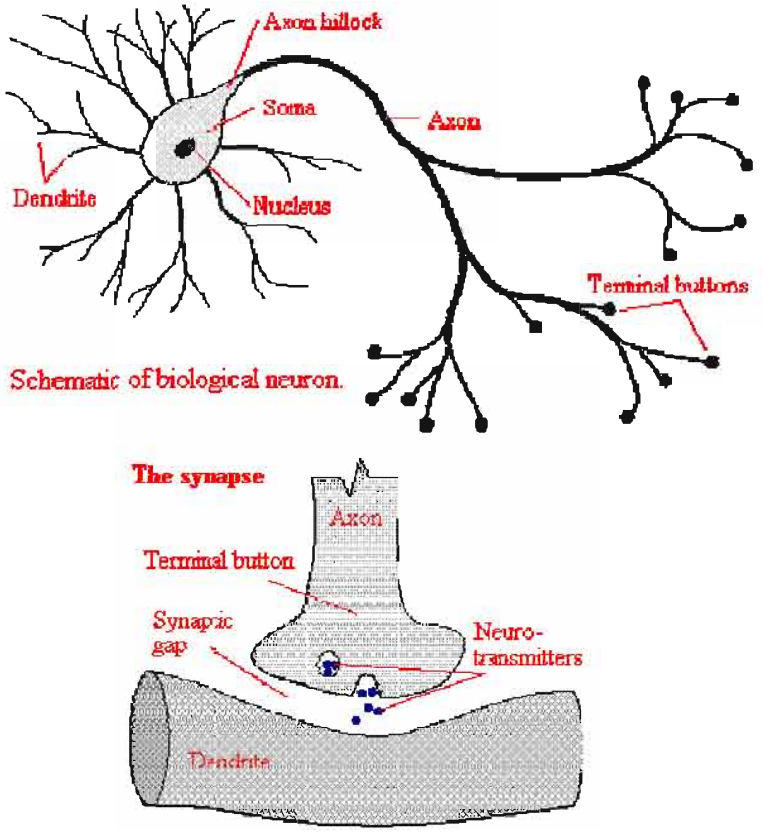


Figure1. Biological neuron

Each terminal button is connected to other neurons across a small gap called a synapse. The physical and neurochemical characteristics of each synapse determines the strength and polarity of the new input signal. This is where the brain is the most flexible, and the most vulnerable. Changing the constitution of various neuro-transmitter chemicals can increase or decrease the amount of stimulation that the firing axon imparts on the neighbouring dendrite. Altering the neurotransmitters can also change whether the stimulation is excitatory or inhibitory. Many drugs such as alcohol and LSD have dramatic effects on the production or destruction of these critical chemicals. The infamous nerve gas sarin can kill because it neutralizes a chemical (acetylcholinesterase) that is normally responsible for the destruction of a neurotransmitter (acetylcholine). This means that once a neuron fires, it keeps on triggering all the neurons in the vicinity. One no longer has control over muscles, and suffocation ensues. For more details see (Abdi, 1994, 1999, Anderson, 1995, Reilly *et al.*, 1982).

Learning occurs by changing the effectiveness of the synapses so that the influence of one neuron on another changes.
Biological neurons are believed to be the structural constituents of the brain. A neural network can:

- learn by adapting its synaptic weights to changes in the surrounding environments;
- handle imprecise, fuzzy, noisy, and probabilistic information; and
- generalize from known tasks or examples to unknown ones.

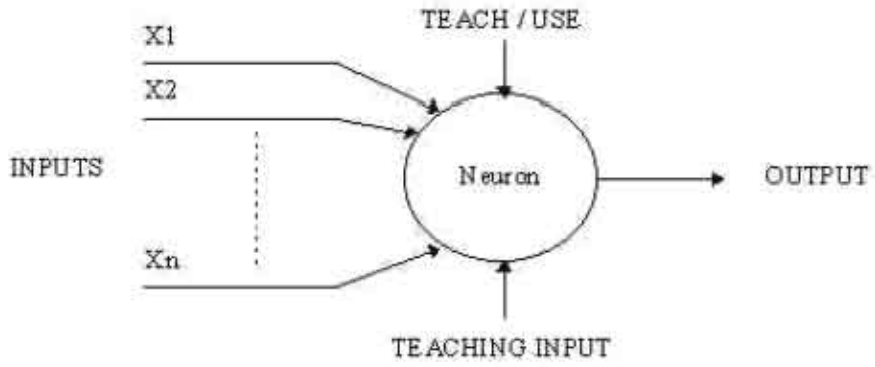


**Figure 2. A simple artificial neuron**

In more practical terms neural networks are non-linear statistical data modeling tools. They can be used to model complex relationships between inputs and outputs or to find patterns in data.

A neural network can learn. One of the neural net paradigms is supervised learning. In supervised learning, a neural net directly compares the network output with a known correct or desired answer in the training process. A neural network is characterized by:

- network topology,
- connection strength between neurons,
- node properties,
- internal controls, and
- the updating rules.

### 2.2 Artificial neural networks

Artificial neural networks (ANN) are an attempt to mimic some, or all, of these characteristics. An artificial neural network (ANN) is either software or hardware that can simulate biological neural networks. Artificial neural networks can perform computations and have learning abilities. These features distinguish neural network software from other software. See (Patterson, D.,1996; Carling, A.,1992).

An artificial neural network (ANN), also called a simulated neural network (SNN) or commonly just neural network (NN) is an interconnected group of artificial neurons that uses a mathematical or computational model for information processing based on a connectionist approach to computation. In most cases an ANN is an adaptive system that changes its structure based on external or internal information that flows through the network.

#### 2.2.1 A simple model of an artificial neuron

As complicated as the biological neuron is, it may be simulated by a very simple model.

An artificial neuron (see Figure 2) is a device with many inputs and one output. The neuron has two modes of operation; the training mode and the using mode. In the training mode, the neuron can be trained to fire (or not), for particular input patterns. In the using mode, when a taught input pattern is detected at the input, its associated output becomes the current output. If the input pattern does not belong in the taught list of input patterns, the firing rule is used to determine whether to fire or not.

