U3 DEEP LEARNING AND NEURAL NETWORKS

U3.E4 NEURAL NETWORKS TOPOLOGY

Artificial Intelligence Technician

March 2021, Version 1

Co-funded by the Erasmus+ Programme of the European Union

The Development and Research on Innovative Vocational Educational Skills project (DRIVES) is co-funded by the Erasmus+ Programme of the European Union under the agreement 591988-EPP-1-2017-1-CZ-EPPKA2-SSA-B. The European Commission support for the production of this publication does not constitute endorsement of the contents which reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

LEARNING OBJECTIVES

The student is able to

AIT.U3.E4.PC1	Understand the key parameters in a neural network's architecture.
AIT.U3.E4.PC2	Knows some of the most common neural networks like perceptron, multi-layer perceptron, feed-forward, backpropagation, etc.
AIT.U3.E4.PC3	Understand the main differences between those types of architectures.
AIT.U3.E4.PC4	Know the different use cases of each architecture.
AIT.U3.E4.PC5	Critically select the architecture that best fits a specific problem or situation.

ARTIFICIAL NEURAL NETWORK ARCHITECTURE

- Artificial Neural Networks (ANNs) are commonly refered as **Neural Networks** (NNs).

- NNs are made up of many artificial neurons.

- **One** neuron can perform a **simple decision**.

- Many **connected** neurons can make more **complex decisions**.

- A neuron can have **any number of inputs** from **one** to ***n***, where *n* is the **total number of inputs**.

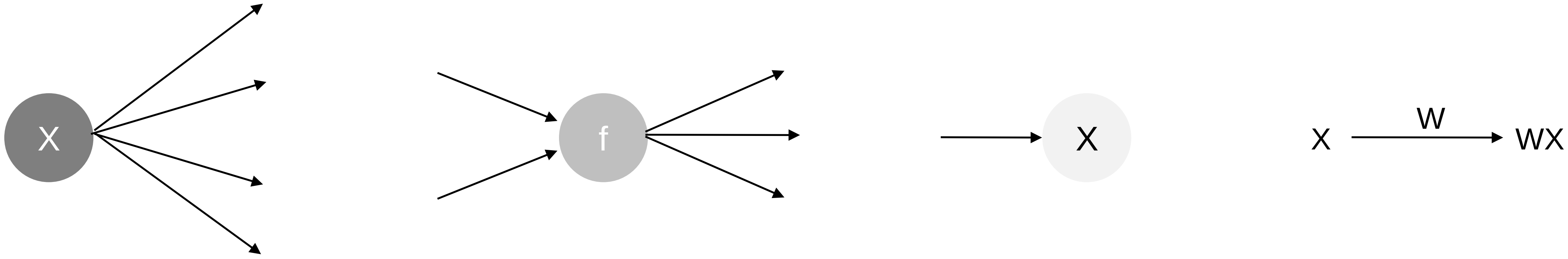
- **Each input** into the neuron has its own **weight associated** with it. A weight is nothing more than a floating point number that is adjusted during network training.

ANNs simulate neurons structures in software, basicaly they are **‘digital versions’ of neurons**, synapses, and connection strengths.

By feeding training examples (“experience”) to an ANN and adjusting their weights accordingly, an ANN **learns complex functions** much like a biological brain.

ARTIFICIAL NEURAL NETWORK ARCHITECTURE

Mathematically, a **network** is represented by a **weighted**, **directed** <u>graph</u> with the following elements:



Input Node **Computing Node** **Output Node** **Edge/Link**

The <u>**nodes**</u> of a network are either input variables, computational elements, or output variables.

Input Node

An input variable is determined independently of the network (it is an exogenous variable).

Computing Node

A computational element is a function of input variables to which it is connected by incoming links, and its unique output value is disseminated by outgoing links. These outgoing links may be to output nodes or to the inputs of other computational elements.

