

LibTopoART 0.97 Reference Manual

Generated by Doxygen 1.9.6

1 Namespace Index	1
1.1 Package List	1
2 Hierarchical Index	1
2.1 Class Hierarchy	1
3 Class Index	5
3.1 Class List	5
4 Namespace Documentation	9
4.1 LibTopoART Namespace Reference	9
4.1.1 Enumeration Type Documentation	12
4.2 LibTopoART_samples Namespace Reference	13
5 Class Documentation	14
5.1 LibTopoART.CategoryInfo Struct Reference	14
5.1.1 Detailed Description	14
5.1.2 Constructor & Destructor Documentation	14
5.2 LibTopoART.F2_output Class Reference	15
5.2.1 Detailed Description	15
5.3 LibTopoART.Fast_Episodic_TopoART Class Reference	16
5.3.1 Detailed Description	20
5.3.2 Constructor & Destructor Documentation	20
5.3.3 Member Function Documentation	20
5.4 LibTopoART.Fast_TopoART Class Reference	23
5.4.1 Detailed Description	26
5.4.2 Constructor & Destructor Documentation	27
5.4.3 Member Function Documentation	27
5.5 LibTopoART.Fast_TopoART_AM Class Reference	28
5.5.1 Detailed Description	32
5.5.2 Constructor & Destructor Documentation	32
5.5.3 Member Function Documentation	33
5.6 LibTopoART.Fast_TopoART_base Class Reference	38
5.6.1 Detailed Description	40
5.6.2 Member Function Documentation	40
5.7 LibTopoART.Fast_TopoART_C Class Reference	44
5.7.1 Detailed Description	49
5.7.2 Constructor & Destructor Documentation	49
5.7.3 Member Function Documentation	50
5.8 LibTopoART.Fast_TopoART_R Class Reference	54
5.8.1 Detailed Description	59
5.8.2 Constructor & Destructor Documentation	59
5.8.3 Member Function Documentation	60
5.9 LibTopoART.Hypersphere_TopoART Class Reference	64

5.9.1 Detailed Description	67
5.9.2 Constructor & Destructor Documentation	67
5.10 LibTopoART.Hypersphere_TopoART_C Class Reference	68
5.10.1 Detailed Description	72
5.10.2 Constructor & Destructor Documentation	73
5.10.3 Member Function Documentation	74
5.11 LibTopoART.IAccess_TopoART< _AccessType > Interface Template Reference	76
5.11.1 Detailed Description	77
5.11.2 Member Function Documentation	77
5.12 LibTopoART.IAccess_TopoART_AM< _AccessType > Interface Template Reference	78
5.12.1 Detailed Description	80
5.12.2 Member Function Documentation	80
5.13 LibTopoART.IAccess_TopoART_C< _AccessType > Interface Template Reference	81
5.13.1 Detailed Description	83
5.13.2 Member Function Documentation	83
5.14 LibTopoART.IAccess_TopoART_R< _AccessType > Interface Template Reference	85
5.14.1 Detailed Description	86
5.14.2 Member Function Documentation	86
5.15 LibTopoART.IAccessAssociativeRecall< _AccessType > Interface Template Reference	88
5.15.1 Detailed Description	89
5.15.2 Member Function Documentation	89
5.16 LibTopoART.IAccessEpisodicRecall< _AccessType > Interface Template Reference	91
5.16.1 Detailed Description	91
5.16.2 Member Function Documentation	92
5.17 LibTopoART.IAdaptationStateCheck Interface Reference	93
5.17.1 Detailed Description	93
5.17.2 Member Function Documentation	93
5.18 LibTopoART.IAssociativeRecall Interface Reference	94
5.18.1 Detailed Description	95
5.19 LibTopoART.ICategoryAccess Interface Reference	95
5.19.1 Detailed Description	95
5.19.2 Member Function Documentation	95
5.20 LibTopoART.IEndRecall Interface Reference	96
5.20.1 Detailed Description	96
5.21 LibTopoART.IEpisodic_TopoART Interface Reference	96
5.21.1 Detailed Description	98
5.22 LibTopoART.IEpisodicRecall Interface Reference	98
5.22.1 Detailed Description	99
5.23 LibTopoART.IFast_Episodic_TopoART Interface Reference	99
5.23.1 Detailed Description	100
5.24 LibTopoART.IFast_TopoART Interface Reference	101
5.24.1 Detailed Description	101

5.25 LibTopoART.IFast_TopoART_AM Interface Reference	102
5.25.1 Detailed Description	103
5.26 LibTopoART.IFast_TopoART_C Interface Reference	103
5.26.1 Detailed Description	105
5.27 LibTopoART.IFast_TopoART_R Interface Reference	105
5.27.1 Detailed Description	107
5.28 LibTopoART.IFastAssociativeRecall Interface Reference	107
5.28.1 Detailed Description	108
5.29 LibTopoART.IFastEpisodicRecall Interface Reference	108
5.29.1 Detailed Description	110
5.30 LibTopoART.IHypersphere_TopoART Interface Reference	110
5.30.1 Detailed Description	111
5.31 LibTopoART.InvalidClassIDException Class Reference	111
5.31.1 Detailed Description	111
5.32 LibTopoART.InvalidFileException Class Reference	111
5.32.1 Detailed Description	111
5.33 LibTopoART.InvalidModuleIndexException Class Reference	111
5.33.1 Detailed Description	111
5.34 LibTopoART.InvalidNumberException Class Reference	112
5.34.1 Detailed Description	112
5.35 LibTopoART.InvalidSizeException Class Reference	112
5.35.1 Detailed Description	112
5.36 LibTopoART.InvalidStateException Class Reference	112
5.36.1 Detailed Description	112
5.37 LibTopoART.InvalidTypeException Class Reference	112
5.37.1 Detailed Description	112
5.38 LibTopoART.ITopoART Interface Reference	113
5.38.1 Detailed Description	113
5.39 LibTopoART.ITopoART_AM Interface Reference	114
5.39.1 Detailed Description	115
5.40 LibTopoART.ITopoART_AM_base Interface Reference	115
5.40.1 Detailed Description	116
5.41 LibTopoART.ITopoART_base Interface Reference	116
5.41.1 Detailed Description	117
5.41.2 Member Function Documentation	117
5.41.3 Property Documentation	118
5.42 LibTopoART.ITopoART_C Interface Reference	118
5.42.1 Detailed Description	120
5.43 LibTopoART.ITopoART_C_base Interface Reference	120
5.43.1 Detailed Description	121
5.44 LibTopoART.ITopoART_R Interface Reference	121
5.44.1 Detailed Description	123

5.45 LibTopoART.ITopoART_R_base Interface Reference	123
5.45.1 Detailed Description	124
5.46 LibTopoART.LibTopoART_control Struct Reference	124
5.46.1 Detailed Description	125
5.47 LibTopoART.LibTopoART_info Struct Reference	125
5.47.1 Detailed Description	125
5.48 LibTopoART.Network_base Class Reference	125
5.48.1 Detailed Description	126
5.48.2 Property Documentation	126
5.49 LibTopoART.TopoART Class Reference	126
5.49.1 Detailed Description	129
5.49.2 Constructor & Destructor Documentation	129
5.49.3 Member Function Documentation	130
5.50 LibTopoART.TopoART_C Class Reference	133
5.50.1 Detailed Description	136
5.50.2 Constructor & Destructor Documentation	136
5.50.3 Member Function Documentation	137
5.51 LibTopoART.TopoART_C_prediction Struct Reference	139
5.51.1 Detailed Description	139
5.51.2 Constructor & Destructor Documentation	139
5.52 LibTopoART.TopoART_R Class Reference	140
5.52.1 Detailed Description	143
5.52.2 Constructor & Destructor Documentation	143
5.52.3 Member Function Documentation	144
5.53 LibTopoART.TopoART_R_prediction<_ElementType> Struct Template Reference	146
5.53.1 Detailed Description	147
5.53.2 Constructor & Destructor Documentation	147
5.54 LibTopoART_samples.Episodic_TopopART_sample1 Class Reference	147
5.54.1 Detailed Description	147
5.55 LibTopoART_samples.Episodic_TopopART_sample2 Class Reference	147
5.55.1 Detailed Description	148
5.56 LibTopoART_samples.TopopART_AM_sample1 Class Reference	148
5.56.1 Detailed Description	148
5.57 LibTopoART_samples.TopopART_AM_sample2 Class Reference	148
5.57.1 Detailed Description	148
5.58 LibTopoART_samples.TopopART_C_sample1 Class Reference	149
5.58.1 Detailed Description	149
5.59 LibTopoART_samples.TopopART_C_sample2 Class Reference	149
5.59.1 Detailed Description	149
5.60 LibTopoART_samples.TopopART_R_sample1 Class Reference	149
5.60.1 Detailed Description	149
5.61 LibTopoART_samples.TopopART_R_sample2 Class Reference	149

5.61.1 Detailed Description	150
5.62 LibTopoART_samples.TopoART_R_sample3 Class Reference	150
5.62.1 Detailed Description	150
5.63 LibTopoART_samples.TopoART_sample1 Class Reference	150
5.63.1 Detailed Description	150
5.64 LibTopoART_samples.TopoART_sample2 Class Reference	150
5.64.1 Detailed Description	150
5.65 LibTopoART_samples.TopoART_sample3 Class Reference	151
5.65.1 Detailed Description	151
Index	153

1 Namespace Index

1.1 Package List

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

LibTopoART	9
LibTopoART_samples	13

2 Hierarchical Index

2.1 Class Hierarchy

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

LibTopoART.IAccess_TopοART< byte >	76
LibTopoART.IFast_TopοART	101
LibTopoART.Fast_TopοART_base	38
LibTopoART.Fast_Episodic_TopοART	16
LibTopoART.Fast_TopοART	23
LibTopoART.Fast_TopοART_AM	28
LibTopoART.Fast_TopοART_C	44
LibTopoART.Fast_TopοART_R	54
LibTopoART.IFast_Episodic_TopοART	99
LibTopoART.Fast_Episodic_TopοART	16
LibTopoART.IFast_TopοART_AM	102

LibTopoART.Fast_TopoART_AM	28
LibTopoART.IFast_TopoART_C	103
LibTopoART.Fast_TopoART_C	44
LibTopoART.IFast_TopoART_R	105
LibTopoART.Fast_TopoART_R	54
LibTopoART.IAccess_TopoART< decimal >	76
LibTopoART.ITopoART	113
LibTopoART.IEpisodic_TopoART	96
LibTopoART.IFast_Episodic_TopoART	99
LibTopoART.IFast_TopoART	101
LibTopoART.IHypersphere_TopoART	110
LibTopoART.Hypersphere_TopoART	64
LibTopoART.Hypersphere_TopoART_C	68
LibTopoART.ITopoART_AM	114
LibTopoART.IFast_TopoART_AM	102
LibTopoART.ITopoART_C	118
LibTopoART.Hypersphere_TopoART_C	68
LibTopoART.IFast_TopoART_C	103
LibTopoART.TopoART_C	133
LibTopoART.ITopoART_R	121
LibTopoART.IFast_TopoART_R	105
LibTopoART.TopoART_R	140
LibTopoART.TopoART	126
LibTopoART.Hypersphere_TopoART	64
LibTopoART.TopoART_C	133
LibTopoART.TopoART_R	140
LibTopoART.IAccess_TopoART_AM< byte >	78
LibTopoART.IFast_TopoART_AM	102
LibTopoART.IAccess_TopoART_AM< decimal >	78
LibTopoART.ITopoART_AM	114
LibTopoART.IAccess_TopoART_C< byte >	81
LibTopoART.IFast_TopoART_C	103

LibTopoART.IAccess_TopoART_C< decimal >	81
LibTopoART.ITopoART_C	118
LibTopoART.IAccess_TopoART_R< byte >	85
LibTopoART.IFast_TopoART_R	105
LibTopoART.IAccess_TopoART_R< decimal >	85
LibTopoART.ITopoART_R	121
LibTopoART.IAccessAssociativeRecall< byte >	88
LibTopoART.IFastAssociativeRecall	107
LibTopoART.IFast_TopoART_AM	102
LibTopoART.IAccessAssociativeRecall< decimal >	88
LibTopoART.IAssociativeRecall	94
LibTopoART.IFastAssociativeRecall	107
LibTopoART.ITopoART_AM	114
LibTopoART.IAccessEpisodicRecall< byte >	91
LibTopoART.IFastEpisodicRecall	108
LibTopoART.IFast_Episodic_TopoART	99
LibTopoART.IAccessEpisodicRecall< decimal >	91
LibTopoART.IEpisodicRecall	98
LibTopoART.IEpisodic_TopoART	96
LibTopoART.IFastEpisodicRecall	108
LibTopoART.CategoryInfo	14
LibTopoART.F2_output	15
LibTopoART.IAdaptationStateCheck	93
LibTopoART.ITopoART	113
LibTopoART.ICategoryAccess	95
LibTopoART.Fast_TopoART_base	38
LibTopoART.TopoART	126
LibTopoART.IEndRecall	96
LibTopoART.IAccessAssociativeRecall< _AccessType >	88
LibTopoART.IAccessEpisodicRecall< _AccessType >	91
LibTopoART.InvalidClassIDException	111
LibTopoART.InvalidFileException	111

LibTopoART.InvalidModuleIndexException	111
LibTopoART.InvalidNumberException	112
LibTopoART.InvalidSizeException	112
LibTopoART.InvalidStateException	112
LibTopoART.InvalidTypeException	112
LibTopoART.ITopoART_base	116
LibTopoART.IAccess_TopoART< _AccessType >	76
LibTopoART.ITopoART_AM_base	115
LibTopoART.IAccess_TopoART_AM< _AccessType >	78
LibTopoART.ITopoART_C_base	120
LibTopoART.IAccess_TopoART_C< _AccessType >	81
LibTopoART.ITopoART_R_base	123
LibTopoART.IAccess_TopoART_R< _AccessType >	85
LibTopoART.LibTopoART_control	124
LibTopoART.LibTopoART_info	125
LibTopoART.Network_base	125
LibTopoART.Fast_TopoART_base	38
LibTopoART.TopoART	126
LibTopoART.TopoART_C_prediction	139
LibTopoART.TopoART_R_prediction< _ElementType >	146
LibTopoART_samples.Episodic_TopoART_sample1	147
LibTopoART_samples.Episodic_TopoART_sample2	147
LibTopoART_samples.TopoART_AM_sample1	148
LibTopoART_samples.TopoART_AM_sample2	148
LibTopoART_samples.TopoART_C_sample1	149
LibTopoART_samples.TopoART_C_sample2	149
LibTopoART_samples.TopoART_R_sample1	149
LibTopoART_samples.TopoART_R_sample2	149
LibTopoART_samples.TopoART_R_sample3	150
LibTopoART_samples.TopoART_sample1	150
LibTopoART_samples.TopoART_sample2	150
LibTopoART_samples.TopoART_sample3	151

3 Class Index

3.1 Class List

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

LibTopoART.CategoryInfo	
Struct CategoryInfo summarises information about a node's category	14
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	15
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."	16
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."	28
LibTopoART.Fast_TopoART_base	
Base class providing functionality common to several TopoART networks	38
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."

68

LibTopoART.IAccess_TopoART<_AccessType>

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

76

LibTopoART.IAccess_TopoART_AM<_AccessType>

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

78

LibTopoART.IAccess_TopoART_C<_AccessType>

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

81

LibTopoART.IAccess_TopoART_R<_AccessType>

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

85

LibTopoART.IAccessAssociativeRecall<_AccessType>

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

88

LibTopoART.IAccessEpisodicRecall<_AccessType>

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

91

LibTopoART.IAdaptationStateCheck

Interface enabling checks whether certain adaptations of a network occurred

93

LibTopoART.IAssociativeRecall

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

94

LibTopoART.ICategoryAccess

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

95

LibTopoART.IEndRecall

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

96

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

96

LibTopoART.IEpisodicRecall

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

98

[**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 `byte` or of type `decimal` as well as adaptation state control

99

[**LibTopoART.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

101

[**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 `byte` or of type `decimal` as well as adaptation state control

102

[**LibTopoART.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

103

[**LibTopoART.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

105

[**LibTopoART.IFastAssociativeRecall**](#)

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

107

[**LibTopoART.IFastEpisodicRecall**](#)

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

108

[**LibTopoART.IHypersphere_TopoART**](#)

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

110

[**LibTopoART.InvalidClassIDException**](#)

Exception signalling an invalid class ID

111

[**LibTopoART.InvalidFileException**](#)

Exception signalling an invalid file

111

[**LibTopoART.InvalidModuleIndexException**](#)

Exception signalling an invalid module index

111

[**LibTopoART.InvalidNumberException**](#)

Exception signalling an invalid number

112

[**LibTopoART.InvalidSizeException**](#)

Exception signalling an invalid size

112

[**LibTopoART.InvalidStateException**](#)

Exception signalling an invalid state of the neural network

112

[**LibTopoART.InvalidTypeException**](#)

Exception signalling an invalid type

112

[**LibTopoART.ITopoART**](#)

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

113

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	114
LibTopoART.ITopoART_AM_base	Interface summarising the basic TopoART-AM functionality excluding learning and prediction	115
LibTopoART.ITopoART_base	Interface summarising the basic TopoART functionality excluding learning and prediction	116
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	118
LibTopoART.ITopoART_C_base	Interface summarising the basic TopoART-C functionality excluding learning and prediction	120
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	121
LibTopoART.ITopoART_R_base	Interface summarising the basic TopoART-R functionality excluding learning and prediction	123
LibTopoART.LibTopoART_control	Struct <code>LibTopoART_control</code> provides fields to control the general behaviour of LibTopoART	124
LibTopoART.LibTopoART_info	Struct <code>LibTopoART_info</code> provides some metainformation regarding the respective implementation of LibTopoART	125
LibTopoART.Network_base	Class <code>Network_base</code> provides the functionality required by all neural network implementations of LibTopoART	125
LibTopoART.TopoART	Class <code>TopoART</code> 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."	126
LibTopoART.TopoART_C	Class <code>TopoART_C</code> 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."	133
LibTopoART.TopoART_C_prediction	Struct <code>TopoART_C_prediction</code> contains a prediction made by a TopoART-C network	139
LibTopoART.TopoART_R	Class <code>TopoART_R</code> 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."	140
LibTopoART.TopoART_R_prediction<_ElementType>	Struct <code>TopoART_R_prediction</code> contains a prediction made by a TopoART-R network	146

LibTopoART_samples.Episodic_TopoART_sample1	Episodic clustering sample using synthetic two-dimensional data. [C#]	147
LibTopoART_samples.Episodic_TopoART_sample2	Episodic clustering sample using real-world video data. [C#]	147
LibTopoART_samples.TopoART_AM_sample1	Sample using TopoART-AM with synthetic two-dimensional data. [C#]	148
LibTopoART_samples.TopoART_AM_sample2	Learning of bidirectional associations between images. [F#]	148
LibTopoART_samples.TopoART_C_sample1	Simple classification sample. [C#]	149
LibTopoART_samples.TopoART_C_sample2	Classification sample using more complex synthetic two-dimensional data with associated class IDs. [C#]	149
LibTopoART_samples.TopoART_R_sample1	Regression sample using TopoART-R. (simplified version) [C#]	149
LibTopoART_samples.TopoART_R_sample2	Regression sample using TopoART-R. (advanced version) [C#]	149
LibTopoART_samples.TopoART_R_sample3	Pixel-wise regression analysis of an image using TopoART-R. [F#]	150
LibTopoART_samples.TopoART_sample1	Simple clustering sample. [C#]	150
LibTopoART_samples.TopoART_sample2	Clustering sample using more complex synthetic two-dimensional data. [C#]	150
LibTopoART_samples.TopoART_sample3	Clustering sample using very noisy synthetic two-dimensional data. [VB]	151

4 Namespace Documentation

4.1 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](#)
Interface providing access to the basic TopoART functionality using input elements of type `_AccessType`.
- interface [IAccess_TopoART_AM](#)
Interface providing access to the basic TopoART-AM functionality using input elements of type `_AccessType`.
- interface [IAccess_TopoART_C](#)
Interface providing access to the basic TopoART-C functionality using input elements of type `_AccessType`.
- interface [IAccess_TopoART_R](#)
Interface providing access to the basic TopoART-R functionality using input elements of type `_AccessType`.
- interface [IAccessAssociativeRecall](#)
Interface providing access to the basic associative recall functionality using stimulus elements and recall result elements of type `_AccessType`.
- interface [IAccessEpisodicRecall](#)
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_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](#)

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](#)

 = 0x00ff , [ADDED_PERMANENT_NODE](#) = 0x0100 ,

[ADAPTED_PERMANENT_WEIGHT](#) = 0x0200 , [ADDED_PERMANENT_EDGE](#) = 0x0400 , [ANY_PERMANENT_ADAPTATION](#)

 = 0xff00 }

Enumeration specifying possible adaptation states.
- enum [VerbosityLevel](#) : uint { [Important](#) , [Normal](#) , [Verbose](#) }

Enumeration specifying possible adaptation states.

