

ANN_cpp

Generated by Doxygen 1.9.1

1 Class Index	1
1.1 Class List	1
2 Class Documentation	3
2.1 activation Struct Reference	3
2.1.1 Detailed Description	3
2.2 data Struct Reference	3
2.3 dataset Struct Reference	4
2.4 Layer Class Reference	4
2.4.1 Detailed Description	5
2.4.2 Constructor & Destructor Documentation	5
2.4.2.1 Layer() [1/2]	5
2.4.2.2 Layer() [2/2]	5
2.4.3 Member Function Documentation	6
2.4.3.1 ApplyAllGradient()	6
2.4.3.2 getActivationFunction()	6
2.4.3.3 getNbInput()	6
2.4.3.4 getNbNodes()	7
2.4.3.5 getNode()	7
2.4.3.6 getSumOfWeightedError()	7
2.4.3.7 initError()	8
2.4.3.8 operator=()	8
2.4.3.9 processError()	8
2.4.3.10 processOutputs()	8
2.4.4 Friends And Related Function Documentation	9
2.4.4.1 read	9
2.4.4.2 write	9
2.5 Network Class Reference	10
2.5.1 Detailed Description	11
2.5.2 Constructor & Destructor Documentation	11
2.5.2.1 Network() [1/3]	11
2.5.2.2 Network() [2/3]	11
2.5.2.3 Network() [3/3]	11
2.5.3 Member Function Documentation	12
2.5.3.1 getLayer()	12
2.5.3.2 getLayerSize()	12
2.5.3.3 getNumberLayers()	12
2.5.3.4 LoadNetwork()	13
2.5.3.5 operator=()	13
2.5.3.6 processOutputs()	13
2.5.3.7 SaveNetwork()	13
2.5.3.8 train()	14

2.5.4 Friends And Related Function Documentation	14
2.5.4.1 read	14
2.5.4.2 write	14
2.6 Node Class Reference	15
2.6.1 Detailed Description	16
2.6.2 Constructor & Destructor Documentation	16
2.6.2.1 Node()	16
2.6.3 Member Function Documentation	16
2.6.3.1 applyGradient()	16
2.6.3.2 getActivationFunction()	17
2.6.3.3 getBias()	17
2.6.3.4 getNbInput()	17
2.6.3.5 getWeight()	17
2.6.3.6 getWeightedError()	18
2.6.3.7 initError()	18
2.6.3.8 operator=()	18
2.6.3.9 processError()	19
2.6.3.10 processOutputs()	19
2.6.4 Friends And Related Function Documentation	19
2.6.4.1 read	19
2.6.4.2 write	20
Index	21

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

activation	Type of the activation with its index to find it back from serialization	3
data	3
dataset	4
Layer	Class Layer	4
Network	Class Network	10
Node	Class Node	15

Chapter 2

Class Documentation

2.1 activation Struct Reference

Type of the activation with its index to find it back from serialization.

```
#include <activationFunction.hpp>
```

Public Attributes

- activationFunction **function**
- activationFunction **derivative**
- int **index**

2.1.1 Detailed Description

Type of the activation with its index to find it back from serialization.

The documentation for this struct was generated from the following file:

- /home/marc/Documents/1. Développement/4. C++/1. Neural Network/ANN_cpp/src/activationFunction.hpp

2.2 data Struct Reference

Public Attributes

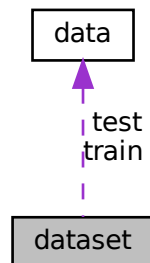
- std::vector< std::vector< double > > **inputs**
- std::vector< std::vector< double > > **outputs**
- std::vector< std::string > **classes**
- int **dataCount**

The documentation for this struct was generated from the following file:

- /home/marc/Documents/1. Développement/4. C++/1. Neural Network/ANN_cpp/src/dataset.hpp

2.3 dataset Struct Reference

Collaboration diagram for dataset:



Public Attributes

- `std::string` **name**
- `int` **nbInput**
- `int` **nbOutput**
- `data` **train**
- `data` **test**

The documentation for this struct was generated from the following file:

- `/home/marc/Documents/1. Développement/4. C++/1. Neural Network/ANN_cpp/src/dataset.hpp`

