The explanatory power of Artificial Neural Networks

Michel Verleysen
Université catholique de Louvain – Research Centre for Information Technology (CERTI)
3, place du Levant
B-1348 Louvain-la-Neuve, Belgium
verleysen@dice.ucl.ac.be

Abstract

Many engineering problems include some kind of recognition: from automatic character recognition to the control of steel quality in a steelworks, through the fault detection in nuclear plants or the prediction of financial rates, it is impossible to enumerate all domains where the key challenge is to identify an input-output relationship between variables or concepts. When the physical relationship is difficult to tackle, models are developed to approximate it.

There are many ways to develop such models. Linear ones are used in many cases, even if it known that the linearity limitation will make the model inadequate. Non-linear models are the solution, but they suffer from many limitations, related to the concept of recognition itself: what is the relation to be recognised if it is only known through examples? Artificial neural networks (ANN), i.e. models based on the remote analogy with the information processing in a human brain, try to answer to this question. ANN models are built (trained) on examples, the purpose being to keep the equilibrium between a correct training and a useful (in some cases meaningful) representation.

Despite the fact that ANN are known to be "blind", or non-explanatory, we intend to show that it is possible to feed to or to extract knowledge from these models; the step towards an explanatory power is then straightforward. But the real question is to know to what extend it is possible to interpret the results of such a "non-explanatory" model: what is the real difference between extracting representable knowledge from a computational model, and using a "blind" model to predict, classify or recognise some relationship?

Introduction

Information technology is a keyword in our modern world. More than a fashion, computer science, artificial intelligence, and many other scientific breakthroughs transform our today life in a way that was unthinkable twenty years ago. Who was able to predict at the time that a pizza could be ordered through Internet and paid with electronic money, and that the same Internet network would be used to book a plane ticket, to read a scientific article or to consult the weather forecasts in a holiday resort?

What electronics and computer science makes feasible is however limited by the inventiveness of engineers, who traditionally build machines able to efficiently achieve repetitive tasks. These tasks have to be described in terms of rules and sequences of operations, or software, and this is probably the main limitation of the state-of-the-art in computer science. In short, what is easily described and analysed is also easily programmed, and thus easily solved by the powerful machines built today. But what is more difficult to describe in terms of rules is hard to solve, just because the programming languages and the way how computers work are not adapted to. For example, multiplying large matrices together, computing the trajectory of a space shuttle or drawing up the balance sheets of a company are tasks which may seem hard because they are computationally intensive, but which are in fact easily "solved" by a computer since they are easily described in terms or rules (mathematical, physical or legal ones respectively). On the other hand, recognising his/her neighbour is an easy task for everybody, but face recognition is a very hard problem for any computer, the reason being that the problem is hard (quite impossible) to describe by rules. Obviously, we don't recognise faces by looking at the hair cut and colour, the eyes colour and shape, the respective location of the neck and the eyes,…; our brain rather analyses a face image as a whole, and gives a decision (for example the name of the person) according to a global perception of this image.

Such comments make the background of the artificial neural networks (ANN) field. Already in the fifties, but more specifically in the early eighties, researchers tried to understand how the (human) brain works, or maybe more realistically how a few neurons can communicate together, exchange information and adapt their functionality to the past experience, in order to replicate their behaviour in computers of a new generation, which would in turn be more adapted to face recognition and other perception tasks such as speech and image recognition, sensory-motor control, fuzzy concepts association,…

Traditional artificial intelligence (AI) was not the right answer to this new challenge. Expert systems for example are now considered as tools able to integrate a qualitative dimension in the processing of information (Cottrell, 1995), rather than the miracle solution to "intelligent" problems. ANN may be considered as artificial intelligence too, while ANN techniques are radically different from "traditional" AI ones.

Modelisation

The domain of artificial neural networks is large and multidisciplinary, and this has two consequences. On the positive side, knowledge and experience acquired in each field concerned by ANN (from biology to electronics, through mathematics, statistics, control, computer science,…) is used to build new theories and concepts, as we will detail below. On the other side, transdisciplinary research is naturally less specific and thus less advanced in a particular field. The interest of ANN research is found in the transdisciplinary character itself, the purpose being to use ideas from some fields (biology and neuroscience), tools from others (mathematics, statistics and computer science), to build new tools (neuroengineering) that can be used in many application areas (control, recognition, time series prediction, data analysis,…). We will limit our discussion to neuroengineering, an area that we will try to define below.