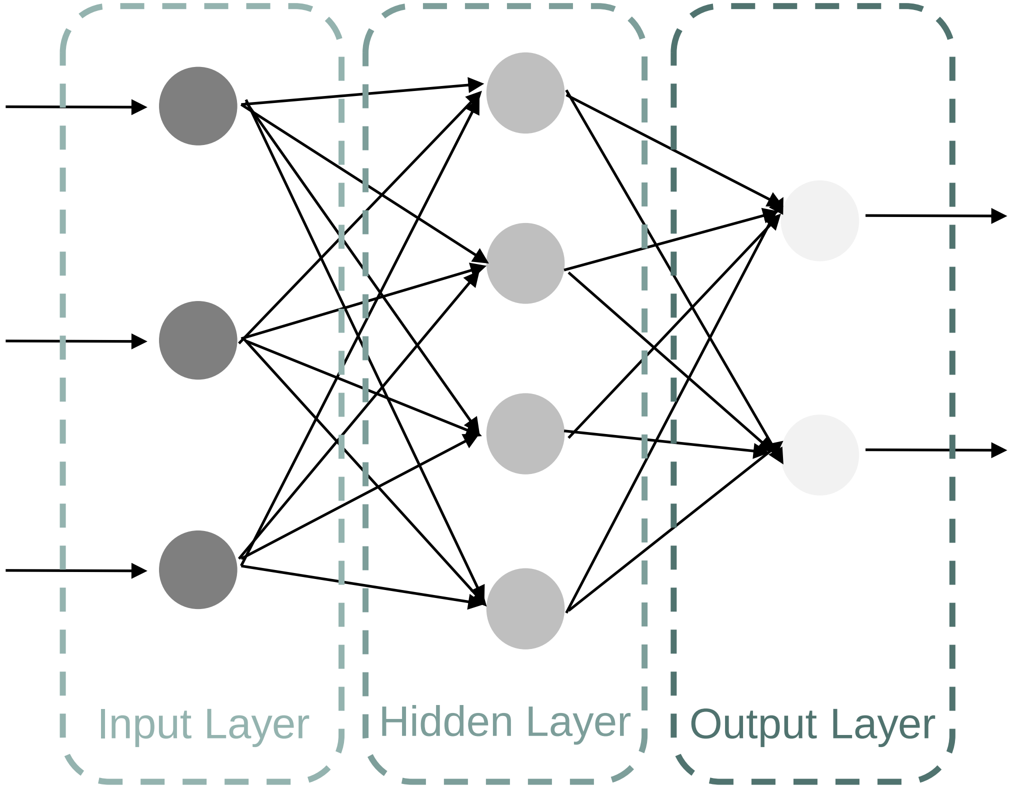
Output Node

Output variables are either certain input variables (a degenerate case) or the responses of selected computational elements, i.e., the value established by a linear node performing a weighted summation of its inputs.

Edge/Link

The real-valued signal variable x at the input end is directed to the other end and established at the value wx.

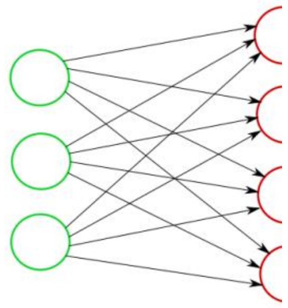
A **simple NN** includes an **input layer**, a **hidden layer** and an **output (or target) layer**.



The layers are connected through nodes and these connections form a "network" of interconnected nodes.

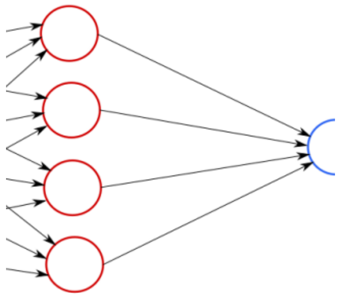
DRIVES
Development and Research on Innovative Vocational Education Skills

Input Layer



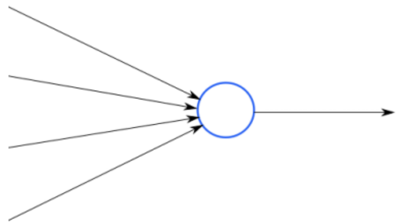
The **input layer** is composed of artificial **input** neurons and brings the initial data into the system for further processing by subsequent **layers** of artificial neurons.

Hidden Layer



Hidden layers, simply put, are **layers** of mathematical functions each designed to produce an output specific to an intended result.

Output Layer



The **output layer** is responsible for producing the result. There must always be one **output layer** in a neural network.