2.4 Layer Class Reference

Class [Layer](#).

```
#include <layer.hpp>
```

Public Member Functions

- [Layer](#) ()
Construct a new empty [Layer](#) object.
- [Layer](#) ([activation](#) function, `int` nbInput, `int` nbNodes)
Construct a new [Layer](#) object.
- [Layer](#) & [operator=](#) (const [Layer](#) &)=default
Equal operator for the [Layer](#) object.
- `std::vector< double >` [processOutputs](#) (`std::vector< double >` inputs)
process the outputs of this layer

- void `initError` (std::vector< double > inputs, std::vector< double > outputs, std::vector< double > desired↵ Outputs)
Method to process the error if the layer is the output layer.
- void `processError` (std::vector< double > inputs, [Layer](#) nextLayer)
Method to process the error of this layer.
- void `ApplyAllGradient` (std::vector< double > inputs)
Apply all the gradients to update the weights of all the nodes.
- double `getSumOfWeightedError` (int index)
Get the sum of Weighted Error for the input node index.
- int `getNbInput` ()
Get the NbInput object.
- int `getNbNodes` ()
Get the NbNodes object.
- [Node](#) `getNode` (int index)
Get the [Node](#) object at the position index.
- [activation](#) `getActivationFunction` ()
Get the Activation Function object.

Friends

- std::ostream & `write` (std::ostream &out, [Layer](#) &obj)
Overload of the write function for the [Layer](#) object.
- std::istream & `read` (std::istream &in, [Layer](#) &obj)
Overload of the read function for the [Layer](#) object.

2.4.1 Detailed Description

Class [Layer](#).

which implements all the behaviour of a layer

2.4.2 Constructor & Destructor Documentation

2.4.2.1 `Layer()` [1/2]

```
Layer::Layer ( )
```

Construct a new empty [Layer](#) object.

Activation function used by the nodes of this layer

2.4.2.2 `Layer()` [2/2]

```
Layer::Layer (
    activation function,
    int nbInput,
    int nbNodes )
```

Construct a new [Layer](#) object.

Parameters

<i>function</i>	: activation function used by the nodes of this layer
<i>nbInput</i>	: number of inputs in this layer
<i>nbNodes</i>	: number of nodes in this layer

2.4.3 Member Function Documentation

2.4.3.1 ApplyAllGradient()

```
void Layer::ApplyAllGradient (
    std::vector< double > inputs )
```

Apply all the gradients to update the weights of all the nodes.

Parameters

<i>inputs</i>	
---------------	--

2.4.3.2 getActivationFunction()

```
activation Layer::getActivationFunction ( )
```

Get the Activation Function object.

Returns

activation

2.4.3.3 getNbInput()

```
int Layer::getNbInput ( )
```

Get the NbInput object.

Returns

int

2.4.3.4 getNbNodes()

```
int Layer::getNbNodes ( )
```

Get the NbNodes object.

Returns

int

2.4.3.5 getNode()

```
Node Layer::getNode (
    int index )
```

Get the [Node](#) object at the position index.

Parameters

<i>index</i>	: the position of the node we want to access.
--------------	---

Returns

[Node](#)

2.4.3.6 getSumOfWeightedError()

```
double Layer::getSumOfWeightedError (
    int index )
```

Get the sum of Weighted Error for the input node index.

Parameters

<i>index</i>	
--------------	--

Returns

double

2.4.3.7 initError()

```
void Layer::initError (
    std::vector< double > inputs,
    std::vector< double > outputs,
    std::vector< double > desiredOutputs )
```

Method to process the error if the layer is the output layer.

Parameters

<i>inputs</i>	
<i>outputs</i>	
<i>desiredOutputs</i>	

2.4.3.8 operator=()

```
Layer& Layer::operator= (
    const Layer & ) [default]
```

Equal operator for the [Layer](#) object.

Returns

[Layer&](#)

2.4.3.9 processError()

```
void Layer::processError (
    std::vector< double > inputs,
    Layer nextLayer )
```

Method to process the error of this layer.

Parameters

<i>inputs</i>	
<i>nextLayer</i>	

2.4.3.10 processOutputs()

```
std::vector< double > Layer::processOutputs (
    std::vector< double > inputs )
```

process the outputs of this layer

Parameters

<i>inputs</i>	: vector of inputs
---------------	--------------------

Returns

`std::vector<double>` : vector of outputs

2.4.4 Friends And Related Function Documentation

2.4.4.1 read

```
std::istream& read (
    std::istream & in,
    Layer & obj ) [friend]
```

Overload of the read function for the [Layer](#) object.