4.1.1 Enumeration Type Documentation

4.1.1.1 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.

4.1.1.2 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.

4.2 LibTopoART_samples Namespace Reference**Classes**

- class [Episodic_TopoART_sample1](#)
Episodic clustering sample using synthetic two-dimensional data. [C#].
- class [Episodic_TopoART_sample2](#)
Episodic clustering sample using real-world video data. [C#].
- class [TopoART_AM_sample1](#)
Sample using TopoART-AM with synthetic two-dimensional data. [C#].
- class [TopoART_AM_sample2](#)
Learning of bidirectional associations between images. [F#].
- class [TopoART_C_sample1](#)
Simple classification sample. [C#].
- class [TopoART_C_sample2](#)
Classification sample using more complex synthetic two-dimensional data with associated class IDs. [C#].

- class [TopoART_R_sample1](#)
Regression sample using TopoART-R. (simplified version) [C#].
- class [TopoART_R_sample2](#)
Regression sample using TopoART-R. (advanced version) [C#].
- class [TopoART_R_sample3](#)
Pixel-wise regression analysis of an image using TopoART-R. [F#].
- class [TopoART_sample1](#)
Simple clustering sample. [C#].
- class [TopoART_sample2](#)
Clustering sample using more complex synthetic two-dimensional data. [C#].
- class [TopoART_sample3](#)
Clustering sample using very noisy synthetic two-dimensional data. [VB].

5 Class Documentation

5.1 LibTopoART.CategoryInfo Struct Reference

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

Public Member Functions

- [CategoryInfo](#) (decimal[] [spatial_weights](#), decimal[]? [temporal_weights](#), long [cluster_ID](#), long [class_ID](#))
This constructor sets the instance variables `spatial_weights`, `temporal_weights`, `cluster_ID`, and `class_ID` 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 support temporal learning.
- readonly long [cluster_ID](#)
Instance variable `cluster_ID` represents the cluster ID of the considered node.
- readonly long [class_ID](#)
Instance variable `class_ID` 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`.

5.1.1 Detailed Description

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

5.1.2 Constructor & Destructor Documentation

5.1.2.1 `CategoryInfo()` `LibTopoART.CategoryInfo.CategoryInfo (decimal[] spatial_weights, decimal?[] temporal_weights, long cluster_ID, long class_ID)`

This constructor sets the instance variables `spatial_weights`, `temporal_weights`, `cluster_ID`, and `class_ID` of struct `CategoryInfo`.

Parameters

<i>spatial_weights</i>	The spatial weights to be set.
<i>temporal_weights</i>	The temporal weights to be set.
<i>cluster_ID</i>	The cluster ID to be set.
<i>class_ID</i>	The class ID to be set.

5.2 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 Member Functions

- **F2_output ()**

This constructor sets all instance variables of class `F2_output` to `LibTopoART_info.UNDEFINED`.

Public Attributes

- decimal **bm_node_activation**

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

- long **bm_node_ID**

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

- long **bm_cluster_ID**

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

- decimal **bm_permanent_node_activation**

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

- long **bm_permanent_node_ID**

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

- long **bm_permanent_cluster_ID**

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

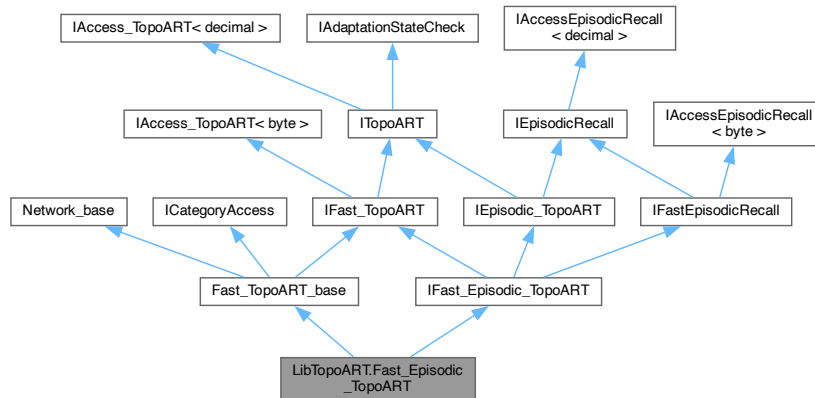
5.2.1 Detailed Description

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

5.3 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 input_length, long module_number, 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.
- 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[]? recall_result, out decimal F3_activation)
This method performs a single inter-episode recall step and sets the starting point for intra-episode recall.
- bool `InterEpisodeRecallStep` (out decimal[]? recall_result, out decimal F3_activation)
This method performs a single inter-episode recall step and sets the starting point for intra-episode recall.
- bool `IntraEpisodeRecallStep` (out byte[]? recall_result)
This method performs a single intra-episode recall step.
- bool `IntraEpisodeRecallStep` (out decimal[]? recall_result)
This method performs a single intra-episode recall step.
- void `EndRecall` ()
This method stops the recall process and frees temporary resources.

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

- abstract void [Learn](#) (byte[] input)
This method performs a single training step.
- abstract void [Learn](#) (decimal[] input)
This method performs a single training step.
- void [Dispose](#) ()
Releases all resources used by the [LibTopoART.Fast_TopoART_base](#) object.
- void [ComputeClusterIDs](#) ()
This method computes the cluster IDs for all neurons.
- [F2_output\[\] GetBMOutput](#) (byte[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (byte[] input, bool[]? mask_vector)
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_vector)
This method finds the closest category for a given test input.
- void [SaveText](#) (string path)
This method saves the entire network as a text file.
- void [Save](#) (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void [Save](#) (string path, bool compatibility_mode, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void [ResetAdaptationState](#) ()
This method resets the adaptation state to `AdaptationState.NO_ADAPTATION`.
- [AdaptationState GetAdaptationState](#) (decimal epsilon=0.001m)
This method returns the current adaptation state.
- List< [CategoryInfo](#) >? [GetCategories](#) (long module_index=FINAL_MODULE)
This method collects information on the categories of a specified module.

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

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

- [F2_output\[\] GetBMOutput](#) ([_AccessType\[\]](#) input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) ([_AccessType\[\]](#) input, [bool\[\]](#) mask_vector)
This method finds the closest category for a given test input.
- void [Learn](#) ([_AccessType\[\]](#) input)
This method performs a single training step.
- List< [CategoryInfo](#) >? [GetCategories](#) (long module_index=[LibTopoART_info.FINAL_MODULE](#))
This method collects information on the categories of a specified module.

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

- long [BeginRecall](#) ([_AccessType\[\]](#) stimulus)
This method starts the recall process.
- bool [InterEpisodeRecallStep](#) (out [_AccessType\[\]](#)? recall_result, out decimal F3_activation)
This method performs a single inter-episode recall step and sets the starting point for intra-episode recall.
- bool [IntraEpisodeRecallStep](#) (out [_AccessType\[\]](#)? recall_result)
This method performs a single intra-episode recall step.

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

- long [BeginRecall](#) ([_AccessType\[\]](#) stimulus)
This method starts the recall process.
- bool [InterEpisodeRecallStep](#) (out [_AccessType\[\]](#)? recall_result, out decimal F3_activation)
This method performs a single inter-episode recall step and sets the starting point for intra-episode recall.
- bool [IntraEpisodeRecallStep](#) (out [_AccessType\[\]](#)? recall_result)
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)integer_type] [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)float_type] [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]
Property ModuleNum represents the number of TopoART modules used. (The original TopoART uses two modules.)
- 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.IEpisodic_TopoART](#)

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

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.

5.3.1 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."

5.3.2 Constructor & Destructor Documentation

5.3.2.1 `Fast_Episodic_TopoART()` [1/2] `LibTopoART.Fast_Episodic_TopoART.Fast_Episodic_TopoART (`
 `long input_length,`
 `long module_number,`
 `decimal rho_a,`
 `long t_max)`

This constructor initialises an Episodic TopoART network.

Parameters

<i>input_length</i>	The length of input vectors to be learnt.
<i>module_number</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.

5.3.2.2 `Fast_Episodic_TopoART()` [2/2] `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

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

5.3.3 Member Function Documentation

5.3.3.1 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.

5.3.3.2 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.

5.3.3.3 InterEpisodeRecallStep() [1/2] `bool LibTopoART.Fast_Episodic_TopoART.InterEpisodeRecallStep (out byte?[] recall_result, out decimal F3_activation)`

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

Parameters

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

5.3.3.4 InterEpisodeRecallStep() [2/2] `bool LibTopoART.Fast_Episodic_TopoART.InterEpisodeRecallStep (`
`out decimal?[] recall_result,`
`out decimal F3_activation)`

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

Parameters

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

Returns

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

5.3.3.5 IntraEpisodeRecallStep() [1/2] `bool LibTopoART.Fast_Episodic_TopoART.IntraEpisodeRecallStep (`
`out byte?[] recall_result)`

This method performs a single intra-episode recall step.

Parameters

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

5.3.3.6 IntraEpisodeRecallStep() [2/2] `bool LibTopoART.Fast_Episodic_TopoART.IntraEpisodeRecallStep (`
`out decimal?[] recall_result)`

This method performs a single intra-episode recall step.

Parameters

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

Returns

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

5.3.3.7 Learn() [1/2] `override void LibTopoART.Fast_Episodic_TopoART.Learn (byte[] input) [virtual]`

This method performs a single training step.

The spatial weights are adapted as in the original TopoART network. In contrast, the adaptation of the temporal weight $w_{\{j,2\}^{F2,t}}$ occurring only in Episodic TopoART is slightly different↔:
 $w_{\{j,2\}^{F2,t}}(t+1) = \text{beta_j} * \text{Max}(t_2^{F1}(t), w_{\{j,2\}^{F2,t}}(t) + (1 - \text{beta_j}) * w_{\{j,2\}^{F2,t}}(t))$ for $j = \text{bm}$ or $j = \text{sbm}$. (Note: $w_{\{j,1\}^{F2,t}}$ remains constant over the life time 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].
--------------	--

Implements [LibTopoART.Fast_TopoART_base](#).

5.3.3.8 Learn() [2/2] `override void LibTopoART.Fast_Episodic_TopoART.Learn (decimal[] input) [virtual]`

This method performs a single training step.

The spatial weights are adapted as in the original TopoART network. In contrast, the adaptation of the temporal weight $w_{\{j,2\}^{F2,t}}$ occurring only in Episodic TopoART is slightly different↔:
 $w_{\{j,2\}^{F2,t}}(t+1) = \text{beta_j} * \text{Max}(t_2^{F1}(t), w_{\{j,2\}^{F2,t}}(t) + (1 - \text{beta_j}) * w_{\{j,2\}^{F2,t}}(t))$ for $j = \text{bm}$ or $j = \text{sbm}$. (Note: $w_{\{j,1\}^{F2,t}}$ remains constant over the life time of a node.)

Parameters

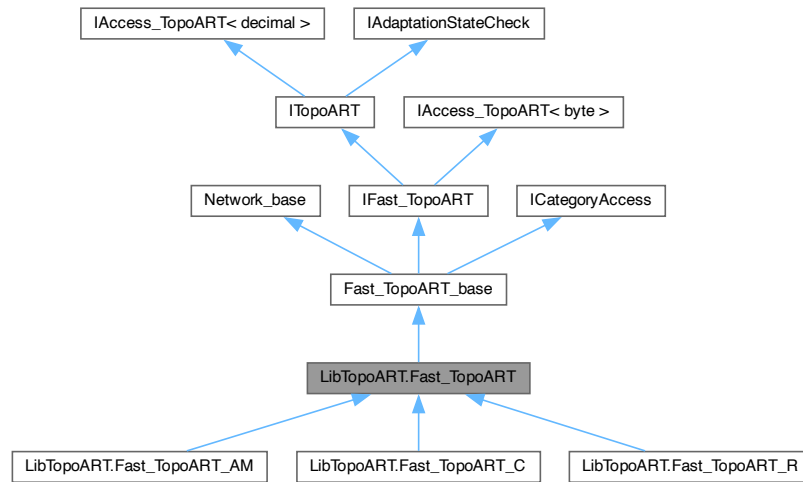
<i>input</i>	The input vector to be learnt.
--------------	--------------------------------

Implements [LibTopoART.Fast_TopoART_base](#).

5.4 LibTopoART.Fast_TopoART Class Reference

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."

Inheritance diagram for LibTopoART.Fast_TopoART:



Public Member Functions

- [Fast_TopoART](#) (long input_length, long module_number, decimal rho_a)
This constructor initialises a TopoART network.
- [Fast_TopoART](#) (string path)
This constructor loads a saved TopoART network.
- override void [Learn](#) (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](#)

- abstract void [Learn](#) (byte[] input)
This method performs a single training step.
- abstract void [Learn](#) (decimal[] input)
This method performs a single training step.
- void [Dispose](#) ()
Releases all resources used by the [LibTopoART.Fast_TopoART_base](#) object.
- void [ComputeClusterIDs](#) ()
This method computes the cluster IDs for all neurons.
- [F2_output\[\]](#) [GetBMOutput](#) (byte[] input)
This method finds the closest category for a given test input.
- [F2_output\[\]](#) [GetBMOutput](#) (byte[] input, bool[]? mask_vector)
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_vector)
This method finds the closest category for a given test input.

- void [SaveText](#) (string path)
This method saves the entire network as a text file.
- void [Save](#) (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void [Save](#) (string path, bool compatibility_mode, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void [ResetAdaptationState](#) ()
This method resets the adaptation state to `AdaptationState.NO_ADAPTATION`.
- [AdaptationState](#) [GetAdaptationState](#) (decimal epsilon=0.001m)
This method returns the current adaptation state.
- List< [CategoryInfo](#) >? [GetCategories](#) (long module_index=FINAL_MODULE)
This method collects information on the categories of a specified module.

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

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

- [F2_output\[\]](#) [GetBMOutput](#) (_AccessType[] input)
This method finds the closest category for a given test input.
- [F2_output\[\]](#) [GetBMOutput](#) (_AccessType[] input, bool[] mask_vector)
This method finds the closest category for a given test input.
- void [Learn](#) (_AccessType[] input)
This method performs a single training step.
- List< [CategoryInfo](#) >? [GetCategories](#) (long module_index=LibTopoART_info.FINAL_MODULE)
This method collects information on the categories of a specified module.

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)integer_type] [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)float_type] [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]
Property ModuleNum represents the number of TopoART modules used. (The original TopoART uses two modules.)
- 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.

5.4.1 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]`.

5.4.2 Constructor & Destructor Documentation

5.4.2.1 Fast_TopoART() [1/2] `LibTopoART.Fast_TopoART.Fast_TopoART (`
 `long input_length,`
 `long module_number,`
 `decimal rho_a)`

This constructor initialises a TopoART network.

Parameters

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

5.4.2.2 Fast_TopoART() [2/2] `LibTopoART.Fast_TopoART.Fast_TopoART (`
 `string path)`

This constructor loads a saved TopoART network.

Parameters

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

Exceptions

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

5.4.3 Member Function Documentation

5.4.3.1 Learn() [1/2] `override void LibTopoART.Fast_TopoART.Learn (`
 `byte[] input) [virtual]`

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].
--------------	--

Implements [LibTopoART.Fast_TopoART_base](#).

Reimplemented in [LibTopoART.Fast_TopoART_C](#), and [LibTopoART.Fast_TopoART_R](#).

5.4.3.2 Learn() [2/2] `override void LibTopoART.Fast_TopoART.Learn (decimal[] input) [virtual]`

This method performs a single training step.

Parameters

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

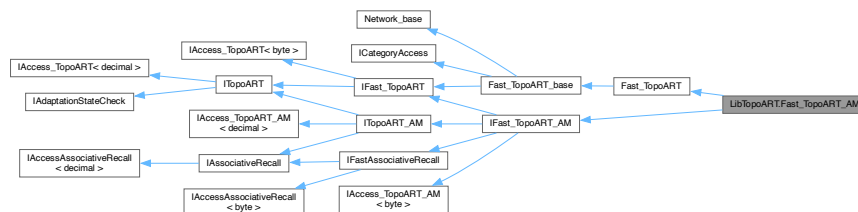
Implements [LibTopoART.Fast_TopoART_base](#).

Reimplemented in [LibTopoART.Fast_TopoART_C](#), and [LibTopoART.Fast_TopoART_R](#).

5.5 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 key_1_length, long key_2_length, long module_number, decimal rho_a)
This constructor initialises a TopoART-AM network.
- [Fast_TopoART_AM](#) (string path)
This constructor loads a saved TopoART-AM network.
- [F2_output\[\] GetBMOutput](#) (byte[] key_1_vec, byte[] key_2_vec)
This method finds the closest category for a given pair of keys.
- [F2_output\[\] GetBMOutput](#) (decimal[] key_1_vec, decimal[] key_2_vec)
This method finds the closest category for a given pair of keys.
- void [Learn](#) (byte[] key_1_vec, byte[] key_2_vec)
This method performs a single training step.
- void [Learn](#) (decimal[] key_1_vec, decimal[] key_2_vec)
This method performs a single training step.
- long [BeginRecallKey1](#) (byte[] key_2_vec, long module_index=[FINAL_MODULE](#))

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

 - long [BeginRecallKey1](#) (decimal[] key_2_vec, long module_index=FINAL_MODULE)

This method starts the recall process for the first key vector.
- long [BeginRecallKey2](#) (byte[] key_1_vec, long module_index=FINAL_MODULE)

This method starts the recall process for the second key vector.
- long [BeginRecallKey2](#) (decimal[] key_1_vec, long module_index=FINAL_MODULE)

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

This method performs a single associative recall step.
- bool [RecallStep](#) (out decimal[]? recall_result, 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 input_length, long module_number, decimal rho_a)

This constructor initialises a TopoART network.
- [Fast_TopoART](#) (string path)

This constructor loads a saved TopoART network.
- override void [Learn](#) (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](#)

- abstract void [Learn](#) (byte[] input)

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

This method performs a single training step.
- void [Dispose](#) ()

Releases all resources used by the [LibTopoART.Fast_TopoART_base](#) object.
- void [ComputeClusterIDs](#) ()

This method computes the cluster IDs for all neurons.
- [F2_output\[\]](#) [GetBMOutput](#) (byte[] input)

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

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_vector)

This method finds the closest category for a given test input.
- void [SaveText](#) (string path)

This method saves the entire network as a text file.
- void [Save](#) (string path, CompressionLevel compression=CompressionLevel.Fastest)

This method saves the entire network as a binary file.
- void [Save](#) (string path, bool compatibility_mode, CompressionLevel compression=CompressionLevel.Fastest)

This method saves the entire network as a binary file.

- void [ResetAdaptationState](#) ()

This method resets the adaptation state to `AdaptationState.NO_ADAPTATION`.

- [AdaptationState GetAdaptationState](#) (decimal epsilon=0.001m)

This method returns the current adaptation state.

- List< [CategoryInfo](#) >? [GetCategories](#) (long module_index=FINAL_MODULE)

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

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

- [F2_output\[\] GetBMOutput](#) (_AccessType[] input)

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

- [F2_output\[\] GetBMOutput](#) (_AccessType[] input, bool[] mask_vector)

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

- void [Learn](#) (_AccessType[] input)

This method performs a single training step.

- 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< byte >](#)

- [F2_output\[\] GetBMOutput](#) (_AccessType[] input)

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

- [F2_output\[\] GetBMOutput](#) (_AccessType[] input, bool[] mask_vector)

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

- void [Learn](#) (_AccessType[] input)

This method performs a single training step.

- List< [CategoryInfo](#) >? [GetCategories](#) (long module_index=LibTopoART_info.FINAL_MODULE)

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

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

- [F2_output\[\] GetBMOutput](#) (_AccessType[] key_1_vec, _AccessType[] key_2_vec)

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

- void [Learn](#) (_AccessType[] key_1_vec, _AccessType[] key_2_vec)

This method performs a single training step.

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

- long [BeginRecallKey1](#) (_AccessType[] key_2_vec, long module_index=LibTopoART_info.FINAL_MODULE)

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

- long [BeginRecallKey2](#) (_AccessType[] key_1_vec, long module_index=LibTopoART_info.FINAL_MODULE)

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

- bool [RecallStep](#) (out _AccessType[]? recall_result, out decimal F3_activation)

This method performs a single associative recall step.

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

- [F2_output\[\] GetBMOutput](#) (_AccessType[] key_1_vec, _AccessType[] key_2_vec)
This method finds the closest category for a given pair of keys.
- void [Learn](#) (_AccessType[] key_1_vec, _AccessType[] key_2_vec)
This method performs a single training step.

