Introduction to Neural Networks and Deep Learning

Introduction to Neural Networks

Andres Mendez-Vazquez

May 1, 2025

Outline

Outline

Cinvestav

What are Neural Networks? [1]

Basic Intuition

The human brain is a highly complex, nonlinear and parallel computer

What are Neural Networks? [1]

Basic Intuition

The human brain is a highly complex, nonlinear and parallel computer

It is organized as a

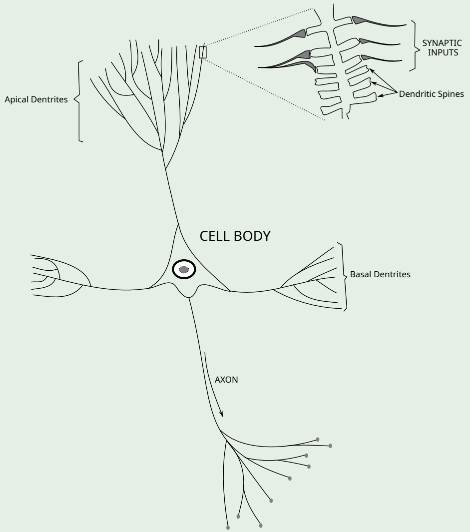
Network with (Ramon y Cajal 1911)

1. Basic Processing Units $\approx$ Neurons
2. Connections $\approx$ Axons and Dendrites

Outline

Example

The Neural Structure



Silicon Chip Vs Neurons

Speed Differential

1. Speeds in silicon chips are in the nanosecond range (10^{-9} s).
2. Speeds in human neural networks are in the millisecond range (10^{-3} s).

Silicon Chip Vs Neurons

Speed Differential

1. Speeds in silicon chips are in the nanosecond range (10^{-9} s).
2. Speeds in human neural networks are in the millisecond range (10^{-3} s).

However

We have massive parallelism on the human brain

1. 10 billion neurons in the human cortex.
2. 60 trillion synapses or connections

Silicon Chip Vs Neurons

Speed Differential

1. Speeds in silicon chips are in the nanosecond range (10^{-9} s).
2. Speeds in human neural networks are in the millisecond range (10^{-3} s).

However

We have massive parallelism on the human brain

1. 10 billion neurons in the human cortex.
2. 60 trillion synapses or connections

High Energy Efficiency

1. Human Brain uses 10^{-16} joules per operation.
2. Best computers use 10^{-6} joules per operation.

Outline

Cinvestav

Pigeon Experiment

Watanabe et al. 1995 [2]

Pigeons as art experts

Pigeon Experiment

Watanabe et al. 1995 [2]

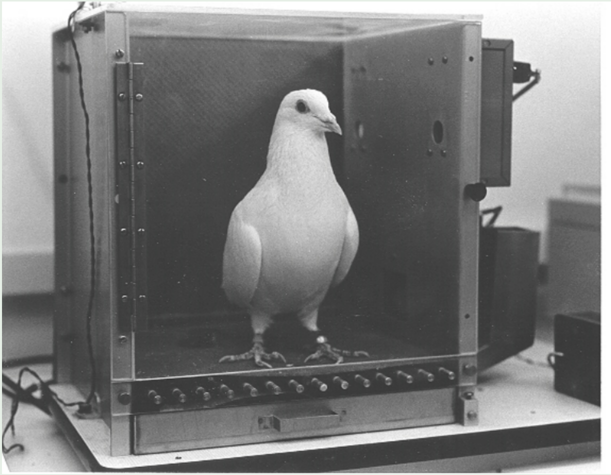
Pigeons as art experts

Experiment

- Pigeon is in a Skinner box
- Then, paintings of two different artists (e.g. Chagall / Van Gogh) are presented to it.
- A Reward is given for pecking when presented a particular artist (e.g. Van Gogh).

The Pigeon in the Skinner Box

Something like this



Results

Something Notable

- Pigeons were able to discriminate between Van Gogh and Chagall with 95% accuracy (when presented with pictures they had been trained on).

Results

Something Notable

- Pigeons were able to discriminate between Van Gogh and Chagall with 95% accuracy (when presented with pictures they had been trained on).
- Discrimination still 85% successful for previously unseen paintings of the artists.

Thus

1. Pigeons do not simply memorize the pictures.
2. They can extract and recognize patterns (the 'style').
3. They generalize from the already seen to make predictions.
4. This is what neural networks (biological and artificial) are good at (unlike conventional computer).

Outline

Formal Definition [1]

Definition

An **artificial neural network** is a massively parallel distributed processor made up of simple processing units. It resembles the brain in two respects:

1. Knowledge is acquired by the network from its environment through a learning process.
2. Inter-neuron connection strengths, known as synaptic weights, are used to store the acquired knowledge.

Inter-neuron connection strengths?

How do the neuron collect this information?

Some way to aggregate information needs to be devised...

Inter-neuron connection strengths?

How do the neuron collect this information?

Some way to aggregate information needs to be devised...

A Classic

Use a summation of product of weights by inputs!!!

Inter-neuron connection strengths?

How do the neuron collect this information?

Some way to aggregate information needs to be devised...

A Classic

Use a summation of product of weights by inputs!!!

