§ 26 A NEURAL NETWORK CORRECTS ITS ASSUMPTIONS

AN ARTIFICIAL NEURAL NETWORK SIMULATES THE BRAIN

An 'Artificial Neural Network' (ANN) is a simple mathematical simulation of nerve cells ('neurons') interconnected with conductors ('axons') in a living organism.
Every human contains an estimated 100 billion neurons, each with many incoming and outgoing connections with other neurons, senses, muscles or organs.
They can correct their connections by weight and, moreover, allow them to die or arise (the latter does not yet function in the artificial simulation).

In a neuron, the incoming signals are combined and the neuron determines to what extent this combination should be passed on as an outgoing signal to a next layer of neurons (decelerated or strengthened).
A large artificial neural network (ANN) can contain 1000 neurons, each with a limited number of unchangeable incoming and outgoing connections.

The input is a set of patterns, the output is their categorization

The input (from observations) of the ANN is a table with numbers 'normalized' between 0 and 1 ('scaling'). Each row of that table forms a pattern of values, for example the sequence of number codes for the successive colors in a oldfasioned windows icon (16x16 = 256 pixels). So each pattern *row* contains a different icon.
Each *column* of the table represents a location bound pixel. Such a pixel column is called 'input neuron'.

There are then 256 input neurons (for each pixel in the same place in all icons). They pass the color code in each column for each row into a smaller number (for example, 100) of 'hidden neurons' (the 'hidden layer' of the ANN).
Each hidden neuron thus receives 256 signals (numbers) of the 256 input neurons per input line and combines them into an output signal to a next layer of neurons.

Each part of a pattern is taken more or less serious for recognition

However, the neurons of a hidden layer do not attach the same value or 'weight' to each incoming column signal.These weights are initially random, but they are later adjusted by an iterative feed-back 'learning process'.

Every hidden neuron has its own preliminary preferences, and resistances against the incoming signals. Each neuron in the 'hidden layer' weighs the (for example 256) incoming signals (columns, color codes per located pixel, numbers between 0 and 1) with their own list of provisional weights per column.
These weights are adjusted in a subsequent round of calculations.

How does such a hidden neuron now combine all those (eg 256) signals into a single outgoing signal (number) for the next layer of hidden neurons? It multiplies each incoming signal with the corresponding provisional weight. It adds up these products.

That sum is edited with a formula[a] that gives them a value between 0 and 1 ('scaling' or simply 0 or 1). The result is passed to the neurons of a subsequent layer.

Recognition errors determine what has to be taken more serious and what less
The next layer can be a smaller number of hidden neurons that do the same with the results of the previous layer, until it reaches a single neuron as the last layer.
This 'output neuron' then also has its own list of provisional weights, with which the output of the ANN is determined in the same way as the outcomes of the previous layer for each row (pattern, eg windows icon).

What you can do with that end result then depends on the pattern that you want to be recognized in the rows that were entered at the beginning. Assuming that you want the ANN to recognize the letter A in the entered patterns (the 16x16 pixel icons), you have to give each row with the pattern that looks like an A the value 0 (1 if not).

This creates a column of ones and zeroes ('desire', desired outcomes) in addition to the list of input values of the ANN. You subtract that from the output values that ANN found, in order to find a list of deviations.
The positive square of each deviation is called 'error'. The sum of all those errors you share by their number to find the average error ('Mean Squared Error' or 'MSE').

Training supposes recognizing serious indicators from known examples
Now you can adjust all weights so that MSE is as small as possible. A computer can do that for you.[b] That does not happen at once. After each adjustment you will see MSE (the percentage of errors) become smaller. It adjusted all weights a bit. You can specify in advance how many rounds it has to make. If you are not satisfied (for example, if the MSE is not smaller than 0.01 or 1% chance of errors), you simply put it back to work.

This process is called 'training the neural network'. The network started with random weights. That stands for a network that does not 'know' anything yet. As soon as the chance of errors is less than 1%, the network has 'learned' to recognize the letter A almost faultlessly. That knowledge is stored in the table weights that determine how seriously the neurons have to take their input to make no mistakes.

