

LibTopoART 1.1 Reference Manual

Generated by Doxygen 1.18.0

1 Directory Hierarchy	2
1.1 Directories	2
2 Namespace Index	2
2.1 Package List	2
3 Hierarchical Index	2
3.1 Class Hierarchy	2
4 Class Index	5
4.1 Class List	5
5 Directory Documentation	9
5.1 LibTopoART Directory Reference	9
5.1.1 Detailed Description	9
6 Namespace Documentation	9
6.1 LibTopoART Namespace Reference	9
6.1.1 Enumeration Type Documentation	13
7 Class Documentation	13
7.1 LibTopoART.CategoryInfo Struct Reference	13
7.1.1 Detailed Description	14
7.1.2 Constructor & Destructor Documentation	14
7.2 LibTopoART.F2_output Class Reference	14
7.2.1 Detailed Description	15
7.3 LibTopoART.Fast_Episodic_TopoART Class Reference	15
7.3.1 Detailed Description	18
7.3.2 Constructor & Destructor Documentation	19
7.3.3 Member Function Documentation	20
7.4 LibTopoART.Fast_TopoART Class Reference	23
7.4.1 Detailed Description	25
7.4.2 Constructor & Destructor Documentation	25
7.4.3 Member Function Documentation	26
7.5 LibTopoART.Fast_TopoART_AM Class Reference	27
7.5.1 Detailed Description	31
7.5.2 Constructor & Destructor Documentation	31
7.5.3 Member Function Documentation	32
7.6 LibTopoART.Fast_TopoART_base Class Reference	36
7.6.1 Detailed Description	39
7.6.2 Member Function Documentation	39
7.7 LibTopoART.Fast_TopoART_C Class Reference	44
7.7.1 Detailed Description	48
7.7.2 Constructor & Destructor Documentation	49

7.7.3 Member Function Documentation	50
7.8 LibTopoART.Fast_TopoART_R Class Reference	54
7.8.1 Detailed Description	58
7.8.2 Constructor & Destructor Documentation	59
7.8.3 Member Function Documentation	60
7.9 LibTopoART.Hypersphere_TopoART Class Reference	64
7.9.1 Detailed Description	67
7.9.2 Constructor & Destructor Documentation	67
7.10 LibTopoART.Hypersphere_TopoART_C Class Reference	69
7.10.1 Detailed Description	73
7.10.2 Constructor & Destructor Documentation	73
7.10.3 Member Function Documentation	74
7.11 LibTopoART.IAccess_TopoART< TAccessType > Interface Template Reference	77
7.11.1 Detailed Description	78
7.11.2 Member Function Documentation	78
7.12 LibTopoART.IAccess_TopoART_AM< TAccessType > Interface Template Reference	79
7.12.1 Detailed Description	81
7.12.2 Member Function Documentation	82
7.13 LibTopoART.IAccess_TopoART_C< TAccessType > Interface Template Reference	82
7.13.1 Detailed Description	84
7.13.2 Member Function Documentation	85
7.14 LibTopoART.IAccess_TopoART_R< TAccessType > Interface Template Reference	86
7.14.1 Detailed Description	88
7.14.2 Member Function Documentation	89
7.15 LibTopoART.IAccessAssociativeRecall< TAccessType > Interface Template Reference	90
7.15.1 Detailed Description	91
7.15.2 Member Function Documentation	92
7.16 LibTopoART.IAccessEpisodicRecall< TAccessType > Interface Template Reference	93
7.16.1 Detailed Description	94
7.16.2 Member Function Documentation	94
7.17 LibTopoART.IAdaptationStateCheck Interface Reference	95
7.17.1 Detailed Description	96
7.17.2 Member Function Documentation	96
7.18 LibTopoART.IAssociativeRecall Interface Reference	96
7.18.1 Detailed Description	97
7.19 LibTopoART.ICategoryAccess Interface Reference	98
7.19.1 Detailed Description	98
7.19.2 Member Function Documentation	98
7.20 LibTopoART.IEndRecall Interface Reference	99
7.20.1 Detailed Description	99
7.21 LibTopoART.IEpisodic_TopoART Interface Reference	99
7.21.1 Detailed Description	101

7.22 LibTopoART.IEpisodicRecall Interface Reference	101
7.22.1 Detailed Description	103
7.23 LibTopoART.IFast_Episodic_TopoART Interface Reference	103
7.23.1 Detailed Description	104
7.24 LibTopoART.IFast_TopoART Interface Reference	105
7.24.1 Detailed Description	106
7.25 LibTopoART.IFast_TopoART_AM Interface Reference	106
7.25.1 Detailed Description	108
7.26 LibTopoART.IFast_TopoART_C Interface Reference	108
7.26.1 Detailed Description	110
7.27 LibTopoART.IFast_TopoART_R Interface Reference	110
7.27.1 Detailed Description	112
7.28 LibTopoART.IFastAssociativeRecall Interface Reference	113
7.28.1 Detailed Description	114
7.29 LibTopoART.IFastEpisodicRecall Interface Reference	114
7.29.1 Detailed Description	115
7.30 LibTopoART.IHypersphere_TopoART Interface Reference	115
7.30.1 Detailed Description	117
7.31 LibTopoART.InvalidClassIDException Class Reference	117
7.31.1 Detailed Description	118
7.32 LibTopoART.InvalidFileException Class Reference	118
7.32.1 Detailed Description	118
7.33 LibTopoART.InvalidModuleIndexException Class Reference	118
7.33.1 Detailed Description	118
7.34 LibTopoART.InvalidNumberException Class Reference	118
7.34.1 Detailed Description	118
7.35 LibTopoART.InvalidSizeException Class Reference	118
7.35.1 Detailed Description	118
7.36 LibTopoART.InvalidStateException Class Reference	118
7.36.1 Detailed Description	119
7.37 LibTopoART.InvalidTypeException Class Reference	119
7.37.1 Detailed Description	119
7.38 LibTopoART.ITopoART Interface Reference	119
7.38.1 Detailed Description	120
7.39 LibTopoART.ITopoART_AM Interface Reference	120
7.39.1 Detailed Description	122
7.40 LibTopoART.ITopoART_AM_base Interface Reference	122
7.40.1 Detailed Description	124
7.41 LibTopoART.ITopoART_base Interface Reference	124
7.41.1 Detailed Description	125
7.41.2 Member Function Documentation	125
7.41.3 Property Documentation	126

7.42 LibTopoART.ITopoART_base_stream Interface Reference	126
7.42.1 Detailed Description	127
7.42.2 Member Function Documentation	127
7.43 LibTopoART.ITopoART_C Interface Reference	128
7.43.1 Detailed Description	129
7.44 LibTopoART.ITopoART_C_base Interface Reference	129
7.44.1 Detailed Description	131
7.45 LibTopoART.ITopoART_R Interface Reference	131
7.45.1 Detailed Description	133
7.46 LibTopoART.ITopoART_R_base Interface Reference	133
7.46.1 Detailed Description	134
7.47 LibTopoART.LibTopoART_control Struct Reference	135
7.47.1 Detailed Description	135
7.48 LibTopoART.LibTopoART_info Struct Reference	135
7.48.1 Detailed Description	135
7.49 LibTopoART.Network_base Class Reference	135
7.49.1 Detailed Description	136
7.49.2 Property Documentation	136
7.50 LibTopoART.TopoART Class Reference	137
7.50.1 Detailed Description	139
7.50.2 Constructor & Destructor Documentation	139
7.50.3 Member Function Documentation	140
7.51 LibTopoART.TopoART_C Class Reference	144
7.51.1 Detailed Description	148
7.51.2 Constructor & Destructor Documentation	148
7.51.3 Member Function Documentation	149
7.52 LibTopoART.TopoART_C_prediction Struct Reference	151
7.52.1 Detailed Description	152
7.52.2 Constructor & Destructor Documentation	152
7.53 LibTopoART.TopoART_R Class Reference	152
7.53.1 Detailed Description	155
7.53.2 Constructor & Destructor Documentation	156
7.53.3 Member Function Documentation	157
7.54 LibTopoART.TopoART_R_prediction< TElementType > Struct Template Reference	159
7.54.1 Detailed Description	160
7.54.2 Member Function Documentation	160

Index	161
--------------	------------

1 Directory Hierarchy

1.1 Directories

LibTopoART	9
------------	---

2 Namespace Index

2.1 Package List

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

LibTopoART	9
----------------------------	---

3 Hierarchical Index

3.1 Class Hierarchy

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

LibTopoART.CategoryInfo	13
LibTopoART.F2_output	14
LibTopoART.IAdaptationStateCheck	95
LibTopoART.ITopoART	119
LibTopoART.IEpisodic_TopoART	99
LibTopoART.IFast_Episodic_TopoART	103
LibTopoART.Fast_Episodic_TopoART	15
LibTopoART.IFast_TopoART	105
LibTopoART.Fast_TopoART_base	36
LibTopoART.Fast_Episodic_TopoART	15
LibTopoART.Fast_TopoART	23
LibTopoART.Fast_TopoART_AM	27
LibTopoART.Fast_TopoART_C	44
LibTopoART.Fast_TopoART_R	54
LibTopoART.IFast_Episodic_TopoART	103
LibTopoART.IFast_TopoART_AM	106
LibTopoART.Fast_TopoART_AM	27

LibTopoART.IFast_TopoART_C	108
LibTopoART.Fast_TopoART_C	44
LibTopoART.IFast_TopoART_R	110
LibTopoART.Fast_TopoART_R	54
LibTopoART.IHypersphere_TopoART	115
LibTopoART.Hypersphere_TopoART	64
LibTopoART.Hypersphere_TopoART_C	69
LibTopoART.ITopoART_AM	120
LibTopoART.IFast_TopoART_AM	106
LibTopoART.ITopoART_C	128
LibTopoART.Hypersphere_TopoART_C	69
LibTopoART.IFast_TopoART_C	108
LibTopoART.TopoART_C	144
LibTopoART.ITopoART_R	131
LibTopoART.IFast_TopoART_R	110
LibTopoART.TopoART_R	152
LibTopoART.TopoART	137
LibTopoART.Hypersphere_TopoART	64
LibTopoART.TopoART_C	144
LibTopoART.TopoART_R	152
LibTopoART.ICategoryAccess	98
LibTopoART.Fast_TopoART_base	36
LibTopoART.TopoART	137
LibTopoART.IEndRecall	99
LibTopoART.IAccessAssociativeRecall< TAccessType >	90
LibTopoART.IAssociativeRecall	96
LibTopoART.IFastAssociativeRecall	113
LibTopoART.IFast_TopoART_AM	106
LibTopoART.ITopoART_AM	120
LibTopoART.IFastAssociativeRecall	113
LibTopoART.IAccessEpisodicRecall< TAccessType >	93
LibTopoART.IEpisodicRecall	101

LibTopoART.IEpisodic_TopoART	99
LibTopoART.IFastEpisodicRecall	114
LibTopoART.IFast_Episodic_TopoART	103
LibTopoART.IFastEpisodicRecall	114
LibTopoART.InvalidClassIDException	117
LibTopoART.InvalidFileException	118
LibTopoART.InvalidModuleIndexException	118
LibTopoART.InvalidNumberException	118
LibTopoART.InvalidSizeException	118
LibTopoART.InvalidStateException	118
LibTopoART.InvalidTypeException	119
LibTopoART.ITopoART_base	124
LibTopoART.IAccess_TopoART< TAccessType >	77
LibTopoART.IFast_TopoART	105
LibTopoART.ITopoART	119
LibTopoART.ITopoART_AM_base	122
LibTopoART.IAccess_TopoART_AM< TAccessType >	79
LibTopoART.IFast_TopoART_AM	106
LibTopoART.ITopoART_AM	120
LibTopoART.ITopoART_C_base	129
LibTopoART.IAccess_TopoART_C< TAccessType >	82
LibTopoART.IFast_TopoART_C	108
LibTopoART.ITopoART_C	128
LibTopoART.ITopoART_R_base	133
LibTopoART.IAccess_TopoART_R< TAccessType >	86
LibTopoART.IFast_TopoART_R	110
LibTopoART.ITopoART_R	131
LibTopoART.ITopoART_base_stream	126
LibTopoART.Fast_TopoART_base	36
LibTopoART.TopoART	137
LibTopoART.LibTopoART_control	135
LibTopoART.LibTopoART_info	135

LibTopoART.Network_base	135
LibTopoART.Fast_TopoART_base	36
LibTopoART.TopoART	137
LibTopoART.TopoART_C_prediction	151
LibTopoART.TopoART_R_prediction < TElementType >	159

4 Class Index

4.1 Class List

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

LibTopoART.CategoryInfo	
Struct CategoryInfo summarises information about a node's category	13
LibTopoART.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	14
LibTopoART.Fast_Episodic_TopoART	
Class Fast_Episodic_TopoART provides an implementation of the Episodic TopoART neural network as proposed in "Marko Tscherepanow, Sina Kühnel, and Sören Riechers (2012). Episodic Clustering of Data Streams Using a Topology-Learning Neural Network. In Proceedings of the European Conference on Artificial Intelligence (ECAI), Workshop on Active and Incremental Learning (AIL), pp. 24-29. Montpellier, France."	15
LibTopoART.Fast_TopoART	
Class Fast_TopoART provides an implementation of the TopoART neural network as proposed in "Marko Tscherepanow (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."	23
LibTopoART.Fast_TopoART_AM	
Class Fast_TopoART_AM provides an implementation of the TopoART-AM neural network as proposed in "Marko Tscherepanow, Marco Kortkamp and Marc Kammer (2011). A Hierarchical ART Network for the Stable Incremental Learning of Topological Structures and Associations from Noisy Data. Neural Networks 24(8): 906-916. Elsevier."	27
LibTopoART.Fast_TopoART_base	
Base class providing functionality common to several TopoART networks	36
LibTopoART.Fast_TopoART_C	
Class Fast_TopoART_C provides an implementation of the TopoART-C neural network as proposed in "Marko Tscherepanow and Sören Riechers (2012). An Incremental On-line Classifier for Imbalanced, Incomplete, and Noisy Data. In Proceedings of the European Conference on Artificial Intelligence (ECAI), Workshop on Active and Incremental Learning (AIL), pp. 18-23. Montpellier, France."	44
LibTopoART.Fast_TopoART_R	
Class Fast_TopoART_R provides an implementation of the TopoART-R neural network as proposed in "Marko Tscherepanow (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."	54

LibTopoART.Hypersphere_TopoART

Class **Hypersphere_TopoART** provides an implementation of the Hypersphere **TopoART** neural network as proposed in "Marko Tscherepanow (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."

64

LibTopoART.Hypersphere_TopoART_C

Class **Hypersphere_TopoART_C** provides an implementation of the Hypersphere **TopoART-C** neural network. Hypersphere **TopoART-C** is a combination of Hypersphere **TopoART** as proposed in "Marko Tscherepanow (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." and **TopoART-C** as proposed in "Marko Tscherepanow and Sören Riechers (2012). An Incremental On-line Classifier for Imbalanced, Incomplete, and Noisy Data. In Proceedings of the European Conference on Artificial Intelligence (ECAI), Workshop on Active and Incremental Learning (AIL), pp. 18-23. Montpellier, France."

69

LibTopoART.IAccess_TopoART< TAccessType >

Interface providing access to the basic **TopoART** functionality using input elements of type **_AccessType**

77

LibTopoART.IAccess_TopoART_AM< TAccessType >

Interface providing access to the basic **TopoART-AM** functionality using input elements of type **_AccessType**

79

LibTopoART.IAccess_TopoART_C< TAccessType >

Interface providing access to the basic **TopoART-C** functionality using input elements of type **_AccessType**

82

LibTopoART.IAccess_TopoART_R< TAccessType >

Interface providing access to the basic **TopoART-R** functionality using input elements of type **_AccessType**

86

LibTopoART.IAccessAssociativeRecall< TAccessType >

Interface providing access to the basic associative recall functionality using stimulus elements and recall result elements of type **TAccessType**

90

LibTopoART.IAccessEpisodicRecall< TAccessType >

Interface providing access to the basic episodic recall functionality using stimulus elements and recall result elements of type **_AccessType**

93

LibTopoART.IAdaptationStateCheck

Interface enabling checks whether certain adaptations of a network occurred

95

LibTopoART.IAssociativeRecall

Interface summarising the associative recall functionality using stimulus elements and recall result elements of type **decimal**

96

LibTopoART.ICategoryAccess

Interface providing access to the learnt categories, e.g., for drawing

98

LibTopoART.IEndRecall

Interface summarising the type-independent functionality to stop the recall process

99

LibTopoART.IEpisodic_TopoART

Interface summarising the Episodic **TopoART** functionality including learning, prediction, episodic recall using input elements, stimulus elements, and recall result elements of type **decimal** as well as adaptation state control

99

LibTopoART.IEpisodicRecall	Interface summarising the episodic recall functionality using stimulus elements and recall result elements of type <code>decimal</code>	101
LibTopoART.IFast_Episodic_TopoART	Interface summarising the Episodic TopoART functionality including learning, prediction, episodic recall using input elements, stimulus elements, and recall result elements of type <code>byte</code> or of type <code>decimal</code> as well as adaptation state control	103
LibTopoART.IFast_TopoART	Interface summarising the TopoART functionality including learning and prediction using input elements of type <code>byte</code> or of type <code>decimal</code> as well as adaptation state control	105
LibTopoART.IFast_TopoART_AM	Interface summarising the Episodic TopoART functionality including learning, prediction, episodic recall using input elements, stimulus elements, and recall result elements of type <code>byte</code> or of type <code>decimal</code> as well as adaptation state control	106
LibTopoART.IFast_TopoART_C	Interface summarising the TopoART-C functionality including learning and prediction using input elements of type <code>byte</code> or of type <code>decimal</code> as well as adaptation state control	108
LibTopoART.IFast_TopoART_R	Interface summarising the TopoART-R functionality including learning and prediction using input elements and output elements of type <code>byte</code> or of type <code>decimal</code> as well as adaptation state control	110
LibTopoART.IFastAssociativeRecall	Interface summarising the associative recall functionality using stimulus elements and recall result elements of type <code>byte</code> or of type <code>decimal</code>	113
LibTopoART.IFastEpisodicRecall	Interface summarising the episodic recall functionality using stimulus elements and recall result elements of type <code>byte</code> or of type <code>decimal</code>	114
LibTopoART.IHypersphere_TopoART	Interface summarising the Hypersphere TopoART functionality including learning and prediction using input elements of type <code>decimal</code> as well as adaptation state control	115
LibTopoART.InvalidClassIDException	Exception signalling an invalid class ID	117
LibTopoART.InvalidFileException	Exception signalling an invalid file	118
LibTopoART.InvalidModuleIndexException	Exception signalling an invalid module index	118
LibTopoART.InvalidNumberException	Exception signalling an invalid number	118
LibTopoART.InvalidSizeException	Exception signalling an invalid size	118
LibTopoART.InvalidStateException	Exception signalling an invalid state of the neural network	118
LibTopoART.InvalidTypeException	Exception signalling an invalid type	119

LibTopoART.ITopoART	Interface summarising the TopoART functionality including learning and prediction using input elements of type <code>decimal</code> as well as adaptation state control	119
LibTopoART.ITopoART_AM	Interface summarising the TopoART-AM functionality including learning, prediction, associative recall using input elements, stimulus elements, and recall result elements of type <code>decimal</code> as well as adaptation state control	120
LibTopoART.ITopoART_AM_base	Interface summarising the basic TopoART-AM functionality excluding learning and prediction	122
LibTopoART.ITopoART_base	Interface summarising the basic TopoART functionality excluding learning and prediction	124
LibTopoART.ITopoART_base_stream	Interface extending the basic TopoART functionality by stream-based saving	126
LibTopoART.ITopoART_C	Interface summarising the TopoART-C functionality including learning and prediction using input elements of type <code>decimal</code> as well as adaptation state control	128
LibTopoART.ITopoART_C_base	Interface summarising the basic TopoART-C functionality excluding learning and prediction	129
LibTopoART.ITopoART_R	Interface summarising the TopoART-R functionality including learning and prediction using input elements and output elements of type <code>decimal</code> as well as adaptation state control	131
LibTopoART.ITopoART_R_base	Interface summarising the basic TopoART-R functionality excluding learning and prediction	133
LibTopoART.LibTopoART_control	Struct LibTopoART_control provides fields to control the general behaviour of LibTopoART	135
LibTopoART.LibTopoART_info	Struct LibTopoART_info provides some metainformation regarding the respective implementation of LibTopoART	135
LibTopoART.Network_base	Class Network_base provides the functionality required by all neural network implementations of LibTopoART	135
LibTopoART.TopoART	Class TopoART provides an implementation of the TopoART neural network as proposed in "Marko Tscherepanow (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."	137
LibTopoART.TopoART_C	Class TopoART_C provides an implementation of the TopoART-C neural network as proposed in "Marko Tscherepanow and Sören Riechers (2012). An Incremental On-line Classifier for Imbalanced, Incomplete, and Noisy Data. In Proceedings of the European Conference on Artificial Intelligence (ECAI), Workshop on Active and Incremental Learning (AIL), pp. 18-23. Montpellier, France."	144
LibTopoART.TopoART_C_prediction	Struct TopoART_C_prediction contains a prediction made by a TopoART-C network	151

LibTopoART.TopoART_R

Class **TopoART_R** provides an implementation of the TopoART-R neural network as proposed in "Marko Tscherepanow (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."

152

LibTopoART.TopoART_R_prediction< TElementType >

Struct **TopoART_R_prediction** contains a prediction made by a TopoART-R network

159

5 Directory Documentation

5.1 LibTopoART Directory Reference

5.1.1 Detailed Description

LibTopoART provides implementations of several neural networks based on the TopoART architecture. This architecture has been developed as a unified machine learning approach tackling frequent problems arising in cognitive robotics and advanced machine learning, such as online-learning, lifelong learning from data streams, as well as incremental learning and prediction from non-stationary data, noisy data, imbalanced data, and incomplete data.

Besides the TopoART neural network itself, **LibTopoART** comprises implementations of Episodic TopoART (episodic clustering of data streams), Hypersphere TopoART (clustering and topology-learning), TopoART-AM (associative memory), TopoART-C and Hypersphere TopoART-C (classification), and TopoART-R (regression).

The reference manual, samples, and visualisation tools can be downloaded from the [LibTopoART website](#).

6 Namespace Documentation

6.1 LibTopoART Namespace Reference

Classes

- struct [CategoryInfo](#)

Struct *CategoryInfo* summarises information about a node's category.

- 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_Episodic_TopoART](#)

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

- class [Fast_TopoART](#)

Class *Fast_TopoART* provides an implementation of the [TopoART](#) neural network as proposed in "Marko Tscherepanow (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 [Fast_TopoART_AM](#)

Class `Fast_TopoART_AM` provides an implementation of the TopoART-AM neural network as proposed in "Marko Tscherepanow, Marco Kortkamp and Marc Kammer (2011). A Hierarchical ART Network for the Stable Incremental Learning of Topological Structures and Associations from Noisy Data. *Neural Networks* 24(8): 906-916. Elsevier."

- 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 (2012). An Incremental On-line Classifier for Imbalanced, Incomplete, and Noisy Data. In *Proceedings of the European Conference on Artificial Intelligence (ECAI), Workshop on Active and Incremental Learning (AIL)*, pp. 18-23. Montpellier, France."

- class `Fast_TopoART_R`

Class `Fast_TopoART_R` provides an implementation of the TopoART-R neural network as proposed in "Marko Tscherepanow (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 `Hypersphere_TopoART`

Class `Hypersphere_TopoART` provides an implementation of the Hypersphere `TopoART` neural network as proposed in "Marko Tscherepanow (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."

- class `Hypersphere_TopoART_C`

Class `Hypersphere_TopoART_C` provides an implementation of the Hypersphere TopoART-C neural network. Hypersphere TopoART-C is a combination of Hypersphere `TopoART` as proposed in "Marko Tscherepanow (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." and TopoART-C as proposed in "Marko Tscherepanow and Sören Riechers (2012). An Incremental On-line Classifier for Imbalanced, Incomplete, and Noisy Data. In *Proceedings of the European Conference on Artificial Intelligence (ECAI), Workshop on Active and Incremental Learning (AIL)*, pp. 18-23. Montpellier, France."

- interface `IAccess_TopoART< TAccessType >`

Interface providing access to the basic `TopoART` functionality using input elements of type `_AccessType`.

- interface `IAccess_TopoART_AM< TAccessType >`

Interface providing access to the basic TopoART-AM functionality using input elements of type `_AccessType`.

- interface `IAccess_TopoART_C< TAccessType >`

Interface providing access to the basic TopoART-C functionality using input elements of type `_AccessType`.

- interface `IAccess_TopoART_R< TAccessType >`

Interface providing access to the basic TopoART-R functionality using input elements of type `_AccessType`.

- interface `IAccessAssociativeRecall< TAccessType >`

Interface providing access to the basic associative recall functionality using stimulus elements and recall result elements of type `TAccessType`.

- interface `IAccessEpisodicRecall< TAccessType >`

Interface providing access to the basic episodic recall functionality using stimulus elements and recall result elements of type `_AccessType`.

- interface `IAdaptationStateCheck`

Interface enabling checks whether certain adaptations of a network occurred.

- interface `IAssociativeRecall`

Interface summarising the associative recall functionality using stimulus elements and recall result elements of type `decimal`.

- interface `ICategoryAccess`

Interface providing access to the learnt categories, e.g., for drawing.

- interface `IEndRecall`

Interface summarising the type-independent functionality to stop the recall process.

- interface `IEpisodic_TopoART`

Interface summarising the Episodic `TopoART` functionality including learning, prediction, episodic recall using input elements, stimulus elements, and recall result elements of type `decimal` as well as adaptation state control.

- interface [IEpisodicRecall](#)
Interface summarising the episodic recall functionality using stimulus elements and recall result elements of type `decimal`.
- interface [IFast_Episodic_TopoART](#)
Interface summarising the Episodic [TopoART](#) functionality including learning, prediction, episodic recall using input elements, stimulus elements, and recall result elements of type `byte` or of type `decimal` as well as adaptation state control.
- interface [IFast_TopoART](#)
Interface summarising the [TopoART](#) functionality including learning and prediction using input elements of type `byte` or of type `decimal` as well as adaptation state control.
- interface [IFast_TopoART_AM](#)
Interface summarising the Episodic [TopoART](#) functionality including learning, prediction, episodic recall using input elements, stimulus elements, and recall result elements of type `byte` or of type `decimal` as well as adaptation state control.
- interface [IFast_TopoART_C](#)
Interface summarising the TopoART-C functionality including learning and prediction using input elements of type `byte` or of type `decimal` as well as adaptation state control.
- interface [IFast_TopoART_R](#)
Interface summarising the TopoART-R functionality including learning and prediction using input elements and output elements of type `byte` or of type `decimal` as well as adaptation state control.
- interface [IFastAssociativeRecall](#)
Interface summarising the associative recall functionality using stimulus elements and recall result elements of type `byte` or of type `decimal`.
- interface [IFastEpisodicRecall](#)
Interface summarising the episodic recall functionality using stimulus elements and recall result elements of type `byte` or of type `decimal`.
- interface [IHypersphere_TopoART](#)
Interface summarising the Hypersphere [TopoART](#) functionality including learning and prediction using input elements of type `decimal` as well as adaptation state control.
- class [InvalidClassIDException](#)
Exception signalling an invalid class ID.
- class [InvalidFileException](#)
Exception signalling an invalid file.
- class [InvalidModuleIndexException](#)
Exception signalling an invalid module index.
- class [InvalidNumberException](#)
Exception signalling an invalid number.
- class [InvalidSizeException](#)
Exception signalling an invalid size.
- class [InvalidStateException](#)
Exception signalling an invalid state of the neural network.
- class [InvalidTypeException](#)
Exception signalling an invalid type.
- interface [ITopoART](#)
Interface summarising the [TopoART](#) functionality including learning and prediction using input elements of type `decimal` as well as adaptation state control.
- interface [ITopoART_AM](#)
Interface summarising the TopoART-AM functionality including learning, prediction, associative recall using input elements, stimulus elements, and recall result elements of type `decimal` as well as adaptation state control.
- interface [ITopoART_AM_base](#)
Interface summarising the basic TopoART-AM functionality excluding learning and prediction.
- interface [ITopoART_base](#)
Interface summarising the basic [TopoART](#) functionality excluding learning and prediction.

- interface [ITopoART_base_stream](#)
Interface extending the basic [TopoART](#) functionality by stream-based saving.
- interface [ITopoART_C](#)
Interface summarising the TopoART-C functionality including learning and prediction using input elements of type `decimal` as well as adaptation state control.
- interface [ITopoART_C_base](#)
Interface summarising the basic TopoART-C functionality excluding learning and prediction.
- interface [ITopoART_R](#)
Interface summarising the TopoART-R functionality including learning and prediction using input elements and output elements of type `decimal` as well as adaptation state control.
- interface [ITopoART_R_base](#)
Interface summarising the basic TopoART-R functionality excluding learning and prediction.
- struct [LibTopoART_control](#)
Struct [LibTopoART_control](#) provides fields to control the general behaviour of [LibTopoART](#).
- struct [LibTopoART_info](#)
Struct [LibTopoART_info](#) provides some metainformation regarding the respective implementation of [LibTopoART](#).
- class [Network_base](#)
Class [Network_base](#) provides the functionality required by all neural network implementations of [LibTopoART](#).
- class [TopoART](#)
Class [TopoART](#) provides an implementation of the [TopoART](#) neural network as proposed in "Marko Tscherepanow (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 (2012). An Incremental On-line Classifier for Imbalanced, Incomplete, and Noisy Data. In Proceedings of the European Conference on Artificial Intelligence (ECAI), Workshop on Active and Incremental Learning (AIL), pp. 18-23. Montpellier, France."
- 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 "Marko Tscherepanow (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](#) < [TElementType](#) >
Struct [TopoART_R_prediction](#) contains a prediction made by a TopoART-R network.

Enumerations

- enum [AdaptationState](#) {
[NO_ADAPTATION](#) = 0 ,
[ADDED_NODE_CANDIDATE](#) = 0x0001 ,
[ADAPTED_NONPERMANENT_WEIGHT](#) = 0x0002 ,
[ADDED_EDGE_CANDIDATE](#) = 0x0004 ,
[REMOVED_NODE_CANDIDATE](#) = 0x0008 ,
[REMOVED_EDGE_CANDIDATE](#) = 0x0010 ,
[ANY_NONPERMANENT_ADAPTATION_MASK](#) = 0x00ff ,
[ADDED_PERMANENT_NODE](#) = 0x0100 ,
[ADAPTED_PERMANENT_WEIGHT](#) = 0x0200 ,
[ADDED_PERMANENT_EDGE](#) = 0x0400 ,
[ANY_PERMANENT_ADAPTATION_MASK](#) = 0xff00 }

Enumeration specifying possible adaptation states.