Something like

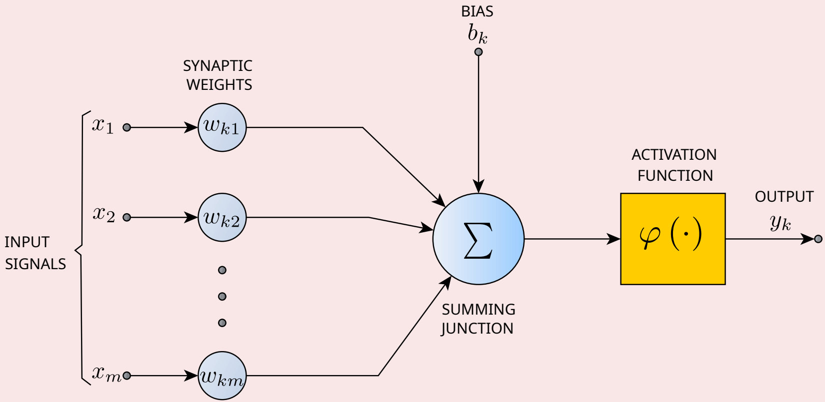
$$\sum_{i=1}^{m} w_i \times x_i$$

Where: w_i is the strength given to signal x_i

However: We still need a way to regulate this "aggregation" (Activation function)

The Model of a Artificial Neuron

Graphical Representation for neuron k



The use of Differential Equations in Neural Networks

It is not a well a known fact

- But the first proposed Neural Network was designed as combination of Differential Equations

Cinvestav

The use of Differential Equations in Neural Networks

It is not a well a known fact

- But the first proposed Neural Network was designed as combination of Differential Equations

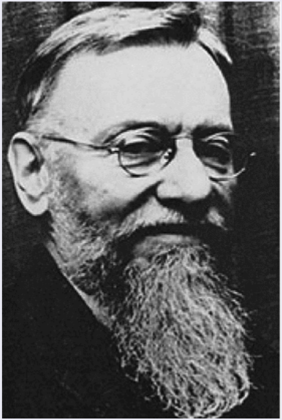
That McCulloch-Pitts model

- It is actually a discrete paraphrasing of such initial idea!!!

History

The study of Neurodynamics began in the 1930's

- With the work of **Nicolas Rashevsky** [3].



Nicolas Rashevsky

Who he was?

- American theoretical physicist who was one of the pioneers of mathematical biology.

Nicolas Rashevsky

Who he was?

- American theoretical physicist who was one of the pioneers of mathematical biology.

He published one of the first books in Mathematical Biophysics

- "Mathematical Biophysics: Physico-Mathematical Foundations of Biology."

Nicolas Rashevsky

Who he was?

- American theoretical physicist who was one of the pioneers of mathematical biology.

He published one of the first books in Mathematical Biophysics

- "Mathematical Biophysics: Physico-Mathematical Foundations of Biology."

And in 1933

- He proposed the first neural network architecture

Cinvestav

A simple Neural Network

Rashevsky proposed a Neural Network based in differential equations

$$
\begin{aligned}
\frac{de}{dt} =& A\boldsymbol{x}\left(t\right) - ae\\
\frac{dj}{dt} =& B\boldsymbol{x}\left(t\right) - bj\\
Output =& Heaviside\left(e - j - \theta\right)
\end{aligned}
$$

A simple Neural Network

Rashevsky proposed a Neural Network based in differential equations

$$\begin{aligned}\frac{de}{dt} =& A\boldsymbol{x}(t) - ae\\ \frac{dj}{dt} =& B\boldsymbol{x}(t) - bj\\ Output =& Heaviside(e - j - \theta)\end{aligned}$$

Something Notable

- Walter Pitts was his student, and together with Warren McCulloch rephrased the previous networks in a discrete version.

Into Big Data

He noticed something quite interesting [4]

- "in physics, one often averages over a large set of discrete events to obtain a continuous model"
  - This represents the large scale behavior of a system...

Into Big Data

He noticed something quite interesting [4]

- "in physics, one often averages over a large set of discrete events to obtain a continuous model"
  - This represents the large scale behavior of a system...

What do we do in Deep Learning with Big Data?

- **Our results are done over million of samples as training sets to get an average training!!!**

Outline

Cinvestav

Basic Elements of an Artificial Neuron (AN) Model

Set of Connecting links

- A signal x_j, at the input of synapse j connected to neuron k is multiplied by the synaptic weight w_{kj}.
- The weight may lie in a negative or positive range.
  - What about the real neuron? In classic literature you only have positive values.

Basic Elements of an Artificial Neuron (AN) Model

Set of Connecting links

- A signal x_j, at the input of synapse j connected to neuron k is multiplied by the synaptic weight w_{kj}.
- The weight may lie in a negative or positive range.
  - What about the real neuron? In classic literature you only have positive values.

A Complex Aggregation Function

- An aggregation function for the input signals, weighted by the respective synapses of the neuron.

Basic Elements of an Artificial Neuron (AN) Model

Set of Connecting links

- A signal x_j, at the input of synapse j connected to neuron k is multiplied by the synaptic weight w_{kj}.
- The weight may lie in a negative or positive range.
  - What about the real neuron? In classic literature you only have positive values.

A Complex Aggregation Function

- An aggregation function for the input signals, weighted by the respective synapses of the neuron.

