Lecture course

Dynamic Neural Networks in Control Problems

Alexander Poznyak

CINVESTAV-México

Septiembre - December 2023

1. Artificial neural networks (ANN) and their variety.

1. Artificial neural networks (ANN) and their variety.
2. Activating functions: Threshold, Piecewise Linear, Sigmoid, Gaussian.

Plan of presentation

1. Artificial neural networks (ANN) and their variety.
2. Activating functions: Threshold, Piecewise Linear, Sigmoid, Gaussian.
3. Learning ability and Reinforcement Learning.

1. Artificial neural networks (ANN) and their variety.
2. Activating functions: Threshold, Piecewise Linear, Sigmoid, Gaussian.
3. Learning ability and Reinforcement Learning.
4. Why DNN are much more preferable compared to FFNN when the modelling of some dynamic process is required?

1. Artificial neural networks (ANN) and their variety.
2. Activating functions: Threshold, Piecewise Linear, Sigmoid, Gaussian.
3. Learning ability and Reinforcement Learning.
4. Why DNN are much more preferable compared to FFNN when the modelling of some dynamic process is required?
5. Some Limitations of ANN.

Main idea of the course

Central question under consideration:

Is it possible to control successfully system
without exact knowledge of their mathematical model?

The main idea of the course:

If yes, how to do that!

Definitions

Artificial intelligence (AI) is an area of computer science that emphasizes the creation of intelligent machines working and reacting like humans.

Some of computer activities withing AI include:

- Speech recognition,
- Learning,
- Planning,
- Problem solving.

- *Artificial intelligence* is a branch of computer science that aims to create intelligent machines. It has become an essential part of the technology industry.

- *Artificial intelligence* is a branch of computer science that aims to create intelligent machines. It has become an essential part of the technology industry.
- Research associated with artificial intelligence is highly technical and specialized. The core problems of artificial intelligence include programming computers for certain traits such as:
  - Knowledge, Reasoning,
  - Problem solving, Perception- Learning,
  - Planning, Ability to manipulate and move objects.

- **Knowledge engineering** is a core part of AI research. Machines can often act and react like humans only if they have abundant information relating to the world. *AI* must have access to objects, categories, properties and relations between all of them to implement knowledge engineering.

The core problems of artificial intelligence

Machine learning and robotics

- **Knowledge engineering** is a core part of AI research. Machines can often act and react like humans only if they have abundant information relating to the world. *AI* must have access to objects, categories, properties and relations between all of them to implement knowledge engineering.
- **Machine learning** is also a core part of AI. Mathematical analysis of machine learning algorithms is often referred to as *Computational Learning Theory*.

- **Knowledge engineering** is a core part of AI research. Machines can often act and react like humans only if they have abundant information relating to the world. *AI* must have access to objects, categories, properties and relations between all of them to implement knowledge engineering.
- **Machine learning** is also a core part of AI. Mathematical analysis of machine learning algorithms is often referred to as *Computational Learning Theory*.
- **Machine perception** deals with the capability to use sensory inputs to deduce the different aspects of the world, while *computer vision* is the power to analyze visual inputs with a few sub-problems such as facial, object and gesture recognition.

- **Knowledge engineering** is a core part of AI research. Machines can often act and react like humans only if they have abundant information relating to the world. *AI* must have access to objects, categories, properties and relations between all of them to implement knowledge engineering.
- **Machine learning** is also a core part of AI. Mathematical analysis of machine learning algorithms is often referred to as *Computational Learning Theory*.
- **Machine perception** deals with the capability to use sensory inputs to deduce the different aspects of the world, while *computer vision* is the power to analyze visual inputs with a few sub-problems such as facial, object and gesture recognition.
- **Robotics** is also a major field related to AI. Robots require intelligence to handle tasks such as object manipulation and navigation, along with sub-problems of localization, motion planning and mapping.

Definition

Artificial Neural Networks (ANN) are computational models inspired by biological neural networks, and are used to approximate functions that are generally unknown.

- The concept of ANN is basically introduced from the subject of biology where neural network plays a important and key role in human body.

Artificial neural networks as one of computational instruments in AI

ANN and neurones

Definition

Artificial Neural Networks (ANN) are computational models inspired by biological neural networks, and are used to approximate functions that are generally unknown.

- The concept of ANN is basically introduced from the subject of biology where neural network plays a important and key role in human body.
- In *human body work* is done with the help of *neural network*: it is just a web of inter connected neurons which are millions and millions in number. By interconnected neurons all the parallel processing is done in human body and the human body is the best example of parallel processing.