- enum `VerbosityLevel` : uint {
`Important` ,
`Normal` ,
`Verbose` }

Enumeration specifying possible adaptation states.

6.1.1 Enumeration Type Documentation

AdaptationState

enum `LibTopoART.AdaptationState`

Enumeration specifying possible adaptation states.

Enumerator

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

VerbosityLevel

enum `LibTopoART.VerbosityLevel` : uint

Enumeration specifying possible adaptation states.

Enumerator

Important	Enables only the most important messages.
Normal	Enables the standard messages.
Verbose	Enables all messages.

7 Class Documentation

7.1 LibTopoART.CategoryInfo Struct Reference

Struct `CategoryInfo` summarises information about a node's category.

Public Member Functions

- [CategoryInfo](#) (decimal[] spatialWeights, decimal[]? temporalWeights, long clusterID, long classID)

This constructor sets the instance variables `spatial_weights`, `temporal_weights`, `clusterID`, and `classID` of struct [CategoryInfo](#).

Public Attributes

- readonly decimal[] **spatial_weights**

Instance variable `spatial_weights` represents the spatial weights of the considered node.

- readonly? decimal[] **temporal_weights**

Instance variable `temporal_weights` represents the temporal weights of the considered node if it supports temporal learning.

- readonly long **clusterID**

Instance variable `clusterID` represents the cluster ID of the considered node.

- readonly long **classID**

Instance variable `classID` represents the class ID of the considered node. If class IDs are not supported by the respective node, this value is set to [LibTopoART_info.UNDEFINED](#).

7.1.1 Detailed Description

Struct [CategoryInfo](#) summarises information about a node's category.

7.1.2 Constructor & Destructor Documentation

CategoryInfo()

```
LibTopoART.CategoryInfo.CategoryInfo (
    decimal[] spatialWeights,
    decimal?[] temporalWeights,
    long clusterID,
    long classID)
```

This constructor sets the instance variables `spatial_weights`, `temporal_weights`, `clusterID`, and `classID` of struct [CategoryInfo](#).

Parameters

<i>spatialWeights</i>	The spatial weights to be set.
<i>temporalWeights</i>	The temporal weights to be set.
<i>clusterID</i>	The cluster ID to be set.
<i>classID</i>	The class ID to be set.

7.2 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 Attributes

- decimal **bm_node_activation** = [LibTopoART_info.UNDEFINED](#)
Instance variable `bm_node_activation` represents the activation of the best-matching node (prediction variant).
- long **bm_node_ID** = [LibTopoART_info.UNDEFINED](#)
Instance variable `bm_node_ID` represents the ID of the best-matching node.
- long **bm_cluster_ID** = [LibTopoART_info.UNDEFINED](#)
Instance variable `bm_cluster_ID` represents the cluster ID of the best-matching node.
- decimal **bm_permanent_node_activation** = [LibTopoART_info.UNDEFINED](#)
Instance variable `bm_permanent_node_activation` represents the activation of the best-matching permanent node (prediction variant).
- long **bm_permanent_node_ID** = [LibTopoART_info.UNDEFINED](#)
Instance variable `bm_permanent_node_ID` represents the ID of the best-matching permanent node.
- long **bm_permanent_cluster_ID** = [LibTopoART_info.UNDEFINED](#)
Instance variable `bm_permanent_cluster_ID` represents the cluster ID of the best-matching permanent node.

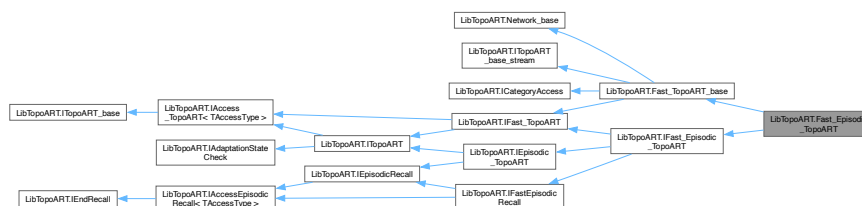
7.2.1 Detailed Description

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

7.3 LibTopoART.Fast Episodic TopoART Class Reference

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

Inheritance diagram for LibTopoART.Fast Episodic TopoART:



Public Member Functions

- `Fast_Episodic_TopoART` (long inputLen, long moduleNum, decimal rho_a, long t_max)
This constructor initialises an Episodic `TopoART` network.
- `Fast_Episodic_TopoART` (string path)
This constructor loads a saved Episodic `TopoART` network.
- `Fast_Episodic_TopoART` (Stream stream)
This constructor loads a saved Episodic `TopoART` network from a stream. The stream is left open.
- override void `Learn` (byte[] input)
This method performs a single training step.

- override void **Learn** (decimal[] input)
This method performs a single training step.
- long **BeginRecall** (byte[] stimulus)
This method starts the recall process.
- long **BeginRecall** (decimal[] stimulus)
This method starts the recall process.
- bool **InterEpisodeRecallStep** (out byte[]? recallResult, out decimal F3_activation)
This method performs a single inter-episode recall step and sets the starting point for intra-episode recall.
- bool **InterEpisodeRecallStep** (out decimal[]? recallResult, 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 byte[]? recallResult)
This method performs a single intra-episode recall step.
- bool **IntraEpisodeRecallStep** (out decimal[]? recallResult)
This method performs a single intra-episode recall step.
- void **EndRecall** ()
This method stops the recall process and frees temporary resources.

Public Member Functions inherited from **LibTopoART.Fast_TopoART_base**

- void **Learn** (byte[] input)
This method performs a single training step.
- 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** (byte[] input)
This method finds the closest category for a given test input.
- **F2_output[] GetBMOutput** (byte[] input, bool[]? mask)
This method finds the closest category for a given test input.
- **F2_output[] GetBMOutput** (decimal[] input)
This method finds the closest category for a given test input.
- **F2_output[] GetBMOutput** (decimal[] input, bool[]? mask)
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 **SaveText** (TextWriter writer)
This method saves the entire network as text to a writer. The writer is flushed but left open.
- void **Save** (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void **Save** (Stream stream, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network to a stream using the binary file format. The stream is left open.
- void **Save** (string path, bool compatibilityMode, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void **Save** (Stream stream, bool compatibilityMode, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network to a stream using the binary file format. The stream is left open.
- 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.
- List< **CategoryInfo** >? **GetCategories** (long moduleIndex=FINAL_MODULE)
This method collects information on the categories of a specified module.

Public Member Functions inherited from [LibTopoART.IAccess_TopoART< TAccessType >](#)

- [F2_output\[\] GetBMOutput](#) (TAccessType[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (TAccessType[] input, bool[] mask)
This method finds the closest category for a given test input.
- void [Learn](#) (TAccessType[] input)
This method performs a single training step.

Public Member Functions inherited from [LibTopoART.IAccessEpisodicRecall< TAccessType >](#)

- long [BeginRecall](#) (TAccessType[] stimulus)
This method starts the recall process.
- bool [InterEpisodeRecallStep](#) (out TAccessType[]? recallResult, 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 TAccessType[]? recallResult)
This method performs a single intra-episode recall step.

Properties

- new decimal **FileFormatVersion** [get]
Property FileFormatVersion returns the version of the file format used by class Episodic_TopoART.
- long **T_max** [get]
Property T_max represents the maximum considered time frame.

Properties inherited from [LibTopoART.Fast_TopoART_base](#)

- decimal **Alpha** [get, set]
Property Alpha represents the choice parameter alpha.
- decimal **Beta_sbm** [get, set]
Property Beta_sbm represents the learning rate of the second best-matching nodes.
- long[] **ClusterNum** [get]
Property ClusterNum represents the number of [TopoART](#) clusters found by each module.
- long[] **NodeNum** [get]
Property NodeNum represents the number of [TopoART](#) nodes used by each module.
- decimal **Rho_a** [get]
Property Rho_a represents the vigilance parameter of the first [TopoART](#) module (TA a).
- string **IntegerType** = Common.Types[(int)integerType] [get]
Property IntegerType returns a string containing the data type used for representing integer variables (IDs, parameters, counters, etc.) internally.
- decimal **FileFormatVersion** [get]
Property FileFormatVersion returns the version of the file format used by class [Fast_TopoART_base](#).
- string **FloatType** = Common.Types[(int)floatType] [get]
Property FloatType returns a string containing the data type used for representing floating point variables (input, weights, etc.) internally.
- decimal **TopoARTFileFormatVersion** [get]
Property TopoARTFileFormatVersion returns the version of the file format used by class [Fast_TopoART_base](#).

Properties inherited from [LibTopoART.Network_base](#)

- long **InputLen** [get]
Property `InputLen` returns the length of the input vector.
- long **LearningSteps** [get]
Property `LearningSteps` represents the total number of performed learning steps.
- long **ModuleNum** [get]
- long **Phi** [get, set]
- long[] **Phis** [get, set]
- long **Tau** [get, set]
Property `Tau` represents the parameter tau required for the removal of nodes and edges.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]
Property `InputLen` returns the length of the input vector.
- long **ModuleNum** [get]
- long **LearningSteps** [get]
Property `LearningSteps` represents the total number of performed learning steps.
- long **Tau** [get, set]
Property `Tau` represents the parameter tau required for the removal of nodes and edges.
- long **Phi** [get, set]
- long[] **Phis** [get, set]

Additional Inherited Members**Static Public Attributes inherited from [LibTopoART.Network_base](#)**

- const long **FINAL_MODULE** = [LibTopoART_info.FINAL_MODULE](#)
Instance variable `FINAL_MODULE` gives the value used for indicating that the [TopoART](#) module with the highest index is to be used.

7.3.1 Detailed Description

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

7.3.2 Constructor & Destructor Documentation

Fast_Episodic_TopoART() [1/3]

```
LibTopoART.Fast_Episodic_TopoART.Fast_Episodic_TopoART (
    long inputLen,
    long moduleNum,
    decimal rho_a,
    long t_max)
```

This constructor initialises an Episodic [TopoART](#) network.

Parameters

<i>inputLen</i>	The length of input vectors to be learnt.
<i>moduleNum</i>	The number of Episodic TopoART modules.
<i>rho_a</i>	The vigilance parameter of the first Episodic TopoART module (ETA a).
<i>t_max</i>	The parameter limiting the considered time frame.

Fast_Episodic_TopoART() [2/3]

```
LibTopoART.Fast_Episodic_TopoART.Fast_Episodic_TopoART (
    string path)
```

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

Parameters

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

Exceptions

InvalidFileException	Thrown when the given file cannot be loaded.
--------------------------------------	--

Fast_Episodic_TopoART() [3/3]

```
LibTopoART.Fast_Episodic_TopoART.Fast_Episodic_TopoART (
    Stream stream)
```

This constructor loads a saved Episodic [TopoART](#) network from a stream. The stream is left open.

Parameters

<i>stream</i>	A readable <code>Stream</code> containing a network in the binary Episodic TopoART file format.
---------------	---

Exceptions

<i>InvalidFileException</i>	Thrown when the given stream cannot be loaded.
---	--

7.3.3 Member Function Documentation**BeginRecall()** [1/2]

```
long LibTopoART.Fast_Episodic_TopoART.BeginRecall (
    byte[ ] stimulus)
```

This method starts the recall process.

Parameters

<i>stimulus</i>	The stimulus (input) which is used to trigger recall. The stimulus elements are internally scaled from [0, 255] to [0, 1].
-----------------	--

Returns

The number of F3 nodes created.

BeginRecall() [2/2]

```
long LibTopoART.Fast_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.

InterEpisodeRecallStep() [1/2]

```
bool LibTopoART.Fast_Episodic_TopoART.InterEpisodeRecallStep (
    out byte?[] recallResult,
    out decimal F3_activation)
```

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

Parameters

<i>recallResult</i>	Returns the recall output for the current step. The elements of the recall result are internally scaled from [0, 1] to [0, 255].
<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.

InterEpisodeRecallStep() [2/2]

```
bool LibTopoART.Fast_Episodic_TopoART.InterEpisodeRecallStep (
    out decimal?[] recallResult,
    out decimal F3_activation)
```

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

Parameters

<i>recallResult</i>	Returns the recall output 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.

IntraEpisodeRecallStep() [1/2]

```
bool LibTopoART.Fast_Episodic_TopoART.IntraEpisodeRecallStep (
    out byte?[] recallResult)
```

This method performs a single intra-episode recall step.

Parameters

<i>recallResult</i>	Returns the recall output for the current step. The elements of the recall result are internally scaled from [0, 1] to [0, 255].
---------------------	--

Returns

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

IntraEpisodeRecallStep() [2/2]

```
bool LibTopoART.Fast_Episodic_TopoART.IntraEpisodeRecallStep (
    out decimal?[] recallResult)
```

This method performs a single intra-episode recall step.

Parameters

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

Returns

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

Learn() [1/2]

```
override void LibTopoART.Fast_Episodic_TopoART.Learn (
    byte[] input)
```

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: $w_{\{j,2\}^{F2,t}}(t+1) = \text{beta_j} * \text{Max}(t_2^{F1}(t), w_{\{j,2\}^{F2,t}}(t) + (1 - \text{beta_j}) * w_{\{j,2\}^{F2,t}}(t))$ for $j = \text{bm}$ or $j = \text{sbm}$. (Note: $w_{\{j,1\}^{F2,t}}$ remains constant over the lifetime of a node.)

Parameters

<i>input</i>	The input vector to be learnt. The input values are internally scaled from [0, 255] to [0, 1].
--------------	--

Learn() [2/2]

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

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: $w_{\{j,2\}^{F2,t}}(t+1) = \text{beta_j} * \text{Max}(t_2^{F1}(t), w_{\{j,2\}^{F2,t}}(t) + (1 - \text{beta_j}) * w_{\{j,2\}^{F2,t}}(t))$ for $j = \text{bm}$ or $j = \text{sbm}$. (Note: $w_{\{j,1\}^{F2,t}}$ remains constant over the lifetime of a node.)

Parameters

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

- *This method saves the entire network as a binary file.*
- void **Save** (Stream stream, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network to a stream using the binary file format. The stream is left open.
- void **Save** (string path, bool compatibilityMode, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void **Save** (Stream stream, bool compatibilityMode, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network to a stream using the binary file format. The stream is left open.
- 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.
- List< **CategoryInfo** >? **GetCategories** (long moduleIndex=FINAL_MODULE)
This method collects information on the categories of a specified module.

Public Member Functions inherited from **LibTopoART.IAccess_TopoART< TAccessType >**

- **F2_output[] GetBMOutput** (TAccessType[] input)
This method finds the closest category for a given test input.
- **F2_output[] GetBMOutput** (TAccessType[] input, bool[] mask)
This method finds the closest category for a given test input.
- void **Learn** (TAccessType[] input)
This method performs a single training step.

Additional Inherited Members

Static Public Attributes inherited from **LibTopoART.Network_base**

- const long **FINAL_MODULE** = **LibTopoART_info.FINAL_MODULE**
Instance variable `FINAL_MODULE` gives the value used for indicating that the `TopoART` module with the highest index is to be used.

Properties inherited from **LibTopoART.Fast_TopoART_base**

- decimal **Alpha** [get, set]
Property `Alpha` represents the choice parameter alpha.
- decimal **Beta_sbm** [get, set]
Property `Beta_sbm` represents the learning rate of the second best-matching nodes.
- long[] **ClusterNum** [get]
Property `ClusterNum` represents the number of `TopoART` clusters found by each module.
- long[] **NodeNum** [get]
Property `NodeNum` represents the number of `TopoART` nodes used by each module.
- decimal **Rho_a** [get]
Property `Rho_a` represents the vigilance parameter of the first `TopoART` module (TA a).
- string **IntegerType** = Common.Types[(int)integerType] [get]
Property `IntegerType` returns a string containing the data type used for representing integer variables (IDs, parameters, counters, etc.) internally.
- decimal **FileFormatVersion** [get]
Property `FileFormatVersion` returns the version of the file format used by class `Fast_TopoART_base`.
- string **FloatType** = Common.Types[(int)floatType] [get]
Property `FloatType` returns a string containing the data type used for representing floating point variables (input, weights, etc.) internally.
- decimal **TopoARTFileFormatVersion** [get]
Property `TopoARTFileFormatVersion` returns the version of the file format used by class `Fast_TopoART_base`.

Properties inherited from [LibTopoART.Network_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- long **LearningSteps** [get]
Property LearningSteps represents the total number of performed learning steps.
- long **ModuleNum** [get]
- long **Phi** [get, set]
- long[] **Phis** [get, set]
- long **Tau** [get, set]
Property Tau represents the parameter tau required for the removal of nodes and edges.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- long **ModuleNum** [get]
- long **LearningSteps** [get]
Property LearningSteps represents the total number of performed learning steps.
- long **Tau** [get, set]
Property Tau represents the parameter tau required for the removal of nodes and edges.
- long **Phi** [get, set]
- long[] **Phis** [get, set]

7.4.1 Detailed Description

Class [Fast_TopoART](#) provides an implementation of the [TopoART](#) neural network as proposed in "Marko Tscherepanow (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 mapped to `int` 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].

7.4.2 Constructor & Destructor Documentation**Fast_TopoART() [1/3]**

```
LibTopoART.Fast_TopoART.Fast_TopoART (
    long inputLen,
    long moduleNum,
    decimal rho_a)
```

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

Parameters

<i>inputLen</i>	The length of input vectors to be learnt.
-----------------	---

<i>moduleNum</i>	The number of TopoART modules.
<i>rho_a</i>	The vigilance parameter of the first TopoART module (TA a).

Fast_TopoART() [2/3]

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

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

Parameters

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

Exceptions

InvalidFileException	Thrown when the given file cannot be loaded.
--------------------------------------	--

Fast_TopoART() [3/3]

```
LibTopoART.Fast_TopoART.Fast_TopoART (  
    Stream stream)
```

This constructor loads a saved [TopoART](#) network from a stream. The stream is left open.

Parameters

<i>stream</i>	A readable <code>Stream</code> containing a network in the binary TopoART file format.
---------------	--

Exceptions

InvalidFileException	Thrown when the given stream cannot be loaded.
--------------------------------------	--

7.4.3 Member Function Documentation

Learn() [1/2]

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

This method performs a single training step.

Parameters

<i>input</i>	The input vector to be learnt. The input values are internally scaled from [0, 255] to [0, 1].
--------------	--

Learn() [2/2]

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

This method performs a single training step.

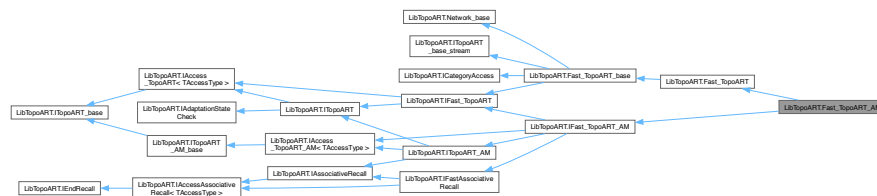
Parameters

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

7.5 LibTopoART.Fast_TopoART_AM Class Reference

Class [Fast_TopoART_AM](#) provides an implementation of the TopoART-AM neural network as proposed in "Marko Tscherepanow, Marco Kortkamp and Marc Kammer (2011). A Hierarchical ART Network for the Stable Incremental Learning of Topological Structures and Associations from Noisy Data. Neural Networks 24(8): 906-916. Elsevier."

Inheritance diagram for LibTopoART.Fast_TopoART_AM:

**Public Member Functions**

- [Fast_TopoART_AM](#) (long key1Len, long key2Len, long moduleNum, decimal rho_a)
This constructor initialises a TopoART-AM network.
- [Fast_TopoART_AM](#) (string path)
This constructor loads a saved TopoART-AM network.
- [Fast_TopoART_AM](#) (Stream stream)
This constructor loads a saved TopoART-AM network from a stream. The stream is left open.
- [F2_output\[\] GetBMOutput](#) (byte[] key1, byte[] key2)
This method finds the closest category for a given pair of keys.
- [F2_output\[\] GetBMOutput](#) (decimal[] key1, decimal[] key2)
This method finds the closest category for a given pair of keys.
- void [Learn](#) (byte[] key1, byte[] key2)
This method performs a single training step.
- void [Learn](#) (decimal[] key1, decimal[] key2)
This method performs a single training step.
- long [BeginRecallKey1](#) (byte[] key2, long moduleIndex=FINAL_MODULE)

- This method starts the recall process for the first key vector.*

 - long **BeginRecallKey1** (decimal[] key2, long moduleIndex=FINAL_MODULE)

This method starts the recall process for the first key vector.
- long **BeginRecallKey2** (byte[] key1, long moduleIndex=FINAL_MODULE)

This method starts the recall process for the second key vector.
- long **BeginRecallKey2** (decimal[] key1, long moduleIndex=FINAL_MODULE)

This method starts the recall process for the second key vector.
- bool **RecallStep** (out byte[]? recallResult, out decimal F3_activation)

This method performs a single associative recall step.
- bool **RecallStep** (out decimal[]? recallResult, out decimal F3_activation)

This method performs a single associative recall step.
- void **EndRecall** ()

This method stops the recall process and frees temporary resources.

Public Member Functions inherited from **LibTopoART.Fast_TopoART**

- **Fast_TopoART** (long inputLen, long moduleNum, decimal rho_a)

*This constructor initialises a **TopoART** network.*
- **Fast_TopoART** (string path)

*This constructor loads a saved **TopoART** network.*
- **Fast_TopoART** (Stream stream)

*This constructor loads a saved **TopoART** network from a stream. The stream is left open.*
- override void **Learn** (byte[] input)

This method performs a single training step.
- override void **Learn** (decimal[] input)

This method performs a single training step.

Public Member Functions inherited from **LibTopoART.Fast_TopoART_base**

- void **Learn** (byte[] input)

This method performs a single training step.
- 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** (byte[] input)

This method finds the closest category for a given test input.
- **F2_output[] GetBMOutput** (byte[] input, bool[]? mask)

This method finds the closest category for a given test input.
- **F2_output[] GetBMOutput** (decimal[] input)

This method finds the closest category for a given test input.
- **F2_output[] GetBMOutput** (decimal[] input, bool[]? mask)

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 **SaveText** (TextWriter writer)

This method saves the entire network as text to a writer. The writer is flushed but left open.
- void **Save** (string path, CompressionLevel compression=CompressionLevel.Fastest)

- This method saves the entire network as a binary file.*
- void [Save](#) (Stream stream, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network to a stream using the binary file format. The stream is left open.
- void [Save](#) (string path, bool compatibilityMode, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void [Save](#) (Stream stream, bool compatibilityMode, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network to a stream using the binary file format. The stream is left open.
- 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.
- List< [CategoryInfo](#) >? [GetCategories](#) (long moduleIndex=FINAL_MODULE)
This method collects information on the categories of a specified module.

Public Member Functions inherited from [LibTopoART.IAccess_TopoART< TAccessType >](#)

- [F2_output\[\]](#) [GetBMOutput](#) (TAccessType[] input)
This method finds the closest category for a given test input.
- [F2_output\[\]](#) [GetBMOutput](#) (TAccessType[] input, bool[] mask)
This method finds the closest category for a given test input.
- void [Learn](#) (TAccessType[] input)
This method performs a single training step.

Public Member Functions inherited from [LibTopoART.IAccess_TopoART_AM< TAccessType >](#)

- [F2_output\[\]](#) [GetBMOutput](#) (TAccessType[] key1, TAccessType[] key2)
This method finds the closest category for a given pair of keys.
- void [Learn](#) (TAccessType[] key1, TAccessType[] key2)
This method performs a single training step.

Public Member Functions inherited from [LibTopoART.IAccessAssociativeRecall< TAccessType >](#)

- long [BeginRecallKey1](#) (TAccessType[] key2, long moduleIndex=[LibTopoART_info.FINAL_MODULE](#))
This method starts the recall process for the first key vector.
- long [BeginRecallKey2](#) (TAccessType[] key1, long moduleIndex=[LibTopoART_info.FINAL_MODULE](#))
This method starts the recall process for the second key vector.
- bool [RecallStep](#) (out TAccessType[]? recallResult, out decimal F3_activation)
This method performs a single associative recall step.

Properties

- new decimal **FileFormatVersion** [get]
Property [FileFormatVersion](#) returns the version of the file format used by class [Fast_TopoART_AM](#).
- long **Key1Len** [get]
Property [Key1Len](#) returns the length of the first key vector.
- long **Key2Len** [get]
Property [Key2Len](#) returns the length of the second key vector.

Properties inherited from [LibTopoART.Fast_TopoART_base](#)

- decimal **Alpha** [get, set]
Property Alpha represents the choice parameter alpha.
- decimal **Beta_sbm** [get, set]
Property Beta_sbm represents the learning rate of the second best-matching nodes.
- long[] **ClusterNum** [get]
Property ClusterNum represents the number of [TopoART](#) clusters found by each module.
- long[] **NodeNum** [get]
Property NodeNum represents the number of [TopoART](#) nodes used by each module.
- decimal **Rho_a** [get]
Property Rho_a represents the vigilance parameter of the first [TopoART](#) module (TA a).
- string **IntegerType** = Common.Types[(int)integerType] [get]
Property IntegerType returns a string containing the data type used for representing integer variables (IDs, parameters, counters, etc.) internally.
- decimal **FileFormatVersion** [get]
Property FileFormatVersion returns the version of the file format used by class [Fast_TopoART_base](#).
- string **FloatType** = Common.Types[(int)floatType] [get]
Property FloatType returns a string containing the data type used for representing floating point variables (input, weights, etc.) internally.
- decimal **TopoARTFileFormatVersion** [get]
Property TopoARTFileFormatVersion returns the version of the file format used by class [Fast_TopoART_base](#).

Properties inherited from [LibTopoART.Network_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- long **LearningSteps** [get]
Property LearningSteps represents the total number of performed learning steps.
- long **ModuleNum** [get]
- long **Phi** [get, set]
- long[] **Phis** [get, set]
- long **Tau** [get, set]
Property Tau represents the parameter tau required for the removal of nodes and edges.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- long **ModuleNum** [get]
- long **LearningSteps** [get]
Property LearningSteps represents the total number of performed learning steps.
- long **Tau** [get, set]
Property Tau represents the parameter tau required for the removal of nodes and edges.
- long **Phi** [get, set]
- long[] **Phis** [get, set]

Additional Inherited Members

Static Public Attributes inherited from LibTopoART.Network_base

- const long **FINAL_MODULE** = LibTopoART_info.FINAL_MODULE

Instance variable *FINAL_MODULE* gives the value used for indicating that the *TopoART* module with the highest index is to be used.

7.5.1 Detailed Description

Class *Fast_TopoART_AM* provides an implementation of the TopoART-AM neural network as proposed in "Marko Tscherepanow, Marco Kortkamp and Marc Kammer (2011). A Hierarchical ART Network for the Stable Incremental Learning of Topological Structures and Associations from Noisy Data. Neural Networks 24(8): 906-916. Elsevier."

Class *Fast_TopoART_AM* requires all input and output to lie in the interval [0, 1].

7.5.2 Constructor & Destructor Documentation

Fast_TopoART_AM() [1/3]

```
LibTopoART.Fast_TopoART_AM.Fast_TopoART_AM (
    long key1Len,
    long key2Len,
    long moduleNum,
    decimal rho_a)
```

This constructor initialises a TopoART-AM network.

Parameters

<i>key1Len</i>	The length of the first key vector to be learnt.
<i>key2Len</i>	The length of the second key vector to be learnt.
<i>moduleNum</i>	The number of TopoART-AM modules.
<i>rho_a</i>	The vigilance parameter of the first TopoART-AM module (TopoART-AM a).

Fast_TopoART_AM() [2/3]

```
LibTopoART.Fast_TopoART_AM.Fast_TopoART_AM (
    string path)
```

This constructor loads a saved TopoART-AM network.

Parameters

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

Exceptions

<i>InvalidFileException</i>	Thrown when the given file cannot be loaded.
---	--

Fast_TopoART_AM() [3/3]

```
LibTopoART.Fast_TopoART_AM.Fast_TopoART_AM (
    Stream stream)
```

This constructor loads a saved TopoART-AM network from a stream. The stream is left open.

Parameters

<i>stream</i>	A readable <code>Stream</code> containing a network in the binary TopoART-AM file format.
---------------	---

Exceptions

<i>InvalidFileException</i>	Thrown when the given stream cannot be loaded.
---	--

7.5.3 Member Function Documentation**BeginRecallKey1() [1/2]**

```
long LibTopoART.Fast_TopoART_AM.BeginRecallKey1 (
    byte[] key2,
    long moduleIndex = FINAL_MODULE)
```

This method starts the recall process for the first key vector.

Parameters

<i>key2</i>	The stimulus (second key vector) which is used to trigger recall. The stimulus elements are internally scaled from [0, 255] to [0, 1].
<i>moduleIndex</i>	Index of the TopoART-AM module to be used for recall. (<code>FINAL_MODULE</code> denotes the module with the highest index.)

Returns

The number of F3 nodes created.

Exceptions

<i>InvalidModuleIndexException</i>	Thrown when <i>moduleIndex</i> is invalid.
--	--

BeginRecallKey1() [2/2]

```
long LibTopoART.Fast_TopoART_AM.BeginRecallKey1 (
    decimal[] key2,
    long moduleIndex = FINAL_MODULE)
```

This method starts the recall process for the first key vector.

Parameters

<i>key2</i>	The stimulus (second key vector) which is used to trigger recall.
<i>moduleIndex</i>	Index of the TopoART-AM module to be used for recall. (FINAL_MODULE denotes the module with the highest index.)

Returns

The number of F3 nodes created.

Exceptions

InvalidModuleIndexException	Thrown when <i>moduleIndex</i> is invalid.
---	--

BeginRecallKey2() [1/2]

```
long LibTopoART.Fast_TopoART_AM.BeginRecallKey2 (
    byte[] key1,
    long moduleIndex = FINAL_MODULE)
```

This method starts the recall process for the second key vector.

Parameters

<i>key1</i>	The stimulus (first key vector) which is used to trigger recall. The stimulus elements are internally scaled from [0, 255] to [0, 1].
<i>moduleIndex</i>	Index of the TopoART-AM module to be used for recall. (FINAL_MODULE denotes the module with the highest index.)

Returns

The number of F3 nodes created.

Exceptions

InvalidModuleIndexException	Thrown when <i>moduleIndex</i> is invalid.
---	--

BeginRecallKey2() [2/2]

```
long LibTopoART.Fast_TopoART_AM.BeginRecallKey2 (
    decimal[] key1,
    long moduleIndex = FINAL_MODULE)
```

This method starts the recall process for the second key vector.

Parameters

<i>key1</i>	The stimulus (first key vector) which is used to trigger recall.
<i>moduleIndex</i>	Index of the TopoART-AM module to be used for recall. (<code>FINAL_MODULE</code> denotes the module with the highest index.)

Returns

The number of F3 nodes created.

Exceptions

InvalidModuleIndexException	Thrown when <i>moduleIndex</i> is invalid.
---	--

GetBMOutput() [1/2]

