

LibTopoART

v0.74

Generated by Doxygen 1.8.13

Sat Apr 22 2017 13:57:32

## Contents

<b>1</b>	<b>Namespace Index</b>	<b>1</b>
1.1	Packages . . . . .	1
<b>2</b>	<b>Hierarchical Index</b>	<b>1</b>
2.1	Class Hierarchy . . . . .	1
<b>3</b>	<b>Class Index</b>	<b>3</b>
3.1	Class List . . . . .	3
<b>4</b>	<b>Namespace Documentation</b>	<b>4</b>
4.1	LibTopoART Namespace Reference . . . . .	4
4.1.1	Enumeration Type Documentation . . . . .	6
4.2	LibTopoART_samples Namespace Reference . . . . .	6
<b>5</b>	<b>Class Documentation</b>	<b>7</b>
5.1	LibTopoART.Episodic_TopoART Class Reference . . . . .	7
5.1.1	Detailed Description . . . . .	8
5.1.2	Constructor & Destructor Documentation . . . . .	8
5.1.3	Member Function Documentation . . . . .	9
5.1.4	Member Data Documentation . . . . .	10
5.2	LibTopoART_samples.Episodic_TopoART_sample1 Class Reference . . . . .	11
5.2.1	Detailed Description . . . . .	11
5.3	LibTopoART_samples.Episodic_TopoART_sample2 Class Reference . . . . .	11
5.3.1	Detailed Description . . . . .	11
5.4	LibTopoART.F2_output Class Reference . . . . .	11
5.4.1	Detailed Description . . . . .	12
5.4.2	Constructor & Destructor Documentation . . . . .	12
5.4.3	Member Data Documentation . . . . .	12
5.5	LibTopoART.Fast_TopoART Class Reference . . . . .	13
5.5.1	Detailed Description . . . . .	14
5.5.2	Constructor & Destructor Documentation . . . . .	14

5.5.3	Member Function Documentation	15
5.6	LibTopoART.Fast_TopoART_base Class Reference	15
5.6.1	Detailed Description	17
5.6.2	Member Function Documentation	17
5.6.3	Member Data Documentation	20
5.6.4	Property Documentation	20
5.7	LibTopoART.Fast_TopoART_C Class Reference	21
5.7.1	Detailed Description	21
5.7.2	Constructor & Destructor Documentation	22
5.7.3	Member Function Documentation	22
5.7.4	Member Data Documentation	24
5.8	LibTopoART.Hypersphere_TopoART Class Reference	24
5.8.1	Detailed Description	25
5.8.2	Constructor & Destructor Documentation	25
5.8.3	Member Data Documentation	26
5.9	LibTopoART.IAdaptationStateCheck Interface Reference	27
5.9.1	Detailed Description	27
5.9.2	Member Function Documentation	27
5.10	LibTopoART.IEpisodic_recall Interface Reference	28
5.10.1	Detailed Description	28
5.10.2	Member Function Documentation	28
5.11	LibTopoART.InvalidClassIDException Class Reference	30
5.11.1	Detailed Description	30
5.12	LibTopoART.InvalidStateException Class Reference	30
5.12.1	Detailed Description	30
5.13	LibTopoART.ITopoART Interface Reference	31
5.13.1	Detailed Description	31
5.13.2	Member Function Documentation	32
5.13.3	Property Documentation	34
5.14	LibTopoART.ITopoART_C Interface Reference	34

5.14.1 Detailed Description . . . . .	34
5.14.2 Member Function Documentation . . . . .	35
5.15 LibTopoART.ITopoART_R Interface Reference . . . . .	36
5.15.1 Detailed Description . . . . .	36
5.15.2 Member Function Documentation . . . . .	37
5.16 LibTopoART.LibTopoART_info Struct Reference . . . . .	38
5.16.1 Detailed Description . . . . .	38
5.16.2 Member Data Documentation . . . . .	38
5.17 LibTopoART.TopoART Class Reference . . . . .	39
5.17.1 Detailed Description . . . . .	41
5.17.2 Constructor & Destructor Documentation . . . . .	41
5.17.3 Member Function Documentation . . . . .	42
5.17.4 Member Data Documentation . . . . .	45
5.17.5 Property Documentation . . . . .	45
5.18 LibTopoART.TopoART_C Class Reference . . . . .	46
5.18.1 Detailed Description . . . . .	47
5.18.2 Constructor & Destructor Documentation . . . . .	47
5.18.3 Member Function Documentation . . . . .	47
5.18.4 Member Data Documentation . . . . .	49
5.19 LibTopoART.TopoART_C_Prediction Struct Reference . . . . .	49
5.19.1 Detailed Description . . . . .	50
5.19.2 Constructor & Destructor Documentation . . . . .	50
5.19.3 Member Data Documentation . . . . .	50
5.20 LibTopoART_samples.TopoART_C_sample1 Class Reference . . . . .	51
5.20.1 Detailed Description . . . . .	51
5.21 LibTopoART_samples.TopoART_C_sample2 Class Reference . . . . .	51
5.21.1 Detailed Description . . . . .	51
5.22 LibTopoART.TopoART_R Class Reference . . . . .	51
5.22.1 Detailed Description . . . . .	52
5.22.2 Constructor & Destructor Documentation . . . . .	52

5.22.3	Member Function Documentation . . . . .	53
5.22.4	Member Data Documentation . . . . .	54
5.23	LibTopoART.TopoART_R_Prediction Struct Reference . . . . .	55
5.23.1	Detailed Description . . . . .	55
5.23.2	Constructor & Destructor Documentation . . . . .	55
5.23.3	Member Function Documentation . . . . .	56
5.23.4	Member Data Documentation . . . . .	56
5.24	LibTopoART_samples.TopoART_R_sample1 Class Reference . . . . .	56
5.24.1	Detailed Description . . . . .	57
5.25	LibTopoART_samples.TopoART_R_sample2 Class Reference . . . . .	57
5.25.1	Detailed Description . . . . .	57
5.26	LibTopoART_samples.TopoART_sample1 Class Reference . . . . .	57
5.26.1	Detailed Description . . . . .	57
5.27	LibTopoART_samples.TopoART_sample2 Class Reference . . . . .	57
5.27.1	Detailed Description . . . . .	57
<b>Index</b>		<b>59</b>

# 1 Namespace Index

## 1.1 Packages

Here are the packages with brief descriptions (if available):

<b>LibTopoART</b>	<b>4</b>
<b>LibTopoART_samples</b>	<b>6</b>

## 2 Hierarchical Index

### 2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

<b>LibTopoART_samples.Episodic_TopoART_sample1</b>	<b>11</b>
<b>LibTopoART_samples.Episodic_TopoART_sample2</b>	<b>11</b>
Exception	

LibTopoART.InvalidClassIDException	30
LibTopoART.InvalidStateException	30
LibTopoART.F2_output	11
LibTopoART.IAdaptationStateCheck	27
LibTopoART.ITopoART	31
LibTopoART.Fast_TopoART_base	15
LibTopoART.Episodic_TopoART	7
LibTopoART.Fast_TopoART	13
LibTopoART.Fast_TopoART_C	21
LibTopoART.ITopoART_C	34
LibTopoART.Fast_TopoART_C	21
LibTopoART.TopoART_C	46
LibTopoART.ITopoART_R	36
LibTopoART.TopoART_R	51
LibTopoART.TopoART	39
LibTopoART.Hypersphere_TopoART	24
LibTopoART.TopoART_C	46
LibTopoART.TopoART_R	51
IDisposable	
LibTopoART.Fast_TopoART_base	15
LibTopoART.TopoART	39
LibTopoART.IEpisodic_recall	28
LibTopoART.Episodic_TopoART	7
LibTopoART.LibTopoART_info	38
LibTopoART.TopoART_C_Prediction	49
LibTopoART_samples.TopoART_C_sample1	51
LibTopoART_samples.TopoART_C_sample2	51
LibTopoART.TopoART_R_Prediction	55
LibTopoART_samples.TopoART_R_sample1	56
LibTopoART_samples.TopoART_R_sample2	57
LibTopoART_samples.TopoART_sample1	57
LibTopoART_samples.TopoART_sample2	57

## 3 Class Index

### 3.1 Class List

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

<a href="#">LibTopoART.Episodic_TopoART</a>	
Class <a href="#">Episodic_TopoART</a> provides an implementation of the Episodic <a href="#">TopoART</a> neural network as proposed in "Marko Tscherepanow, Sina Kühnel, and Sören Riechers, 'Episodic Clustering of Data Streams Using a Topology-Learning Neural Network', European Conference on Artificial Intelligence (ECAI), Workshop on Active and Incremental Learning (AIL), pp. 24-29, 2012."	7
<a href="#">LibTopoART_samples.Episodic_TopoART_sample1</a>	
Sample using synthetic two-dimensional data	11
<a href="#">LibTopoART_samples.Episodic_TopoART_sample2</a>	
Sample using real-world video data	11
<a href="#">LibTopoART.F2_output</a>	
Class <a href="#">F2_output</a> provides the output of a single <a href="#">TopoART</a> module. It is a compressed version of the output vectors y and c.	11
<a href="#">LibTopoART.Fast_TopoART</a>	
Class <a href="#">Fast_TopoART</a> provides an implementation of the <a href="#">TopoART</a> neural network as proposed in "Tscherepanow, Marko (2010). <a href="#">TopoART</a> : A topology learning hierarchical ART network. In Proceedings of the International Conference on Artificial Neural Networks, LNCS 6354 (pp. 157–167). Berlin, Germany: Springer."	13
<a href="#">LibTopoART.Fast_TopoART_base</a>	
Base class providing functionality common to several <a href="#">TopoART</a> networks.	15
<a href="#">LibTopoART.Fast_TopoART_C</a>	
Class <a href="#">Fast_TopoART_C</a> provides an implementation of the TopoART-C neural network as proposed in "Marko Tscherepanow and Sören Riechers, 'An Incremental On-line Classifier for Imbalanced, Incomplete, and Noisy Data', European Conference on Artificial Intelligence (ECAI), Workshop on Active and Incremental Learning (AIL), pp. 18-23, 2012."	21
<a href="#">LibTopoART.Hypersphere_TopoART</a>	
Class <a href="#">Hypersphere_TopoART</a> provides an implementation of the Hypersphere <a href="#">TopoART</a> neural network as proposed in "Tscherepanow, Marko (2012). Incremental On-line Clustering with a Topology-Learning Hierarchical ART Neural Network Using Hyperspherical Categories. In Poster and Industry Proceedings of the Industrial Conference on Data Mining (ICDM), pp. 22–34. Fockendorf, Germany: ibai-publishing."	24
<a href="#">LibTopoART.IAdaptationStateCheck</a>	
Interface enabling checks whether certain adaptations of a network occurred.	27
<a href="#">LibTopoART.IEpisodic_recall</a>	
Interface summarising the episodic recall functionality.	28
<a href="#">LibTopoART.InvalidClassIDException</a>	
Exception signalling an invalid class ID.	30
<a href="#">LibTopoART.InvalidStateException</a>	
Exception signalling an invalid state of the neural network.	30
<a href="#">LibTopoART.ITopoART</a>	
Interface summarising the basic <a href="#">TopoART</a> functionality.	31

<a href="#">LibTopoART.ITopoART_C</a>	
Interface summarising the TopoART-C functionality.	34
<a href="#">LibTopoART.ITopoART_R</a>	
Interface summarising the TopoART-R functionality.	36
<a href="#">LibTopoART.LibTopoART_info</a>	
Struct <a href="#">LibTopoART_info</a> provides some metainformation regarding the respective implementation of <a href="#">LibTopoART</a> .	38
<a href="#">LibTopoART.TopoART</a>	
Class <a href="#">TopoART</a> provides an implementation of the <a href="#">TopoART</a> neural network as proposed in "Tscherepanow, Marko (2010). <a href="#">TopoART</a> : A topology learning hierarchical ART network. In Proceedings of the International Conference on Artificial Neural Networks (ICANN), LNCS 6354, pp. 157–167. Berlin, Germany: Springer."	39
<a href="#">LibTopoART.TopoART_C</a>	
Class <a href="#">TopoART_C</a> provides an implementation of the TopoART-C neural network as proposed in "Marko Tscherepanow and Sören Riechers, 'An Incremental On-line Classifier for Imbalanced, Incomplete, and Noisy Data', European Conference on Artificial Intelligence (ECAI), Workshop on Active and Incremental Learning (AIL), pp. 18-23, 2012."	46
<a href="#">LibTopoART.TopoART_C_Prediction</a>	
Struct <a href="#">TopoART_C_Prediction</a> contains a prediction made by a TopoART-C network.	49
<a href="#">LibTopoART_samples.TopoART_C_sample1</a>	
Classification using TopoART-C. (simple classification task)	51
<a href="#">LibTopoART_samples.TopoART_C_sample2</a>	
Sample using artificial two-dimensional data with associated class IDs	51
<a href="#">LibTopoART.TopoART_R</a>	
Class <a href="#">TopoART_R</a> provides an implementation of the TopoART-R neural network as proposed in "Tscherepanow, Marko (2011). An Extended <a href="#">TopoART</a> Network for the Stable On-Line Learning of Regression Functions. In Proceedings of the International Conference on Neural Information Processing (ICONIP), LNCS 7063, pp. 562–571. Berlin, Germany: Springer."	51
<a href="#">LibTopoART.TopoART_R_Prediction</a>	
Struct <a href="#">TopoART_R_Prediction</a> contains a prediction made by a TopoART-R network.	55
<a href="#">LibTopoART_samples.TopoART_R_sample1</a>	
Regression using TopoART-R. (simplified version)	56
<a href="#">LibTopoART_samples.TopoART_R_sample2</a>	
Regression using TopoART-R. (advanced version)	57
<a href="#">LibTopoART_samples.TopoART_sample1</a>	
Simple TopoART sample	57
<a href="#">LibTopoART_samples.TopoART_sample2</a>	
Sample using artificial two-dimensional data	57

## 4 Namespace Documentation

### 4.1 LibTopoART Namespace Reference

#### Classes

- class [Episodic\\_TopoART](#)



Class [Episodic\\_TopoART](#) provides an implementation of the Episodic [TopoART](#) neural network as proposed in "Marko Tscherepanow, Sina Kühnel, and Sören Riechers, 'Episodic Clustering of Data Streams Using a Topology-Learning Neural Network', European Conference on Artificial Intelligence (ECAI), Workshop on Active and Incremental Learning (AIL), pp. 24-29, 2012."

- class [F2\\_output](#)

Class [F2\\_output](#) provides the output of a single [TopoART](#) module. It is a compressed version of the output vectors  $y$  and  $c$ .

- class [Fast\\_TopoART](#)

Class [Fast\\_TopoART](#) provides an implementation of the [TopoART](#) neural network as proposed in "Tscherepanow, Marko (2010). [TopoART](#): A topology learning hierarchical ART network. In Proceedings of the International Conference on Artificial Neural Networks, LNCS 6354 (pp. 157–167). Berlin, Germany: Springer."

- class [Fast\\_TopoART\\_base](#)

Base class providing functionality common to several [TopoART](#) networks.