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

- long [BeginRecallKey1](#) (_AccessType[] key_2_vec, long module_index=LibTopoART_info.FINAL_MODULE)
This method starts the recall process for the first key vector.
- long [BeginRecallKey2](#) (_AccessType[] key_1_vec, long module_index=LibTopoART_info.FINAL_MODULE)
This method starts the recall process for the second key vector.
- bool [RecallStep](#) (out _AccessType[]? recall_result, 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 TopoART_AM.
- long **Key_1_len** [get]
Property Key_1_len returns the length of the first key vector.
- long **Key_2_len** [get]
Property Key_2_len 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)integer_type] [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)float_type] [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]
Property ModuleNum represents the number of TopoART modules used. (The original TopoART uses two modules.)
- 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.

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.

5.5.1 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 `TopoART_AM` requires all input and output to lie in the interval [0, 1].

5.5.2 Constructor & Destructor Documentation

5.5.2.1 Fast_TopoART_AM() [1/2] `LibTopoART.Fast_TopoART_AM.Fast_TopoART_AM (`
 long *key_1_length*,
 long *key_2_length*,
 long *module_number*,
 decimal *rho_a*)

This constructor initialises a TopoART-AM network.

Parameters

<i>key_1_length</i>	The length of the first key vector to be learnt.
<i>key_2_length</i>	The length of the second key vector to be learnt.
<i>module_number</i>	The number of TopoART-AM modules.
<i>rho_a</i>	The vigilance parameter of the first TopoART-AM module (TopoART-AM a).

5.5.2.2 Fast_TopoART_AM() [2/2] `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>	Throws when the given file cannot be loaded.
-----------------------------	--

5.5.3 Member Function Documentation

5.5.3.1 BeginRecallKey1() [1/2] `long LibTopoART.Fast_TopoART_AM.BeginRecallKey1 (byte[] key_2_vec, long module_index = FINAL_MODULE)`

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

Parameters

<i>key_2_vec</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>module_index</i>	Index of the TopoART-AM module to be used for recall. (<i>FINAL_MODULE</i> denotes the module with the highest index.)

Returns

The number of F3 nodes created.

Exceptions

<i>InvalidModuleIndexException</i>	Throws when <i>module_index</i> is invalid.
------------------------------------	---

5.5.3.2 BeginRecallKey1() [2/2] `long LibTopoART.Fast_TopoART_AM.BeginRecallKey1 (decimal[] key_2_vec, long module_index = FINAL_MODULE)`

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

Parameters

<i>key_2_vec</i>	The stimulus (second key vector) which is used to trigger recall.
<i>module_index</i>	Index of the TopoART-AM module to be used for recall. (<i>FINAL_MODULE</i> denotes the module with the highest index.)

Returns

The number of F3 nodes created.

Exceptions

<i>InvalidModuleIndexException</i>	Throws when <i>module_index</i> is invalid.
------------------------------------	---

5.5.3.3 BeginRecallKey2() [1/2] `long LibTopoART.Fast_TopoART_AM.BeginRecallKey2 (`
`byte[] key_1_vec,`
`long module_index = FINAL_MODULE)`

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

Parameters

<i>key_1_vec</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>module_index</i>	Index of the TopoART-AM module to be used for recall. (<i>FINAL_MODULE</i> denotes the module with the highest index.)

Returns

The number of F3 nodes created.

Exceptions

<i>InvalidModuleIndexException</i>	Throws when <i>module_index</i> is invalid.
------------------------------------	---

5.5.3.4 BeginRecallKey2() [2/2] `long LibTopoART.Fast_TopoART_AM.BeginRecallKey2 (`
`decimal[] key_1_vec,`
`long module_index = FINAL_MODULE)`

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

Parameters

<i>key_1_vec</i>	The stimulus (first key vector) which is used to trigger recall.
<i>module_index</i>	Index of the TopoART-AM module to be used for recall. (<i>FINAL_MODULE</i> denotes the module with the highest index.)

Returns

The number of F3 nodes created.

Exceptions

<i>InvalidModuleIndexException</i>	Throws when <i>module_index</i> is invalid.
------------------------------------	---

5.5.3.5 GetBMOutput() [1/2] `F2_output[] LibTopoART.Fast_TopoART_AM.GetBMOutput (byte[] key_1_vec, byte[] key_2_vec)`

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

Parameters

<i>key_1_vec</i>	The first key vector. The elements of the key vector are internally scaled from [0, 255] to [0, 1].
<i>key_2_vec</i>	The second key vector corresponding to <i>key_1_vec</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.

5.5.3.6 GetBMOutput() [2/2] `F2_output[] LibTopoART.Fast_TopoART_AM.GetBMOutput (decimal[] key_1_vec, decimal[] key_2_vec)`

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

Parameters

<i>key_1_vec</i>	The first key vector.
<i>key_2_vec</i>	The second key vector corresponding to <i>key_1_vec</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.

5.5.3.7 Learn() [1/2] `void LibTopoART.Fast_TopoART_AM.Learn (byte[] key_1_vec, byte[] key_2_vec)`

This method performs a single training step.

Parameters

<i>key_1_vec</i>	The first key vector to be learnt.
<i>key_2_vec</i>	The second key vector corresponding to <i>key_1_vec</i> .

5.5.3.8 Learn() [2/2] `void LibTopoART.Fast_TopoART_AM.Learn (`
`decimal[] key_1_vec,`
`decimal[] key_2_vec)`

This method performs a single training step.

Parameters

<i>key_1_vec</i>	The first key vector to be learnt. The elements of the key vector are internally scaled from [0, 255] to [0, 1].
<i>key_2_vec</i>	The second key vector corresponding to <i>key_1_vec</i> . The elements of the key vector are internally scaled from [0, 255] to [0, 1].

5.5.3.9 RecallStep() [1/2] `bool LibTopoART.Fast_TopoART_AM.RecallStep (`
`out byte?[] recall_result,`
`out decimal F3_activation)`

This method performs a single associative recall step.

Parameters

<i>recall_result</i>	Returns the recall output vector 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.

5.5.3.10 RecallStep() [2/2] `bool LibTopoART.Fast_TopoART_AM.RecallStep (`
`out decimal?[] recall_result,`
`out decimal F3_activation)`

This method performs a single associative recall step.

Parameters

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

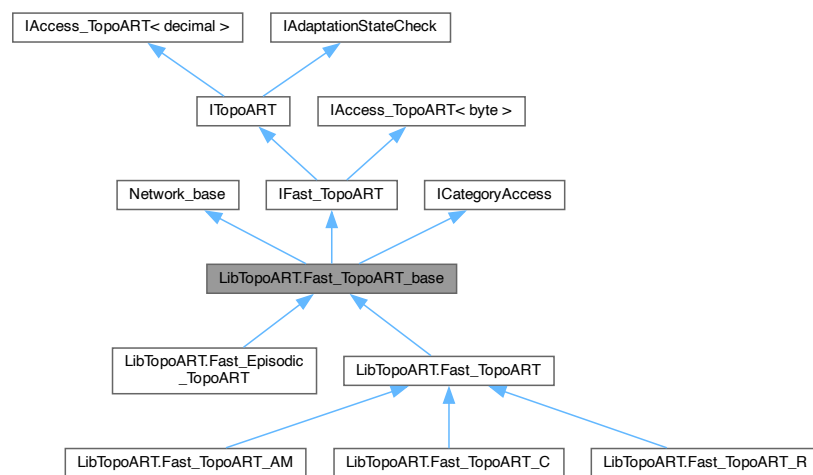
Returns

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

5.6 LibTopoART.Fast_TopoART_base Class Reference

Base class providing functionality common to several TopoART networks.

Inheritance diagram for LibTopoART.Fast_TopoART_base:



Public Member Functions

- abstract void **Learn** (byte[] input)
This method performs a single training step.
- abstract void **Learn** (decimal[] input)
This method performs a single training step.
- void **Dispose** ()
Releases all resources used by the [LibTopoART.Fast_TopoART_base](#) object.
- void **ComputeClusterIDs** ()
This method computes the cluster IDs for all neurons.
- F2_output[] **GetBMOutput** (byte[] input)
This method finds the closest category for a given test input.
- F2_output[] **GetBMOutput** (byte[] input, bool[]? mask_vector)
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_vector)
This method finds the closest category for a given test input.
- void **SaveText** (string path)
This method saves the entire network as a text file.
- void **Save** (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.

- void [Save](#) (string path, bool compatibility_mode, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void [ResetAdaptationState](#) ()
This method resets the adaptation state to `AdaptationState.NO_ADAPTATION`.
- [AdaptationState GetAdaptationState](#) (decimal epsilon=0.001m)
This method returns the current adaptation state.
- List< [CategoryInfo](#) >? [GetCategories](#) (long module_index=FINAL_MODULE)
This method collects information on the categories of a specified module.

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

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

- [F2_output\[\] GetBMOutput](#) (_AccessType[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (_AccessType[] input, bool[] mask_vector)
This method finds the closest category for a given test input.
- void [Learn](#) (_AccessType[] input)
This method performs a single training step.
- List< [CategoryInfo](#) >? [GetCategories](#) (long module_index=LibTopoART_info.FINAL_MODULE)
This method collects information on the categories of a specified module.

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

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.

5.6.1 Detailed Description

Base class providing functionality common to several TopoART networks.

5.6.2 Member Function Documentation

5.6.2.1 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.

5.6.2.2 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

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

Implements [LibTopoART.IAdaptationStateCheck](#).

5.6.2.3 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.

5.6.2.4 GetBMOutput() [2/4] `F2_output[] LibTopoART.Fast_TopoART_base.GetBMOutput (byte[] input, bool?[] mask_vector)`

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

Returns

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

5.6.2.5 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.

5.6.2.6 GetBMOutput() [4/4] `F2_output[] LibTopoART.Fast_TopoART_base.GetBMOutput (decimal[] input, bool?[] mask_vector)`

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

Parameters

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

Returns

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

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

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

Parameters

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

Implements [LibTopoART.ICategoryAccess](#).

5.6.2.8 Learn() [1/2] `abstract void LibTopoART.Fast_TopoART_base.Learn (byte[] input) [pure virtual]`

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].
--------------	--

Implemented in [LibTopoART.Fast_Episodic_TopoART](#), [LibTopoART.Fast_TopoART](#), [LibTopoART.Fast_TopoART_C](#), and [LibTopoART.Fast_TopoART_R](#).

5.6.2.9 Learn() [2/2] `abstract void LibTopoART.Fast_TopoART_base.Learn (decimal[] input) [pure virtual]`

This method performs a single training step.

Parameters

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

Implemented in [LibTopoART.Fast_Episodic_TopoART](#), [LibTopoART.Fast_TopoART](#), [LibTopoART.Fast_TopoART_C](#), and [LibTopoART.Fast_TopoART_R](#).

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

This method resets the adaptation state to `AdaptationState.NO_ADAPTATION`.

Exceptions

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

Implements [LibTopoART.IAdaptationStateCheck](#).

5.6.2.11 Save() [1/2] `void LibTopoART.Fast_TopoART_base.Save (`
`string path,`
`bool compatibility_mode,`
`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>compatibility_mode</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.)

5.6.2.12 Save() [2/2] `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 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.)

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

This method saves the entire network as a text file.

Parameters

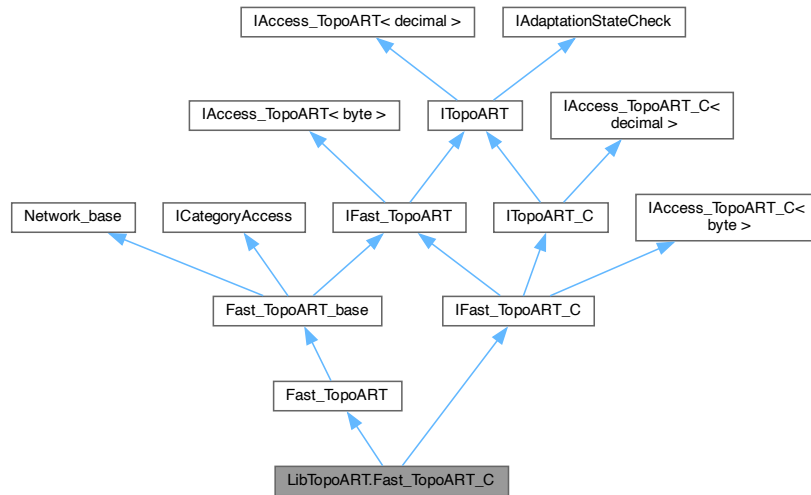
<i>path</i>	A string representing the path of the file to save.
-------------	---

5.7 LibTopoART.Fast_TopoART_C Class Reference

Class `Fast_TopoART_C` provides an implementation of the TopoART-C neural network as proposed in "Marko Tscherepanow and Sören Riechers (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."

Inheritance diagram for LibTopoART.Fast_TopoART_C:



Public Member Functions

- **Fast_TopoART_C** (long input_length, long module_number, decimal rho_a)
This constructor initialises a TopoART-C network.
- **Fast_TopoART_C** (string path)
This constructor loads a saved TopoART-C network.
- override void **Learn** (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 class_ID)
This method performs a single training step.
- void **Learn** (decimal[] input, long class_ID)
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_vector)
This method predicts the class ID using the default value of nu.
- **TopoART_C_prediction Predict** (decimal[] input, bool[]? mask_vector)

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

- [TopoART_C_prediction Predict](#) (byte[] input, bool[]? mask_vector, long nu)

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

- [TopoART_C_prediction Predict](#) (decimal[] input, bool[]? mask_vector, 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 input_length, long module_number, decimal rho_a)

This constructor initialises a TopoART network.

- [Fast_TopoART](#) (string path)

This constructor loads a saved TopoART network.

- override void [Learn](#) (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](#)

- abstract void [Learn](#) (byte[] input)

This method performs a single training step.

- abstract void [Learn](#) (decimal[] input)

This method performs a single training step.

- void [Dispose](#) ()

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

- void [ComputeClusterIDs](#) ()

This method computes the cluster IDs for all neurons.

- [F2_output\[\] GetBMOutput](#) (byte[] input)

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

- [F2_output\[\] GetBMOutput](#) (byte[] input, bool[]? mask_vector)

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_vector)

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

- void [SaveText](#) (string path)

This method saves the entire network as a text file.

- void [Save](#) (string path, CompressionLevel compression=CompressionLevel.Fastest)

This method saves the entire network as a binary file.

- void [Save](#) (string path, bool compatibility_mode, CompressionLevel compression=CompressionLevel.Fastest)

This method saves the entire network as a binary file.

- void [ResetAdaptationState](#) ()

This method resets the adaptation state to `AdaptationState.NO_ADAPTATION`.

- [AdaptationState GetAdaptationState](#) (decimal epsilon=0.001m)

This method returns the current adaptation state.

- List< [CategoryInfo](#) >? [GetCategories](#) (long module_index=FINAL_MODULE)

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

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

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

- [F2_output\[\] GetBMOutput](#) ([_AccessType\[\]](#) input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) ([_AccessType\[\]](#) input, [bool\[\]](#) mask_vector)
This method finds the closest category for a given test input.
- void [Learn](#) ([_AccessType\[\]](#) input)
This method performs a single training step.
- List< [CategoryInfo](#) >? [GetCategories](#) (long module_index=[LibTopoART_info.FINAL_MODULE](#))
This method collects information on the categories of a specified module.

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

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

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

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

- 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)integer_type] [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)float_type] [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]
Property ModuleNum represents the number of TopoART modules used. (The original TopoART uses two modules.)
- 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.

5.7.1 Detailed Description

Class `Fast_TopoART_C` provides an implementation of the TopoART-C neural network as proposed in "Marko Tscherepanow and Sören Riechers (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.

5.7.2 Constructor & Destructor Documentation

5.7.2.1 Fast_TopoART_C() [1/2] `LibTopoART.Fast_TopoART_C.Fast_TopoART_C (`
 `long input_length,`
 `long module_number,`
 `decimal rho_a)`

This constructor initialises a TopoART-C network.

Parameters

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

5.7.2.2 Fast_TopoART_C() [2/2] `LibTopoART.Fast_TopoART_C.Fast_TopoART_C (`
 `string path)`

This constructor loads a saved TopoART-C network.

Parameters

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

Exceptions

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

5.7.3 Member Function Documentation

5.7.3.1 Learn() [1/4] `override void LibTopoART.Fast_TopoART_C.Learn (byte[] 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. The input values are internally scaled from [0, 255] to [0, 1].
--------------	--

Reimplemented from [LibTopoART.Fast_TopoART](#).

5.7.3.2 Learn() [2/4] `void LibTopoART.Fast_TopoART_C.Learn (byte[] input, long class_ID)`

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

Exceptions

<i>InvalidClassIDException</i>	Throws when <i>class_ID</i> is less than 0.
--------------------------------	---

5.7.3.3 Learn() [3/4] `override void LibTopoART.Fast_TopoART_C.Learn (decimal[] input) [virtual]`

This method performs a single training step and sets the class ID corresponding to *input* to `UNDEFINED_CLASS_ID`.

Parameters

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

Reimplemented from [LibTopoART.Fast_TopoART](#).

5.7.3.4 Learn() [4/4] `void LibTopoART.Fast_TopoART_C.Learn (`
`decimal[] input,`
`long class_ID)`

This method performs a single training step.

Parameters

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

Exceptions

<i>InvalidClassIDException</i>	Throws when <i>class_ID</i> is less than 0.
--------------------------------	---

5.7.3.5 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.

5.7.3.6 Predict() [2/8] `TopoART_C_prediction LibTopoART.Fast_TopoART_C.Predict (`
`byte[] input,`
`bool?[] mask_vector)`

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_vector</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.

5.7.3.7 Predict() [3/8] `TopoART_C_prediction` `LibTopoART.Fast_TopoART_C.Predict` (
`byte[] input,`
`bool?[] mask_vector,`
`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_vector</i>	The mask vector corresponding to <i>input</i> .
<i>nu</i>	The maximum cardinality of the set of enclosing categories E and the neighbourhood set N. (This parameter does not modify the network. It may be arbitrarily changed in each prediction step.)

Returns

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

5.7.3.8 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.

5.7.3.9 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.

5.7.3.10 Predict() [6/8] `TopoART_C_prediction LibTopoART.Fast_TopoART_C.Predict (decimal[] input, bool?[] mask_vector)`

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_vector</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.