The inputs each have a weight that they contribute to the neuron, if the input is active. The neuron can have any number of inputs; neurons in the brain can have as many as a thousand inputs.

The firing rule is an important concept in neural networks and accounts for their high flexibility. A firing rule determines how one calculates whether a neuron should fire for any input pattern. It relates to all the input patterns, not only the ones on which the node was trained.

A simple firing rule can be implemented by using Hamming distance technique. The rule goes as follows: Take a collection of training patterns for a node, some of which cause it to fire (the 1-taught set of patterns) and others which prevent it from doing so (the 0-taught set). Then the patterns not in the collection cause the node to fire if, on comparison, they have more input elements in common with the 'nearest' pattern in the 1-taught set than with the 'nearest' pattern in the 0-taught set. If there is a tie, then the pattern remains in the undefined state.

For example, a 3-input neuron is taught to output 1 when the input (X1,X2 and X3) is 111 or 101 and to output 0 when the input is 000 or 001. Then, before applying the firing rule, the truth Table is;

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| X1: | | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| X2: | | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| X3: | | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| OUT: | | 0 | 0 | 0/1 | 0/1 | 0/1 | 1 | 0/1 | 1 |

As an example of the way the firing rule is applied, take the pattern 010. It differs from 000 in 1 element, from 001 in 2 elements, from 101 in 3 elements and from 111 in 2 elements. Therefore, the 'nearest' pattern is 000 which belongs in the 0-taught set. Thus the firing rule requires that the neuron should not fire when the input is 001. On the other hand, 011 is equally distant from two taught patterns that have different outputs and thus the output stays undefined (0/1).

By applying the firing in every column the following truth Table is obtained;

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| X1: | | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| X2: | | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| X3: | | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| OUT: | | 0 | 0 | 0 | 0/1 | 0/1 | 1 | 1 | 1 |

The difference between the two truth tables is called the *generalisation of the neuron.* Therefore the firing rule gives the neuron a sense of similarity and enables it to respond 'sensibly' to patterns not seen during training.

### 2.2.2 A more complicated neuron

The previous neuron doesn't do anything that conventional computers don't do already. A more sophisticated neuron is the McCulloch and Pitts model (MCP). The difference from the previous model is that the inputs are 'weighted', the effect that each input has at decision making is dependent on the weight of the particular input. The weight of an input is a number which when multiplied with the input gives the weighted input. These weighted inputs are then added together and if they exceed a pre-set threshold value, the neuron fires. In any other case the neuron does not fire. (See Figure 3).

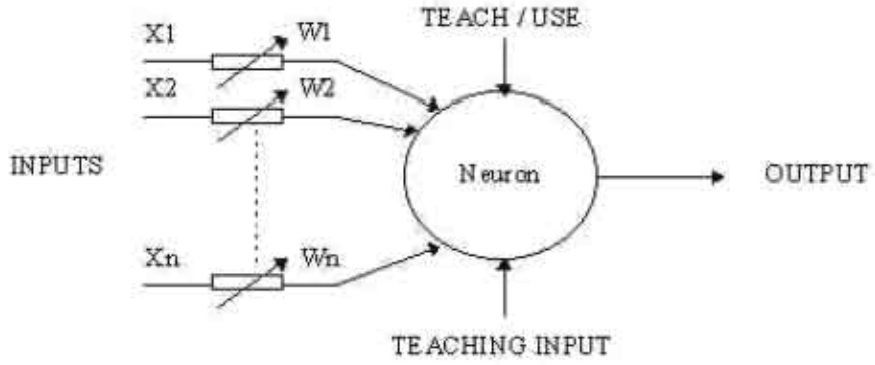


**Figure 3. An MCP neuron**

In mathematical terms, the neuron fires if and only if;

$$X1W1 + X2W2 + X3W3 + ... + XnWn > T$$

The addition of input weights and of the threshold makes this neuron a very flexible and powerful one. The MCP neuron has the ability to adapt to a particular situation by changing its weights and/or threshold. Various algorithms exist that cause the neuron to 'adapt'; the most used ones are the Delta rule and the back error propagation. The former is used in feed-forward networks and the latter in feedback networks.

### 2.3 Architecture of neural networks

#### 2.3.1 Feed-forward networks

Feed-forward ANNs allow signals to travel one way only; from input to output. There is no feedback (loops) i.e. the output of any layer does not affect that same layer. Feed-forward ANNs tend to be straight forward networks that associate inputs with outputs. They are extensively used in pattern recognition. This type of organisation is also referred to as bottom-up or top-down. (See Figure 4).

#### 2.31.2 Feedback networks

Feedback networks can have signals travelling in both directions by introducing loops in the network. Feedback networks are very powerful and can get extremely complicated. Feedback networks are dynamic; their 'state' is changing continuously until they reach an equilibrium point. They remain at the equilibrium point until the input changes and a new equilibrium needs to be found. Feedback architectures are also referred to as interactive or recurrent, although the latter term is often used to denote feedback connections in single-layer organisations.

The commonest type of artificial neural network consists of three groups, or layers, of units: a layer of "input" units is connected to a layer of "hidden" units, which is connected to a layer of "output" units. The activity of the input units represents the raw information that is fed into the network.

The activity of each hidden unit is determined by the activities of the input units and the weights on the connections between the input and the hidden units.

The behaviour of the output units depends on the activity of the hidden units and the weights between the hidden and output units.