- class [Fast\\_TopoART\\_C](#)

Class [Fast\\_TopoART\\_C](#) provides an implementation of the TopoART-C neural network as proposed in "Marko Tscherepanow and Sören Riechers, 'An Incremental On-line Classifier for Imbalanced, Incomplete, and Noisy Data', European Conference on Artificial Intelligence (ECAI), Workshop on Active and Incremental Learning (AIL), pp. 18-23, 2012."

- class [Hypersphere\\_TopoART](#)

Class [Hypersphere\\_TopoART](#) provides an implementation of the Hypersphere [TopoART](#) neural network as proposed in "Tscherepanow, Marko (2012). Incremental On-line Clustering with a Topology-Learning Hierarchical ART Neural Network Using Hyperspherical Categories. In Poster and Industry Proceedings of the Industrial Conference on Data Mining (ICDM), pp. 22–34. Fockendorf, Germany: ibai-publishing."

- interface [IAdaptationStateCheck](#)

Interface enabling checks whether certain adaptations of a network occurred.

- interface [IEpisodic\\_recall](#)

Interface summarising the episodic recall functionality.

- class [InvalidClassIDException](#)

Exception signalling an invalid class ID.

- class [InvalidStateException](#)

Exception signalling an invalid state of the neural network.

- interface [ITopoART](#)

Interface summarising the basic [TopoART](#) functionality.

- interface [ITopoART\\_C](#)

Interface summarising the TopoART-C functionality.

- interface [ITopoART\\_R](#)

Interface summarising the TopoART-R functionality.

- struct [LibTopoART\\_info](#)

Struct [LibTopoART\\_info](#) provides some metainformation regarding the respective implementation of [LibTopoART](#).

- class [TopoART](#)

Class [TopoART](#) provides an implementation of the [TopoART](#) neural network as proposed in "Tscherepanow, Marko (2010). [TopoART](#): A topology learning hierarchical ART network. In Proceedings of the International Conference on Artificial Neural Networks (ICANN), LNCS 6354, pp. 157–167. Berlin, Germany: Springer."

- class [TopoART\\_C](#)

Class [TopoART\\_C](#) provides an implementation of the TopoART-C neural network as proposed in "Marko Tscherepanow and Sören Riechers, 'An Incremental On-line Classifier for Imbalanced, Incomplete, and Noisy Data', European Conference on Artificial Intelligence (ECAI), Workshop on Active and Incremental Learning (AIL), pp. 18-23, 2012."

- struct [TopoART\\_C\\_Prediction](#)

Struct [TopoART\\_C\\_Prediction](#) contains a prediction made by a TopoART-C network.

- class [TopoART\\_R](#)

Class [TopoART\\_R](#) provides an implementation of the TopoART-R neural network as proposed in "Tscherepanow, Marko (2011). An Extended [TopoART](#) Network for the Stable On-Line Learning of Regression Functions. In Proceedings of the International Conference on Neural Information Processing (ICONIP), LNCS 7063, pp. 562–571. Berlin, Germany: Springer."

- struct [TopoART\\_R\\_Prediction](#)

*Struct [TopoART\\_R\\_Prediction](#) contains a prediction made by a TopoART-R network.*

## Enumerations

- enum [AdaptationState](#) {  
[AdaptationState.NO\\_ADAPTATION](#) = 0, [AdaptationState.ADDED\\_NODE\\_CANDIDATE](#) = 0x0001,  
[AdaptationState.ADAPTED\\_NONPERMANENT\\_WEIGHT](#) = 0x0002, [AdaptationState.ADDED\\_EDGE\\_CANDIDATE](#) = 0x0004,  
[AdaptationState.REMOVED\\_NODE\\_CANDIDATE](#) = 0x0008, [AdaptationState.REMOVED\\_EDGE\\_CANDIDATE](#) = 0x0010, [AdaptationState.ANY\\_NONPERMANENT\\_ADAPTATION\\_MASK](#) = 0x00ff, [AdaptationState.ADDED\\_PERMANENT\\_NODE](#) = 0x0100,  
[AdaptationState.ADAPTED\\_PERMANENT\\_WEIGHT](#) = 0x0200, [AdaptationState.ADDED\\_PERMANENT\\_EDGE](#) = 0x0400, [AdaptationState.ANY\\_PERMANENT\\_ADAPTATION\\_MASK](#) = 0xff00 }

*Enumeration specifying possible adaptation states.*

### 4.1.1 Enumeration Type Documentation

#### 4.1.1.1 AdaptationState

enum [LibTopoART.AdaptationState](#) [strong]

Enumeration specifying possible adaptation states.

#### Enumerator

<a href="#">NO_ADAPTATION</a>	No adaptation occurred.
<a href="#">ADDED_NODE_CANDIDATE</a>	Added one or more node candidates.
<a href="#">ADAPTED_NONPERMANENT_WEIGHT</a>	The change of at least a single weight of one node candidate exceeds the given threshold.
<a href="#">ADDED_EDGE_CANDIDATE</a>	Added an edge from/to a node candidate.
<a href="#">REMOVED_NODE_CANDIDATE</a>	Removed one or more node candidates.
<a href="#">REMOVED_EDGE_CANDIDATE</a>	Removed one or more node candidates.
<a href="#">ANY_NONPERMANENT_ADAPTATION_MASK</a>	Mask for all non-permanent adaptations.
<a href="#">ADDED_PERMANENT_NODE</a>	Added one or more permanent nodes.
<a href="#">ADAPTED_PERMANENT_WEIGHT</a>	The change of at least a single weight of one permanent node exceeds the given threshold.
<a href="#">ADDED_PERMANENT_EDGE</a>	Added an edge between two permanent nodes.
<a href="#">ANY_PERMANENT_ADAPTATION_MASK</a>	Mask for all permanent adaptations.

## 4.2 LibTopoART\_samples Namespace Reference

### Classes

- class [Episodic\\_TopoART\\_sample1](#)

*Sample using synthetic two-dimensional data.*

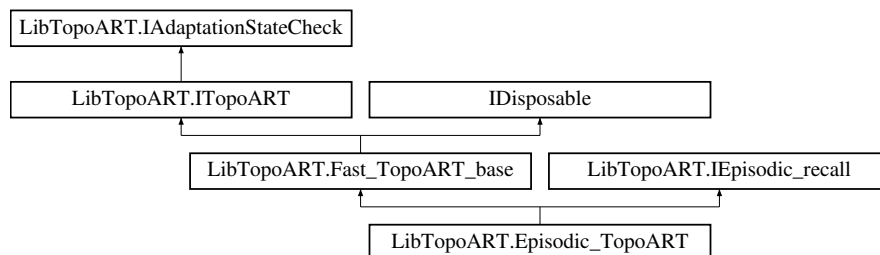
- class [Episodic\\_TopoART\\_sample2](#)  
*Sample using real-world video data.*
- class [TopoART\\_C\\_sample1](#)  
*Classification using TopoART-C. (simple classification task)*
- class [TopoART\\_C\\_sample2](#)  
*Sample using artificial two-dimensional data with associated class IDs.*
- class [TopoART\\_R\\_sample1](#)  
*Regression using TopoART-R. (simplified version)*
- class [TopoART\\_R\\_sample2](#)  
*Regression using TopoART-R. (advanced version)*
- class [TopoART\\_sample1](#)  
*Simple TopoART sample.*
- class [TopoART\\_sample2](#)  
*Sample using artificial two-dimensional data.*

## 5 Class Documentation

### 5.1 LibTopoART.Episodic\_TopoART Class Reference

Class [Episodic\\_TopoART](#) provides an implementation of the Episodic [TopoART](#) neural network as proposed in "Marko Tscherepanow, Sina Kühnel, and Sören Riechers, 'Episodic Clustering of Data Streams Using a Topology-Learning Neural Network', European Conference on Artificial Intelligence (ECAI), Workshop on Active and Incremental Learning (AIL), pp. 24-29, 2012."

Inheritance diagram for LibTopoART.Episodic\_TopoART:



#### Public Member Functions

- [Episodic\\_TopoART](#) (long input\_dimension, long module\_number, decimal rho\_a, long t\_max)  
*This constructor initialises an Episodic [TopoART](#) network.*
- [Episodic\\_TopoART](#) (string path)  
*This constructor loads a saved Episodic [TopoART](#) network.*
- override void [Learn](#) (decimal[] input)  
*This method performs a single training step.*
- long [BeginRecall](#) (decimal[] stimulus)  
*This method starts the recall process.*
- bool [InterEpisodeRecallStep](#) (out decimal[] recall\_result, out decimal F3\_activation)  
*This method performs a single inter-episode recall step and sets the starting point for intra-episode recall.*
- bool [IntraEpisodeRecallStep](#) (out decimal[] recall\_result)  
*This method performs a single intra-episode recall step.*
- void [EndRecall](#) ()  
*This method stops the recall process and frees temporary resources.*

## Public Attributes

- new const decimal `file_format_version` = 0.01m

Instance variable `file_format_version` represents the version of the file format used by class `Episodic_TopoART`.

## Properties

- long `T_max` [get]

Property `T_max` represents the considered time frame.

## Additional Inherited Members

### 5.1.1 Detailed Description

Class `Episodic_TopoART` provides an implementation of the Episodic `TopoART` neural network as proposed in "Marko Tscherepanow, Sina Kühnel, and Sören Riechers, 'Episodic Clustering of Data Streams Using a Topology-Learning Neural Network', European Conference on Artificial Intelligence (ECAI), Workshop on Active and Incremental Learning (AIL), pp. 24-29, 2012."

### 5.1.2 Constructor & Destructor Documentation

#### 5.1.2.1 `Episodic_TopoART()` [1/2]

```
LibTopoART.Episodic_TopoART.Episodic_TopoART (
    long input_dimension,
    long module_number,
    decimal rho_a,
    long t_max )
```

This constructor initialises an Episodic `TopoART` network.

#### Parameters

<code>input_dimension</code>	The dimension of input vectors to be learnt.
<code>module_number</code>	The number of Episodic <code>TopoART</code> modules.
<code>rho_a</code>	The vigilance parameter of the first Episodic <code>TopoART</code> module (ETA a).
<code>t_max</code>	The parameter limiting the considered time frame.

#### 5.1.2.2 `Episodic_TopoART()` [2/2]

```
LibTopoART.Episodic_TopoART.Episodic_TopoART (
    string path )
```

This constructor loads a saved Episodic `TopoART` network.

## Parameters

<i>path</i>	The path of a binary Episodic <a href="#">TopoART</a> file.
-------------	---

## 5.1.3 Member Function Documentation

## 5.1.3.1 BeginRecall()

```
long LibTopoART.Episodic_TopoART.BeginRecall (
    decimal [ ] stimulus )
```

This method starts the recall process.

## Parameters

<i>stimulus</i>	The stimulus (input) which is used to trigger recall.
-----------------	---

## Returns

The number of F3 nodes created.

Implements [LibTopoART.IEpisodic\\_recall](#).

## 5.1.3.2 EndRecall()

```
void LibTopoART.Episodic_TopoART.EndRecall ( )
```

This method stops the recall process and frees temporary resources.

Implements [LibTopoART.IEpisodic\\_recall](#).

## 5.1.3.3 InterEpisodeRecallStep()

```
bool LibTopoART.Episodic_TopoART.InterEpisodeRecallStep (
    out decimal [ ] recall_result,
    out decimal F3_activation )
```

This method performs a single inter-episode recall step and sets the starting point for intra-episode recall.

## Parameters

<i>recall_result</i>	Returns the recall output vector for the current step.
<i>F3_activation</i>	Returns the activation of the current F3 node.

**Returns**

A boolean result indicating whether the recall step was successfully completed, or not.

Implements [LibTopoART.IEpisodic\\_recall](#).

**5.1.3.4 IntraEpisodeRecallStep()**

```
bool LibTopoART.Episodic_TopoART.IntraEpisodeRecallStep (
    out decimal [] recall_result )
```

This method performs a single intra-episode recall step.

**Parameters**

<i>recall_result</i>	Returns the recall output vector for the current step.
----------------------	--

**Returns**

A boolean result indicating whether the recall step was successfully completed, or not.

Implements [LibTopoART.IEpisodic\\_recall](#).

**5.1.3.5 Learn()**

```
override void LibTopoART.Episodic_TopoART.Learn (
    decimal [] input ) [virtual]
```

This method performs a single training step.

The spatial weights are adapted as in the original [TopoART](#) network. In contrast, the adaptation of the temporal weight  $w_{\{j,2\}^{F2,t}}$  occurring only in Episodic [TopoART](#) is slightly different↵  

$$: w_{\{j,2\}^{F2,t}}(t+1) = \text{beta\_j} * \text{Max}(t\_2^{F1}(t), w_{\{j,2\}^{F2,t}}(t) + (1 - \text{beta\_j}) * w_{\{j,2\}^{F2,t}}(t) \text{ for } j = \text{bm or } j = \text{sbm. (Note: } w_{\{j,1\}^{F2,t}} \text{ remains constant over the life time of a node.)}$$

**Parameters**

<i>input</i>	The input vector to be learnt.
--------------	--------------------------------

Implements [LibTopoART.Fast\\_TopoART\\_base](#).

**5.1.4 Member Data Documentation**

### 5.1.4.1 file\_format\_version

```
new const decimal LibTopoART.Episodic_TopoART.file_format_version = 0.01m
```

Instance variable `file_format_version` represents the version of the file format used by class [Episodic\\_TopoART](#).

## 5.2 LibTopoART\_samples.Episodic\_TopoART\_sample1 Class Reference

Sample using synthetic two-dimensional data.

### 5.2.1 Detailed Description

Sample using synthetic two-dimensional data.

Like in Section 4.1 of "Marko Tscherepanow, Sina Kühnel, and Sören Riechers, 'Episodic Clustering of Data Streams Using a Topology-Learning Neural Network', European Conference on Artificial Intelligence (ECAI), Workshop on Active and Incremental Learning (AIL), pp. 24-29, 2012.", an Episodic TopoART network is trained with the well-known Two Spirals dataset. Due to the incorporation of temporal information during learning, Episodic TopoART is capable of creating two clusters each representing one spiral in an unsupervised way. These clusters are formed by the nodes of module b (ETA b).

The resulting network can be visualised using the R script `ShowEpisodicTopoARTResults.R` provided in the subfolder R.

## 5.3 LibTopoART\_samples.Episodic\_TopoART\_sample2 Class Reference

Sample using real-world video data.

### 5.3.1 Detailed Description

Sample using real-world video data.

Like in Section 4.2 of "Marko Tscherepanow, Sina Kühnel, and Sören Riechers, 'Episodic Clustering of Data Streams Using a Topology-Learning Neural Network', European Conference on Artificial Intelligence (ECAI), Workshop on Active and Incremental Learning (AIL), pp. 24-29, 2012.", an Episodic TopoART network is trained with real-world video data. Each image has a size of 64x36 pixels. As each pixel comprises 3 color channels (RGB), the input dimension equals 6912. After finishing training, recall is performed for a single input stimulus.

The recall results can be visualised using the R script `ShowEpisodicTopoARTRecallResults.R` provided in the subfolder R.

## 5.4 LibTopoART.F2\_output Class Reference

Class `F2_output` provides the output of a single [TopoART](#) module. It is a compressed version of the output vectors `y` and `c`.