5.7.3.11 Predict() [7/8] `TopoART_C_prediction LibTopoART.Fast_TopoART_C.Predict (decimal[] input, bool?[] mask_vector, 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_vector</i>	The mask vector corresponding to <i>input</i> .
<i>nu</i>	The maximum cardinality of the set of enclosing categories E and the neighbourhood set N.
<i>LibTopoART 0.97 Reference Manual</i>	(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.

```
5.7.3.12 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 <i>E</i> and the neighbourhood set <i>N</i> . (This parameter does not modify the network. It may be arbitrarily changed in each prediction step.)

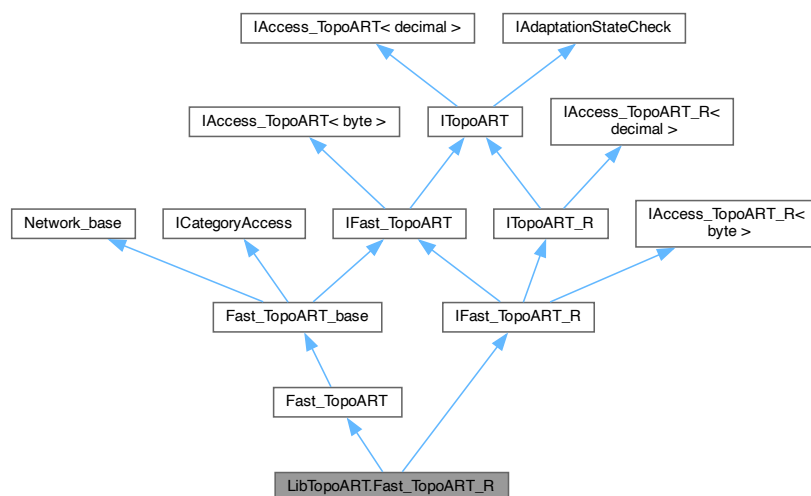
Returns

The predicted class ID.

5.8 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 i_length, long d_length, long module_number, decimal rho_a)
This constructor initialises a TopoART-R network.
- [Fast_TopoART_R](#) (string path)
This constructor loads a saved TopoART-R network.
- 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[] i_vec, byte[] d_vec)
This method performs a single training step.
- void [Learn](#) (decimal[] i_vec, decimal[] d_vec)
This method performs a single training step.
- byte[] [Predict](#) (byte[] i_vec)
This method predicts the dependent variables using the default value of nu.
- decimal[] [Predict](#) (decimal[] i_vec)
This method predicts the dependent variables using the default value of nu.
- byte[] [Predict](#) (byte[] i_vec, long nu)
This method predicts the dependent variables using a custom value of nu.
- decimal[] [Predict](#) (decimal[] i_vec, long nu)
This method predicts the dependent variables using a custom value of nu.
- [TopoART_R_prediction](#)< byte > [Predict](#) (byte[] i_vec, bool[] m_i_vec)
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 m_i_vec to true.
- [TopoART_R_prediction](#)< decimal > [Predict](#) (decimal[] i_vec, bool[] m_i_vec)
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 m_i_vec to true.
- [TopoART_R_prediction](#)< byte > [Predict](#) (byte[] i_vec, bool[] m_i_vec, 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 m_i_vec to true.
- [TopoART_R_prediction](#)< decimal > [Predict](#) (decimal[] i_vec, bool[] m_i_vec, 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 m_i_vec to true.

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

- [Fast_TopoART](#) (long input_length, long module_number, decimal rho_a)
This constructor initialises a TopoART network.
- [Fast_TopoART](#) (string path)
This constructor loads a saved TopoART network.
- override void [Learn](#) (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](#)

- abstract void [Learn](#) (byte[] input)
This method performs a single training step.
- abstract void [Learn](#) (decimal[] input)
This method performs a single training step.
- void [Dispose](#) ()
Releases all resources used by the [LibTopoART.Fast_TopoART_base](#) object.
- void [ComputeClusterIDs](#) ()
This method computes the cluster IDs for all neurons.
- [F2_output\[\] GetBMOutput](#) (byte[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (byte[] input, bool[]? mask_vector)
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_vector)
This method finds the closest category for a given test input.
- void [SaveText](#) (string path)
This method saves the entire network as a text file.
- void [Save](#) (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void [Save](#) (string path, bool compatibility_mode, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void [ResetAdaptationState](#) ()
This method resets the adaptation state to `AdaptationState.NO_ADAPTATION`.
- [AdaptationState GetAdaptationState](#) (decimal epsilon=0.001m)
This method returns the current adaptation state.
- List< [CategoryInfo](#) >? [GetCategories](#) (long module_index=FINAL_MODULE)
This method collects information on the categories of a specified module.

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

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

- [F2_output\[\] GetBMOutput](#) ([_AccessType\[\]](#) input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) ([_AccessType\[\]](#) input, [bool\[\]](#) mask_vector)
This method finds the closest category for a given test input.
- void [Learn](#) ([_AccessType\[\]](#) input)
This method performs a single training step.
- List< [CategoryInfo](#) >? [GetCategories](#) (long module_index=[LibTopoART_info.FINAL_MODULE](#))
This method collects information on the categories of a specified module.

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

- void [Learn](#) ([_AccessType\[\]](#) i_vec, [_AccessType\[\]](#) d_vec)
This method performs a single training step.
- [_AccessType\[\] Predict](#) ([_AccessType\[\]](#) i_vec)
This method predicts the dependent variables using the default value of nu.
- [_AccessType\[\] Predict](#) ([_AccessType\[\]](#) i_vec, long nu)
This method predicts the dependent variables using a custom value of nu.
- [TopoART_R_prediction< _AccessType > Predict](#) ([_AccessType\[\]](#) i_vec, [bool\[\]](#) m_i_vec)
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 m_i_vec to true.
- [TopoART_R_prediction< _AccessType > Predict](#) ([_AccessType\[\]](#) i_vec, [bool\[\]](#) m_i_vec, 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 m_i_vec to true.

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

- void [Learn](#) ([_AccessType\[\]](#) i_vec, [_AccessType\[\]](#) d_vec)
This method performs a single training step.
- [_AccessType\[\] Predict](#) ([_AccessType\[\]](#) i_vec)
This method predicts the dependent variables using the default value of nu.
- [_AccessType\[\] Predict](#) ([_AccessType\[\]](#) i_vec, long nu)
This method predicts the dependent variables using a custom value of nu.
- [TopoART_R_prediction< _AccessType > Predict](#) ([_AccessType\[\]](#) i_vec, [bool\[\]](#) m_i_vec)
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 m_i_vec to true.
- [TopoART_R_prediction< _AccessType > Predict](#) ([_AccessType\[\]](#) i_vec, [bool\[\]](#) m_i_vec, 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 m_i_vec 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)integer_type] [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)float_type] [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]
Property ModuleNum represents the number of TopoART modules used. (The original TopoART uses two modules.)
- 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.

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.

5.8.1 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].

5.8.2 Constructor & Destructor Documentation

5.8.2.1 Fast_TopoART_R() [1/2] `LibTopoART.Fast_TopoART_R.Fast_TopoART_R (`
 long *i_length*,
 long *d_length*,
 long *module_number*,
 decimal *rho_a*)

This constructor initialises a TopoART-R network.

Parameters

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

5.8.2.2 Fast_TopoART_R() [2/2] `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>	Throws when the given file cannot be loaded.
-----------------------------	--

5.8.3 Member Function Documentation

5.8.3.1 Learn() [1/4] `void LibTopoART.Fast_TopoART_R.Learn (`
`byte[] i_vec,`
`byte[] d_vec)`

This method performs a single training step.

Parameters

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

5.8.3.2 Learn() [2/4] `override void LibTopoART.Fast_TopoART_R.Learn (`
`byte[] 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. The input values are internally scaled from [0, 255] to [0, 1].
--------------	--

Reimplemented from [LibTopoART.Fast_TopoART](#).

5.8.3.3 Learn() [3/4] `void LibTopoART.Fast_TopoART_R.Learn (`
`decimal[] i_vec,`
`decimal[] d_vec)`

This method performs a single training step.

Parameters

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

5.8.3.4 Learn() [4/4] `override void LibTopoART.Fast_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.Fast_TopoART](#).

5.8.3.5 Predict() [1/8] `byte[] LibTopoART.Fast_TopoART_R.Predict (byte[] i_vec)`

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

Parameters

<i>i_vec</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.

5.8.3.6 Predict() [2/8] `TopoART_R_prediction< byte > LibTopoART.Fast_TopoART_R.Predict (byte[] i_vec, bool[] m_i_vec)`

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 *m_i_vec* to true.

Parameters

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

Returns

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

5.8.3.7 Predict() [3/8] `TopoART_R_prediction`< byte > `LibTopoART.Fast_TopoART_R.Predict` (
 `byte[] i_vec,`
 `bool[] m_i_vec,`
 `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 `m_i_vec` to `true`.

Parameters

<code>i_vec</code>	The input vector (independent variables). The elements of the input vector are internally scaled from [0, 255] to [0, 1].
<code>m_i_vec</code>	The mask vector corresponding to <code>i_vec</code> .
<code>nu</code>	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. The elements of the predicted vectors are internally scaled from [0, 1] to [0, 255].

5.8.3.8 Predict() [4/8] `byte[]` `LibTopoART.Fast_TopoART_R.Predict` (
 `byte[] i_vec,`
 `long nu`)

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

Parameters

<code>i_vec</code>	The input vector (independent variables). The elements of the input vector are internally scaled from [0, 255] to [0, 1].
<code>nu</code>	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. The elements of the predicted output vector are internally scaled from [0, 1] to [0, 255].

5.8.3.9 Predict() [5/8] `decimal[] LibTopoART.Fast_TopoART_R.Predict (decimal[] i_vec)`

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

Parameters

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

Returns

The predicted values for all dependent variables.

5.8.3.10 Predict() [6/8] `TopoART_R_prediction< decimal > LibTopoART.Fast_TopoART_R.Predict (decimal[] i_vec, bool[] m_i_vec)`

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 `m_i_vec` to `true`.

Parameters

<code>i_vec</code>	The input vector (independent variables).
<code>m_i_vec</code>	The mask vector corresponding to <code>i_vec</code> .

Returns

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

5.8.3.11 Predict() [7/8] `TopoART_R_prediction< decimal > LibTopoART.Fast_TopoART_R.Predict (decimal[] i_vec, bool[] m_i_vec, 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 `m_i_vec` to `true`.

Parameters

<code>i_vec</code>	The input vector (independent variables).
<code>m_i_vec</code>	The mask vector corresponding to <code>i_vec</code> .
<code>nu</code>	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.

5.8.3.12 Predict() [8/8] `decimal[] LibTopoART.Fast_TopoART_R.Predict (`
`decimal[] i_vec,`
`long nu)`

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

Parameters

<code>i_vec</code>	The input vector (independent variables).
<code>nu</code>	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.)

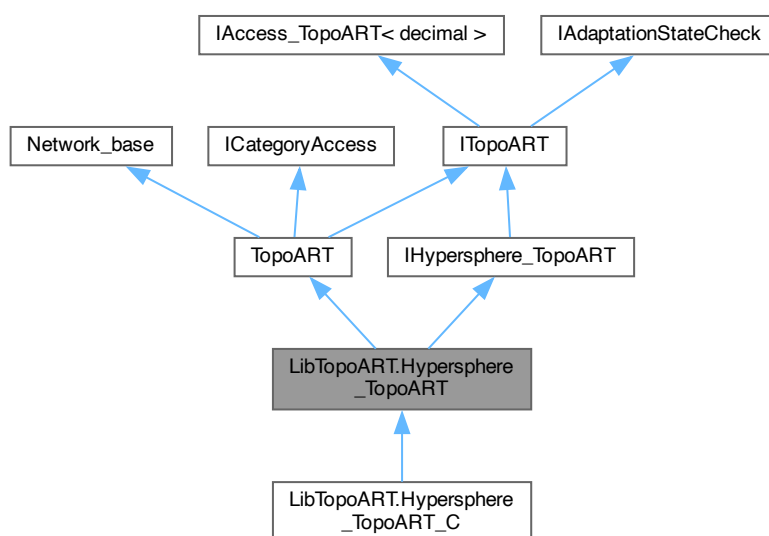
Returns

The predicted values for all dependent variables.

5.9 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 input_length, long module_number, decimal rho_a)
This constructor initialises a Hypersphere TopoART network and sets the radial extend parameter to $\text{Math}.\sqrt{\text{input_length}}/2$.
- [Hypersphere_TopoART](#) (long input_length, long module_number, decimal rho_a, decimal R)
This constructor initialises a Hypersphere TopoART network.
- [Hypersphere_TopoART](#) (string path)
This constructor loads a saved Hypersphere TopoART network.

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

- [TopoART](#) (long input_length, long module_number, decimal rho_a)
This constructor initialises a TopoART network.
- [TopoART](#) (string path)
This constructor loads a saved TopoART network.
- void [Dispose](#) ()
Releases all resources used by the [LibTopoART.TopoART](#) object.
- void [ComputeClusterIDs](#) ()
This method computes the cluster IDs for all neurons.
- [F2_output\[\] GetBMOutput](#) (decimal[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (decimal[] input, bool[]? mask_vector)
This method finds the closest category for a given test input.
- virtual void [Learn](#) (decimal[] input)
This method performs a single training step.
- void [SaveText](#) (string path)
This method saves the entire network as a text file.
- void [Save](#) (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void [ResetAdaptationState](#) ()
This method resets the adaptation state to `AdaptationState.NO_ADAPTATION`.
- [AdaptationState GetAdaptationState](#) (decimal epsilon=0.001m)
This method returns the current adaptation state.
- List< [CategoryInfo](#) >? [GetCategories](#) (long module_index=FINAL_MODULE)
This method collects information on the categories of a specified module.

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

- [F2_output\[\] GetBMOutput](#) (_AccessType[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (_AccessType[] input, bool[] mask_vector)
This method finds the closest category for a given test input.
- void [Learn](#) (_AccessType[] input)
This method performs a single training step.
- 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 module_index=LibTopoART_info.FINAL_MODULE)
This method collects information on the categories of a specified module.

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)integer_type] [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)float_type] [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]
Property ModuleNum represents the number of TopoART modules used. (The original TopoART uses two modules.)
- 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.IHypersphere_TopoART](#)

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

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.

5.9.1 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`.

5.9.2 Constructor & Destructor Documentation

5.9.2.1 `Hypersphere_TopoART()` [1/3] `LibTopoART.Hypersphere_TopoART.Hypersphere_TopoART (`
`long input_length,`
`long module_number,`
`decimal rho_a)`

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

Parameters

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

5.9.2.2 `Hypersphere_TopoART()` [2/3] `LibTopoART.Hypersphere_TopoART.Hypersphere_TopoART (`
`long input_length,`
`long module_number,`
`decimal rho_a,`
`decimal R)`

This constructor initialises a Hypersphere TopoART network.

Parameters

<code>input_length</code>	The length of input vectors to be learnt.
---------------------------	---

Parameters

<i>module_number</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.

5.9.2.3 Hypersphere_TopoART() [3/3] `LibTopoART.Hypersphere_TopoART.Hypersphere_TopoART (
string path)`

This constructor loads a saved Hypersphere TopoART network.

Parameters

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

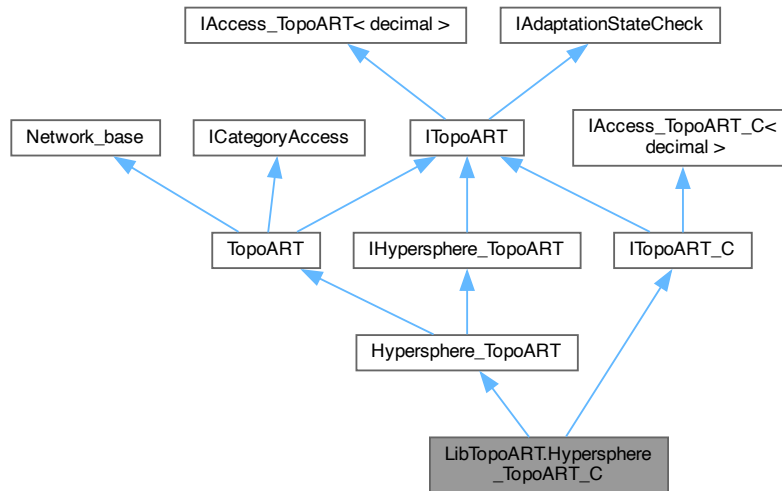
Exceptions

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

5.10 LibTopoART.Hypersphere_TopoART_C Class Reference

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."

Inheritance diagram for LibTopoART.Hypersphere_TopoART_C:



Public Member Functions

- [Hypersphere_TopoART_C](#) (long input_length, long module_number, decimal rho_a)
This constructor initialises a Hypersphere TopoART-C network and sets the radial extend parameter to $\text{Math}.\sqrt{\text{input_length}}/2$.
- [Hypersphere_TopoART_C](#) (long input_length, long module_number, decimal rho_a, decimal R)
This constructor initialises a Hypersphere TopoART-C network.
- [Hypersphere_TopoART_C](#) (string path)
This constructor loads a saved Hypersphere TopoART-C network.
- override void [Learn](#) (decimal[] input)
This method performs a single training step and sets the class ID corresponding to input to `UNDEFINED_CLASS_ID`.
- void [Learn](#) (decimal[] input, long class_ID)
This method performs a single training step.
- long [Predict](#) (decimal[] input)
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_vector)
This method predicts the class ID using the default value of nu.
- [TopoART_C_prediction Predict](#) (decimal[] input, bool[]? mask_vector, long nu)
This method predicts the class ID using a custom value of nu.

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

- [Hypersphere_TopoART](#) (long input_length, long module_number, decimal rho_a)
This constructor initialises a Hypersphere TopoART network and sets the radial extend parameter to $\text{Math}.\sqrt{\text{input_length}}/2$.
- [Hypersphere_TopoART](#) (long input_length, long module_number, decimal rho_a, decimal R)
This constructor initialises a Hypersphere TopoART network.
- [Hypersphere_TopoART](#) (string path)
This constructor loads a saved Hypersphere TopoART network.

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

- [TopoART](#) (long input_length, long module_number, decimal rho_a)
This constructor initialises a TopoART network.
- [TopoART](#) (string path)
This constructor loads a saved TopoART network.
- void [Dispose](#) ()
Releases all resources used by the [LibTopoART.TopoART](#) object.
- void [ComputeClusterIDs](#) ()
This method computes the cluster IDs for all neurons.
- [F2_output\[\] GetBMOutput](#) (decimal[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (decimal[] input, bool[]? mask_vector)
This method finds the closest category for a given test input.
- virtual void [Learn](#) (decimal[] input)
This method performs a single training step.
- void [SaveText](#) (string path)
This method saves the entire network as a text file.
- void [Save](#) (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void [ResetAdaptationState](#) ()
This method resets the adaptation state to `AdaptationState.NO_ADAPTATION`.
- [AdaptationState GetAdaptationState](#) (decimal epsilon=0.001m)
This method returns the current adaptation state.
- List< [CategoryInfo](#) >? [GetCategories](#) (long module_index=[FINAL_MODULE](#))
This method collects information on the categories of a specified module.

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

- [F2_output\[\] GetBMOutput](#) (_AccessType[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (_AccessType[] input, bool[] mask_vector)
This method finds the closest category for a given test input.
- void [Learn](#) (_AccessType[] input)
This method performs a single training step.
- 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 module_index=[LibTopoART_info.FINAL_MODULE](#))
This method collects information on the categories of a specified module.

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

- void [Learn](#) (_AccessType[] input, long class_ID)
This method performs a single training step.
- long [Predict](#) (_AccessType[] input)
This method predicts the class ID using the default value of nu.
- long [Predict](#) (_AccessType[] input, long nu)
This method predicts the class ID using a custom value of nu.
- [TopoART_C_prediction Predict](#) (_AccessType[] input, bool[] mask_vector)
This method predicts the class ID using the default value of nu.
- [TopoART_C_prediction Predict](#) (_AccessType[] input, bool[] mask_vector, 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 predict 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** [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)integer_type] [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)float_type] [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]
Property ModuleNum represents the number of TopoART modules used. (The original TopoART uses two modules.)
- 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.IHypersphere_TopoART](#)

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

5.10.1 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`.

5.10.2 Constructor & Destructor Documentation

5.10.2.1 Hypersphere_TopoART_C() [1/3] LibTopoART.Hypersphere_TopoART_C.Hypersphere_TopoART_C (
 long *input_length*,
 long *module_number*,
 decimal *rho_a*)

This constructor initialises a Hypersphere TopoART-C network and sets the radial extend parameter to $\text{Math}.\sqrt{\text{input_length}}/2$.

Parameters

<i>input_length</i>	The length of input vectors to be learnt.
<i>module_number</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).

5.10.2.2 Hypersphere_TopoART_C() [2/3] LibTopoART.Hypersphere_TopoART_C.Hypersphere_TopoART_C (
 long *input_length*,
 long *module_number*,
 decimal *rho_a*,
 decimal *R*)

This constructor initialises a Hypersphere TopoART-C network.

Parameters

<i>input_length</i>	The length of input vectors to be learnt.
<i>module_number</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.

5.10.2.3 Hypersphere_TopoART_C() [3/3] 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>	Throws when the given file cannot be loaded.
-----------------------------	--

5.10.3 Member Function Documentation

5.10.3.1 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](#).

5.10.3.2 Learn() [2/2] `void LibTopoART.Hypersphere_TopoART_C.Learn (decimal[] input, long class_ID)`

This method performs a single training step.

Parameters

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

Exceptions

<i>InvalidClassIDException</i>	Throws when <i>class_ID</i> is less than 0.
--------------------------------	---

5.10.3.3 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.

5.10.3.4 Predict() [2/4] [TopoART_C_prediction](#) LibTopoART.Hypersphere_TopoART_C.Predict (
 decimal[] *input*,
 bool?[] *mask_vector*)

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_vector</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.

5.10.3.5 Predict() [3/4] [TopoART_C_prediction](#) LibTopoART.Hypersphere_TopoART_C.Predict (
 decimal[] *input*,
 bool?[] *mask_vector*,
 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_vector</i>	The mask vector corresponding to <i>input</i> .
<i>nu</i>	The maximum cardinality of the set of enclosing categories E and the neighbourhood set N. (This parameter does not modify the network. It may be arbitrarily changed in each prediction step.)

Returns

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

5.10.3.6 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.)

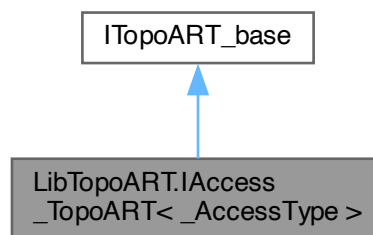
Returns

The predicted class ID.

5.11 LibTopoART.IAccess_TopopART< _AccessType > Interface Template Reference

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

Inheritance diagram for LibTopoART.IAccess_TopopART< _AccessType >:

**Public Member Functions**

- `F2_output[] GetBMOutput (_AccessType[] input)`
This method finds the closest category for a given test input.
- `F2_output[] GetBMOutput (_AccessType[] input, bool[] mask_vector)`
This method finds the closest category for a given test input.
- `void Learn (_AccessType[] 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.

5.11.1 Detailed Description

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

Type Constraints

`_AccessType` : struct
`_AccessType` : IConvertible

5.11.2 Member Function Documentation

5.11.2.1 GetBMOutput() [1/2] `F2_output[] LibTopoART.IAccess_TopoART<_AccessType>.GetBMOutput (`
`AccessType[] 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.

5.11.2.2 GetBMOutput() [2/2] `F2_output [] LibTopoART.IAccess_TopoART< _AccessType >.GetBMOutput (`
`_AccessType [] input,`
`bool [] mask_vector)`

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

Parameters

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

Returns

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

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

This method performs a single training step.

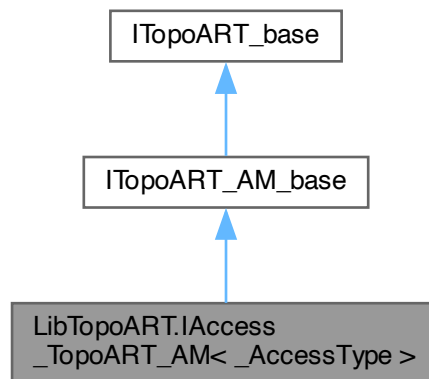
Parameters

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

5.12 LibTopoART.IAccess_TopoART_AM< _AccessType > 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<_AccessType>:



Public Member Functions

- [F2_output\[\] GetBMOutput](#) (_AccessType[] key_1_vec, _AccessType[] key_2_vec)
This method finds the closest category for a given pair of keys.
- void [Learn](#) (_AccessType[] key_1_vec, _AccessType[] key_2_vec)
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 **Key_1_len** [get]
Property `K_1_len` returns the length of the first key vector.
- long **Key_2_len** [get]
Property `K_2_len` 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.

5.12.1 Detailed Description

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

Type Constraints

`_AccessType` : struct
`_AccessType` : IConvertible

5.12.2 Member Function Documentation

5.12.2.1 GetBMOutput() `F2_output[] LibTopoART.IAccess_TopoART_AM< _AccessType >.GetBMOutput (_AccessType[] key_1_vec, _AccessType[] key_2_vec)`

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

Parameters

<code>key_1_vec</code>	The first key vector.
<code>key_2_vec</code>	The second key vector corresponding to <code>key_1_vec</code> .

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.

5.12.2.2 Learn() void `LibTopoART.IAccess_TopoART_AM< _AccessType >.Learn (`
 `_AccessType[] key_1_vec,`
 `_AccessType[] key_2_vec)`

This method performs a single training step.

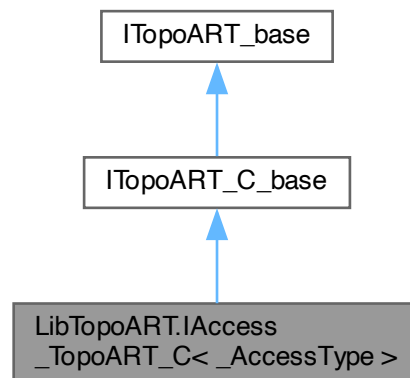
Parameters

<code>key_1_vec</code>	The first key vector to be learnt.
<code>key_2_vec</code>	The second key vector corresponding to <code>key_1_vec</code> .

5.13 LibTopoART.IAccess_TopoART_C< _AccessType > 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< _AccessType >`:

**Public Member Functions**

- void `Learn (_AccessType[] input, long class_ID)`
This method performs a single training step.
- long `Predict (_AccessType[] input)`
This method predicts the class ID using the default value of nu.
- long `Predict (_AccessType[] input, long nu)`

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

- [TopoART_C_prediction Predict](#) ([_AccessType\[\]](#) input, [bool\[\]](#) mask_vector)

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

- [TopoART_C_prediction Predict](#) ([_AccessType\[\]](#) input, [bool\[\]](#) mask_vector, 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.

5.13.1 Detailed Description

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

Type Constraints

`_AccessType` : *struct*
`_AccessType` : *Convertible*

5.13.2 Member Function Documentation

5.13.2.1 Learn() void `LibTopoART.IAccess_TopoART_C< _AccessType >.Learn` (
`_AccessType[] input`,
`long class_ID`)

This method performs a single training step.

Parameters

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

5.13.2.2 Predict() [1/4] long `LibTopoART.IAccess_TopoART_C< _AccessType >.Predict` (
`_AccessType[] 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.

5.13.2.3 Predict() [2/4] `TopoART_C_prediction` `LibTopoART.IAccess_TopoART_C< _AccessType >.Predict` (
`_AccessType[] input`,
`bool[] mask_vector`)

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_vector</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.

5.13.2.4 Predict() [3/4] `TopoART_C_prediction LibTopoART.IAccess_TopoART_C< _AccessType >.Predict (`
`(`
`_AccessType[] input,`
`bool[] mask_vector,`
`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_vector</i>	The mask vector corresponding to <i>input</i> .
<i>nu</i>	The maximum cardinality of the set of enclosing categories E and the neighbourhood set N. (This parameter does not modify the network. It may be arbitrarily changed in each prediction step.)

Returns

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

5.13.2.5 Predict() [4/4] `long LibTopoART.IAccess_TopoART_C< _AccessType >.Predict (`
`_AccessType[] 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.)

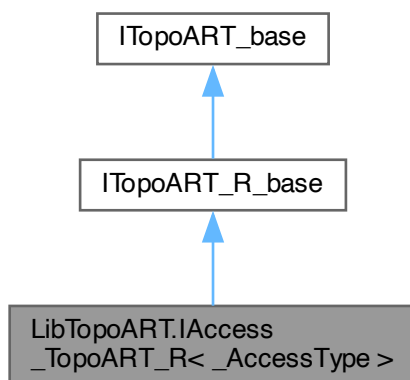
Returns

The predicted class ID.

5.14 LibTopoART.IAccess_TopoART_R< _AccessType > 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< _AccessType >:



Public Member Functions

- void **Learn** (`_AccessType[] i_vec`, `_AccessType[] d_vec`)
This method performs a single training step.
- `_AccessType[] Predict` (`_AccessType[] i_vec`)
This method predicts the dependent variables using the default value of `nu`.
- `_AccessType[] Predict` (`_AccessType[] i_vec`, long `nu`)
This method predicts the dependent variables using a custom value of `nu`.
- `TopoART_R_prediction< _AccessType > Predict` (`_AccessType[] i_vec`, `bool[] m_i_vec`)
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 `m_i_vec` to `true`.
- `TopoART_R_prediction< _AccessType > Predict` (`_AccessType[] i_vec`, `bool[] m_i_vec`, 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 `m_i_vec` 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.

5.14.1 Detailed Description

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

Type Constraints

`_AccessType` : struct
`_AccessType` : IConvertible

5.14.2 Member Function Documentation

5.14.2.1 Learn() void [LibTopoART.IAccess_TopoART_R](#)< `_AccessType` >.Learn (
 `_AccessType[] i_vec`,
 `_AccessType[] d_vec`)

This method performs a single training step.

Parameters

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

5.14.2.2 Predict() [1/4] `_AccessType[] LibTopoART.IAccess_TopoART_R<_AccessType>.Predict (_AccessType[] i_vec)`

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

Parameters

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

Returns

The predicted values for all dependent variables.

5.14.2.3 Predict() [2/4] `TopoART_R_prediction<_AccessType> LibTopoART.IAccess_TopoART_R<_AccessType>.Predict (_AccessType[] i_vec, bool[] m_i_vec)`

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 *m_i_vec* to `true`.

Parameters

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

Returns

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

5.14.2.4 Predict() [3/4] `TopoART_R_prediction<_AccessType> LibTopoART.IAccess_TopoART_R<_AccessType>.Predict (_AccessType[] i_vec, bool[] m_i_vec, 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 *m_i_vec* to `true`.

Parameters

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

Returns

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

5.14.2.5 Predict() [4/4] `_AccessType[] LibTopoART.IAccess_TopoART_R<_AccessType>.Predict (`
`_AccessType[] i_vec,`
`long nu)`

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

Parameters

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

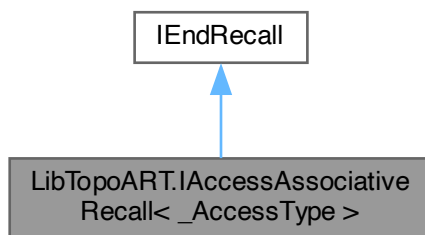
Returns

The predicted values for all dependent variables.

5.15 LibTopoART.IAccessAssociativeRecall<_AccessType> Interface Template Reference

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

Inheritance diagram for LibTopoART.IAccessAssociativeRecall< _AccessType >:



Public Member Functions

- long [BeginRecallKey1](#) (_AccessType[] key_2_vec, long module_index=[LibTopoART_info.FINAL_MODULE](#))
This method starts the recall process for the first key vector.
- long [BeginRecallKey2](#) (_AccessType[] key_1_vec, long module_index=[LibTopoART_info.FINAL_MODULE](#))
This method starts the recall process for the second key vector.
- bool [RecallStep](#) (out _AccessType[]? recall_result, 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.

5.15.1 Detailed Description

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

Type Constraints

`_AccessType` : **struct**
`_AccessType` : **IConvertible**

5.15.2 Member Function Documentation

5.15.2.1 BeginRecallKey1() long [LibTopoART.IAccessAssociativeRecall](#)< _AccessType >.BeginRecallKey1 (
 _accessType[] key_2_vec,
 long module_index = [LibTopoART_info.FINAL_MODULE](#))

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

Parameters

<i>key_2_vec</i>	The stimulus (second key vector) which is used to trigger recall.
<i>module_index</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.