Activation function (Squashing function)

- It limits the amplitude of the output of a neuron.
- It maps the permissible range of the output signal to an interval.

Outline

Cinvestav

Mathematically

Adder

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \tag{1}$$

Mathematically

Adder

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \tag{1}$$

1. $x_1, x_2, ..., x_m$ are the input signals.
2. $w_{k1}, w_{k2}, ..., w_{km}$ are the synaptic weights.
3. It is also known as "Affine Transformation."

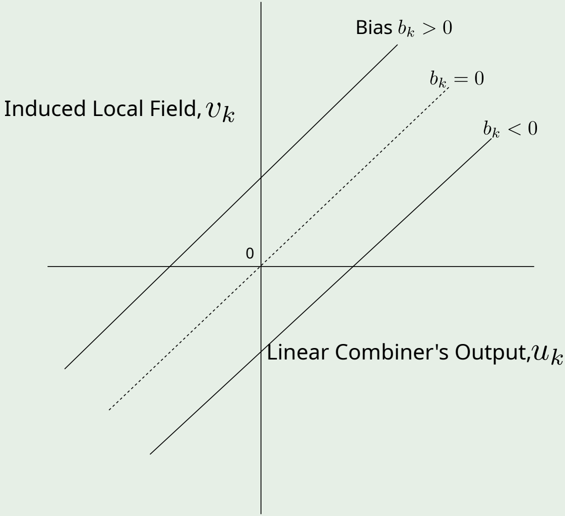
Activation function

$$y_k = \varphi\left(u_k + b_k\right) \tag{2}$$

1. y_k output of neuron.
2. φ is the activation function.

Integrating the Bias

The Affine Transformation



Thus

Final Equation

$$
\begin{aligned}
v_k &= \sum_{j=0}^{m} w_{kj} x_j \\
y_k &= \varphi\left(v_k\right)
\end{aligned}
$$

Cinvestav

Thus

Final Equation

$$v_k = \sum_{j=0}^{m} w_{kj} x_j$$
$$y_k = \varphi\left(v_k\right)$$

With

$x_0 = 1$ and $w_{k0} = b_k$

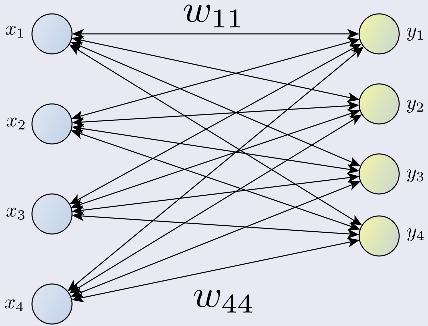
Cinvestav

Outline

Cinvestav

Energy Based Network

Bidirectional Associative Memory (BAM) [5]



A Little of Linear Algebra

Here, we can denote the weights as $n \times k$ matrix $\boldsymbol{W}$

- The n corresponds to the n dimensional vector $\boldsymbol{x}_0$
- The k corresponds to the k dimensional vector $\boldsymbol{y}_0$

A Little of Linear Algebra

Here, we can denote the weights as $n \times k$ matrix $\boldsymbol{W}$

- The n corresponds to the n dimensional vector $\boldsymbol{x}_0$
- The k corresponds to the k dimensional vector $\boldsymbol{y}_0$

Therefore the mapping is build in the following way given the feedback

$$
\begin{aligned}
\boldsymbol{y}_0 &= \text{sgn}\left(\boldsymbol{x}_0 \boldsymbol{W}\right) \\
\boldsymbol{x}_1^T &= \text{sgn}\left(\boldsymbol{W} \boldsymbol{y}_0\right) \\
\boldsymbol{y}_1 &= \text{sgn}\left(\boldsymbol{x}_1 \boldsymbol{W}\right) \\
\ldots
\end{aligned}
$$

Cinvestav

This is done until a stable state is reached

Meaning

$$\boldsymbol{y} = \text{sgn}\left(\boldsymbol{x}\boldsymbol{W}\right)$$
$$\boldsymbol{x}^T = \text{sgn}\left(\boldsymbol{W}\boldsymbol{y}\right)$$

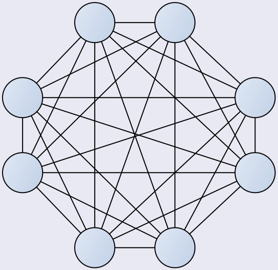
This is done until a stable state is reached

Meaning

$$\boldsymbol{y} = \text{sgn}\left(\boldsymbol{x}\boldsymbol{W}\right)$$
$$\boldsymbol{x}^T = \text{sgn}\left(\boldsymbol{W}\boldsymbol{y}\right)$$

A Notable Example

- The Hopfield Networks



Outline

Cinvestav

Types of Activation Functions I

Threshold Function

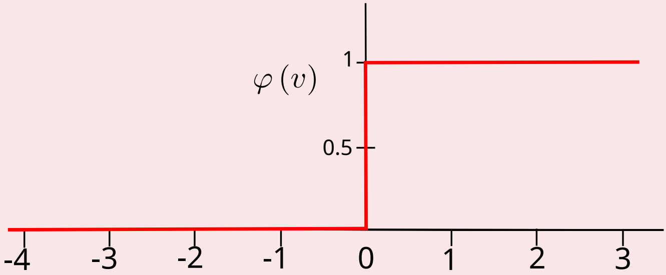
$$\varphi(v) = \begin{cases} 1 & \text{if } v \geq 0 \\ 0 & \text{if } v < 0 \end{cases} \text{ (Heaviside Function)} \quad (3)$$

Types of Activation Functions I

Threshold Function

$$\varphi(v) = \begin{cases} 1 & \text{if } v \geq 0 \\ 0 & \text{if } v < 0 \end{cases} \text{ (Heaviside Function)} \quad (3)$$

In the following picture



Thus

We can use this activation function

- To generate the first Neural Network Model

Thus

We can use this activation function

- To generate the first Neural Network Model

Clearly

- The model uses the summation as aggregation operator and a threshold function.