```
F2_output[] LibTopoART.Fast_TopoART_AM.GetBMOutput (
    byte[] key1,
    byte[] key2)
```

This method finds the closest category for a given pair of keys.

Parameters

<i>key1</i>	The first key vector. The elements of the key vector are internally scaled from [0, 255] to [0, 1].
<i>key2</i>	The second key vector corresponding to <i>key1</i> . The elements of the key vector are internally scaled from [0, 255] to [0, 1].

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-AM module.

GetBMOutput() [2/2]

```
F2_output [ ] LibTopoART.Fast_TopoART_AM.GetBMOutput (
    decimal [ ] key1,
    decimal [ ] key2)
```

This method finds the closest category for a given pair of keys.

Parameters

<i>key1</i>	The first key vector.
<i>key2</i>	The second key vector corresponding to <i>key1</i> .

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-AM module.

Learn() [1/2]

```
void LibTopoART.Fast_TopoART_AM.Learn (
    byte [ ] key1,
    byte [ ] key2)
```

This method performs a single training step.

Parameters

<i>key1</i>	The first key vector to be learnt.
<i>key2</i>	The second key vector corresponding to <i>key1</i> .

Learn() [2/2]

```
void LibTopoART.Fast_TopoART_AM.Learn (
    decimal [ ] key1,
    decimal [ ] key2)
```

This method performs a single training step.

Parameters

<i>key1</i>	The first key vector to be learnt. The elements of the key vector are internally scaled from [0, 255] to [0, 1].
<i>key2</i>	The second key vector corresponding to <i>key1</i> . The elements of the key vector are internally scaled from [0, 255] to [0, 1].

This method performs a single associative recall step.

<i>recallResult</i>	Returns the recall output for the current step. The elements of the recall result are internally scaled from [0, 1] to [0, 255].
<i>F3_activation</i>	Returns the activation of the current F3 node.

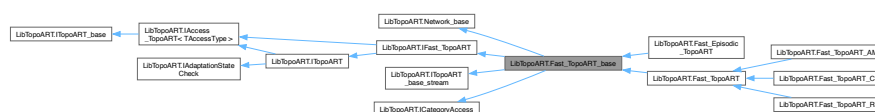
A boolean result indicating whether the recall step was successfully completed or not.

This method performs a single associative recall step.

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

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

Inheritance diagram for LibTopoART.Fast_TopoART_base:



Public Member Functions

- void [Learn](#) (byte[] input)
This method performs a single training step.
- 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](#) (byte[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (byte[] input, bool[]? mask)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (decimal[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (decimal[] input, bool[]? mask)
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 [SaveText](#) (TextWriter writer)
This method saves the entire network as text to a writer. The writer is flushed but left open.
- void [Save](#) (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void [Save](#) (Stream stream, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network to a stream using the binary file format. The stream is left open.
- void [Save](#) (string path, bool compatibilityMode, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void [Save](#) (Stream stream, bool compatibilityMode, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network to a stream using the binary file format. The stream is left open.
- 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.
- List< [CategoryInfo](#) >? [GetCategories](#) (long moduleIndex=FINAL_MODULE)
This method collects information on the categories of a specified module.

Public Member Functions inherited from [LibTopoART.IAccess_TopoART< TAccessType >](#)

- [F2_output\[\] GetBMOutput](#) (TAccessType[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (TAccessType[] input, bool[] mask)
This method finds the closest category for a given test input.
- void [Learn](#) (TAccessType[] input)
This method performs a single training step.

Properties

- decimal **Alpha** [get, set]
Property Alpha represents the choice parameter alpha.
- decimal **Beta_sbm** [get, set]
Property Beta_sbm represents the learning rate of the second best-matching nodes.
- long[] **ClusterNum** [get]
Property ClusterNum represents the number of [TopoART](#) clusters found by each module.
- long[] **NodeNum** [get]
Property NodeNum represents the number of [TopoART](#) nodes used by each module.
- decimal **Rho_a** [get]
Property Rho_a represents the vigilance parameter of the first [TopoART](#) module (TA a).
- string **IntegerType** = Common.Types[(int)integerType] [get]
Property IntegerType returns a string containing the data type used for representing integer variables (IDs, parameters, counters, etc.) internally.
- decimal **FileFormatVersion** [get]
Property FileFormatVersion returns the version of the file format used by class [Fast_TopoART_base](#).
- string **FloatType** = Common.Types[(int)floatType] [get]
Property FloatType returns a string containing the data type used for representing floating point variables (input, weights, etc.) internally.
- decimal **TopoARTFileFormatVersion** [get]
Property TopoARTFileFormatVersion returns the version of the file format used by class [Fast_TopoART_base](#).

Properties inherited from [LibTopoART.Network_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- long **LearningSteps** [get]
Property LearningSteps represents the total number of performed learning steps.
- long **ModuleNum** [get]
- long **Phi** [get, set]
- long[] **Phis** [get, set]
- long **Tau** [get, set]
Property Tau represents the parameter tau required for the removal of nodes and edges.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- long **ModuleNum** [get]
- long **LearningSteps** [get]
Property LearningSteps represents the total number of performed learning steps.
- long **Tau** [get, set]
Property Tau represents the parameter tau required for the removal of nodes and edges.
- long **Phi** [get, set]
- long[] **Phis** [get, set]

Additional Inherited Members

Static Public Attributes inherited from [LibTopoART.Network_base](#)

- const long **FINAL_MODULE** = [LibTopoART_info.FINAL_MODULE](#)
Instance variable `FINAL_MODULE` gives the value used for indicating that the [TopoART](#) module with the highest index is to be used.

7.6.1 Detailed Description

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

7.6.2 Member Function Documentation

Dispose()

```
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.

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.

Exceptions

InvalidStateException	Thrown when the network is in an invalid state.
InvalidNumberException	Thrown when the number of edges of an F2 node is greater than <code>int.MaxValue</code> .

Implements [LibTopoART.IAdaptationStateCheck](#).

GetBMOutput() [1/4]

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

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

Parameters

<i>input</i>	The input vector $x(t)$. The input values are internally scaled from [0, 255] to [0, 1].
--------------	---

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.

GetBMOutput() [2/4]

```
F2_output [ ] LibTopoART.Fast_TopoART_base.GetBMOutput (
    byte[] input,
    bool?[] mask)
```

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

Parameters

<i>input</i>	The input vector $x(t)$. The input values are internally scaled from [0, 255] to [0, 1].
<i>mask</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.

GetBMOutput() [3/4]

```
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.

GetBMOutput() [4/4]

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

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

Parameters

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

GetCategories()

```
List< CategoryInfo >? LibTopoART.Fast_TopoART_base.GetCategories (
    long moduleIndex = FINAL_MODULE)
```

This method collects information on the categories of a specified module.

Parameters

<i>moduleIndex</i>	The index of the module the categories of which are to be analysed.
--------------------	---

Returns

A list containing information about the respective categories.

Exceptions

InvalidModuleIndexException	Thrown when <i>moduleIndex</i> is invalid.
InvalidNumberException	Thrown when the number of nodes of a module is greater than <code>int.MaxValue</code> .

Implements [LibTopoART.ICategoryAccess](#).

Learn() [1/2]

```
void LibTopoART.Fast_TopoART_base.Learn (
    byte[] input) [abstract]
```

This method performs a single training step.

Parameters

<i>input</i>	The input vector to be learnt. The input values are internally scaled from [0, 255] to [0, 1].
--------------	--

Learn() [2/2]

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

This method performs a single training step.

Parameters

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

ResetAdaptationState()

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

This method resets the adaptation state to [AdaptationState.NO_ADAPTATION](#).

Exceptions

InvalidNumberException	Thrown when the number of edges of an F2 node is greater than <code>int.MaxValue</code> .
--	---

Implements [LibTopoART.IAdaptationStateCheck](#).

Save() [1/4]

```
void LibTopoART.Fast_TopoART_base.Save (
    Stream stream,
    bool compatibilityMode,
    CompressionLevel compression = CompressionLevel::Fastest)
```

This method saves the entire network to a stream using the binary file format. The stream is left open.

Parameters

<i>stream</i>	A writable <code>Stream</code> the network is saved to.
<i>compatibilityMode</i>	If true, the network is saved in compatibility mode.

<i>compression</i>	Compression level of the saved data (Compression is not supported by LibTopoART v0.93 and below.)
--------------------	---

Save() [2/4]

```
void LibTopoART.Fast_TopoART_base.Save (
    Stream stream,
    CompressionLevel compression = CompressionLevel::Fastest)
```

This method saves the entire network to a stream using the binary file format. The stream is left open.

Parameters

<i>stream</i>	A writable <code>Stream</code> the network is saved to.
<i>compression</i>	Compression level of the saved data (Compression is not supported by LibTopoART v0.93 and below.)

Implements [LibTopoART.ITopoART_base_stream](#).

Save() [3/4]

```
void LibTopoART.Fast_TopoART_base.Save (
    string path,
    bool compatibilityMode,
    CompressionLevel compression = CompressionLevel::Fastest)
```

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.
<i>compatibilityMode</i>	If true, the file is saved in compatibility mode.
<i>compression</i>	Compression level of the save file (Compression is not supported by LibTopoART v0.93 and below.)

Save() [4/4]

```
void LibTopoART.Fast_TopoART_base.Save (
    string path,
    CompressionLevel compression = CompressionLevel::Fastest)
```

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.
-------------	--

Public Member Functions

- [Fast_TopoART_C](#) (long inputLen, long moduleNum, decimal rho_a)
This constructor initialises a TopoART-C network.
- [Fast_TopoART_C](#) (string path)
This constructor loads a saved TopoART-C network.
- [Fast_TopoART_C](#) (Stream stream)
This constructor loads a saved TopoART-C network from a stream. The stream is left open.
- override void [Learn](#) (byte[] input)
This method performs a single training step and sets the class ID corresponding to input to UNDEFINED_CLASS↔_ID.
- 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](#) (byte[] input, long classID)
This method performs a single training step.
- void [Learn](#) (decimal[] input, long classID)
This method performs a single training step.
- long [Predict](#) (byte[] input)
This method predicts the class ID using the default value of nu.
- long [Predict](#) (decimal[] input)
This method predicts the class ID using the default value of nu.
- long [Predict](#) (byte[] input, long nu)
This method predicts the class ID using a custom value of nu.
- long [Predict](#) (decimal[] input, long nu)
This method predicts the class ID using a custom value of nu.
- [TopoART_C_prediction Predict](#) (byte[] input, bool[]? mask)
This method predicts the class ID using the default value of nu.
- [TopoART_C_prediction Predict](#) (decimal[] input, bool[]? mask)
This method predicts the class ID using the default value of nu.
- [TopoART_C_prediction Predict](#) (byte[] input, bool[]? mask, long nu)
This method predicts the class ID using a custom value of nu.
- [TopoART_C_prediction Predict](#) (decimal[] input, bool[]? mask, long nu)
This method predicts the class ID using a custom value of nu.

Public Member Functions inherited from [LibTopoART.Fast_TopoART](#)

- [Fast_TopoART](#) (long inputLen, long moduleNum, decimal rho_a)
This constructor initialises a TopoART network.
- [Fast_TopoART](#) (string path)
This constructor loads a saved TopoART network.
- [Fast_TopoART](#) (Stream stream)
This constructor loads a saved TopoART network from a stream. The stream is left open.
- override void [Learn](#) (byte[] input)
This method performs a single training step.
- override void [Learn](#) (decimal[] input)
This method performs a single training step.

Public Member Functions inherited from [LibTopoART.Fast_TopoART_base](#)

- void [Learn](#) (byte[] input)
This method performs a single training step.
- 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](#) (byte[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (byte[] input, bool[]? mask)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (decimal[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (decimal[] input, bool[]? mask)
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 [SaveText](#) (TextWriter writer)
This method saves the entire network as text to a writer. The writer is flushed but left open.
- void [Save](#) (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void [Save](#) (Stream stream, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network to a stream using the binary file format. The stream is left open.
- void [Save](#) (string path, bool compatibilityMode, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void [Save](#) (Stream stream, bool compatibilityMode, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network to a stream using the binary file format. The stream is left open.
- 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.
- List< [CategoryInfo](#) >? [GetCategories](#) (long moduleIndex=FINAL_MODULE)
This method collects information on the categories of a specified module.

Public Member Functions inherited from [LibTopoART.IAccess_TopoART](#) < [TAccessType](#) >

- [F2_output\[\] GetBMOutput](#) (TAccessType[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (TAccessType[] input, bool[] mask)
This method finds the closest category for a given test input.
- void [Learn](#) (TAccessType[] input)
This method performs a single training step.

Public Member Functions inherited from [LibTopoART.IAccess_TopoART_C< TAccessType >](#)

- void [Learn](#) (TAccessType[] input, long classID)
This method performs a single training step.
- long [Predict](#) (TAccessType[] input)
This method predicts the class ID using the default value of nu.
- long [Predict](#) (TAccessType[] input, long nu)
This method predicts the class ID using a custom value of nu.
- [TopoART_C_prediction Predict](#) (TAccessType[] input, bool[] mask)
This method predicts the class ID using the default value of nu.
- [TopoART_C_prediction Predict](#) (TAccessType[] input, bool[] mask, long nu)
This method predicts the class ID using a custom value of nu.

Static Public Attributes

- const long **UNDEFINED_CLASS_ID** = -2
Instance variable `UNDEFINED_CLASS_ID` gives the value used for indicating that an input sample was predicted to belong to the undefined class; i.e., no class ID was provided for such input samples during training.

Static Public Attributes inherited from [LibTopoART.Network_base](#)

- const long **FINAL_MODULE** = [LibTopoART_info.FINAL_MODULE](#)
Instance variable `FINAL_MODULE` gives the value used for indicating that the [TopoART](#) module with the highest index is to be used.

Properties

- new decimal **FileFormatVersion** [get]
Property `FileFormatVersion` returns the version of the file format used by class [Fast_TopoART_C](#).
- long **Nu** [get, set]
*Property `Nu` represents the default value used for the maximum cardinality of the set of enclosing categories *E* and the neighbourhood set *N* during prediction. If the parameter `nu` is not explicitly provided for prediction, this property will be applied. (This parameter does not modify the network. It may be arbitrarily changed for each prediction step.).*
- bool **SkipEdgeLearning** [get, set]
Property `SkipEdgeLearning` enables/disables the [TopoART](#) edge learning mechanism. If the topology of the input data is not required, disabling edge learning may decrease the processing time needed for training.

Properties inherited from [LibTopoART.Fast_TopoART_base](#)

- decimal **Alpha** [get, set]
*Property `Alpha` represents the choice parameter *alpha*.*
- decimal **Beta_sbm** [get, set]
Property `Beta_sbm` represents the learning rate of the second best-matching nodes.
- long[] **ClusterNum** [get]
Property `ClusterNum` represents the number of [TopoART](#) clusters found by each module.
- long[] **NodeNum** [get]
Property `NodeNum` represents the number of [TopoART](#) nodes used by each module.
- decimal **Rho_a** [get]
*Property `Rho_a` represents the vigilance parameter of the first [TopoART](#) module (*TA a*).*
- string **IntegerType** = Common.Types[(int)integerType] [get]

Property *IntegerType* returns a string containing the data type used for representing integer variables (IDs, parameters, counters, etc.) internally.

- decimal **FileFormatVersion** [get]

Property *FileFormatVersion* returns the version of the file format used by class [Fast_TopoART_base](#).

- string **FloatType** = Common.Types[(int)floatType] [get]

Property *FloatType* returns a string containing the data type used for representing floating point variables (input, weights, etc.) internally.

- decimal **TopoARTFileFormatVersion** [get]

Property *TopoARTFileFormatVersion* returns the version of the file format used by class [Fast_TopoART_base](#).

Properties inherited from [LibTopoART.Network_base](#)

- long **InputLen** [get]

Property *InputLen* returns the length of the input vector.

- long **LearningSteps** [get]

Property *LearningSteps* represents the total number of performed learning steps.

- long **ModuleNum** [get]

- long **Phi** [get, set]

- long[] **Phis** [get, set]

- long **Tau** [get, set]

Property *Tau* represents the parameter tau required for the removal of nodes and edges.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]

Property *InputLen* returns the length of the input vector.

- long **ModuleNum** [get]

- long **LearningSteps** [get]

Property *LearningSteps* represents the total number of performed learning steps.

- long **Tau** [get, set]

Property *Tau* represents the parameter tau required for the removal of nodes and edges.

- long **Phi** [get, set]

- long[] **Phis** [get, set]

7.7.1 Detailed Description

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

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

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.

7.7.2 Constructor & Destructor Documentation

Fast_TopoART_C() [1/3]

```
LibTopoART.Fast_TopoART_C.Fast_TopoART_C (
    long  inputLen,
    long  moduleNum,
    decimal rho_a)
```

This constructor initialises a TopoART-C network.

Parameters

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

Fast_TopoART_C() [2/3]

```
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.
-------------	--------------------------------------

Exceptions

<i>InvalidFileException</i>	Thrown when the given file cannot be loaded.
---	--

Fast_TopoART_C() [3/3]

```
LibTopoART.Fast_TopoART_C.Fast_TopoART_C (
    Stream stream)
```

This constructor loads a saved TopoART-C network from a stream. The stream is left open.

Parameters

<i>stream</i>	A readable <code>Stream</code> containing a network in the binary TopoART-C file format.
---------------	--

Exceptions

InvalidFileException	Thrown when the given stream cannot be loaded.
--------------------------------------	--

7.7.3 Member Function Documentation**Learn()** [1/4]

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

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. The input values are internally scaled from [0, 255] to [0, 1].
--------------	--

Learn() [2/4]

```
void LibTopoART.Fast_TopoART_C.Learn (  
    byte[] input,  
    long classID)
```

This method performs a single training step.

Parameters

<i>input</i>	The input vector to be learnt. The elements of the input vector are internally scaled from [0, 255] to [0, 1].
<i>classID</i>	The class ID corresponding to <i>input</i> . (must be equal to or larger than 0)

Exceptions

InvalidClassIDException	Thrown when <i>classID</i> is less than 0.
---	--

Learn() [3/4]

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

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.
--------------	--------------------------------

Learn() [4/4]

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

This method performs a single training step.

Parameters

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

Exceptions

<i>InvalidClassIDException</i>	Thrown when <i>classID</i> is less than 0.
--	--

Predict() [1/8]

```
long LibTopoART.Fast_TopoART_C.Predict (
    byte[] input)
```

This method predicts the class ID using the default value of nu.

Parameters

<i>input</i>	The input vector the class ID of which is to be predicted. The elements of the input vector are internally scaled from [0, 255] to [0, 1].
--------------	--

Returns

The predicted class ID.

Predict() [2/8]

```
TopoART_C_prediction LibTopoART.Fast_TopoART_C.Predict (
    byte[] input,
    bool?[] mask)
```

This method predicts the class ID using the default value of nu.

Parameters

<i>input</i>	The input vector the class ID of which is to be predicted. The elements of the input vector are internally scaled from [0, 255] to [0, 1].
<i>mask</i>	The mask vector corresponding to <i>input</i> .

Returns

An object of type `TopoART_C_prediction` containing the predicted class ID and a corresponding confidence value.

Predict() [3/8]

```
TopoART_C_prediction LibTopoART.Fast_TopoART_C.Predict (
    byte[] input,
    bool?[] mask,
    long nu)
```

This method predicts the class ID using a custom value of nu.

Parameters

<i>input</i>	The input vector the class ID of which is to be predicted. The elements of the input vector are internally scaled from [0, 255] to [0, 1].
<i>mask</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.

Predict() [4/8]

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

This method predicts the class ID using a custom value of nu.

Parameters

<i>input</i>	The input vector the class ID of which is to be predicted. The elements of the input vector are internally scaled from [0, 255] to [0, 1].
<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.

Predict() [5/8]

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

This method predicts the class ID using the default value of nu.

Parameters

<i>input</i>	The input vector the class ID of which is to be predicted.
--------------	--

Returns

The predicted class ID.

Predict() [6/8]

```
TopoART_C_prediction LibTopoART.Fast_TopoART_C.Predict (
    decimal[ ] input,
    bool?[ ] mask)
```

This method predicts the class ID using the default value of nu.

Parameters

<i>input</i>	The input vector the class ID of which is to be predicted.
<i>mask</i>	The mask vector corresponding to <i>input</i> .

Returns

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

Predict() [7/8]

```
TopoART_C_prediction LibTopoART.Fast_TopoART_C.Predict (
    decimal[ ] input,
    bool?[ ] mask,
    long nu)
```

This method predicts the class ID using a custom value of nu.

Parameters

<i>input</i>	The input vector the class ID of which is to be predicted.
<i>mask</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.

Predict() [8/8]

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

This method predicts the class ID using a custom value of nu.

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.)

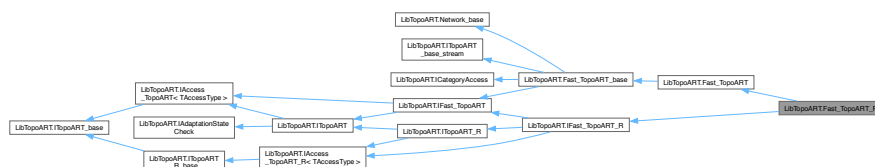
Returns

The predicted class ID.

7.8 LibTopoART.Fast_TopoART_R Class Reference

Class `Fast_TopoART_R` provides an implementation of the TopoART-R neural network as proposed in "Marko Tscherepanow (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.Fast_TopoART_R`:



Public Member Functions

- [Fast_TopoART_R](#) (long iLen, long dLen, long moduleNum, decimal rho_a)
This constructor initialises a TopoART-R network.
- [Fast_TopoART_R](#) (string path)
This constructor loads a saved TopoART-R network.
- [Fast_TopoART_R](#) (Stream stream)
This constructor loads a saved TopoART-R network from a stream. The stream is left open.
- override void [Learn](#) (byte[] input)
This method performs a single training step. The independent variables and the dependent variables are automatically separated.
- override void [Learn](#) (decimal[] input)
This method performs a single training step. The independent variables and the dependent variables are automatically separated.
- void [Learn](#) (byte[] input, byte[] output)
This method performs a single training step.
- void [Learn](#) (decimal[] input, decimal[] output)
This method performs a single training step.
- byte[] [Predict](#) (byte[] input)
This method predicts the dependent variables using the default value of nu.
- decimal[] [Predict](#) (decimal[] input)
This method predicts the dependent variables using the default value of nu.
- byte[] [Predict](#) (byte[] input, long nu)
This method predicts the dependent variables using a custom value of nu.
- decimal[] [Predict](#) (decimal[] input, long nu)
This method predicts the dependent variables using a custom value of nu.
- TopoART_R_prediction< byte > [Predict](#) (byte[] input, bool[] mask)
This method predicts the dependent variables for a given set of independent variables using the default value of nu. Unknown values of independent variables can be signified by setting the corresponding value of mask to true.
- TopoART_R_prediction< decimal > [Predict](#) (decimal[] input, bool[] mask)
This method predicts the dependent variables for a given set of independent variables using the default value of nu. Unknown values of independent variables can be signified by setting the corresponding value of mask to true.
- TopoART_R_prediction< byte > [Predict](#) (byte[] input, bool[] mask, long nu)
This method predicts the dependent variables for a given set of independent variables using a custom value of nu. Unknown values of independent variables can be signified by setting the corresponding value of mask to true.
- TopoART_R_prediction< decimal > [Predict](#) (decimal[] input, bool[] mask, long nu)
This method predicts the dependent variables for a given set of independent variables using a custom value of nu. Unknown values of independent variables can be signified by setting the corresponding value of mask to true.

Public Member Functions inherited from [LibTopoART.Fast_TopoART](#)

- [Fast_TopoART](#) (long inputLen, long moduleNum, decimal rho_a)
This constructor initialises a TopoART network.
- [Fast_TopoART](#) (string path)
This constructor loads a saved TopoART network.
- [Fast_TopoART](#) (Stream stream)
This constructor loads a saved TopoART network from a stream. The stream is left open.
- override void [Learn](#) (byte[] input)
This method performs a single training step.
- override void [Learn](#) (decimal[] input)
This method performs a single training step.

Public Member Functions inherited from [LibTopoART.Fast_TopoART_base](#)

- void [Learn](#) (byte[] input)
This method performs a single training step.
- 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](#) (byte[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (byte[] input, bool[]? mask)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (decimal[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (decimal[] input, bool[]? mask)
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 [SaveText](#) (TextWriter writer)
This method saves the entire network as text to a writer. The writer is flushed but left open.
- void [Save](#) (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void [Save](#) (Stream stream, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network to a stream using the binary file format. The stream is left open.
- void [Save](#) (string path, bool compatibilityMode, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void [Save](#) (Stream stream, bool compatibilityMode, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network to a stream using the binary file format. The stream is left open.
- 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.
- List< [CategoryInfo](#) >? [GetCategories](#) (long moduleIndex=FINAL_MODULE)
This method collects information on the categories of a specified module.

Public Member Functions inherited from [LibTopoART.IAccess_TopoART](#) < [TAccessType](#) >

- [F2_output\[\] GetBMOutput](#) (TAccessType[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (TAccessType[] input, bool[] mask)
This method finds the closest category for a given test input.
- void [Learn](#) (TAccessType[] input)
This method performs a single training step.

Public Member Functions inherited from [LibTopoART.IAccess_TopoART_R< TAccessType >](#)

- void [Learn](#) (TAccessType[] input, TAccessType[] output)
This method performs a single training step.
- TAccessType[] [Predict](#) (TAccessType[] input)
This method predicts the dependent variables using the default value of nu.
- TAccessType[] [Predict](#) (TAccessType[] input, long nu)
This method predicts the dependent variables using a custom value of nu.
- TopoART_R_prediction< TAccessType > [Predict](#) (TAccessType[] input, bool[] mask)
This method predicts the dependent variables for a given set of independent variables using the default value of nu. Unknown values of independent variables can be signified by setting the corresponding value of mask to true.
- TopoART_R_prediction< TAccessType > [Predict](#) (TAccessType[] input, bool[] mask, long nu)
This method predicts the dependent variables for a given set of independent variables using a custom value of nu. Unknown values of independent variables can be signified by setting the corresponding value of mask to true.

Properties

- long **D_len** [get]
Property D_len returns the length of the output vector (dependent variables).
- new decimal **FileFormatVersion** [get]
Property FileFormatVersion returns the version of the file format used by class [Fast_TopoART_R](#).
- long **I_len** [get]
Property I_len returns the length of the input vector (independent variables).
- long **Nu** [get, set]
Property Nu represents the default value used for the maximum cardinality of the neighbourhood set N during prediction. If the parameter nu is not explicitly provided for prediction, this property will be applied. (This parameter does not modify the network. It may be arbitrarily changed for each prediction step.).
- bool **SkipEdgeLearning** [get, set]
Property SkipEdgeLearning enables/disables the [TopoART](#) edge learning mechanism. If the topology of the input data is not required, disabling edge learning may decrease the processing time needed for training.

Properties inherited from [LibTopoART.Fast_TopoART_base](#)

- decimal **Alpha** [get, set]
Property Alpha represents the choice parameter alpha.
- decimal **Beta_sbm** [get, set]
Property Beta_sbm represents the learning rate of the second best-matching nodes.
- long[] **ClusterNum** [get]
Property ClusterNum represents the number of [TopoART](#) clusters found by each module.
- long[] **NodeNum** [get]
Property NodeNum represents the number of [TopoART](#) nodes used by each module.
- decimal **Rho_a** [get]
Property Rho_a represents the vigilance parameter of the first [TopoART](#) module (TA a).
- string **IntegerType** = Common.Types[(int)integerType] [get]
Property IntegerType returns a string containing the data type used for representing integer variables (IDs, parameters, counters, etc.) internally.
- decimal **FileFormatVersion** [get]
Property FileFormatVersion returns the version of the file format used by class [Fast_TopoART_base](#).
- string **FloatType** = Common.Types[(int)floatType] [get]
Property FloatType returns a string containing the data type used for representing floating point variables (input, weights, etc.) internally.
- decimal **TopoARTFileFormatVersion** [get]
Property TopoARTFileFormatVersion returns the version of the file format used by class [Fast_TopoART_base](#).

Properties inherited from [LibTopoART.Network_base](#)

- long **InputLen** [get]
Property `InputLen` returns the length of the input vector.
- long **LearningSteps** [get]
Property `LearningSteps` represents the total number of performed learning steps.
- long **ModuleNum** [get]
- long **Phi** [get, set]
- long[] **Phis** [get, set]
- long **Tau** [get, set]
Property `Tau` represents the parameter tau required for the removal of nodes and edges.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]
Property `InputLen` returns the length of the input vector.
- long **ModuleNum** [get]
- long **LearningSteps** [get]
Property `LearningSteps` represents the total number of performed learning steps.
- long **Tau** [get, set]
Property `Tau` represents the parameter tau required for the removal of nodes and edges.
- long **Phi** [get, set]
- long[] **Phis** [get, set]

Additional Inherited Members**Static Public Attributes inherited from [LibTopoART.Network_base](#)**

- const long **FINAL_MODULE** = [LibTopoART_info.FINAL_MODULE](#)
Instance variable `FINAL_MODULE` gives the value used for indicating that the [TopoART](#) module with the highest index is to be used.

7.8.1 Detailed Description

Class [Fast_TopoART_R](#) provides an implementation of the TopoART-R neural network as proposed in "Marko Tscherepanow (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."

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

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

7.8.2 Constructor & Destructor Documentation

Fast_TopoART_R() [1/3]

```
LibTopoART.Fast_TopoART_R.Fast_TopoART_R (
    long iLen,
    long dLen,
    long moduleNum,
    decimal rho_a)
```

This constructor initialises a TopoART-R network.

Parameters

<i>iLen</i>	The length of the input vector (independent variables) to be learnt.
<i>dLen</i>	The length of the output vector (dependent variables) to be learnt.
<i>moduleNum</i>	The number of TopoART-R modules.
<i>rho_a</i>	The vigilance parameter of the first TopoART-R module (TopoART-R a).

Fast_TopoART_R() [2/3]

```
LibTopoART.Fast_TopoART_R.Fast_TopoART_R (
    string path)
```

This constructor loads a saved TopoART-R network.

Parameters

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

Exceptions

<i>InvalidFileException</i>	Thrown when the given file cannot be loaded.
---	--

Fast_TopoART_R() [3/3]

```
LibTopoART.Fast_TopoART_R.Fast_TopoART_R (
    Stream stream)
```

This constructor loads a saved TopoART-R network from a stream. The stream is left open.