## Public Member Functions

- [F2\\_output](#) ()

*This constructor sets all instance variables of class [F2\\_output](#) to [LibTopoART\\_info.UNDEFINED](#).*

## Public Attributes

- decimal [bm\\_node\\_activation](#)

*Instance variable `bm_node_activation` represents the activation of the best-matching node (prediction variant).*

- long [bm\\_node\\_ID](#)

*Instance variable `bm_node_ID` represents the ID of the best-matching node.*

- long [bm\\_cluster\\_ID](#)

*Instance variable `bm_cluster_ID` represents the cluster ID of the best-matching node.*

- decimal [bm\\_permanent\\_node\\_activation](#)

*Instance variable `bm_permanent_node_activation` represents the activation of the best-matching permanent node (prediction variant).*

- long [bm\\_permanent\\_node\\_ID](#)

*Instance variable `bm_permanent_node_ID` represents the ID of the best-matching permanent node.*

- long [bm\\_permanent\\_cluster\\_ID](#)

*Instance variable `bm_permanent_cluster_ID` represents the cluster ID of the best-matching permanent node.*

### 5.4.1 Detailed Description

Class [F2\\_output](#) provides the output of a single [TopoART](#) module. It is a compressed version of the output vectors `y` and `c`.

### 5.4.2 Constructor & Destructor Documentation

#### 5.4.2.1 [F2\\_output](#)()

```
LibTopoART.F2_output.F2_output ( )
```

This constructor sets all instance variables of class [F2\\_output](#) to [LibTopoART\\_info.UNDEFINED](#).

### 5.4.3 Member Data Documentation

#### 5.4.3.1 [bm\\_cluster\\_ID](#)

```
long LibTopoART.F2_output.bm_cluster_ID
```

Instance variable `bm_cluster_ID` represents the cluster ID of the best-matching node.



5.4.3.2 `bm_node_activation`

```
decimal LibTopoART.F2_output.bm_node_activation
```

Instance variable `bm_node_activation` represents the activation of the best-matching node (prediction variant).

5.4.3.3 `bm_node_ID`

```
long LibTopoART.F2_output.bm_node_ID
```

Instance variable `bm_node_ID` represents the ID of the best-matching node.

5.4.3.4 `bm_permanent_cluster_ID`

```
long LibTopoART.F2_output.bm_permanent_cluster_ID
```

Instance variable `bm_permanent_cluster_ID` represents the cluster ID of the best-matching permanent node.

5.4.3.5 `bm_permanent_node_activation`

```
decimal LibTopoART.F2_output.bm_permanent_node_activation
```

Instance variable `bm_permanent_node_activation` represents the activation of the best-matching permanent node (prediction variant).

5.4.3.6 `bm_permanent_node_ID`

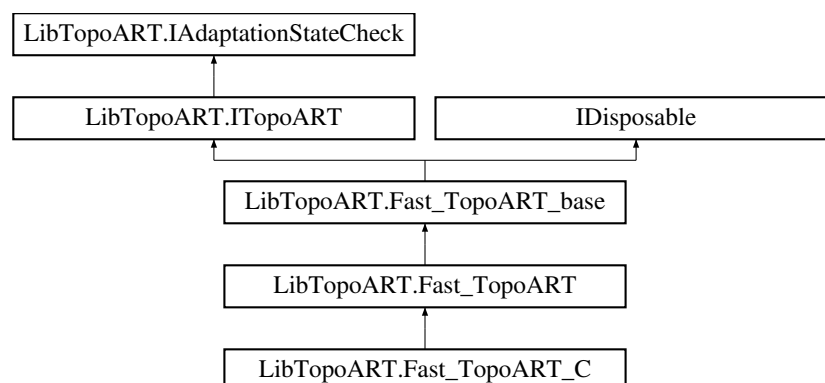
```
long LibTopoART.F2_output.bm_permanent_node_ID
```

Instance variable `bm_permanent_node_ID` represents the ID of the best-matching permanent node.

## 5.5 LibTopoART.Fast\_TopoART Class Reference

Class `Fast_TopoART` provides an implementation of the `TopoART` neural network as proposed in "Tscherepanow, Marko (2010). `TopoART`: A topology learning hierarchical ART network. In Proceedings of the International Conference on Artificial Neural Networks, LNCS 6354 (pp. 157–167). Berlin, Germany: Springer."

Inheritance diagram for `LibTopoART.Fast_TopoART`:



## Public Member Functions

- [Fast\\_TopoART](#) (long input\_dimension, long module\_number, decimal rho\_a)  
*This constructor initialises a [TopoART](#) network.*
- [Fast\\_TopoART](#) (string path)  
*This constructor loads a saved [TopoART](#) network.*
- override void [Learn](#) (decimal[] input)  
*This method performs a single training step.*

## Additional Inherited Members

### 5.5.1 Detailed Description

Class [Fast\\_TopoART](#) provides an implementation of the [TopoART](#) neural network as proposed in "Tscherepanow, Marko (2010). [TopoART](#): A topology learning hierarchical ART network. In Proceedings of the International Conference on Artificial Neural Networks, LNCS 6354 (pp. 157–167). Berlin, Germany: Springer."

Internally, real-valued data are mapped to `long` variables. Therefore, computations are accelerated but less accurate. As a consequence, the results may differ slightly from class [TopoART](#).

Class [Fast\\_TopoART](#) requires all input to lie in the interval [0,1].

### 5.5.2 Constructor & Destructor Documentation

#### 5.5.2.1 [Fast\\_TopoART\(\)](#) [1/2]

```
LibTopoART.Fast_TopoART.Fast_TopoART (
    long input_dimension,
    long module_number,
    decimal rho_a )
```

This constructor initialises a [TopoART](#) network.

#### Parameters

<i>input_dimension</i>	The dimension of input vectors to be learnt.
<i>module_number</i>	The number of <a href="#">TopoART</a> modules.
<i>rho_a</i>	The vigilance parameter of the first <a href="#">TopoART</a> module (TA a).

#### 5.5.2.2 [Fast\\_TopoART\(\)](#) [2/2]

```
LibTopoART.Fast_TopoART.Fast_TopoART (
    string path )
```

This constructor loads a saved [TopoART](#) network.

## Parameters

<i>path</i>	The path of a binary <a href="#">TopoART</a> file.
-------------	--

## 5.5.3 Member Function Documentation

## 5.5.3.1 Learn()

```
override void LibTopoART.Fast_TopoART.Learn (
    decimal [] input ) [virtual]
```

This method performs a single training step.

## Parameters

<i>input</i>	The input vector to be learnt.
--------------	--------------------------------

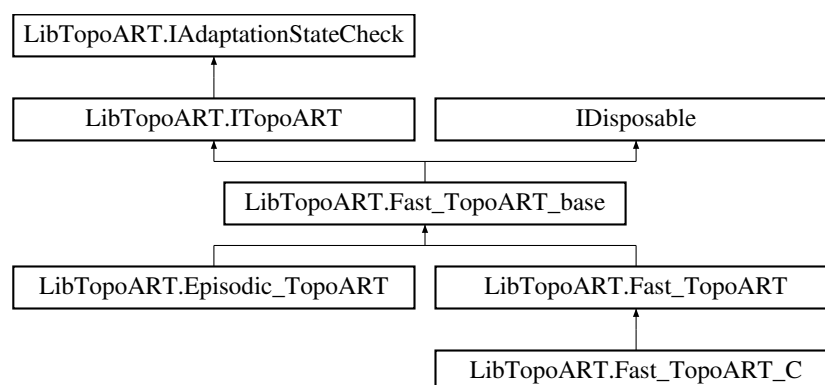
Implements [LibTopoART.Fast\\_TopoART\\_base](#).

Reimplemented in [LibTopoART.Fast\\_TopoART\\_C](#).

## 5.6 LibTopoART.Fast\_TopoART\_base Class Reference

Base class providing functionality common to several [TopoART](#) networks.

Inheritance diagram for LibTopoART.Fast\_TopoART\_base:



## Public Member Functions

- abstract void [Learn](#) (decimal[] input)  
*This method performs a single training step.*
- void [Dispose](#) ()  
*Releases all resources used by the [LibTopoART.Fast\\_TopoART\\_base](#) object.*

- void [ComputeClusterIDs](#) ()  
*This method computes the cluster IDs for all neurons.*
- [F2\\_output](#) [] [GetBMOutput](#) (decimal[] input)  
*This method finds the closest category for a given test input.*
- [F2\\_output](#) [] [GetBMOutput](#) (decimal[] input, bool[] mask\_vector)  
*This method finds the closest category for a given test input.*
- void [SaveText](#) (string path)  
*This method saves the entire network as a text file.*
- void [Save](#) (string path)  
*This method saves the entire network as a binary file.*
- void [ResetAdaptationState](#) ()  
*This method resets the adaptation state to [AdaptationState.NO\\_ADAPTATION](#).*
- [AdaptationState](#) [GetAdaptationState](#) (decimal epsilon=0.001m)  
*This method returns the current adaptation state.*

### Public Attributes

- const decimal [file\\_format\\_version](#) = 0.09m  
*Instance variable `file_format_version` represents the version of the file format used by class [Fast\\_TopoART](#).*
- const string [integer\\_base\\_type](#) = "long"  
*Instance variable `integer_base_type` provides a string containing the data type used for representing integer variables (IDs, parameters, counters, etc.) internally.*
- const string [float\\_base\\_type](#) = "int"  
*Instance variable `float_base_type` provides a string containing the data type used for representing floating point variables (input, weights, etc.) internally.*

### Protected Member Functions

- virtual void [Dispose](#) (bool disposing)  
*Release resources used by the [LibTopoART.Fast\\_TopoART\\_base](#) object.*

### Properties

- long [ModuleNum](#) [get]
- long [LearningSteps](#) [get]  
*Property `LearningSteps` represents the total number of performed learning steps.*
- decimal [Rho\\_a](#) [get]  
*Property `Rho_a` represents the vigilance parameter of the first [TopoART](#) module (TA a).*
- decimal [Beta\\_sbm](#) [get, set]  
*Property `Beta_sbm` represents the learning rate of the second best-matching nodes.*
- long [Tau](#) [get, set]  
*Property `Tau` represents the parameter tau required for the removal of nodes and edges.*
- long [Phi](#) [get, set]
- decimal [Alpha](#) [get, set]  
*Property `Alpha` represents the choice parameter alpha.*
- long [] [NodeNum](#) [get]  
*Property `NodeNum` represents the number of [TopoART](#) nodes used by each module.*
- long [] [ClusterNum](#) [get]  
*Property `ClusterNum` represents the number of [TopoART](#) clusters found by each module.*

### 5.6.1 Detailed Description

Base class providing functionality common to several [TopoART](#) networks.

### 5.6.2 Member Function Documentation

#### 5.6.2.1 ComputeClusterIDs()

```
void LibTopoART.Fast_TopoART_base.ComputeClusterIDs ( )
```

This method computes the cluster IDs for all neurons.

Implements [LibTopoART.ITopoART](#).

#### 5.6.2.2 Dispose() [1/2]

```
void LibTopoART.Fast_TopoART_base.Dispose ( )
```

Releases all resources used by the [LibTopoART.Fast\\_TopoART\\_base](#) object.

Call [Dispose\(\)](#) when you are finished using the [LibTopoART.Fast\\_TopoART\\_base](#). The [Dispose\(\)](#) method leaves the [LibTopoART.Fast\\_TopoART\\_base](#) in an unusable state. After calling [Dispose\(\)](#), you must release all references to the [LibTopoART.Fast\\_TopoART\\_base](#) so the garbage collector can reclaim the memory that the [LibTopoART.Fast\\_TopoART\\_base](#) was occupying.

#### 5.6.2.3 Dispose() [2/2]

```
virtual void LibTopoART.Fast_TopoART_base.Dispose (
    bool disposing ) [protected], [virtual]
```

Release resources used by the [LibTopoART.Fast\\_TopoART\\_base](#) object.

#### Parameters

<i>disposing</i>	If set to <code>true</code> all managed resources are released.
------------------	---

#### 5.6.2.4 GetAdaptationState()

```
AdaptationState LibTopoART.Fast_TopoART_base.GetAdaptationState (
    decimal epsilon = 0.001m )
```

This method returns the current adaptation state.

**Parameters**

<i>epsilon</i>	The threshold for weight adaptations to be considered.
----------------	--

**Returns**

An enumeration describing the adaptation state.

Implements [LibTopoART.IAdaptationStateCheck](#).

**5.6.2.5 GetBMOutput()** [1/2]

```
F2_output [ ] LibTopoART.Fast_TopoART_base.GetBMOutput (
    decimal [ ] input )
```

This method finds the closest category for a given test input.

**Parameters**

<i>input</i>	The input vector $x(t)$ .
--------------	---------------------------

**Returns**

An array of type [F2\\_output](#). Each entry contains the ID of the best-matching node and the corresponding cluster ID for one [TopoART](#) module.

Implements [LibTopoART.ITopoART](#).

**5.6.2.6 GetBMOutput()** [2/2]

```
F2_output [ ] LibTopoART.Fast_TopoART_base.GetBMOutput (
    decimal [ ] input,
    bool [ ] mask_vector )
```

This method finds the closest category for a given test input.

**Parameters**

<i>input</i>	The input vector $x(t)$ .
<i>mask_vector</i>	A mask vector excluding individual dimensions of $x(t)$ from the computation. (Setting an element of the mask vector to <code>true</code> , excludes the corresponding elements of $x(t)$ .)

**Returns**

An array of type [F2\\_output](#). Each entry contains the ID of the best-matching node and the corresponding cluster ID for one [TopoART](#) module.

Implements [LibTopoART.ITopoART](#).

#### 5.6.2.7 Learn()

```
abstract void LibTopoART.Fast_TopoART_base.Learn (
    decimal [] input ) [pure virtual]
```

This method performs a single training step.

##### Parameters

<i>input</i>	The input vector to be learnt.
--------------	--------------------------------

Implements [LibTopoART.ITopoART](#).

Implemented in [LibTopoART.Fast\\_TopoART\\_C](#), [LibTopoART.Episodic\\_TopoART](#), and [LibTopoART.Fast\\_TopoART](#).

#### 5.6.2.8 ResetAdaptationState()

```
void LibTopoART.Fast_TopoART_base.ResetAdaptationState ( )
```

This method resets the adaptation state to [AdaptationState.NO\\_ADAPTATION](#).

Implements [LibTopoART.IAdaptationStateCheck](#).

#### 5.6.2.9 Save()

```
void LibTopoART.Fast_TopoART_base.Save (
    string path )
```

This method saves the entire network as a binary file.

##### Parameters

<i>path</i>	A string representing the path of the file to save.
-------------	---

Implements [LibTopoART.ITopoART](#).

#### 5.6.2.10 SaveText()

```
void LibTopoART.Fast_TopoART_base.SaveText (
    string path )
```

This method saves the entire network as a text file.

**Parameters**

<i>path</i>	A string representing the path of the file to save.
-------------	---

Implements [LibTopoART.ITopoART](#).

**5.6.3 Member Data Documentation****5.6.3.1 file\_format\_version**

```
const decimal LibTopoART.Fast_TopoART_base.file_format_version = 0.09m
```

Instance variable `file_format_version` represents the version of the file format used by class [Fast\\_TopoART](#).

**5.6.3.2 float\_base\_type**

```
const string LibTopoART.Fast_TopoART_base.float_base_type = "int"
```

Instance variable `float_base_type` provides a string containing the data type used for representing floating point variables (input, weights, etc.) internally.