```
5.15.2.2 BeginRecallKey2() long LibTopoART.IAccessAssociativeRecall< _AccessType >.BeginRecallKey2 (
    _AccessType[] key_1_vec,
    long module_index = LibTopoART_info.FINAL_MODULE )
```

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

Parameters

<i>key_1_vec</i>	The stimulus (first key vector) which is used to trigger recall.
<i>module_index</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.

```
5.15.2.3 RecallStep() bool LibTopoART.IAccessAssociativeRecall< _AccessType >.RecallStep (
    out _AccessType?[] recall_result,
    out decimal F3_activation )
```

This method performs a single associative recall step.

Parameters

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

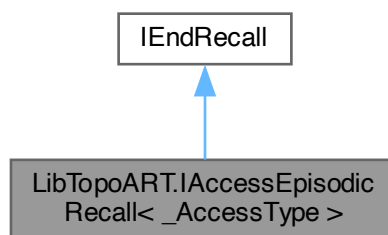
Returns

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

5.16 LibTopoART.IAccessEpisodicRecall< _AccessType > 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< _AccessType >:

**Public Member Functions**

- long [BeginRecall](#) (`_AccessType[]` stimulus)
This method starts the recall process.
- bool [InterEpisodeRecallStep](#) (out `_AccessType[]`? recall_result, out decimal F3_activation)
This method performs a single inter-episode recall step and sets the starting point for intra-episode recall.
- bool [IntraEpisodeRecallStep](#) (out `_AccessType[]`? recall_result)
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.

5.16.1 Detailed Description

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

Type Constraints

`_AccessType` : *struct*
`_AccessType` : *Convertible*

5.16.2 Member Function Documentation

5.16.2.1 BeginRecall() `long LibTopoART.IAccessEpisodicRecall< _AccessType >.BeginRecall (_AccessType[] 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.

5.16.2.2 InterEpisodeRecallStep() `bool LibTopoART.IAccessEpisodicRecall< _AccessType >.InterEpisodeRecallStep (out _AccessType?[] recall_result, out decimal F3_activation)`

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

Parameters

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

Returns

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

5.16.2.3 IntraEpisodeRecallStep() `bool LibTopoART.IAccessEpisodicRecall< _AccessType >.IntraEpisodeRecallStep (out _AccessType?[] recall_result)`

This method performs a single intra-episode recall step.

Parameters

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

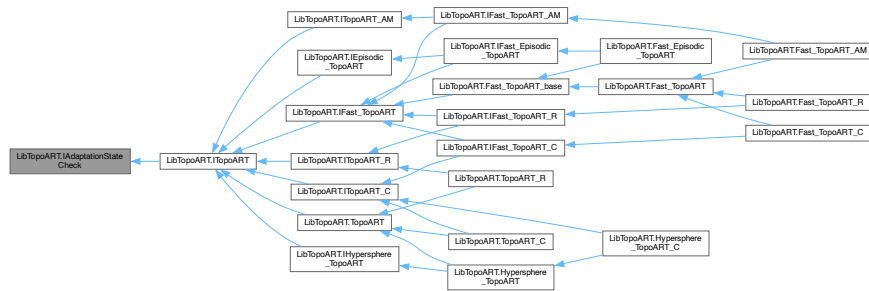
Returns

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

5.17 LibTopoART.IAdaptationStateCheck Interface Reference

Interface enabling checks whether certain adaptations of a network occurred.

Inheritance diagram for LibTopoART.IAdaptationStateCheck:



Public Member Functions

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

5.17.1 Detailed Description

Interface enabling checks whether certain adaptations of a network occurred.

5.17.2 Member Function Documentation