Parameters

<i>stream</i>	A readable <code>Stream</code> containing a network in the binary TopoART-R file format.
---------------	--

Exceptions

<i>InvalidFileException</i>	Thrown when the given stream cannot be loaded.
---	--

7.8.3 Member Function Documentation

Learn() [1/4]

```
override void LibTopoART.Fast_TopoART_R.Learn (
    byte[] input)
```

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. The input values are internally scaled from [0, 255] to [0, 1].
--------------	--

Learn() [2/4]

```
void LibTopoART.Fast_TopoART_R.Learn (
    byte[] input,
    byte[] output)
```

This method performs a single training step.

Parameters

<i>input</i>	The input vector (independent variables) to be learnt. The elements of the input vector are internally scaled from [0, 255] to [0, 1].
<i>output</i>	The output vector (dependent variables) corresponding to <i>input</i> . The elements of the output vector are internally scaled from [0, 255] to [0, 1].

Learn() [3/4]

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

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.
--------------	--------------------------------

Learn() [4/4]

```
void LibTopoART.Fast_TopoART_R.Learn (
    decimal[] input,
    decimal[] output)
```

This method performs a single training step.

Parameters

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

Predict() [1/8]

```
byte[] LibTopoART.Fast_TopoART_R.Predict (
    byte[] input)
```

This method predicts the dependent variables using the default value of `nu`.

Parameters

<i>input</i>	The input vector (independent variables). The elements of the input vector are internally scaled from [0, 255] to [0, 1].
--------------	---

Returns

The predicted values for all dependent variables. If no prediction is possible (e.g., for an untrained network), all values are set to the value `NO_PREDICTION` of `struct TopoART_R_prediction`.

Predict() [2/8]

```
TopoART_R_prediction< byte > LibTopoART.Fast_TopoART_R.Predict (
    byte[] input,
    bool[] mask)
```

This method predicts the dependent variables for a given set of independent variables using the default value of `nu`. Unknown values of independent variables can be signified by setting the corresponding value of *mask* to `true`.

Parameters

<i>input</i>	The input vector (independent variables). The elements of the input vector are internally scaled from [0, 255] to [0, 1].
<i>mask</i>	The mask vector corresponding to <i>input</i> .

Returns

An object of type `TopoART_R_prediction` containing the predicted values for the unknown independent variables and all dependent variables. If no prediction is possible (e.g., for an untrained network), all values are set to `NO_PREDICTION`.

Predict() [3/8]

```
TopoART_R_prediction< byte > LibTopoART.Fast_TopoART_R.Predict (
    byte[] input,
    bool[] mask,
    long nu)
```

This method predicts the dependent variables for a given set of independent variables using a custom value of `nu`. Unknown values of independent variables can be signified by setting the corresponding value of `mask` to `true`.

Parameters

<i>input</i>	The input vector (independent variables). The elements of the input vector are internally scaled from [0, 255] to [0, 1].
<i>mask</i>	The mask vector corresponding to <i>input</i> .
<i>nu</i>	The maximum cardinality of the neighbourhood set N. (In the original TopoART-R network, <i>nu</i> 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. The elements of the predicted vectors are internally scaled from [0, 1] to [0, 255]. If no prediction is possible (e.g., for an untrained network), all values are set to `NO_PREDICTION`.

Predict() [4/8]

```
byte[] LibTopoART.Fast_TopoART_R.Predict (
    byte[] input,
    long nu)
```

This method predicts the dependent variables using a custom value of `nu`.

Parameters

<i>input</i>	The input vector (independent variables). The elements of the input vector are internally scaled from [0, 255] to [0, 1].
<i>nu</i>	The maximum cardinality of the neighbourhood set N. (In the original TopoART-R network, <i>nu</i> 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. The elements of the predicted output vector are internally scaled from [0, 1] to [0, 255]. If no prediction is possible (e.g., for an untrained network), all values are set to the value `NO_PREDICTION` of struct `TopoART_R_prediction`.

Predict() [5/8]

```
decimal[ ] LibTopoART.Fast_TopoART_R.Predict (
    decimal[ ] input)
```

This method predicts the dependent variables using the default value of `nu`.

Parameters

<i>input</i>	The input vector (independent variables).
--------------	---

Returns

The predicted values for all dependent variables. If no prediction is possible (e.g., for an untrained network), all values are set to the value `NO_PREDICTION` of `struct TopoART_R_prediction`.

Predict() [6/8]

```
TopoART_R_prediction< decimal > LibTopoART.Fast_TopoART_R.Predict (
    decimal[ ] input,
    bool[ ] mask)
```

This method predicts the dependent variables for a given set of independent variables using the default value of `nu`. Unknown values of independent variables can be signified by setting the corresponding value of `mask` to `true`.

Parameters

<i>input</i>	The input vector (independent variables).
<i>mask</i>	The mask vector corresponding to <i>input</i> .

Returns

An object of type `TopoART_R_prediction` containing the predicted values for the unknown independent variables and all dependent variables. If no prediction is possible (e.g., for an untrained network), all values are set to `NO_PREDICTION`.

Predict() [7/8]

```
TopoART_R_prediction< decimal > LibTopoART.Fast_TopoART_R.Predict (
    decimal[ ] input,
    bool[ ] mask,
    long nu)
```

This method predicts the dependent variables for a given set of independent variables using a custom value of `nu`. Unknown values of independent variables can be signified by setting the corresponding value of `mask` to `true`.

Parameters

<i>input</i>	The input vector (independent variables).
<i>mask</i>	The mask vector corresponding to <i>input</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. If no prediction is possible (e.g., for an untrained network), all values are set to `NO_PREDICTION`.

Predict() [8/8]

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

This method predicts the dependent variables using a custom value of nu.

Parameters

<i>input</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.)

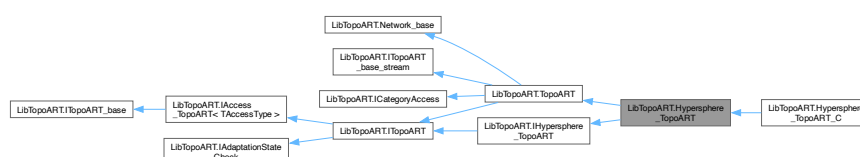
Returns

The predicted values for all dependent variables. If no prediction is possible (e.g., for an untrained network), all values are set to the value `NO_PREDICTION` of struct `TopoART_R_prediction`.

7.9 LibTopoART.Hypersphere_TopoART Class Reference

Class `Hypersphere_TopoART` provides an implementation of the Hypersphere `TopoART` neural network as proposed in "Marko Tscherepanow (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 inputLen, long moduleNum, decimal rho_a)
This constructor initialises a Hypersphere [TopoART](#) network and sets the radial extend parameter to $\text{Math}.\sqrt{\text{inputLen}}/2$.
- [Hypersphere_TopoART](#) (long inputLen, long moduleNum, decimal rho_a, decimal R)
This constructor initialises a Hypersphere [TopoART](#) network.
- [Hypersphere_TopoART](#) (string path)
This constructor loads a saved Hypersphere [TopoART](#) network.
- [Hypersphere_TopoART](#) (Stream stream)
This constructor loads a saved Hypersphere [TopoART](#) network from a stream. The stream is left open.

Public Member Functions inherited from [LibTopoART.TopoART](#)

- [TopoART](#) (long inputLen, long moduleNum, decimal rho_a)
This constructor initialises a [TopoART](#) network.
- [TopoART](#) (string path)
This constructor loads a saved [TopoART](#) network.
- [TopoART](#) (Stream stream)
This constructor loads a saved [TopoART](#) network from a stream. The stream is left open.
- 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)
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 [SaveText](#) (TextWriter writer)
This method saves the entire network as text to a writer. The writer is flushed but left open.
- void [Save](#) (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void [Save](#) (Stream stream, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network to a stream using the binary file format. The stream is left open.
- 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.
- List< [CategoryInfo](#) >? [GetCategories](#) (long moduleIndex=FINAL_MODULE)
This method collects information on the categories of a specified module.

Public Member Functions inherited from [LibTopoART.IAccess_TopoART< TAccessType >](#)

- [F2_output\[\]](#) [GetBMOutput](#) (TAccessType[] input)
This method finds the closest category for a given test input.
- [F2_output\[\]](#) [GetBMOutput](#) (TAccessType[] input, bool[] mask)
This method finds the closest category for a given test input.
- void [Learn](#) (TAccessType[] input)
This method performs a single training step.

Properties

- new decimal **FileFormatVersion** [get]
Property FileFormatVersion returns the version of the file format used by class [Hypersphere_TopoART](#).
- decimal **HypersphereTopoARTFileFormatVersion** [get]
Property HypersphereTopoARTFileFormatVersion returns the version of the file format used by class [Hypersphere_TopoART](#).
- decimal **R** [get]
Property R represents the radial extend parameter R.

Properties inherited from [LibTopoART.TopoART](#)

- decimal **Alpha** [get, set]
Property Alpha represents the choice parameter alpha.
- decimal **Beta_sbm** [get, set]
Property Beta_sbm represents the learning rate of the second best-matching nodes.
- long[] **ClusterNum** [get]
Property ClusterNum represents the number of [TopoART](#) clusters found by each module.
- long[] **NodeNum** [get]
Property NodeNum represents the number of [TopoART](#) nodes used by each module.
- decimal **Rho_a** [get]
Property Rho_a represents the vigilance parameter of the first [TopoART](#) module (TA a).
- string **IntegerType** = Common.Types[(int)integerType] [get]
Property IntegerType returns a string containing the data type used for representing integer variables (IDs, parameters, counters, etc.) internally.
- decimal **FileFormatVersion** [get]
Property FileFormatVersion returns the version of the file format used by class [TopoART](#).
- string **FloatType** = Common.Types[(int)floatType] [get]
Property FloatType returns a string containing the data type used for representing floating point variables (input, weights, etc.) internally.
- decimal **TopoARTFileFormatVersion** [get]
Property TopoARTFileFormatVersion returns the version of the file format used by class [TopoART](#).

Properties inherited from [LibTopoART.Network_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- long **LearningSteps** [get]
Property LearningSteps represents the total number of performed learning steps.
- long **ModuleNum** [get]
- long **Phi** [get, set]
- long[] **Phis** [get, set]
- long **Tau** [get, set]
Property Tau represents the parameter tau required for the removal of nodes and edges.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- long **ModuleNum** [get]
- long **LearningSteps** [get]
Property LearningSteps represents the total number of performed learning steps.
- long **Tau** [get, set]
Property Tau represents the parameter tau required for the removal of nodes and edges.
- long **Phi** [get, set]
- long[] **Phis** [get, set]

Additional Inherited Members

Static Public Attributes inherited from [LibTopoART.Network_base](#)

- const long **FINAL_MODULE** = [LibTopoART_info.FINAL_MODULE](#)
Instance variable FINAL_MODULE gives the value used for indicating that the [TopoART](#) module with the highest index is to be used.

7.9.1 Detailed Description

Class [Hypersphere_TopoART](#) provides an implementation of the Hypersphere [TopoART](#) neural network as proposed in "Marko Tscherepanow (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.

7.9.2 Constructor & Destructor Documentation

[Hypersphere_TopoART\(\)](#) [1/4]

```
LibTopoART.Hypersphere_TopoART.Hypersphere_TopoART (
    long inputLen,
    long moduleNum,
    decimal rho_a)
```

This constructor initialises a Hypersphere [TopoART](#) network and sets the radial extend parameter to `Math.Sqrt(inputLen)/2`.

Parameters

<i>inputLen</i>	The length of input vectors to be learnt.
<i>moduleNum</i>	The number of Hypersphere TopoART modules.
<i>rho_a</i>	The vigilance parameter of the first Hypersphere TopoART module (HTA a).

Hypersphere_TopoART() [2/4]

```
LibTopoART.Hypersphere_TopoART.Hypersphere_TopoART (
    long inputLen,
    long moduleNum,
    decimal rho_a,
    decimal R)
```

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

Parameters

<i>inputLen</i>	The length of input vectors to be learnt.
<i>moduleNum</i>	The number of Hypersphere TopoART modules.
<i>rho_a</i>	The vigilance parameter of the first Hypersphere TopoART module (HTA a).
<i>R</i>	The radial extend parameter.

Hypersphere_TopoART() [3/4]

```
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 TopoART file.
-------------	--

Exceptions

InvalidFileException	Thrown when the given file cannot be loaded.
--------------------------------------	--

Hypersphere_TopoART() [4/4]

```
LibTopoART.Hypersphere_TopoART.Hypersphere_TopoART (
    Stream stream)
```

This constructor loads a saved Hypersphere [TopoART](#) network from a stream. The stream is left open.

Parameters

<i>stream</i>	A readable <code>Stream</code> containing a network in the binary Hypersphere TopoART file format.
---------------	--

Public Member Functions inherited from [LibTopoART.Hypersphere_TopoART](#)

- [Hypersphere_TopoART](#) (long inputLen, long moduleNum, decimal rho_a)
This constructor initialises a Hypersphere [TopoART](#) network and sets the radial extend parameter to $\text{Math}.\sqrt{\text{inputLen}}/2$.
- [Hypersphere_TopoART](#) (long inputLen, long moduleNum, decimal rho_a, decimal R)
This constructor initialises a Hypersphere [TopoART](#) network.
- [Hypersphere_TopoART](#) (string path)
This constructor loads a saved Hypersphere [TopoART](#) network.
- [Hypersphere_TopoART](#) (Stream stream)
This constructor loads a saved Hypersphere [TopoART](#) network from a stream. The stream is left open.

Public Member Functions inherited from [LibTopoART.TopoART](#)

- [TopoART](#) (long inputLen, long moduleNum, decimal rho_a)
This constructor initialises a [TopoART](#) network.
- [TopoART](#) (string path)
This constructor loads a saved [TopoART](#) network.
- [TopoART](#) (Stream stream)
This constructor loads a saved [TopoART](#) network from a stream. The stream is left open.
- 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)
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 [SaveText](#) (TextWriter writer)
This method saves the entire network as text to a writer. The writer is flushed but left open.
- void [Save](#) (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void [Save](#) (Stream stream, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network to a stream using the binary file format. The stream is left open.
- 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.
- List< [CategoryInfo](#) >? [GetCategories](#) (long moduleIndex=FINAL_MODULE)
This method collects information on the categories of a specified module.

Public Member Functions inherited from [LibTopoART.IAccess_TopoART](#) < [TAccessType](#) >

- [F2_output\[\]](#) [GetBMOutput](#) ([TAccessType\[\]](#) input)
This method finds the closest category for a given test input.
- [F2_output\[\]](#) [GetBMOutput](#) ([TAccessType\[\]](#) input, bool[] mask)
This method finds the closest category for a given test input.
- void [Learn](#) ([TAccessType\[\]](#) input)
This method performs a single training step.

Public Member Functions inherited from [LibTopoART.IAccess_TopoART_C< TAccessType >](#)

- void [Learn](#) (TAccessType[] input, long classID)
This method performs a single training step.
- long [Predict](#) (TAccessType[] input)
This method predicts the class ID using the default value of nu.
- long [Predict](#) (TAccessType[] input, long nu)
This method predicts the class ID using a custom value of nu.
- [TopoART_C_prediction Predict](#) (TAccessType[] input, bool[] mask)
This method predicts the class ID using the default value of nu.
- [TopoART_C_prediction Predict](#) (TAccessType[] input, bool[] mask, long nu)
This method predicts the class ID using a custom value of nu.

Static Public Attributes

- const long **UNDEFINED_CLASS_ID** = -2
Instance variable `UNDEFINED_CLASS_ID` gives the value used for indicating that an input sample was predicted to belong to the undefined class; i.e., no class ID was provided for such input samples during training.

Static Public Attributes inherited from [LibTopoART.Network_base](#)

- const long **FINAL_MODULE** = [LibTopoART_info.FINAL_MODULE](#)
Instance variable `FINAL_MODULE` gives the value used for indicating that the [TopoART](#) module with the highest index is to be used.

Properties

- new decimal **FileFormatVersion** [get]
Property `FileFormatVersion` returns the version of the file format used by class [Hypersphere_TopoART_C](#).
- long **Nu** = Common.TopoART_C_default_nu [get, set]
Property `Nu` represents the default value used for the maximum cardinality of the set of enclosing categories E and the neighbourhood set N during prediction. If the parameter `nu` is not explicitly provided for prediction, this property will be applied. (This parameter does not modify the network. It may be arbitrarily changed for each prediction step.).
- bool **SkipEdgeLearning** [get, set]
Property `SkipEdgeLearning` enables/disables the [TopoART](#) edge learning mechanism. If the topology of the input data is not required, disabling edge learning may decrease the processing time needed for training.

Properties inherited from [LibTopoART.Hypersphere_TopoART](#)

- new decimal **FileFormatVersion** [get]
Property `FileFormatVersion` returns the version of the file format used by class [Hypersphere_TopoART](#).
- decimal **HypersphereTopoARTFileFormatVersion** [get]
Property `HypersphereTopoARTFileFormatVersion` returns the version of the file format used by class [Hypersphere_TopoART](#).
- decimal **R** [get]
Property `R` represents the radial extend parameter R.

Properties inherited from [LibTopoART.TopoART](#)

- decimal **Alpha** [get, set]
Property Alpha represents the choice parameter alpha.
- decimal **Beta_sbm** [get, set]
Property Beta_sbm represents the learning rate of the second best-matching nodes.
- long[] **ClusterNum** [get]
Property ClusterNum represents the number of [TopoART](#) clusters found by each module.
- long[] **NodeNum** [get]
Property NodeNum represents the number of [TopoART](#) nodes used by each module.
- decimal **Rho_a** [get]
Property Rho_a represents the vigilance parameter of the first [TopoART](#) module (TA a).
- string **IntegerType** = Common.Types[(int)integerType] [get]
Property IntegerType returns a string containing the data type used for representing integer variables (IDs, parameters, counters, etc.) internally.
- decimal **FileFormatVersion** [get]
Property FileFormatVersion returns the version of the file format used by class [TopoART](#).
- string **FloatType** = Common.Types[(int)floatType] [get]
Property FloatType returns a string containing the data type used for representing floating point variables (input, weights, etc.) internally.
- decimal **TopoARTFileFormatVersion** [get]
Property TopoARTFileFormatVersion returns the version of the file format used by class [TopoART](#).

Properties inherited from [LibTopoART.Network_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- long **LearningSteps** [get]
Property LearningSteps represents the total number of performed learning steps.
- long **ModuleNum** [get]
- long **Phi** [get, set]
- long[] **Phis** [get, set]
- long **Tau** [get, set]
Property Tau represents the parameter tau required for the removal of nodes and edges.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- long **ModuleNum** [get]
- long **LearningSteps** [get]
Property LearningSteps represents the total number of performed learning steps.
- long **Tau** [get, set]
Property Tau represents the parameter tau required for the removal of nodes and edges.
- long **Phi** [get, set]
- long[] **Phis** [get, set]

7.10.1 Detailed Description

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

In contrast to classes [TopoART_C](#) and [Fast_TopoART_C](#), class [Hypersphere_TopoART_C](#) does not require all input to lie in the interval [0, 1]. The input range is controlled by the radial extend parameter R.

7.10.2 Constructor & Destructor Documentation

Hypersphere_TopoART_C() [1/4]

```
LibTopoART.Hypersphere_TopoART_C.Hypersphere_TopoART_C (
    long inputLen,
    long moduleNum,
    decimal rho_a)
```

This constructor initialises a Hypersphere TopoART-C network and sets the radial extend parameter to `Math.Sqrt(inputLen)/2`.

Parameters

<i>inputLen</i>	The length of input vectors to be learnt.
<i>moduleNum</i>	The number of Hypersphere TopoART-C modules.
<i>rho_a</i>	The vigilance parameter of the first Hypersphere TopoART-C module (HTA-C a).

Hypersphere_TopoART_C() [2/4]

```
LibTopoART.Hypersphere_TopoART_C.Hypersphere_TopoART_C (
    long inputLen,
    long moduleNum,
    decimal rho_a,
    decimal R)
```

This constructor initialises a Hypersphere TopoART-C network.

Parameters

<i>inputLen</i>	The length of input vectors to be learnt.
<i>moduleNum</i>	The number of Hypersphere TopoART-C modules.
<i>rho_a</i>	The vigilance parameter of the first Hypersphere TopoART-C module (HTA-C a).

<i>R</i>	The radial extend parameter.
----------	------------------------------

Hypersphere_TopoART_C() [3/4]

```
LibTopoART.Hypersphere_TopoART_C.Hypersphere_TopoART_C (
    string path)
```

This constructor loads a saved Hypersphere TopoART-C network.

Parameters

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

Exceptions

<i>InvalidFileException</i>	Thrown when the given file cannot be loaded.
---	--

Hypersphere_TopoART_C() [4/4]

```
LibTopoART.Hypersphere_TopoART_C.Hypersphere_TopoART_C (
    Stream stream)
```

This constructor loads a saved Hypersphere TopoART-C network from a stream. The stream is left open.

Parameters

<i>stream</i>	A readable <code>Stream</code> containing a network in the binary Hypersphere TopoART-C file format.
---------------	--

Exceptions

<i>InvalidFileException</i>	Thrown when the given stream cannot be loaded.
---	--

7.10.3 Member Function Documentation

Learn() [1/2]

```
override void LibTopoART.Hypersphere_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](#).

Learn() [2/2]

```
void LibTopoART.Hypersphere_TopoART_C.Learn (  
    decimal[] input,  
    long classID)
```

This method performs a single training step.

Parameters

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

Exceptions

InvalidClassIDException	Thrown when <i>classID</i> is less than 0.
---	--

Predict() [1/4]

```
long LibTopoART.Hypersphere_TopoART_C.Predict (  
    decimal[] input)
```

This method predicts the class ID using the default value of nu.

Parameters

<i>input</i>	The input vector the class ID of which is to be predicted.
--------------	--

Returns

The predicted class ID.

Predict() [2/4]

```
TopoART_C_prediction LibTopoART.Hypersphere_TopoART_C.Predict (
    decimal[] input,
    bool?[] mask)
```

This method predicts the class ID using the default value of nu.

Parameters

<i>input</i>	The input vector the class ID of which is to be predicted.
<i>mask</i>	The mask vector corresponding to <i>input</i> .

Returns

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

Predict() [3/4]

```
TopoART_C_prediction LibTopoART.Hypersphere_TopoART_C.Predict (
    decimal[] input,
    bool?[] mask,
    long nu)
```

This method predicts the class ID using a custom value of nu.

Parameters

<i>input</i>	The input vector the class ID of which is to be predicted.
<i>mask</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.

Predict() [4/4]

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

This method predicts the class ID using a custom value of nu.

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.)

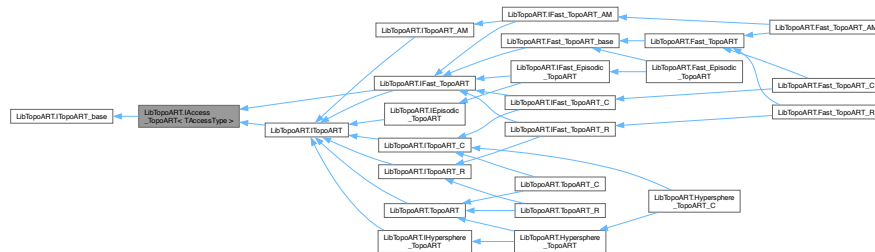
Returns

The predicted class ID.

7.11 LibTopoART.IAccess_TopoART< TAccessType > Interface Template Reference

Interface providing access to the basic [TopoART](#) functionality using input elements of type `_AccessType`.

Inheritance diagram for LibTopoART.IAccess_TopoART< TAccessType >:

**Public Member Functions**

- `F2_output[ ] GetBMOutput (TAccessType[ ] input)`
This method finds the closest category for a given test input.
- `F2_output[ ] GetBMOutput (TAccessType[ ] input, bool[ ] mask)`
This method finds the closest category for a given test input.
- `void Learn (TAccessType[ ] input)`
This method performs a single training step.

Public Member Functions inherited from [LibTopoART.ITopoART_base](#)

- `void ComputeClusterIDs ()`
This method computes the cluster IDs for all neurons.
- `void SaveText (string path)`
This method saves the entire network as a text file.
- `void Save (string path, CompressionLevel compression=CompressionLevel.Fastest)`
This method saves the entire network as a binary file.

Additional Inherited Members

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- 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]
- long[] **Phis** [get, set]
- decimal **Alpha** [get, set]
Property Alpha represents the choice parameter alpha.

7.11.1 Detailed Description

Interface providing access to the basic [TopoART](#) functionality using input elements of type `_AccessType`.

Type Constraints

TAccessType : struct

TAccessType : IConvertible

7.11.2 Member Function Documentation

GetBMOutput() [1/2]

```
F2_output[] LibTopoART.IAccess_TopoART< TAccessType >.GetBMOutput (
    TAccessType[] 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.

GetBMOutput() [2/2]

```
F2_output [ ] LibTopoART.IAccess_TopoART< TAccessType >.GetBMOutput (
    TAccessType [ ] input,
    bool [ ] mask)
```

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

Parameters

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

Learn()

```
void LibTopoART.IAccess_TopoART< TAccessType >.Learn (
    TAccessType [ ] input)
```

This method performs a single training step.

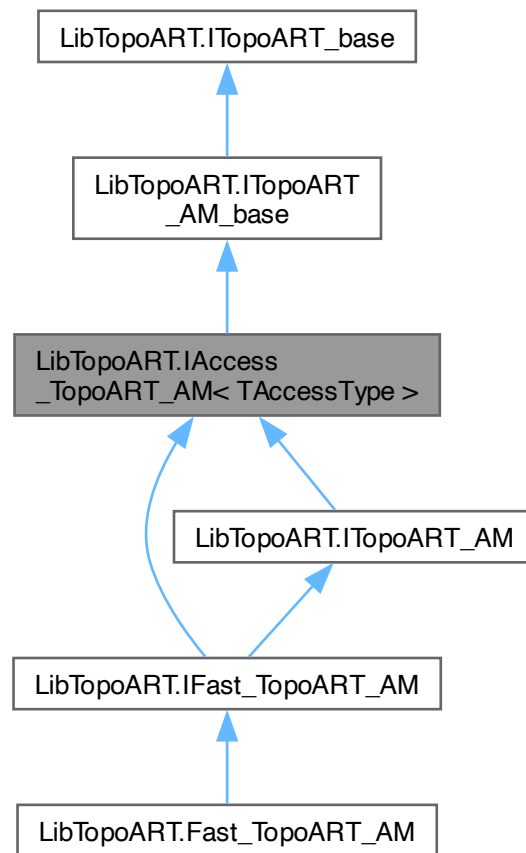
Parameters

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

7.12 LibTopoART.IAccess_TopoART_AM< TAccessType > Interface Template Reference

Interface providing access to the basic TopoART-AM functionality using input elements of type `_AccessType`.

Inheritance diagram for LibTopoART.IAccess_TopoART_AM< TAccessType >:



Public Member Functions

- **F2_output[] GetBMOutput** (TAccessType[] key1, TAccessType[] key2)
This method finds the closest category for a given pair of keys.
- void **Learn** (TAccessType[] key1, TAccessType[] key2)
This method performs a single training step.

Public Member Functions inherited from LibTopoART.ITopoART_base

- void **ComputeClusterIDs** ()
This method computes the cluster IDs for all neurons.
- void **SaveText** (string path)
This method saves the entire network as a text file.
- void **Save** (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.

Additional Inherited Members

Properties inherited from [LibTopoART.ITopoART_AM_base](#)

- long **Key1Len** [get]
Property Key1Len returns the length of the first key vector.
- long **Key2Len** [get]
Property Key2Len returns the length of the second key vector.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- 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]
- long[] **Phis** [get, set]
- decimal **Alpha** [get, set]
Property Alpha represents the choice parameter alpha.

7.12.1 Detailed Description

Interface providing access to the basic TopoART-AM functionality using input elements of type `_AccessType`.

Type Constraints

TAccessType : *struct*

TAccessType : *IConvertible*

7.12.2 Member Function Documentation

GetBMOutput()

```
F2_output[ ] LibTopoART.IAccess_TopoART_AM< TAccessType >.GetBMOutput (
    TAccessType[ ] key1,
    TAccessType[ ] key2)
```

This method finds the closest category for a given pair of keys.

Parameters

<i>key1</i>	The first key vector.
<i>key2</i>	The second key vector corresponding to <i>key1</i> .

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-AM module.

Learn()

```
void LibTopoART.IAccess_TopoART_AM< TAccessType >.Learn (
    TAccessType[ ] key1,
    TAccessType[ ] key2)
```

This method performs a single training step.

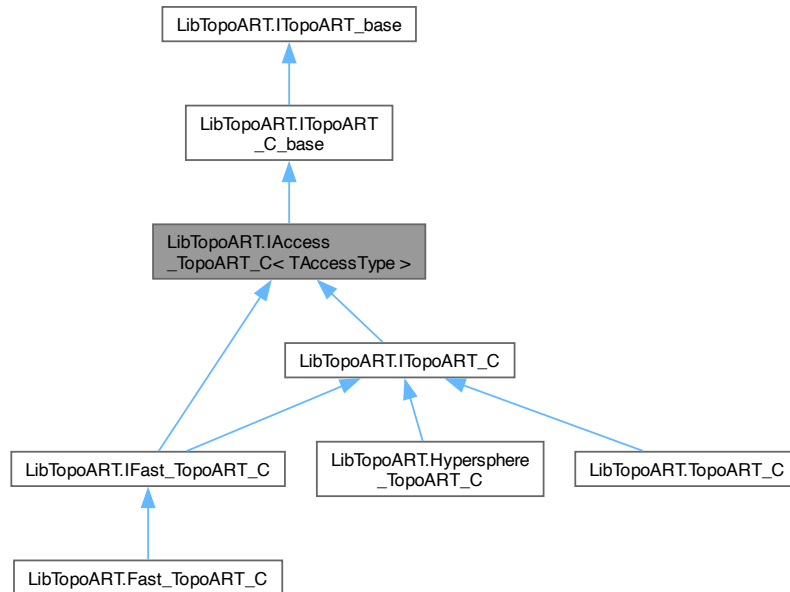
Parameters

<i>key1</i>	The first key vector to be learnt.
<i>key2</i>	The second key vector corresponding to <i>key1</i> .

7.13 LibTopoART.IAccess_TopoART_C< TAccessType > Interface Template Reference

Interface providing access to the basic TopoART-C functionality using input elements of type `_AccessType`.

Inheritance diagram for LibTopoART.IAccess_TopoART_C< TAccessType >:



Public Member Functions

- void **Learn** (TAccessType[] input, long classID)
This method performs a single training step.
- long **Predict** (TAccessType[] input)
This method predicts the class ID using the default value of nu.
- long **Predict** (TAccessType[] input, long nu)
This method predicts the class ID using a custom value of nu.
- **TopoART_C_prediction Predict** (TAccessType[] input, bool[] mask)
This method predicts the class ID using the default value of nu.
- **TopoART_C_prediction Predict** (TAccessType[] input, bool[] mask, long nu)
This method predicts the class ID using a custom value of nu.

Public Member Functions inherited from LibTopoART.ITopoART_base

- void **ComputeClusterIDs** ()
This method computes the cluster IDs for all neurons.
- void **SaveText** (string path)
This method saves the entire network as a text file.
- void **Save** (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.

Additional Inherited Members

Properties inherited from [LibTopoART.ITopoART_C_base](#)

- long **Nu** [get, set]
Property Nu represents the default value used for the maximum cardinality of the set of enclosing categories E and the neighbourhood set N during prediction. If the parameter nu is not explicitly provided for prediction, this property will be applied. (This parameter does not modify the network. It may be arbitrarily changed for each prediction step.).
- bool **SkipEdgeLearning** [get, set]
Property SkipEdgeLearning enables/disables the [TopoART](#) edge learning mechanism. If the topology of the input data is not required, disabling edge learning may decrease the processing time needed for training.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- 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]
- long[] **Phis** [get, set]
- decimal **Alpha** [get, set]
Property Alpha represents the choice parameter alpha.

7.13.1 Detailed Description

Interface providing access to the basic TopoART-C functionality using input elements of type `_AccessType`.

Type Constraints

***TAAccessType* : struct**

***TAAccessType* : IConvertible**

7.13.2 Member Function Documentation

Learn()