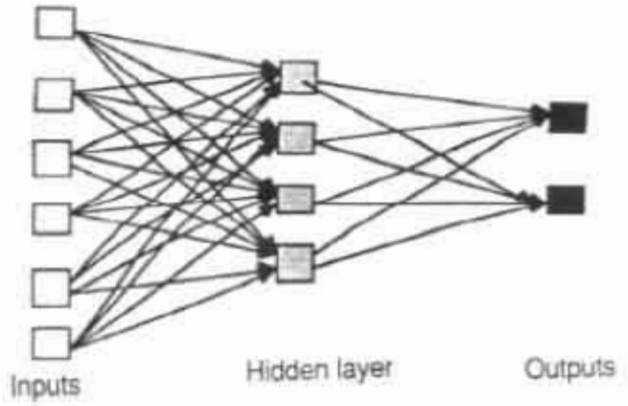


**Figure 4. An example of a simple feedforward network**

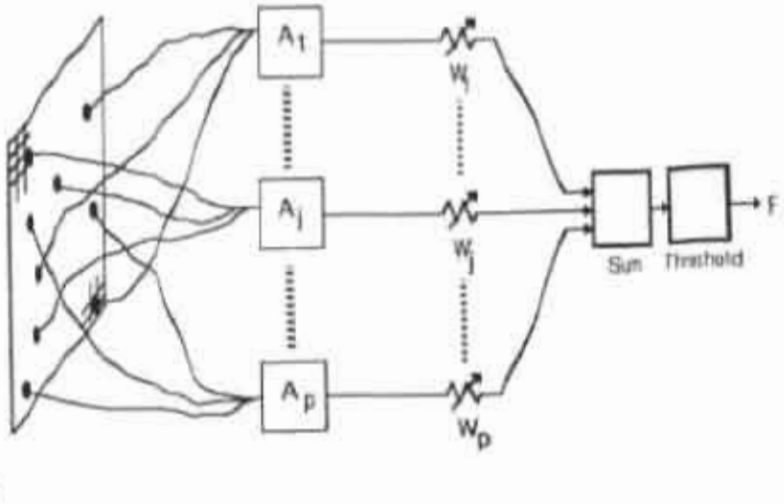


**Figure 5. Perceptron**

This simple type of network is interesting because the hidden units are free to construct their own representations of the input. The weights between the input and hidden units determine when each hidden unit is active, and so by modifying these weights, a hidden unit can choose what it represents. (Mandic, 2001).

We also distinguish single-layer and multi-layer architectures. The single-layer organisation, in which all units are connected to one another, constitutes the most general case and is of more potential computational power than hierarchically structured multi-layer organisations. In multi-layer networks, units are often numbered by layer, instead of following a global numbering. (Fukushima, 1975).

### 2.4 Perceptrons

The most influential work on neural nets in the 60's went under the heading of 'perceptrons' a term coined by Frank Rosenblatt. The perceptron turns out to be an MCP model ( neuron with weighted inputs ) with some additional, fixed, pre-processing. Units labelled A1, A2, Aj , Ap are called association units and their task is to extract specific, localised featured from the input images. Perceptrons mimic the basic idea behind the

mammalian visual system. They were mainly used in pattern recognition even though their capabilities extended a lot more. See Figure 5 (Minsky and Papert, 1969).

### 2.5 Hopfield model and boltzmann machine

One of the most popular networks is the Hopfield network. This recurrent net is completely connected. The Boltzmann Machine is closely related to the Hopfield model. The Boltzmann Machine is a special type of neural network, in which each neuron configuration has a certain probability to appear. (The name comes from the following fact: the Boltzmann Machine is a probabilistic neural network which forms a Markov chain; the invariant distribution of the Markov chain is similar to the "Boltzmann Distribution" in statistical physics).

Most software is "programmed" to perform certain tasks. These tasks are fixed. For example, chess game software will not play solitary. Neural network software is not programmed; it is trained. Neural networks do what they are trained to do. What you train is what you get.

The neural network attempts to simulate the human brain. The human brain has front-end subsystems and can only handle preprocessed information. The front-end subsystems for humans are eyes, ears, ... The same is also true for neural networks.

There are two types of data used by neural network systems: user data (or application data), and neural data. Neural networks use neural data. User data depends on the applications. Put it in another way: neural nets speak neural language, users speak user language.

The information processed by a neural network has to be prepared by a front-end subsystem. This is called data encoding. A neural network can not usually handle the user-application data directly. Similarly, after neural computation, the result usually does not make sense to humans directly; the front-end system is responsible for converting the neural output data back into user-application data. This is called data decoding. Front-end systems are basically a language translator. A neural network system consists of the front-end subsystem and a neural network subsystem:

### 2.6 Training neural networks

Neural networks can be explicitly programmed to perform a task by manually creating the topology and then setting the weights of each link and threshold. However, this by-passes one of the unique strengths of neural nets: the ability to program themselves. The most basic method of training a neural network is trial and error. If the network isn't behaving the way it should, change the weighting of a random link by a random amount. If the accuracy of the network declines, undo the change and make a different one. It takes time, but the trial and error method does produce results. The neural network to the left is a simple one that could be constructed using such a trial and error method. The task is to mirror the status of the input row onto the output row. To do this it would have to invent a binary to decimal encoding and decoding scheme with which it could pass the information through the bottle-neck of the two neurons in the centre.

Unfortunately, the number of possible weightings rises exponentially as one adds new neurons, making large general-purpose neural nets impossible to construct using trial and error methods. In the early 1980s two researchers, David Rumelhart and David Parker, independently rediscovered an old calculus-based learning algorithm. The back-propagation algorithm compares the result that was obtained with the result that was expected. It then uses this information to systematically modify the weights throughout the neural network. This training takes only a fraction of the time that trial and error method take. It can also be reliably used to train networks on only a portion of the data, since it makes inferences. The resulting networks are often correctly configured to answer problems that they have never been specifically trained on.

As useful as back-propagation is, there are often easier ways to train a network. For specific-purpose networks, such as pattern recognition, the operation of the network and the training method are relatively easy to observe even though the networks might have hundreds of neurons.

Also referred to as connectionist architectures, parallel distributed processing, and neuromorphic systems, an artificial neural network (ANN) is an information-processing paradigm inspired by the way the densely interconnected, parallel structure of the mammalian brain processes information. Artificial neural networks are

collections of mathematical models that emulate some of the observed properties of biological nervous systems and draw on the analogies of adaptive biological learning. The key element of the ANN paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements that are analogous to neurons and are tied together with weighted connections that are analogous to synapses.

Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons. This is true of ANNs as well. Learning typically occurs by example through training, or exposure to a truthed set of input/output data where the training algorithm iteratively adjusts the connection weights (synapses). These connection weights store the knowledge necessary to solve specific problems.