Modelisation is a primary goal in many scientific domains. Science itself tries to create and define models whose aim is to be as generic as possible. Modern science assumes that fundamental laws exist and control all physical phenomena. A strong assumption is that these laws are observable, even if we don't know them; Maxwell's equation are a good example. Many electromagnetic phenomena were observed, understood and even modelled through equations before Maxwell; his unified theory has the merit to be compatible with previous observations and laws, but also to be more general and thus more simple in the concepts. One could find many other examples, such as Einstein's relativity,…

We generally assume that the laws are observable through experiences, but also that the experiences can be repeated indefinitely; this is an essential basic statistical concept: many experiences, or samples, are needed to build a theory, i.e. to discover a "model" and/or to fit its parameters. We will come back below on the dilemmas between building and fitting models, and also between fitting and generalisation.

It seems that science is build around the concept of models. But obviously, a human brain does not work in the same way. In the context of the face recognition problem mentioned above, nobody tries to create a (mental) model of face characteristics before recognising someone in the street. Our conceptualisation is fuzzier, more ambiguous, but richer too; we do not build "classical" models, but we build something else, much more difficult to describe. As stated above, this difficulty in the task or problem description is reflected into the inadequacy of traditional computers to handle the problem. How to "program" the solution of a problem which is itself difficult to describe? The pioneering ideas of neural networks were an attempt to answer to this deadlock: if a human brain can easily handle problems which seem complex for a traditional computer, building machines that "imitate" the human brain in some way could be the solution.

Neuroscience and neuroengineering

To "imitate" the human brain could seem a science-fiction goal. Of course, the intention is not to build a super-computer having human capabilities… On the contrary, the goal of the ANN field is to get ideas from the brain biology and to use them in new computer architectures, in order to build machines which are more adapted to solve perception tasks.

Table 1 lists compares some pertinent characteristics of a "traditional computer" (based on Von Neumann's architecture) and of a human brain. It should be noticed that the items listed are indicative and qualitative only, and that each of them should merit a detailed discussion…

traditional computer	human brain
single processor	massive parallelism (neurons and synapses)
high speed (100 MHz)	slow speed (100 Hz)
sequences of instructions (software)	learning (adaptation)
uniqueness of solutions	fuzzy behaviour and many possible solutions
very sensitive to errors	fault-tolerant
deterministic and ordered variables	fuzzy and non-quantified concepts

Table 1: a few indicative characteristics of traditional computers and human brains

What makes a human brain so powerful (compared to our PCs…) in tackling perceptive problems, such as reading, speech recognition, sensory-motor coordination, face recognition,..? The two main ideas are first the massive parallelism of neurons and synapses contained in the brain, and secondly the adaptation of the huge number of parameters (synaptic coefficients) according to the experience. Based on these two concepts, researchers first tried to model how real neurons and synapses operate (see for example the pioneering work of MacCullogh and Pitts 1943), and then tried to imitate this mode of operation into artificial machines… and models! Most ANN models are far from the biological reality; modelling brain neurons and synapses, and developing computational tools able to perform efficiently in perceptive tasks are different businesses, despite a common inspiration. Werbos (1997) distinguishes between neuroscience and neuroengineering: the first research field aims at understanding how a brain works, the second one at mimicking its potentialities. Figure 1 illustrates this difference: "Neuroengineering tries to develop algorithms and architectures, inspired by what is known about brain functioning, to imitate brain capabilities which are not yet achieved by other means. By demonstrating algorithm capabilities and properties, it may raise issues which feed back to questions or hypotheses from neuroscience" (Werbos 1997).

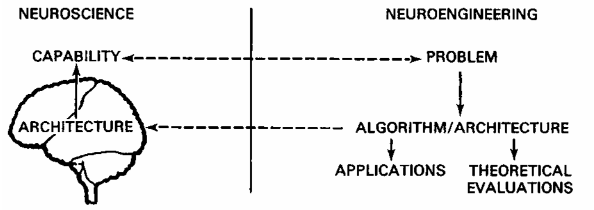

Figure 1: neuroscience and neuroengineering. From (Werbos, 1991).