Perceptron was introduced by Frank Rosenblatt in 1957.

It is the basic unit of a neural network (an artificial neuron).

It is a single layer binary linear classifier commonly used to classify the data into two parts.

It is used in supervised learning.

A linear decision boundary is drawn, allowing the distinction between the two linearly separable classes. If the sum of the input signals exceeds a certain threshold, it outputs a signal; otherwise, there is no output.

Perceptron consists of **five** parts:

1. N **inputs**, $x_1 \dots x_n$
2. **Weights** for each input, $w_1 \dots w_n$
3. A **bias** input x_0 (constant) and associated weight w_0
4. **Weighted sum of inputs**, $y = w_0x_0 + w_1x_1 + \dots + w_nx_n$
5. A **threshold** or **activation function**

   1 if $y > t$

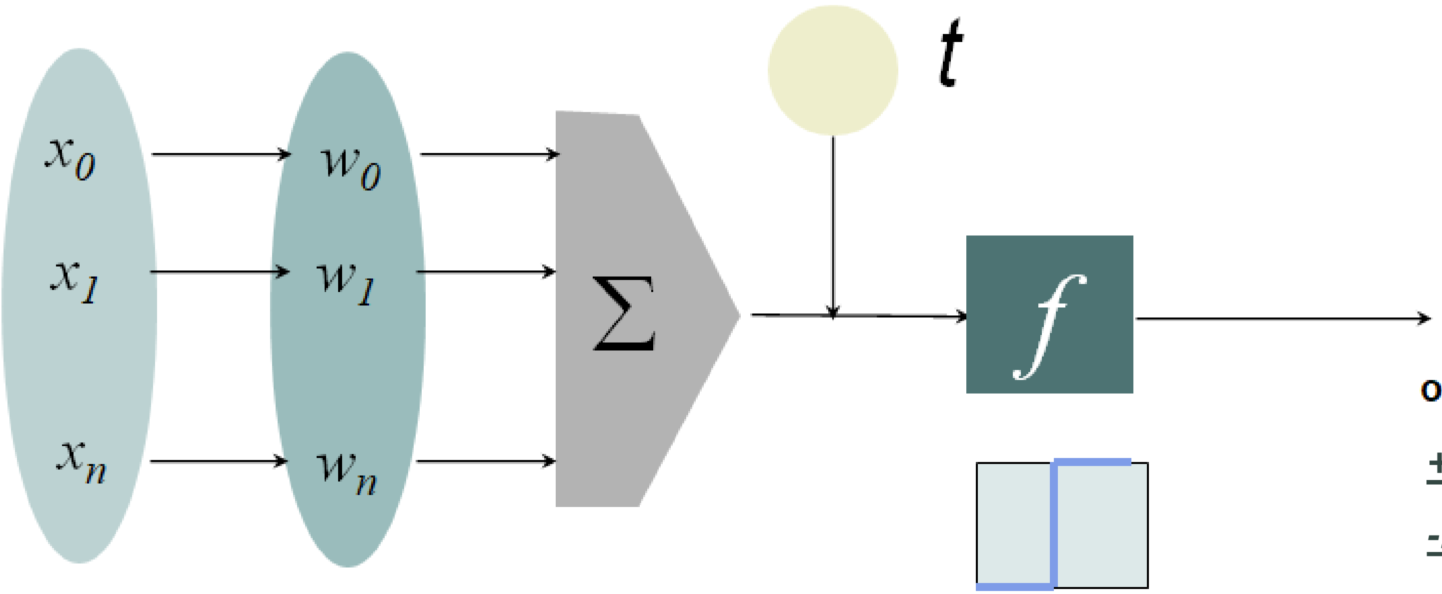
   0 or -1 if $y <= t$

The Perceptron algorithm **learns** the **weights** for the **input signals** in order to draw a **linear decision boundary** that allows to **distinguish** between the **two linearly separable classes**.

Activation functions are used to map the input between the required values, i.e (0, 1) or (-1, 1), depending on which activation function is used.

The *n*-dimensional input vector **x** is mapped into variable **y** by means of the scalar product and a nonlinear function mapping.