Although ANNs have been around since the late 1950's, it wasn't until the mid-1980's that algorithms became sophisticated enough for general applications. Today ANNs are being applied to an increasing number of real- world problems of considerable complexity. They are good pattern recognition engines and robust classifiers, with the ability to generalize in making decisions about imprecise input data. They offer ideal solutions to a variety of classification problems such as speech, character and signal recognition, as well as functional prediction and system modeling where the physical processes are not understood or are highly complex. ANNs may also be applied to control problems, where the input variables are measurements used to drive an output actuator, and the network learns the control function. The advantage of ANNs lies in their resilience against distortions in the input data and their capability of learning. They are often good at solving problems that are too complex for conventional technologies (e.g., problems that do not have an algorithmic solution or for which an algorithmic solution is too complex to be found) and are often well suited to problems that people are good at solving, but for which traditional methods are not.

One of the major advantages of *neural networks* is that, theoretically, they are capable of approximating any continuous function, and thus the researcher does not need to have any hypotheses about the underlying model, or even to some extent, which variables matter.

## 3. Comparison of neural networks with other data mining techniques

### 3.1 Neural networks versus conventional computing techniques

Neural networks take a different approach to problem solving than that of conventional computers. Conventional computers use an algorithmic approach i.e. the computer follows a set of instructions in order to solve a problem. Unless the specific steps that the computer needs to follow are known the computer cannot solve the problem. That restricts the problem solving capability of conventional computers to problems that we already understand and know how to solve. But computers would be so much more useful if they could do things that we don't exactly know how to do.

Neural networks process information in a similar way the human brain does. The network is composed of a large number of highly interconnected processing elements (neurones) working in parallel to solve a specific problem. Neural networks learn by example. They cannot be programmed to perform a specific task. The examples must be selected carefully otherwise useful time is wasted or even worse the network might be functioning incorrectly. The disadvantage is that because the network finds out how to solve the problem by itself, its operation can be unpredictable.

On the other hand, conventional computers use a cognitive approach to problem solving; the way the problem is to solved must be known and stated in small unambiguous instructions. These instructions are then converted to a high level language program and then into machine code that the computer can understand. These machines are totally predictable; if anything goes wrong is due to a software or hardware fault.

Neural networks and conventional algorithmic computers are not in competition but complement each other. There are tasks are more suited to an algorithmic approach like arithmetic operations and tasks that are more suited to neural networks. Even more, a large number of tasks, require systems that use a combination of the two approaches (normally a conventional computer is used to supervise the neural network) in order to perform at maximum efficiency.

### 3.2 Main drawbacks of classical statistical techniques

Classical statistical techniques have the following main drawbacks :

- They impose restrictions on the number of input data: one is limited to a few inputs among dozens or hundreds available, imposing a priori variable selection, with all the inherent pitfalls ;
- Regressions are performed using simple dependency functions (linear, logarithmic) that are not very realistic ;
- The hypothesis is made that there only one dependency function over the whole data set, instead of many distinct niches ;
- Other hypotheses imposed by their underlying theories (normal distributions, equiprobabilities, uncorrelated variables) known to be violated, but that are necessary for their good operation ;
- The need to use hare-brained methods to transform data.

In all cases one has to call on an expert of the method and perform many weeks, even a few months' worth of work.

More fundamentally, to quote Professor Peter D.M. MacDonald, of McMaster University, Ontario, Canada, about The Saarenvirta data set, 'Traditional approaches to statistical inference fail with large databases, however, because with thousands or millions of cases and hundreds or thousands of variables there will be a high level of redundancy among the variables, there will be spurious relationships, and even the weakest relationships will be highly significant by any statistical test. The objective is to build a model with significant predictive power. It is not enough just to find which relationships are statistically significant'.

### 3.3 Neural nets and linear statistics

Comparing neural networks with linear statistics, the following conclusions are drawn:

- Neural nets need less constraining hypotheses (the dependency must be a function, nothing more);
- Qualitative (enumerative) data are straightforwardly handled ;
- No preprocessing, «binning», simplification or reduction of quantitative variables is necessary ;
- It has been demonstrated that linear regression and logistic regression are particular cases of neural nets (with one layer and a linear threshold function). The same thing happens with Principal Component Analysis (PCA) whose values are contained in the weights of the neurodes of a one-layer, linear-threshold function network performing self-association (i.e. with outputs being the same as the inputs, and fewer neurodes than in/outputs in the intermediate layer: see at the end of the document for a picture).

Neural networks do what every good «new» theory does: they encompass and generalise the previous statistical techniques. For more details see (Ripley, 1996).

### 3.4 Neural networks and scorecards

Neural models are «nonlinear», that is, they are better able to account for the complexity of human behaviour than linear, more simplistic ones. For instance, if a loan's risk decreases with age for renters and increases with age for owners, a neural network will do what a scorecard cannot do and «understand» the relation between both variables with respect to the outcome, thus performing better reclassifyings.

Taking into account the relation between age and lodging situation, rather than striking a weighting on account of each variable separately, is known as nonlinearity, and is the main advantage of neural networks.

### 3.5 Neural networks and expert systems

When we compare the neural networks with expert systems, the following conclusions are drawn:

- The «knowledge» has to be specially formatted by a «knowledge engineer» starting from the expert's knowledge. Using expert systems, the time required for analysis and formatting are increased and not decreased;
- Experts systems are slow to develop, and slow and expensive to update;
- Material problems are not the only ones: there are fundamental, epistemological problems like: it is not appropriate to represent knowledge in the form of decision rules or decision trees . No tolerance to missing or erroneous values;
- The need to manually update (at a cost) expert systems each time expertise changes, while with a neural net, it is enough to launch a new learning process.

The immense advantage of neural nets is that one goes directly from factual data to the model without any human work, without tainting the result with oversimplification or preconceived ideas.

**3.6 Neural networks and typical profiles**

- The typical profile reflect the centre (centroid) of the cloud, not its shape ;
- One does not know how many niches there are; the centroid might lie in the middle of two niches and outside both.

**3.7 Neural networks and «by hand» modeling**

- This calls on experts, and is costly and slow (3-4 models per year) ;
- One has to wait until the «expert» has developed his own «mental model»

Neural networks build the model directly without waiting for anyone to build their own «expertise».