Testing supposes applying serious indicators on unknown examples
You can now test that without giving the right answer. If you keep all adjusted weights at MSE = 0.01 and add a new pattern row to the input list, without your corresponding good (0) or error (1), then that output is 0 ('this is an A!) or 1 ('this is not an A'). You then have a 99% chance that that is true.

a For example $1/(1-e^{-1})$.
b In Excel, for example, there is a 'solver' plug-in that traverses all formulas that have passed the input on the way to the output, and adjusts the weights used to minimize the end result, maximize it, or add the value you can choose.
In this case, you want to make the average error (MSE) as small as possible. So you choose 'minimize'.

How it works, probably nobody can explain to you, but it works, and it is being used on a massive scale.[a]

Neural networks are used in order to recognize patterns
For example, credit providers have many characteristics of their debtors such as age, home ownership, gender, and so on. From experience, they also know who will or will not pay back on time. The question, of course, is whether a pattern for defaulters can be recognized in the row of features per person. It is better not to give a loan to that risk group

For example, you can also recognize a pattern in:
- the course of stock prices of various funds in the past and respond to this;
- the sale of products in different seasons, holiday periods and so on;
- the course of the weather in different weather conditions in previous days;
- properties of plant species to trace their name.

The last application is a classification problem. That is best served by more than one hidden layer. They sequester to a result successively. Different problems may therefore require different network configurations. Many different suitable configurations have been found in the human body for various tasks.

For example, 'circular configurations' are held responsible for our memory. In addition, the neurons and axons can also be very different and their connections can change. Connections can degenerate or be newly created. Our neural system is differentiated and flexible in details, but the *number* of neurons is largely fixed from birth.

If our pattern recognition proceeds according to the scheme of artificial neural networks, it could for example explain how we combine the totally different information from different senses into the constant awareness of a variable object ('synaesthesia'). You can now also imagine how the actions of dozens of muscles are coordinated as a training-based pattern into one specific activity.

EVEN IF YOU KNOW HOW IT WORKS, YOU STILL MAY NOT UNDERSTAND HOW IT WORKS
Such a mathematical simulation of biological examples can lead to other assumptions about our functioning. It appears to be a barely comprehensible alternative to current mathematical optimization techniques. The ANN expresses the weights in numbers, but our own neural network uses chemical reactions. The ANN adjusts these numbers in a round of mathematical feed-back operations ('solver' in Excel). How does that feed back work chemically in a living organism? I cannot answer these questions.

a Choong(2009)Build Neural Network with Excel(WWW)XLPert Enterprise provides a clear explanation for some operating ANN-applications in Excel. These are downloadable from http://www.xlpert.com/buy-now.html for a small fee.
Professional software is offered via http://www.neurosolutions.com/.

YOU CAN SIMULATE A SIMPLE NEURAL NETWORK EVEN IN EXCEL

If you want to present to ANN the list of the shopkeeper from ***Fig. 101*** (p122), you first have to convert that list (***Fig. 138*** **BCD**) into numbers between 0 and 1 ('scaling' in columns **H, I,** and **J**). The shopkeeper must specify whether (**E**) is satisfied (0) or not (1). In this case he is satisfied when the turnover minus advertising (**E**) is greater or less than 10, but ANN is not aware of that motive. She only sees the patterns of **H, I,** and **J**. 'ANN' now becomes the student you will train and test.[a]

	A	B	C	D	E	F	G	H	I	J	K L M N O P Q R S T U
1	month	turnover	price	advertising	t-a	fill in the wish list: 1 of 0		turnover	price	advertising	
2	1	18	4,00	10	8	1	too little profit	0,13	1,00	0,17	
3	2	20	3,75	10	10	0	OK	0,38	0,88	0,17	
4	3	24	3,25	12	12	0	OK	0,88	0,63	0,33	
5	4	17	3,75	8	9	1	too little profit	0,00	0,88	0,00	
6	5	22	3,00	12	10	0	OK	0,63	0,50	0,33	
7	6	21	3,50	14	7	1	too little profit	0,50	0,75	0,50	
8	7	25	2,75	20	5	1	too little profit	1,00	0,38	1,00	
9	8	23	3,00	18	5	1	too little profit	0,75	0,50	0,83	TEST RULE
10					B9-D9			(B9-B$12)/(B$11-B$12)			
11	MAX	25,00	4,00	20,00	ROUNDDOWN(MAX(D$2:D$9);0)						
12	MIN	17,00	2,00	8,00	ROUNDDOWN(MIN(D$2:D$9);0)			normalized values			