3.



output *y*

+1 output -> *the* ***neuron*** *has been* ***triggered***

-1 or 0 output -> *the neuron was* ***not triggered***

Input vector x	Weight vector w	Weighted sum	Activation function	
1.	2.	4.	5.	

Activation function $= \sum w_i x_i - t$

PERCEPTRON: FUNCTIONING

1. All the inputs ***x*** are **multiplied** with their weights ***w***.

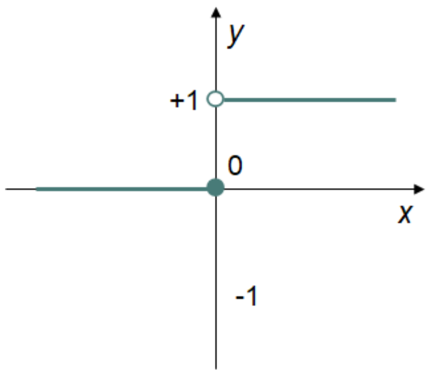
2. **Add** all the multiplied values — **Weighted Sum.**

3. **Add** the **bias** or **threshold** value to the Weighted Sum.

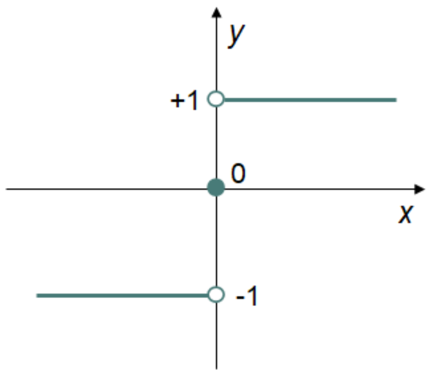
   A bias adjusts the boundary away from the origin without any dependence on the input value, allowing to shift the activation function curve up or down.

4. **Apply** that weighted sum to the correct **Activation Function**.

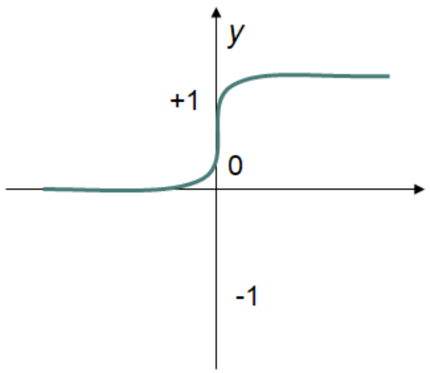
The **step**, **sign** and **sigmoid** functions are examples of activation functions.



Step Function



Sign Function



Sigmoid Function

5. In Perceptron, the **predicted output** is **compared** with the **known output**. If it **<u>does not match</u>**, the **error** is propagated **backward** to allow **weight adjustment**.

Optimal weight coefficients are **automatically** learned.

- Perceptron will always converge if the data is separable by a hyperplane.

- Perceptron only works with linearly separable classes and fails to solve non-linear problems. Many challenging AI problems are not linearly separable and thus the Perceptron was discovered to have critical weakness.

- In the case of modeling logic gates, for example, some scientists discover and claim that Perceptron cannot even learn the exclusive-or problem (XOR) since it is not linearly separable.

Perceptron limitations do not apply to feed-forward networks with intermediate or hidden nonlinear units.

DRIVES
Development and Research on Innovative Vocational Education Skills

An architecture is a family $\mathcal{N} = \{\eta\}$ of networks having the same directed graph and node functions but with possibly different weights on the links.

Based on their network topology, ANN's can be segmented into **five types** of networks.

- Feed-Forward Neural Netwoks
- Recurrent Neural Networks
- Convolutional Neural Networks
- Deconvolutional Neural Networks
- Modular Neural Networks

FEED-FOWARD NEURAL NETWORKS (FFNN)

An ANN is **feed-forward** if there exists an **ordering of neurons** such that every neuron is only connected to a neuron further down the ordering, i.e., there is only one directional signal flow.

It is the simplest type of neural network.

This type of network does not have cycles, data flows unidirectionally from input to output.

This neural network may or may not have hidden layers.

Feed-forward networks have a **front propagated wave** and **no backpropagation** usually by employing a **classifying activation function**.