Outline

Cinvestav

McCulloch-Pitts model [6]

McCulloch-Pitts model (Pioneers of Neural Networks in the 1940's)

$$\text{Output } y_k = \begin{cases} 1 & \text{if } v_k \geq \theta \\ 0 & \text{if } v_k < \theta \end{cases} \tag{4}$$

McCulloch-Pitts model [6]

McCulloch-Pitts model (Pioneers of Neural Networks in the 1940's)

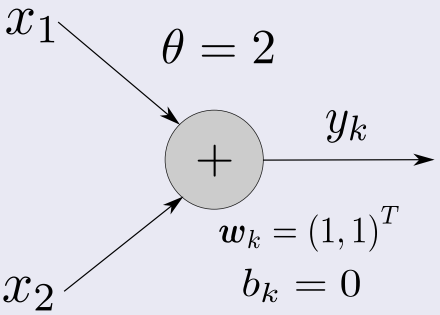
$$\text{Output } y_k = \begin{cases} 1 & \text{if } v_k \geq \theta \\ 0 & \text{if } v_k < \theta \end{cases} \tag{4}$$

with induced local field $\boldsymbol{w}_k = (1, 1)^T$

$$v_k = \sum_{j=1}^{m} w_{kj} x_j + b_k \tag{5}$$

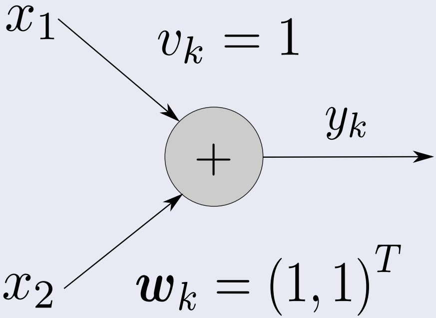
It is possible to do classic operations in Boolean Algebra

AND Gate



In the other hand

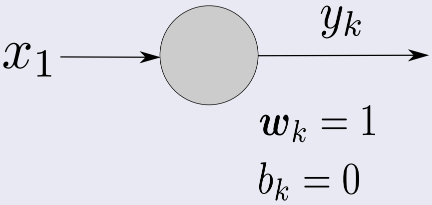
OR Gate



Finally

NOT Gate

$$y_k = \begin{cases} 1 & \text{if } v_k \leq \theta = 0 \\ 0 & \text{if } v_k > \theta = 0 \end{cases}$$



And the impact is further understood if you look at this paper

Claude Shannon

- "A Symbolic Analysis of Relay and Switching Circuits"
  - Shannon proved that his switching circuits could be used to simplify the arrangement of the electromechanical relays
  - These circuits could solve all problems that Boolean algebra could solve.

And the impact is further understood if you look at this paper

Claude Shannon

- "A Symbolic Analysis of Relay and Switching Circuits"
  - Shannon proved that his switching circuits could be used to simplify the arrangement of the electromechanical relays
  - These circuits could solve all problems that Boolean algebra could solve.

Basically, he proved that computer circuits

- They can solve computational complex problems... then neural networks can simulate them...

Outline

Cinvestav

More advanced activation function

Piecewise-Linear Function

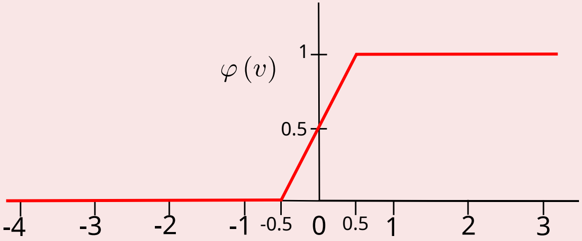
$$\varphi(v) = \begin{cases} 1 & \text{if } v_k \geq \frac{1}{2} \\ v & \text{if } -\frac{1}{2} < v_k < \frac{1}{2} \\ 0 & \text{if } v \leq -\frac{1}{2} \end{cases} \quad (6)$$

More advanced activation function

Piecewise-Linear Function

$$\varphi(v) = \begin{cases} 1 & \text{if } v_k \geq \frac{1}{2} \\ v & \text{if } -\frac{1}{2} < v_k < \frac{1}{2} \\ 0 & \text{if } v \leq -\frac{1}{2} \end{cases} \tag{6}$$

Example



Remarks

Notes about Piecewise-Linear function

The amplification factor inside the linear region of operation is assumed to be unity.

Remarks

Notes about Piecewise-Linear function

The amplification factor inside the linear region of operation is assumed to be unity.