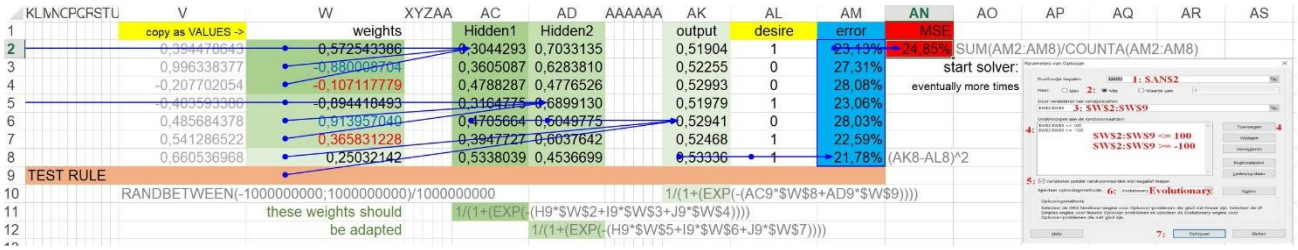
Fig. 138 The wish list and the scaling into normalized values in Excel

Teaching ANN

You first teach ANN how she should evaluate the patterns from month 1 to 7. The eighth month you keep as a test rule to see if ANN shares the judgment of the shopkeeper with the experience of a pattern that is unknown to her. You can also deduce that from 'sales minus advertising', but ANN has no idea of that motive. She only sees the patterns H, I, J evaluated with a 1 or 0 (1 means 'too little profit').

Fig. 138 and ***Fig. 139*** give a complete picture of this simple ANN (6 neurons) in Excel. It consists of the three scaled Input neurons (the columns H, I and J of **Fig. 138**). **Fig. 139** shows a list of provisional weights, two hidden neurons (Hidden 1 and 2) and one Output neuron.

Hidden 1 and 2 (in **Fig. 139** distinguished by shades of green) now come into action. They will both assess every input of the 3 Input Neurons, each with their own weights. The Output neuron will in turn produce the results from Hidden 1 and 2 with two different weights to process output. From that output, the wish list 'desire' is subtracted, so that only the deviations remain. The square of each deviation is called 'error' and the mean of those squares is called 'MSE'.



	K L M N O P Q R S T U	V	W	X Y Z A A	AC	AD	A A A A A A	AK	AL	AM	AN	AO	AP	AQ	AR	AS
1		copy as VALUES ->	weights		Hidden1	Hidden2		output	desire	error	MSE					
2		0,394478643	0,572543386		0,3044293	0,7033135		0,51904	1	23,13%	24,85%	SUM(AM2:AM8)/COUNTA(AM2:AM8)				
3		0,996338377	-0,880008704		0,3605087	0,6283810		0,52255	0	27,31%		start solver:				
4		-0,207702054	-0,107117779		0,4788287	0,4776526		0,52993	0	28,08%		eventually more times				
5		-0,403593386	-0,894410493		0,3164775	0,6899130		0,51979	1	23,06%						
6		0,485684378	0,913957040		0,4705664	0,6049775		0,52941	0	28,03%						
7		0,541286522	0,365831228		0,3947727	0,6037642		0,52468	1	22,59%						
8		0,660536968	0,25032142		0,5338039	0,4536699		0,53336	1	21,78%	(AK8-AL8)^2					
9	TEST RULE															
10		RANDBETWEEN(-1000000000;1000000000)/1000000000						1/(1+(EXP(-(AC9*W8+AD9*W9))))								
11			these weights should		1/(1+(EXP(-(H9*W2+I9*W3+J9*W4))))											
12			be adapted			1/(1+(EXP(-(H9*W5+I9*W6+J9*W7))))										

Fig. 139 ANN does not know anything yet; she has 25% chance to make mistakes

a This simple example has been worked out in Excel according to the more comprehensive scheme of Choong(2009)Build Neural Network with Excel(WWW)XLPert Enterprise.

You have made Excel generate random numbers between 1 and -1 in column V and copied their *values* (not their formulas) as weights in column W. Excel immediately starts generating new random numbers again so that they are no longer the same in ***Fig. 139***, but that does no longer play a role. ANN has not yet learned anything with these random weights, and the number of errors coincides (MSE = 25%). Excel now can start *training* ANN.