```
5.17.2.1 GetAdaptationState() AdaptationState LibTopoART.IAdaptationStateCheck.GetAdaptation←
State (
    decimal epsilon = 0.001m )
```

This method returns the current adaptation state.

Parameters

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

Returns

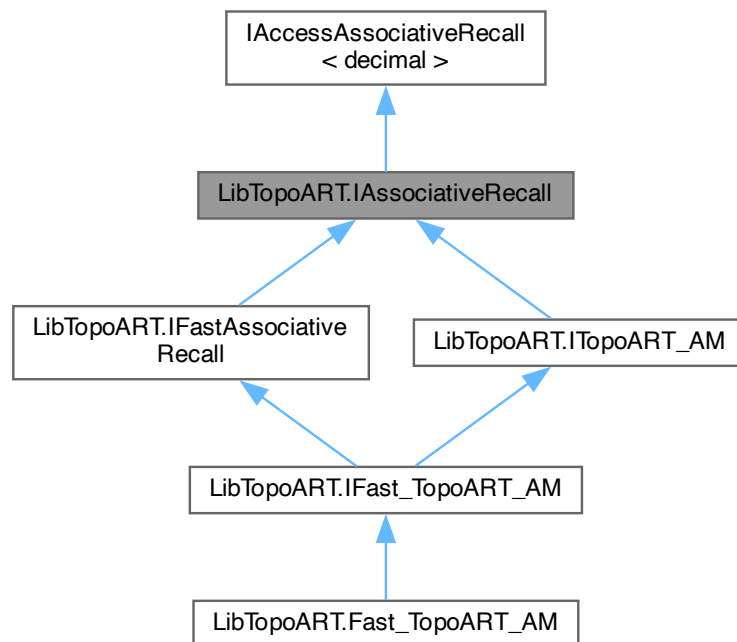
An enumeration describing the adaptation state.

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

5.18 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< decimal >](#)

- long [BeginRecallKey1](#) (_AccessType[] key_2_vec, long module_index=LibTopoART_info.FINAL_MODULE)
This method starts the recall process for the first key vector.
- long [BeginRecallKey2](#) (_AccessType[] key_1_vec, long module_index=LibTopoART_info.FINAL_MODULE)
This method starts the recall process for the second key vector.
- bool [RecallStep](#) (out _AccessType[]? recall_result, out decimal F3_activation)
This method performs a single associative recall step.

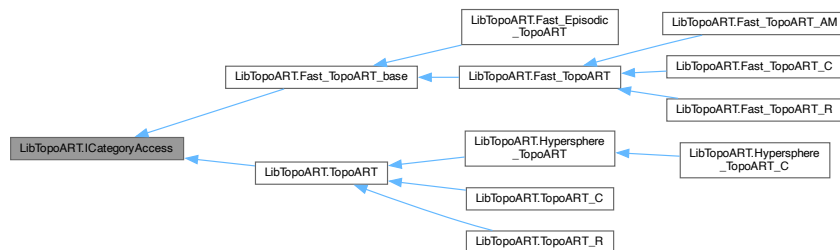
5.18.1 Detailed Description

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

5.19 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 module_index=[LibTopoART_info.FINAL_MODULE](#))
This method collects information on the categories of a specified module.

5.19.1 Detailed Description

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

5.19.2 Member Function Documentation

5.19.2.1 GetCategories() List< [CategoryInfo](#) >? LibTopoART.ICategoryAccess.GetCategories (long module_index = [LibTopoART_info.FINAL_MODULE](#))

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

Parameters

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

Returns

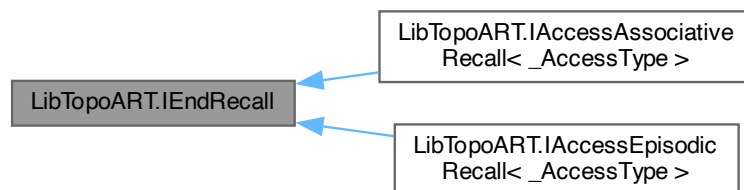
A list containing information about the respective categories.

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

5.20 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.

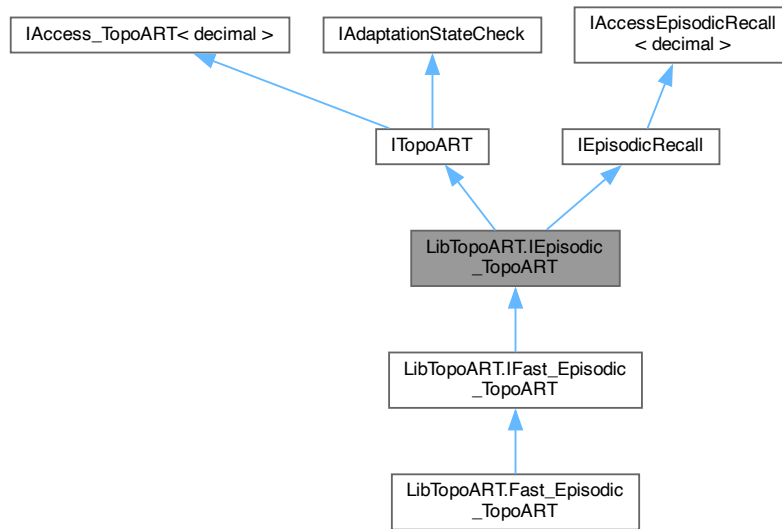
5.20.1 Detailed Description

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

5.21 LibTopoART.IEpisodic_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 `decimal` as well as adaptation state control.

Inheritance diagram for LibTopoART.IEpisodic_TopoART:



Properties

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

Additional Inherited Members

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

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

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](#) < decimal >

- long [BeginRecall](#) (_AccessType[] stimulus)
This method starts the recall process.
- bool [InterEpisodeRecallStep](#) (out _AccessType[]? recall_result, out decimal F3_activation)
This method performs a single inter-episode recall step and sets the starting point for intra-episode recall.
- bool [IntraEpisodeRecallStep](#) (out _AccessType[]? recall_result)
This method performs a single intra-episode recall step.

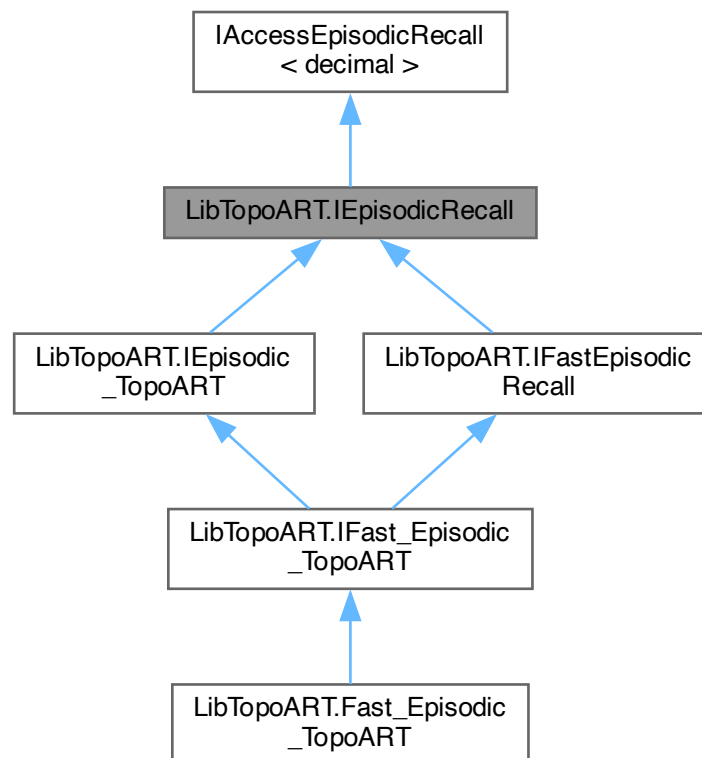
5.21.1 Detailed Description

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.

5.22 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< decimal >

- long [BeginRecall](#) (_AccessType[] stimulus)
This method starts the recall process.
- bool [InterEpisodeRecallStep](#) (out _AccessType[]? recall_result, out decimal F3_activation)
This method performs a single inter-episode recall step and sets the starting point for intra-episode recall.
- bool [IntraEpisodeRecallStep](#) (out _AccessType[]? recall_result)
This method performs a single intra-episode recall step.

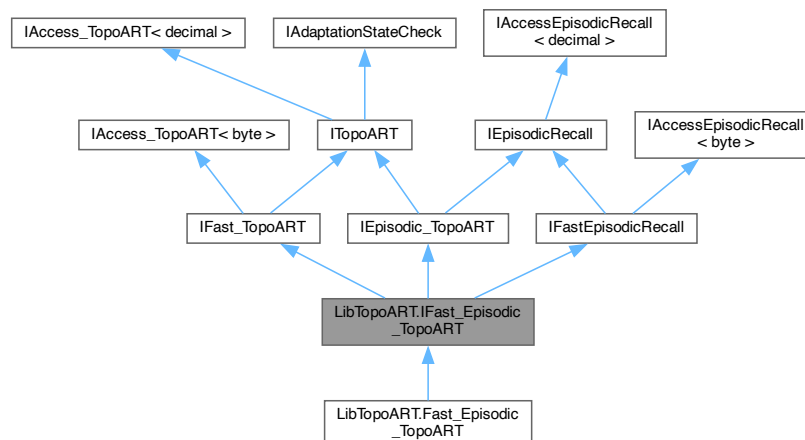
5.22.1 Detailed Description

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

5.23 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< decimal >

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

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< decimal >](#)

- long [BeginRecall](#) (_AccessType[] stimulus)
This method starts the recall process.
- bool [InterEpisodeRecallStep](#) (out _AccessType[]? recall_result, out decimal F3_activation)
This method performs a single inter-episode recall step and sets the starting point for intra-episode recall.
- bool [IntraEpisodeRecallStep](#) (out _AccessType[]? recall_result)
This method performs a single intra-episode recall step.

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

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

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

- long [BeginRecall](#) (_AccessType[] stimulus)
This method starts the recall process.
- bool [InterEpisodeRecallStep](#) (out _AccessType[]? recall_result, out decimal F3_activation)
This method performs a single inter-episode recall step and sets the starting point for intra-episode recall.
- bool [IntraEpisodeRecallStep](#) (out _AccessType[]? recall_result)
This method performs a single intra-episode recall step.

Properties inherited from [LibTopoART.IEpisodic_TopoART](#)

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

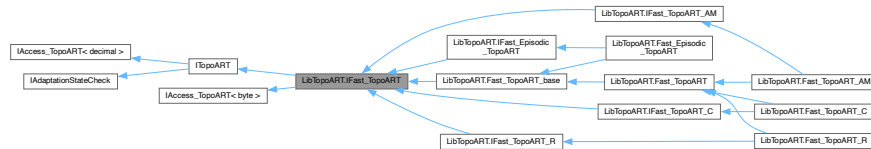
5.23.1 Detailed Description

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.

5.24 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< decimal >](#)

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

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< byte >](#)

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

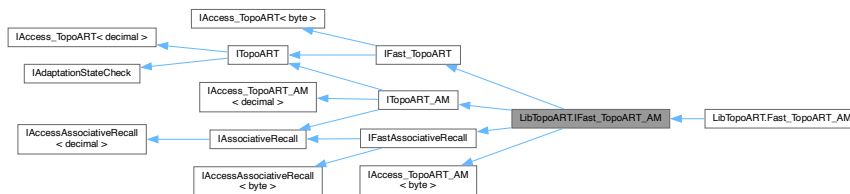
5.24.1 Detailed Description

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.

5.25 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< decimal >

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

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< decimal >

- `F2_output[] GetBMOutput (_AccessType[] key_1_vec, _AccessType[] key_2_vec)`
This method finds the closest category for a given pair of keys.
- `void Learn (_AccessType[] key_1_vec, _AccessType[] key_2_vec)`
This method performs a single training step.

Public Member Functions inherited from LibTopoART.IAccessAssociativeRecall< decimal >

- `long BeginRecallKey1 (_AccessType[] key_2_vec, long module_index=LibTopoART_info.FINAL_MODULE)`
This method starts the recall process for the first key vector.
- `long BeginRecallKey2 (_AccessType[] key_1_vec, long module_index=LibTopoART_info.FINAL_MODULE)`
This method starts the recall process for the second key vector.
- `bool RecallStep (out _AccessType[]? recall_result, out decimal F3_activation)`
This method performs a single associative recall step.

Public Member Functions inherited from LibTopoART.IAccess_TopoART< byte >

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

Public Member Functions inherited from LibTopoART.IAccess_TopoART_AM< byte >

- **F2_output[]** **GetBMOutput** (_AccessType[] key_1_vec, _AccessType[] key_2_vec)
This method finds the closest category for a given pair of keys.
- void **Learn** (_AccessType[] key_1_vec, _AccessType[] key_2_vec)
This method performs a single training step.

Public Member Functions inherited from LibTopoART.IAccessAssociativeRecall< byte >

- long **BeginRecallKey1** (_AccessType[] key_2_vec, long module_index=LibTopoART_info.FINAL_MODULE)
This method starts the recall process for the first key vector.
- long **BeginRecallKey2** (_AccessType[] key_1_vec, long module_index=LibTopoART_info.FINAL_MODULE)
This method starts the recall process for the second key vector.
- bool **RecallStep** (out _AccessType[]? recall_result, out decimal F3_activation)
This method performs a single associative recall step.

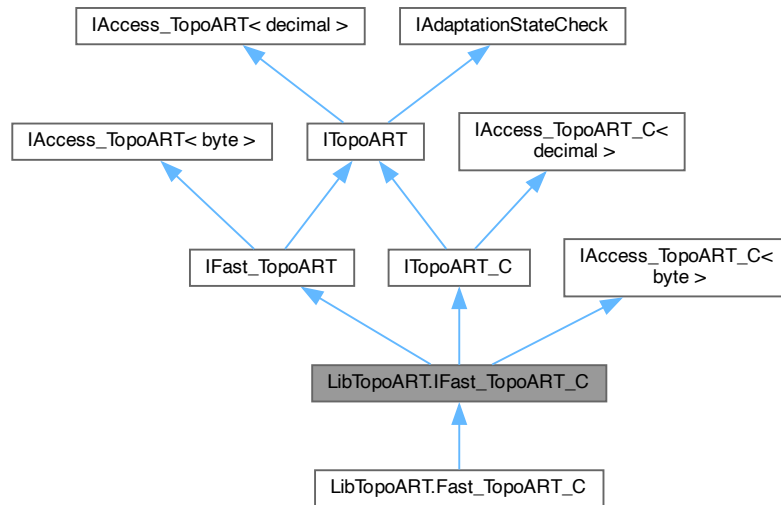
5.25.1 Detailed Description

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.

5.26 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< decimal >

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

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< decimal >

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

Public Member Functions inherited from LibTopoART.IAccess_TopoART< byte >

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

Public Member Functions inherited from LibTopoART.IAccess_TopoART_C< byte >

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

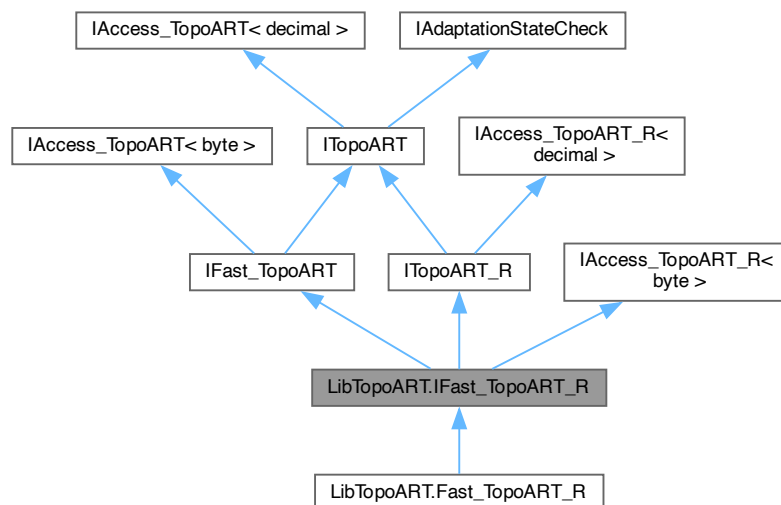
5.26.1 Detailed Description

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.

5.27 LibTopoART.IFast_TopoART_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_TopoART_R:



Additional Inherited Members

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

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

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< decimal >](#)

- void [Learn](#) ([_AccessType\[\]](#) i_vec, [_AccessType\[\]](#) d_vec)
This method performs a single training step.
- [_AccessType\[\] Predict](#) ([_AccessType\[\]](#) i_vec)
This method predicts the dependent variables using the default value of nu.
- [_AccessType\[\] Predict](#) ([_AccessType\[\]](#) i_vec, long nu)
This method predicts the dependent variables using a custom value of nu.
- [TopoART_R_prediction< _AccessType > Predict](#) ([_AccessType\[\]](#) i_vec, [bool\[\]](#) m_i_vec)
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 m_i_vec to true.
- [TopoART_R_prediction< _AccessType > Predict](#) ([_AccessType\[\]](#) i_vec, [bool\[\]](#) m_i_vec, 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 m_i_vec to true.

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

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

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

- void [Learn](#) (_AccessType[] i_vec, _AccessType[] d_vec)
This method performs a single training step.
- _AccessType[] [Predict](#) (_AccessType[] i_vec)
This method predicts the dependent variables using the default value of nu.
- _AccessType[] [Predict](#) (_AccessType[] i_vec, long nu)
This method predicts the dependent variables using a custom value of nu.
- [TopoART_R_prediction](#)< _AccessType > [Predict](#) (_AccessType[] i_vec, bool[] m_i_vec)
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 m_i_vec to true.
- [TopoART_R_prediction](#)< _AccessType > [Predict](#) (_AccessType[] i_vec, bool[] m_i_vec, 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 m_i_vec to true.

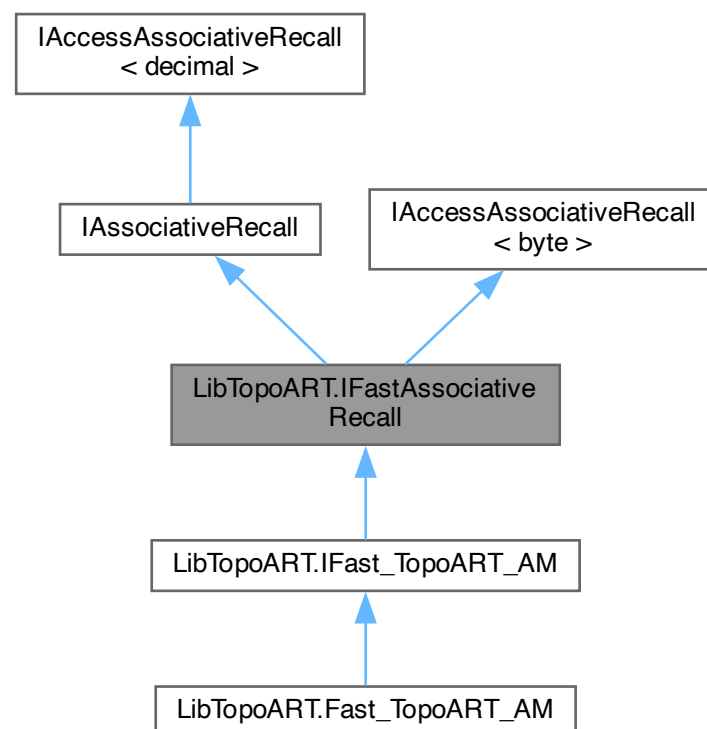
5.27.1 Detailed Description

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.

5.28 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](#) < decimal >

- long [BeginRecallKey1](#) (_AccessType[] key_2_vec, long module_index=LibTopoART_info.FINAL_MODULE)
This method starts the recall process for the first key vector.
- long [BeginRecallKey2](#) (_AccessType[] key_1_vec, long module_index=LibTopoART_info.FINAL_MODULE)
This method starts the recall process for the second key vector.
- bool [RecallStep](#) (out _AccessType[]? recall_result, out decimal F3_activation)
This method performs a single associative recall step.

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

- long [BeginRecallKey1](#) (_AccessType[] key_2_vec, long module_index=LibTopoART_info.FINAL_MODULE)
This method starts the recall process for the first key vector.
- long [BeginRecallKey2](#) (_AccessType[] key_1_vec, long module_index=LibTopoART_info.FINAL_MODULE)
This method starts the recall process for the second key vector.
- bool [RecallStep](#) (out _AccessType[]? recall_result, out decimal F3_activation)
This method performs a single associative recall step.

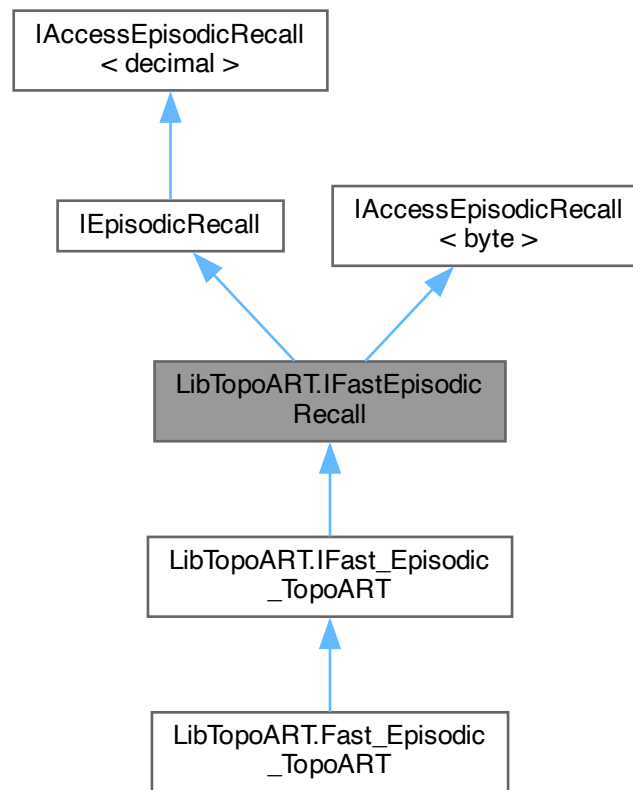
5.28.1 Detailed Description

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

5.29 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< decimal >

- long [BeginRecall](#) (_AccessType[] stimulus)
This method starts the recall process.
- bool [InterEpisodeRecallStep](#) (out _AccessType[]? recall_result, out decimal F3_activation)
This method performs a single inter-episode recall step and sets the starting point for intra-episode recall.
- bool [IntraEpisodeRecallStep](#) (out _AccessType[]? recall_result)
This method performs a single intra-episode recall step.

Public Member Functions inherited from LibTopoART.IAccessEpisodicRecall< byte >

- long [BeginRecall](#) (_AccessType[] stimulus)
This method starts the recall process.
- bool [InterEpisodeRecallStep](#) (out _AccessType[]? recall_result, out decimal F3_activation)
This method performs a single inter-episode recall step and sets the starting point for intra-episode recall.
- bool [IntraEpisodeRecallStep](#) (out _AccessType[]? recall_result)
This method performs a single intra-episode recall step.

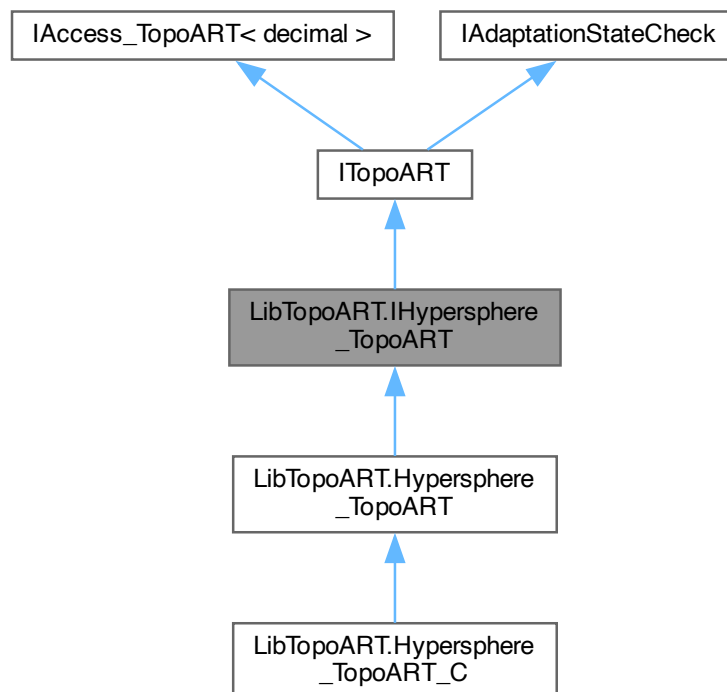
5.29.1 Detailed Description

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

5.30 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*.

Additional Inherited Members

Public Member Functions inherited from LibTopoART.IAccess_TopoART< decimal >

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

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.

5.30.1 Detailed Description

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

5.31 LibTopoART.InvalidClassIDException Class Reference

Exception signalling an invalid class ID.

Inherits `System.Exception`.

5.31.1 Detailed Description

Exception signalling an invalid class ID.

5.32 LibTopoART.InvalidFileException Class Reference

Exception signalling an invalid file.

Inherits `System.Exception`.

5.32.1 Detailed Description

Exception signalling an invalid file.

5.33 LibTopoART.InvalidModuleIndexException Class Reference

Exception signalling an invalid module index.

Inherits `System.Exception`.

5.33.1 Detailed Description

Exception signalling an invalid module index.

5.34 LibTopoART.InvalidNumberException Class Reference

Exception signalling an invalid number.

Inherits System.Exception.

5.34.1 Detailed Description

Exception signalling an invalid number.

5.35 LibTopoART.InvalidSizeException Class Reference

Exception signalling an invalid size.

Inherits System.Exception.

5.35.1 Detailed Description

Exception signalling an invalid size.

5.36 LibTopoART.InvalidStateException Class Reference

Exception signalling an invalid state of the neural network.

Inherits System.Exception.

5.36.1 Detailed Description

Exception signalling an invalid state of the neural network.

5.37 LibTopoART.InvalidTypeException Class Reference

Exception signalling an invalid type.

Inherits System.Exception.

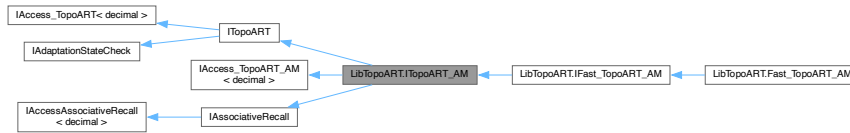
5.37.1 Detailed Description

Exception signalling an invalid type.

5.39 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< decimal >](#)

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

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< decimal >](#)

- `F2_output[] GetBMOutput (_AccessType[] key_1_vec, _AccessType[] key_2_vec)`
This method finds the closest category for a given pair of keys.
- `void Learn (_AccessType[] key_1_vec, _AccessType[] key_2_vec)`
This method performs a single training step.

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

- `long BeginRecallKey1 (_AccessType[] key_2_vec, long module_index=LibTopoART_info.FINAL_MODULE)`
This method starts the recall process for the first key vector.
- `long BeginRecallKey2 (_AccessType[] key_1_vec, long module_index=LibTopoART_info.FINAL_MODULE)`
This method starts the recall process for the second key vector.
- `bool RecallStep (out _AccessType[]? recall_result, out decimal F3_activation)`
This method performs a single associative recall step.

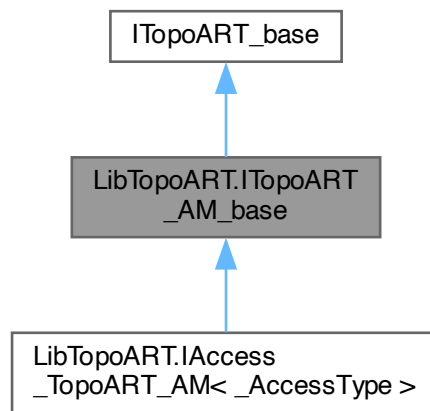
5.39.1 Detailed Description

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.

5.40 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 **Key_1_len** [get]
Property `Key_1_len` returns the length of the first key vector.
- long **Key_2_len** [get]
Property `Key_2_len` 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.

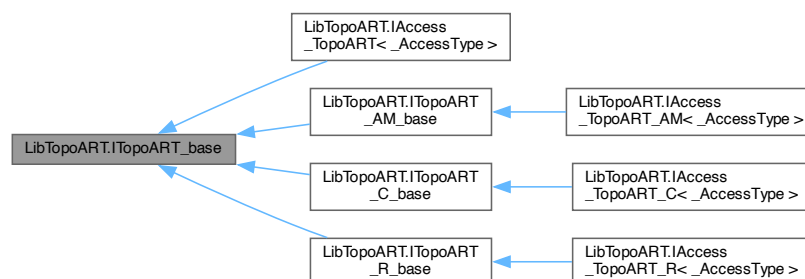
5.40.1 Detailed Description

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

5.41 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.

5.41.1 Detailed Description

Interface summarising the basic TopoART functionality excluding learning and prediction.

5.41.2 Member Function Documentation

5.41.2.1 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.)

5.41.2.2 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.
-------------	---

5.41.3 Property Documentation

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

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

5.41.3.2 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.

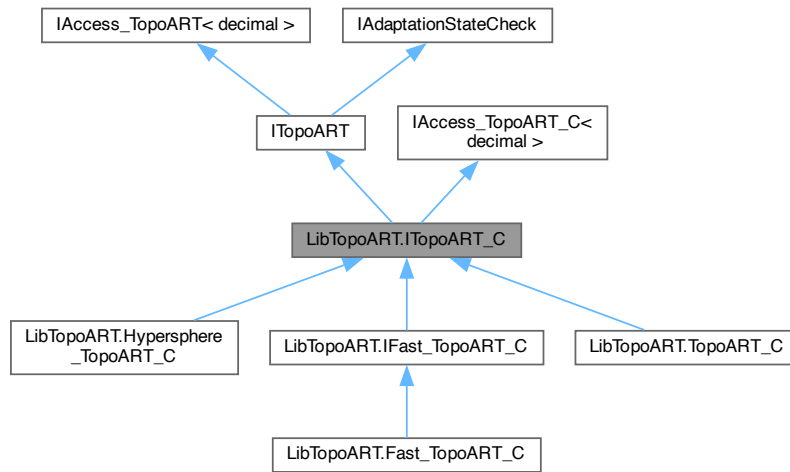
5.41.3.3 Phis `long [] LibTopoART.ITopoART_base.Phis [get], [set]`

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 for the propagation of input to subsequent TopoART modules can be controlled in a task dependent manner.

5.42 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< decimal >](#)

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

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< decimal >](#)

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

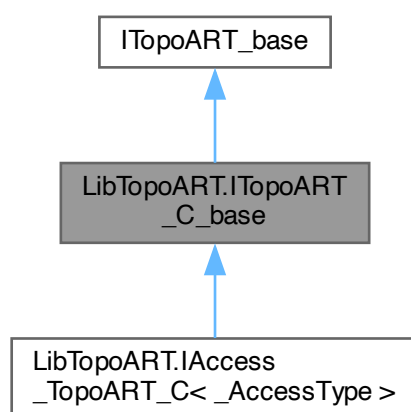
5.42.1 Detailed Description

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

5.43 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 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.

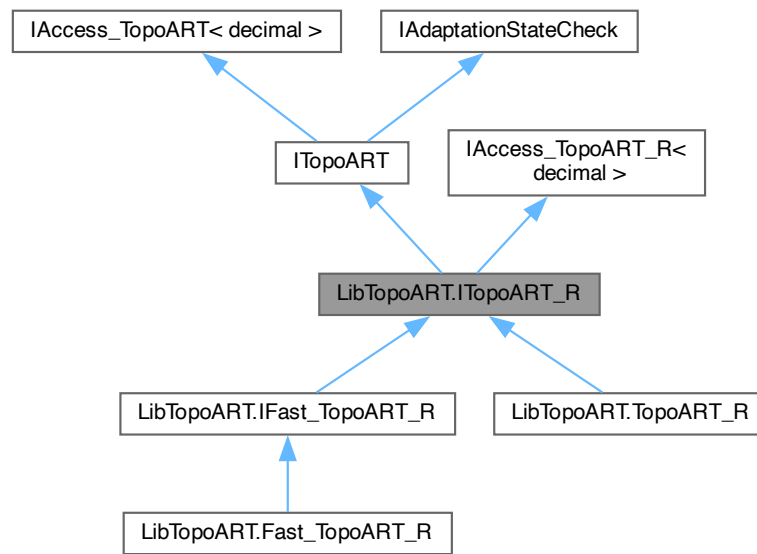
5.43.1 Detailed Description

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

5.44 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< decimal >](#)

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

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< decimal >](#)

- void [Learn](#) ([_AccessType\[\]](#) i_vec, [_AccessType\[\]](#) d_vec)
This method performs a single training step.
- [_AccessType\[\]](#) [Predict](#) ([_AccessType\[\]](#) i_vec)
This method predicts the dependent variables using the default value of nu.

- `_AccessType[] Predict (_AccessType[] i_vec, long nu)`