**5.6.3.3 integer\_base\_type**

```
const string LibTopoART.Fast_TopoART_base.integer_base_type = "long"
```

Instance variable `integer_base_type` provides a string containing the data type used for representing integer variables (IDs, parameters, counters, etc.) internally.

**5.6.4 Property Documentation****5.6.4.1 ModuleNum**

```
long LibTopoART.Fast_TopoART_base.ModuleNum [get]
```

Property `ModuleNum` represents the number of [TopoART](#) modules used. (The original [TopoART](#) uses two modules.)

**5.6.4.2 Phi**

```
long LibTopoART.Fast_TopoART_base.Phi [get], [set]
```

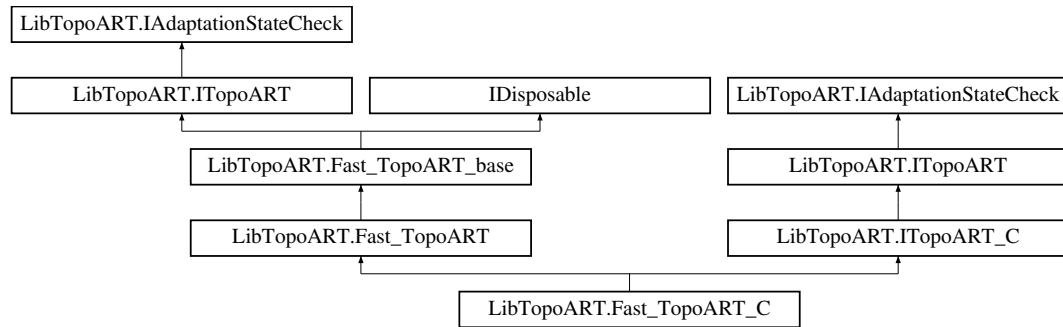
Property `Phi` represents the parameter `phi` required for the removal of nodes and edges as well as for the propagation of input to subsequent [TopoART](#) modules.



## 5.7 LibTopoART.Fast\_TopoART\_C Class Reference

Class [Fast\\_TopoART\\_C](#) provides an implementation of the TopoART-C neural network as proposed in "Marko Tscherepanow and Sören Riechers, 'An Incremental On-line Classifier for Imbalanced, Incomplete, and Noisy Data', European Conference on Artificial Intelligence (ECAI), Workshop on Active and Incremental Learning (AIL), pp. 18-23, 2012."

Inheritance diagram for LibTopoART.Fast\_TopoART\_C:



### Public Member Functions

- [Fast\\_TopoART\\_C](#) (long input\_dimension, long module\_number, decimal rho\_a)  
*This constructor initialises a TopoART-C network.*
- [Fast\\_TopoART\\_C](#) (string path)  
*This constructor loads a saved TopoART-C network.*
- override void [Learn](#) (decimal[] input)  
*This method performs a single training step and sets the class ID corresponding to input to `UNDEFINED_CLASS_ID`.*
- void [Learn](#) (decimal[] input, long class\_ID)  
*This method performs a single training step.*
- long [Predict](#) (decimal[] input, long nu)  
*This method predicts the class ID.*
- [TopoART\\_C\\_Prediction Predict](#) (decimal[] input, bool[] mask\_vector, long nu)  
*This method predicts the class ID.*

### Public Attributes

- const long [UNDEFINED\\_CLASS\\_ID](#) = -2  
*Instance variable `UNDEFINED_CLASS_ID` gives the value used for indicating that an input sample was predict to belong to the undefined class; i.e, no class ID was provided for such input samples during training.*
- new const decimal [file\\_format\\_version](#) = 0.01m  
*Instance variable `file_format_version` represents the version of the file format used by class [Fast\\_TopoART\\_C](#).*

### Additional Inherited Members

#### 5.7.1 Detailed Description

Class [Fast\\_TopoART\\_C](#) provides an implementation of the TopoART-C neural network as proposed in "Marko Tscherepanow and Sören Riechers, 'An Incremental On-line Classifier for Imbalanced, Incomplete, and Noisy Data', European Conference on Artificial Intelligence (ECAI), Workshop on Active and Incremental Learning (AIL), pp. 18-23, 2012."

Class [Fast\\_TopoART\\_C](#) requires all input except the class IDs to lie in the interval [0,1]. The class IDs are signed integer values.

## 5.7.2 Constructor & Destructor Documentation

### 5.7.2.1 Fast\_TopoART\_C() [1/2]

```
LibTopoART.Fast_TopoART_C.Fast_TopoART_C (
    long input_dimension,
    long module_number,
    decimal rho_a )
```

This constructor initialises a TopoART-C network.

#### Parameters

<i>input_dimension</i>	The dimension of input vectors to be learnt.
<i>module_number</i>	The number of TopoART-C modules.
<i>rho_a</i>	The vigilance parameter of the first TopoART-C module (TopoART-C a).

### 5.7.2.2 Fast\_TopoART\_C() [2/2]

```
LibTopoART.Fast_TopoART_C.Fast_TopoART_C (
    string path )
```

This constructor loads a saved TopoART-C network.

#### Parameters

<i>path</i>	The path of a binary TopoART-C file.
-------------	--------------------------------------

## 5.7.3 Member Function Documentation

### 5.7.3.1 Learn() [1/2]

```
override void LibTopoART.Fast_TopoART_C.Learn (
    decimal [] input ) [virtual]
```

This method performs a single training step and sets the class ID corresponding to *input* to UNDEFINED\_CLASS\_ID.

#### Parameters

<i>input</i>	The input vector to be learnt.
--------------	--------------------------------

Reimplemented from [LibTopoART.Fast\\_TopoART](#).

**5.7.3.2 Learn()** [2/2]

```
void LibTopoART.Fast_TopoART_C.Learn (
    decimal [ ] input,
    long class_ID )
```

This method performs a single training step.

**Parameters**

<i>input</i>	The input vector to be learnt.
<i>class_ID</i>	The class ID corresponding to <i>input</i> . (must be equal to or larger than 0)

Implements [LibTopoART.ITopoART\\_C](#).

**5.7.3.3 Predict()** [1/2]

```
long LibTopoART.Fast_TopoART_C.Predict (
    decimal [ ] input,
    long nu )
```

This method predicts the class ID.

**Parameters**

<i>input</i>	The input vector the class ID of which is to be predicted.
<i>nu</i>	The maximum cardinality of the set of enclosing categories E and the neighbourhood set N. (This parameter does not modify the network. It may be arbitrarily changed in each prediction step.)

**Returns**

The predicted class ID.

Implements [LibTopoART.ITopoART\\_C](#).

**5.7.3.4 Predict()** [2/2]

```
TopoART\_C\_Prediction LibTopoART.Fast_TopoART_C.Predict (
    decimal [ ] input,
    bool [ ] mask_vector,
    long nu )
```

This method predicts the class ID.

**Parameters**

<i>input</i>	The input vector the class ID of which is to be predicted.
<i>mask_vector</i>	The mask vector corresponding to <i>input</i> .
<i>nu</i>	The maximum cardinality of the set of enclosing categories E and the neighbourhood set N. (This parameter does not modify the network. It may be arbitrarily changed in each prediction step.)

**Returns**

An object of type [TopoART\\_C\\_Prediction](#) containing the predicted class ID and a corresponding confidence value.

Implements [LibTopoART.ITopoART\\_C](#).

**5.7.4 Member Data Documentation****5.7.4.1 file\_format\_version**

```
new const decimal LibTopoART.Fast_TopoART_C.file_format_version = 0.01m
```

Instance variable `file_format_version` represents the version of the file format used by class [Fast\\_TopoART\\_C](#).

**5.7.4.2 UNDEFINED\_CLASS\_ID**

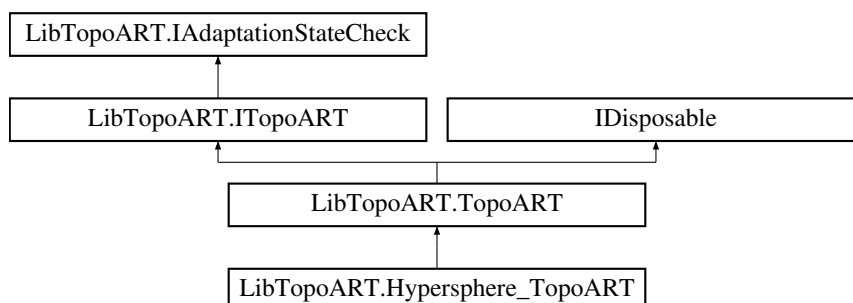
```
const long LibTopoART.Fast_TopoART_C.UNDEFINED_CLASS_ID = -2
```

Instance variable `UNDEFINED_CLASS_ID` gives the value used for indicating that an input sample was predict to belong to the undefined class; i.e, no class ID was provided for such input samples during training.

**5.8 LibTopoART.Hypersphere\_TopoART Class Reference**

Class [Hypersphere\\_TopoART](#) provides an implementation of the Hypersphere [TopoART](#) neural network as proposed in "Tscherepanow, Marko (2012). Incremental On-line Clustering with a Topology-Learning Hierarchical ART Neural Network Using Hyperspherical Categories. In Poster and Industry Proceedings of the Industrial Conference on Data Mining (ICDM), pp. 22–34. Fockendorf, Germany: ibai-publishing."

Inheritance diagram for `LibTopoART.Hypersphere_TopoART`:



## Public Member Functions

- [Hypersphere\\_TopoART](#) (long input\_dimension, long module\_number, decimal rho\_a)  
*This constructor initialises a Hypersphere [TopoART](#) network and sets the radial extend parameter to  $\text{Math}.\sqrt{\text{input\_dimension}}/2$ .*
- [Hypersphere\\_TopoART](#) (long input\_dimension, long module\_number, decimal rho\_a, decimal R)  
*This constructor initialises a Hypersphere [TopoART](#) network.*
- [Hypersphere\\_TopoART](#) (string path)  
*This constructor loads a saved Hypersphere [TopoART](#) network.*

## Public Attributes

- new const decimal [file\\_format\\_version](#) = 0.01m  
*Instance variable [file\\_format\\_version](#) represents the version of the file format used by class [Hypersphere\\_TopoART](#).*

## Properties

- decimal [R](#) [get]  
*Property [R](#) represents the radial extend parameter  $R$ .*

## Additional Inherited Members

## 5.8.1 Detailed Description

Class [Hypersphere\\_TopoART](#) provides an implementation of the Hypersphere [TopoART](#) neural network as proposed in "Tscherepanow, Marko (2012). Incremental On-line Clustering with a Topology-Learning Hierarchical ART Neural Network Using Hyperspherical Categories. In Poster and Industry Proceedings of the Industrial Conference on Data Mining (ICDM), pp. 22–34. Fockendorf, Germany: ibai-publishing."

In contrast to class [TopoART](#), class [Hypersphere\\_TopoART](#) does not require all input to lie in the interval [0,1]. The input range is controlled by the radial extend parameter  $R$

## 5.8.2 Constructor &amp; Destructor Documentation

5.8.2.1 [Hypersphere\\_TopoART\(\)](#) [1/3]

```
LibTopoART.Hypersphere_TopoART.Hypersphere_TopoART (
    long input_dimension,
    long module_number,
    decimal rho_a )
```

This constructor initialises a Hypersphere [TopoART](#) network and sets the radial extend parameter to  $\text{Math}.\sqrt{\text{input\_dimension}}/2$ .

## Parameters

<i>input_dimension</i>	The dimension of input vectors to be learnt.
<i>module_number</i>	The number of Hypersphere <a href="#">TopoART</a> modules.
<i>rho_a</i>	The vigilance parameter of the first Hypersphere <a href="#">TopoART</a> module (HTA a).

5.8.2.2 [Hypersphere\\_TopoART\(\)](#) [2/3]

```
LibTopoART.Hypersphere_TopoART.Hypersphere_TopoART (
    long input_dimension,
    long module_number,
    decimal rho_a,
    decimal R )
```

This constructor initialises a Hypersphere [TopoART](#) network.

## Parameters

<i>input_dimension</i>	The dimension of input vectors to be learnt.
<i>module_number</i>	The number of Hypersphere <a href="#">TopoART</a> modules.
<i>rho_a</i>	The vigilance parameter of the first Hypersphere <a href="#">TopoART</a> module (HTA a).
<i>R</i>	The radial extend parameter.

5.8.2.3 [Hypersphere\\_TopoART\(\)](#) [3/3]

```
LibTopoART.Hypersphere_TopoART.Hypersphere_TopoART (
    string path )
```

This constructor loads a saved Hypersphere [TopoART](#) network.

## Parameters

<i>path</i>	The path of a binary Hypersphere <a href="#">TopoART</a> file.
-------------	--

## 5.8.3 Member Data Documentation

5.8.3.1 [file\\_format\\_version](#)

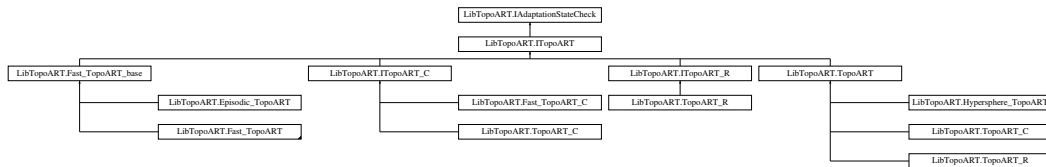
```
new const decimal LibTopoART.Hypersphere_TopoART.file_format_version = 0.01m
```

Instance variable `file_format_version` represents the version of the file format used by class [Hypersphere\\_TopoART](#).

## 5.9 LibTopoART.IAdaptationStateCheck Interface Reference

Interface enabling checks whether certain adaptations of a network occurred.

Inheritance diagram for LibTopoART.IAdaptationStateCheck:



### Public Member Functions

- void [ResetAdaptationState](#) ()  
*This method resets the adaptation state to `AdaptationState.NO_ADAPTATION`.*
- [AdaptationState GetAdaptationState](#) (decimal epsilon=0.001m)  
*This method returns the current adaptation state.*

#### 5.9.1 Detailed Description

Interface enabling checks whether certain adaptations of a network occurred.

#### 5.9.2 Member Function Documentation

##### 5.9.2.1 GetAdaptationState()

```
AdaptationState LibTopoART.IAdaptationStateCheck.GetAdaptationState (
    decimal epsilon = 0.001m )
```

This method returns the current adaptation state.

#### Parameters

<i>epsilon</i>	The threshold for weight adaptations to be considered.
----------------	--

#### Returns

An enumeration describing the adaptation state.

Implemented in [LibTopoART.TopoART](#), and [LibTopoART.Fast\\_TopoART\\_base](#).

### 5.9.2.2 ResetAdaptationState()

```
void LibTopoART.IAdaptationStateCheck.ResetAdaptationState ( )
```

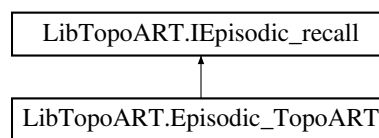
This method resets the adaptation state to [AdaptationState.NO\\_ADAPTATION](#).

Implemented in [LibTopoART.TopoART](#), and [LibTopoART.Fast\\_TopoART\\_base](#).

## 5.10 LibTopoART.IEpisodic\_recall Interface Reference

Interface summarising the episodic recall functionality.