- Particularly, ANN are inspired by the behavior of neurons and the electrical signals they convey between input (such as from the eyes or nerve endings in the hand), processing, and output from the brain (such as reacting to light, touch, or heat).

- Particularly, ANN are inspired by the behavior of neurons and the electrical signals they convey between input (such as from the eyes or nerve endings in the hand), processing, and output from the brain (such as reacting to light, touch, or heat).
- Most ANN bear *only some resemblance* to their more complex biological counterparts, but are very effective at their intended tasks (e.g. classification or segmentation). Some ANNs are *adaptive systems* and are used for example to model populations and environments, which constantly change.

- Particularly, ANN are inspired by the behavior of neurons and the electrical signals they convey between input (such as from the eyes or nerve endings in the hand), processing, and output from the brain (such as reacting to light, touch, or heat).
- Most ANN bear *only some resemblance* to their more complex biological counterparts, but are very effective at their intended tasks (e.g. classification or segmentation). Some ANNs are *adaptive systems* and are used for example to model populations and environments, which constantly change.
- Neural networks can be *hardware*- (neurons are represented by physical components) or *software-based* (computer models), and can use a variety of topologies and learning algorithms.

There exist two types of ANNs:

- *Feedforward* NNs or FFNN,
- *Recurrent* (in discrete time) or *Differential* (in continuos time) neural networks.

Definition

Feedforward NN were the first and arguably most simple type of artificial neural network devised. In this network the information moves in only one direction-forward (see Fig.1): from the input nodes data goes through the hidden nodes (if any) and to the output nodes.

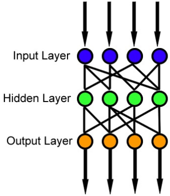


Figure 1: Feedforward ANN.

- *There are no cycles or loops in the network*. Feedforward networks (also known as Associative) can be constructed from different types of units, e.g. binary McCulloch-Pitts neurons, the simplest example being the perceptron.
- Continuous neurons, frequently with sigmoid activation functions, are used in the context of backpropagation of error.

Definition

Recurrent neural network (RNN), also known as **Auto Associative** or **Feedback Network**, is a class of artificial neural network, where connections between units form a directed cycle (Fig.2).

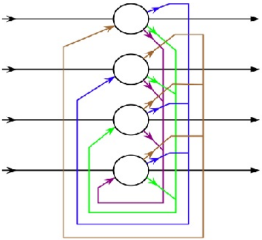


Figure 2: Feedback ANN of the Hopfield type.

- This creates an internal state of the network which allows it to exhibit **dynamic temporal behavior**.
- Unlike FFNN, **RNNs can use their internal memory** to process arbitrary sequences of inputs.
- In RNN the signal **travel in both forward and back the directions** by introducing loops in the network.

The main simple structures of RNN are

- the **Hopfield** *network*,
- the **Elman** *network*,
- the **Jordan** *network*.

- The **Hopfield network** is of historic interest although it is not a general RNN, as it is not designed to process sequences of patterns (see Fig.2).
- Instead it requires stationary inputs it is a RNN in which all connections are symmetric. Invented by John Hopfield in 1982, it guarantees that its **dynamics will converge**.
- If the connections are trained using Hebbian learning, then the Hopfield network can perform as **robust content-addressable memory, resistant to connection alteration**.
- A variation on the Hopfield network is the **bidirectional associative memory** (BAM). The BAM has two layers, either of which can be driven as an input, to recall an association and produce an output on the other layer.

The following special case of the basic architecture above was employed by Jeff Elman (1993).

A three-layer network is used (arranged horizontally as x, y, and z in the illustration (see Fig.3), with the addition of a set of "context units" m.

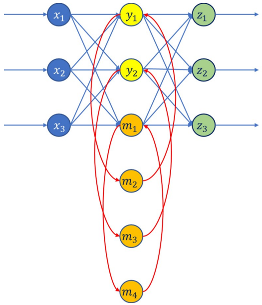


Figure 3: Elman recursive ANN.

- There are **connections from the middle (hidden) layer to these context units** fixed with a weight of one.
- At each time step, the **input is propagated in a standard feed-forward fashion**, and **then a learning rule is applied**.
- The **fixed back connections** result in the context units always **maintaining a copy of the previous values of the hidden units** (since they propagate over the connections before the learning rule is applied).
- Thus the network can maintain a sort of state, allowing it to perform such tasks as sequence-prediction that are beyond the power of **a standard multilayer perceptron**.

- **Jordan networks**, due to Michael I. Jordan (1996), are similar to Elman networks.
- The context units are however fed **from the output layer instead of the hidden layer**.
- The context units in a Jordan network are also referred to as the state layer, and have a **recurrent connection to themselves** with no other nodes on this connection.