An important point to notice is that most neuroengineering researchers do not try to imitate *how* real neurons work, but try to imitate *what* they can do. This is clearly a specific assumption: our interest is to build something which works and which can be *used*, whatever is the way to reach the goal. In that sense, neuroengineering is often far from neuroscience, because most ANNs are far from any biological plausibility…

Does we have to model reality?

Maybe a better question should be "What is reality?". Statisticians (Mouchart 1998) consider that the starting point of any analysis consists in observations, and not in reality. Indeed what could be reality if it is not observable? In any situation, we have a (finite) set of observations, and we assume that these data represent reality. We could for example measure the tide at a specific coast location, each day during ten years, and try to guess (or to "predict") what will be the tide during the next two years. By limiting our observations to one value each day during ten years, we assume that the process which governs tides is entirely described by this finite set of observations. This is obviously not the case, nor it is in most modelisations of natural or physical phenomena. However, we might be happy with our tide prediction, depending on its accuracy; it has no sense to expect an infinite precision in the forecast, first because we understand that we will never get it, but secondly because it is not useful too!

What we try to elaborate is then a useful model of the observations, rather than a theoretical complex model of reality. This model should thus be in accordance with the data themselves (obviously!), but also with some kind of "expected reality behind the observations"; this is illustrated in figure 2, where the dots represent the observations, and the plain lines three models of the reality. The plain line (a) is not acceptable, since it does not fit the data adequately, while both plain lines (b) and (c) fit the data; however, everybody will agree that the plain line (b) is closer from reality than line (c). This phenomenon is known as "overfitting": the expected result of a modelisation is line (b), while mathematical criteria measuring how the observations are fitted will give the preference to line (c) if some precautions are not taken.

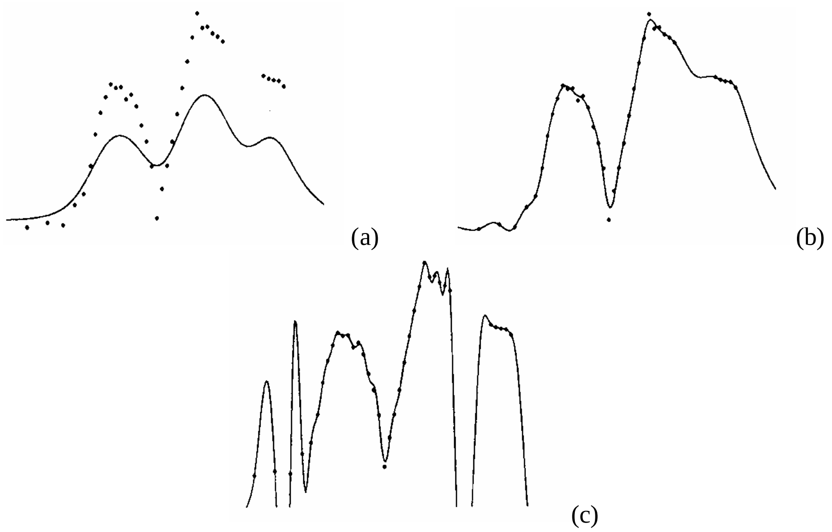

Figure 2: fitting and overfitting data. From (MacKay 1995).

What are Artificial Neural Networks?

ANNs are tools invented to model observations, and not reality. This is a strong assumption, or limitation depending on the point-of-view. ANNs are *learning* models: they try to modelize something (a relation, a function, clusters, a dynamical process,…) by building a model which is as general as possible, but which includes a lot of parameters which are then fitted to achieve the expected goal. Some people will call them non-parametric models, since they have the universal approximation property (see below). This is not exactly true in a statistical context, but the distinction between parametric and non-parametric models becomes senseless at this point.

ANNs are useful tools in data analysis and statistics; in fact, they differ from traditional data analysis or statistic techniques by their implementation, but not by their goals. One of the main characteristics of ANNs versus classical methods is that ANNs are essentially non-linear. They are thus inherently more powerful, since they can perform non-linear *and* linear analysis, where other methods are limited to find linear relationships between data. This advantage is however balanced by an increased complexity, both at the implementation and the computational point-of-views. Before going further in the discussion bout ANNs and their explanatory power, we will briefly describe a few ANN models and their possible applications.