Inheritance diagram for LibTopoART.IEpisodic\_recall:



### Public Member Functions

- long [BeginRecall](#) (decimal[ ] stimulus)  
*This method starts the recall process.*
- bool [InterEpisodeRecallStep](#) (out decimal[ ] recall\_result, out decimal F3\_activation)  
*This method performs a single inter-episode recall step and sets the starting point for intra-episode recall.*
- bool [IntraEpisodeRecallStep](#) (out decimal[ ] recall\_result)  
*This method performs a single intra-episode recall step.*
- void [EndRecall](#) ()  
*This method stops the recall process and frees temporary resources.*

### 5.10.1 Detailed Description

Interface summarising the episodic recall functionality.

### 5.10.2 Member Function Documentation

#### 5.10.2.1 BeginRecall()

```
long LibTopoART.IEpisodic_recall.BeginRecall (
    decimal [ ] stimulus )
```

This method starts the recall process.



**Parameters**

<i>stimulus</i>	The stimulus (input) which is used to trigger recall.
-----------------	---

**Returns**

The number of F3 nodes created.

Implemented in [LibTopoART.Episodic\\_TopoART](#).

**5.10.2.2 EndRecall()**

```
void LibTopoART.IEpisodic_recall.EndRecall ( )
```

This method stops the recall process and frees temporary resources.

Implemented in [LibTopoART.Episodic\\_TopoART](#).

**5.10.2.3 InterEpisodeRecallStep()**

```
bool LibTopoART.IEpisodic_recall.InterEpisodeRecallStep (
    out decimal [] recall_result,
    out decimal F3_activation )
```

This method performs a single inter-episode recall step and sets the starting point for intra-episode recall.

**Parameters**

<i>recall_result</i>	Returns the recall output vector for the current step.
<i>F3_activation</i>	Returns the activation of the current F3 node.

**Returns**

A boolean result indicating whether the recall step was successfully completed, or not.

Implemented in [LibTopoART.Episodic\\_TopoART](#).

**5.10.2.4 IntraEpisodeRecallStep()**

```
bool LibTopoART.IEpisodic_recall.IntraEpisodeRecallStep (
    out decimal [] recall_result )
```

This method performs a single intra-episode recall step.

**Parameters**

<i>recall_result</i>	Returns the recall output vector for the current step.
----------------------	--

**Returns**

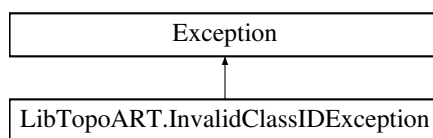
A boolean result indicating whether the recall step was successfully completed, or not.

Implemented in [LibTopoART.Episodic\\_TopoART](#).

**5.11 LibTopoART.InvalidClassIDException Class Reference**

Exception signalling an invalid class ID.

Inheritance diagram for LibTopoART.InvalidClassIDException:

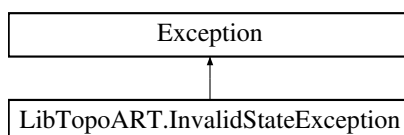
**5.11.1 Detailed Description**

Exception signalling an invalid class ID.

**5.12 LibTopoART.InvalidStateException Class Reference**

Exception signalling an invalid state of the neural network.

Inheritance diagram for LibTopoART.InvalidStateException:

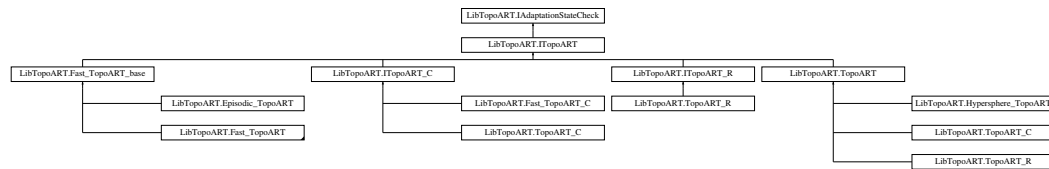
**5.12.1 Detailed Description**

Exception signalling an invalid state of the neural network.

## 5.13 LibTopoART.ITopoART Interface Reference

Interface summarising the basic [TopoART](#) functionality.

Inheritance diagram for LibTopoART.ITopoART:



### Public Member Functions

- void [ComputeClusterIDs](#) ()  
*This method computes the cluster IDs for all neurons.*
- [F2\\_output](#) [] [GetBMOutput](#) (decimal[] input)  
*This method finds the closest category for a given test input.*
- [F2\\_output](#) [] [GetBMOutput](#) (decimal[] input, bool[] mask\_vector)  
*This method finds the closest category for a given test input.*
- void [Learn](#) (decimal[] input)  
*This method performs a single training step.*
- void [SaveText](#) (string path)  
*This method saves the entire network as a text file.*
- void [Save](#) (string path)  
*This method saves the entire network as a binary file.*

### Properties

- long [] [NodeNum](#) [get]  
*Property NodeNum represents the number of [TopoART](#) nodes used by each module.*
- long [] [ClusterNum](#) [get]  
*Property ClusterNum represents the number of [TopoART](#) clusters found by each module.*
- long [ModuleNum](#) [get]
- long [LearningSteps](#) [get]  
*Property LearningSteps represents the total number of performed learning steps.*
- decimal [Beta\\_sbm](#) [get, set]  
*Property Beta\_sbm represents the learning rate of the second best-matching nodes.*
- decimal [Rho\\_a](#) [get]  
*Property Rho\_a represents the vigilance parameter of the first [TopoART](#) module (TA a).*
- long [Tau](#) [get, set]  
*Property Tau represents the parameter tau required for the removal of nodes and edges.*
- long [Phi](#) [get, set]
- decimal [Alpha](#) [get, set]  
*Property Alpha represents the choice parameter alpha.*

#### 5.13.1 Detailed Description

Interface summarising the basic [TopoART](#) functionality.

### 5.13.2 Member Function Documentation

#### 5.13.2.1 ComputeClusterIDs()

```
void LibTopoART.ITopoART.ComputeClusterIDs ( )
```

This method computes the cluster IDs for all neurons.

Implemented in [LibTopoART.TopoART](#), and [LibTopoART.Fast\\_TopoART\\_base](#).

#### 5.13.2.2 GetBMOutput() [1/2]

```
F2_output [ ] LibTopoART.ITopoART.GetBMOutput (
    decimal [ ] input )
```

This method finds the closest category for a given test input.

##### Parameters

<i>input</i>	The input vector $x(t)$ .
--------------	---------------------------

##### Returns

An array of type [F2\\_output](#). Each entry contains the ID of the best-matching node and the corresponding cluster ID for one [TopoART](#) module.

Implemented in [LibTopoART.TopoART](#), and [LibTopoART.Fast\\_TopoART\\_base](#).

#### 5.13.2.3 GetBMOutput() [2/2]

```
F2_output [ ] LibTopoART.ITopoART.GetBMOutput (
    decimal [ ] input,
    bool [ ] mask_vector )
```

This method finds the closest category for a given test input.

##### Parameters

<i>input</i>	The input vector $x(t)$ .
<i>mask_vector</i>	A mask vector excluding individual dimensions of $x(t)$ from the computation. (Setting an element of the mask vector to <code>true</code> , excludes the corresponding elements of $x(t)$ .)

**Returns**

An array of type `F2_output`. Each entry contains the ID of the best-matching node and the corresponding cluster ID for one `TopoART` module.

Implemented in `LibTopoART.TopoART`, and `LibTopoART.Fast_TopoART_base`.

**5.13.2.4 Learn()**

```
void LibTopoART.ITopoART.Learn (
    decimal [ ] input )
```

This method performs a single training step.

**Parameters**

<i>input</i>	The input vector to be learnt.
--------------	--------------------------------

Implemented in `LibTopoART.TopoART`, `LibTopoART.Fast_TopoART_C`, `LibTopoART.Episodic_TopoART`, `LibTopoART.TopoART_R`, `LibTopoART.Fast_TopoART`, `LibTopoART.TopoART_C`, and `LibTopoART.Fast_TopoART_base`.

**5.13.2.5 Save()**

```
void LibTopoART.ITopoART.Save (
    string path )
```

This method saves the entire network as a binary file.

**Parameters**

<i>path</i>	A string representing the path of the file to save.
-------------	---

Implemented in `LibTopoART.TopoART`, and `LibTopoART.Fast_TopoART_base`.

**5.13.2.6 SaveText()**

```
void LibTopoART.ITopoART.SaveText (
    string path )
```

This method saves the entire network as a text file.

**Parameters**

<i>path</i>	A string representing the path of the file to save.
-------------	---

Implemented in `LibTopoART.TopoART`, and `LibTopoART.Fast_TopoART_base`.

### 5.13.3 Property Documentation

#### 5.13.3.1 ModuleNum

```
long LibTopoART.ITopoART.ModuleNum [get]
```

Property `ModuleNum` represents the number of [TopoART](#) modules used. (The original [TopoART](#) uses two modules.)

#### 5.13.3.2 Phi

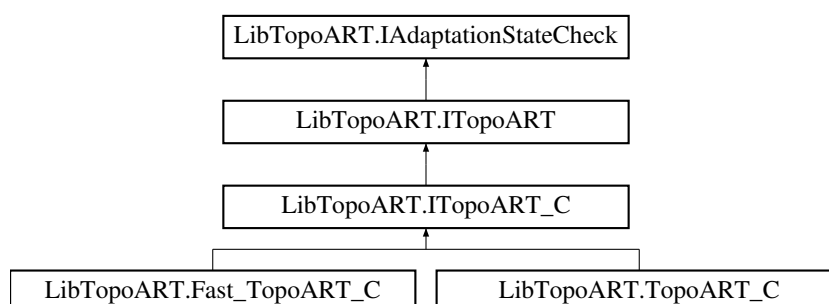
```
long LibTopoART.ITopoART.Phi [get], [set]
```

Property `Phi` represents the parameter `phi` required for the removal of nodes and edges as well as for the propagation of input to subsequent [TopoART](#) modules.

## 5.14 LibTopoART.ITopoART\_C Interface Reference

Interface summarising the TopoART-C functionality.

Inheritance diagram for `LibTopoART.ITopoART_C`:



### Public Member Functions

- void [Learn](#) (decimal[] input, long class\_ID)  
*This method performs a single training step.*
- long [Predict](#) (decimal[] input, long nu)  
*This method predicts the class ID.*
- [TopoART\\_C\\_Prediction Predict](#) (decimal[] input, bool[] mask\_vector, long nu)  
*This method predicts the class ID.*

### Additional Inherited Members

#### 5.14.1 Detailed Description

Interface summarising the TopoART-C functionality.

## 5.14.2 Member Function Documentation

## 5.14.2.1 Learn()

```
void LibTopoART.ITopoART_C.Learn (
    decimal [ ] input,
    long class_ID )
```

This method performs a single training step.

## Parameters

<i>input</i>	The input vector to be learnt.
<i>class_ID</i>	The class ID corresponding to <i>input</i> .

Implemented in [LibTopoART.Fast\\_TopoART\\_C](#), and [LibTopoART.TopoART\\_C](#).

## 5.14.2.2 Predict() [1/2]

```
long LibTopoART.ITopoART_C.Predict (
    decimal [ ] input,
    long nu )
```

This method predicts the class ID.

## Parameters

<i>input</i>	The input vector the class ID of which is to be predicted.
<i>nu</i>	The maximum cardinality of the set of enclosing categories E and the neighbourhood set N. (This parameter does not modify the network. It may be arbitrarily changed in each prediction step.)

## Returns

The predicted class ID.

Implemented in [LibTopoART.Fast\\_TopoART\\_C](#), and [LibTopoART.TopoART\\_C](#).

## 5.14.2.3 Predict() [2/2]

```
TopoART_C_Prediction LibTopoART.ITopoART_C.Predict (
    decimal [ ] input,
    bool [ ] mask_vector,
    long nu )
```

This method predicts the class ID.

**Parameters**

<i>input</i>	The input vector the class ID of which is to be predicted.
<i>mask_vector</i>	The mask vector corresponding to <i>input</i> .
<i>nu</i>	The maximum cardinality of the set of enclosing categories E and the neighbourhood set N. (This parameter does not modify the network. It may be arbitrarily changed in each prediction step.)

**Returns**

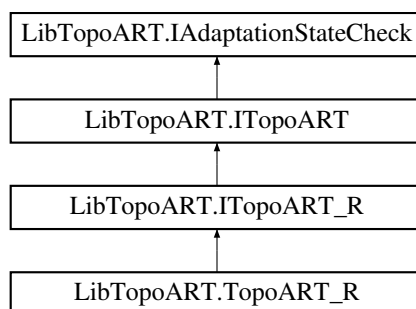
An object of type [TopoART\\_C\\_Prediction](#) containing the predicted class ID and a corresponding confidence value.

Implemented in [LibTopoART.Fast\\_TopopART\\_C](#), and [LibTopoART.TopoART\\_C](#).

**5.15 LibTopoART.ITopoART\_R Interface Reference**

Interface summarising the TopoART-R functionality.

Inheritance diagram for LibTopoART.ITopoART\_R:

**Public Member Functions**

- void [Learn](#) (decimal[] i\_vec, decimal[] d\_vec)  
*This method performs a single training step.*
- decimal[] [Predict](#) (decimal[] i\_vec, long nu=10)  
*This method predicts the dependent variables.*
- [TopoART\\_R\\_Prediction Predict](#) (decimal[] i\_vec, bool[] m\_i\_vec, long nu=10)  
*This method predicts the dependent variables for a given set of independent variables. Unknown values of independent variables can be signified by setting the corresponding value of m\_i\_vec to true.*

**Additional Inherited Members****5.15.1 Detailed Description**

Interface summarising the TopoART-R functionality.



## 5.15.2 Member Function Documentation

## 5.15.2.1 Learn()

```
void LibTopoART.ITopoART_R.Learn (
    decimal [ ] i_vec,
    decimal [ ] d_vec )
```

This method performs a single training step.

## Parameters

<i>i_vec</i>	The input vector (independent variables) to be learnt.
<i>d_vec</i>	The output vector (dependent variables) corresponding to <i>i_vec</i> .

Implemented in [LibTopoART.TopoART\\_R](#).

## 5.15.2.2 Predict() [1/2]

```
decimal [ ] LibTopoART.ITopoART_R.Predict (
    decimal [ ] i_vec,
    long nu = 10 )
```

This method predicts the dependent variables.

## Parameters

<i>i_vec</i>	The input vector (independent variables).
<i>nu</i>	The maximum cardinality of the neighbourhood set N. (In the original TopoART-R network, nu is fixed to 10. But task-specific adaptations might lead to an improved prediction accuracy. This parameter does not modify the network. It may be arbitrarily changed in each prediction step.)

## Returns

The predicted values for all dependent variables.

Implemented in [LibTopoART.TopoART\\_R](#).

## 5.15.2.3 Predict() [2/2]

```
TopoART\_R\_Prediction LibTopoART.ITopoART_R.Predict (
    decimal [ ] i_vec,
    bool [ ] m_i_vec,
    long nu = 10 )
```

This method predicts the dependent variables for a given set of independent variables. Unknown values of independent variables can be signified by setting the corresponding value of *m\_i\_vec* to `true`.