Training ANN

You ask the solver to adjust ANN's weights so that her MSE (mean chance on errors) is as small as possible. On his menu you choose the possibility to minimize cell AN2 by adjusting W2 to W9 (the weights) with the evolutionary method. Once you have given that assignment, you see MSE drop to 0% after many rounds (***Fig. 140***).

The solver stops with the announcement that he has found a solution. If that is not less than 1%, then let him work again. The *training* is successful! Now you will *test* ANN.

	KLMNCPGFSTU	V	W	XYZAA	AC	AD	AAAAAA	AK	AL	AM	AN	AO	AP	AQ	AR
1		copy as VALUES ->	weights		Hidden1	Hidden2		output	desire	error	MSE				
2		-0,863189410	-59,640409916		0,9997061	1,0000000		1,00000	1	0,00%	0,00%	SUM(AM2:AM8)/COUNTA(AM2:AM8)			
3		-0,679462612	3,792799049		0,0007080	1,0000000		0,00000	0	0,00%	training succeeded!				
4		0,338076115	70,764977951		0,0000000	1,0000000		0,00000	0	0,00%					
5		-0,320147445	24,823910659		0,9650648	1,0000000		1,00000	1	0,00%					
6		-0,866879983	26,334807400		0,0000076	1,0000000		0,00000	0	0,00%					
7		0,477615253	-26,553737818		0,9997767	1,0000000		1,00000	1	0,00%					
8		0,375445642	66,86685797		0,9999964	0,9997101		1,00000	1	0,00%	(AK8-AL8)^2				
9	TEST RULE	0,033343660	-31,36994359		Test: copy the rule above			0,00000	<-yet false!	you judged too little profit (1)					
10		RANDBETWEEN(-1000000000;1000000000)/1000000000						1/(1+(EXP(-(AC9*W8+AD9*W9))))							
11			these weights give		1/(1+(EXP(-(H9*W2+I9*W3+J9*W4))))										
12			a good result			1/(1+(EXP(-(H9*W5+I9*W6+J9*W7))))									

Fig. 140 ANN has adjusted her weights; she can be tested assessing new patterns

Testing ANN

You copy the formulas AC8 into AC9, AD8 into AD9 and AK8 into AK9 (***Fig. 141***). Hidden 1, Hidden 2 and the Output neuron then display the data of that line in ***Fig. 138*** (H9, I9, J9) with new weights. ANN now gives the correct opinion about the unknown eighth pattern: 1 ('too little profit')!

	KLMNCPGFSTU	V	W	XYZAA	AC	AD	AAAAAA	AK	AL	AM	AN	AO	AP	AQ	AR
1		copy as VALUES ->	weights		Hidden1	Hidden2		output	desire	error	MSE				
2		0,783705692	-59,640409916		0,9997061	1,0000000		1,00000	1	0,00%	0,00%	SUM(AM2:AM8)/COUNTA(AM2:AM8)			
3		-0,907532463	3,792799049		0,0007080	1,0000000		0,00000	0	0,00%	training and test succeeded!				
4		0,756921663	70,764977951		0,0000000	1,0000000		0,00000	0	0,00%					
5		-0,832476586	24,823910659		0,9650648	1,0000000		1,00000	1	0,00%					
6		0,865998598	26,334807400		0,0000076	1,0000000		0,00000	0	0,00%					
7		-0,923172413	-26,553737818		0,9997767	1,0000000		1,00000	1	0,00%					
8		0,358947774	66,86685797		0,9999964	0,9997101		1,00000	1	0,00%	(AK8-AL8)^2				
9	TEST RULE	-0,874606373	-31,36994359		0,9999999	0,9999360		1,00000	<-right!	you judged too little profit (1)					
10		RANDBETWEEN(-1000000000;1000000000)/1000000000						1/(1+(EXP(-(AC9*W8+AD9*W9))))							
11			these weights give		1/(1+(EXP(-(H9*W2+I9*W3+J9*W4))))										
12			a good result			1/(1+(EXP(-(H9*W5+I9*W6+J9*W7))))									

Fig. 141 The judgment of ANN is consistent with the original judgment examples

The ANN detailed here is a simple example that you can expand with many more neurons: you can fill in the rows and columns of the input languages with much more data. The instrument is numeric, but non-numerical data can be differentiated by numerical coding or given a separate column. The 'desire' column can thus contain more values than 0 or 1 or be divided into more columns with more criteria ('this is an A', 'this is a B').