Special Cases

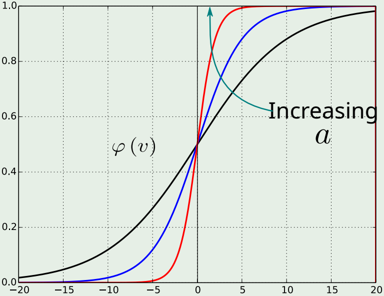
- A linear combiner arises if the linear region of operation is maintained without running into saturation.
- The piecewise-linear function reduces to a threshold function if the amplification factor of the linear region is made infinitely large.

A better choice!!!

Sigmoid function

$$\varphi(v) = \frac{1}{1 + \exp\{-av\}} \tag{7}$$

Where a is a slope parameter.



Outline

Cinvestav

The Problem of the Vanishing Gradient

When using a non-linearity

- However, there is a drawback when using Back-Propagation (As we saw in Machine Learning) under a sigmoid function

$$s(x) = \frac{1}{1+e^{-x}}$$

The Problem of the Vanishing Gradient

When using a non-linearity

- However, there is a drawback when using Back-Propagation (As we saw in Machine Learning) under a sigmoid function

$$s(x) = \frac{1}{1 + e^{-x}}$$

Because if we imagine a Deep Neural Network as a series of layer functions f_i

$$y(A) = f_t \circ f_{t-1} \circ \cdots \circ f_2 \circ f_1(A)$$

- With f_t is the last layer.

Cinvestav

Then, using the Chain Rule

Therefore, we finish with a sequence of derivatives

$$\frac{\partial y\left(A\right)}{\partial w_{1i}}=\frac{\partial f_t\left(f_{t-1}\right)}{\partial f_{t-1}}\cdot\frac{\partial f_{t-1}\left(f_{t-2}\right)}{\partial f_{t-2}}\cdot\cdots\cdot\frac{\partial f_2\left(f_1\right)}{\partial f_2}\cdot\frac{\partial f_1\left(A\right)}{\partial w_{1i}}$$

Therefore

Given the commutativity of the product

- You could put together the derivative of the sigmoid's

$$f(x) = \frac{ds(x)}{dx} = \frac{e^{-x}}{\left(1 + e^{-x}\right)^2}$$

Therefore

Given the commutativity of the product

- You could put together the derivative of the sigmoid's

$$f(x) = \frac{ds(x)}{dx} = \frac{e^{-x}}{(1+e^{-x})^2}$$

Therefore, deriving again

$$\frac{df(x)}{dx} = -\frac{e^{-x}}{(1+e^{-x})^2} + \frac{2(e^{-x})^2}{(1+e^{-x})^3}$$

Cinvestav

Therefore

Given the commutativity of the product

- You could put together the derivative of the sigmoid's

$$f(x) = \frac{ds(x)}{dx} = \frac{e^{-x}}{\left(1+e^{-x}\right)^2}$$

Therefore, deriving again

$$\frac{df(x)}{dx} = -\frac{e^{-x}}{\left(1+e^{-x}\right)^2} + \frac{2\left(e^{-x}\right)^2}{\left(1+e^{-x}\right)^3}$$

After making $\frac{df(x)}{dx} = 0$

- We have the maximum is at $x = 0$

Cinvestav

Therefore

The maximum for the derivative of the sigmoid

- $f(0) = 0.25$

Therefore

The maximum for the derivative of the sigmoid

- $f(0) = 0.25$

Therefore, Given a **Deep** Convolutional Network

- We could finish with

$$\lim_{k \to \infty} \left(\frac{ds(x)}{dx} \right)^k = \lim_{k \to \infty} (0.25)^k \to 0$$

Cinvestav

Therefore

The maximum for the derivative of the sigmoid

- $f(0) = 0.25$

Therefore, Given a **Deep** Convolutional Network

- We could finish with

$$\lim_{k \to \infty} \left(\frac{ds(x)}{dx} \right)^k = \lim_{k \to \infty} (0.25)^k \to 0$$

A Vanishing Derivative or Vanishing Gradient

- Making quite difficult to do train a deeper network using this activation function for Deep Learning and even in Shallow Learning

Outline

Cinvestav

Thus

The need to introduce a new function

$$f(x) = x^{+} = \max(0, x)$$

Thus

The need to introduce a new function

$$f\left(x\right)=x^{+}=\max\left(0,x\right)$$

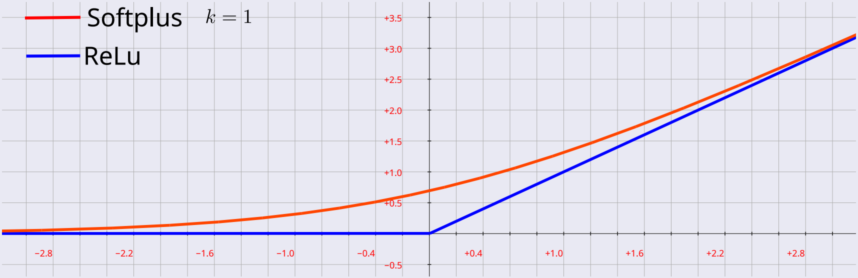
It is called ReLu or Rectifier

With a smooth approximation (Softplus function)