## Parameters

<i>i_vec</i>	The input vector (independent variables).
<i>m_i_vec</i>	The mask vector corresponding to <i>i_vec</i> .
<i>nu</i>	The maximum cardinality of the neighbourhood set N. (In the original TopoART-R network, nu is fixed to 10. But task-specific adaptations might lead to an improved prediction accuracy. This parameter does not modify the network. It may be arbitrarily changed in each prediction step.)

## Returns

An object of type [TopoART\\_R\\_Prediction](#) containing the predicted values for the unknown independent variables and all dependent variables.

Implemented in [LibTopoART.TopoART\\_R](#).

## 5.16 LibTopoART.LibTopoART\_info Struct Reference

Struct [LibTopoART\\_info](#) provides some metainformation regarding the respective implementation of [LibTopoART](#).

## Public Attributes

- const decimal [version](#) = 0.74m  
*Instance variable `version` represents the version of [LibTopoART](#).*
- const long [UNDEFINED](#) = -1  
*Instance variable `UNDEFINED` gives the value used for indicating undefined and uninitialised variables.*

## Static Public Attributes

- static readonly string [] [networks](#)  
*Instance variable `networks` provides a string array containing the networks implemented in the current version of [LibTopoART](#) and the corresponding class names.*

### 5.16.1 Detailed Description

Struct [LibTopoART\\_info](#) provides some metainformation regarding the respective implementation of [LibTopoART](#).

### 5.16.2 Member Data Documentation

## 5.16.2.1 networks

```
readonly string [] LibTopoART.LibTopoART_info.networks [static]
```

**Initial value:**

```
= {
    "Episodic TopoART (class Episodic_TopoART)",
    "Hypersphere TopoART (class Hypersphere_TopoART)",
    "TopoART (class TopoART, class Fast_TopoART)",
    "TopoART-C (class TopoART_C, class Fast_TopoART_C)",
    "TopoART-R (class TopoART_R)"}
```

Instance variable `networks` provides a string array containing the networks implemented in the current version of [LibTopoART](#) and the corresponding class names.

## 5.16.2.2 UNDEFINED

```
const long LibTopoART.LibTopoART_info.UNDEFINED = -1
```

Instance variable `UNDEFINED` gives the value used for indicating undefined and uninitialised variables.

## 5.16.2.3 version

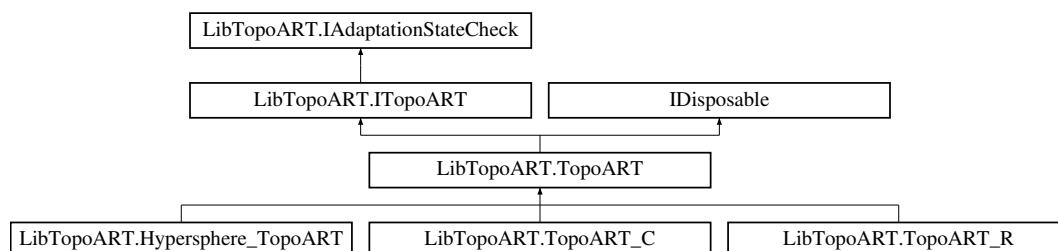
```
const decimal LibTopoART.LibTopoART_info.version = 0.74m
```

Instance variable `version` represents the version of [LibTopoART](#).

## 5.17 LibTopoART.TopoART Class Reference

Class [TopoART](#) provides an implementation of the [TopoART](#) neural network as proposed in "Tscherepanow, Marko (2010). [TopoART](#): A topology learning hierarchical ART network. In Proceedings of the International Conference on Artificial Neural Networks (ICANN), LNCS 6354, pp. 157–167. Berlin, Germany: Springer."

Inheritance diagram for LibTopoART.TopoART:



## Public Member Functions

- [TopoART](#) (long input\_dimension, long module\_number, decimal rho\_a)  
*This constructor initialises a [TopoART](#) network.*
- [TopoART](#) (string path)  
*This constructor loads a saved [TopoART](#) network.*
- void [Dispose](#) ()  
*Releases all resources used by the [LibTopoART.TopoART](#) object.*
- void [ComputeClusterIDs](#) ()  
*This method computes the cluster IDs for all neurons.*
- [F2\\_output](#) [ ] [GetBMOutput](#) (decimal [ ] input)  
*This method finds the closest category for a given test input.*
- [F2\\_output](#) [ ] [GetBMOutput](#) (decimal [ ] input, bool [ ] mask\_vector)  
*This method finds the closest category for a given test input.*
- virtual void [Learn](#) (decimal [ ] input)  
*This method performs a single training step.*
- void [SaveText](#) (string path)  
*This method saves the entire network as a text file.*
- void [Save](#) (string path)  
*This method saves the entire network as a binary file.*
- void [ResetAdaptationState](#) ()  
*This method resets the adaptation state to [AdaptationState.NO\\_ADAPTATION](#).*
- [AdaptationState](#) [GetAdaptationState](#) (decimal epsilon=0.001m)  
*This method returns the current adaptation state.*

## Public Attributes

- const decimal [file\\_format\\_version](#) = 0.09m  
*Instance variable [file\\_format\\_version](#) represents the version of the file format used by class [TopoART](#).*
- const string [integer\\_base\\_type](#) = "long"  
*Instance variable [integer\\_base\\_type](#) provides a string containing the data type used for representing integer variables (IDs, parameters, counters, etc.) internally.*
- const string [float\\_base\\_type](#) = "decimal"  
*Instance variable [float\\_base\\_type](#) provides a string containing the data type used for representing floating point variables (input, weights, etc.) internally.*

## Protected Member Functions

- virtual void [Dispose](#) (bool disposing)  
*Release resources used by the [LibTopoART.TopoART](#) object.*

## Properties

- long [] [NodeNum](#) [get]  
*Property `NodeNum` represents the number of [TopoART](#) nodes used by each module.*
- long [] [ClusterNum](#) [get]  
*Property `ClusterNum` represents the number of [TopoART](#) clusters found by each module.*
- long [ModuleNum](#) [get]
- long [LearningSteps](#) [get]  
*Property `LearningSteps` represents the total number of performed learning steps.*
- decimal [Rho\\_a](#) [get]  
*Property `Rho_a` represents the vigilance parameter of the first [TopoART](#) module (TA a).*
- decimal [Beta\\_sbm](#) [get, set]  
*Property `Beta_sbm` represents the learning rate of the second best-matching nodes.*
- long [Tau](#) [get, set]  
*Property `Tau` represents the parameter tau required for the removal of nodes and edges.*
- long [Phi](#) [get, set]
- decimal [Alpha](#) [get, set]  
*Property `Alpha` represents the choice parameter alpha.*

### 5.17.1 Detailed Description

Class [TopoART](#) provides an implementation of the [TopoART](#) neural network as proposed in "Tscherepanow, Marko (2010). [TopoART](#): A topology learning hierarchical ART network. In Proceedings of the International Conference on Artificial Neural Networks (ICANN), LNCS 6354, pp. 157–167. Berlin, Germany: Springer."

Internally, real-valued data are stored in `decimal` variables. Hence, computations are rather slow but very accurate.

Class [TopoART](#) requires all input to lie in the interval [0,1].

### 5.17.2 Constructor & Destructor Documentation

#### 5.17.2.1 [TopoART\(\)](#) [1/2]

```
LibTopoART.TopoART.TopoART (
    long input_dimension,
    long module_number,
    decimal rho_a )
```

This constructor initialises a [TopoART](#) network.

#### Parameters

<i>input_dimension</i>	The dimension of input vectors to be learnt.
<i>module_number</i>	The number of <a href="#">TopoART</a> modules.
<i>rho_a</i>	The vigilance parameter of the first <a href="#">TopoART</a> module (TA a).

### 5.17.2.2 TopoART() [2/2]

```
LibTopoART.TopoART.TopoART (
    string path )
```

This constructor loads a saved [TopoART](#) network.

#### Parameters

<i>path</i>	The path of a binary <a href="#">TopoART</a> file.
-------------	--

## 5.17.3 Member Function Documentation

### 5.17.3.1 ComputeClusterIDs()

```
void LibTopoART.TopoART.ComputeClusterIDs ( )
```

This method computes the cluster IDs for all neurons.

Implements [LibTopoART.ITopoART](#).

### 5.17.3.2 Dispose() [1/2]

```
void LibTopoART.TopoART.Dispose ( )
```

Releases all resources used by the [LibTopoART.TopoART](#) object.

Call [Dispose\(\)](#) when you are finished using the [LibTopoART.TopoART](#). The [Dispose\(\)](#) method leaves the [LibTopoART.TopoART](#) in an unusable state. After calling [Dispose\(\)](#), you must release all references to the [LibTopoART.TopoART](#) so the garbage collector can reclaim the memory that the [LibTopoART.TopoART](#) was occupying.

### 5.17.3.3 Dispose() [2/2]

```
virtual void LibTopoART.TopoART.Dispose (
    bool disposing ) [protected], [virtual]
```

Release resources used by the [LibTopoART.TopoART](#) object.

#### Parameters

<i>disposing</i>	If set to <code>true</code> all managed resources are released.
------------------	---

### 5.17.3.4 GetAdaptationState()

```
AdaptationState LibTopoART.TopoART.GetAdaptationState (
```

```
decimal epsilon = 0.001m )
```

This method returns the current adaptation state.

#### Parameters

<i>epsilon</i>	The threshold for weight adaptations to be considered.
----------------	--

#### Returns

An enumeration describing the adaptation state.

Implements [LibTopoART.IAdaptationStateCheck](#).

#### 5.17.3.5 GetBMOutput() [1/2]

```
F2_output [ ] LibTopoART.TopoART.GetBMOutput (
    decimal [ ] input )
```

This method finds the closest category for a given test input.

#### Parameters

<i>input</i>	The input vector x(t).
--------------	------------------------

#### Returns

An array of type [F2\\_output](#). Each entry contains the ID of the best-matching node and the corresponding cluster ID for one [TopoART](#) module.

Implements [LibTopoART.ITopoART](#).

#### 5.17.3.6 GetBMOutput() [2/2]

```
F2_output [ ] LibTopoART.TopoART.GetBMOutput (
    decimal [ ] input,
    bool [ ] mask_vector )
```

This method finds the closest category for a given test input.

#### Parameters

<i>input</i>	The input vector x(t).
<i>mask_vector</i>	A mask vector excluding individual dimensions of x(t) from the computation. (Setting an element of the mask vector to <code>true</code> , excludes the corresponding elements of x(t).)

**Returns**

An array of type [F2\\_output](#). Each entry contains the ID of the best-matching node and the corresponding cluster ID for one [TopoART](#) module.

Implements [LibTopoART.ITopoART](#).

**5.17.3.7 Learn()**

```
virtual void LibTopoART.TopoART.Learn (
    decimal [] input ) [virtual]
```

This method performs a single training step.

**Parameters**

<i>input</i>	The input vector to be learnt.
--------------	--------------------------------

Implements [LibTopoART.ITopoART](#).

Reimplemented in [LibTopoART.TopoART\\_R](#), and [LibTopoART.TopoART\\_C](#).

**5.17.3.8 ResetAdaptationState()**

```
void LibTopoART.TopoART.ResetAdaptationState ( )
```

This method resets the adaptation state to [AdaptationState.NO\\_ADAPTATION](#).

Implements [LibTopoART.IAdaptationStateCheck](#).

**5.17.3.9 Save()**

```
void LibTopoART.TopoART.Save (
    string path )
```

This method saves the entire network as a binary file.

**Parameters**

<i>path</i>	A <code>string</code> representing the path of the file to save.
-------------	--

Implements [LibTopoART.ITopoART](#).



## 5.17.3.10 SaveText()

```
void LibTopoART.TopoART.SaveText (
    string path )
```

This method saves the entire network as a text file.

## Parameters

<i>path</i>	A string representing the path of the file to save.
-------------	---

Implements [LibTopoART.ITopoART](#).

## 5.17.4 Member Data Documentation

## 5.17.4.1 file\_format\_version

```
const decimal LibTopoART.TopoART.file_format_version = 0.09m
```

Instance variable `file_format_version` represents the version of the file format used by class [TopoART](#).

## 5.17.4.2 float\_base\_type

```
const string LibTopoART.TopoART.float_base_type = "decimal"
```

Instance variable `float_base_type` provides a string containing the data type used for representing floating point variables (input, weights, etc.) internally.

## 5.17.4.3 integer\_base\_type

```
const string LibTopoART.TopoART.integer_base_type = "long"
```

Instance variable `integer_base_type` provides a string containing the data type used for representing integer variables (IDs, parameters, counters, etc.) internally.

## 5.17.5 Property Documentation

## 5.17.5.1 ModuleNum

```
long LibTopoART.TopoART.ModuleNum [get]
```

Property `ModuleNum` represents the number of [TopoART](#) modules used. (The original [TopoART](#) uses two modules.)

### 5.17.5.2 Phi

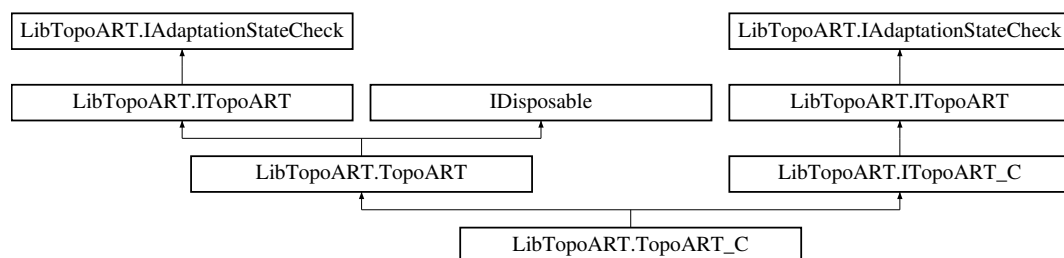
```
long LibTopoART.TopoART.Phi [get], [set]
```

Property `Phi` represents the parameter  $\phi$  required for the removal of nodes and edges as well as for the propagation of input to subsequent [TopoART](#) modules.

## 5.18 LibTopoART.TopoART\_C Class Reference

Class [TopoART\\_C](#) provides an implementation of the TopoART-C neural network as proposed in "Marko Tscherepanow and Sören Riechers, 'An Incremental On-line Classifier for Imbalanced, Incomplete, and Noisy Data', European Conference on Artificial Intelligence (ECAI), Workshop on Active and Incremental Learning (AIL), pp. 18-23, 2012."

Inheritance diagram for `LibTopoART.TopoART_C`:



### Public Member Functions

- [TopoART\\_C](#) (long input\_dimension, long module\_number, decimal rho\_a)  
*This constructor initialises a TopoART-C network.*
- [TopoART\\_C](#) (string path)  
*This constructor loads a saved TopoART-C network.*
- override void [Learn](#) (decimal[] input)  
*This method performs a single training step and sets the class ID corresponding to input to `UNDEFINED_CLASS`↔  
`_ID`.*
- void [Learn](#) (decimal[] input, long class\_ID)  
*This method performs a single training step.*
- long [Predict](#) (decimal[] input, long nu)  
*This method predicts the class ID.*
- [TopoART\\_C\\_Prediction Predict](#) (decimal[] input, bool[] mask\_vector, long nu)  
*This method predicts the class ID.*

### Public Attributes

- const long [UNDEFINED\\_CLASS\\_ID](#) = -2  
*Instance variable `UNDEFINED_CLASS_ID` gives the value used for indicating that an input sample was predict to belong to the undefined class; i.e, no class ID was provided for such input samples during training.*
- new const decimal [file\\_format\\_version](#) = 0.01m  
*Instance variable `file_format_version` represents the version of the file format used by class [TopoART\\_C](#).*

## Additional Inherited Members

## 5.18.1 Detailed Description

Class [TopoART\\_C](#) provides an implementation of the TopoART-C neural network as proposed in "Marko Tscherepanow and Sören Riechers, 'An Incremental On-line Classifier for Imbalanced, Incomplete, and Noisy Data', European Conference on Artificial Intelligence (ECAI), Workshop on Active and Incremental Learning (AIL), pp. 18-23, 2012."

