
Neuron-Matrix Documentation

Release 1.0

Hadrien Renaud-Lebret, Guillaume Bressan, Pierre Browne

Feb 10, 2017

CONTENTS:

1	NeuralNetwork module	1
2	Processsing Utilitaries	3
2.1	getdata module	3
2.2	fileio module	3
3	Indices and tables	5
	Python Module Index	7

NEURALNETWORK MODULE

Python module for NeuralNetwork class.

It provides an implementation of a NeuralNetwork with utilities. It is NOT bounded to learning on images or even to learning on samples in different files.

```
class neuralnet.NeuralNetwork (geometry, function=<ufunc 'tanh'>, function_derivate=<function  
                                inv_cosh>, logistic_function_param=(1, 0), learning_factor=0.1,  
                                momentum=0)
```

NeuralNetwork class.

```
__call__ (input_values)
```

Apply the Neural Network to the input values.

Warning DOESN'T SAVE the result for a learning after. Use `apply()` in this case.

Parameters `input_values` – as an iterable of numeric values between 0 and 1.

```
__init__ (geometry, function=<ufunc 'tanh'>, function_derivate=<function inv_cosh>, logis-  
          tic_function_param=(1, 0), learning_factor=0.1, momentum=0)
```

Initialisation of the NeuralNetwork.

Parameters

- **geometry** (*str*) – string describing the format of the NeuralNetwork: '456:12:24:3' will create a network with a first layer with 456 neurons, a second with 12, a third with 24 and the last with 3.
- **function** – vectorized function (see `numpy.vectorize`)
- **function_derivate** – its (vectorized) function
- **logistic_function_param** (*tuple*) – (mu, x0) parameters send to :iso_fonction: and :deri_iso_fonction: slope and offset of the logistic function.

```
apply (input_values)
```

Apply the NeuralNetwork to the input values.

Parameters `input_values` – as an iterable of numeric values between 0 and 1.

```
backpropagation (expected_output)
```

Apply the backpropagation algorithm.

```
dist (expected_output)
```

Calc the distance of the result to the expected_output.

```
get_geometry ()
```

Return self.geometry.

Returns self.geometry modified to render like the one passed as an argument of `__init__()`.

Return type `str`

learn (*sample, results, limit_iterations=50, maximal_distance=0.25*)

Learning algorithm on the given examples.

First algorithm.

learn2 (*sample, results, limit_iterations=50, maximal_distance=0.25*)

Learning algorithm on the given examples.

Method given by H             [here](#).

randomize_factors ()

Randomize the transition matrix.

set_transition_matrix (*matrixes*)

Set the transition_matrix to the correct values.

to_json ()

Return an expression of the NeuralNetwork in json.

`neuralnet.deriso_fonction` (*fonction, mu=1, x0=0*)

Creator of fonctions, linearly translated, for derivative.

`neuralnet.inv_cosh` (*x*)

Return 1 / cosh(x).

`neuralnet.iso_fonction` (*fonction, mu=1, x0=0*)

Creator of fonctions, linearly translated.

`neuralnet.learning_progress_display` (***args*)

Display the progress of the learning algorithm.

`neuralnet.alphabet`

Alphabet used. The order is the most important thing. Only the *\$n\$* first values are considered, where *\$n\$* is the length of the alphabet.

`neuralnet.default_values`

PROCESSING UTILITARIES

2.1 getdata module

Module for neuron-matrix.

```
class getdata.IteratorMultiple (lengths)
    Bases: object
```

Iterator through an unknown number of lists.

```
getdata.get_data (proc='1', ranges={'learning_factor': [0.1], 'learning_algo': ['default'], 'maximal_distance': [0.2], 'limit_iterations': [50], 'momentum': [0.5]}, **kwargs)
    Process with different parameters the sample.
```

```
getdata.procedure1 (alphabet='abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789,
    ?,:!èàè\')+-'=', learning_algo='default', learning_directory='LearningSample', testing_directory='TestSample', learning_factor=0.1, momentum=0.5, limit_iterations=50, maximal_distance=0.2)
    Function that execute procedure1.
```

It does: - create a new NeuralNetwork object - randomize its transition_matrix - learn on a given dataset - test on a given dataset

```
getdata.default_ranges
    Ranges on which it iterate by default.
```

2.2 fileio module

Python module for neuron-matrix.

It implements the main input/output functions used in neuron-matrix.

```
fileio.find_examples (directory, alphabet)
    Iterate through the directory to find all processable examples and return them.
```

```
fileio.is_convertible_to_float (string)
    Function that determine if a string can be safely convert to a float.
```

```
fileio.learn_on_folder (neurnet, directory, alphabet, learning_algo='default', **args)
    Make neurnet learn on every example in the directory.
```

```
fileio.read_sample (file_text)
    Function reading an sample in a file and returning the corresponding matrix.
    Return the result as a numpy array.
```

`fileio.save_image(matrix, file_name)`

Function saving matrix as an image.

`fileio.test_on_folder(neurnet, directory, alphabet, **args)`

Make neurnet test every example in the directory.

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`

PYTHON MODULE INDEX

f

fileio, 3

g

getdata, 3

n

neuralnet, 1

Symbols

`__call__()` (neuralnet.NeuralNetwork method), 1
`__init__()` (neuralnet.NeuralNetwork method), 1

A

`alphabet` (in module neuralnet), 2
`apply()` (neuralnet.NeuralNetwork method), 1

B

`backpropagation()` (neuralnet.NeuralNetwork method), 1

D

`default_ranges` (in module getdata), 3
`default_values` (in module neuralnet), 2
`deri_iso_fonction()` (in module neuralnet), 2
`dist()` (neuralnet.NeuralNetwork method), 1

F

`fileio` (module), 3
`find_examples()` (in module fileio), 3

G

`get_data()` (in module getdata), 3
`get_geometry()` (neuralnet.NeuralNetwork method), 1
`getdata` (module), 3

I

`inv_cosh()` (in module neuralnet), 2
`is_convertible_to_float()` (in module fileio), 3
`iso_fonction()` (in module neuralnet), 2
`IteratorMultiple` (class in getdata), 3

L

`learn()` (neuralnet.NeuralNetwork method), 2
`learn2()` (neuralnet.NeuralNetwork method), 2
`learn_on_folder()` (in module fileio), 3
`learning_progress_display()` (in module neuralnet), 2

N

`neuralnet` (module), 1
`NeuralNetwork` (class in neuralnet), 1

P

`procedure1()` (in module getdata), 3

R

`randomize_factors()` (neuralnet.NeuralNetwork method), 2
`read_sample()` (in module fileio), 3

S

`save_image()` (in module fileio), 3
`set_transition_matrix()` (neuralnet.NeuralNetwork method), 2

T

`test_on_folder()` (in module fileio), 4
`to_json()` (neuralnet.NeuralNetwork method), 2