Parameters

<i>in</i>	
<i>obj</i>	

Returns

`std::istream&`

2.4.4.2 write

```
std::ostream& write (
    std::ostream & out,
    Layer & obj ) [friend]
```

Overload of the write function for the [Layer](#) object.

Parameters

<i>out</i>	
<i>obj</i>	

Returns

std::ostream&

The documentation for this class was generated from the following files:

- /home/marc/Documents/1. Développement/4. C++/1. Neural Network/ANN_cpp/src/layer.hpp
- /home/marc/Documents/1. Développement/4. C++/1. Neural Network/ANN_cpp/src/layer.cpp

2.5 Network Class Reference

Class [Network](#).

```
#include <network.hpp>
```

Public Member Functions

- [Network](#) ()
Construct a new empty [Network](#) object.
- [Network](#) (std::vector< int > Size, std::vector< [activation](#) > activationFunctions)
Construct a new [Network](#) object.
- [Network](#) (std::vector< int > Size, [activation](#) activationfunction)
Construct a new [Network](#) object.
- [Network](#) & [operator=](#) (const [Network](#) &)=default
Equal operator for the [Network](#) object.
- std::vector< double > [processOutputs](#) (std::vector< double > inputs)
Process of the network layer by layer.
- void [train](#) ([dataset](#) ds)
Process the training of the model on the selected dataset.
- int [getLayerSize](#) (int index)
Get the [Layer](#) Size object.
- int [getNumberLayers](#) ()
Get the Number Layers object.
- [Layer](#) [getLayer](#) (int index)
Get the [Layer](#) object.
- void [LoadNetwork](#) (std::string path)
Load a network from a binary file situated on the path.
- void [SaveNetwork](#) (std::string path)
Save the network to a binary file situated on the path.

Friends

- std::ostream & [write](#) (std::ostream &out, [Network](#) &obj)
Overload of the write function for the [Network](#) object.
- std::istream & [read](#) (std::istream &in, [Network](#) &obj)
Overload of the read function for the [Network](#) object.

2.5.1 Detailed Description

Class [Network](#).

which implements all the behaviour of a [Network](#).

2.5.2 Constructor & Destructor Documentation

2.5.2.1 [Network\(\)](#) [1/3]

```
Network::Network ( )
```

Construct a new empty [Network](#) object.

Size of the differents layers

2.5.2.2 [Network\(\)](#) [2/3]

```
Network::Network (
    std::vector< int > Size,
    std::vector< activation > activationFunctions )
```

Construct a new [Network](#) object.

Parameters

<i>Size</i>	: the differents sizes of the layers.
<i>activationFunctions</i>	: the differents activations function used by the layers.

2.5.2.3 [Network\(\)](#) [3/3]

```
Network::Network (
    std::vector< int > Size,
    activation activationfunction )
```

Construct a new [Network](#) object.

Parameters

<i>Size</i>	: the differents sizes of the layers.
<i>activationfunction</i>	: the activation function used by all the layers.

2.5.3 Member Function Documentation

2.5.3.1 `getLayer()`

```
Layer Network::getLayer (
    int index )
```

Get the [Layer](#) object.

Parameters

<i>index</i>	
--------------	--

Returns

[Layer](#)

2.5.3.2 `getLayerSize()`

```
int Network::getLayerSize (
    int index )
```

Get the [Layer](#) Size object.

Parameters

<i>index</i>	
--------------	--

Returns

int

2.5.3.3 `getNumberLayers()`

```
int Network::getNumberLayers ( )
```

Get the Number Layers object.

Returns

int

2.5.3.4 LoadNetwork()

```
void Network::LoadNetwork (
    std::string path )
```

Load a network from a binary file situated on the path.

Parameters

<i>path</i>	: path to the network .bin file
-------------	---------------------------------

2.5.3.5 operator=()

```
Network& Network::operator= (
    const Network & ) [default]
```

Equal operator for the `Network` object.

Returns

`Network&`

2.5.3.6 processOutputs()

```
std::vector< double > Network::processOutputs (
    std::vector< double > inputs )
```

Process of the network layer by layer.

Parameters

<i>inputs</i>	: vector of inputs.
---------------	---------------------

Returns

`std::vector<double>` : vector of outputs.

2.5.3.7 SaveNetwork()

```
void Network::SaveNetwork (
    std::string path )
```

Save the network to a binary file situated on the path.

