ANN_cpp

Generated by Doxygen 1.9.1

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

Class Documentation

2.1 activation Struct Reference

Public Attributes

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

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 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
- 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.

2.2.1 Detailed Description

Class Layer.

which implements all the behaviour of a layer

2.2.2 Constructor & Destructor Documentation

2.2.2.1 Layer() [1/2]

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

Construct a new empty Layer object.

Activation function used by the nodes of this layer

2.2.2.2 Layer() [2/2]

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

Construct a new Layer object.

Parameters

function	: activation function used by the nodes of this layer
nbInput	: number of inputs in this layer
nbNodes	: number of nodes in this layer

2.2.3 Member Function Documentation

2.2.3.1 getActivationFunction()

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

Get the Activation Function object.

Returns

activation

2.2.3.2 getNbInput()

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

Get the NbInput object.

Returns

int

2.2.3.3 getNbNodes()

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

Get the NbNodes object.

Returns

int

2.2.3.4 getNode()

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

Get the Node object at the position index.

Parameters

index	: the position of the node we want to access.

Returns

Node

2.2.3.5 operator=()

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

Equal operator for the Layer object.

Returns

Layer&

2.2.3.6 processOutputs()

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

process the outputs of this layer

Parameters

inputs	: vector of inputs

Returns

std::vector<double> : vector of outputs

2.2.4 Friends And Related Function Documentation

2.2.4.1 read

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

Overload of the read function for the Layer object.

Parameters

in	
obj	

Returns

std::istream&

2.2.4.2 write

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

Overload of the write function for the Layer object.

Parameters

out	
obj	

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

2.3 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.
- 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.

2.3.1 Detailed Description

Class Network.

which implements all the behaviour of a Network.

2.3.2 Constructor & Destructor Documentation

2.3.2.1 Network() [1/3]

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

Construct a new empty Network object.

Size of the differents layers

2.3.2.2 Network() [2/3]

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

Construct a new Network object.

Parameters

Size	: the differents sizes of the layers.
activationFunctions	: the differents activations function used by the layers.

2.3.2.3 Network() [3/3]

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

Construct a new Network object.

Parameters

Size	: the differents sizes of the layers.
activationfunction	: the activation function used by all the layers.

2.3.3 Member Function Documentation

2.3.3.1 getLayer()

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

Get the Layer object.

Parameters

index	

Returns

Layer

2.3.3.2 getLayerSize()

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

Get the Layer Size object.

Parameters

index	

Returns

int

2.3.3.3 getNumberLayers()

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

Get the Number Layers object.

Returns

int

2.3.3.4 LoadNetwork()

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

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

Parameters

path	: path to the network .bin file

2.3.3.5 operator=()

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

Equal operator for the Network object.

Returns

Network&

2.3.3.6 processOutputs()

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

Process of the network layer by layer.

Parameters

inputs	: vector of inputs.

Returns

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

2.3.3.7 SaveNetwork()

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

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

Parameters

path	: path to the newly created .bin file

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

2.4 Node Class Reference

Class Node.

`#include <node.hpp>`

Public Member Functions

- Node (activation function, int nbIn)

  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.
- 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.

2.4.1 Detailed Description

Class Node.

Class which represent the behaviour of a Node in an Artificial Neural Network.

2.4.2 Constructor & Destructor Documentation

2.4.2.1 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

nb↩ In	corresponds to the number of inputs of the created Node.

2.4.3 Member Function Documentation

2.4.3.1 getActivationFunction()

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

Get the Activation Function of the Node.

Returns

activation : the Node Activation Function.

2.4.3.2 getBias()

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

Get the Bias.

Returns

double : the bias.

2.4.3.3 getNbInput()

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

Get the number of input.

Returns

int : the number of input.

2.4.3.4 getWeight()

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

Get one of the weight.

Parameters

index	of the weight we want.

Returns

double : the weight.

2.4.3.5 operator=()

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

Equal operator for the node object.

Returns

the address Node of the left sided Node object.

2.4.3.6 processOutputs()

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

Method to process the outputs of the node.

Method which realise the calculation of the ouput 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

inputs	: the inputs of the node.

Returns

double : the state of the node after the calculation.

2.4.4 Friends And Related Function Documentation

2.4.4.1 read

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

Overload of the read function for the node object.

Parameters

in	
obj	

Returns

std::istream&

2.4.4.2 write

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

Overload of the write function for the node object.

Parameters

out	
obj	

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