This method predicts the dependent variables using a custom value of `nu`.
- `TopoART_R_prediction<_AccessType> Predict (_AccessType[] i_vec, bool[] m_i_vec)`
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 `m_i_vec` to `true`.
- `TopoART_R_prediction<_AccessType> Predict (_AccessType[] i_vec, bool[] m_i_vec, 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 `m_i_vec` to `true`.

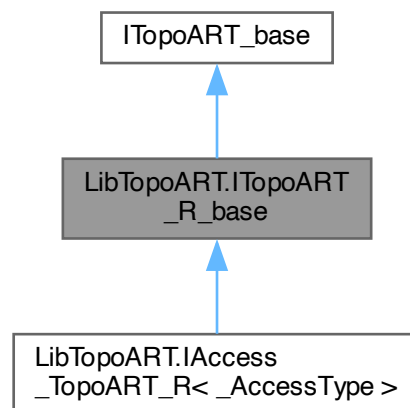
5.44.1 Detailed Description

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.

5.45 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.

5.45.1 Detailed Description

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

5.46 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](#).

5.46.1 Detailed Description

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

5.47 LibTopoART.LibTopoART_info Struct Reference

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

Static Public Attributes

- const decimal **version** = 0.97m
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.

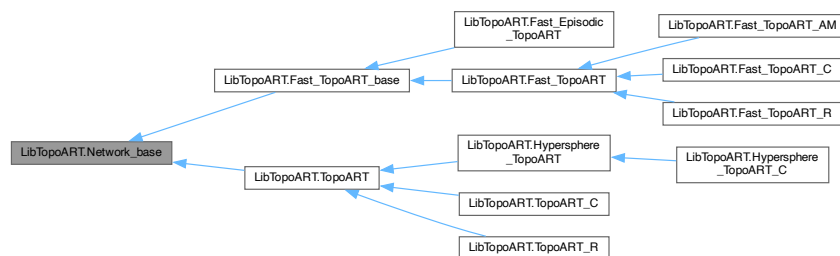
5.47.1 Detailed Description

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

5.48 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]
Property `ModuleNum` represents the number of TopoART modules used. (The original TopoART uses two modules.)
- 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.

5.48.1 Detailed Description

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

5.48.2 Property Documentation

5.48.2.1 **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.

5.48.2.2 **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 for the propagation of input to subsequent TopoART modules can be controlled in a task dependent manner.

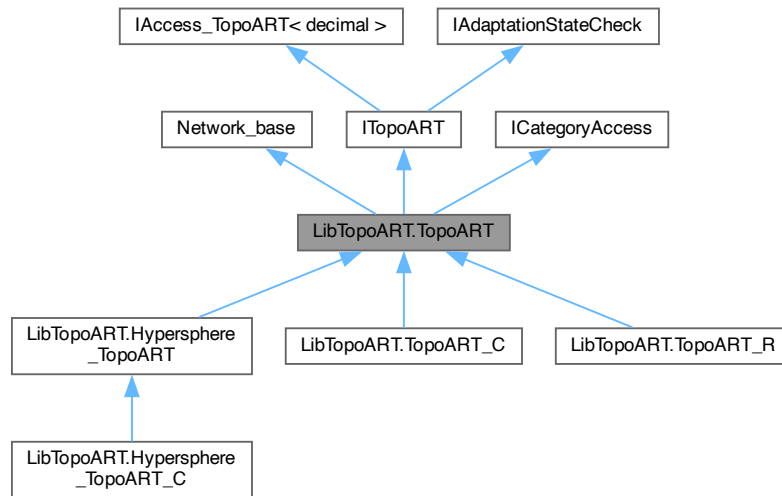
Exceptions

<i><code>InvalidSizeException</code></i>	Throws when the array length does not fit the module number.
--	--

5.49 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 input_length, long module_number, decimal rho_a)
This constructor initialises a TopoART network.
- [TopoART](#) (string path)
This constructor loads a saved TopoART network.
- void [Dispose](#) ()
Releases all resources used by the [LibTopoART.TopoART](#) object.
- void [ComputeClusterIDs](#) ()
This method computes the cluster IDs for all neurons.
- [F2_output\[\] GetBMOutput](#) (decimal[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (decimal[] input, bool[]? mask_vector)
This method finds the closest category for a given test input.
- virtual void [Learn](#) (decimal[] input)
This method performs a single training step.
- void [SaveText](#) (string path)
This method saves the entire network as a text file.
- void [Save](#) (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void [ResetAdaptationState](#) ()
This method resets the adaptation state to `AdaptationState.NO_ADAPTATION`.
- [AdaptationState GetAdaptationState](#) (decimal epsilon=0.001m)
This method returns the current adaptation state.
- List< [CategoryInfo](#) >? [GetCategories](#) (long module_index=FINAL_MODULE)
This method collects information on the categories of a specified module.

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

- [F2_output\[\] GetBMOutput](#) ([_AccessType\[\]](#) input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) ([_AccessType\[\]](#) input, [bool\[\]](#) mask_vector)
This method finds the closest category for a given test input.
- void [Learn](#) ([_AccessType\[\]](#) input)
This method performs a single training step.
- 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 module_index=[LibTopoART_info.FINAL_MODULE](#))
This method collects information on the categories of a specified module.

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

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.

5.49.1 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 $[0, 1]$.

5.49.2 Constructor & Destructor Documentation

5.49.2.1 TopoART() [1/2] `LibTopoART.TopoART.TopoART (`
`long input_length,`
`long module_number,`
`decimal rho_a)`

This constructor initialises a TopoART network.

Parameters

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

5.49.2.2 TopoART() [2/2] `LibTopoART.TopoART.TopoART (`
`string path)`

This constructor loads a saved TopoART network.

Parameters

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

Exceptions

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

5.49.3 Member Function Documentation

5.49.3.1 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.

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

Implements [LibTopoART.IAdaptationStateCheck](#).

5.49.3.3 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.

5.49.3.4 GetBMOutput() [2/2] `F2_output[] LibTopoART.TopoART.GetBMOutput (decimal[] input, bool?[] mask_vector)`

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

Parameters

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

Returns

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

5.49.3.5 GetCategories() `List< CategoryInfo >? LibTopoART.TopoART.GetCategories (long module_index = FINAL_MODULE)`

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

Parameters

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

Implements [LibTopoART.ICategoryAccess](#).

5.49.3.6 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_TopoART_C](#), [LibTopoART.TopoART_C](#), and [LibTopoART.TopoART_R](#).

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

This method resets the adaptation state to `AdaptationState.NO_ADAPTATION`.

Exceptions

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

Implements [LibTopoART.IAdaptationStateCheck](#).

5.49.3.8 Save() `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.)

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

This method saves the entire network as a text file.

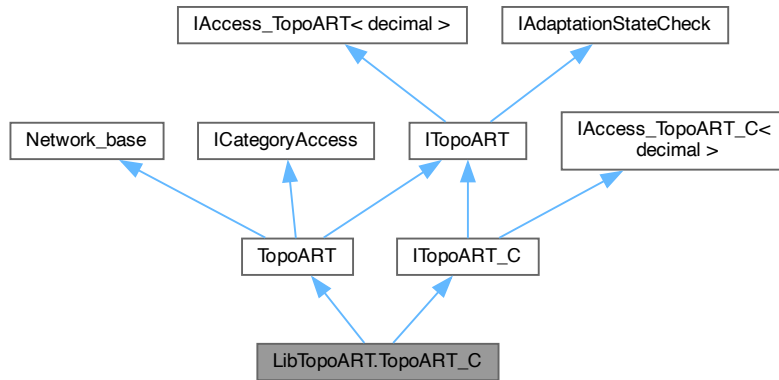
Parameters

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

5.50 LibTopoART.TopoART_C Class Reference

Class `TopoART_C` provides an implementation of the TopoART-C neural network as proposed in "Marko Tscherepanow and Sören Riechers (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."

Inheritance diagram for LibTopoART.TopoART_C:



Public Member Functions

- `TopoART_C` (long input_length, long module_number, decimal rho_a)
This constructor initialises a TopoART-C network.
- `TopoART_C` (string path)
This constructor loads a saved TopoART-C network.
- override void `Learn` (decimal[] input)
This method performs a single training step and sets the class ID corresponding to input to `UNDEFINED_CLASS`↔`_ID`.
- void `Learn` (decimal[] input, long class_ID)
This method performs a single training step.
- long `Predict` (decimal[] input)
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_vector)
This method predicts the class ID using the default value of nu.
- `TopoART_C_prediction Predict` (decimal[] input, bool[]? mask_vector, long nu)
This method predicts the class ID using a custom value of nu.

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

- [TopoART](#) (long input_length, long module_number, decimal rho_a)
This constructor initialises a TopoART network.
- [TopoART](#) (string path)
This constructor loads a saved TopoART network.
- void [Dispose](#) ()
Releases all resources used by the [LibTopoART.TopoART](#) object.
- void [ComputeClusterIDs](#) ()
This method computes the cluster IDs for all neurons.
- [F2_output\[\] GetBMOutput](#) (decimal[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (decimal[] input, bool[]? mask_vector)
This method finds the closest category for a given test input.
- virtual void [Learn](#) (decimal[] input)
This method performs a single training step.
- void [SaveText](#) (string path)
This method saves the entire network as a text file.
- void [Save](#) (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void [ResetAdaptationState](#) ()
This method resets the adaptation state to `AdaptationState.NO_ADAPTATION`.
- [AdaptationState GetAdaptationState](#) (decimal epsilon=0.001m)
This method returns the current adaptation state.
- List< [CategoryInfo](#) >? [GetCategories](#) (long module_index=[FINAL_MODULE](#))
This method collects information on the categories of a specified module.

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

- [F2_output\[\] GetBMOutput](#) (_AccessType[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (_AccessType[] input, bool[] mask_vector)
This method finds the closest category for a given test input.
- void [Learn](#) (_AccessType[] input)
This method performs a single training step.
- 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 module_index=[LibTopoART_info.FINAL_MODULE](#))
This method collects information on the categories of a specified module.

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

- void [Learn](#) (_AccessType[] input, long class_ID)
This method performs a single training step.
- long [Predict](#) (_AccessType[] input)
This method predicts the class ID using the default value of nu.
- long [Predict](#) (_AccessType[] input, long nu)
This method predicts the class ID using a custom value of nu.
- [TopoART_C_prediction Predict](#) (_AccessType[] input, bool[] mask_vector)
This method predicts the class ID using the default value of nu.
- [TopoART_C_prediction Predict](#) (_AccessType[] input, bool[] mask_vector, 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 predict 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** [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)integer_type]` [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)float_type]` [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]
Property `ModuleNum` represents the number of TopoART modules used. (The original TopoART uses two modules.)
- 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.

5.50.1 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.

5.50.2 Constructor & Destructor Documentation

5.50.2.1 TopoART_C() [1/2] `LibTopoART.TopoART_C.TopoART_C (`
 `long input_length,`
 `long module_number,`
 `decimal rho_a)`

This constructor initialises a TopoART-C network.

Parameters

<i>input_length</i>	The length of input vectors to be learnt.
<i>module_number</i>	The number of TopoART-C modules.
<i>rho_a</i>	The vigilance parameter of the first TopoART-C module (TopoART-C ₁).

5.50.2.2 TopoART_C() [2/2] `LibTopoART.TopoART_C.TopoART_C (`
`string path)`

This constructor loads a saved TopoART-C network.

Parameters

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

Exceptions

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

5.50.3 Member Function Documentation

5.50.3.1 Learn() [1/2] `override void LibTopoART.TopoART_C.Learn (`
`decimal[] input) [virtual]`

This method performs a single training step and sets the class ID corresponding to *input* to `UNDEFINED_CLASS←_ID`.

Parameters

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

Reimplemented from [LibTopoART.TopoART](#).

5.50.3.2 Learn() [2/2] `void LibTopoART.TopoART_C.Learn (`
`decimal[] input,`
`long class_ID)`

This method performs a single training step.

Parameters

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

Exceptions

<i>InvalidClassIDException</i>	Throws when <i>class_ID</i> is less than 0.
--------------------------------	---

5.50.3.3 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.

5.50.3.4 Predict() [2/4] `TopoART_C_prediction LibTopoART.TopoART_C.Predict (decimal[] input, bool?[] mask_vector)`

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_vector</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.

5.50.3.5 Predict() [3/4] `TopoART_C_prediction LibTopoART.TopoART_C.Predict (decimal[] input, bool?[] mask_vector, 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_vector</i>	The mask vector corresponding to <i>input</i> .
<i>nu</i>	The maximum cardinality of the set of enclosing categories E and the neighbourhood set N. (This parameter does not modify the network. It may be arbitrarily changed in each prediction step.)

Returns

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

5.50.3.6 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.

5.51 LibTopoART.TopoART_C_prediction Struct Reference

Struct `TopoART_C_prediction` contains a prediction made by a TopoART-C network.

Public Member Functions

- [TopoART_C_prediction](#) (long [class_ID](#), decimal [confidence](#))
This constructor sets the instance variables `class_ID` and `confidence` of struct `TopoART_C_prediction`.

Public Attributes

- readonly long **class_ID**
Instance variable `class_ID` gives the predicted class ID.
- readonly decimal **confidence**
Instance variable `confidence` provides a confidence for the predicted class ID.

5.51.1 Detailed Description

Struct `TopoART_C_prediction` contains a prediction made by a TopoART-C network.

5.51.2 Constructor & Destructor Documentation

5.51.2.1 TopoART_C_prediction() `LibTopoART.TopoART_C_prediction.TopoART_C_prediction (`
`long class_ID,`
`decimal confidence)`

This constructor sets the instance variables `class_ID` and `confidence` of struct `TopoART_C_prediction`.

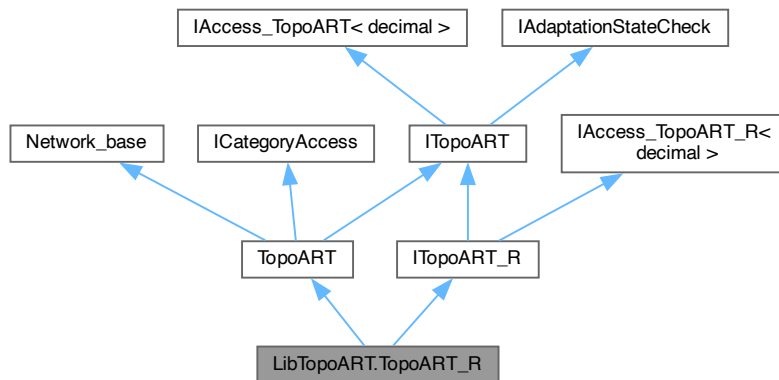
Parameters

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

5.52 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 *i_length*, long *d_length*, long *module_number*, decimal *rho_a*)
This constructor initialises a TopoART-R network.
- `TopoART_R` (string *path*)
This constructor loads a saved TopoART-R network.
- override void `Learn` (decimal[] *input*)
This method performs a single training step. The independent variables and the dependent variables are automatically separated.
- void `Learn` (decimal[] *i_vec*, decimal[] *d_vec*)
This method performs a single training step.
- decimal[] `Predict` (decimal[] *i_vec*)
This method predicts the dependent variables using the default value of nu.
- decimal[] `Predict` (decimal[] *i_vec*, long *nu*)
This method predicts the dependent variables using a custom value of nu.
- `TopoART_R_prediction`< decimal > `Predict` (decimal[] *i_vec*, bool[] *m_i_vec*)
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 m_i_vec to true.
- `TopoART_R_prediction`< decimal > `Predict` (decimal[] *i_vec*, bool[] *m_i_vec*, 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 m_i_vec to true.

Public Member Functions inherited from LibTopoART.TopoART

- [TopoART](#) (long input_length, long module_number, decimal rho_a)
This constructor initialises a TopoART network.
- [TopoART](#) (string path)
This constructor loads a saved TopoART network.
- void [Dispose](#) ()
Releases all resources used by the LibTopoART.TopoART object.
- void [ComputeClusterIDs](#) ()
This method computes the cluster IDs for all neurons.
- [F2_output\[\] GetBMOutput](#) (decimal[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (decimal[] input, bool[]? mask_vector)
This method finds the closest category for a given test input.
- virtual void [Learn](#) (decimal[] input)
This method performs a single training step.
- void [SaveText](#) (string path)
This method saves the entire network as a text file.
- void [Save](#) (string path, CompressionLevel compression=CompressionLevel.Fastest)
This method saves the entire network as a binary file.
- void [ResetAdaptationState](#) ()
This method resets the adaptation state to `AdaptationState.NO_ADAPTATION`.
- [AdaptationState GetAdaptationState](#) (decimal epsilon=0.001m)
This method returns the current adaptation state.
- List< [CategoryInfo](#) >? [GetCategories](#) (long module_index=FINAL_MODULE)
This method collects information on the categories of a specified module.

Public Member Functions inherited from LibTopoART.IAccess_TopoART< decimal >

- [F2_output\[\] GetBMOutput](#) (_AccessType[] input)
This method finds the closest category for a given test input.
- [F2_output\[\] GetBMOutput](#) (_AccessType[] input, bool[] mask_vector)
This method finds the closest category for a given test input.
- void [Learn](#) (_AccessType[] input)
This method performs a single training step.
- 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 module_index=LibTopoART_info.FINAL_MODULE)
This method collects information on the categories of a specified module.

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

- void [Learn](#) ([_AccessType\[\]](#) i_vec, [_AccessType\[\]](#) d_vec)
This method performs a single training step.
- [_AccessType\[\] Predict](#) ([_AccessType\[\]](#) i_vec)
This method predicts the dependent variables using the default value of nu.
- [_AccessType\[\] Predict](#) ([_AccessType\[\]](#) i_vec, long nu)
This method predicts the dependent variables using a custom value of nu.
- [TopoART_R_prediction< _AccessType > Predict](#) ([_AccessType\[\]](#) i_vec, [bool\[\]](#) m_i_vec)
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 m_i_vec to true.
- [TopoART_R_prediction< _AccessType > Predict](#) ([_AccessType\[\]](#) i_vec, [bool\[\]](#) m_i_vec, 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 m_i_vec 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** [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\[\(int\)integer_type\]](#) [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\[\(int\)float_type\]](#) [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]
Property ModuleNum represents the number of TopoART modules used. (The original TopoART uses two modules.)
- 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.

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.

5.52.1 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].

5.52.2 Constructor & Destructor Documentation

5.52.2.1 TopoART_R() [1/2] `LibTopoART.TopoART_R.TopoART_R (`
 long *i_length*,
 long *d_length*,
 long *module_number*,
 decimal *rho_a*)

This constructor initialises a TopoART-R network.

Parameters