If the average error (MSE) is not small enough yet, you can extend the configuration with more hidden layers than Hidden1 and Hidden2 used here. For different problems you can design different (consecutive, side by side, branched or circular) network configurations.

USUAL STATISTICAL METHODS ONLY RECOGNIZE WELL-KNOWN PATTERNS

What is the difference with statistical 'multivariate analysis' methods? According to some statisticians, ANN offers nothing more than 'non-linear multivariate analysis' and 'discriminant analysis' that can also be performed with conventional statistical software.[a]

This is a one-sided vision from mainly numerical data analysis. It reveals an extensive background of unspoken pattern-recognizing mathematical assumptions (graphs) among statisticians.

The regression patterns are limited in statistical practice to known mathematical functions (linear, exponential, polynomial, etc.). It was already difficult to come up with a job for the distribution of income in **Fig. 100** p121.
Then try to come up with a mathematical function that describes the letter A.
An ANN does not have that restriction: every pattern can be recognized.

If you present an ANN thousand data sets (lines) of one hundred per line of data (columns) and say on each line whether the set is linear, exponential or polynomial (for example in the wish column coded as 1, ½, 0), then can that ANN after training recognize a subsequent data set in this way as such, even though there is no question of 'understanding'.

You can also use this method to recognize formulas such as $ax + b$, $ax^2 + bx + c$ and so on. That goes with their operation in the software of Excel all by itself, but to *make* that software a human neural network was necessary.
In order to be able to make such curves in all their parametric variation and finally even make sense of them, another power is needed.

This ability must not only be able to distinguish identifiable patterns that meet a 'desired' characteristic, but can also dissect a pattern in detached properties (analysis and abstraction) that have not yet been named in columns (input neurons). We had pre-appointed the import categories and were then able to deduct the advertising (and all other costs) from the turnover to also name something as 'profit'.

This not only involves combining predetermined constituent data that makes a pattern recognizable, but also to analyze a given pattern and to make new patterns from those parts. 'Judgment ability' in the classical sense is attributing characteristics to existing parts and wholes and their distinctive or summarizing naming.

a Sarle(1994)Neural Networks and Statistical Models(SAS Users Group)Nineteenth Annual downloadable via http://www.sascommunity.org/seugi/SEUGI1994/Neural Networks and Statistical Models.pdf

The creative ability to come up with patterns that do not yet exist as a whole or in parts (designs) goes even further.
I do not know if ANN configurations can be thought of. Perhaps the connections should be changed. They are still stable in an ANN.

To make an ANN that knows and can do everything what a statistician or a normal person knows and can do, you will need innumerable neurons in different configurations for different tasks. They are apparently spatially separated in the living neural networks. Then you have to train and test that network. The latter may require a training that is just as long-lasting as what people need to come to the 'years of the discernment' before you can clone.

This upbringing now mainly consists of pre-saying and checking whether it is 'understood' (in the limited sense of 'usable for other wishes, in other situations, with different patterns'). The corrections in relation to a precooked list of 'good answers' are, in principle, still external. In evolution, the outside world is basically merciless: wrong is dead. Foremost power is then a favor: first simulate, then do (self-correction).

Every success has to be remembered and can be called up for repetition with an unblocking stimulus in order to survive. That course of action must descend into another configuration in the neural network as a routine that makes time-consuming simulations unnecessary in advance. That makes the reaction speed competitive with a threat (eg fighting, approaching or fleeing). For unknown circumstances, the ability of representation remains.

In the described ANN the import and its classification were precooked.
That applies also for an explanatory statistician, but for the reliable and valid handling of existing formulas, in addition to this description, a long-term training in mathematical foundations and their convincing evidence is assumed.
Those bases include a series of steps that suppose each other in a strict order.

All our actions (eg muscular movement), however, suppose (perhaps less strictly) such a series of preceding, learned, tacit, 'self-evident' routines, assumptions and representations.
Subsequently, if you detach all routine, ready to change usual suppositions then you keep the domain of designing:
searching for possibilities that are not yet true or probable, and even not necessarily wanted by anyone.

§ 27 MATHEMATICS EXTENDS AND REDUCES IMAGINATION

Counting and calculating supposes repetition
The previous sections show the essential role of *repetition* in measuring (geometry), *repetition* in similar cases (probability), *repetition* of the same action (fractal iteration) and *repeated* adaptation (ANN). The repetition is briefly noted in numbers and