GNG documentation

June 6, 2014

`GNG` *Constructor of GrowingNeuralGas object*

Description

Construct GNG object

Usage

```
GNG
```

Arguments

`beta`	coefficient. Decrease the error variables of all node nodes by this fraction. Forgetting rate. Default 0.99
`alpha`	Alpha coefficient. Decrease the error variables of the nodes neighboring to the newly inserted node by this fraction. Default 0.5
`uniformgrid.optimization`	TRUE/FALSE. You cannot use utility option with this, so please pass FALSE here then.
`lazyheap.optimization`	TRUE/FALSE. You cannot use utility option with this, so please pass FALSE here then.
`max.nodes`	Maximum number of nodes (after reaching this size it will continue running, but won't add new nodes)
`eps_w`	Default 0.05
`eps_n`	Default 0.0006
`dataset.type`	Dataset type. Possibilities gng.dataset.bagging, gng.dataset.bagging.prob, gng.dataset.sequential
`experimental_utility_option`	EXPERIMENTAL Utility option gng.experimental.utility.option.off / gng.experimental.utility.option.basi
`experimental_utility_k`	EXPERIMENTAL Utility option constant

load_model_filename
: Set to path to file from which load serialized model

experimental_vertex_extra_data
: if TRUE each example should have additional coordinate, that will be voted in graph. Each node (that you can get using node function) will have extra_data field that will be equal to mean of samples around given node. If used with probability dataset example layout is <vertex position> <vertex_extra_data> <sampling_probability>, for example c(0.3, 0.6, 0.7, 100, 0.7)

Format

NULL

Examples

```
# Default GNG instance, without optimitzations and vertex dimensionality 3

gng <- GNG(dataset_type=gng.dataset.bagging.prob, max_nodes=max_nodes, dim=3)

# Construct GNG loaded from file with uniform grid

gng <- GNG(dataset_type=gng.dataset.bagging.prob, max_nodes=max_nodes, dim=3,
uniformgrid_optimization=TRUE,  lazyheap_optimization=FALSE,
uniformgrid_boundingbox_sides=c(3,3,3), uniformgrid_boundingbox_origin=c(-0.5,-0.5,-0.5),
load_model_filename="sphere_simple.bin")
```

Growing-Neural-Gas *Efficient C++ implementation of online clustering algorithm Growing Neural Gas.*

Description

This package contains fast C++ implementation of online clustering algorithm Growing Neural Gas. It produces topological graph, that you can easily convert to igraph, or you can dump your model to optimized binary file and load it later on.

Details

Package:	Growing-Neural-Gas
Type:	Package
Version:	0.6
Date:	2014-06-03
License:	MIT License

For overview of usage please see exemplary code in demo/ folder.

Author(s)

Stanislaw Jastrzebski

Maintainer: Stanislaw Jastrzebski <staszek<at>gmail<dot>com>

References

~~ Literature or other references for background information ~~

Examples

```
gng <- GNG(dataset_type=gng.dataset.bagging.prob, max_nodes=300, dim=3)

insert_examples(gng, preset=gng.preset.sphere, N=10000, prob=0.8)

run(gng)

insert_examples(gng, preset=gng.preset.box, N=10000, prob=0.8)

plot(gng, mode=gng.plot.2d.errors, vertex.color=gng.plot.color.cluster, layout=gng.plot.layout.igraph.v2d)
```

---

centroids.gng *centroids*

Description

Using infomap.communities finds communities and for each community pick node with biggest betweenness score

Usage

```
centroids(gng)
```

Format

NULL

Details

Get centroids

Examples

```
# Print position of the first centroid
print(node(gng, centroids(gng)[1])$pos)
```

convert_igraph.gng *convert_igraph*

Description

Converts to igraph (O(n) method, writing intermediately to disk)

Usage

```
convert_igraph(gng)
```

Format

NULL

Examples

```
convert_igraph(gng)
```

dump_model.gng *dump_model*

Description

Writes graph to a disk space efficient binary format. It can be used in GNG constructor to reconstruct graph from file.