$$f\left(x\right)=\frac{\ln\left(1+e^{kx}\right)}{k}$$

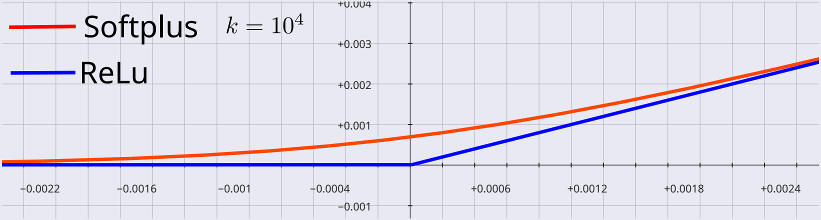
We have

When $k = 1$



Increase k

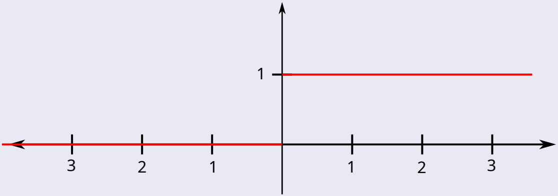
When $k = 10^4$



However, it seems to be

People are using the following derivative

$$\frac{d}{dx}ReLu\left(x\right)=\begin{cases}0 & x<0\\1 & x\geq 0\end{cases}$$

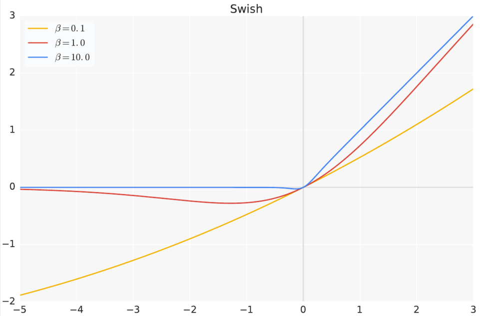


Example at Google Brain

"SWISH: A SELF-GATED ACTIVATION FUNCTION" [7]

$$S(x) = \frac{x}{1 + \exp\{-\beta x\}}$$

- Here β is a trainable parameter



Some Properties of the Swish

If $\beta = 1$

- We have the Sigmoid-weighted Linear Unit (SiL), proposed in reinforcement learning

Cinvestav

Some Properties of the Swish

If $\beta = 1$

- We have the Sigmoid-weighted Linear Unit (SiL), proposed in reinforcement learning

As $\beta \to \infty$

- The sigmoid component approaches a 0-1 function, Then what?

Some Properties of the Swish

If $\beta = 1$

- We have the Sigmoid-weighted Linear Unit (SiL), proposed in reinforcement learning

As $\beta \to \infty$

- The sigmoid component approaches a 0-1 function, Then what?

When $\beta = 0$

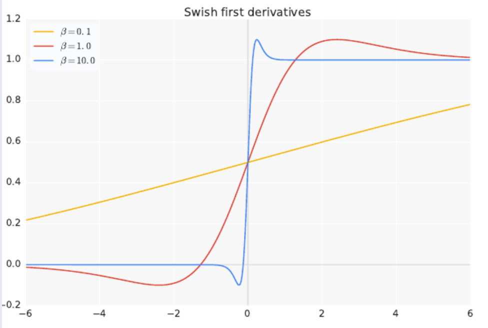
$$S(x) = \frac{x}{2}$$

Cinvestav

Thus

Swish interpolate Between the linear function and the ReLU function

- Not only that but at the derivatives



However

We need to analyze more activation functions

- So, we reach the objective of finding one that has smooth derivatives.

However

We need to analyze more activation functions

- So, we reach the objective of finding one that has smooth derivatives.

And Several Derivatives

- It is going to be nice...

Final Remarks

Although, ReLu functions

- They can handle the problem of vanishing problem

Final Remarks

Although, ReLu functions

- They can handle the problem of vanishing problem

However, as we will see, saturation starts to appear as a problem

- As in Hebbian Learning!!!

Outline

Cinvestav

Neural Network As a Graph [1]

Definition

A neural network is a directed graph consisting of nodes with interconnecting synaptic and activation links.

Neural Network As a Graph [1]

Definition

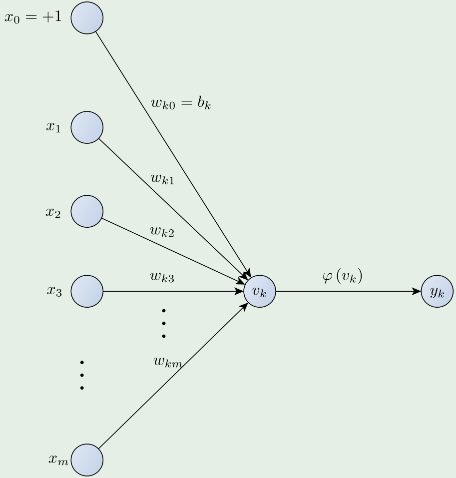
A neural network is a directed graph consisting of nodes with interconnecting synaptic and activation links.

Properties

1. Each neuron is represented by an function
2. Each link represent a weight.
3. The weighted sum of the input signals defines the local field.
4. The activation function maps local field to an output.

Example

Simple Perceptron



Some Observations

Observation

- A partially complete graph describing a neural architecture has the following characteristics:
  - Source nodes supply input signals to the graph.
  - Each neuron is represented by a single node called a computation node.
  - The communication links provide directions of signal flow in the graph.