The **mathematical model** of the Elman and Jordan networks (only some parameters are different) are governed by the following system of equations

$$\boxed{\left.\begin{array}{c} h_t = \sigma_h \left(W_h x_t + U_h y_{t-1} + b_h \right) \\ \\ y_t = \sigma_y \left(W_y h_t + b_y \right) \end{array}\right\}} \tag{1}$$

where

- x_t is input vector,
- h_t is hidden layer vector,
- y_t is output vector,
- W_h, U_h and b_h, b_y are weighting matrices and vectors,
- σ_h and σ_y are activating vector functions with the widely used components.

Activation functions are basically the transfer function which is output from a artificial neuron and it send signals to the other artificial neuron.

There are four form of Activation Functions $\sigma = \sigma(x)$: Threshold, Piecewise Linear, Sigmoid and Gaussian all are different from each other (see Table 1).

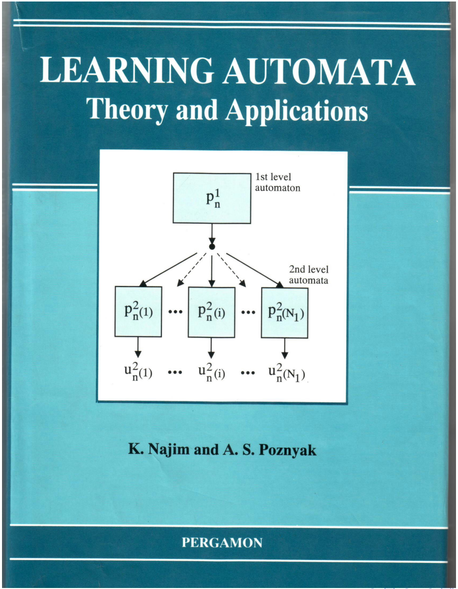
Threshold:	$\sigma_{\text{Th}}(x) = \frac{\sigma_{\max}}{2}\left(1 + \operatorname{sign}(x - x_0)\right)$
Piecewise Linear	$\sigma_{\text{PL}}(x) = \begin{cases} \sigma_{\max} & \text{if} \quad x > x_r \\ \sigma_{\max}\dfrac{x - x_l}{x_r - x_l} & \text{if} \quad x \in [x_l, x_r] \\ 0 & \text{if} \quad x < x_l \end{cases}$
Sigmoid	$\sigma_{\text{Sig}}(x) = \dfrac{\sigma_{\max}}{1 + \exp\{-\alpha x\}}, \alpha > 0$
Gaussian	$\sigma_{\text{Gauss}}(x) = \dfrac{1}{\sqrt{2\pi}\sigma_0}\exp\left\{-\dfrac{x^2}{2\sigma_0^2}\right\}$

Table 1: Activation functions.

In ANN most of the learning rules are used to develop models of processes, while adopting the network to the changing environment and discovering useful knowledge. These *Learning methods* are

- Supervised,
- Unsupervised,
- Reinforcement Learning.

One of the first publications on Learnning Procedures



Consider *Reinforcement Learning (RL) in Differential Neural Network (DNN)* which realizes a special tuning the weight matrix parameters providing the desired behavior of this DNN. In particular, in the DNN model given by

$$\boxed{\frac{d}{dt}\hat{x}(t) = A\hat{x}(t) + W_{\sigma}(t)\,\sigma(\hat{x}(t)) + W_{\varphi}(t)\,\varphi(\hat{x}(t))\,u(y(t))} \tag{2}$$

where

$\hat{x}(t) \in R^{n}$ is a vector of state estimates,

$\sigma(\hat{x}(t)) \in R^{k_{\sigma}}$, $\varphi(\hat{x}(t)) \in R^{k_{\varphi} \times m}$ are vector and matrix of activating functions,

$u(y(t), t) \in R^{m}$ is an external signal (or control action) depending on measurable system output $y(t)$.

- The **dynamic NN** model (2) may be treated as **an DNN-software sensor** or **a state observer** of some dynamic process $x(t) \in R^n$ which can not be measured directly on-line by different natural reasonings.
- The RL process consists in the realization of an adequate adjustment of weights matrix $W_\sigma(t)$ and $W_\varphi(t)$process, namely,

$$\boxed{\left.\begin{array}{c} \dot{W}_\sigma(t) = \Phi_\sigma(W_\sigma(t), \hat{x}(t), u(y(t))) \\ \\ \dot{W}_\varphi(t) = \Phi_\varphi(W_\varphi(t), \hat{x}(t), u(y(t))) \end{array}\right\}} \quad (3)$$

  in such a way that the current state estimates $\hat{x}(t)$ would be as closed as possible to the current real state $x(t)$ of the modelled real-live system.
- The **number of artificial neurons** is defined by the numbers $k_\sigma$and k_φ of activating functions components. In fact, these numbers define the complexity of the used DNN. The RL-rules (3) are specific for each considered problem.