Class [TopoART\\_C](#) requires all input except the class IDs to lie in the interval [0,1]. The class IDs are signed integer values.

## 5.18.2 Constructor &amp; Destructor Documentation

## 5.18.2.1 TopoART\_C() [1/2]

```
LibTopoART.TopoART_C.TopoART_C (
    long  input_dimension,
    long  module_number,
    decimal rho_a )
```

This constructor initialises a TopoART-C network.

## Parameters

<i>input_dimension</i>	The dimension of input vectors to be learnt.
<i>module_number</i>	The number of TopoART-C modules.
<i>rho_a</i>	The vigilance parameter of the first TopoART-C module (TopoART-C a).

## 5.18.2.2 TopoART\_C() [2/2]

```
LibTopoART.TopoART_C.TopoART_C (
    string path )
```

This constructor loads a saved TopoART-C network.

## Parameters

<i>path</i>	The path of a binary TopoART-C file.
-------------	--------------------------------------

## 5.18.3 Member Function Documentation

### 5.18.3.1 Learn() [1/2]

```
override void LibTopoART.TopoART_C.Learn (
    decimal [] input ) [virtual]
```

This method performs a single training step and sets the class ID corresponding to *input* to UNDEFINED\_CLASS\_ID.

#### Parameters

<i>input</i>	The input vector to be learnt.
--------------	--------------------------------

Reimplemented from [LibTopoART.TopoART](#).

### 5.18.3.2 Learn() [2/2]

```
void LibTopoART.TopoART_C.Learn (
    decimal [] input,
    long class_ID )
```

This method performs a single training step.

#### Parameters

<i>input</i>	The input vector to be learnt.
<i>class_ID</i>	The class ID corresponding to <i>input</i> . (must be equal to or larger than 0)

Implements [LibTopoART.ITopoART\\_C](#).

### 5.18.3.3 Predict() [1/2]

```
long LibTopoART.TopoART_C.Predict (
    decimal [] input,
    long nu )
```

This method predicts the class ID.

#### Parameters

<i>input</i>	The input vector the class ID of which is to be predicted.
<i>nu</i>	The maximum cardinality of the set of enclosing categories E and the neighbourhood set N. (This parameter does not modify the network. It may be arbitrarily changed in each prediction step.)

#### Returns

The predicted class ID.

Implements [LibTopoART.ITopoART\\_C](#).

## 5.18.3.4 Predict() [2/2]

```
TopoART_C_Prediction LibTopoART.TopoART_C.Predict (
    decimal [ ] input,
    bool [ ] mask_vector,
    long nu )
```

This method predicts the class ID.

## Parameters

<i>input</i>	The input vector the class ID of which is to be predicted.
<i>mask_vector</i>	The mask vector corresponding to <i>input</i> .
<i>nu</i>	The maximum cardinality of the set of enclosing categories E and the neighbourhood set N. (This parameter does not modify the network. It may be arbitrarily changed in each prediction step.)

## Returns

An object of type [TopoART\\_C\\_Prediction](#) containing the predicted class ID and a corresponding confidence value.

Implements [LibTopoART.ITopoART\\_C](#).

## 5.18.4 Member Data Documentation

## 5.18.4.1 file\_format\_version

```
new const decimal LibTopoART.TopoART_C.file_format_version = 0.01m
```

Instance variable `file_format_version` represents the version of the file format used by class [TopoART\\_C](#).

## 5.18.4.2 UNDEFINED\_CLASS\_ID

```
const long LibTopoART.TopoART_C.UNDEFINED_CLASS_ID = -2
```

Instance variable `UNDEFINED_CLASS_ID` gives the value used for indicating that an input sample was predict to belong to the undefined class; i.e, no class ID was provided for such input samples during training.

## 5.19 LibTopoART.TopoART\_C\_Prediction Struct Reference

Struct [TopoART\\_C\\_Prediction](#) contains a prediction made by a TopoART-C network.

## Public Member Functions

- [TopoART\\_C\\_Prediction](#) (long [class\\_ID](#), decimal [confidence](#))

*This constructor sets the instance variables `class_ID` and `confidence` of struct [TopoART\\_C\\_Prediction](#).*

## Public Attributes

- long [class\\_ID](#)

*Instance variable `class_ID` gives the predicted class ID.*

- decimal [confidence](#)

*Instance variable `confidence` provides a confidence for the predicted class ID.*

### 5.19.1 Detailed Description

Struct [TopoART\\_C\\_Prediction](#) contains a prediction made by a TopoART-C network.

### 5.19.2 Constructor & Destructor Documentation

#### 5.19.2.1 TopoART\_C\_Prediction()

```
LibTopoART.TopoART_C_Prediction.TopoART_C_Prediction (
    long class_ID,
    decimal confidence )
```

This constructor sets the instance variables `class_ID` and `confidence` of struct [TopoART\\_C\\_Prediction](#).

#### Parameters

<code>class_ID</code>	The class ID to be set.
<code>confidence</code>	The value of the confidence to be set.

### 5.19.3 Member Data Documentation

#### 5.19.3.1 class\_ID

```
long LibTopoART.TopoART_C_Prediction.class_ID
```

Instance variable `class_ID` gives the predicted class ID.

## 5.19.3.2 confidence

```
decimal LibTopoART.TopoART_C_Prediction.confidence
```

Instance variable `confidence` provides a confidence for the predicted class ID.

## 5.20 LibTopoART\_samples.TopoART\_C\_sample1 Class Reference

Classification using TopoART-C. (simple classification task)

## 5.20.1 Detailed Description

Classification using TopoART-C. (simple classification task)

This sample demonstrates training and several possibilities for prediction at the example of a simple classification task.

## 5.21 LibTopoART\_samples.TopoART\_C\_sample2 Class Reference

Sample using artificial two-dimensional data with associated class IDs.

## 5.21.1 Detailed Description

Sample using artificial two-dimensional data with associated class IDs.

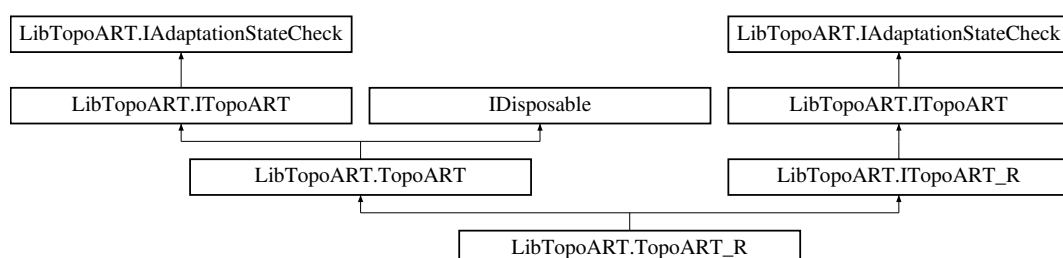
Train TopoART-C with a two-dimensional dataset similar to the one used in "Marko Tscherepanow and Sören Riechers, 'An Incremental On-line Classifier for Imbalanced, Incomplete, and Noisy Data', European Conference on Artificial Intelligence (ECAI), Workshop on Active and Incremental Learning (AIL), pp. 18-23, 2012." This dataset comprises six clusters (each containing 15,000 samples) as well as 10,000 noise samples which were mixed randomly. The samples are divided into two classes. Each sample belonging to one of the six clusters is assigned a class ID depending on its position. In contrast, noise samples receive a random class ID.

The resulting neural network can be visualised using the R script `ShowTopoARTCResults.R` which is provided in the subfolder `R`.

## 5.22 LibTopoART.TopoART\_R Class Reference

Class `TopoART_R` provides an implementation of the TopoART-R neural network as proposed in "Tscherepanow, Marko (2011). An Extended TopoART Network for the Stable On-Line Learning of Regression Functions. In Proceedings of the International Conference on Neural Information Processing (ICONIP), LNCS 7063, pp. 562–571. Berlin, Germany: Springer."

Inheritance diagram for LibTopoART.TopoART\_R:



## Public Member Functions

- [TopoART\\_R](#) (long *i\_dimension*, long *d\_dimension*, long *module\_number*, decimal *rho\_a*)  
*This constructor initialises a TopoART-R network.*
- [TopoART\\_R](#) (string *path*)  
*This constructor loads a saved TopoART-R network.*
- override void [Learn](#) (decimal[] *input*)  
*This method performs a single training step. The independent variables and the dependent variables are automatically separated.*
- void [Learn](#) (decimal[] *i\_vec*, decimal[] *d\_vec*)  
*This method performs a single training step.*
- decimal[] [Predict](#) (decimal[] *i\_vec*, long *nu*=10)  
*This method predicts the dependent variables.*
- [TopoART\\_R\\_Prediction Predict](#) (decimal[] *i\_vec*, bool[] *m\_i\_vec*, long *nu*=10)  
*This method predicts the dependent variables for a given set of independent variables. Unknown values of independent variables can be signified by setting the corresponding value of *m\_i\_vec* to `true`.*

## Public Attributes

- new const decimal [file\\_format\\_version](#) = 0.01m  
*Instance variable `file_format_version` represents the version of the file format used by class [TopoART\\_R](#).*

## Additional Inherited Members

### 5.22.1 Detailed Description

Class [TopoART\\_R](#) provides an implementation of the TopoART-R neural network as proposed in "Tscherepanow, Marko (2011). An Extended [TopoART](#) Network for the Stable On-Line Learning of Regression Functions. In Proceedings of the International Conference on Neural Information Processing (ICONIP), LNCS 7063, pp. 562–571. Berlin, Germany: Springer."

Class [TopoART\\_R](#) requires all input and output to lie in the interval [0,1].

### 5.22.2 Constructor & Destructor Documentation

#### 5.22.2.1 [TopoART\\_R\(\)](#) [1/2]

```
LibTopoART.TopoART_R.TopoART_R (
    long i_dimension,
    long d_dimension,
    long module_number,
    decimal rho_a )
```

This constructor initialises a TopoART-R network.

#### Parameters

<i>i_dimension</i>	The dimension of the input vector (independent variables) to be learnt.
<i>d_dimension</i>	The dimension of the output vector (dependent variables) to be learnt.
<i>module_number</i>	The number of TopoART-R modules. <span style="float: right;">Generated on Sat Apr 22 2017 13:57:32 for LibTopoART by Doxygen</span>
<i>rho_a</i>	The vigilance parameter of the first TopoART-R module (TopoART-R a).



## 5.22.2.2 TopoART\_R() [2/2]

```
LibTopoART.TopoART_R.TopoART_R (
    string path )
```

This constructor loads a saved TopoART-R network.

## Parameters

<i>path</i>	The path of a binary TopoART-R file.
-------------	--------------------------------------

## 5.22.3 Member Function Documentation

## 5.22.3.1 Learn() [1/2]

```
override void LibTopoART.TopoART_R.Learn (
    decimal [] input ) [virtual]
```

This method performs a single training step. The independent variables and the dependent variables are automatically separated.

## Parameters

<i>input</i>	The input vector to be learnt.
--------------	--------------------------------

Reimplemented from [LibTopoART.TopoART](#).

## 5.22.3.2 Learn() [2/2]

```
void LibTopoART.TopoART_R.Learn (
    decimal [] i_vec,
    decimal [] d_vec )
```

This method performs a single training step.

## Parameters

<i>i_vec</i>	The input vector (independent variables) to be learnt.
<i>d_vec</i>	The output vector (dependent variables) corresponding to <i>i_vec</i> .

Implements [LibTopoART.ITopoART\\_R](#).

### 5.22.3.3 Predict() [1/2]

```
decimal [ ] LibTopoART.TopoART_R.Predict (
    decimal [ ] i_vec,
    long nu = 10 )
```

This method predicts the dependent variables.

#### Parameters

<i>i_vec</i>	The input vector (independent variables).
<i>nu</i>	The maximum cardinality of the neighbourhood set N. (In the original TopoART-R network, nu is fixed to 10. But task-specific adaptations might lead to an improved prediction accuracy. This parameter does not modify the network. It may be arbitrarily changed in each prediction step.)

#### Returns

The predicted values for all dependent variables.

Implements [LibTopoART.ITopoART\\_R](#).

### 5.22.3.4 Predict() [2/2]

```
TopoART_R_Prediction LibTopoART.TopoART_R.Predict (
    decimal [ ] i_vec,
    bool [ ] m_i_vec,
    long nu = 10 )
```

This method predicts the dependent variables for a given set of independent variables. Unknown values of independent variables can be signified by setting the corresponding value of *m\_i\_vec* to `true`.

#### Parameters

<i>i_vec</i>	The input vector (independent variables).
<i>m_i_vec</i>	The mask vector corresponding to <i>i_vec</i> .
<i>nu</i>	The maximum cardinality of the neighbourhood set N. (In the original TopoART-R network, nu is fixed to 10. But task-specific adaptations might lead to an improved prediction accuracy. This parameter does not alter the network. It may be arbitrarily changed in each prediction step.)

#### Returns

An object of type [TopoART\\_R\\_Prediction](#) containing the predicted values for the unknown independent variables and all dependent variables.

Implements [LibTopoART.ITopoART\\_R](#).

## 5.22.4 Member Data Documentation

## 5.22.4.1 file\_format\_version

```
new const decimal LibTopoART.TopoART_R.file_format_version = 0.01m
```

Instance variable `file_format_version` represents the version of the file format used by class [TopoART\\_R](#).

## 5.23 LibTopoART.TopoART\_R\_Prediction Struct Reference

Struct [TopoART\\_R\\_Prediction](#) contains a prediction made by a TopoART-R network.

## Public Member Functions

- [TopoART\\_R\\_Prediction](#) (decimal[] `i_vec_prediction`, decimal[] `d_vec_prediction`)  
*This constructor sets the instance variables `i_vec_prediction` and `d_vec_prediction` of struct [TopoART\\_R\\_Prediction](#).*
- void [PrintPredictions](#) ()  
*This method prints the predictions on the console.*

## Public Attributes

- const decimal [NO\\_PREDICTION](#) = [LibTopoART\\_info.UNDEFINED](#)  
*Instance variable `NO_PREDICTION` provides a default prediction to signify variables that are presented to the network; i.e., these variables are known and no prediction is computed for them.*
- decimal [] [i\\_vec\\_prediction](#)  
*Instance variable `i_vec_prediction` represents predictions for unknown independent variables.*
- decimal [] [d\\_vec\\_prediction](#)  
*Instance variable `d_vec_prediction` provides the predictions for the dependent variables.*

## 5.23.1 Detailed Description

Struct [TopoART\\_R\\_Prediction](#) contains a prediction made by a TopoART-R network.