Some Observations

Observation

- A partially complete graph describing a neural architecture has the following characteristics:
  - Source nodes supply input signals to the graph.
  - Each neuron is represented by a single node called a computation node.
  - The communication links provide directions of signal flow in the graph.

However

Other Representations exist!!!

Some Observations

Observation

- A partially complete graph describing a neural architecture has the following characteristics:
  - Source nodes supply input signals to the graph.
  - Each neuron is represented by a single node called a computation node.
  - The communication links provide directions of signal flow in the graph.

However

Other Representations exist!!!

Three main representations ones

- Block diagram, providing a functional description of the network.
- Signal-flow graph, providing a complete description of signal flow in the network.
  - Then one we plan to use.
- Architectural graph, describing the network layout.

Outline

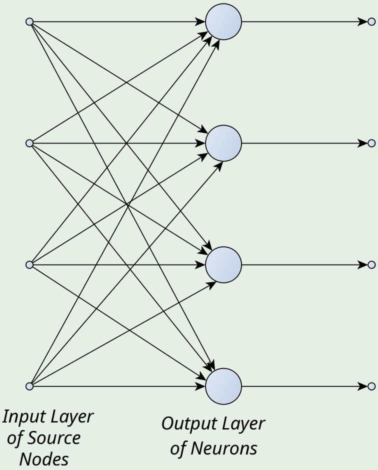
Cinvestav

Outline

Cinvestav

Single-Layer Feedforward Networks

We begin with something quite simple



Observations

Observations

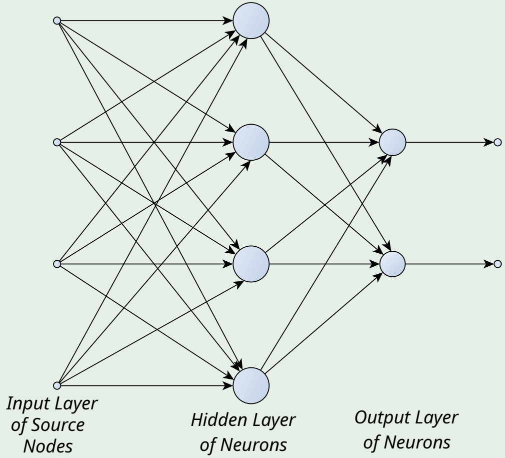
This network is know as a strictly feed-forward or acyclic type.

Outline

Cinvestav

Multilayer Feedforward Networks

Stacking Layers



Observations

Observations

1. This network contains a series of hidden layer.

Observations

Observations

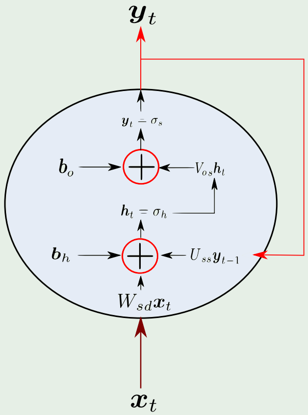
1. This network contains a series of hidden layer.
2. Each hidden layers allows for classification of the new output space of the previous hidden layer.

Outline

Cinvestav

Recurrent Networks

Connecting the back with the front



Observations

Observations

1. This network has not self-feedback loops.

Observations

Observations

1. This network has not self-feedback loops.
2. It has something known as unit delay operator $B = z^{-1}$.

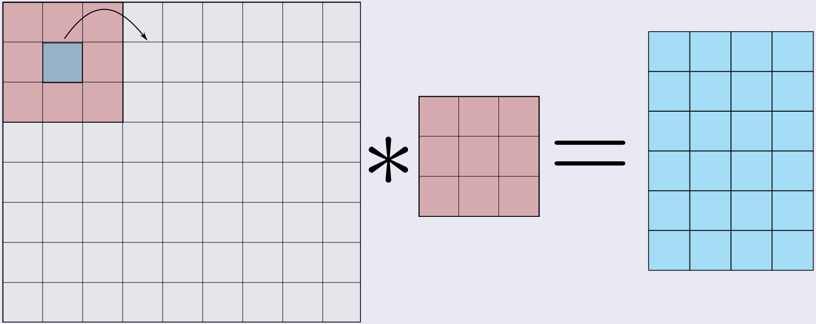
Outline

Cinvestav

Convolutional Deep Learners

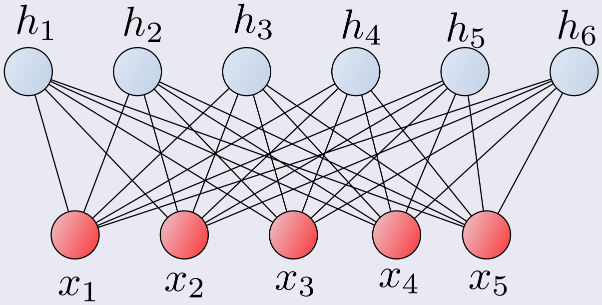
Using the principle of locality [8, 9]

Horizontal Stride $r = 2$



Restricted Boltzmann Machines

Energy Based Architectures [10]



Outline

Cinvestav

Knowledge Representation

Definition

- By Fischler and Firschein, 1987
  - “Knowledge refers to stored information or models used by a person or machine to interpret, predict, and appropriately respond to the outside world. ”

Knowledge Representation

Definition

- By Fischler and Firschein, 1987
  - "Knowledge refers to stored information or models used by a person or machine to interpret, predict, and appropriately respond to the outside world. "

It consists of two kinds of information:

1. The known world state, represented by facts about what is and what has been known.
2. Observations (measurements) of the world, obtained by means of sensors.