**3.8 Main advantages of neural networks**

We can outline the following advantages of neural networks comparing with other data mining techniques :

- Ability to account for any functional dependency. The network discovers (learns, models) the nature of the dependency without needing to be prompted. No need to postulate a model, to amend it, etc.;
- One goes straight from the data to the model without intermediary, without recoding, without binning, without simplification or questionable interpretation;
- Insensitivity to « moderate » noise or unreliability in the data.
- No conditions on the predicted variable: it can be a Yes/No output, a continuous value, one or more classes among n, etc.;
- Ease of handling, much less human work than traditional statistical analysis. No competence in math, statistics or informatics is required;
- No need to manually detect collinearities;
- In segmentation, the net determines by itself how many clusters there are in each class;
- For the novice user, the idea of learning is easier to understand than the complexities of multivariate statistics;
- Speed of use: 10 microseconds when hardwired, a few milliseconds on a 1 GHz computer;
- Spatial relations (geomarketing etc.) are easily analysed and modelled;
- The final model is continuous and derivable and lends itself easily to further work: detection of centroids (typical profiles) by «gradient ascent», sensitivity computation (by partial derivation), display with any desired precision, etc.;
- Explanation and prediction - the consumers' decision-making processes are badly understood and so far badly modelled by causal models. In other words, bringing out behaviour rules as such seems inherently difficult. This reminds us of the bitter failures of expert systems. The neural net works better exactly because it models associations and not causes;
- The «black box» problem - validating the model. The neural model is validated using a number of examples that were excluded from the learning set, called the «test set». Expected and predicted values

are compared. Performance is evaluated using the reclassifying rate (when classifying) or the average deviation between expected and predicted values when predicting a numerical value. This validation method is as objective as can be, is common in data analysis and can be applied to any kind of predictor, even to an expert person.

### 3.9 Main drawbacks of neural nets

Of course, neural networks do not dispense one from knowing their problem, defining the classes in a meaningful way, not omitting important variables, etc. The main problem is that the neural net is a «black box»: it does not explain its decision. Let us notice,though, that the same can be said about more traditional techniques, regression analysis for instance.

An important disadvantage, however, is that the final solution depends on the initial conditions of the network, and, it is virtually impossible to "interpret" the solution in traditional, analytic terms, such as those used to build theories that explain phenomena.

### 3.10 Why use neural networks?

Neural networks have two fundamental advantages :

- The ability to represent any function, linear or not, simple or complicated. Neural nets are what mathematicians call «universal approximators» (Kolmogorov, 1957);
- The ability to learn from representative examples, by «error backpropagation» (Le Cun, McClelland, 1986, Werbos, 1974). Model building, or «learning», is automatic.

Neural networks, with their remarkable ability to derive meaning from complicated or imprecise data, can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques. A trained neural network can be thought of as an "expert" in the category of information it has been given to analyse. This expert can then be used to provide projections given new situations of interest and answer "what if" questions.

Other reasons for their application include:

- Adaptive learning: An ability to learn how to do tasks based on the data given for training or initial experience;
- Self-Organisation: An ANN can create its own organisation or representation of the information it receives during learning time;
- Real Time Operation: ANN computations may be carried out in parallel, and special hardware devices are being designed and manufactured which take advantage of this capability;
- Fault Tolerance via Redundant Information Coding: Partial destruction of a network leads to the corresponding degradation of performance. However, some network capabilities may be retained even with major network damage.

However, building neural model belongs to data analysis and not to magic (even though, to quote Arthur C. Clarke, sufficiently advanced technology is indistinguishable from magic). The data must be explicative and in a large enough amount. Neural networks do not perform miracles. But if used sensibly they can produce some amazing results.

## 4. Applications of neural networks to data mining processes in practice

Neural networks have broad applicability to real world business problems. In fact, they have already been successfully applied in many industries.

The tasks to which artificial neural networks are applied tend to fall within the following broad categories:

- Function approximation, or regression analysis, including time series prediction and modeling.

- Classification, including pattern and sequence recognition, novelty detection and sequential decision making.
- Data processing, including filtering, clustering, blind source separation and compression.

Application areas include system identification and control (vehicle control, process control), game-playing and decision making (backgammon, chess, racing), pattern recognition (radar systems, face identification, object recognition and more), sequence recognition (gesture, speech, handwritten text recognition), medical diagnosis, financial applications, knowledge discovery in databases, visualization and e-mail spam filtering.

Since neural networks are best at identifying patterns or trends in data (for the purposes of data mining), they are well suited for prediction or forecasting needs including:

- sales forecasting ;
- industrial process control ;
- customer research ;
- data validation ;
- risk management;
- target marketing .

ANN are also used in the following specific paradigms: recognition of speakers in communications; diagnosis in medicine; recovery of telecommunications from faulty software; interpretation of multimeaning words; undersea mine detection; texture analysis; three-dimensional object recognition; hand-written word recognition; facial recognition, etc.

**4.1 Neural networks in medicine**

Artificial Neural Networks (ANN) are currently a 'hot' research area in medicine and it is believed that they will receive extensive application to biomedical systems in the next few years. At the moment, the research is mostly on modelling parts of the human body and recognising diseases from various scans (e.g. cardiograms, CAT scans, ultrasonic scans, etc.).

Neural networks also contribute to other areas of research such as neurology and psychology. They are regularly used to model parts of living organisms and to investigate the internal mechanisms of the brain.

Neural networks are ideal in recognising diseases using scans since there is no need to provide a specific algorithm on how to identify the disease. Neural networks learn by example so the details of how to recognise the disease are not needed. What is needed is a set of examples that are representative of all the variations of the disease. The quantity of examples is not as important as the 'quantity'. The examples need to be selected very carefully if the system is to perform reliably and efficiently.

Neural Networks are used experimentally to model the human cardiovascular system. Diagnosis can be achieved by building a model of the cardiovascular system of an individual and comparing it with the real time physiological measurements taken from the patient. If this routine is carried out regularly, potential harmful medical conditions can be detected at an early stage and thus make the process of combating the disease much easier.

A model of an individual's cardiovascular system must mimic the relationship among physiological variables (i.e., heart rate, systolic and diastolic blood pressures, and breathing rate) at different physical activity levels. If a model is adapted to an individual, then it becomes a model of the physical condition of that individual. The simulator will have to be able to adapt to the features of any individual without the supervision of an expert. This calls for a neural network.