Supervised networks

Supervised networks can be viewed as black boxes implementing a relation (a function) which is known through examples. By examples, we mean input-output pairs, as in the example of figure 2 where the X-axis represents the input and the Y-axis the output of a scalar function; in this example, the function is "known" through 37 examples, illustrated by dots. ANNs can of course cope with vector input and outputs instead of scalar ones. The principle of supervised networks is illustrated in figure 3. The neural network implements an input-output relationship, parametrized at random before learning. Learning consists in several steps:

1. the inputs of an input-output pair presented to the network;
2. the ANN computes the associated outputs;

3. the outputs computed by the ANN is compared to (substracted from) the desired outputs, i.e. the outputs of the input-output pair;
4. the result of the comparison is used to slightly modify the network parameters (the "weights") in order to make the ANN better approximate the input-output pair;
5. operations 1 to 4 are repeated for all known input-output pairs (the observations), usually several times for each pair.

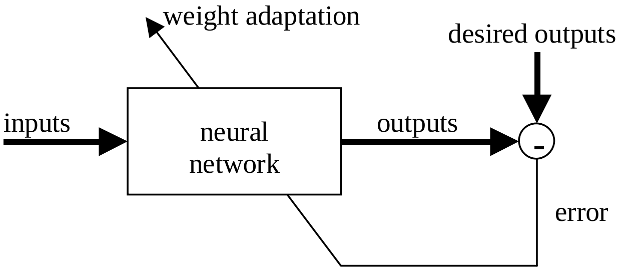

Figure 3: supervised neural network

If the learning is carefully realised (this operation usually requires some tuning which is difficult to achieve in an automatic manner), all observations are "learned" by the network. In a subsequent phase (which is called generalisation), new inputs can be presented to the ANN which will calculate corresponding outputs. In our Figure 2 example, the dots are learned while the plain line represents possible generalisation.

As we already mentioned, overfitting can occur in neural networks, exactly as in many other function approximation procedures. It is possible to avoid (or to limit) overfitting by using only a part of the available observations for learning, and using the other part in order to test the generalisation property, but this goes beyond the scope of our discussion.

What makes neural networks different from other approximation techniques is the content of the black box called "neural network" in Figure 3. A widely known ANN is the multi-layer perceptron (MLP), sketched in Figure 4. Circles represent computing units ("neurons"), which implement a non-linear function of the sum their inputs. Each arrow represents a connection between the output of a neuron A and the input of a neuron B, and is associated to a parameter (weight) which is multiplied by the output of neuron A before entering neuron B. Each neuron in a layer is connected to all neurons in the next layer.

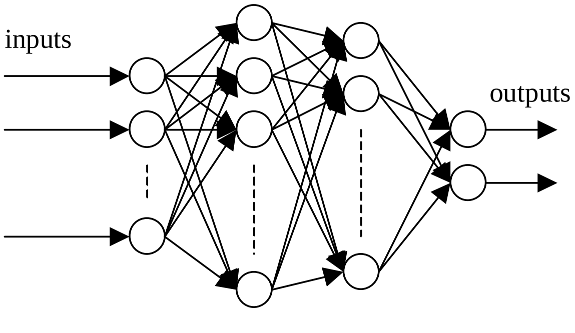

Figure 4: Multi-Layer Perceptron

The "model" is characterised by the number of inputs and the number of outputs (dictated by the problem), the number of layers, the number of neurons in each hidden layer (hidden layers are those which are neither connected to the inputs neither to the outputs, two in our example), and also by the non-linear function implemented in the neurons (some variations around the sigmoid can exist). The parameters of the model are the weights associated to each synapse, and also in some case supplementary parameters in the non-linear functions.

Neural networks are also characterized by a learning rule, i.e. a way to modify the weights according to observations (input-output pairs), in order to make the network slowly better approximate each of the known data.