To illustrate the advantages of DNN, compared to FFNN during the simulation of dynamic processes, let us consider the simplest 1-st order dynamic model given by

$$\boxed{\dot{x}_t = -ax_t + f_t,\, a > 0} \tag{4}$$

Applying the **Laplace transformation** $L\{\cdot\}$ to both parts of this equation we obtain

$$\boxed{X = \frac{1}{s+a}F = \frac{1}{s}\left(\frac{1}{1+a/s}\right)F = \frac{1}{s}\sum_{t=1}^{\infty}\left[\frac{(-1)^t}{t!}\left(\frac{a}{s}\right)^t\right]F = \left[\sum_{t=0}^{\infty}\frac{c_t}{s^{t+1}}\right]F}$$

where the operator $\dfrac{1}{s}$ represents the *simple operation of integration*, so that

$$X = L\{x_t\},\, sX = L\{\dot{x}_t\},\, F = L\{f_t\},\; c_t = (-1)^t\,\tfrac{a^t}{t!},\; L^{-1}\left\{\frac{1}{s}F\right\} = \int_{\tau=0}^{t} f_\tau d\tau$$

This implies the following *feedforward realization* of the solution x_t of the differential model (4):

$$x_t = \sum_{t=0}^{\infty} c_t \left[\int\limits_{\tau_{t+1}=0}^{t} \left(\cdots \int\limits_{\tau_2=0}^{\tau_3} \left(\int\limits_{\tau_1=0}^{\tau_2} f_{\tau_1} d\tau_1 \right) d\tau_2 \cdots \right) d\tau_{t+1} \right]. \tag{5}$$

The n **final-terms approximation** $\tilde{x}_{t,n}$ of (5) is

$$\boxed{\tilde{x}_{t,n} = \sum_{t=0}^{n} c_t \left[\int\limits_{\tau_{t+1}=0}^{t} \left(\cdots \int\limits_{\tau_2=0}^{\tau_3} \left(\int\limits_{\tau_1=0}^{\tau_2} f_{\tau_1} d\tau_1 \right) d\tau_2 \cdots \right) d\tau_{t+1} \right]} \tag{6}$$

The block-diagramm of (6) is depicted at Figure 3:

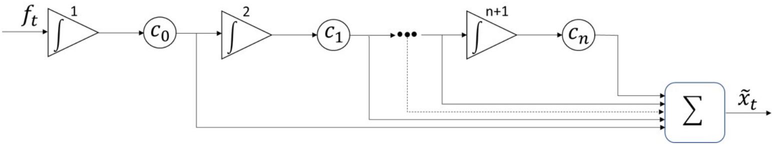


Figure 4: Block diagram of static realization of solution x_t.

Corollary

As one can see that here (in Figure 4) all signals move only **ahead** *(feedforward): no feedback involved! Notice also that for any finite number of integrators and amplifiers the signal* $\tilde{x}_{t,n}$ *is only the approximation of the process* x_t*. So, to have a "good" approximation it is required large enough number of basic elements (usually* $n \geq 10^4$*) that makes the realizing electronic device sufficiently complex!*

On the other hand, the block-diagramm, realizing the exact (non approximative) solution x_t of (4) with the use of only **one feedback**, is depicted at Figure 5:

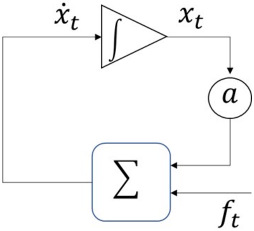


Figure 5: Direct feedback realization.

Reinforcement Learning (RL) in Differential Neural Network (DNN)

Why DNN are much more preferable compared to FFNN (6)

Corollary

Obviously, that the technical realization of the solution x_t of (4), using a feedback concept DNN, turns out to be **much simple** *compared with FFNN!*

In this technological era every has **Merits** and some **Demerits**: in others words, there is a Limitation with every system which makes the ANN technology weak in some points. Some **limitations of ANN** are as follows:

- ANN **is not a daily life** general purpose problem solver.
- There is **no structured methodology** available in ANN.
- There is **no single standardized paradigm** for ANN development.
- The **Output Quality** of an ANN may be **unpredictable**.
- Many ANN systems **does not describe how they solve problems**.
- **Black box** Nature.
- Greater **computational burden**.
- **Proneness to over fitting**.
- **Empirical nature of model development**.