Parameters

<i>path</i>	: path to the newly created .bin file
-------------	---------------------------------------

2.5.3.8 train()

```
void Network::train (
    dataset ds )
```

Process the training of the model on the selected dataset.

Parameters

<i>ds</i>	
-----------	--

2.5.4 Friends And Related Function Documentation**2.5.4.1 read**

```
std::istream& read (
    std::istream & in,
    Network & obj ) [friend]
```

Overload of the read function for the [Network](#) object.

Parameters

<i>in</i>	
<i>obj</i>	

Returns

std::istream&

2.5.4.2 write

```
std::ostream& write (
    std::ostream & out,
    Network & obj ) [friend]
```

Overload of the write function for the [Network](#) object.

Parameters

<i>out</i>	
<i>obj</i>	

Returns

std::ostream&

The documentation for this class was generated from the following files:

- /home/marc/Documents/1. Développement/4. C++/1. Neural Network/ANN_cpp/src/network.hpp
- /home/marc/Documents/1. Développement/4. C++/1. Neural Network/ANN_cpp/src/network.cpp

2.6 Node Class Reference

Class [Node](#).

```
#include <node.hpp>
```

Public Member Functions

- [Node](#) ([activation](#) function, int nbln)
Construct a new [Node](#) object.
- [Node](#) ()
Empty constructor for [Node](#) object.
- [Node](#) & [operator=](#) (const [Node](#) &)=default
Equal operator for the node object.
- double [processOutputs](#) (std::vector< double > inputs)
Method to process the outputs of the node.
- void [initError](#) (std::vector< double > inputs, double output, double desiredOutput)
Method to process the error if the node is situated on the output layer.
- void [processError](#) (std::vector< double > inputs, double sumOfWeightedError)
Method to process the error using the backpropagation method.
- double [getWeightedError](#) (int index)
Get the weighted error that we'll use for the backpropagation.
- void [applyGradient](#) (std::vector< double > inputs)
Apply Gradients calculated by the backpropagation.
- double [getWeight](#) (int index)
Get one of the weight.
- double [getBias](#) ()
Get the Bias.
- int [getNbInput](#) ()
Get the number of input.
- [activation](#) [getActivationFunction](#) ()
Get the Activation Function of the [Node](#).

Friends

- `std::ostream & write (std::ostream &out, Node &obj)`
Overload of the write function for the node object.
- `std::istream & read (std::istream &in, Node &obj)`
Overload of the read function for the node object.

2.6.1 Detailed Description

Class [Node](#).

Class which represent the behaviour of a [Node](#) in an Artificial Neural [Network](#).

2.6.2 Constructor & Destructor Documentation

2.6.2.1 Node()

```
Node::Node (
    activation function,
    int nbIn )
```

Construct a new [Node](#) object.

Construct a new [Node](#) object by creating random weights and a random bias.

Parameters

<i>nbIn</i>	corresponds to the number of inputs of the created Node .
-------------	---

2.6.3 Member Function Documentation

2.6.3.1 applyGradient()

```
void Node::applyGradient (
    std::vector< double > inputs )
```

Apply Gradients calculated by the backpropagation.

Parameters

<i>inputs</i>	: the inputs of the node.
---------------	---------------------------

2.6.3.2 getActivationFunction()

```
activation Node::getActivationFunction ( )
```

Get the Activation Function of the [Node](#).

Returns

activation : the [Node](#) Activation Function.

2.6.3.3 getBias()

```
double Node::getBias ( )
```

Get the Bias.

Returns

double : the bias.

2.6.3.4 getNbInput()

```
int Node::getNbInput ( )
```

Get the number of input.

Returns

int : the number of input.

2.6.3.5 getWeight()

```
double Node::getWeight (
    int index )
```

Get one of the weight.

Parameters

<i>index</i>	of the weight we want.
--------------	------------------------

Returns

double : the weight.

2.6.3.6 getWeightedError()

```
double Node::getWeightedError (
    int index )
```

Get the weighted error that we'll use for the backpropagation.

Parameters

<i>index</i>	
--------------	--

Returns

double

2.6.3.7 initError()

```
void Node::initError (
    std::vector< double > inputs,
    double output,
    double desiredOutput )
```

Method to process the error if the node is situated on the output layer.