MLPs have the "universal approximation" property: under weak conditions, MLPs are able to approximate any function from R^n to R^p, with an unlimited precision, provided that there are enough neurons in the hidden layers. This property makes the success of MLPs: in theory, any task formulated as an approximation problem can be solved…

Some comments however limit the practical use of MLP:

- the universal approximation property does not indicate how many neurons are needed in the hidden layers;
- two hidden layers are enough to ensure the universal approximation property, but in some cases experience shows than less neurons in more layers could be a better solution;
- the universal approximation property does not indicate how to compute the weights (i.e. the learning rule).

It should be noticed that learning is really a complicated task. Learning in MLP is an optimisation procedure, which can be stuck in local minima, which is not guaranteed to convergence in practical situations,… Nevertheless, despite these limitations, efficient learning rules have been proposed in the literature, and the MLP is widely used in many various application areas where some kind of approximation is needed. MLPs can also be used in classification tasks; see below for a brief review of possible neural network applications.

There exist many other supervised neural networks, devoted to approximation and classification tasks (radial-basis function networks, learning vector quantization, adaptive resonance theory,…). They differ from MLP by the architecture of the networks, but also by their performances in specific situations. One of the today's weak points of the neural network research is certainly to know which kind of network is best adapted to a class of problems.

Unsupervised networks

A radically different class of ANNs are the unsupervised networks. Figure 5 shows that weights are adapted in unsupervised networks without using any external information about the quality of outputs (without "teacher").

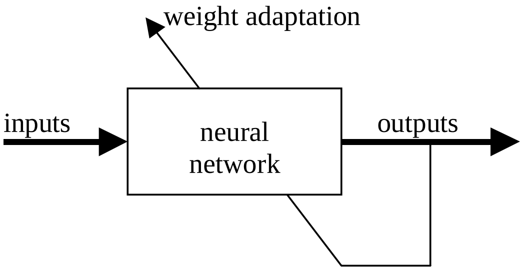

Figure 5: unsupervised neural network

Unsupervised learning is not a well-defined task: usually no criterion used to evaluate the quality of learning with respect to a consign (or at least the criterions are less intuitive). However, unsupervised networks were found to have computational capabilities that can be used in many applications too.

We will illustrate the concept of unsupervised networks through an example, the Kohonen self-organizing map (Kohonen 1997).

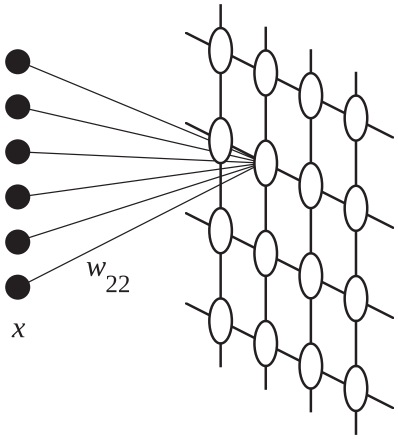

Figure 6: Kohonen's self-organizing feature map

Figure 6 illustrates the basic concept of Kohonen maps. Computational units (circles on the figure) are arranged on a (usually 2-dimensional) grid. The function implemented by each unit is a non-linear function of the weighed sum of its inputs. As in the MLP, each neuron is connected to each input, but unlike the MLP, it is also connected to other neurons in the layer (in fact its neighbours as shown in Figure 6). In order to understand how a Kohonen map works, let's go in some more details about the learning. The connections between the input and the neurons have adjustable weights, which will be set by the learning algorithm; lateral connections between neurons have fixed weights determined by the structure of the network, but which are not modified during learning. After a random initialisation of the weights,

1. an input vector x is presented to the network;
2. the neuron whose weight vector is "closest" from x (according to some distance measure) is selected;
3. the weights of this neuron and of its neighbours are adapted in order to be "closer" from the input vector x;
4. steps 1 to 3 are repeated several times for each input vector x.

No step in the learning algorithm requires any kind of knowledge about the quality of the outputs. But what kind of property could have such network if there is no "desired" behaviour? Actually, the Kohonen map, as other unsupervised networks, has an "emergent" property: it can be shown that "close" input vectors (again according to some distance measure) will activate neurons which are topologically close on the grid (a neuron is activated by an input when its weight is closest from the input than any other weight in the map). The Kohonen map thus realises some kind of projection from a high-dimensional space onto a lower dimensional (here 2-dimensional) space, respecting the topology between vectors; the projection is discrete since there is a finite number of neurons in the map.