```
void LibTopoART.IAccess_TopoART_C< TAccessType >.Learn (
    TAccessType[] input,
    long classID)
```

This method performs a single training step.

Parameters

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

Predict() [1/4]

```
long LibTopoART.IAccess_TopoART_C< TAccessType >.Predict (
    TAccessType[] input)
```

This method predicts the class ID using the default value of nu.

Parameters

<i>input</i>	The input vector the class ID of which is to be predicted.
--------------	--

Returns

The predicted class ID.

Predict() [2/4]

```
TopoART_C_prediction LibTopoART.IAccess_TopoART_C< TAccessType >.Predict (
    TAccessType[] input,
    bool[] mask)
```

This method predicts the class ID using the default value of nu.

Parameters

<i>input</i>	The input vector the class ID of which is to be predicted.
<i>mask</i>	The mask vector corresponding to <i>input</i> .

Returns

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

Predict() [3/4]

```
TopoART_C_prediction LibTopoART.IAccess_TopoART_C< TAccessType >.Predict (
    TAccessType[] input,
    bool[] mask,
    long nu)
```

This method predicts the class ID using a custom value of nu.

Parameters

<i>input</i>	The input vector the class ID of which is to be predicted.
<i>mask</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.

Predict() [4/4]

```
long LibTopoART.IAccess_TopoART_C< TAccessType >.Predict (
    TAccessType[] input,
    long nu)
```

This method predicts the class ID using a custom value of nu.

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.)

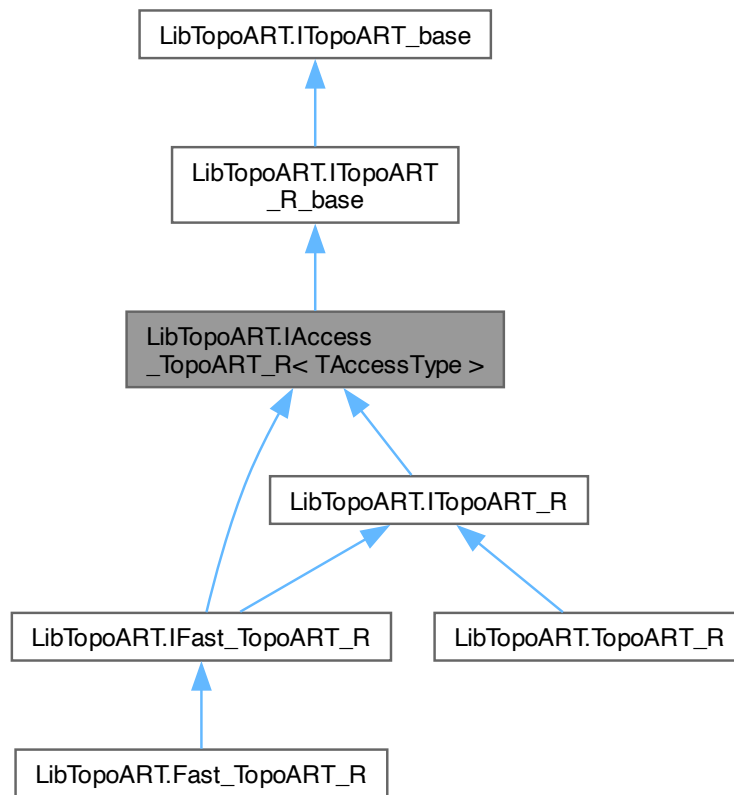
Returns

The predicted class ID.

7.14 LibTopoART.IAccess_TopoART_R< TAccessType > Interface Template Reference

Interface providing access to the basic TopoART-R functionality using input elements of type `_AccessType`.

Inheritance diagram for LibTopoART.IAccess_TopoART_R< TAccessType >:



Public Member Functions

- void **Learn** (TAccessType[] input, TAccessType[] output)
This method performs a single training step.
- TAccessType[] **Predict** (TAccessType[] input)
This method predicts the dependent variables using the default value of nu.
- TAccessType[] **Predict** (TAccessType[] input, long nu)
This method predicts the dependent variables using a custom value of nu.
- TopoART_R_prediction< TAccessType > **Predict** (TAccessType[] input, bool[] mask)
This method predicts the dependent variables for a given set of independent variables using the default value of nu. Unknown values of independent variables can be signified by setting the corresponding value of mask to true.
- TopoART_R_prediction< TAccessType > **Predict** (TAccessType[] input, bool[] mask, long nu)
This method predicts the dependent variables for a given set of independent variables using a custom value of nu. Unknown values of independent variables can be signified by setting the corresponding value of mask to true.

Public Member Functions inherited from LibTopoART.ITopoART_base

- void **ComputeClusterIDs** ()
This method computes the cluster IDs for all neurons.

- void **SaveText** (string path)
This method saves the entire network as a text file.
- void **Save** (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.

Additional Inherited Members

Properties inherited from **LibTopoART.ITopoART_R_base**

- long **D_len** [get]
Property D_len returns the length of the output vector (dependent variables).
- long **I_len** [get]
Property I_len returns the length of the input vector (independent variables).
- long **Nu** [get, set]
Property Nu represents the default value used for the maximum cardinality of the neighbourhood set N during prediction. If the parameter nu is not explicitly provided for prediction, this property will be applied. (This parameter does not modify the network. It may be arbitrarily changed for each prediction step.).
- bool **SkipEdgeLearning** [get, set]
*Property SkipEdgeLearning enables/disables the **TopoART** edge learning mechanism. If the topology of the input data is not required, disabling edge learning may decrease the processing time needed for training.*

Properties inherited from **LibTopoART.ITopoART_base**

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- 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]
- long[] **Phis** [get, set]
- decimal **Alpha** [get, set]
Property Alpha represents the choice parameter alpha.

7.14.1 Detailed Description

Interface providing access to the basic TopoART-R functionality using input elements of type `_AccessType`.

Type Constraints

TAcessType : *struct*

TAcessType : *ICconvertible*

7.14.2 Member Function Documentation

Learn()

```
void LibTopoART.IAccess_TopoART_R< TAccessType >.Learn (
    TAccessType[] input,
    TAccessType[] output)
```

This method performs a single training step.

Parameters

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

Predict() [1/4]

```
TAccessType[] LibTopoART.IAccess_TopoART_R< TAccessType >.Predict (
    TAccessType[] input)
```

This method predicts the dependent variables using the default value of `nu`.

Parameters

<i>input</i>	The input vector (independent variables).
--------------	---

Returns

The predicted values for all dependent variables.

Predict() [2/4]

```
TopoART_R_prediction< TAccessType > LibTopoART.IAccess_TopoART_R< TAccessType >.Predict (
    TAccessType[] input,
    bool[] mask)
```

This method predicts the dependent variables for a given set of independent variables using the default value of `nu`. Unknown values of independent variables can be signified by setting the corresponding value of *mask* to `true`.

Parameters

<i>input</i>	The input vector (independent variables).
<i>mask</i>	The mask vector corresponding to <i>input</i> .

Returns

An object of type `TopoART_R_prediction` containing the predicted values for the unknown independent variables and all dependent variables.

Predict() [3/4]

```
TopoART_R_prediction< TAccessType > LibTopoART.IAccess_TopoART_R< TAccessType >.Predict (
    TAccessType[] input,
    bool[] mask,
    long nu)
```

This method predicts the dependent variables for a given set of independent variables using a custom value of `nu`. Unknown values of independent variables can be signified by setting the corresponding value of `mask` to `true`.

Parameters

<i>input</i>	The input vector (independent variables).
<i>mask</i>	The mask vector corresponding to <i>input</i> .
<i>nu</i>	The maximum cardinality of the neighbourhood set N. (In the original TopoART-R network, <code>nu</code> 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.

Predict() [4/4]

```
TAccessType[] LibTopoART.IAccess_TopoART_R< TAccessType >.Predict (
    TAccessType[] input,
    long nu)
```

This method predicts the dependent variables using a custom value of `nu`.

Parameters

<i>input</i>	The input vector (independent variables).
<i>nu</i>	The maximum cardinality of the neighbourhood set N. (In the original TopoART-R network, <code>nu</code> 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.)

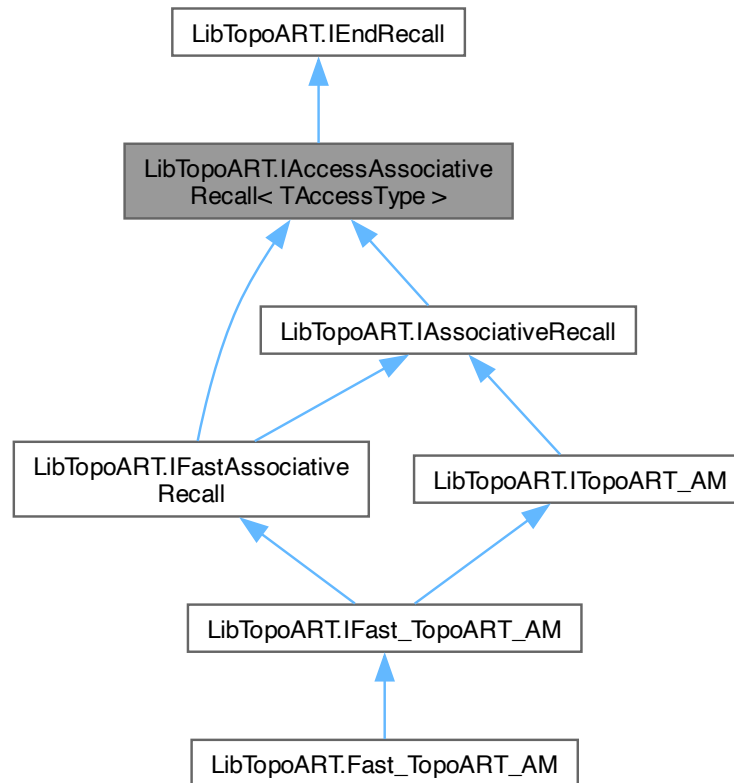
Returns

The predicted values for all dependent variables.

7.15 LibTopoART.IAccessAssociativeRecall< TAccessType > Interface Template Reference

Interface providing access to the basic associative recall functionality using stimulus elements and recall result elements of type `TAccessType`.

Inheritance diagram for LibTopoART.IAccessAssociativeRecall< TAccessType >:



Public Member Functions

- long [BeginRecallKey1](#) (TAccessType[] key2, long moduleIndex=[LibTopoART_info.FINAL_MODULE](#))
This method starts the recall process for the first key vector.
- long [BeginRecallKey2](#) (TAccessType[] key1, long moduleIndex=[LibTopoART_info.FINAL_MODULE](#))
This method starts the recall process for the second key vector.
- bool [RecallStep](#) (out TAccessType[]? recallResult, out decimal F3_activation)
This method performs a single associative recall step.

Public Member Functions inherited from [LibTopoART.IEndRecall](#)

- void [EndRecall](#) ()
This method stops the recall process and frees temporary resources.

7.15.1 Detailed Description

Interface providing access to the basic associative recall functionality using stimulus elements and recall result elements of type TAccessType.

Type Constraints

TAccessType : *struct*

TAccessType : *ICconvertible*

7.15.2 Member Function Documentation

BeginRecallKey1()

```
long LibTopoART.IAccessAssociativeRecall< TAccessType >.BeginRecallKey1 (
    TAccessType[] key2,
    long moduleIndex = LibTopoART_info.FINAL_MODULE)
```

This method starts the recall process for the first key vector.

Parameters

<i>key2</i>	The stimulus (second key vector) which is used to trigger recall.
<i>moduleIndex</i>	Index of the TopoART-AM module to be used for recall. (LibTopoART_info.FINAL_MODULE denotes the module with the highest index.)

Returns

The number of F3 nodes created.

BeginRecallKey2()

```
long LibTopoART.IAccessAssociativeRecall< TAccessType >.BeginRecallKey2 (
    TAccessType[] key1,
    long moduleIndex = LibTopoART_info.FINAL_MODULE)
```

This method starts the recall process for the second key vector.

Parameters

<i>key1</i>	The stimulus (first key vector) which is used to trigger recall.
<i>moduleIndex</i>	Index of the TopoART-AM module to be used for recall. (LibTopoART_info.FINAL_MODULE denotes the module with the highest index.)

Returns

The number of F3 nodes created.

RecallStep()

```
bool LibTopoART.IAccessAssociativeRecall< TAccessType >.RecallStep (
    out TAccessType?[] recallResult,
    out decimal F3_activation)
```

This method performs a single associative recall step.

Parameters

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

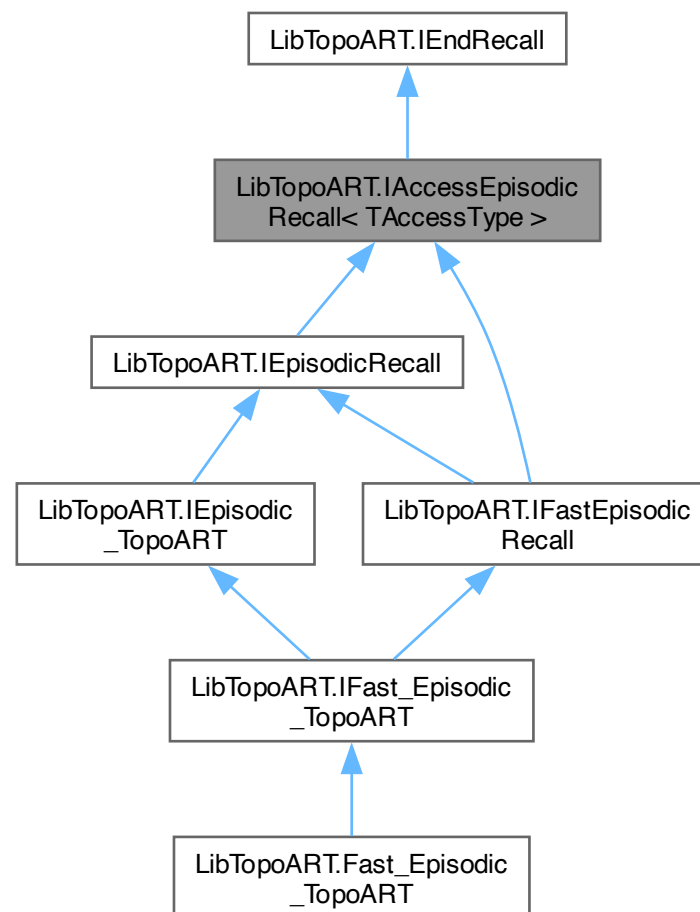
Returns

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

7.16 LibTopoART.IAccessEpisodicRecall< TAccessType > Interface Template Reference

Interface providing access to the basic episodic recall functionality using stimulus elements and recall result elements of type `_AccessType`.

Inheritance diagram for LibTopoART.IAccessEpisodicRecall< TAccessType >:



Public Member Functions

- long [BeginRecall](#) (TAccessType[] stimulus)
This method starts the recall process.
- bool [InterEpisodeRecallStep](#) (out TAccessType[]? recallResult, 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 TAccessType[]? recallResult)
This method performs a single intra-episode recall step.

Public Member Functions inherited from [LibTopoART.IEndRecall](#)

- void [EndRecall](#) ()
This method stops the recall process and frees temporary resources.

7.16.1 Detailed Description

Interface providing access to the basic episodic recall functionality using stimulus elements and recall result elements of type `_AccessType`.

Type Constraints

TAccessType : struct

TAccessType : IConvertible

7.16.2 Member Function Documentation

BeginRecall()

```
long LibTopoART.IAccessEpisodicRecall< TAccessType >.BeginRecall (
    TAccessType[] 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.

```
bool LibTopoART.IAccessEpisodicRecall< TAccessType >.InterEpisodeRecallStep (
    out TAccessType?[] recallResult,
    out decimal F3 activation)
```

Parameters

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

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

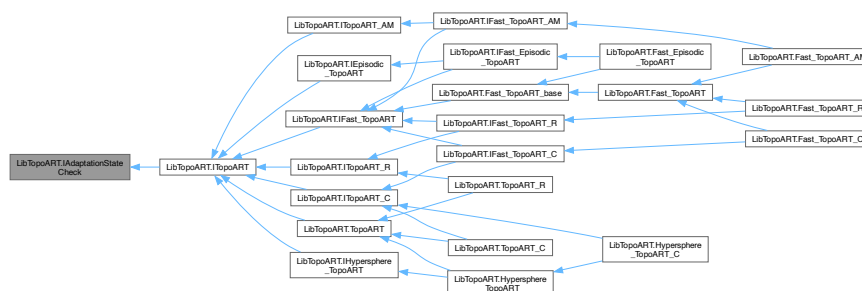
```
bool LibTopoART.IAccessEpisodicRecall< TAccessType >.IntraEpisodeRecallStep (
    out TAccessType?[] recallResult)
```

Parameters

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

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

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.

7.17.1 Detailed Description

Interface enabling checks whether certain adaptations of a network occurred.

7.17.2 Member Function Documentation

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

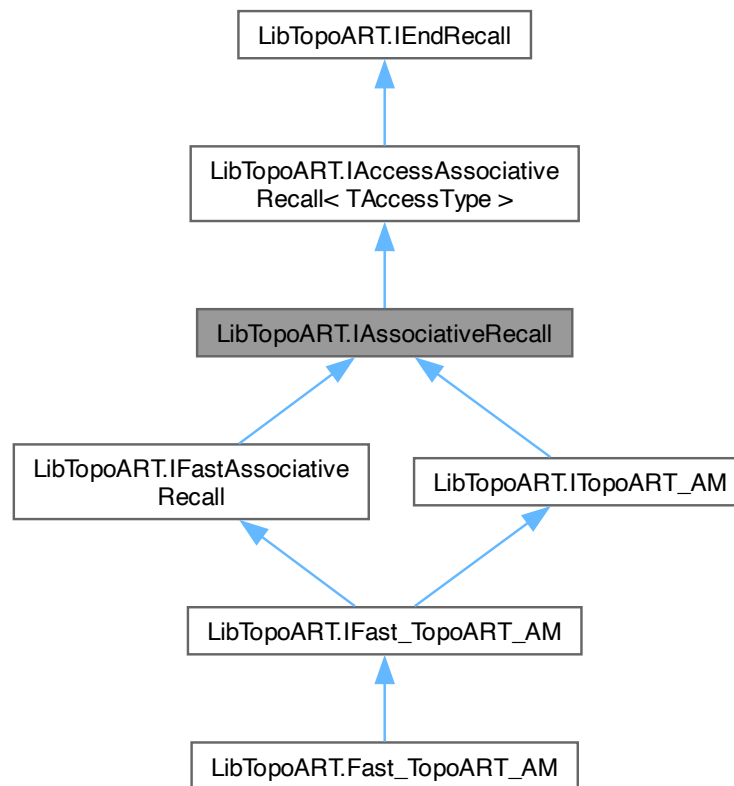
An enumeration describing the adaptation state.

Implemented in [LibTopoART.Fast_TopoART_base](#), and [LibTopoART.TopoART](#).

7.18 LibTopoART.IAssociativeRecall Interface Reference

Interface summarising the associative recall functionality using stimulus elements and recall result elements of type `decimal`.

Inheritance diagram for LibTopoART.IAssociativeRecall:



Additional Inherited Members

Public Member Functions inherited from LibTopoART.IAccessAssociativeRecall< TAccessType >

- long [BeginRecallKey1](#) (TAccessType[] key2, long moduleIndex=LibTopoART_info.FINAL_MODULE)
This method starts the recall process for the first key vector.
- long [BeginRecallKey2](#) (TAccessType[] key1, long moduleIndex=LibTopoART_info.FINAL_MODULE)
This method starts the recall process for the second key vector.
- bool [RecallStep](#) (out TAccessType[]? recallResult, out decimal F3_activation)
This method performs a single associative recall step.

Public Member Functions inherited from LibTopoART.IEndRecall

- void **EndRecall** ()
This method stops the recall process and frees temporary resources.

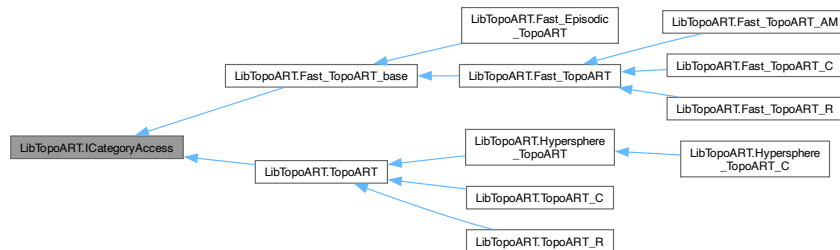
7.18.1 Detailed Description

Interface summarising the associative recall functionality using stimulus elements and recall result elements of type `decimal`.

7.19 LibTopoART.ICategoryAccess Interface Reference

Interface providing access to the learnt categories, e.g., for drawing.

Inheritance diagram for LibTopoART.ICategoryAccess:



Public Member Functions

- List< [CategoryInfo](#) >? [GetCategories](#) (long moduleIndex=[LibTopoART_info.FINAL_MODULE](#))
This method collects information on the categories of a specified module.

7.19.1 Detailed Description

Interface providing access to the learnt categories, e.g., for drawing.

7.19.2 Member Function Documentation

GetCategories()

```
List< CategoryInfo >? LibTopoART.ICategoryAccess.GetCategories (
    long moduleIndex = LibTopoART\_info.FINAL\_MODULE)
```

This method collects information on the categories of a specified module.

Parameters

<i>moduleIndex</i>	The index of the module information on the categories of which is to be returned.
--------------------	---

Returns

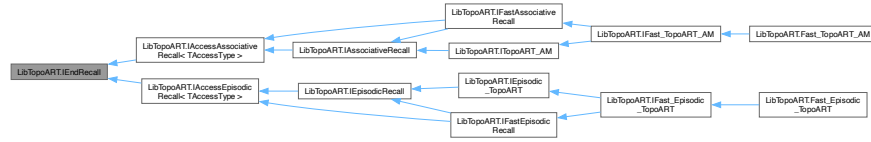
A list containing information about the respective categories.

Implemented in [LibTopoART.Fast_TopoART_base](#), and [LibTopoART.TopoART](#).

7.20 LibTopoART.IEndRecall Interface Reference

Interface summarising the type-independent functionality to stop the recall process.

Inheritance diagram for LibTopoART.IEndRecall:



Public Member Functions

- void **EndRecall** ()
This method stops the recall process and frees temporary resources.

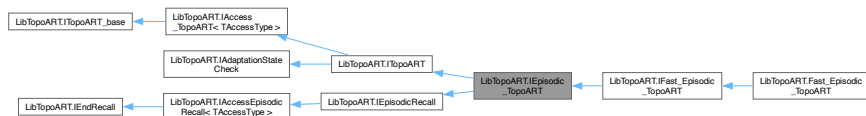
7.20.1 Detailed Description

Interface summarising the type-independent functionality to stop the recall process.

7.21 LibTopoART.IEpisodic_TopopART Interface Reference

Interface summarising the Episodic [TopoART](#) functionality including learning, prediction, episodic recall using input elements, stimulus elements, and recall result elements of type `decimal` as well as adaptation state control.

Inheritance diagram for LibTopoART.IEpisodic_TopopART:



Properties

- long **T_max** [get]
Property `T_max` represents the maximum considered time frame.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- 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]
- long[] **Phis** [get, set]
- decimal **Alpha** [get, set]
Property Alpha represents the choice parameter alpha.

Additional Inherited Members

Public Member Functions inherited from [LibTopoART.IAccess_TopoART](#) < [TAccessType](#) >

- [F2_output\[\] GetBMOutput](#) ([TAccessType\[\]](#) input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) ([TAccessType\[\]](#) input, [bool\[\]](#) mask)
This method finds the closest category for a given test input.
- void [Learn](#) ([TAccessType\[\]](#) input)
This method performs a single training step.

Public Member Functions inherited from [LibTopoART.ITopoART_base](#)

- void **ComputeClusterIDs** ()
This method computes the cluster IDs for all neurons.
- void [SaveText](#) (string path)
This method saves the entire network as a text file.
- void [Save](#) (string path, [CompressionLevel](#) compression=[CompressionLevel.Fastest](#))
This method saves the entire network as a binary file.

Public Member Functions inherited from [LibTopoART.IAdaptationStateCheck](#)

- 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 Member Functions inherited from [LibTopoART.IAccessEpisodicRecall< TAccessType >](#)

- long [BeginRecall](#) (TAccessType[] stimulus)
This method starts the recall process.
- bool [InterEpisodeRecallStep](#) (out TAccessType[]? recallResult, 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 TAccessType[]? recallResult)
This method performs a single intra-episode recall step.

Public Member Functions inherited from [LibTopoART.IEndRecall](#)

- void [EndRecall](#) ()
This method stops the recall process and frees temporary resources.

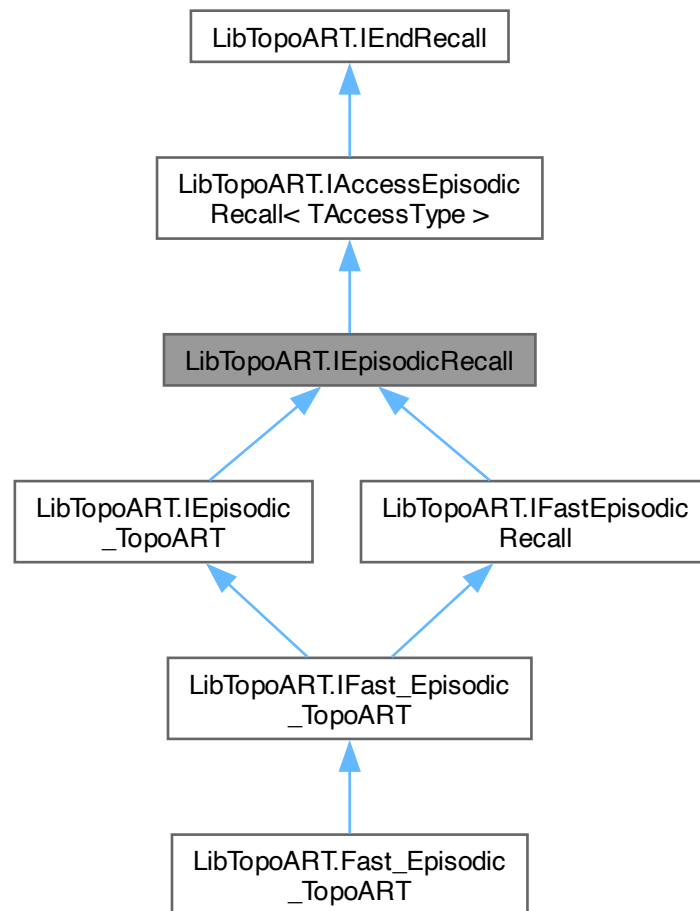
7.21.1 Detailed Description

Interface summarising the Episodic [TopoART](#) functionality including learning, prediction, episodic recall using input elements, stimulus elements, and recall result elements of type `decimal` as well as adaptation state control.

7.22 LibTopoART.IEpisodicRecall Interface Reference

Interface summarising the episodic recall functionality using stimulus elements and recall result elements of type `decimal`.

Inheritance diagram for LibTopoART.IEpisodicRecall:



Additional Inherited Members

Public Member Functions inherited from [LibTopoART.IAccessEpisodicRecall< TAccessType >](#)

- long [BeginRecall](#) (TAccessType[] stimulus)
This method starts the recall process.
- bool [InterEpisodeRecallStep](#) (out TAccessType[]? recallResult, 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 TAccessType[]? recallResult)
This method performs a single intra-episode recall step.

Public Member Functions inherited from [LibTopoART.IEndRecall](#)

- void [EndRecall](#) ()
This method stops the recall process and frees temporary resources.

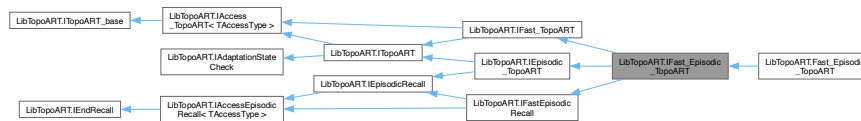
7.22.1 Detailed Description

Interface summarising the episodic recall functionality using stimulus elements and recall result elements of type `decimal`.

7.23 LibTopoART.IFast_Episodic_TopoART Interface Reference

Interface summarising the Episodic [TopoART](#) functionality including learning, prediction, episodic recall using input elements, stimulus elements, and recall result elements of type `byte` or of type `decimal` as well as adaptation state control.

Inheritance diagram for LibTopoART.IFast_Episodic_TopoART:



Additional Inherited Members

Public Member Functions inherited from [LibTopoART.IAccess_TopoART< TAccessType >](#)

- `F2_output[]` [GetBMOutput](#) (`TAccessType[]` input)
This method finds the closest category for a given test input.
- `F2_output[]` [GetBMOutput](#) (`TAccessType[]` input, `bool[]` mask)
This method finds the closest category for a given test input.
- `void` [Learn](#) (`TAccessType[]` input)
This method performs a single training step.

Public Member Functions inherited from [LibTopoART.ITopoART_base](#)

- `void` [ComputeClusterIDs](#) ()
This method computes the cluster IDs for all neurons.
- `void` [SaveText](#) (string path)
This method saves the entire network as a text file.
- `void` [Save](#) (string path, `CompressionLevel` compression=`CompressionLevel.Fastest`)
This method saves the entire network as a binary file.