The architecture of a **feed-forward network** is defined by a **directed and acyclic graph** and the choice of **node functions**.

A directed acyclic graph is one in which no node has a directed path away from it and back to it - there are no closed "cycles."

A directed, acyclic graph can be levelized into layers:

i) The initial or zeroth layer L_o contains the input nodes. These nodes have no incoming links attached to them.

ii) The first layer L_1 consists of those nodes with inputs provided by links from L_o nodes. The ith layer L_i consists of those nodes having inputs provided by links from L_j nodes for $j < i$.

iii) The final layer contains the output nodes.

FEED-FOWARD NEURAL NETWORKS (FFNN)

Feed-forward networks with hidden nonlinear units are **universal approximators**, capable of approximating any bounded continuous function with arbitrarily small errors.

These types of Neural Networks are **responsive** to **noisy data** and **easy to maintain**.

Feedforward neural networks are used in computer vision and speech recognition when classifying target classes is difficult.

FEED-FOWARD NEURAL NETWORKS (FFNN) APPLICATIONS

Supervised Learning

Data is neither sequential nor time-dependent

Responsive to Noisy Data

Computer Vision and Speech Recognition

DRIVES
Development and Research on Innovative Vocational Education Skills

An important special case of an **FFNN** is the **Multilayer Perceptron (MLP)**.

This type of network is derived from the Perceptron.

- As mentioned, the Perceptron algorithm only solves linearly-separable classes.
- It is based on a *treshold* function, which is not the most suitable for these types of problems.
- A better solution to the problem of learning weights is to use **standard optimization techniques**.

DRIVES
Development and Research on Innovative Vocational Education Skills

- In this case, an **error function** is used, which is expressed in terms of the neural network output. The goal of the network then becomes to find the values for the **weights** such that the error function is at its minimum value.

- **Gradient descent techniques** can then be used to determine the **impact** of the **weights** on the value of the **error function**. The **error function** must be **differentiable**, which means it should be **continuous.** The threshold function is not continuous, and so is unsuitable.

- When a function is **differentiable**, it is possible to develop a means of adjusting the weights in a perceptron over as many layers as may be necessary.

DRIVES
Development and Research on Innovative Vocational Education Skills

MLP is the most commonly used feedforward architecture.

In this type of NN, the links to the ith layer L_i come **only** from the **immediately preceding** layer L_{i-1}.

1. Initialise the network, with all weights set to random numbers between -1 and +1.

2. Present the first training pattern and obtain the output.

3. Compare the network output with the target output.

4. Propagate the error backwards.

(i) Correct the output layer of weights using the formula:

$$w_{ho} = w_{ho} + (\eta \delta_o o_h)$$

Where:

- w_{ho} is the weight connecting hidden unit *h* with output unit *o*.
- η is the learning rate.
- o_h is the output at hidden unit *h*.
- δ_o is given by: $\delta_o = o_o(1 - o_o)(t_o - o_o)$

o_o is the output at node *o* of the output layer,and *t−o* is the target output for that node

(ii) Correct the input weights using the formula:

$$w_{ih} = w_{ih} + (\eta \delta_h o_i)$$

Where:

- w_{ih} is the weight connecting node i of the input layer with node h of the hidden layer.
- o_i is the input at node i of the input layer, η is the learning rate.
- δ_h is given by: $\delta_h = o_h(1 - o_h) \sum_o (\delta_o w_{ho})$

5. Calculate the **error**, by taking the average difference between the target and the output vector.

6. Repeat the process from step 2 for **each pattern** in the training set to complete **one epoch**.

7. Shuffle the **training set randomly**, to prevent the network from being influenced by the order of the data.

8. Repeat the process from step 2 for a set number of epochs, or **until** the **error ceases to change**.

DRIVES
Development and Research on Innovative Vocational Education Skills

A **recurrent** NN, also known as **feedback** NN, works on the principle of **saving** the **output** of a layer and **feeding** it **back** to the **input** to help predict the outcome of the layer.

The output neurons can be connected to their inputs.

The feedback network feeds information back into itself to achieve the best-evolved results internally.

Signals in this type of ANN can continuously circulate.

A **Recurrent** network is a neural network with feedback (closed loop) connections.