The primary goal of Kohonen maps is not function approximation, but projection and clustering. By projection, we mean that data in a high-dimensional space can be projected (because of the topological property) on a 2-dimensional space; the projection is of course non-linear, which makes this network different from Principal Component Analysis for example. The clustering task consists in grouping together data which are similar, among a large database in a high-dimensional space. Examples of Kohonen maps applications will be given in the next section.

Neural network applications

ANNs being mostly developed by engineers, it is not surprising to find most of the ANN applications in the engineering field. Nevertheless, ANN models can be and are used in all fields where some kind of approximation or analysis has to be performed on data collected from unknown processes. Applications fields include medicine, physical sciences, economics, business, computer science, arts,…

Below is a list of application examples that can be found in (Fiesler et al., 1997) and (Kohonen 1997):

Supervised learning can be used for:

- classification of chromosomes for the diagnosis of genetic abnormality;
- intracardiac electrogram recognition in implantable cardioverter defibrillators;
- optimal robot trajectory planning;
- modeling of a polymerization reactor;
- control of telescope adaptive optics;
- prediction of financial time series.

Unsupervised learning can be used for:

- analysis of socio-economic situations;
- classification of rock samples to determine archaeological origin;
- parsing of linguistic expressions;
- appraisal of land value of shore parcels;
- pitch classification of musical notes.

These applications were not chosen to be representative of all fields where ANN can be used, but to show the diversity of domains which are not restricted to engineering sciences. We will detail one of these applications, the analysis of socio-economic situations (Blayo et al. 1991).

Non-linear dimension reduction is a typical task performed by supervised and unsupervised ANNs. When data are high-dimensional, such reduction can be interesting for two reasons; the first one is simply because low-dimensional data are easier to analyse by hand and to visualise, and the second is that the solution of problems in high-dimensional spaces (classification for example) usually requires an exceeding number of data (observations) to reach acceptable performances, while the same level of performances can be reached with much less data in lower-dimensional space (this is a consequence of the empty-space phenomenon, known in data analysis).

Our example consists in analysing the socio-economic situation of 52 countries, according to six variables: the annual increase, the infant mortality, the illiteracy ration, the school attendance, the GIP (gross internal product per inhabitant), and the annual GIP increase. Each country is thus represented by a point in a six-dimensional space, which is normalised in each direction in order to give the same importance to each variable.

Viewing or analysing points in a six-dimensional space is quite difficult. For this reason, a standard procedure is to use PCA (Principal Component Analysis) to project the six-dimensional space on a two-dimensional one. Projection means that some similarity criterion should be respected, i.e. that close vectors in the initial space will remain close in the resulting space. This condition is verified for PCA; however, PCA is a linear projection, and is thus able to cancel any linear relationship between variables, but not non-linear ones. Kohonen maps are able to capture non-linear relationships, the result being a better "unfolding" of the six-dimensional data in a two-dimensional plane. Figures 7 shows the six-dimensional database of 52 countries projected on a two-dimensional plane, respectively by the PCA (a) and the Kohonen maps (b) methods. Both have their advantages and drawbacks; nevertheless, it should be noticed that the most frequent use of this kind of projection is to facilitate a subsequent interpretation; Kohonen maps clearly outperform PCA in this context, because of the better unfolding which leads to a better repartition of countries on the map.

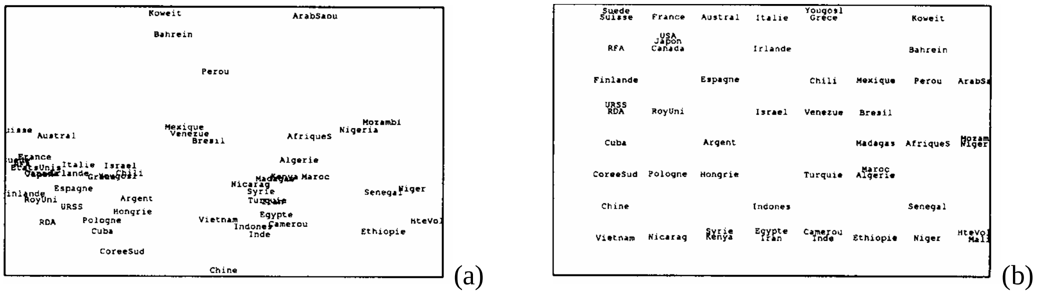

Figure 7: socioeconomic situation of 52 countries projected by PCA (a) and Kohonen maps (b). From (Blayo, 1991).