Public Member Functions inherited from [LibTopoART.IAdaptationStateCheck](#)

- `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 Member Functions inherited from [LibTopoART.IAccessEpisodicRecall< TAccessType >](#)

- long [BeginRecall](#) (TAccessType[] stimulus)
This method starts the recall process.
- bool [InterEpisodeRecallStep](#) (out TAccessType[]? recallResult, 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 TAccessType[]? recallResult)
This method performs a single intra-episode recall step.

Public Member Functions inherited from [LibTopoART.IEndRecall](#)

- void [EndRecall](#) ()
This method stops the recall process and frees temporary resources.

Properties inherited from [LibTopoART.IEpisodic_TopoART](#)

- long [T_max](#) [get]
Property T_max represents the maximum considered time frame.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long [InputLen](#) [get]
Property InputLen returns the length of the input vector.
- 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]
- long[] [Phis](#) [get, set]
- decimal [Alpha](#) [get, set]
Property Alpha represents the choice parameter alpha.

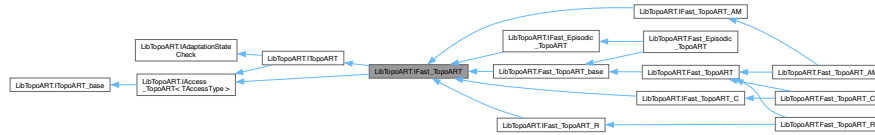
7.23.1 Detailed Description

Interface summarising the Episodic [TopoART](#) functionality including learning, prediction, episodic recall using input elements, stimulus elements, and recall result elements of type `byte` or of type `decimal` as well as adaptation state control.

7.24 LibTopoART.IFast_TopoART Interface Reference

Interface summarising the [TopoART](#) functionality including learning and prediction using input elements of type `byte` or of type `decimal` as well as adaptation state control.

Inheritance diagram for LibTopoART.IFast_TopoART:



Additional Inherited Members

Public Member Functions inherited from [LibTopoART.IAccess_TopoART < TAccessType >](#)

- `F2_output[]` [GetBMOutput](#) (`TAccessType[]` input)
This method finds the closest category for a given test input.
- `F2_output[]` [GetBMOutput](#) (`TAccessType[]` input, `bool[]` mask)
This method finds the closest category for a given test input.
- void [Learn](#) (`TAccessType[]` input)
This method performs a single training step.

Public Member Functions inherited from [LibTopoART.ITopoART_base](#)

- void [ComputeClusterIDs](#) ()
This method computes the cluster IDs for all neurons.
- void [SaveText](#) (string path)
This method saves the entire network as a text file.
- void [Save](#) (string path, `CompressionLevel` compression=`CompressionLevel.Fastest`)
This method saves the entire network as a binary file.

Public Member Functions inherited from [LibTopoART.IAdaptationStateCheck](#)

- 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.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- 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]
- long[] **Phis** [get, set]
- decimal **Alpha** [get, set]
Property Alpha represents the choice parameter alpha.

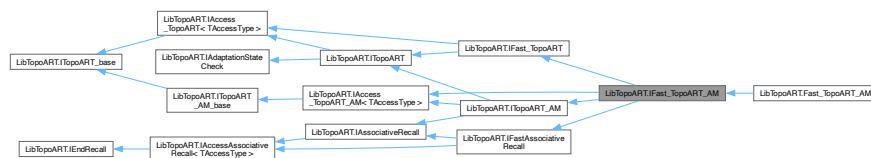
7.24.1 Detailed Description

Interface summarising the [TopoART](#) functionality including learning and prediction using input elements of type `byte` or of type `decimal` as well as adaptation state control.

7.25 LibTopoART.IFast_TopoART_AM Interface Reference

Interface summarising the Episodic [TopoART](#) functionality including learning, prediction, episodic recall using input elements, stimulus elements, and recall result elements of type `byte` or of type `decimal` as well as adaptation state control.

Inheritance diagram for LibTopoART.IFast_TopoART_AM:



Additional Inherited Members

Public Member Functions inherited from [LibTopoART.IAccess_TopoART< TAccessType >](#)

- `F2_output[] GetBMOutput (TAccessType[] input)`
This method finds the closest category for a given test input.
- `F2_output[] GetBMOutput (TAccessType[] input, bool[] mask)`
This method finds the closest category for a given test input.
- `void Learn (TAccessType[] input)`
This method performs a single training step.

Public Member Functions inherited from [LibTopoART.ITopoART_base](#)

- void **ComputeClusterIDs** ()
This method computes the cluster IDs for all neurons.
- void **SaveText** (string path)
This method saves the entire network as a text file.
- void **Save** (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.

Public Member Functions inherited from [LibTopoART.IAdaptationStateCheck](#)

- 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 Member Functions inherited from [LibTopoART.IAccess_TopoART_AM](#)< [TAccessType](#) >

- [F2_output\[\]](#) **GetBMOutput** ([TAccessType\[\]](#) key1, [TAccessType\[\]](#) key2)
This method finds the closest category for a given pair of keys.
- void **Learn** ([TAccessType\[\]](#) key1, [TAccessType\[\]](#) key2)
This method performs a single training step.

Public Member Functions inherited from [LibTopoART.IAccessAssociativeRecall](#)< [TAccessType](#) >

- long **BeginRecallKey1** ([TAccessType\[\]](#) key2, long moduleIndex=[LibTopoART_info.FINAL_MODULE](#))
This method starts the recall process for the first key vector.
- long **BeginRecallKey2** ([TAccessType\[\]](#) key1, long moduleIndex=[LibTopoART_info.FINAL_MODULE](#))
This method starts the recall process for the second key vector.
- bool **RecallStep** (out [TAccessType\[\]](#)? recallResult, out decimal F3_activation)
This method performs a single associative recall step.

Public Member Functions inherited from [LibTopoART.IEndRecall](#)

- void **EndRecall** ()
This method stops the recall process and frees temporary resources.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- 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]
- long[] **Phis** [get, set]
- decimal **Alpha** [get, set]
Property Alpha represents the choice parameter alpha.

Properties inherited from [LibTopoART.ITopoART_AM_base](#)

- long **Key1Len** [get]
Property Key1Len returns the length of the first key vector.
- long **Key2Len** [get]
Property Key2Len returns the length of the second key vector.

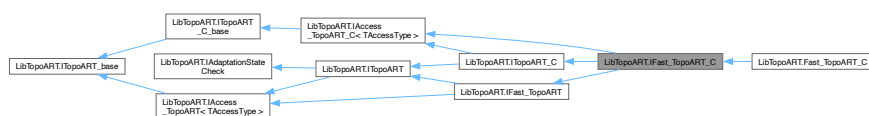
7.25.1 Detailed Description

Interface summarising the Episodic [TopoART](#) functionality including learning, prediction, episodic recall using input elements, stimulus elements, and recall result elements of type `byte` or of type `decimal` as well as adaptation state control.

7.26 LibTopoART.IFast_TopoART_C Interface Reference

Interface summarising the TopoART-C functionality including learning and prediction using input elements of type `byte` or of type `decimal` as well as adaptation state control.

Inheritance diagram for LibTopoART.IFast_TopoART_C:



Additional Inherited Members

Public Member Functions inherited from [LibTopoART.IAccess_TopoART< TAccessType >](#)

- [F2_output\[\] GetBMOutput](#) (TAccessType[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (TAccessType[] input, bool[] mask)
This method finds the closest category for a given test input.
- void [Learn](#) (TAccessType[] input)
This method performs a single training step.

Public Member Functions inherited from [LibTopoART.ITopoART_base](#)

- void [ComputeClusterIDs](#) ()
This method computes the cluster IDs for all neurons.
- void [SaveText](#) (string path)
This method saves the entire network as a text file.
- void [Save](#) (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.

Public Member Functions inherited from [LibTopoART.IAdaptationStateCheck](#)

- 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 Member Functions inherited from [LibTopoART.IAccess_TopoART_C< TAccessType >](#)

- void [Learn](#) (TAccessType[] input, long classID)
This method performs a single training step.
- long [Predict](#) (TAccessType[] input)
This method predicts the class ID using the default value of nu.
- long [Predict](#) (TAccessType[] input, long nu)
This method predicts the class ID using a custom value of nu.
- [TopoART_C_prediction Predict](#) (TAccessType[] input, bool[] mask)
This method predicts the class ID using the default value of nu.
- [TopoART_C_prediction Predict](#) (TAccessType[] input, bool[] mask, long nu)
This method predicts the class ID using a custom value of nu.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- 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]
- long[] **Phis** [get, set]
- decimal **Alpha** [get, set]
Property Alpha represents the choice parameter alpha.

Properties inherited from [LibTopoART.ITopoART_C_base](#)

- long **Nu** [get, set]
Property Nu represents the default value used for the maximum cardinality of the set of enclosing categories E and the neighbourhood set N during prediction. If the parameter nu is not explicitly provided for prediction, this property will be applied. (This parameter does not modify the network. It may be arbitrarily changed for each prediction step.).
- bool **SkipEdgeLearning** [get, set]
Property SkipEdgeLearning enables/disables the [TopoART](#) edge learning mechanism. If the topology of the input data is not required, disabling edge learning may decrease the processing time needed for training.

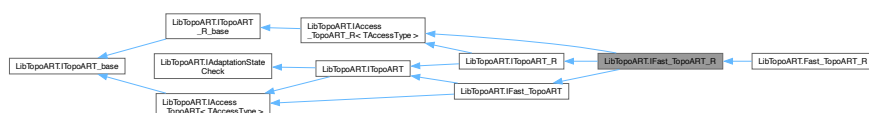
7.26.1 Detailed Description

Interface summarising the TopoART-C functionality including learning and prediction using input elements of type `byte` or of type `decimal` as well as adaptation state control.

7.27 LibTopoART.IFast_TopopART_R Interface Reference

Interface summarising the TopoART-R functionality including learning and prediction using input elements and output elements of type `byte` or of type `decimal` as well as adaptation state control.

Inheritance diagram for LibTopoART.IFast_TopopART_R:



Additional Inherited Members

Public Member Functions inherited from [LibTopoART.IAccess_TopoART< TAccessType >](#)

- [F2_output\[\] GetBMOutput](#) (TAccessType[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (TAccessType[] input, bool[] mask)
This method finds the closest category for a given test input.
- void [Learn](#) (TAccessType[] input)
This method performs a single training step.

Public Member Functions inherited from [LibTopoART.ITopoART_base](#)

- void [ComputeClusterIDs](#) ()
This method computes the cluster IDs for all neurons.
- void [SaveText](#) (string path)
This method saves the entire network as a text file.
- void [Save](#) (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.

Public Member Functions inherited from [LibTopoART.IAdaptationStateCheck](#)

- 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 Member Functions inherited from [LibTopoART.IAccess_TopoART_R< TAccessType >](#)

- void [Learn](#) (TAccessType[] input, TAccessType[] output)
This method performs a single training step.
- TAccessType[] [Predict](#) (TAccessType[] input)
This method predicts the dependent variables using the default value of nu.
- TAccessType[] [Predict](#) (TAccessType[] input, long nu)
This method predicts the dependent variables using a custom value of nu.
- TopoART_R_prediction< TAccessType > [Predict](#) (TAccessType[] input, bool[] mask)
This method predicts the dependent variables for a given set of independent variables using the default value of nu. Unknown values of independent variables can be signified by setting the corresponding value of mask to true.
- TopoART_R_prediction< TAccessType > [Predict](#) (TAccessType[] input, bool[] mask, long nu)
This method predicts the dependent variables for a given set of independent variables using a custom value of nu. Unknown values of independent variables can be signified by setting the corresponding value of mask to true.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- 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]
- long[] **Phis** [get, set]
- decimal **Alpha** [get, set]
Property Alpha represents the choice parameter alpha.

Properties inherited from [LibTopoART.ITopoART_R_base](#)

- long **D_len** [get]
Property D_len returns the length of the output vector (dependent variables).
- long **I_len** [get]
Property I_len returns the length of the input vector (independent variables).
- long **Nu** [get, set]
Property Nu represents the default value used for the maximum cardinality of the neighbourhood set N during prediction. If the parameter nu is not explicitly provided for prediction, this property will be applied. (This parameter does not modify the network. It may be arbitrarily changed for each prediction step.).
- bool **SkipEdgeLearning** [get, set]
Property SkipEdgeLearning enables/disables the [TopoART](#) edge learning mechanism. If the topology of the input data is not required, disabling edge learning may decrease the processing time needed for training.

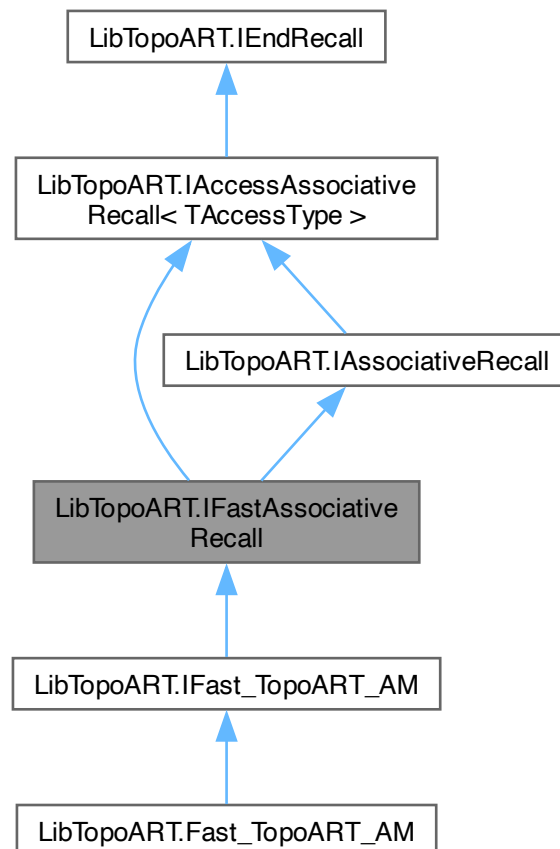
7.27.1 Detailed Description

Interface summarising the TopoART-R functionality including learning and prediction using input elements and output elements of type `byte` or of type `decimal` as well as adaptation state control.

7.28 LibTopoART.IFastAssociativeRecall Interface Reference

Interface summarising the associative recall functionality using stimulus elements and recall result elements of type `byte` or of type `decimal`.

Inheritance diagram for LibTopoART.IFastAssociativeRecall:



Additional Inherited Members

Public Member Functions inherited from **LibTopoART.IAccessAssociativeRecall< TAccessType >**

- long **BeginRecallKey1** (TAccessType[] key2, long moduleIndex=LibTopoART_info.FINAL_MODULE)
This method starts the recall process for the first key vector.
- long **BeginRecallKey2** (TAccessType[] key1, long moduleIndex=LibTopoART_info.FINAL_MODULE)
This method starts the recall process for the second key vector.
- bool **RecallStep** (out TAccessType[]? recallResult, out decimal F3_activation)
This method performs a single associative recall step.

Public Member Functions inherited from [LibTopoART.IEndRecall](#)

- void **EndRecall** ()

This method stops the recall process and frees temporary resources.

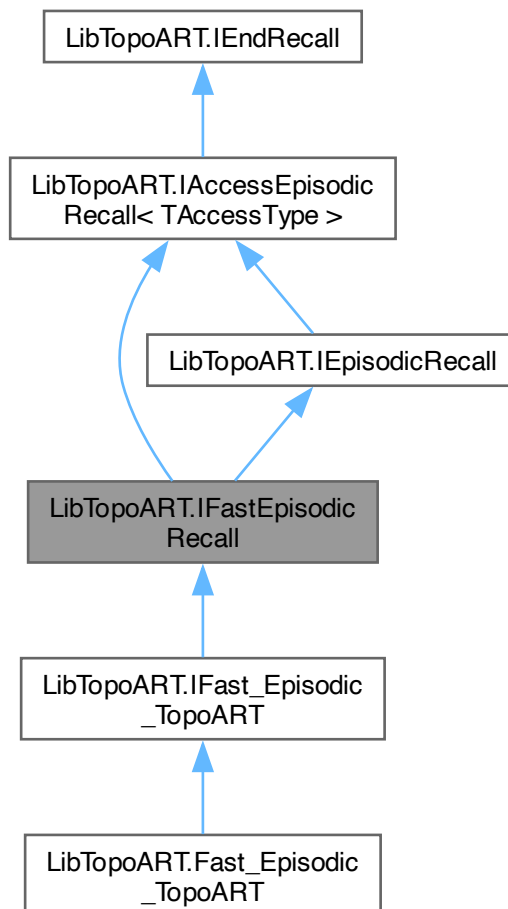
7.28.1 Detailed Description

Interface summarising the associative recall functionality using stimulus elements and recall result elements of type `byte` or of type `decimal`.

7.29 LibTopoART.IFastEpisodicRecall Interface Reference

Interface summarising the episodic recall functionality using stimulus elements and recall result elements of type `byte` or of type `decimal`.

Inheritance diagram for LibTopoART.IFastEpisodicRecall:



Additional Inherited Members**Public Member Functions inherited from [LibTopoART.IAccessEpisodicRecall](#)< [TAccessType](#) >**

- long [BeginRecall](#) (TAccessType[] stimulus)
This method starts the recall process.
- bool [InterEpisodeRecallStep](#) (out TAccessType[]? recallResult, 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 TAccessType[]? recallResult)
This method performs a single intra-episode recall step.

Public Member Functions inherited from [LibTopoART.IEndRecall](#)

- void [EndRecall](#) ()
This method stops the recall process and frees temporary resources.

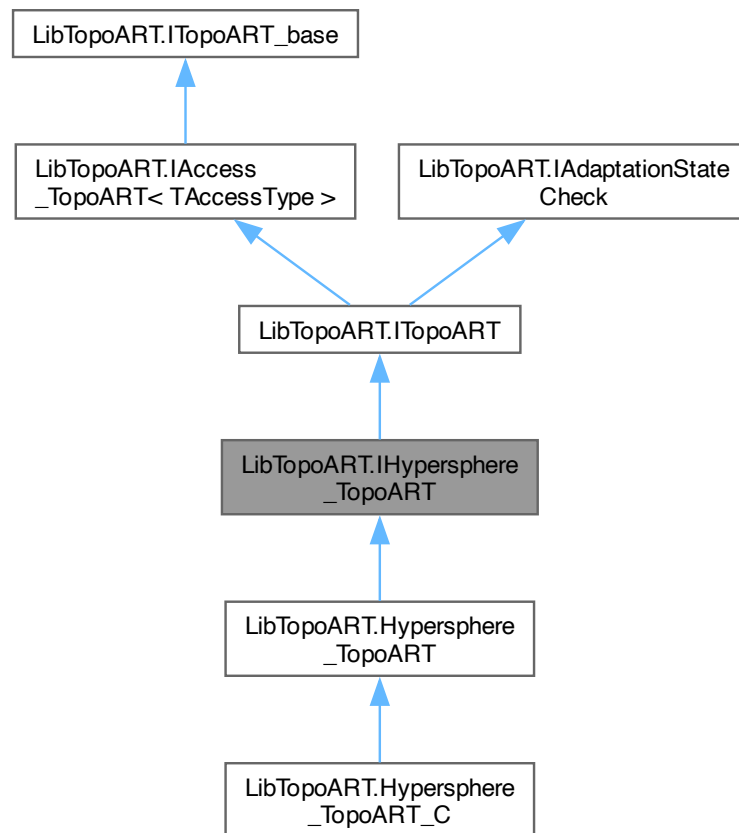
7.29.1 Detailed Description

Interface summarising the episodic recall functionality using stimulus elements and recall result elements of type `byte` or of type `decimal`.

7.30 LibTopoART.IHypersphere_TopoART Interface Reference

Interface summarising the Hypersphere [TopoART](#) functionality including learning and prediction using input elements of type `decimal` as well as adaptation state control.

Inheritance diagram for LibTopoART.IHypersphere_TopoART:



Properties

- decimal **R** [get]
Property R represents the radial extend parameter R.

Properties inherited from LibTopoART.ITopoART_base

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- 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]
- long[] **Phis** [get, set]
- decimal **Alpha** [get, set]
Property Alpha represents the choice parameter alpha.

Additional Inherited Members

Public Member Functions inherited from [LibTopoART.IAccess_TopoART< TAccessType >](#)

- [F2_output\[\] GetBMOutput](#) (TAccessType[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (TAccessType[] input, bool[] mask)
This method finds the closest category for a given test input.
- void [Learn](#) (TAccessType[] input)
This method performs a single training step.

Public Member Functions inherited from [LibTopoART.ITopoART_base](#)

- void **ComputeClusterIDs** ()
This method computes the cluster IDs for all neurons.
- void [SaveText](#) (string path)
This method saves the entire network as a text file.
- void [Save](#) (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.

Public Member Functions inherited from [LibTopoART.IAdaptationStateCheck](#)

- 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.

7.30.1 Detailed Description

Interface summarising the Hypersphere [TopoART](#) functionality including learning and prediction using input elements of type `decimal` as well as adaptation state control.

7.31 LibTopoART.InvalidClassIDException Class Reference

Exception signalling an invalid class ID.

7.31.1 Detailed Description

Exception signalling an invalid class ID.

7.32 LibTopoART.InvalidFileException Class Reference

Exception signalling an invalid file.

7.32.1 Detailed Description

Exception signalling an invalid file.

7.33 LibTopoART.InvalidModuleIndexException Class Reference

Exception signalling an invalid module index.

7.33.1 Detailed Description

Exception signalling an invalid module index.

7.34 LibTopoART.InvalidNumberException Class Reference

Exception signalling an invalid number.

7.34.1 Detailed Description

Exception signalling an invalid number.

7.35 LibTopoART.InvalidSizeException Class Reference

Exception signalling an invalid size.

7.35.1 Detailed Description

Exception signalling an invalid size.

7.36 LibTopoART.InvalidStateException Class Reference

Exception signalling an invalid state of the neural network.

7.36.1 Detailed Description

Exception signalling an invalid state of the neural network.

7.37 LibTopoART.InvalidTypeException Class Reference

Exception signalling an invalid type.

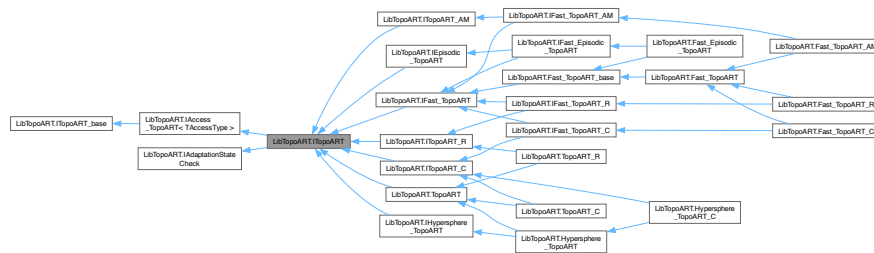
7.37.1 Detailed Description

Exception signalling an invalid type.

7.38 LibTopoART.ITopoART Interface Reference

Interface summarising the [TopoART](#) functionality including learning and prediction using input elements of type `decimal` as well as adaptation state control.

Inheritance diagram for LibTopoART.ITopoART:



Additional Inherited Members

Public Member Functions inherited from [LibTopoART.IAccess_TopoART< TAccessType >](#)

- [F2_output\[\] GetBMOutput](#) (TAccessType[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (TAccessType[] input, bool[] mask)
This method finds the closest category for a given test input.
- void [Learn](#) (TAccessType[] input)
This method performs a single training step.

Public Member Functions inherited from [LibTopoART.ITopoART_base](#)

- void [ComputeClusterIDs](#) ()
This method computes the cluster IDs for all neurons.
- void [SaveText](#) (string path)
This method saves the entire network as a text file.
- void [Save](#) (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.

Public Member Functions inherited from [LibTopoART.IAdaptationStateCheck](#)

- 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.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]
Property `InputLen` returns the length of the input vector.
- 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]
- long[] **Phis** [get, set]
- decimal **Alpha** [get, set]
Property `Alpha` represents the choice parameter alpha.

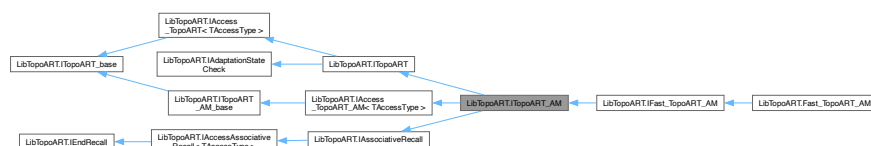
7.38.1 Detailed Description

Interface summarising the [TopoART](#) functionality including learning and prediction using input elements of type `decimal` as well as adaptation state control.

7.39 LibTopoART.ITopoART_AM Interface Reference

Interface summarising the TopoART-AM functionality including learning, prediction, associative recall using input elements, stimulus elements, and recall result elements of type `decimal` as well as adaptation state control.

Inheritance diagram for [LibTopoART.ITopoART_AM](#):



Additional Inherited Members

Public Member Functions inherited from [LibTopoART.IAccess_TopoART< TAccessType >](#)

- [F2_output\[\] GetBMOutput](#) (TAccessType[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (TAccessType[] input, bool[] mask)
This method finds the closest category for a given test input.
- void [Learn](#) (TAccessType[] input)
This method performs a single training step.

Public Member Functions inherited from [LibTopoART.ITopoART_base](#)

- void [ComputeClusterIDs](#) ()
This method computes the cluster IDs for all neurons.
- void [SaveText](#) (string path)
This method saves the entire network as a text file.
- void [Save](#) (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.

Public Member Functions inherited from [LibTopoART.IAdaptationStateCheck](#)

- 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 Member Functions inherited from [LibTopoART.IAccess_TopoART_AM< TAccessType >](#)

- [F2_output\[\] GetBMOutput](#) (TAccessType[] key1, TAccessType[] key2)
This method finds the closest category for a given pair of keys.
- void [Learn](#) (TAccessType[] key1, TAccessType[] key2)
This method performs a single training step.

Public Member Functions inherited from [LibTopoART.IAccessAssociativeRecall< TAccessType >](#)

- long [BeginRecallKey1](#) (TAccessType[] key2, long moduleIndex=LibTopoART_info.FINAL_MODULE)
This method starts the recall process for the first key vector.
- long [BeginRecallKey2](#) (TAccessType[] key1, long moduleIndex=LibTopoART_info.FINAL_MODULE)
This method starts the recall process for the second key vector.
- bool [RecallStep](#) (out TAccessType[]? recallResult, out decimal F3_activation)
This method performs a single associative recall step.

Public Member Functions inherited from [LibTopoART.IEndRecall](#)

- void [EndRecall](#) ()
This method stops the recall process and frees temporary resources.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- 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]
- long[] **Phis** [get, set]
- decimal **Alpha** [get, set]
Property Alpha represents the choice parameter alpha.

Properties inherited from [LibTopoART.ITopoART_AM_base](#)

- long **Key1Len** [get]
Property Key1Len returns the length of the first key vector.
- long **Key2Len** [get]
Property Key2Len returns the length of the second key vector.

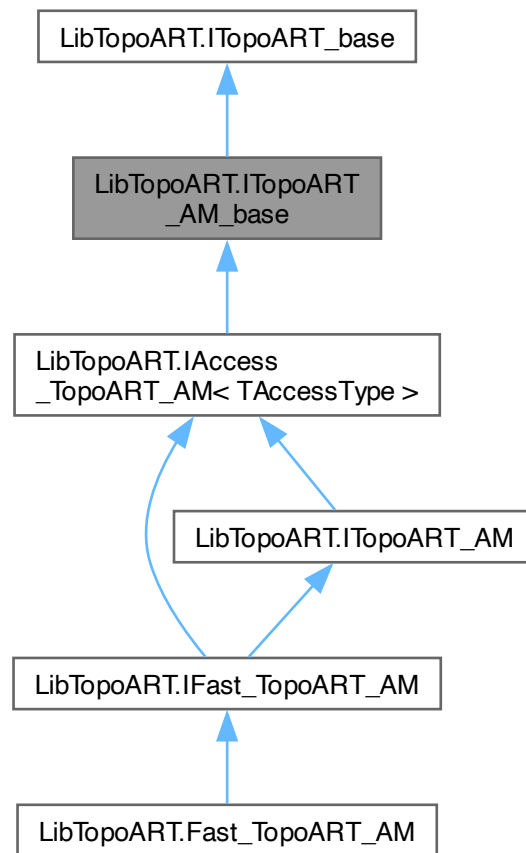
7.39.1 Detailed Description

Interface summarising the TopoART-AM functionality including learning, prediction, associative recall using input elements, stimulus elements, and recall result elements of type `decimal` as well as adaptation state control.

7.40 LibTopoART.ITopoART_AM_base Interface Reference

Interface summarising the basic TopoART-AM functionality excluding learning and prediction.

Inheritance diagram for LibTopoART.ITopoART_AM_base:



Properties

- long **Key1Len** [get]
Property Key1Len returns the length of the first key vector.
- long **Key2Len** [get]
Property Key2Len returns the length of the second key vector.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- 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]
- long[] **Phis** [get, set]

- decimal **Alpha** [get, set]

Property Alpha represents the choice parameter alpha.

Additional Inherited Members

Public Member Functions inherited from **LibTopoART.ITopoART_base**

- void **ComputeClusterIDs** ()

This method computes the cluster IDs for all neurons.

- void **SaveText** (string path)

This method saves the entire network as a text file.

- void **Save** (string path, CompressionLevel compression=CompressionLevel.Fastest)

This method saves the entire network as a binary file.

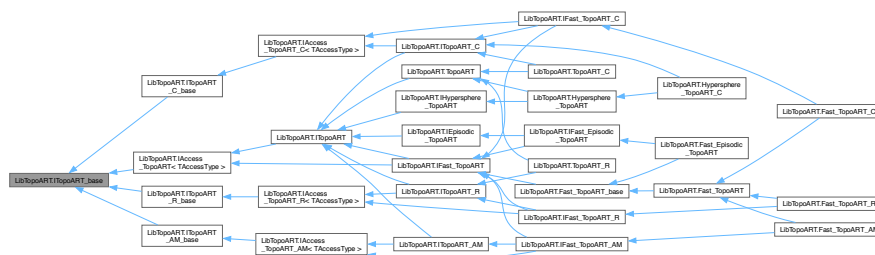
7.40.1 Detailed Description

Interface summarising the basic TopoART-AM functionality excluding learning and prediction.

7.41 LibTopoART.ITopoART_base Interface Reference

Interface summarising the basic **TopoART** functionality excluding learning and prediction.

Inheritance diagram for LibTopoART.ITopoART_base:



Public Member Functions

- void **ComputeClusterIDs** ()

This method computes the cluster IDs for all neurons.

- void **SaveText** (string path)

This method saves the entire network as a text file.