## 5.23.2 Constructor &amp; Destructor Documentation

## 5.23.2.1 TopoART\_R\_Prediction()

```
LibTopoART.TopoART_R_Prediction.TopoART_R_Prediction (
    decimal [] i_vec_prediction,
    decimal [] d_vec_prediction )
```

This constructor sets the instance variables `i_vec_prediction` and `d_vec_prediction` of struct [TopoART\\_R\\_Prediction](#).

**Parameters**

<i>i_vec_prediction</i>	The prediction results for the independent variables to be set.
<i>d_vec_prediction</i>	The prediction results for the dependent variables to be set.

**5.23.3 Member Function Documentation****5.23.3.1 PrintPredictions()**

```
void LibTopoART.TopoART_R_Prediction.PrintPredictions ( )
```

This method prints the predictions on the console.

**5.23.4 Member Data Documentation****5.23.4.1 d\_vec\_prediction**

```
decimal [ ] LibTopoART.TopoART_R_Prediction.d_vec_prediction
```

Instance variable `d_vec_prediction` provides the predictions for the dependent variables.

**5.23.4.2 i\_vec\_prediction**

```
decimal [ ] LibTopoART.TopoART_R_Prediction.i_vec_prediction
```

Instance variable `i_vec_prediction` represents predictions for unknown independent variables.

**5.23.4.3 NO\_PREDICTION**

```
const decimal LibTopoART.TopoART_R_Prediction.NO_PREDICTION = LibTopoART_info.UNDEFINED
```

Instance variable `NO_PREDICTION` provides a default prediction to signify variables that are presented to the network; i.e., these variables are known and no prediction is computed for them.

**5.24 LibTopoART\_samples.TopoART\_R\_sample1 Class Reference**

Regression using TopoART-R. (simplified version)

### 5.24.1 Detailed Description

Regression using TopoART-R. (simplified version)

This sample trains a TopoART-R network with 100 points sampled from a sine function. Then, sine values are predicted for 25 random values.

The predicted results can be visualised using the R script `ShowTopoARTResults.R` provided in the subfolder `R`.

## 5.25 LibTopoART\_samples.TopoART\_R\_sample2 Class Reference

Regression using TopoART-R. (advanced version)

### 5.25.1 Detailed Description

Regression using TopoART-R. (advanced version)

This sample trains a TopoART-R network with 100 points sampled from a sine function. Then, sine values are predicted for 25 random values.

The predicted results can be visualised using the R script `ShowTopoARTResults.R` provided in the subfolder `R`.

## 5.26 LibTopoART\_samples.TopoART\_sample1 Class Reference

Simple TopoART sample.

### 5.26.1 Detailed Description

Simple TopoART sample.

First, a dataset comprised of 10 samples is learned by a TopoART network. Afterwards, the training samples are slightly modified by random values and used for predicting cluster labels.

## 5.27 LibTopoART\_samples.TopoART\_sample2 Class Reference

Sample using artificial two-dimensional data.

### 5.27.1 Detailed Description

Sample using artificial two-dimensional data.

Train TopoART or Hypersphere TopoART with a two-dimensional dataset similar to the one used in "Marko Tscherepanow and Sören Riechers, 'An Incremental On-line Classifier for Imbalanced, Incomplete, and Noisy Data', European Conference on Artificial Intelligence (ECAI), Workshop on Active and Incremental Learning (AIL), pp. 18-23, 2012." This dataset comprises six clusters (each containing 15,000 samples) as well as 10,000 noise samples. These samples were mixed randomly.

The resulting neural network can be visualised using the R script `ShowTopoARTResults.R` or the R script `ShowHypersphereTopoARTResults.R`, respectively. Both R scripts are provided in the subfolder `R`.



## Index

AdaptationState  
    LibTopoART, 6

BeginRecall  
    LibTopoART::Episodic\_TopoART, 9  
    LibTopoART::IEpisodic\_recall, 28

bm\_cluster\_ID  
    LibTopoART::F2\_output, 12

bm\_node\_ID  
    LibTopoART::F2\_output, 13

bm\_node\_activation  
    LibTopoART::F2\_output, 12

bm\_permanent\_cluster\_ID  
    LibTopoART::F2\_output, 13

bm\_permanent\_node\_ID  
    LibTopoART::F2\_output, 13

bm\_permanent\_node\_activation  
    LibTopoART::F2\_output, 13

class\_ID  
    LibTopoART::TopoART\_C\_Prediction, 50

ComputeClusterIDs  
    LibTopoART::Fast\_TopoART\_base, 17  
    LibTopoART::ITopoART, 32  
    LibTopoART::TopoART, 42

confidence  
    LibTopoART::TopoART\_C\_Prediction, 50

d\_vec\_prediction  
    LibTopoART::TopoART\_R\_Prediction, 56

Dispose  
    LibTopoART::Fast\_TopoART\_base, 17  
    LibTopoART::TopoART, 42

EndRecall  
    LibTopoART::Episodic\_TopoART, 9  
    LibTopoART::IEpisodic\_recall, 29

Episodic\_TopoART  
    LibTopoART::Episodic\_TopoART, 8

F2\_output  
    LibTopoART::F2\_output, 12

Fast\_TopoART\_C  
    LibTopoART::Fast\_TopoART\_C, 22

Fast\_TopoART  
    LibTopoART::Fast\_TopoART, 14

file\_format\_version  
    LibTopoART::Episodic\_TopoART, 10  
    LibTopoART::Fast\_TopoART\_base, 20  
    LibTopoART::Fast\_TopoART\_C, 24  
    LibTopoART::Hypersphere\_TopoART, 26  
    LibTopoART::TopoART\_C, 49  
    LibTopoART::TopoART\_R, 54  
    LibTopoART::TopoART, 45

float\_base\_type  
    LibTopoART::Fast\_TopoART\_base, 20

LibTopoART::TopoART, 45

GetAdaptationState  
    LibTopoART::Fast\_TopoART\_base, 17  
    LibTopoART::IAdaptationStateCheck, 27  
    LibTopoART::TopoART, 42

GetBMOutput  
    LibTopoART::Fast\_TopoART\_base, 18  
    LibTopoART::ITopoART, 32  
    LibTopoART::TopoART, 43

Hypersphere\_TopoART  
    LibTopoART::Hypersphere\_TopoART, 25, 26

i\_vec\_prediction  
    LibTopoART::TopoART\_R\_Prediction, 56

integer\_base\_type  
    LibTopoART::Fast\_TopoART\_base, 20  
    LibTopoART::TopoART, 45

InterEpisodeRecallStep  
    LibTopoART::Episodic\_TopoART, 9  
    LibTopoART::IEpisodic\_recall, 29

IntraEpisodeRecallStep  
    LibTopoART::Episodic\_TopoART, 10  
    LibTopoART::IEpisodic\_recall, 29

Learn  
    LibTopoART::Episodic\_TopoART, 10  
    LibTopoART::Fast\_TopoART\_base, 19  
    LibTopoART::Fast\_TopoART\_C, 22, 23  
    LibTopoART::Fast\_TopoART, 15  
    LibTopoART::ITopoART\_C, 35  
    LibTopoART::ITopoART\_R, 37  
    LibTopoART::ITopoART, 33  
    LibTopoART::TopoART\_C, 47, 48  
    LibTopoART::TopoART\_R, 53  
    LibTopoART::TopoART, 44

LibTopoART.Episodic\_TopoART, 7

LibTopoART.F2\_output, 11

LibTopoART.Fast\_TopoART\_base, 15

LibTopoART.Fast\_TopoART\_C, 21

LibTopoART.Fast\_TopoART, 13

LibTopoART.Hypersphere\_TopoART, 24

LibTopoART.IAdaptationStateCheck, 27

LibTopoART.IEpisodic\_recall, 28

LibTopoART.ITopoART\_C, 34

LibTopoART.ITopoART\_R, 36

LibTopoART.ITopoART, 31

LibTopoART.InvalidClassIDException, 30

LibTopoART.InvalidStateException, 30

LibTopoART.LibTopoART\_info, 38

LibTopoART.TopoART\_C\_Prediction, 49

LibTopoART.TopoART\_R\_Prediction, 55

LibTopoART.TopoART\_C, 46

LibTopoART.TopoART\_R, 51

LibTopoART.TopoART, 39

- LibTopoART::Episodic\_TopoART
  - BeginRecall, [9](#)
  - EndRecall, [9](#)
  - Episodic\_TopoART, [8](#)
  - file\_format\_version, [10](#)
  - InterEpisodeRecallStep, [9](#)
  - IntraEpisodeRecallStep, [10](#)
  - Learn, [10](#)
- LibTopoART::F2\_output
  - bm\_cluster\_ID, [12](#)
  - bm\_node\_ID, [13](#)
  - bm\_node\_activation, [12](#)
  - bm\_permanent\_cluster\_ID, [13](#)
  - bm\_permanent\_node\_ID, [13](#)
  - bm\_permanent\_node\_activation, [13](#)
  - F2\_output, [12](#)
- LibTopoART::Fast\_TopoART\_base
  - ComputeClusterIDs, [17](#)
  - Dispose, [17](#)
  - file\_format\_version, [20](#)
  - float\_base\_type, [20](#)
  - GetAdaptationState, [17](#)
  - GetBMOutput, [18](#)
  - integer\_base\_type, [20](#)
  - Learn, [19](#)
  - ModuleNum, [20](#)
  - Phi, [20](#)
  - ResetAdaptationState, [19](#)
  - Save, [19](#)
  - SaveText, [19](#)
- LibTopoART::Fast\_TopoART\_C
  - Fast\_TopoART\_C, [22](#)
  - file\_format\_version, [24](#)
  - Learn, [22, 23](#)
  - Predict, [23](#)
  - UNDEFINED\_CLASS\_ID, [24](#)
- LibTopoART::Fast\_TopoART
  - Fast\_TopoART, [14](#)
  - Learn, [15](#)
- LibTopoART::Hypersphere\_TopoART
  - file\_format\_version, [26](#)
  - Hypersphere\_TopoART, [25, 26](#)
- LibTopoART::IAdaptationStateCheck
  - GetAdaptationState, [27](#)
  - ResetAdaptationState, [27](#)
- LibTopoART::IEpisodic\_recall
  - BeginRecall, [28](#)
  - EndRecall, [29](#)
  - InterEpisodeRecallStep, [29](#)
  - IntraEpisodeRecallStep, [29](#)
- LibTopoART::ITopoART\_C
  - Learn, [35](#)
  - Predict, [35](#)
- LibTopoART::ITopoART\_R
  - Learn, [37](#)
  - Predict, [37](#)
- LibTopoART::ITopoART
  - ComputeClusterIDs, [32](#)
  - GetBMOutput, [32](#)
  - Learn, [33](#)
  - ModuleNum, [34](#)
  - Phi, [34](#)
  - Save, [33](#)
  - SaveText, [33](#)
- LibTopoART::LibTopoART\_info
  - networks, [38](#)
  - UNDEFINED, [39](#)
  - version, [39](#)
- LibTopoART::TopoART\_C\_Prediction
  - class\_ID, [50](#)
  - confidence, [50](#)
  - TopoART\_C\_Prediction, [50](#)
- LibTopoART::TopoART\_R\_Prediction
  - d\_vec\_prediction, [56](#)
  - i\_vec\_prediction, [56](#)
  - NO\_PREDICTION, [56](#)
  - PrintPredictions, [56](#)
  - TopoART\_R\_Prediction, [55](#)
- LibTopoART::TopoART\_C
  - file\_format\_version, [49](#)
  - Learn, [47, 48](#)
  - Predict, [48, 49](#)
  - TopoART\_C, [47](#)
  - UNDEFINED\_CLASS\_ID, [49](#)
- LibTopoART::TopoART\_R
  - file\_format\_version, [54](#)
  - Learn, [53](#)
  - Predict, [53, 54](#)
  - TopoART\_R, [52, 53](#)
- LibTopoART::TopoART
  - ComputeClusterIDs, [42](#)
  - Dispose, [42](#)
  - file\_format\_version, [45](#)
  - float\_base\_type, [45](#)
  - GetAdaptationState, [42](#)
  - GetBMOutput, [43](#)
  - integer\_base\_type, [45](#)
  - Learn, [44](#)
  - ModuleNum, [45](#)
  - Phi, [45](#)
  - ResetAdaptationState, [44](#)
  - Save, [44](#)
  - SaveText, [44](#)
  - TopoART, [41](#)
- LibTopoART\_samples, [6](#)
- LibTopoART\_samples.Episodic\_TopoART\_sample1, [11](#)
- LibTopoART\_samples.Episodic\_TopoART\_sample2, [11](#)
- LibTopoART\_samples.TopoART\_C\_sample1, [51](#)
- LibTopoART\_samples.TopoART\_C\_sample2, [51](#)
- LibTopoART\_samples.TopoART\_R\_sample1, [56](#)
- LibTopoART\_samples.TopoART\_R\_sample2, [57](#)
- LibTopoART\_samples.TopoART\_sample1, [57](#)
- LibTopoART\_samples.TopoART\_sample2, [57](#)
- LibTopoART, [4](#)
  - AdaptationState, [6](#)
- ModuleNum



- LibTopoART::Fast\_TopoART\_base, [20](#)
- LibTopoART::ITopoART, [34](#)
- LibTopoART::TopoART, [45](#)

NO\_PREDICTION

- LibTopoART::TopoART\_R\_Prediction, [56](#)

networks

- LibTopoART::LibTopoART\_info, [38](#)

Phi

- LibTopoART::Fast\_TopoART\_base, [20](#)
- LibTopoART::ITopoART, [34](#)
- LibTopoART::TopoART, [45](#)

Predict

- LibTopoART::Fast\_TopoART\_C, [23](#)
- LibTopoART::ITopoART\_C, [35](#)
- LibTopoART::ITopoART\_R, [37](#)
- LibTopoART::TopoART\_C, [48](#), [49](#)
- LibTopoART::TopoART\_R, [53](#), [54](#)

PrintPredictions

- LibTopoART::TopoART\_R\_Prediction, [56](#)

ResetAdaptationState

- LibTopoART::Fast\_TopoART\_base, [19](#)
- LibTopoART::!AdaptationStateCheck, [27](#)
- LibTopoART::TopoART, [44](#)

Save

- LibTopoART::Fast\_TopoART\_base, [19](#)
- LibTopoART::ITopoART, [33](#)
- LibTopoART::TopoART, [44](#)

SaveText

- LibTopoART::Fast\_TopoART\_base, [19](#)
- LibTopoART::ITopoART, [33](#)
- LibTopoART::TopoART, [44](#)

TopoART\_C\_Prediction

- LibTopoART::TopoART\_C\_Prediction, [50](#)

TopoART\_R\_Prediction

- LibTopoART::TopoART\_R\_Prediction, [55](#)

TopoART\_C

- LibTopoART::TopoART\_C, [47](#)

TopoART\_R

- LibTopoART::TopoART\_R, [52](#), [53](#)

TopoART

- LibTopoART::TopoART, [41](#)

UNDEFINED\_CLASS\_ID

- LibTopoART::Fast\_TopoART\_C, [24](#)
- LibTopoART::TopoART\_C, [49](#)

UNDEFINED

- LibTopoART::LibTopoART\_info, [39](#)

version

- LibTopoART::LibTopoART\_info, [39](#)