The explanatory power of ANNs

Neural networks have the reputation to behave as black boxes; this can furthermore be understood as a lack of explanatory power. We would like to comment these two arguments, and to point out some explanatory capabilities of ANNs.

ANN are usually considered as black boxes by reference to MLP, and more particularly to their hidden layers and connections. Obviously, it is quite difficult to interpret the value of hidden layers weights in MLP, because they form a part of the approximation process and of its optimisation procedure than can hardly be dissociated from the whole. Nevertheless, in our opinion, this does not mean that ANN are block boxes! First because MLP are far from being the only existing ANN model, and secondly because the explanatory power of ANN can be found elsewhere than in the weight values…

Explanatory power of models, and in particular of ANNs, can be seen at different levels. For the sake of simplicity, we will distinguish four levels, but some of them could be interleaved.

These levels are:

1. reality level: how a model could be used to interpret the reality hidden behind data?
2. data level: how a model could be used to interpret data (observations), whatever is the reality behind the data?
3. model level: does the structure of a model reveal some useful information about the reality or the data?
4. model's parameters level: does the value of the model's parameters (weights for example) give some useful information about the reality or the data?

The two first levels have a different nature than the two last ones. Reality and data level do not imply any understanding of the model or how it works; in other words, these levels are application or data-dependent, and the comments will be identical for an ANN model or other ones. On the contrary, model and model's parameters level depend on the mathematical tool used to analyse data, and raise more specific comments about ANNs.

1. Reality level

As mentioned above, reality can only be analysed through observations. Analysing the reality itself through any kind of model is thus just impossible. Nevertheless, it does not mean that comments about the reality could not be make, based on the data analysis realised by the model. For example, if a forecasting of the employment situation in a country shows that that number of unemployed persons will increase, it could mean that social charges are too high; nevertheless, this is a socioeconomic-only conclusion that is not related to the model, unlike the forecasting itself.

2. Data level

ANNs, like any other mathematical data analysis tool, use data in a way that is completely independent from the application. Selecting the data and transforming them in a way that is suitable to the model is preprocessing, and usually quite independent from the ANN (or other model). Data-level analysis is really where ANN can be explanatory: the purpose can be to fit data, to predict, to smooth, to detect outliers, to detect clusters,… ANNs are exactly as explicative as any other model at that level: they extract all information that is possible to extract from the data and present it in a way that is more suitable for further problem-level analysis.

3. Model level

The question of interpretability is more complex at this level. How a model itself, and not the results produced by the model, can help to interpret data mostly depends on the model's structure. MLPs are for example very different from Kohonen maps in that matter. MLP internal units and layers do not have any other meaning than being degrees-of-freedom added to the structure in order to reach a solution or a goal; the more complex the problem is, the more units and layers will be necessary. On the contrary, Kohonen maps units have an interpretation in the input space: since each unit is only influenced locally, it can be associated to a limited region of the input space which could, depending on the application (but also on the preprocessing), have some signification; for example, some units or groups of units in our socio-economic example (Figure 7 (b)) are associated to countries under development, some to capitalist ones,… Model-level interpretation is close from the application; however, unlike reality-level one, it uses information contained in the structure of the model, not in its computational results.

4. Parameters level

The reason why ANNs are often considered as non-explanatory black boxes is that the weights (the internal parameters) of some ANN models, like MLPs, are difficult to interpret. Although there are some attempts to use the informative contents of the weight values, it must be recognised that in most cases, such interpretation is quite basic (for example, sign or nullity of some weights can be interpreted in the same way as derivatives in the case of simple functions, but this is usually limited to weights connected to input or output units). Nevertheless, with other kinds of models, interpretation similar to model-level one is possible; in our socio-economic example (Figure 7 (b)), typical weights to units associated to a class of countries can be used as a reference, in order to compare other countries or to make other analyses. Without generality, parameters-level interpretation is usually easier with unsupervised models than with supervised ones; but some supervised models (like RBFN, Radial-Basis Function Networks) also permit some kind of interpretability, mostly for the same reasons of locality in the input space as in Kohonen maps.