- void **Save** (string path, CompressionLevel compression=CompressionLevel.Fastest)

This method saves the entire network as a binary file.

Properties

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- 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]
- long[] **Phis** [get, set]
- decimal **Alpha** [get, set]
Property Alpha represents the choice parameter alpha.

7.41.1 Detailed Description

Interface summarising the basic [TopoART](#) functionality excluding learning and prediction.

7.41.2 Member Function Documentation

Save()

```
void LibTopoART.ITopoART_base.Save (
    string path,
    CompressionLevel compression = CompressionLevel.Fastest)
```

This method saves the entire network as a binary file.

Parameters

<i>path</i>	A string representing the path of the file to save.
<i>compression</i>	Compression level of the save file (Compression is not supported by LibTopoART v0.93 and below.)

Implemented in [LibTopoART.Fast_TopoART_base](#), and [LibTopoART.TopoART](#).

SaveText()

```
void LibTopoART.ITopoART_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.
-------------	---

Implemented in [LibTopoART.Fast_TopoART_base](#), and [LibTopoART.TopoART](#).

7.41.3 Property Documentation

ModuleNum

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

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

Phi

```
long LibTopoART.ITopoART_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.

Phis

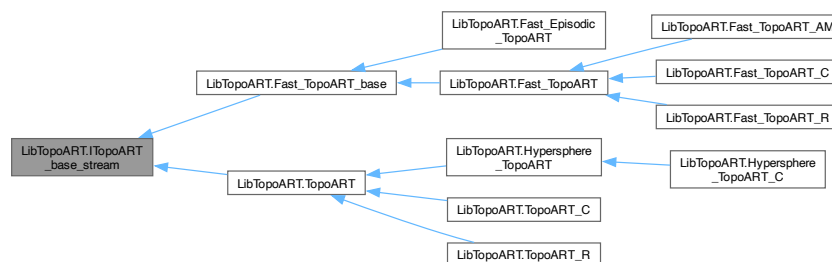
```
long [ ] LibTopoART.ITopoART_base.Phis [get], [set]
```

Property `Phis` constitutes an extension of property `Phi` that enables individual values of `phi` for each module. By this, the removal of nodes and edges as well as the propagation of input to subsequent [TopoART](#) modules can be controlled in a task-dependent manner.

7.42 LibTopoART.ITopoART_base_stream Interface Reference

Interface extending the basic [TopoART](#) functionality by stream-based saving.

Inheritance diagram for `LibTopoART.ITopoART_base_stream`:



Public Member Functions

- void [SaveText](#) (TextWriter writer)
This method saves the entire network as text to a writer. The writer is flushed but left open.
- void [Save](#) (Stream stream, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network to a stream using the binary file format. The stream is left open.

7.42.1 Detailed Description

Interface extending the basic [TopoART](#) functionality by stream-based saving.

7.42.2 Member Function Documentation

Save()

```
void LibTopoART.ITopoART_base_stream.Save (
    Stream stream,
    CompressionLevel compression = CompressionLevel.Fastest)
```

This method saves the entire network to a stream using the binary file format. The stream is left open.

Parameters

<i>stream</i>	A writable <code>Stream</code> the network is saved to.
<i>compression</i>	Compression level of the saved data (Compression is not supported by LibTopoART v0.93 and below.)

Implemented in [LibTopoART.Fast_TopoART_base](#), and [LibTopoART.TopoART](#).

SaveText()

```
void LibTopoART.ITopoART_base_stream.SaveText (
    TextWriter writer)
```

This method saves the entire network as text to a writer. The writer is flushed but left open.

Parameters

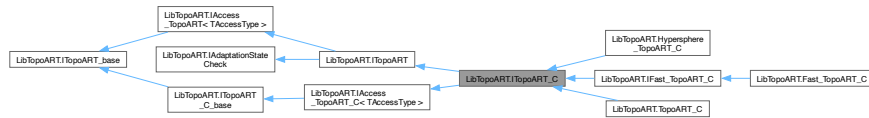
<i>writer</i>	A <code>TextWriter</code> the network is saved to.
---------------	--

Implemented in [LibTopoART.Fast_TopoART_base](#), and [LibTopoART.TopoART](#).

7.43 LibTopoART.ITopoART_C Interface Reference

Interface summarising the TopoART-C functionality including learning and prediction using input elements of type `decimal` as well as adaptation state control.

Inheritance diagram for LibTopoART.ITopoART_C:



Additional Inherited Members

Public Member Functions inherited from LibTopoART.IAccess_TopoART< TAccessType >

- `F2_output[] GetBMOutput (TAccessType[] input)`
This method finds the closest category for a given test input.
- `F2_output[] GetBMOutput (TAccessType[] input, bool[] mask)`
This method finds the closest category for a given test input.
- `void Learn (TAccessType[] input)`
This method performs a single training step.

Public Member Functions inherited from LibTopoART.ITopoART_base

- `void ComputeClusterIDs ()`
This method computes the cluster IDs for all neurons.
- `void SaveText (string path)`
This method saves the entire network as a text file.
- `void Save (string path, CompressionLevel compression=CompressionLevel.Fastest)`
This method saves the entire network as a binary file.

Public Member Functions inherited from LibTopoART.IAdaptationStateCheck

- `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 Member Functions inherited from LibTopoART.IAccess_TopoART_C< TAccessType >

- `void Learn (TAccessType[] input, long classID)`
This method performs a single training step.
- `long Predict (TAccessType[] input)`
This method predicts the class ID using the default value of nu.
- `long Predict (TAccessType[] input, long nu)`
This method predicts the class ID using a custom value of nu.
- `TopoART_C_prediction Predict (TAccessType[] input, bool[] mask)`
This method predicts the class ID using the default value of nu.
- `TopoART_C_prediction Predict (TAccessType[] input, bool[] mask, long nu)`
This method predicts the class ID using a custom value of nu.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- 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]
- long[] **Phis** [get, set]
- decimal **Alpha** [get, set]
Property Alpha represents the choice parameter alpha.

Properties inherited from [LibTopoART.ITopoART_C_base](#)

- long **Nu** [get, set]
Property Nu represents the default value used for the maximum cardinality of the set of enclosing categories E and the neighbourhood set N during prediction. If the parameter nu is not explicitly provided for prediction, this property will be applied. (This parameter does not modify the network. It may be arbitrarily changed for each prediction step.).
- bool **SkipEdgeLearning** [get, set]
Property SkipEdgeLearning enables/disables the [TopoART](#) edge learning mechanism. If the topology of the input data is not required, disabling edge learning may decrease the processing time needed for training.

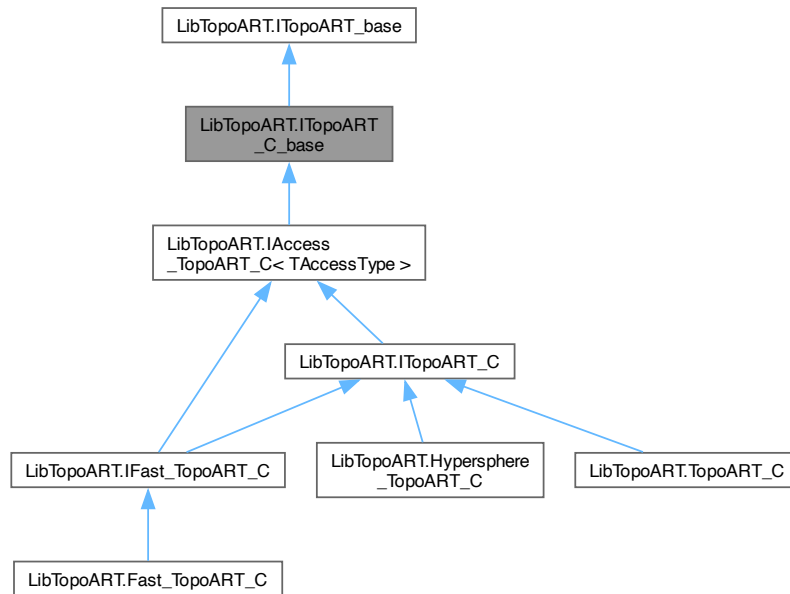
7.43.1 Detailed Description

Interface summarising the TopoART-C functionality including learning and prediction using input elements of type `decimal` as well as adaptation state control.

7.44 LibTopoART.ITopoART_C_base Interface Reference

Interface summarising the basic TopoART-C functionality excluding learning and prediction.

Inheritance diagram for LibTopoART.ITopoART_C_base:



Properties

- long **Nu** [get, set]

Property *Nu* represents the default value used for the maximum cardinality of the set of enclosing categories *E* and the neighbourhood set *N* during prediction. If the parameter *nu* is not explicitly provided for prediction, this property will be applied. (This parameter does not modify the network. It may be arbitrarily changed for each prediction step.).

- bool **SkipEdgeLearning** [get, set]

Property *SkipEdgeLearning* enables/disables the [TopoART](#) edge learning mechanism. If the topology of the input data is not required, disabling edge learning may decrease the processing time needed for training.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]

Property *InputLen* returns the length of the input vector.

- 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]
- long[] **Phis** [get, set]
- decimal **Alpha** [get, set]

Property Alpha represents the choice parameter α .

Additional Inherited Members

Public Member Functions inherited from LibTopoART.ITopoART_base

- void **ComputeClusterIDs** ()
This method computes the cluster IDs for all neurons.
- void **SaveText** (string path)
This method saves the entire network as a text file.
- void **Save** (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.

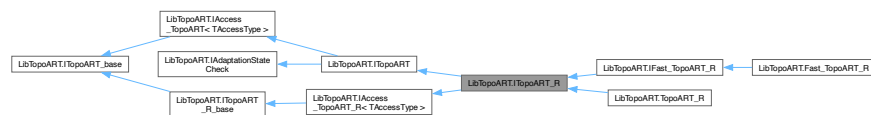
7.44.1 Detailed Description

Interface summarising the basic TopoART-C functionality excluding learning and prediction.

7.45 LibTopoART.ITopoART_R Interface Reference

Interface summarising the TopoART-R functionality including learning and prediction using input elements and output elements of type `decimal` as well as adaptation state control.

Inheritance diagram for LibTopoART.ITopoART_R:



Additional Inherited Members

Public Member Functions inherited from [LibTopoART.IAccess_TopoART< TAccessType >](#)

- `F2_output[] GetBMOutput (TAccessType[] input)`
This method finds the closest category for a given test input.
- `F2_output[] GetBMOutput (TAccessType[] input, bool[] mask)`
This method finds the closest category for a given test input.
- `void Learn (TAccessType[] input)`
This method performs a single training step.

Public Member Functions inherited from [LibTopoART.ITopoART_base](#)

- void **ComputeClusterIDs** ()
This method computes the cluster IDs for all neurons.
- void **SaveText** (string path)
This method saves the entire network as a text file.
- void **Save** (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.

Public Member Functions inherited from [LibTopoART.IAdaptationStateCheck](#)

- 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 Member Functions inherited from [LibTopoART.IAccess_TopoART_R< TAccessType >](#)

- void **Learn** (TAccessType[] input, TAccessType[] output)
This method performs a single training step.
- TAccessType[] **Predict** (TAccessType[] input)
This method predicts the dependent variables using the default value of nu.
- TAccessType[] **Predict** (TAccessType[] input, long nu)
This method predicts the dependent variables using a custom value of nu.
- TopoART_R_prediction< TAccessType > **Predict** (TAccessType[] input, bool[] mask)
This method predicts the dependent variables for a given set of independent variables using the default value of nu. Unknown values of independent variables can be signified by setting the corresponding value of mask to true.
- TopoART_R_prediction< TAccessType > **Predict** (TAccessType[] input, bool[] mask, long nu)
This method predicts the dependent variables for a given set of independent variables using a custom value of nu. Unknown values of independent variables can be signified by setting the corresponding value of mask to true.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- 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]
- long[] **Phis** [get, set]
- decimal **Alpha** [get, set]
Property Alpha represents the choice parameter alpha.

Properties inherited from LibTopoART.ITopoART_R_base

- long **D_len** [get]
Property *D_len* returns the length of the output vector (dependent variables).
- long **I_len** [get]
Property *I_len* returns the length of the input vector (independent variables).
- long **Nu** [get, set]
Property *Nu* represents the default value used for the maximum cardinality of the neighbourhood set *N* during prediction. If the parameter *nu* is not explicitly provided for prediction, this property will be applied. (This parameter does not modify the network. It may be arbitrarily changed for each prediction step.).
- bool **SkipEdgeLearning** [get, set]
Property *SkipEdgeLearning* enables/disables the [TopoART](#) edge learning mechanism. If the topology of the input data is not required, disabling edge learning may decrease the processing time needed for training.

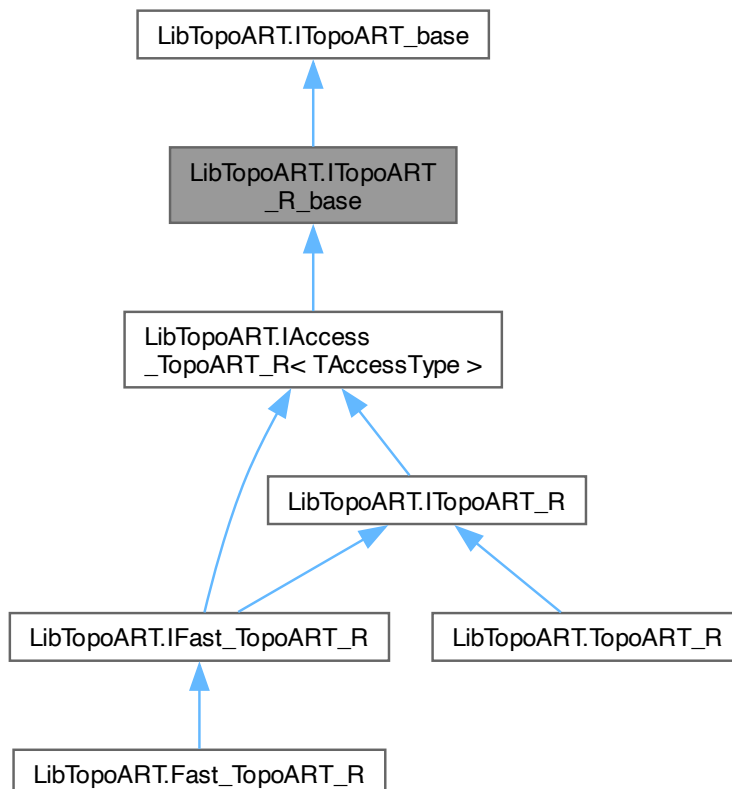
7.45.1 Detailed Description

Interface summarising the TopoART-R functionality including learning and prediction using input elements and output elements of type `decimal` as well as adaptation state control.

7.46 LibTopoART.ITopoART_R_base Interface Reference

Interface summarising the basic TopoART-R functionality excluding learning and prediction.

Inheritance diagram for LibTopoART.ITopoART_R_base:



Properties

- long **D_len** [get]
Property D_len returns the length of the output vector (dependent variables).
- long **I_len** [get]
Property I_len returns the length of the input vector (independent variables).
- long **Nu** [get, set]
Property Nu represents the default value used for the maximum cardinality of the neighbourhood set N during prediction. If the parameter nu is not explicitly provided for prediction, this property will be applied. (This parameter does not modify the network. It may be arbitrarily changed for each prediction step.).
- bool **SkipEdgeLearning** [get, set]
Property SkipEdgeLearning enables/disables the [TopoART](#) edge learning mechanism. If the topology of the input data is not required, disabling edge learning may decrease the processing time needed for training.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- 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]
- long[] **Phis** [get, set]
- decimal **Alpha** [get, set]
Property Alpha represents the choice parameter alpha.

Additional Inherited Members

Public Member Functions inherited from [LibTopoART.ITopoART_base](#)

- void **ComputeClusterIDs** ()
This method computes the cluster IDs for all neurons.
- void **SaveText** (string path)
This method saves the entire network as a text file.
- void **Save** (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.

7.46.1 Detailed Description

Interface summarising the basic TopoART-R functionality excluding learning and prediction.

7.47 LibTopoART.LibTopoART_control Struct Reference

Struct [LibTopoART_control](#) provides fields to control the general behaviour of [LibTopoART](#).

Static Public Attributes

- static [VerbosityLevel](#) **verbosity** = [VerbosityLevel.Verbose](#)
Instance variable `verbosity` enables controlling the number of messages issued by [LibTopoART](#).

7.47.1 Detailed Description

Struct [LibTopoART_control](#) provides fields to control the general behaviour of [LibTopoART](#).

7.48 LibTopoART.LibTopoART_info Struct Reference

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

Static Public Attributes

- const decimal **version** = 1.01m
Instance variable `version` represents the version of [LibTopoART](#).
- 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.
- const long **UNDEFINED** = -1
Instance variable `UNDEFINED` gives the value used for indicating undefined and uninitialised variables.
- const long **FINAL_MODULE** = [UNDEFINED](#)
Instance variable `FINAL_MODULE` gives the value used for indicating that the [TopoART](#) module with the highest index is to be used.

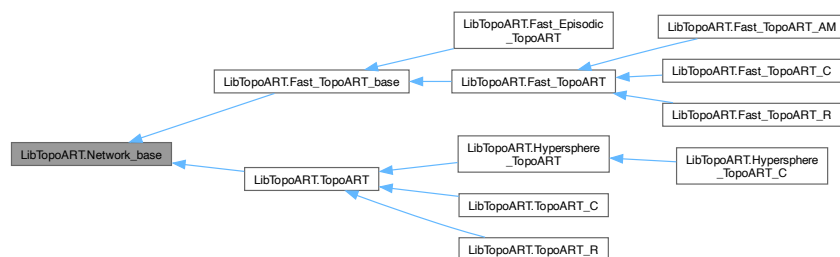
7.48.1 Detailed Description

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

7.49 LibTopoART.Network_base Class Reference

Class [Network_base](#) provides the functionality required by all neural network implementations of [LibTopoART](#).

Inheritance diagram for [LibTopoART.Network_base](#):



Static Public Attributes

- `const long FINAL_MODULE = LibTopoART_info.FINAL_MODULE`

Instance variable `FINAL_MODULE` gives the value used for indicating that the [TopoART](#) module with the highest index is to be used.

Properties

- `long InputLen` [get]

Property `InputLen` returns the length of the input vector.

- `long LearningSteps` [get]

Property `LearningSteps` represents the total number of performed learning steps.

- `long ModuleNum` [get]

- `long Phi` [get, set]

- `long[] Phis` [get, set]

- `long Tau` [get, set]

Property `Tau` represents the parameter `tau` required for the removal of nodes and edges.

7.49.1 Detailed Description

Class [Network_base](#) provides the functionality required by all neural network implementations of [LibTopoART](#).

7.49.2 Property Documentation

ModuleNum

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

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

Phi

```
long LibTopoART.Network_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.

Phis

```
long [] LibTopoART.Network_base.Phis [get], [set]
```

Property `Phis` constitutes an extension of property `Phi` that enables individual values of `phi` for each module. By this, the removal of nodes and edges as well as the propagation of input to subsequent [TopoART](#) modules can be controlled in a task-dependent manner.

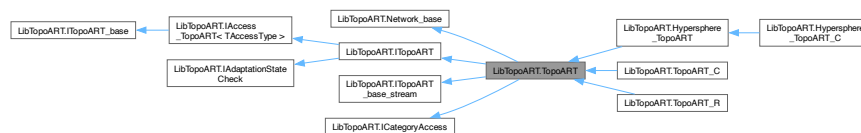
Exceptions

InvalidSizeException	Thrown when the array length does not fit the module number.
--------------------------------------	--

7.50 LibTopoART.TopoART Class Reference

Class [TopoART](#) provides an implementation of the [TopoART](#) neural network as proposed in "Marko Tscherepanow (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 inputLen, long moduleNum, decimal rho_a)
This constructor initialises a [TopoART](#) network.
- [TopoART](#) (string path)
This constructor loads a saved [TopoART](#) network.
- [TopoART](#) (Stream stream)
This constructor loads a saved [TopoART](#) network from a stream. The stream is left open.
- 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)
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 [SaveText](#) (TextWriter writer)
This method saves the entire network as text to a writer. The writer is flushed but left open.
- void [Save](#) (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.

- void **Save** (Stream stream, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network to a stream using the binary file format. The stream is left open.
- 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.
- List< **CategoryInfo** >? **GetCategories** (long moduleIndex=FINAL_MODULE)
This method collects information on the categories of a specified module.

Public Member Functions inherited from **LibTopoART.IAccess_TopoART< TAccessType >**

- **F2_output[] GetBMOutput** (TAccessType[] input)
This method finds the closest category for a given test input.
- **F2_output[] GetBMOutput** (TAccessType[] input, bool[] mask)
This method finds the closest category for a given test input.
- void **Learn** (TAccessType[] input)
This method performs a single training step.

Properties

- decimal **Alpha** [get, set]
Property Alpha represents the choice parameter alpha.
- decimal **Beta_sbm** [get, set]
Property Beta_sbm represents the learning rate of the second best-matching nodes.
- long[] **ClusterNum** [get]
*Property ClusterNum represents the number of **TopoART** clusters found by each module.*
- long[] **NodeNum** [get]
*Property NodeNum represents the number of **TopoART** nodes used by each module.*
- decimal **Rho_a** [get]
*Property Rho_a represents the vigilance parameter of the first **TopoART** module (TA a).*
- string **IntegerType** = Common.Types[(int)integerType] [get]
Property IntegerType returns a string containing the data type used for representing integer variables (IDs, parameters, counters, etc.) internally.
- decimal **FileFormatVersion** [get]
*Property FileFormatVersion returns the version of the file format used by class **TopoART**.*
- string **FloatType** = Common.Types[(int)floatType] [get]
Property FloatType returns a string containing the data type used for representing floating point variables (input, weights, etc.) internally.
- decimal **TopoARTFileFormatVersion** [get]
*Property TopoARTFileFormatVersion returns the version of the file format used by class **TopoART**.*

Properties inherited from **LibTopoART.Network_base**

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- long **LearningSteps** [get]
Property LearningSteps represents the total number of performed learning steps.
- long **ModuleNum** [get]
- long **Phi** [get, set]
- long[] **Phis** [get, set]
- long **Tau** [get, set]
Property Tau represents the parameter tau required for the removal of nodes and edges.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]
Property `InputLen` returns the length of the input vector.
- long **ModuleNum** [get]
- long **LearningSteps** [get]
Property `LearningSteps` represents the total number of performed learning steps.
- long **Tau** [get, set]
Property `Tau` represents the parameter tau required for the removal of nodes and edges.
- long **Phi** [get, set]
- long[] **Phis** [get, set]

Additional Inherited Members**Static Public Attributes inherited from [LibTopoART.Network_base](#)**

- const long **FINAL_MODULE** = [LibTopoART_info.FINAL_MODULE](#)
Instance variable `FINAL_MODULE` gives the value used for indicating that the [TopoART](#) module with the highest index is to be used.

7.50.1 Detailed Description

Class [TopoART](#) provides an implementation of the [TopoART](#) neural network as proposed in "Marko Tscherepanow (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].

7.50.2 Constructor & Destructor Documentation**TopoART() [1/3]**

```
LibTopoART.TopoART.TopoART (
    long inputLen,
    long moduleNum,
    decimal rho_a)
```

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

Parameters

<i>inputLen</i>	The length of input vectors to be learnt.
<i>moduleNum</i>	The number of TopoART modules.
<i>rho_a</i>	The vigilance parameter of the first TopoART module (TA a).

TopoART() [2/3]

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

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

Parameters

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

Exceptions

InvalidFileException	Thrown when the given file cannot be loaded.
--------------------------------------	--

TopoART() [3/3]

```
LibTopoART.TopoART.TopoART (
    Stream stream)
```

This constructor loads a saved [TopoART](#) network from a stream. The stream is left open.

Parameters

<i>stream</i>	A readable <code>Stream</code> containing a network in the binary TopoART file format.
---------------	--

Exceptions

InvalidFileException	Thrown when the given stream cannot be loaded.
--------------------------------------	--

7.50.3 Member Function Documentation**Dispose()**

```
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.

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.

Exceptions

<i>InvalidStateException</i>	Thrown when the network is in an invalid state.
<i>InvalidNumberException</i>	Thrown when the number of edges of an F2 node is greater than <code>int.MaxValue</code> .

Implements [LibTopoART.IAdaptationStateCheck](#).

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.

GetBMOutput() [2/2]

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

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

Parameters

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

GetCategories()

```
List< CategoryInfo >? LibTopoART.TopoART.GetCategories (
    long moduleIndex = FINAL\_MODULE)
```

This method collects information on the categories of a specified module.

Parameters

<i>moduleIndex</i>	The index of the module the categories of which are to be analysed.
--------------------	---

Returns

A list containing information about the respective categories.

Exceptions

<i>InvalidModuleIndexException</i>	Thrown when <i>moduleIndex</i> is invalid.
<i>InvalidNumberException</i>	Thrown when the number of nodes of a module is greater than <code>int.MaxValue</code> .

Implements [LibTopoART.ICategoryAccess](#).

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.
--------------	--------------------------------

Reimplemented in [LibTopoART.Hypersphere_TopopART_C](#), [LibTopoART.TopoART_C](#), and [LibTopoART.TopoART_R](#).

ResetAdaptationState()

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

This method resets the adaptation state to [AdaptationState.NO_ADAPTATION](#).

Exceptions

InvalidNumberException	Thrown when the number of edges of an F2 node is greater than <code>int.MaxValue</code> .
--	---

Implements [LibTopoART.IAdaptationStateCheck](#).

Save() [1/2]

```
void LibTopoART.TopoART.Save (
    Stream stream,
    CompressionLevel compression = CompressionLevel::Fastest)
```

This method saves the entire network to a stream using the binary file format. The stream is left open.

Parameters

<i>stream</i>	A writable <code>Stream</code> the network is saved to.
<i>compression</i>	Compression level of the saved data (Compression is not supported by LibTopoART v0.93 and below.)

Implements [LibTopoART.ITopoART_base_stream](#).

Save() [2/2]

```
void LibTopoART.TopoART.Save (
    string path,
    CompressionLevel compression = CompressionLevel::Fastest)
```

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.
<i>compression</i>	Compression level of the save file (Compression is not supported by LibTopoART v0.93 and below.)

Implements [LibTopoART.ITopoART_base](#).

This method saves the entire network as a text file.

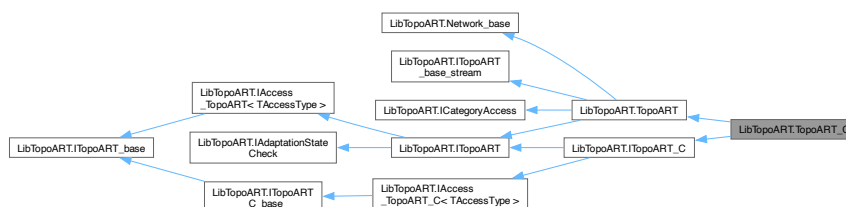
<i>path</i>	A <code>string</code> representing the path of the file to save.
-------------	--

SaveText() [2/2]

<i>writer</i>	A <code>TextWriter</code> the network is saved to.
---------------	--

7.51 LibTopoART.TopoART_C Class Reference

Inheritance diagram for LibTopoART.TopoART_C:



Public Member Functions

- [TopoART_C](#) (long inputLen, long moduleNum, decimal rho_a)
This constructor initialises a TopoART-C network.
- [TopoART_C](#) (string path)
This constructor loads a saved TopoART-C network.
- [TopoART_C](#) (Stream stream)
This constructor loads a saved TopoART-C network from a stream. The stream is left open.
- 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 classID)
This method performs a single training step.
- long [Predict](#) (decimal[] input)
This method predicts the class ID using the default value of nu.
- long [Predict](#) (decimal[] input, long nu)
This method predicts the class ID using a custom value of nu.
- [TopoART_C_prediction Predict](#) (decimal[] input, bool[]? mask)
This method predicts the class ID using the default value of nu.
- [TopoART_C_prediction Predict](#) (decimal[] input, bool[]? mask, long nu)
This method predicts the class ID using a custom value of nu.

Public Member Functions inherited from [LibTopoART.TopoART](#)

- [TopoART](#) (long inputLen, long moduleNum, decimal rho_a)
This constructor initialises a [TopoART](#) network.
- [TopoART](#) (string path)
This constructor loads a saved [TopoART](#) network.
- [TopoART](#) (Stream stream)
This constructor loads a saved [TopoART](#) network from a stream. The stream is left open.
- 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)
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 [SaveText](#) (TextWriter writer)
This method saves the entire network as text to a writer. The writer is flushed but left open.
- void [Save](#) (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void [Save](#) (Stream stream, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network to a stream using the binary file format. The stream is left open.
- 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.
- List< [CategoryInfo](#) >? [GetCategories](#) (long moduleIndex=FINAL_MODULE)
This method collects information on the categories of a specified module.

Public Member Functions inherited from [LibTopoART.IAccess_TopoART< TAccessType >](#)

- [F2_output\[\] GetBMOutput](#) (TAccessType[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (TAccessType[] input, bool[] mask)
This method finds the closest category for a given test input.
- void [Learn](#) (TAccessType[] input)
This method performs a single training step.

Public Member Functions inherited from [LibTopoART.IAccess_TopoART_C< TAccessType >](#)

- void [Learn](#) (TAccessType[] input, long classID)
This method performs a single training step.
- long [Predict](#) (TAccessType[] input)
This method predicts the class ID using the default value of nu.
- long [Predict](#) (TAccessType[] input, long nu)
This method predicts the class ID using a custom value of nu.
- [TopoART_C_prediction Predict](#) (TAccessType[] input, bool[] mask)
This method predicts the class ID using the default value of nu.
- [TopoART_C_prediction Predict](#) (TAccessType[] input, bool[] mask, long nu)
This method predicts the class ID using a custom value of nu.

Static Public Attributes

- const long **UNDEFINED_CLASS_ID** = -2
Instance variable `UNDEFINED_CLASS_ID` gives the value used for indicating that an input sample was predicted to belong to the undefined class; i.e., no class ID was provided for such input samples during training.

Static Public Attributes inherited from [LibTopoART.Network_base](#)

- const long **FINAL_MODULE** = [LibTopoART_info.FINAL_MODULE](#)
Instance variable `FINAL_MODULE` gives the value used for indicating that the [TopoART](#) module with the highest index is to be used.

Properties

- new decimal **FileFormatVersion** [get]
Property `FileFormatVersion` returns the version of the file format used by class [TopoART_C](#).
- long **Nu** = Common.TopoART_C_default_nu [get, set]
Property `Nu` represents the default value used for the maximum cardinality of the set of enclosing categories E and the neighbourhood set N during prediction. If the parameter `nu` is not explicitly provided for prediction, this property will be applied. (This parameter does not modify the network. It may be arbitrarily changed for each prediction step.).
- bool **SkipEdgeLearning** [get, set]
Property `SkipEdgeLearning` enables/disables the [TopoART](#) edge learning mechanism. If the topology of the input data is not required, disabling edge learning may decrease the processing time needed for training.