Usage

```
dump_model(gng)
```

Arguments

filename	Dump destination

Format

NULL

Details

Dump model to binary

Examples

```
dump_model(gng, graph.bin)
```

error_statistics.gng *error_statistics*

Description

Gets vector with errors for every second of execution

Usage

```
error_statistics(gng)
```

Format

NULL

Examples

```
error_statistics(gng)
```

insert_examples.gng *insert_examples*

Description

Insert (inefficiently) examples to algorithm dataset. For efficient insertion use gng$inser_examples, and for setting memory pointer use gng$set_memory_move_examples

Usage

```
insert_examples(gng, M)
```

Arguments

examples	Matrix with examples of dimensionality N rows x C columns, where C columns = dim (passed as parameter to GNG object) + 1 or 0 (1 if vertex_extra_data is TRUE) + 1 or 0 (1 if dataset_type=gng.dataset.bagging.prob).
preset	Use only if you are adding exemplary dataset. Possibilities: gng.preset.sphere, gng.preset.box. You can specify preset params: N=1000, center=c(0.5,0.5,0.5), prob=-1.

Format

NULL

Note

Complicated memory layout of the matrix is due to need for memory efficiency. In the future versions you can expect wrapper simplifying addition and also streaming from disk file

Examples

```
#Add preset examples with probability of being sampled (this assumed GNG was created with gng.dataset.bagging.prob
insert_examples(gng, preset=gng.preset.sphere)

#Insert efficiently examples
M <- matrix(0, ncol=3, nrow=10)
M[1,] = c(4,5,6)
gng$insert_examples(M)

#Set memory of the algorithm to point in memory. Note: you cannot remove this matrix while
in execution or you will experience memory error
gng$set_memory_move_examples(M)
```

---

mean_error.gng *mean_error*

Description

Gets mean error of the graph (note: blocks the execution, O(n))

Usage

```
mean_error(gng)
```

Format

NULL

Examples

```
mean_error(gng)
```

node.gng *node*

Description

Retrieves node from resulting graph

Usage

```
node(gng, 10)
```

Arguments

gng_id	Id of the node to retrieve. This is the id returned by functions like predict, or centroids

Format

NULL

Details

Get GNG node

Examples

```
print(node(gng, 10)$pos)
```

pause.gng *pause*

Description

Pause algorithm

Usage

```
pause(gng)
```

Format

NULL

Examples

```
pause(gng)
```

plot.gng *plot*

Description

Plot resulting graph using igraph plotting, or using rgl 3d engine.

Usage

```
plot(gng)
```

Arguments

mode	gng.plot.rgl3d (3d plot), gng.plot.2d (igraph plot) or gng.plot.2d.errors (igraph plot with mean error log plot)
layout	layout to be used when plotting. Possible values: gng.plot.layour.igraph.v2d (first two dimensions), gng.plot.layout.igraph.auto (auto layout from igraph) gng.plot.layout.igraph.fruchterman.fast (fast fruchterman reingold layout),or any function accepting igraph graph and returning layout
vertex.color	how to color vertexes. Possible values: gng.plot.color.cluster(vertex color is set to fastgreedy.community clustering), gng.plot.color.extra(rounds to integer extra dim if present), gng.plot.color.none(every node is white),

Format

NULL

Details

Plot GNG

Note

If you want to "power-use" plotting and plot for instance a subgraph, you might be interested in exporting igraph with convert_igraph function and plotting it using/reusing function from this package: .visualizeIGraph2d

Examples

```
# Plots igraph using first 2 coordinates and colors according to clusters
plot(gng, mode=gng.plot.2d.errors, layout=gng.plot.layout.v2d, vertex.color=gng.plot.color.cluster)

# Plot rgl (make sure you have installed rgl library)
plot(gng, mode=gng.plot.rgl, layout=gng.plot.layout.v2d, vertex.color=gng.plot.color.cluster)

# For more possibilities see gng.plot.* constants
```

run.gng *run*

Description

Run algorithm (in parallel)

Usage

```
run(gng)
```

Format

NULL

Examples

```
run(gng)
```

terminate.gng *pause*

Description

Terminate algorithm

Usage

```
terminate(gng)
```

Format

NULL

Examples

```
terminate(gng)
```

Index