Another reason that justifies the use of ANN technology, is the ability of ANNs to provide sensor fusion which is the combining of values from several different sensors. Sensor fusion enables the ANNs to learn complex relationships among the individual sensor values, which would otherwise be lost if the values were individually analysed. In medical modelling and diagnosis, this implies that even though each sensor in a set may be sensitive only to a specific physiological variable, ANNs are capable of detecting complex medical conditions by fusing the data from the individual biomedical sensors.

ANNs are used experimentally to implement electronic noses. Electronic noses have several potential applications in telemedicine. Telemedicine is the practice of medicine over long distances via a communication link. The electronic nose would identify odours in the remote surgical environment. These identified odours would then be electronically transmitted to another site where an door generation system would recreate them. Because the sense of smell can be an important sense to the surgeon, telesmell would enhance telepresent surgery.

An application developed in the mid-1980s called the "instant physician" trained an autoassociative memory neural network to store a large number of medical records, each of which includes information on symptoms, diagnosis, and treatment for a particular case. After training, the net can be presented with input consisting of a set of symptoms; it will then find the full stored pattern that represents the "best" diagnosis and treatment.

### 4.2 Neural networks in business

Business is a diverted field with several general areas of specialisation such as accounting or financial analysis. Almost any neural network application would fit into one business area or financial analysis.
There is some potential for using neural networks for business purposes, including resource allocation and scheduling. There is also a strong potential for using neural networks for database mining, that is, searching for patterns implicit within the explicitly stored information in databases. Most of the funded work in this area is classified as proprietary. Thus, it is not possible to report on the full extent of the work going on. Most work is applying neural networks, such as the Hopfield-Tank network for optimization and scheduling.

#### 4.2.1 Banking

ANN are useful for predicting the issue of loans (credit scoring) and predicting the recovery of bad loans (recovery scoring). They are applied for forecasting the behaviour of new customers, identifying «good risks» and offering them loans.

#### 4.2.2 Finance

ANN are used for redicting share prices, volatilities. They help for position-taking, Portfolio and asset management.

#### 4.2.3 Industry

ANN predicting demand for a product or service for better production planning. Servo-control for machines or chemical reactors (through «model inversion») is applied.

#### 4.2.4 Marketing

There is a marketing application which has been integrated with a neural network system. The Airline Marketing Tactician (a trademark abbreviated as AMT) is a computer system made of various intelligent technologies including expert systems. A feedforward neural network is integrated with the AMT and was trained using back-propagation to assist the marketing control of airline seat allocations. The adaptive neural approach was amenable to rule expression. Additionaly, the application's environment changed rapidly and constantly, which required a continuously adaptive solution. The system is used to monitor and recommend booking advice for each departure. Such information has a direct impact on the profitability of an airline and can provide a technological advantage for users of the system.

While it is significant that neural networks have been applied to this problem, it is also important to see that this intelligent technology can be integrated with expert systems and other approaches to make a functional system. Neural networks were used to discover the influence of undefined interactions by the various variables. While these interactions were not defined, they were used by the neural system to develop useful conclusions. It is also noteworthy to see that neural networks can influence the bottom line.

Predicting function of neural networks facilitates the process of marketing targeting and turnover prediction; analysis and prediction of crime, tax return analysis for fraud detection; risk analysis for renters; economic forecasting; etc. (Bigus, 1996).

Equipped with accurate forecasts, the customers are able to save on resources or improve their return on investment. Neural nets do not replace staff, but they improve an organisation's operation by letting them build models they would not have had the time to make using older techniques.

#### 4.2.5 Credit evaluation

The HNC company, founded by Robert Hecht-Nielsen, has developed several neural network applications. One of them is the Credit Scoring system which increase the profitability of the existing model up to 27%. The HNC neural systems were also applied to mortgage screening. A neural network automated mortgage insurance underwritting system was developed by the Nestor Company. This system was trained with 5048 applications of which 2597 were certified. The data related to property and borrower qualifications. In a conservative mode the system agreed on the underwritters on 97% of the cases. In the liberal model the system agreed 84% of the cases. This is system run on an Apollo DN3000 and used 250K memory while processing a case file in approximately 1 sec.

## 5. Neural network software

**Neural network software** is used to simulate, research, develop and apply artificial neural networks, biological neural networks and in some cases a wider array of adaptive systems.

### 5.1 Simulators

Neural network simulators are software applications that are used to simulate the behavior of artificial or biological neural networks. They focus on one or a limited number of specific types of neural networks. They are typically stand-alone and not intended to produce general neural networks that can be integrated in other software. Simulators usually have some form of built-in visualization to monitor the training process. Some simulators also visualize the physical structure of the neural network.

#### 5.1.1 Research simulators

Historically, the most common type of neural network software was intended for researching neural network structures and algorithms. The primary purpose of this type of software is to, through simulation, gain a better understanding of the behavior and properties of neural networks. Today in the study of artificial neural networks, simulators have largely been replaced by more general component based development environments as research platforms.

Commonly used artificial neural network simulators include the Stuttgart Neural Network Simulator (SNNS), PDP++ and Java NNS.

SNNS (Stuttgart Neural Network Simulator) is a neural network simulator originally developed at the University of Stuttgart. While it was originally built for X11 under Unix, there are Windows ports. Its successor JavaNNS never reached the same popularity.

PDP++ (Parallel Distributed Processing++) is neural simulation software written in C++ for the creation of connectionist models. Development initially began in 1995 at Carnegie Mellon University, and is currently in version 3.2, with version 4.0 well underway at the University of Colorado at Boulder. The software is featured in the textbook Computational Explorations in Cognitive Neuroscience.

PDP++ features a modular design, based on the principles of object-oriented programming. It has been known to work in Microsoft Windows, various unices, Solaris, Darwin and several other operating systems. C-Super-Script (variously, CSS and C^C), a built-in C++-like interpreted scripting language, allows access to virtually all simulator objects and can initiate all the same actions as the GUI, and more.