Examples: BAM, Hopfield, Boltzmann machine, and recurrent backpropagation networks

The **first layer** is formed similar to the **feed-forward** neural network with the **product** of the **sum of the weights** and the **features**. The recurrent neural network process starts once this is computed, which means that from one time step to the next each neuron will remember some information it had in the previous time-step.

Thus, **each neuron** acts like a **memory cell** in performing computations. In this process, the neural network works on the **front propagation** and remembers what information it needs for later use. If the prediction is **wrong,** the **learning rate** or **error correction** is used to make small changes so that it will **gradually** work towards making the **right** prediction during the **back propagation**.

RECURRENT NEURAL NETWORKS (RNN)

The architectures range from **fully interconnected** to **partially connected** networks.

Fully connected networks do not have distinct input layers of nodes, and each node has input from all other nodes. Feedback to the node itself is possible.

Simple partially recurrent neural networks have been used to learn strings of characters. Athough some nodes are part of a feedforward structure, other nodes provide the sequential context and receive feedback from other nodes.

- Sequence Prediction Problems
- Time Series Prediction
- Natural Language Processing
- Video Tagging

Convolutional Neural Networks (CNNs) are analogous to traditional ANNs in that they are made up of neurons that **self-optimise** through learning. Each neuron receives an input and performs an operation - the basis of countless ANNs.

CNNs are similar to FFNNs, where the neurons have learnable weights and biases.

CNNS are designed to take advantage of images (2D).

Its application has been in signal and image processing which takes over OpenCV in the field of computer vision.

The input layer of a **Convolutional** network will hold the pixel values of the image, and the last layer will contain loss functions associated with the classes.

DRIVES
Development and Research on Innovative Vocational Education Skills

Convolutional neural networks are similar to feed forward neural networks. One of the **largest limitations** of traditional forms of ANN is that they tend to **struggle** with the **computational complexity** required to **compute image data**.

The only notable difference between **Convolutional Neural Networks** (**CNNs)** and traditional ANNs is that **CNNs** are primarily used in the field of **pattern recognition** within **images**. This allows us to **encode** image-specific features into the architecture, making the network more suited for **image-focused tasks** - whilst further reducing the parameters required to set up the model.

DRIVES
Development and Research on Innovative Vocational Education Skills

CNNs are comprised of **three** types of layers:

CONVOLUTIONAL LAYERS

The output of neurons connected to local regions of the input will be determined by the convolutional layer by calculating the scalar product between their weights and the region connected to the input volume.

POOLING LAYERS

The pooling layer will then simply perform **downsampling** along the **spatial dimensionality** of the given input, **reducing** the **number of parameters** within that activation even further.

FULLY-CONNECTED LAYERS

The fully-connected layers will then perform the same functions as standard ANNs and attempt to produce **class scores** from the **activations** for the classification.

ReLu may be used between these layers to improve performance.

When these layers are stacked, a CNN architecture has been formed.

DRIVES
Development and Research on Innovative Vocational Education Skills

Through this simple method of transformation, **CNNs** are able to **transform** the **original input layer by layer** using **convolutional** and **downsampling** techniques to produce **class scores** for **classification** and **regression** purposes.

Computer vision techniques are dominated by **convolutional neural networks** because of their **accuracy** in **image classification**.

CONVOLUTIONAL NEURAL NETWORKS (CNN)

DRIVES
Development and Research on Innovative Vocational Education Skills

- Computer Vision
- Facial Recognition
- OCR
- Image Analysis

DRIVES
Development and Research on Innovative Vocational Education Skills

A **deconvolutional neural network**, also known as **deconvs** or **transposed convolutional** neural network, performs an **inverse convolution** model. Some authors describe deconvolutional models as "**reverse engineering**" the input parameters of a convolutional neural network model.

A deconvolutional neural network constructs upwards from processed data.

Signal deconvolution can be used in both image synthesis and analysis.

This NN allows an unsupervised construction of hierarchical image representations.

These representations can be used for both **low-level tasks** such as **denoising**, as well as providing **features** for **object recognition**.

Convolutional Networks are a **bottom-up** approach in which the **input signal** is passed through **multiple layers of convolutions**, **non-linearities**, and **sub-sampling**.