Some comments and conclusions

Artificial neural networks are often known as black-boxes, whose results are difficult to interpret from the application point-of-view. We believe that this reputation is only justified at the level of the parameters value, for some ANN models. On the contrary, ANNs are powerful approximation and classification models, and their power can be exploited to analyse behaviours of data that are difficult to analyse with other techniques.

But the real question is to know what kind of explanation is needed from an ANN model. What is explanation? We believe that any kind of (model) interpretation becomes justified once the information provided by the interpretation is *useful*. The usefulness is obviously subjective and depends on the context; however, usefulness must also be measured at the right level. When a human brain analyses or resolves a problem, interpretability is of course also a key point. Nevertheless, no one has the idea to look at the synaptic coefficients or membrane potentials in the brain in order to get some information about the problem! Similarly, explanation from an ANN model should be mainly searched at the data level, where ANNs (and other similar techniques) excel. In addition to that, some knowledge can be extracted from the model and parameters level as detailed above, but this should be seen as a supplementary source of information, not as the principal one.

Other more technical aspects were not addressed here, although they are linked to knowledge and interpretability in neural networks. Using problem-related information to enhance the performances of an ANN model in one example. As shown by the MLP and Kohonen maps examples, neural nets work as closed models, trained through observations. It must be admitted that it is difficult to find context-independent methods to add other kind of information in a network, such as uncertainty about data, probability of "wrong" observations,… Common methods usually try to convert such information in a form that is more suitable for an ANN, such as resampling (Abu-Mostafa 1995) or quantization of data,…

A similar problem arises with non-numerical data (such as colors), or unordered ones (such as class labels). Using ANN-like methods with such data usually necessitates a preprocessing aimed at converting them into more usual numerical ordered data. This preprocessing has a computational goal only; it must be understood that the consequence of such artificial transformation of data is that the numerical or ordered character of the results cannot be used for any kind of interpretability! Unfortunately, this is not always verified in the literature…

ANN performances are also closely related to the definition of error criteria, as illustrated in Figure 3; inadequate criteria can lead to wrong interpretation of results! This is a crucial problem with ANN: criteria are very often based on some least-mean square (LMS) error, which is "natural" with linear approximation methods, but much less with non-linear ones; LMS is often used with non-linear models just because we don't have any better idea…

Artificial neural networks are computational models, and must be considered as such. They are aimed to model data; their explanatory power is thus related to the observations, and not to the reality behind them. In this context, interpretation and explanation is where ANN excel: more than modelling data, they are able to generalise on situations that were not "learned". Of course, generalisation is only successful when learning makes it possible, i.e. when the reality is "sufficiently" described by the observations. This is why reality and data should not be mixed up: explanation from a computational model is interesting when it is useful, i.e. when it gives information that is contained but hidden in the data.

References

Abu-Mostafa Y., 1995, "Hints", Neural Computation 7, 639-671.

Blayo F. and Demartines P., 1991, Data analysis: how to compare Kohonen neural networks to other techniques?, in: A. Preito (ed.), Artificial neural networks, Lecture Notes in Computer Science 540, 469-476.

Cottrell M., 1995, Introduction aux réseaux de neurones artificiels et spécificités des méthodes neuronales, in: E. de Bodt, E.-F. Henrion eds., Les réseaux de neurones en finance: conception et applications (D-Facto, Brussels), 23-58.

Fiesler E. and Baele R. eds., 1997, Handbook of neural computation (IOP and University Press).

MacKay D., 1995, Bayesian methods for supervised neural networks, in: M.A. Arbid (ed.), The handbook of brain theory and neural networks (MIT Press, Cambridge MA), 144-149.

McCulloch W.S. and Pitts W.H., 1943, A logical calculus of the ideas immanent in nervous activity, Bull. Math. Biophys. 5, 115-133.

Mouchart M., 1998, title, this book.

Werbos P., 1997, What is a neural network?, in: E. Fiesler, R. Baele eds., Handbook of neural computation (IOP and University Press), A2.2:1-A2.2:4.

Kohonen T., 1997, Self-organizing maps (Springer-Verlag, Berlin Heidelberg), 2nd edition.