Parameters

<i>input</i>	
<i>output</i>	
<i>desiredOutput</i>	

2.6.3.8 operator=()

```
Node& Node::operator= (
    const Node & ) [default]
```

Equal operator for the node object.

Returns

the address [Node](#) of the left sided [Node](#) object.

2.6.3.9 processError()

```
void Node::processError (
    std::vector< double > inputs,
    double sumOfWeightedError )
```

Method to process the error using the backpropagation method.

Parameters

<i>input</i>	
<i>sumOfWeightedError</i>	

2.6.3.10 processOutputs()

```
double Node::processOutputs (
    std::vector< double > inputs )
```

Method to process the outputs of the node.

Method which realise the calculation of the output by doing the dot product of the weights by the inputs. Then it add the bias and finally it use the Activation Function on the resulting scalar.

Parameters

<i>inputs</i>	: the inputs of the node.
---------------	---------------------------

Returns

double : the state of the node after the calculation.

2.6.4 Friends And Related Function Documentation

2.6.4.1 read

```
std::istream& read (
    std::istream & in,
    Node & obj ) [friend]
```

Overload of the read function for the node object.

Parameters

<i>in</i>	
<i>obj</i>	

Returns

std::istream&

2.6.4.2 write

```
std::ostream& write (
    std::ostream & out,
    Node & obj ) [friend]
```

Overload of the write function for the node object.

Parameters

<i>out</i>	
<i>obj</i>	

Returns

std::ostream&

The documentation for this class was generated from the following files:

- /home/marc/Documents/1. Développement/4. C++/1. Neural Network/ANN_cpp/src/node.hpp
- /home/marc/Documents/1. Développement/4. C++/1. Neural Network/ANN_cpp/src/node.cpp

Index

- activation, [3](#)
- ApplyAllGradient
 - Layer, [6](#)
- applyGradient
 - Node, [16](#)
- data, [3](#)
- dataset, [4](#)
- getActivationFunction
 - Layer, [6](#)
 - Node, [17](#)
- getBias
 - Node, [17](#)
- getLayer
 - Network, [12](#)
- getLayerSize
 - Network, [12](#)
- getNbInput
 - Layer, [6](#)
 - Node, [17](#)
- getNbNodes
 - Layer, [6](#)
- getNode
 - Layer, [7](#)
- getNumberLayers
 - Network, [12](#)
- getSumOfWeightedError
 - Layer, [7](#)
- getWeight
 - Node, [17](#)
- getWeightedError
 - Node, [18](#)
- initError
 - Layer, [7](#)
 - Node, [18](#)
- Layer, [4](#)
 - ApplyAllGradient, [6](#)
 - getActivationFunction, [6](#)
 - getNbInput, [6](#)
 - getNbNodes, [6](#)
 - getNode, [7](#)
 - getSumOfWeightedError, [7](#)
 - initError, [7](#)
 - Layer, [5](#)
 - operator=, [8](#)
 - processError, [8](#)
 - processOutputs, [8](#)
- read, [9](#)
- write, [9](#)
- LoadNetwork
 - Network, [12](#)
- Network, [10](#)
 - getLayer, [12](#)
 - getLayerSize, [12](#)
 - getNumberLayers, [12](#)
 - LoadNetwork, [12](#)
 - Network, [11](#)
 - operator=, [13](#)
 - processOutputs, [13](#)
 - read, [14](#)
 - SaveNetwork, [13](#)
 - train, [14](#)
 - write, [14](#)
- Node, [15](#)
 - applyGradient, [16](#)
 - getActivationFunction, [17](#)
 - getBias, [17](#)
 - getNbInput, [17](#)
 - getWeight, [17](#)
 - getWeightedError, [18](#)
 - initError, [18](#)
 - Node, [16](#)
 - operator=, [18](#)
 - processError, [18](#)
 - processOutputs, [19](#)
 - read, [19](#)
 - write, [20](#)
- operator=
 - Layer, [8](#)
 - Network, [13](#)
 - Node, [18](#)
- processError
 - Layer, [8](#)
 - Node, [18](#)
- processOutputs
 - Layer, [8](#)
 - Network, [13](#)
 - Node, [19](#)
- read
 - Layer, [9](#)
 - Network, [14](#)
 - Node, [19](#)
- SaveNetwork

Network, [13](#)

train

Network, [14](#)

write

Layer, [9](#)

Network, [14](#)

Node, [20](#)