Deconvolutional Networks, on the other hand, follow a **top-down** approach, attempting to generate the **input signal** by a **sum** over **convolutions** of the **feature maps** (rather than the input) with **learned filters**.

Deconvolutional networks strive to find **lost features** or **signals** that previously may have **not** been deemed **important** to a convolutional neural network's task. A signal may be lost due to having been convoluted with other signals.

DECONVOLUTIONAL NEURAL NETWORKS (DNN)

- Extraction of Features from Hierarchical Data
- Removes Pixel-wise and Channel-wise Correlations
- Image Synthesis and Analysis
- Find Lost Features

Biological research has revealed that the **human brain** functions as a collection of **small networks**. As a result, **Modular Neural Networks (MNN)** arose, in which a **collection** of **different networks work independently** to solve problems towards the output.

Each neural network has a distinct set of inputs.

These networks do not interact or signal each other in accomplishing the tasks.

According to the system to model the decomposition into functions has to be meaningfull.

A **modular neural network** has the **advantage** of breaking down a large computational process into smaller components, which **reduces complexity**.

MODULAR NEURAL NETWORKS (MNN)

Instead of starting with a fully connected network and then letting the desired functions emerge through prolonged learning, this approach **starts with a structured modular network architecture**.

Modular Neural Networks are based on the "**divide and conquer**" method, which consists in **breaking down a task** into **smaller** and **less complex subtasks** so that each task is learned by different NN modules. Afterwards, the **network reuses** the **learning** of **each subtask** to **solve** the **whole problem**.

These networks possess an initial architecture that consists of **many modules**. While **connections** within modules may be **dense**, the **modules** themselves are only **sparsely interconnected**.

MODULAR NEURAL NETWORKS (MNN)

Function Approximation

Human Recognition and Biometric Authentication

Time Series Problems

Unsupervised clustering

The aim of designing an artificial neural network is manifold:

- **Optimize performance:** Usually by minimizing the neural network's expected loss for the learning task on unseen test data.
- **Minimize resources:** Reducing the amount of computational (computing power, time, space, etc.) and human (time, effort, etc.) resources required to train the network.

Most deep learning models are designed through trial, error and expert knowledge. Because this manual design process is rarely interpretable or repeatable, there is little formal knowledge about how neural networks work - aside from having a neural network design that may work well for a specific learning task.

DRIVES
Development and Research on Innovative Vocational Education Skills

The aim of designing an artificial neural network is manifold:

- **Maximize the level of automaticity:** The number of decisions that must be made by a human in the neural network design process is inversely proportional to the level of automaticity. As a result, automacity increases as the number of decisions that must be made by a human in the design process decreases.

- **Decrease model's complexity:** Reducing the complexity of the model or the size of the network.

DRIVES
Development and Research on Innovative Vocational Education Skills

- ANNs simulate neurons structures in software, basicaly they are ‘digital versions’ of neurons, synapses, and connection strengths.

- Mathematically, a **network** is represented by Input Nodes, Computing Nodes, Output Nodes, and Edges/Links.

- A **simple NN** includes an **input layer**, a **hidden layer** and an **output (or target) layer**.

- Perceptron is the basic unit of a neural network. It is a single layer binary linear classifier commonly used to classify the data into two parts.

- Perceptrons with a threshold logic function as an activation function are well suited for classification tasks involving linearly separable classes.

- Perceptron only works with linearly separable classes and fails to solve non-linear problems.

- Based on their network topology, ANN’s can be segmented into five types of networks: Feed-Forward Neural Netwoks (FFNN), Recurrent Neural Networks (RNN), Convolutional Neural Networks (CNN) , Deconvolutional Neural Networks (DNN), and Modular Neural Networks (MNN).

- An ANN is feed-forward if there is only one directional signal flow, i.e., data flows unidirectionally from input to output (no cycles).

DRIVES
Development and Research on Innovative Vocational Education Skills

- An important special case of an FFNN is the Multilayer Perceptron (MLP), in which an error function is used, which is expressed in terms of the neural network output. The goal of the network then becomes to find the values for the weights such that the error function is at its minimum value.