In the study of biological neural networks however, simulation software is still the only available approach. In such simulators the physical biological and chemical properties of neural tissue, as well as the electromagnetic impulses between the neurons are studied.
Commonly used biological network simulators include XNBC and the BNN Toolbox for MATLAB.

### 5.1.2 Data analysis simulators

Unlike the research simulators, the data analysis simulators are intended for practical applications of artificial neural networks. Their primary focus is on data mining and forecasting. Data analysis simulators usually have some form of preprocessing capabilities. Unlike the more general development environments data analysis simulators use a relatively simple static neural network that can be configured. A majority of the data analysis simulators on the market use backpropagation networks or self-organizing maps as their core. The advantage of this type of software is that it is relatively easy to use. This however comes at the cost of limited capability. Some data analysis simulators work in conjunction with other computational environments, such as Microsoft Excel or MATLAB. Data analysis simulators include: Ward Systems NeuroShell, Palisade, NeuralTools, Alyuda NeuroIntelligence, Netlab, EasyNN-plus, NeuralWorks Predict and dozens of others.

Alyuda NeuroIntelligence is an Neural network software for experts designed for intelligent support in applying neural networks to solve real-world forecasting, classification and function approximation problems; to use intelligent features to preprocess datasets; to find efficient architecture and to analyze performance and apply the neural network to new data. Experts can create and test their solutions much faster, increase their productivity and improve results.

NeuralWorks Predict is an integrated, state-of-the-art tool for rapidly creating and deploying prediction and classification applications. Predict combines neural network technology with genetic algorithms, statistics, and fuzzy logic to automatically find optimal or near-optimal solutions for a wide range of problems. Predict incorporates years of modeling and analysis experience gained from working with customers faced with a wide variety of analysis and interpretation problems. In Microsoft Windows environments NeuralWorks Predict can be run either as an add-in for Microsoft Excel to take advantage of Excel's rich data handling and graphing capabilities, or as a command line program that offers powerful batch mode processing. In UNIX and Linux environments, NeuralWorks Predict runs as a command line program.

## 5.2 Development environments

Development environments for neural networks differ from the software described above primarily on two accounts – they can be used to develop custom types of neural networks and they support deployment of the neural network outside the environment. In some cases they have advanced preprocessing, analysis and visualization capabilities.
Popular development environments are:

- the MathWorks NN Toolbox which is based on the MATLAB computing environment;
- the GBLearn2 library of the Lush programming language.

The Neural Network Toolbox extends MATLAB with tools for designing, implementing, visualizing, and simulating neural networks. Neural networks are invaluable for applications where formal analysis would be difficult or impossible, such as pattern recognition and nonlinear system identification and control. The Neural Network Toolbox provides comprehensive support for many proven network paradigms, as well as graphical user interfaces (GUIs) that enable you to design and manage your networks. The modular, open, and extensible design of the toolbox simplifies the creation of customized functions and networks.

### 5.2.1 Component based environments

A more modern type of development environments that are currently favored in both industrial and scientific use are based on a component based paradigm. The neural network is constructed by connecting adaptive filter components in a pipe filter flow. This allows for greater flexibility as custom networks can be built as well as custom components used by the network. In many cases this allows a combination of adaptive

and non-adaptive components to work together. The data flow is controlled by a control system which is exchangeable as well as the adaptation algorithms. The other important feature is deployment capabilities. With the advent of component-based frameworks such as .NET and Java, component based development environments are capable of deploying the developed neural network to these frameworks as inheritable components. In addition some software can also deploy these components to several platforms, such as embedded systems.

Component based development environments include: JOONE, Peltarion Synapse and NeuroDimension NeuroSolutions.

JOONE (Java Object Oriented Neural Engine) is a component based neural network framework built in Java. Joone consists of a component-based architecture based on linkable components that can be extended to build new learning algorithms and neural networks architectures.

Components are plug-in code modules that are linked to produce an information flow. New components can be added and reused. Beyond simulation, Joone also has to some extent multi-platform deployment capabilities.

Joone has an GUI Editor to graphically create and test any neural network, and a distributed training environment that allows for neural networks to be trained on multiple remote machines.

As of 2006, Joone is the only free component based neural network development environment available. Unlike the two other (commercial) systems that are in existence, Synapse and NeuroSolutions, it is written in Java and has direct cross-platform support. A limited number of components exist and the graphical development environment is rudimentary so it has significantly fewer features than its commercial counterparts.

Joone can be considered to be more of a neural network framework than a full integrated development environment. Unlike its commercial counterparts, it has a strong focus on code-based development of neural networks rather than visual construction.

While in theory Joone can be used to construct a wider array of adaptive systems (including those with non-adaptive elements), its focus is on backpropagation based neural networks.

NeuroSolutions is a neural network development environment developed by NeuroDimension. It combines a modular, icon-based (component based) network design interface with an implementation of advanced learning procedures, such as conjugate gradients, Levenberg-Marquardt and backpropagation through time. The software is used to design, train and deploy neural network (supervised learning and unsupervised learning) models to perform a wide variety of tasks such as data mining, classification, function approximation, multivariate regression and time-series prediction.

Synapse is a component based development environment for neural networks and adaptive systems. Created by Peltarion, Synapse allows data mining, statistical analysis, visualization, preprocessing, design and training of neural networks and adaptive systems and the deployment of them. It utilizes a plug-in based architecture making it a general platform for signal processing. The first version of the product was released in May 2006.

The disadvantage of component-based development environments is that they are more complex than simulators. They require more learning to fully operate and are far more complicated to develop. The latter leads to a reduction in software diversity. Simple simulators and custom neural nets have traditionally had a large amateur development scene. There are hundreds of different neural network simulators, but only a few component-based development environments.

### 5.3 Custom neural networks

The majority implementations of neural networks available are however custom implementations in various programming languages and on various platforms. Basic types of neural networks are simple to implement directly. There are also many programming libraries that contain neural network functionality and that can be used in custom implementations.

## 6. Conclusion

Data Mining has a lot to gain from neural networks. Their ability to learn makes them very flexible and powerful. Furthermore there is no need to devise an algorithm in order to perform a specific task; i.e. there is no need to understand the internal mechanisms of that task. They are also very well suited for real time systems because of their fast response and computational times which are due to their parallel architecture.