Properties inherited from [LibTopoART.TopoART](#)

- decimal **Alpha** [get, set]
Property Alpha represents the choice parameter alpha.
- decimal **Beta_sbm** [get, set]
Property Beta_sbm represents the learning rate of the second best-matching nodes.
- long[] **ClusterNum** [get]
Property ClusterNum represents the number of [TopoART](#) clusters found by each module.
- long[] **NodeNum** [get]
Property NodeNum represents the number of [TopoART](#) nodes used by each module.
- decimal **Rho_a** [get]
Property Rho_a represents the vigilance parameter of the first [TopoART](#) module (TA a).
- string **IntegerType** = Common.Types[(int)integerType] [get]
Property IntegerType returns a string containing the data type used for representing integer variables (IDs, parameters, counters, etc.) internally.
- decimal **FileFormatVersion** [get]
Property FileFormatVersion returns the version of the file format used by class [TopoART](#).
- string **FloatType** = Common.Types[(int)floatType] [get]
Property FloatType returns a string containing the data type used for representing floating point variables (input, weights, etc.) internally.
- decimal **TopoARTFileFormatVersion** [get]
Property TopoARTFileFormatVersion returns the version of the file format used by class [TopoART](#).

Properties inherited from [LibTopoART.Network_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- long **LearningSteps** [get]
Property LearningSteps represents the total number of performed learning steps.
- long **ModuleNum** [get]
- long **Phi** [get, set]
- long[] **Phis** [get, set]
- long **Tau** [get, set]
Property Tau represents the parameter tau required for the removal of nodes and edges.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]
Property InputLen returns the length of the input vector.
- long **ModuleNum** [get]
- long **LearningSteps** [get]
Property LearningSteps represents the total number of performed learning steps.
- long **Tau** [get, set]
Property Tau represents the parameter tau required for the removal of nodes and edges.
- long **Phi** [get, set]
- long[] **Phis** [get, set]

7.51.1 Detailed Description

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

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

7.51.2 Constructor & Destructor Documentation

TopoART_C() [1/3]

```
LibTopoART.TopoART_C.TopoART_C (
    long  inputLen,
    long  moduleNum,
    decimal rho_a)
```

This constructor initialises a TopoART-C network.

Parameters

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

TopoART_C() [2/3]

```
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.
-------------	--------------------------------------

Exceptions

InvalidFileException	Thrown when the given file cannot be loaded.
--------------------------------------	--

TopoART_C() [3/3]

```
LibTopoART.TopoART_C.TopoART_C (
    Stream stream)
```

This constructor loads a saved TopoART-C network from a stream. The stream is left open.

Parameters

<i>stream</i>	A readable <code>Stream</code> containing a network in the binary TopoART-C file format.
---------------	--

Exceptions

<i>InvalidFileException</i>	Thrown when the given stream cannot be loaded.
---	--

7.51.3 Member Function Documentation**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](#).

Learn() [2/2]

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

This method performs a single training step.

Parameters

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

Exceptions

InvalidClassIDException	Thrown when <i>classID</i> is less than 0.
---	--

Predict() [1/4]

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

This method predicts the class ID using the default value of nu.

Parameters

<i>input</i>	The input vector the class ID of which is to be predicted.
--------------	--

Returns

The predicted class ID.

Predict() [2/4]

```
TopoART_C_prediction LibTopoART.TopoART_C.Predict (  
    decimal[] input,  
    bool?[] mask)
```

This method predicts the class ID using the default value of nu.

Parameters

<i>input</i>	The input vector the class ID of which is to be predicted.
<i>mask</i>	The mask vector corresponding to <i>input</i> .

Returns

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

Predict() [3/4]

```
TopoART_C_prediction LibTopoART.TopoART_C.Predict (  
    decimal[] input,  
    bool?[] mask,  
    long nu)
```

This method predicts the class ID using a custom value of nu.

Parameters

<i>input</i>	The input vector the class ID of which is to be predicted.
<i>mask</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.

Predict() [4/4]

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

This method predicts the class ID using a custom value of nu.

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.

7.52 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 classID, decimal confidence)

This constructor sets the instance variables `classID` and `confidence` of struct [TopoART_C_prediction](#).

Public Attributes

- readonly long **classID**

Instance variable `classID` gives the predicted class ID.

- readonly decimal **confidence**

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

7.52.1 Detailed Description

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

7.52.2 Constructor & Destructor Documentation

TopoART_C_prediction()

```
LibTopoART.TopoART_C_prediction.TopoART_C_prediction (
    long classID,
    decimal confidence)
```

This constructor sets the instance variables `classID` and `confidence` of struct [TopoART_C_prediction](#).

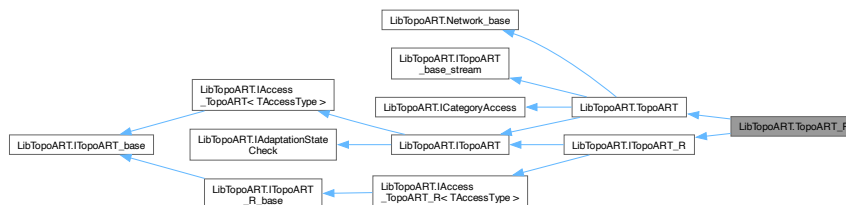
Parameters

<i>classID</i>	The class ID to be set.
<i>confidence</i>	The value of the confidence to be set.

7.53 LibTopoART.TopoART_R Class Reference

Class [TopoART_R](#) provides an implementation of the TopoART-R neural network as proposed in "Marko Tscherepanow (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 iLen, long dLen, long moduleNum, decimal rho_a)
This constructor initialises a TopoART-R network.
- [TopoART_R](#) (string path)
This constructor loads a saved TopoART-R network.
- [TopoART_R](#) (Stream stream)
This constructor loads a saved TopoART-R network from a stream. The stream is left open.
- 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[] input, decimal[] output)

- *This method performs a single training step.*
- decimal[] [Predict](#) (decimal[] input)
 - This method predicts the dependent variables using the default value of nu.*
- decimal[] [Predict](#) (decimal[] input, long nu)
 - This method predicts the dependent variables using a custom value of nu.*
- TopoART_R_prediction< decimal > [Predict](#) (decimal[] input, bool[] mask)
 - This method predicts the dependent variables for a given set of independent variables using the default value of nu. Unknown values of independent variables can be signified by setting the corresponding value of mask to true.*
- TopoART_R_prediction< decimal > [Predict](#) (decimal[] input, bool[] mask, long nu)
 - This method predicts the dependent variables for a given set of independent variables using a custom value of nu. Unknown values of independent variables can be signified by setting the corresponding value of mask to true.*

Public Member Functions inherited from [LibTopoART.TopoART](#)

- [TopoART](#) (long inputLen, long moduleNum, decimal rho_a)
 - This constructor initialises a [TopoART](#) network.*
- [TopoART](#) (string path)
 - This constructor loads a saved [TopoART](#) network.*
- [TopoART](#) (Stream stream)
 - This constructor loads a saved [TopoART](#) network from a stream. The stream is left open.*
- 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)
 - 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 [SaveText](#) (TextWriter writer)
 - This method saves the entire network as text to a writer. The writer is flushed but left open.*
- void [Save](#) (string path, CompressionLevel compression=CompressionLevel.Fastest)
 - This method saves the entire network as a binary file.*
- void [Save](#) (Stream stream, CompressionLevel compression=CompressionLevel.Fastest)
 - This method saves the entire network to a stream using the binary file format. The stream is left open.*
- 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.*
- List< [CategoryInfo](#) >? [GetCategories](#) (long moduleIndex=FINAL_MODULE)
 - This method collects information on the categories of a specified module.*

Public Member Functions inherited from [LibTopoART.IAccess_TopoART](#)< [TAccessType](#) >

- F2_output[] [GetBMOutput](#) ([TAccessType](#)[] input)
 - This method finds the closest category for a given test input.*
- F2_output[] [GetBMOutput](#) ([TAccessType](#)[] input, bool[] mask)
 - This method finds the closest category for a given test input.*
- void [Learn](#) ([TAccessType](#)[] input)
 - This method performs a single training step.*

Public Member Functions inherited from [LibTopoART.IAccess_TopoART_R< TAccessType >](#)

- void [Learn](#) (TAccessType[] input, TAccessType[] output)
This method performs a single training step.
- TAccessType[] [Predict](#) (TAccessType[] input)
This method predicts the dependent variables using the default value of nu.
- TAccessType[] [Predict](#) (TAccessType[] input, long nu)
This method predicts the dependent variables using a custom value of nu.
- TopoART_R_prediction< TAccessType > [Predict](#) (TAccessType[] input, bool[] mask)
This method predicts the dependent variables for a given set of independent variables using the default value of nu. Unknown values of independent variables can be signified by setting the corresponding value of mask to true.
- TopoART_R_prediction< TAccessType > [Predict](#) (TAccessType[] input, bool[] mask, long nu)
This method predicts the dependent variables for a given set of independent variables using a custom value of nu. Unknown values of independent variables can be signified by setting the corresponding value of mask to true.

Properties

- long **D_len** [get]
Property D_len returns the length of the output vector (dependent variables).
- new decimal **FileFormatVersion** [get]
Property FileFormatVersion returns the version of the file format used by class [TopoART_R](#).
- long **I_len** [get]
Property I_len returns the length of the input vector (independent variables).
- long **Nu** = Common.TopoART_R_default_nu [get, set]
Property Nu represents the default value used for the maximum cardinality of the neighbourhood set N during prediction. If the parameter nu is not explicitly provided for prediction, this property will be applied. (This parameter does not modify the network. It may be arbitrarily changed for each prediction step.).
- bool **SkipEdgeLearning** [get, set]
Property SkipEdgeLearning enables/disables the [TopoART](#) edge learning mechanism. If the topology of the input data is not required, disabling edge learning may decrease the processing time needed for training.

Properties inherited from [LibTopoART.TopoART](#)

- decimal **Alpha** [get, set]
Property Alpha represents the choice parameter alpha.
- decimal **Beta_sbm** [get, set]
Property Beta_sbm represents the learning rate of the second best-matching nodes.
- long[] **ClusterNum** [get]
Property ClusterNum represents the number of [TopoART](#) clusters found by each module.
- long[] **NodeNum** [get]
Property NodeNum represents the number of [TopoART](#) nodes used by each module.
- decimal **Rho_a** [get]
Property Rho_a represents the vigilance parameter of the first [TopoART](#) module (TA a).
- string **IntegerType** = Common.Types[(int)integerType] [get]
Property IntegerType returns a string containing the data type used for representing integer variables (IDs, parameters, counters, etc.) internally.
- decimal **FileFormatVersion** [get]
Property FileFormatVersion returns the version of the file format used by class [TopoART](#).
- string **FloatType** = Common.Types[(int)floatType] [get]
Property FloatType returns a string containing the data type used for representing floating point variables (input, weights, etc.) internally.
- decimal **TopoARTFileFormatVersion** [get]
Property TopoARTFileFormatVersion returns the version of the file format used by class [TopoART](#).

Properties inherited from [LibTopoART.Network_base](#)

- long **InputLen** [get]
Property `InputLen` returns the length of the input vector.
- long **LearningSteps** [get]
Property `LearningSteps` represents the total number of performed learning steps.
- long **ModuleNum** [get]
- long **Phi** [get, set]
- long[] **Phis** [get, set]
- long **Tau** [get, set]
Property `Tau` represents the parameter tau required for the removal of nodes and edges.

Properties inherited from [LibTopoART.ITopoART_base](#)

- long **InputLen** [get]
Property `InputLen` returns the length of the input vector.
- long **ModuleNum** [get]
- long **LearningSteps** [get]
Property `LearningSteps` represents the total number of performed learning steps.
- long **Tau** [get, set]
Property `Tau` represents the parameter tau required for the removal of nodes and edges.
- long **Phi** [get, set]
- long[] **Phis** [get, set]

Additional Inherited Members**Static Public Attributes inherited from [LibTopoART.Network_base](#)**

- const long **FINAL_MODULE** = [LibTopoART_info.FINAL_MODULE](#)
Instance variable `FINAL_MODULE` gives the value used for indicating that the [TopoART](#) module with the highest index is to be used.

7.53.1 Detailed Description

Class [TopoART_R](#) provides an implementation of the TopoART-R neural network as proposed in "Marko Tscherepanow (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].

7.53.2 Constructor & Destructor Documentation

TopoART_R() [1/3]

```
LibTopoART.TopoART_R.TopoART_R (
    long iLen,
    long dLen,
    long moduleNum,
    decimal rho_a)
```

This constructor initialises a TopoART-R network.

Parameters

<i>iLen</i>	The length of the input vector (independent variables) to be learnt.
<i>dLen</i>	The length of the output vector (dependent variables) to be learnt.
<i>moduleNum</i>	The number of TopoART-R modules.
<i>rho_a</i>	The vigilance parameter of the first TopoART-R module (TopoART-R a).

TopoART_R() [2/3]

```
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.
-------------	--------------------------------------

Exceptions

<i>InvalidFileException</i>	Thrown when the given file cannot be loaded.
---	--

TopoART_R() [3/3]

```
LibTopoART.TopoART_R.TopoART_R (
    Stream stream)
```

This constructor loads a saved TopoART-R network from a stream. The stream is left open.

Parameters

<i>stream</i>	A readable <code>Stream</code> containing a network in the binary TopoART-R file format.
---------------	--

Exceptions

<i>InvalidFileException</i>	Thrown when the given stream cannot be loaded.
---	--

7.53.3 Member Function Documentation**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](#).

Learn() [2/2]

```
void LibTopoART.TopoART_R.Learn (
    decimal[] input,
    decimal[] output)
```

This method performs a single training step.

Parameters

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

Predict() [1/4]

```
decimal[] LibTopoART.TopoART_R.Predict (
    decimal[] input)
```

This method predicts the dependent variables using the default value of nu.

Parameters

<i>input</i>	The input vector (independent variables).
--------------	---

Returns

The predicted values for all dependent variables. If no prediction is possible (e.g., for an untrained network), all values are set to the value `NO_PREDICTION` of struct `TopoART_R_prediction`.

Predict() [2/4]

```
TopoART_R_prediction< decimal > LibTopoART.TopoART_R.Predict (
    decimal[] input,
    bool[] mask)
```

This method predicts the dependent variables for a given set of independent variables using the default value of `nu`. Unknown values of independent variables can be signified by setting the corresponding value of `mask` to `true`.

Parameters

<i>input</i>	The input vector (independent variables).
<i>mask</i>	The mask vector corresponding to <i>input</i> .

Returns

An object of type `TopoART_R_prediction` containing the predicted values for the unknown independent variables and all dependent variables. If no prediction is possible (e.g., for an untrained network), all values are set to `NO_PREDICTION`.

Predict() [3/4]

```
TopoART_R_prediction< decimal > LibTopoART.TopoART_R.Predict (
    decimal[] input,
    bool[] mask,
    long nu)
```

This method predicts the dependent variables for a given set of independent variables using a custom value of `nu`. Unknown values of independent variables can be signified by setting the corresponding value of `mask` to `true`.

Parameters

<i>input</i>	The input vector (independent variables).
<i>mask</i>	The mask vector corresponding to <i>input</i> .
<i>nu</i>	The maximum cardinality of the neighbourhood set N. (In the original TopoART-R network, <code>nu</code> 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. If no prediction is possible (e.g., for an untrained network), all values are set to `NO_PREDICTION`.

Predict() [4/4]

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

This method predicts the dependent variables using a custom value of nu.

Parameters

<i>input</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. If no prediction is possible (e.g., for an untrained network), all values are set to the value `NO_PREDICTION` of struct `TopoART_R_prediction`.

7.54 LibTopoART.TopoART_R_prediction< TElementType > Struct Template Reference

Struct `TopoART_R_prediction` contains a prediction made by a TopoART-R network.

Public Member Functions

- [TopoART_R_prediction](#) (TElementType[] iVecPrediction, TElementType[] dVecPrediction)
This constructor sets the instance variables `i_vec_prediction` and `d_vec_prediction` of struct `TopoART_R_prediction`.
- [TopoART_R_prediction](#) (long iLen, long dLen)
This constructor initialises the instance variables `i_vec_prediction` and `d_vec_prediction` of struct `TopoART_R_prediction` with arrays of the given lengths in which all elements are set to `NO_PREDICTION`.
- void **PrintPredictions** ()
This method prints the predictions on the console.

Public Attributes

- readonly TElementType **NO_PREDICTION**
Instance variable `NO_PREDICTION` provides a default prediction for variables that are presented to the network; i.e., these variables are known and no prediction is computed for them. ATTENTION: `NO_PREDICTION` may be ambiguous depending on `TElementType`.
- readonly TElementType[] **i_vec_prediction**
Instance variable `i_vec_prediction` represents predictions for unknown independent variables.
- readonly TElementType[] **d_vec_prediction**
Instance variable `d_vec_prediction` provides the predictions for the dependent variables.

7.54.1 Detailed Description

Struct `TopoART_R_prediction` contains a prediction made by a TopoART-R network.

Type Constraints

***TElementType* : struct**

***TElementType* : IConvertible**

7.54.2 Member Function Documentation

TopoART_R_prediction() [1/2]

```
LibTopoART.TopoART_R_prediction< TElementType >.TopoART_R_prediction (
    long iLen,
    long dLen)
```

This constructor initialises the instance variables `i_vec_prediction` and `d_vec_prediction` of struct `TopoART_R_prediction` with arrays of the given lengths in which all elements are set to `NO_PREDICTION`.

Parameters

<i>iLen</i>	The length of the input vector (independent variables).
<i>dLen</i>	The length of the output vector (dependent variables).

TopoART_R_prediction() [2/2]

```
LibTopoART.TopoART_R_prediction< TElementType >.TopoART_R_prediction (
    TElementType[] iVecPrediction,
    TElementType[] dVecPrediction)
```

This constructor sets the instance variables `i_vec_prediction` and `d_vec_prediction` of struct `TopoART_R_prediction`.

Parameters

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

Index

- AdaptationState
 - LibTopoART, [13](#)
- ADAPTED_NONPERMANENT_WEIGHT
 - LibTopoART, [13](#)
- ADAPTED_PERMANENT_WEIGHT
 - LibTopoART, [13](#)
- ADDED_EDGE_CANDIDATE
 - LibTopoART, [13](#)
- ADDED_NODE_CANDIDATE
 - LibTopoART, [13](#)
- ADDED_PERMANENT_EDGE
 - LibTopoART, [13](#)
- ADDED_PERMANENT_NODE
 - LibTopoART, [13](#)
- ANY_NONPERMANENT_ADAPTATION_MASK
 - LibTopoART, [13](#)
- ANY_PERMANENT_ADAPTATION_MASK
 - LibTopoART, [13](#)
- BeginRecall
 - LibTopoART.Fast_Episodic_TopoART, [20](#)
 - LibTopoART.IAccessEpisodicRecall< TAccessType >, [94](#)
- BeginRecallKey1
 - LibTopoART.Fast_TopoART_AM, [32](#)
 - LibTopoART.IAccessAssociativeRecall< TAccessType >, [92](#)
- BeginRecallKey2
 - LibTopoART.Fast_TopoART_AM, [33](#)
 - LibTopoART.IAccessAssociativeRecall< TAccessType >, [92](#)
- CategoryInfo
 - LibTopoART.CategoryInfo, [14](#)
- Dispose
 - LibTopoART.Fast_TopoART_base, [39](#)
 - LibTopoART.TopoART, [140](#)
- Fast_Episodic_TopoART
 - LibTopoART.Fast_Episodic_TopoART, [19](#)
- Fast_TopoART
 - LibTopoART.Fast_TopoART, [25](#), [26](#)
- Fast_TopoART_AM
 - LibTopoART.Fast_TopoART_AM, [31](#), [32](#)
- Fast_TopoART_C
 - LibTopoART.Fast_TopoART_C, [49](#)
- Fast_TopoART_R
 - LibTopoART.Fast_TopoART_R, [59](#)
- GetAdaptationState
 - LibTopoART.Fast_TopoART_base, [39](#)
 - LibTopoART.IAdaptationStateCheck, [96](#)
 - LibTopoART.TopoART, [140](#)
- GetBMOutput
 - LibTopoART.Fast_TopoART_AM, [34](#)
- LibTopoART.Fast_TopoART_base, [39](#), [40](#)
- LibTopoART.IAccess_TopoART< TAccessType >, [78](#)
- LibTopoART.IAccess_TopoART_AM< TAccessType >, [82](#)
- LibTopoART.TopoART, [141](#)
- GetCategories
 - LibTopoART.Fast_TopoART_base, [41](#)
 - LibTopoART.ICategoryAccess, [98](#)
 - LibTopoART.TopoART, [142](#)
- Hypersphere_TopoART
 - LibTopoART.Hypersphere_TopoART, [67](#), [68](#)
- Hypersphere_TopoART_C
 - LibTopoART.Hypersphere_TopoART_C, [73](#), [74](#)
- Important
 - LibTopoART, [13](#)
- InterEpisodeRecallStep
 - LibTopoART.Fast_Episodic_TopoART, [20](#), [21](#)
 - LibTopoART.IAccessEpisodicRecall< TAccessType >, [94](#)
- IntraEpisodeRecallStep
 - LibTopoART.Fast_Episodic_TopoART, [21](#)
 - LibTopoART.IAccessEpisodicRecall< TAccessType >, [95](#)
- Learn
 - LibTopoART.Fast_Episodic_TopoART, [22](#)
 - LibTopoART.Fast_TopoART, [26](#), [27](#)
 - LibTopoART.Fast_TopoART_AM, [35](#)
 - LibTopoART.Fast_TopoART_base, [41](#), [42](#)
 - LibTopoART.Fast_TopoART_C, [50](#), [51](#)
 - LibTopoART.Fast_TopoART_R, [60](#)
 - LibTopoART.Hypersphere_TopoART_C, [74](#), [75](#)
 - LibTopoART.IAccess_TopoART< TAccessType >, [79](#)
 - LibTopoART.IAccess_TopoART_AM< TAccessType >, [82](#)
 - LibTopoART.IAccess_TopoART_C< TAccessType >, [85](#)
 - LibTopoART.IAccess_TopoART_R< TAccessType >, [89](#)
 - LibTopoART.TopoART, [142](#)
 - LibTopoART.TopoART_C, [149](#)
 - LibTopoART.TopoART_R, [157](#)
- LibTopoART, [9](#)
 - AdaptationState, [13](#)
 - ADAPTED_NONPERMANENT_WEIGHT, [13](#)
 - ADAPTED_PERMANENT_WEIGHT, [13](#)
 - ADDED_EDGE_CANDIDATE, [13](#)
 - ADDED_NODE_CANDIDATE, [13](#)
 - ADDED_PERMANENT_EDGE, [13](#)
 - ADDED_PERMANENT_NODE, [13](#)
 - ANY_NONPERMANENT_ADAPTATION_MASK, [13](#)

- ANY_PERMANENT_ADAPTATION_MASK, 13
- Important, 13
- NO_ADAPTATION, 13
- Normal, 13
- REMOVED_EDGE_CANDIDATE, 13
- REMOVED_NODE_CANDIDATE, 13
- Verbose, 13
- VerbosityLevel, 13
- LibTopoART.DirectoryReference, 9
- LibTopoART.CategoryInfo, 13
 - CategoryInfo, 14
- LibTopoART.F2_output, 14
- LibTopoART.Fast_Episodic_TopoART, 15
 - BeginRecall, 20
 - Fast_Episodic_TopoART, 19
 - InterEpisodeRecallStep, 20, 21
 - IntraEpisodeRecallStep, 21
 - Learn, 22
- LibTopoART.Fast_TopoART, 23
 - Fast_TopoART, 25, 26
 - Learn, 26, 27
- LibTopoART.Fast_TopoART_AM, 27
 - BeginRecallKey1, 32
 - BeginRecallKey2, 33
 - Fast_TopoART_AM, 31, 32
 - GetBMOOutput, 34
 - Learn, 35
 - RecallStep, 35, 36
- LibTopoART.Fast_TopoART_base, 36
 - Dispose, 39
 - GetAdaptationState, 39
 - GetBMOOutput, 39, 40
 - GetCategories, 41
 - Learn, 41, 42
 - ResetAdaptationState, 42
 - Save, 42, 43
 - SaveText, 44
- LibTopoART.Fast_TopoART_C, 44
 - Fast_TopoART_C, 49
 - Learn, 50, 51
 - Predict, 51–54
- LibTopoART.Fast_TopoART_R, 54
 - Fast_TopoART_R, 59
 - Learn, 60
 - Predict, 61–64
- LibTopoART.Hypersphere_TopoART, 64
 - Hypersphere_TopoART, 67, 68
- LibTopoART.Hypersphere_TopoART_C, 69
 - Hypersphere_TopoART_C, 73, 74
 - Learn, 74, 75
 - Predict, 75, 76
- LibTopoART.IAccess_TopoART< TAccessType >, 77
 - GetBMOOutput, 78
 - Learn, 79
- LibTopoART.IAccess_TopoART_AM< TAccessType >, 79
 - GetBMOOutput, 82
 - Learn, 82
- LibTopoART.IAccess_TopoART_C< TAccessType >, 82
 - Learn, 85
 - Predict, 85, 86
- LibTopoART.IAccess_TopoART_R< TAccessType >, 86
 - Learn, 89
 - Predict, 89, 90
- LibTopoART.IAccessAssociativeRecall< TAccessType >, 90
 - BeginRecallKey1, 92
 - BeginRecallKey2, 92
 - RecallStep, 92
- LibTopoART.IAccessEpisodicRecall< TAccessType >, 93
 - BeginRecall, 94
 - InterEpisodeRecallStep, 94
 - IntraEpisodeRecallStep, 95
- LibTopoART.IAdaptationStateCheck, 95
 - GetAdaptationState, 96
- LibTopoART.IAssociativeRecall, 96
- LibTopoART.ICategoryAccess, 98
 - GetCategories, 98
- LibTopoART.IEndRecall, 99
- LibTopoART.IEpisodic_TopoART, 99
- LibTopoART.IEpisodicRecall, 101
- LibTopoART.IFast_Episodic_TopoART, 103
- LibTopoART.IFast_TopoART, 105
- LibTopoART.IFast_TopoART_AM, 106
- LibTopoART.IFast_TopoART_C, 108
- LibTopoART.IFast_TopoART_R, 110
- LibTopoART.IFastAssociativeRecall, 113
- LibTopoART.IFastEpisodicRecall, 114
- LibTopoART.IHypersphere_TopoART, 115
- LibTopoART.InvalidClassIDException, 117
- LibTopoART.InvalidFileException, 118
- LibTopoART.InvalidModuleIndexException, 118
- LibTopoART.InvalidNumberException, 118
- LibTopoART.InvalidSizeException, 118
- LibTopoART.InvalidStateException, 118
- LibTopoART.InvalidTypeException, 119
- LibTopoART.ITopoART, 119
- LibTopoART.ITopoART_AM, 120
- LibTopoART.ITopoART_AM_base, 122
- LibTopoART.ITopoART_base, 124
 - ModuleNum, 126
 - Phi, 126
 - Phis, 126
 - Save, 125
 - SaveText, 125
- LibTopoART.ITopoART_base_stream, 126
 - Save, 127
 - SaveText, 127
- LibTopoART.ITopoART_C, 128
- LibTopoART.ITopoART_C_base, 129
- LibTopoART.ITopoART_R, 131
- LibTopoART.ITopoART_R_base, 133
- LibTopoART.LibTopoART_control, 135
- LibTopoART.LibTopoART_info, 135
- LibTopoART.Network_base, 135

- ModuleNum, 136
- Phi, 136
- Phis, 136
- LibTopoART.TopoART, 137
 - Dispose, 140
 - GetAdaptationState, 140
 - GetBMOOutput, 141
 - GetCategories, 142
 - Learn, 142
 - ResetAdaptationState, 142
 - Save, 143
 - SaveText, 143, 144
 - TopoART, 139, 140
- LibTopoART.TopoART_C, 144
 - Learn, 149
 - Predict, 150, 151
 - TopoART_C, 148
- LibTopoART.TopoART_C_prediction, 151
 - TopoART_C_prediction, 152
- LibTopoART.TopoART_R, 152
 - Learn, 157
 - Predict, 157, 158
 - TopoART_R, 156
- LibTopoART.TopoART_R_prediction< TElementType >, 159
 - TopoART_R_prediction, 160
- ModuleNum
 - LibTopoART.ITopoART_base, 126
 - LibTopoART.Network_base, 136
- NO_ADAPTATION
 - LibTopoART, 13
- Normal
 - LibTopoART, 13
- Phi
 - LibTopoART.ITopoART_base, 126
 - LibTopoART.Network_base, 136
- Phis
 - LibTopoART.ITopoART_base, 126
 - LibTopoART.Network_base, 136
- Predict
 - LibTopoART.Fast_TopoART_C, 51–54
 - LibTopoART.Fast_TopoART_R, 61–64
 - LibTopoART.Hypersphere_TopoART_C, 75, 76
 - LibTopoART.IAccess_TopoART_C< TAccessType >, 85, 86
 - LibTopoART.IAccess_TopoART_R< TAccessType >, 89, 90
 - LibTopoART.TopoART_C, 150, 151
 - LibTopoART.TopoART_R, 157, 158
- RecallStep
 - LibTopoART.Fast_TopoART_AM, 35, 36
 - LibTopoART.IAccessAssociativeRecall< TAccessType >, 92
- REMOVED_EDGE_CANDIDATE
 - LibTopoART, 13
- REMOVED_NODE_CANDIDATE
 - LibTopoART, 13
- ResetAdaptationState
 - LibTopoART.Fast_TopoART_base, 42
 - LibTopoART.TopoART, 142
- Save
 - LibTopoART.Fast_TopoART_base, 42, 43
 - LibTopoART.ITopoART_base, 125
 - LibTopoART.ITopoART_base_stream, 127
 - LibTopoART.TopoART, 143
- SaveText
 - LibTopoART.Fast_TopoART_base, 44
 - LibTopoART.ITopoART_base, 125
 - LibTopoART.ITopoART_base_stream, 127
 - LibTopoART.TopoART, 143, 144
- TopoART
 - LibTopoART.TopoART, 139, 140
- TopoART_C
 - LibTopoART.TopoART_C, 148
- TopoART_C_prediction
 - LibTopoART.TopoART_C_prediction, 152
- TopoART_R
 - LibTopoART.TopoART_R, 156
- TopoART_R_prediction
 - LibTopoART.TopoART_R_prediction< TElementType >, 160
- Verbose
 - LibTopoART, 13
- VerbosityLevel
 - LibTopoART, 13