Knowledge Representation

Definition

- By Fischler and Firschein, 1987
  - "Knowledge refers to stored information or models used by a person or machine to interpret, predict, and appropriately respond to the outside world. "

It consists of two kinds of information:

1. The known world state, represented by facts about what is and what has been known.
2. Observations (measurements) of the world, obtained by means of sensors.

Observations can be

1. Labeled
2. Unlabeled

Outline

Set of Training Data

Training Data

- It consist of input-output pairs (x, y)
  - $x=$ input signal
  - $y=$ desired output

Set of Training Data

Training Data

- It consist of input-output pairs (x, y)
  - $x=$ input signal
  - $y=$ desired output

Thus, we have the following phases of designing a Neuronal Network

1. Choose appropriate architecture
2. Train the network - learning.
   1. Use the Training Data!!!
3. Test the network with data not seen before
   1. Use a set of pairs that where not shown to the network so the y component is guessed.
4. Then, you can see how well the network behaves - Generalization Phase.

Outline

Cinvestav

Representing Knowledge in a Neural Networks

Notice the Following

The subject of knowledge representation inside an artificial network is very complicated.

Representing Knowledge in a Neural Networks

Notice the Following

The subject of knowledge representation inside an artificial network is very complicated.

However: Pattern Classifiers Vs Neural Networks

1. Pattern Classifiers are first designed and then validated by the environment.
2. Neural Networks learns the environment by using the data from it!!!
   1. However, they are even designed!!!

Representing Knowledge in a Neural Networks

Notice the Following

The subject of knowledge representation inside an artificial network is very complicated.

However: Pattern Classifiers Vs Neural Networks

1. Pattern Classifiers are first designed and then validated by the environment.
2. Neural Networks learns the environment by using the data from it!!!
   1. However, they are even designed!!!

Kurt Hornik et al. proved (1989)

"Standard multilayer feedforward networks with as few as one hidden layer using arbitrary squashing functions are capable of approximating any **Borel measurable function**" (Basically many of the known ones!!!)

Rules Knowledge Representation

Rule 1

- Similar inputs from similar classes should usually produce similar representation.
  - We can use a Metric to measure that similarity!!!

Rules Knowledge Representation

Rule 1

- Similar inputs from similar classes should usually produce similar representation.
  - We can use a Metric to measure that similarity!!!

Examples

1. $d\left(\boldsymbol{x_i}, \boldsymbol{x_j}\right) = \left\|\boldsymbol{x_i} - \boldsymbol{x_j}\right\|$ (Classic Euclidean Metric).
2. $d_{ij}^2 = \left(\boldsymbol{x}_i - \boldsymbol{\mu}_i\right)^T \sum^{-1} \left(\boldsymbol{x}_j - \boldsymbol{\mu}_j\right)$ (Mahalanobis distance) where
   1. $\boldsymbol{\mu}_i = E\left[\boldsymbol{x}_i\right]$.
   2. $\sum = E\left[\left(\boldsymbol{x}_i - \boldsymbol{\mu}_i\right)\left(\boldsymbol{x}_i - \boldsymbol{\mu}_i\right)^T\right] = E\left[\left(\boldsymbol{x}_j - \boldsymbol{\mu}_j\right)\left(\boldsymbol{x}_j - \boldsymbol{\mu}_j\right)^T\right]$.

More

Rule 2

- Items to be categorized as separate classes should be given widely different representations in the network.

More

Rule 2

- Items to be categorized as separate classes should be given widely different representations in the network.

Rule 3

- If a particular feature is important, then there should be a large number of neurons involved in the representation.

More

Rule 2

- Items to be categorized as separate classes should be given widely different representations in the network.

Rule 3

- If a particular feature is important, then there should be a large number of neurons involved in the representation.

Rule 4

- Prior information and invariance should be built into the design:
  - Thus, simplify the network by not learning that data.

[1] S. Haykin, *Neural Networks and Learning Machines*.
No. v. 10 in Neural networks and learning machines, Prentice Hall, 2009.

[2] S. Watanabe, J. Sakamoto, and M. Wakita, "Pigeons'discrimination of paintings by monet and picasso," *Journal of the experimental analysis of behavior*, vol. 63, no. 2, pp. 165–174, 1995.

[3] N. Rashevsky *et al.*, "Mathematical biophysics," 1938.

[4] P. Cull, "The mathematical biophysics of nicolas rashevsky," *Biosystems*, vol. 88, no. 3, pp. 178–184, 2007.

[5] R. Rojas, *Neural networks: a systematic introduction*.
Springer Science & Business Media, 2013.

[6] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

Cinvestav

[7] P. Ramachandran, B. Zoph, and Q. Le, "Swish: a self-gated activation function," 10 2017.

[8] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[9] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[10] F. Higuera and J. Jimenez, "Boltzmann approach to lattice gas simulations," *EPL (Europhysics Letters)*, vol. 9, no. 7, p. 663, 1989.

Cinvestav