The neural net supplies answers, but not explanations. Indeed, the neural model embodies correlations (like intuitive associations), not causal relations (explanations). Examining the neural network itself only shows us meaningless numeric values. The neural model is a 'black box'. On the other hand, this model being continuous and derivable, one can, beyond simple interrogation, 'explore' it to determine typical profiles, the degree of explicative power of each variable, reclassify collections of examples to determine their associated probabilities, visualize data and predictions, very predictive: the models are very faithful to reality.

Perhaps the most exciting aspect of neural networks is the possibility that some day 'consious' networks might be produced. There is a number of scientists arguing that conciousness is a 'mechanical' property and that 'consious' neural networks are a realistic possibility.

Neural networks have a huge potential and their integration with artificial intelligence, fuzzy logic and other related subjects will increase their future application to Data mining.

## 7. References

ABDI, H. 1994. A neural network primer. *Journal of Biological Systems,* **2:** 247-281.

ABDI, H., VALENTIN, D., EDELMAN, B.E. 1999. *Neural Networks.* Thousand Oaks: Sage.

ANDERSON, J.A. 1995. *An Introduction to Neural Networks.* ISBN 0-262-01144-1. The MII Press, Cambridge, Massachustts.

ARBIB, M.A. (Ed.) 1995. *The Handbook of Brain Theory and Neural Networks.* The MII Press, Cambridge, Massachustts.

BERRY, M.J.A. and LINOFF, G.S. 2000. *Mastering data mining.* NewYork:Wiley.

BIGUS, J.P. 1996. *Data Mining with Neural Networks*, McGraw-Hill, New York.

BISHOP, C. 1995. *Neural Networks for Pattern Recognition.* Oxford: University Press.

CARLING, A. 1992. *Introducing Neural Networks.*: Sigma Press, Wilmslow, UK.

EDELSTEIN, H.A. 1999. *Introduction to data mining and knowledge discovery (3rd ed).* Potomac, MD: Two Crows Corp.

FAUSETT, L. 1994. *Fundamentals of Neural Networks.* Prentice Hall, NewYork FAYYAD, U.M., PIATETSKY-SHAPIRO, G., SMYTH, P. and UTHURUSAMY, R. 1996. Advances in knowledge discovery and data mining., MIT Press, Cambridge.

FUKUSHIMA, K. 1975. Cognitron: A Self-Organizing Multilayered Neural Network. *Biological Cybernetics* **20:** 121–136.

GARDNER, E.J. and DERRIDA, B. 1988. Optimal storage properties of neural network models. *Journal of Physics A* **21:** 271–284.

HAN, J., KAMBER, M. 2000. *Data mining: Concepts and Techniques*, Morgan-Kaufman, New York.

HASTIE, T., TIBSHIRANI, R. and FRIEDMAN, J.H. 2001. *The elements of statistical learning : Data mining, inference, and prediction,* Springer, NewYork.

HAYKIN, S. 1994. *Neural Networks: A Comprehensive Foundation.* Prentice Hall. New York.

KLIMASAUSKAS, CC. 1989. The 1989 Neuro Computing Bibliography. The MII Press, Cambridge, Massachustts.

MAASS, W. and MARKRAM, H. 2002, On the computational power of recurrent circuits of spiking neurons. *Journal of Computer and System Sciences* **69(4):** 593–616.

MACKAY, DAVID. 2003. *Information Theory, Inference, and Learning Algorithms.* Cambridge University Press Pregbon, D. (1997) data mining. Statistical computing and graphyics newesletters, 7,p.8.

MANDIC, D. and CHAMBERS, J. 2001. *Recurrent Neural Networks for Prediction: Architectures, Learning algorithms and Stability*. Wiley.
MINSKY, M.L. and PAPERT, S.A. 1969. *Perceptrons, An introduction to computational geometry*, MIT press, expanded edition.
PATTERSON, D. 1996. *Artificial Neural Networks*. Singapore: Prentice Hall.
PREGIBON, D. 1997. *Data Mining*. Statistical Computing and Graphics, **7**,8.
REILLY, D.L., COOPER, L.N. and ELBAUM, C. 1982. A Neural Model for Category Learning. *Biological Cybernetics* **45:** 35–41.
RIPLEY, B.D. 1996. *Pattern Recognition and Neural Networks*. Cambridge University Press.
WASSERMAN, P.D. 1989. *Neural computing theory and practice*. Van Nostrand Reinhold.
WEISS, S.M. and INDURKHYA, N. 1997. *Predictive data mining: A practical guide*. Morgan-Kaufman, NewYork.
WESTPHAL, C., BLAXTON, T. 1998. *Data mining solutions*. Wiley, NewYork.
WITTEN, I.H. and FRANK, E. 2000. *Data mining*. Morgan-Kaufmann, NewYork.
ALYUDA Research Company. http://www.alyuda.com/neural-network-software.htm

A Novel Approach to Modelling and Diagnosing the Cardiovascular System.
http://www.emsl.pnl.gov:2080/docs/cie/neural/papers2/keller.wcnn95.abs.html
Artificial Neural Networks in Medicine.
http://www.emsl.pnl.gov:2080/docs/cie/techbrief/NN.techbrief.html
Data Mining.
http://datamining.itsc.uah.edu/index.jsp

Electronic Statistics Textbook, Stat Soft Inc., 1984-2006, http://www.statsoft.com/textbook/stdatmin.html.
Kdnuggets. http://www.kdnuggets.com/index.html

MATLAB
http://en.wikipedia.org/wiki/MATLAB

Neural Networks at Pacific Northwest National Laboratory
http://www.emsl.pnl.gov:2080/docs/cie/neural/neural.homepage.html

Neural network software
http://en.wikipedia.org/wiki/Neural_network_software

Neuralware
http://www.neuralware.com/products.jsp

Portal on forecasting with artificial neural networks
http://www.neural-forecasting.com/neural_forecasting_associations.htm

Stergiou Ch. and Siganos D., Neural Networks,
http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html
Stuttgart Neural Network Simulator
http://en.wikipedia.org/wiki/Stuttgart_Neural_Network_Simulator

---

Received 2 December 2006
Accepted 28 June 2007