- An RNN, also known as Feedback NN, feeds the output of a layer back to the input to help predict the outcome of the layer, i.e., signals in this type of ANN can continuously circulate.

- Convolutional Neural Networks (CNNs) are similar to feed forward neural networks but are more suited for image-focused tasks. They are comprised of three types of layers: convolutional layers, pooling layers and fully-connected layers. ReLu may be used between these layers to improve performance.

- A deconvolutional neural network follows a top-down approach and performs an inverse convolution model. This NN allows an unsupervised construction of hierarchical image representations.

- Modular Neural Networks (MNN) are characterized by a collection of different networks that work independently to solve problems towards the output. A modular neural network has the advantage of breaking down a large computational process into smaller components, which reduces complexity.

REFERENCES

Abreu, S. (2019). Automated Architecture Design for Deep Neural Networks. *arXiv preprint arXiv:1908.10714*.

Stephen, I. (1990). Perceptron-based learning algorithms. *IEEE Transactions on neural networks*, *50*(2), 179.

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, *65*(6), 386.

Fine, T. L. (2006). *Feedforward neural network methodology*. Springer Science & Business Media.

Noriega, L. (2005). Multilayer perceptron tutorial. *School of Computing. Staffordshire University*.

https://analyticsindiamag.com/6-types-of-artificial-neural-networks-currently-being-used-in-todays-technology/

REFERENCES

Medsker, L. R., & Jain, L. C. (2001). Recurrent neural networks. *Design and Applications*, *5*.

Albawi, S., Mohammed, T. A., & Al-Zawi, S. (2017, August). Understanding of a convolutional neural network. In *2017 International Conference on Engineering and Technology (ICET)* (pp. 1-6). Ieee.

O'Shea, K., & Nash, R. (2015). An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*.

Zeiler, M. D., Krishnan, D., Taylor, G. W., & Fergus, R. (2010, June). Deconvolutional networks. In *2010 IEEE Computer Society Conference on computer vision and pattern recognition* (pp. 2528-2535). IEEE.

Ronco, E., & Gawthrop, P. (1995). Modular neural networks: a state of the art. *Rapport Technique CSC-95026, Center of System and Control, University of Glasgow. http://www. mech. gla. ac. uk/control/report. Html*.

Murre, J. M. (2014). *Learning and categorization in modular neural networks*. Psychology Press.

Diana Ferreira

- PhD student in Biomedical Engineering
- Research Collaborator of the Algoritmi Research Center

0000-0003-2326-2153

Regina Sousa

- PhD student in Biomedical Engineering
- Research Collaborator of the Algoritmi Research Center

0000-0002-2988-196X

José Machado

- Associate Professor with Habilitation at the University of Minho
- Integrated Researcher of the Algoritmi Research Center

0000-0003-4121-6169

António Abelha

- Assistant Professor at the University of Minho
- Integrated Researcher of the Algoritmi Research Center

[0000-0001-6457-0756]

Victor Alves

- Assistant Professor at the University of Minho
- Integrated Researcher of the Algoritmi Research Center

[0000-0003-1819-7051]

REFERENCE TO AUTHORS

This Training Material has been certified according to the rules of **ECQA – European Certification and Qualification Association.**

The Training Material was developed within the international job role committee **“Artificial Intelligence Technician”:**

UMINHO – University of Minho (https://www.uminho.pt/PT)

The development of the training material was partly funded by the EU under Blueprint Project DRIVES.

DRIVES

Development and Research on Innovative Vocational Education Skills

Thank you for your attention

DRIVES project is project under **The Blueprint for Sectoral Cooperation on Skills in Automotive Sector**, as part of New Skills Agenda.

The aim of the Blueprint is **to support an overall sectoral strategy and to develop concrete actions to address short and medium term skills needs.**

Follow DRIVES project at:

More information at:

www.project-drives.eu

Co-funded by the Erasmus+ Programme of the European Union

The Development and Research on Innovative Vocational Educational Skills project (DRIVES) is co-funded by the Erasmus+ Programme of the European Union under the agreement 591988-EPP-1-2017-1-CZ-EPPKA2-SSA-B. The European Commission support for the production of this publication does not constitute endorsement of the contents which reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.