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

5.52.2.2 TopoART_R() [2/2] `LibTopoART.TopoART_R.TopoART_R (`
`string path)`

This constructor loads a saved TopoART-R network.

Parameters

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

Exceptions

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

5.52.3 Member Function Documentation

5.52.3.1 Learn() [1/2] `void LibTopoART.TopoART_R.Learn (`
`decimal[] i_vec,`
`decimal[] d_vec)`

This method performs a single training step.

Parameters

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

5.52.3.2 Learn() [2/2] `override void LibTopoART.TopoART_R.Learn (`
`decimal[] input) [virtual]`

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

Parameters

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

Reimplemented from [LibTopoART.TopoART](#).

5.52.3.3 Predict() [1/4] `decimal[] LibTopoART.TopoART_R.Predict (decimal[] i_vec)`

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

Parameters

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

Returns

The predicted values for all dependent variables.

5.52.3.4 Predict() [2/4] `TopoART_R_prediction< decimal > LibTopoART.TopoART_R.Predict (decimal[] i_vec, bool[] m_i_vec)`

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 `m_i_vec` to `true`.

Parameters

<code>i_vec</code>	The input vector (independent variables).
<code>m_i_vec</code>	The mask vector corresponding to <code>i_vec</code> .

Returns

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

5.52.3.5 Predict() [3/4] `TopoART_R_prediction< decimal > LibTopoART.TopoART_R.Predict (decimal[] i_vec, bool[] m_i_vec, 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 `m_i_vec` to `true`.

Parameters

<code>i_vec</code>	The input vector (independent variables).
<code>m_i_vec</code>	The mask vector corresponding to <code>i_vec</code> .
<code>nu</code>	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.

5.52.3.6 Predict() [4/4] `decimal[] LibTopoART.TopoART_R.Predict (decimal[] i_vec, long nu)`

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

Parameters

<code>i_vec</code>	The input vector (independent variables).
<code>nu</code>	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.

5.53 LibTopoART.TopoART_R_prediction<_ElementType> Struct Template Reference

Struct `TopoART_R_prediction` contains a prediction made by a TopoART-R network.

Public Member Functions

- [TopoART_R_prediction](#) (`_ElementType[] i_vec_prediction, _ElementType[] d_vec_prediction`)
This constructor sets the instance variables `i_vec_prediction` and `d_vec_prediction` of struct `TopoART_R_prediction`.
- `void PrintPredictions ()`
This method prints the predictions on the console.

Public Attributes

- `readonly _ElementType 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 `_ElementType`.
- `readonly _ElementType[] i_vec_prediction`
Instance variable `i_vec_prediction` represents predictions for unknown independent variables.
- `readonly _ElementType[] d_vec_prediction`
Instance variable `d_vec_prediction` provides the predictions for the dependent variables.

5.53.1 Detailed Description

Struct `TopoART_R_prediction` contains a prediction made by a TopoART-R network.

Type Constraints

`_ElementType` : *struct*
`_ElementType` : *Convertible*

5.53.2 Constructor & Destructor Documentation

5.53.2.1 TopoART_R_prediction() `LibTopoART.TopoART_R_prediction< _ElementType >.TopoART_R_prediction`
 (
 `_ElementType[] i_vec_prediction,`
 `_ElementType[] d_vec_prediction`)

This constructor sets the instance variables `i_vec_prediction` and `d_vec_prediction` of struct `TopoART_R_prediction`.

Parameters

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

5.54 LibTopoART_samples.Episodic_TopoART_sample1 Class Reference

Episodic clustering sample using synthetic two-dimensional data. [C#].

5.54.1 Detailed Description

Episodic clustering sample using synthetic two-dimensional data. [C#].

Like in Section 4.1 of "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.", an Episodic TopoART network is trained with the well-known Two Spirals dataset. Due to the incorporation of temporal information during learning, Episodic TopoART is capable of creating two clusters each representing one spiral in an unsupervised way. These clusters are formed by the nodes of module b (ETA b).

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

5.55 LibTopoART_samples.Episodic_TopoART_sample2 Class Reference

Episodic clustering sample using real-world video data. [C#].

5.55.1 Detailed Description

Episodic clustering sample using real-world video data. [C#].

Like in Section 4.2 of "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.", an Episodic TopoART network is trained with real-world video data. Each image has a size of 64x36 pixels. As each pixel comprises 3 color channels (RGB), the input length equals 6912. After finishing training, recall is performed for a single input stimulus.

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

5.56 LibTopoART_samples.TopoART_AM_sample1 Class Reference

Sample using TopoART-AM with synthetic two-dimensional data. [C#].

5.56.1 Detailed Description

Sample using TopoART-AM with synthetic two-dimensional data. [C#].

A TopoART-AM network is trained with the well-known Two Spirals dataset augmented with additional information. The resulting network maps two-dimensional points lying on each spiral (`key_1`) to their Euclidean distance from the origin and the corresponding spiral ID (`key_2`). Therefore, it can recall spiral points if a distance and a spiral ID are given, and vice versa.

The resulting network can be visualised using the script `ShowTopoARTAMResults` provided for R and MATLAB in the subfolder `visualisation`.

5.57 LibTopoART_samples.TopoART_AM_sample2 Class Reference

Learning of bidirectional associations between images. [F#].

5.57.1 Detailed Description

Learning of bidirectional associations between images. [F#].

Similar to Section 4.2 of "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.", a TopoART-AM network is trained with real-world image data. There are two kinds of images grouped into owners and objects. TopoART-AM learns a bi-directional mapping between images of these two groups. Each image has a size of about 34500 pixels. As each pixel comprises 3 color channels (RGB) and the input vector encompasses a key from each group, the total length of the input vector is larger than 200,000. After finishing training, recall is performed for a single input stimulus of each group. The recall results are saved in the folder `results/recall/ObjectsOwners_dataset_recall_results`.

/summary>

5.58 LibTopoART_samples.TopoART_C_sample1 Class Reference

Simple classification sample. [C#].

5.58.1 Detailed Description

Simple classification sample. [C#].

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

5.59 LibTopoART_samples.TopoART_C_sample2 Class Reference

Classification sample using more complex synthetic two-dimensional data with associated class IDs. [C#].

5.59.1 Detailed Description

Classification sample using more complex synthetic two-dimensional data with associated class IDs. [C#].

Train TopoART-C with a two-dimensional dataset similar to the one used 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." This dataset comprises six clusters (each containing 15,000 samples) as well as 10,000 noise samples which were mixed randomly. The samples are divided into two classes. Each sample belonging to one of the six clusters is assigned a class ID depending on its position. In contrast, noise samples receive a random class ID.

The resulting neural network can be visualised using the script `ShowTopoARTCResults` or the script `Show↔HypersphereTopoARTCResults`, respectively. Both scripts are provided for R and MATLAB in the subfolder `visualisation`.

5.60 LibTopoART_samples.TopoART_R_sample1 Class Reference

Regression sample using TopoART-R. (simplified version) [C#].

5.60.1 Detailed Description

Regression sample using TopoART-R. (simplified version) [C#].

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

The predicted results can be visualised using the script `ShowTopoARTRResults` provided for R and MATLAB in the subfolder `visualisation`.

5.61 LibTopoART_samples.TopoART_R_sample2 Class Reference

Regression sample using TopoART-R. (advanced version) [C#].

5.61.1 Detailed Description

Regression sample using TopoART-R. (advanced version) [C#].

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

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

5.62 LibTopoART_samples.TopoART_R_sample3 Class Reference

Pixel-wise regression analysis of an image using TopoART-R. [F#].

5.62.1 Detailed Description

Pixel-wise regression analysis of an image using TopoART-R. [F#].

This sample trains a TopoART-R network with a colour image depicting the Mandelbrot set. The pixel coordinates are used as independent variables and the corresponding colour values as dependent variables. The accuracy of the regression function can be controlled by means of the vigilance parameter `rho_a`.

The training image `Mandelbrot_reference.png` and the predicted image `TopoART-R_Mandelbrot_regression.png` are written into the subfolder `results/regression`.

5.63 LibTopoART_samples.TopoART_sample1 Class Reference

Simple clustering sample. [C#].

5.63.1 Detailed Description

Simple clustering sample. [C#].

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

5.64 LibTopoART_samples.TopoART_sample2 Class Reference

Clustering sample using more complex synthetic two-dimensional data. [C#].

5.64.1 Detailed Description

Clustering sample using more complex synthetic two-dimensional data. [C#].

Train TopoART or Hypersphere TopoART with a two-dimensional dataset similar to the one used 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." This dataset comprises six clusters (each containing 15,000 samples) as well as 10,000 noise samples. These samples were mixed randomly.

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

5.65 LibTopoART_samples.TopoART_sample3 Class Reference

Clustering sample using very noisy synthetic two-dimensional data. [VB].

5.65.1 Detailed Description

Clustering sample using very noisy synthetic two-dimensional data. [VB].

Train TopoART or Hypersphere TopoART with a two-dimensional dataset similar to the one used 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." The dataset applied here comprises 1,000,000 samples equally allotted to six clusters (each containing one sixth of the samples). Additionally, 1,000,000 uniformly distributed random samples are added. Finally, all samples are randomly shuffled.

Due to the randomness involved, the results differ between different runs of this program. However, they are qualitatively comparable: The first module creates a coarse clustering of the data while the second module refines it to the six clusters of the undisturbed portion of the dataset.

These results show the abilities of TopoART to cope with a high amount of noise data and produce stable results independent of the sample order.

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

Index

- AdaptationState
 - LibTopoART, [12](#)
- 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, 21](#)
 - LibTopoART.IAccessEpisodicRecall< _AccessType >, [92](#)
- BeginRecallKey1
 - LibTopoART.Fast_TopoART_AM, [33](#)
 - LibTopoART.IAccessAssociativeRecall< _AccessType >, [89](#)
- BeginRecallKey2
 - LibTopoART.Fast_TopoART_AM, [34](#)
 - LibTopoART.IAccessAssociativeRecall< _AccessType >, [90](#)
- CategoryInfo
 - LibTopoART.CategoryInfo, [14](#)
- Dispose
 - LibTopoART.Fast_TopoART_base, [40](#)
 - LibTopoART.TopoART, [130](#)
- Fast_Episodic_TopoART
 - LibTopoART.Fast_Episodic_TopoART, [20](#)
- Fast_TopoART
 - LibTopoART.Fast_TopoART, [27](#)
- Fast_TopoART_AM
 - LibTopoART.Fast_TopoART_AM, [32, 33](#)
- Fast_TopoART_C
 - LibTopoART.Fast_TopoART_C, [49](#)
- Fast_TopoART_R
 - LibTopoART.Fast_TopoART_R, [59](#)
- GetAdaptationState
 - LibTopoART.Fast_TopoART_base, [40](#)
 - LibTopoART.IAdaptationStateCheck, [93](#)
 - LibTopoART.TopoART, [130](#)
- GetBMOutput
 - LibTopoART.Fast_TopoART_AM, [35](#)
- LibTopoART.Fast_TopoART_base, [41, 42](#)
- LibTopoART.IAccess_TopoART< _AccessType >, [77, 78](#)
- LibTopoART.IAccess_TopoART_AM< _AccessType >, [80](#)
- LibTopoART.TopoART, [130, 131](#)
- GetCategories
 - LibTopoART.Fast_TopoART_base, [42](#)
 - LibTopoART.ICategoryAccess, [95](#)
 - LibTopoART.TopoART, [131](#)
- Hypersphere_TopoART
 - LibTopoART.Hypersphere_TopoART, [67, 68](#)
- Hypersphere_TopoART_C
 - LibTopoART.Hypersphere_TopoART_C, [73](#)
- Important
 - LibTopoART, [13](#)
- InterEpisodeRecallStep
 - LibTopoART.Fast_Episodic_TopoART, [21](#)
 - LibTopoART.IAccessEpisodicRecall< _AccessType >, [92](#)
- IntraEpisodeRecallStep
 - LibTopoART.Fast_Episodic_TopoART, [22](#)
 - LibTopoART.IAccessEpisodicRecall< _AccessType >, [92](#)
- Learn
 - LibTopoART.Fast_Episodic_TopoART, [22, 23](#)
 - LibTopoART.Fast_TopoART, [27, 28](#)
 - LibTopoART.Fast_TopoART_AM, [35, 37](#)
 - LibTopoART.Fast_TopoART_base, [43](#)
 - LibTopoART.Fast_TopoART_C, [50, 51](#)
 - LibTopoART.Fast_TopoART_R, [60, 61](#)
 - LibTopoART.Hypersphere_TopoART_C, [74](#)
 - LibTopoART.IAccess_TopoART< _AccessType >, [78](#)
 - LibTopoART.IAccess_TopoART_AM< _AccessType >, [81](#)
 - LibTopoART.IAccess_TopoART_C< _AccessType >, [83](#)
 - LibTopoART.IAccess_TopoART_R< _AccessType >, [86](#)
 - LibTopoART.TopoART, [131](#)
 - LibTopoART.TopoART_C, [137](#)
 - LibTopoART.TopoART_R, [144](#)
- LibTopoART, [9](#)
 - AdaptationState, [12](#)
 - 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.CategoryInfo, 14
 - CategoryInfo, 14
- LibTopoART.F2_output, 15
- LibTopoART.Fast_Episodic_TopoART, 16
 - BeginRecall, 20, 21
 - Fast_Episodic_TopoART, 20
 - InterEpisodeRecallStep, 21
 - IntraEpisodeRecallStep, 22
 - Learn, 22, 23
- LibTopoART.Fast_TopoART, 23
 - Fast_TopoART, 27
 - Learn, 27, 28
- LibTopoART.Fast_TopoART_AM, 28
 - BeginRecallKey1, 33
 - BeginRecallKey2, 34
 - Fast_TopoART_AM, 32, 33
 - GetBMOOutput, 35
 - Learn, 35, 37
 - RecallStep, 37
- LibTopoART.Fast_TopoART_base, 38
 - Dispose, 40
 - GetAdaptationState, 40
 - GetBMOOutput, 41, 42
 - GetCategories, 42
 - Learn, 43
 - ResetAdaptationState, 43
 - Save, 44
 - 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, 61
 - Predict, 61–64
- LibTopoART.Hypersphere_TopoART, 64
 - Hypersphere_TopoART, 67, 68
- LibTopoART.Hypersphere_TopoART_C, 68
 - Hypersphere_TopoART_C, 73
 - Learn, 74
 - Predict, 74, 75
- LibTopoART.IAccess_TopoART< _AccessType >, 76
 - GetBMOOutput, 77, 78
 - Learn, 78
- LibTopoART.IAccess_TopoART_AM< _AccessType >, 78
 - GetBMOOutput, 80
 - Learn, 81
- LibTopoART.IAccess_TopoART_C< _AccessType >, 81
 - Learn, 83
 - Predict, 83, 84
- LibTopoART.IAccess_TopoART_R< _AccessType >, 85
 - Learn, 86
 - Predict, 87, 88
- LibTopoART.IAccessAssociativeRecall< _AccessType >, 88
 - BeginRecallKey1, 89
 - BeginRecallKey2, 90
 - RecallStep, 90
- LibTopoART.IAccessEpisodicRecall< _AccessType >, 91
 - BeginRecall, 92
 - InterEpisodeRecallStep, 92
 - IntraEpisodeRecallStep, 92
- LibTopoART.IAdaptationStateCheck, 93
 - GetAdaptationState, 93
- LibTopoART.IAssociativeRecall, 94
- LibTopoART.ICategoryAccess, 95
 - GetCategories, 95
- LibTopoART.IEndRecall, 96
- LibTopoART.IEpisodic_TopoART, 96
- LibTopoART.IEpisodicRecall, 98
- LibTopoART.IFast_Episodic_TopoART, 99
- LibTopoART.IFast_TopoART, 101
- LibTopoART.IFast_TopoART_AM, 102
- LibTopoART.IFast_TopoART_C, 103
- LibTopoART.IFast_TopoART_R, 105
- LibTopoART.IFastAssociativeRecall, 107
- LibTopoART.IFastEpisodicRecall, 108
- LibTopoART.IHypersphere_TopoART, 110
- LibTopoART.InvalidClassIDException, 111
- LibTopoART.InvalidFileException, 111
- LibTopoART.InvalidModuleIndexException, 111
- LibTopoART.InvalidNumberException, 112
- LibTopoART.InvalidSizeException, 112
- LibTopoART.InvalidStateException, 112
- LibTopoART.InvalidTypeException, 112
- LibTopoART.ITopoART, 113
- LibTopoART.ITopoART_AM, 114
- LibTopoART.ITopoART_AM_base, 115
- LibTopoART.ITopoART_base, 116
 - ModuleNum, 118
 - Phi, 118
 - Phis, 118
 - Save, 117
 - SaveText, 118
- LibTopoART.ITopoART_C, 118
- LibTopoART.ITopoART_C_base, 120
- LibTopoART.ITopoART_R, 121
- LibTopoART.ITopoART_R_base, 123
- LibTopoART.LibTopoART_control, 124
- LibTopoART.LibTopoART_info, 125
- LibTopoART.Network_base, 125
 - Phi, 126
 - Phis, 126
- LibTopoART.TopoART, 126
 - Dispose, 130

- GetAdaptationState, 130
- GetBMOOutput, 130, 131
- GetCategories, 131
- Learn, 131
- ResetAdaptationState, 132
- Save, 132
- SaveText, 132
- TopoART, 129
- LibTopoART.TopoART_C, 133
 - Learn, 137
 - Predict, 138, 139
 - TopoART_C, 136, 137
- LibTopoART.TopoART_C_prediction, 139
 - TopoART_C_prediction, 139
- LibTopoART.TopoART_R, 140
 - Learn, 144
 - Predict, 144–146
 - TopoART_R, 143, 144
- LibTopoART.TopoART_R_prediction< _ElementType >, 146
 - TopoART_R_prediction, 147
- LibTopoART_samples, 13
- LibTopoART_samples.Episodic_TopopART_sample1, 147
- LibTopoART_samples.Episodic_TopopART_sample2, 147
- LibTopoART_samples.TopoART_AM_sample1, 148
- LibTopoART_samples.TopoART_AM_sample2, 148
- LibTopoART_samples.TopoART_C_sample1, 149
- LibTopoART_samples.TopoART_C_sample2, 149
- LibTopoART_samples.TopoART_R_sample1, 149
- LibTopoART_samples.TopoART_R_sample2, 149
- LibTopoART_samples.TopoART_R_sample3, 150
- LibTopoART_samples.TopoART_sample1, 150
- LibTopoART_samples.TopoART_sample2, 150
- LibTopoART_samples.TopoART_sample3, 151
- ModuleNum
 - LibTopoART.ITopoART_base, 118
- NO_ADAPTATION
 - LibTopoART, 13
- Normal
 - LibTopoART, 13
- Phi
 - LibTopoART.ITopoART_base, 118
 - LibTopoART.Network_base, 126
- Phis
 - LibTopoART.ITopoART_base, 118
 - LibTopoART.Network_base, 126
- Predict
 - LibTopoART.Fast_TopopART_C, 51–54
 - LibTopoART.Fast_TopopART_R, 61–64
 - LibTopoART.Hypersphere_TopopART_C, 74, 75
 - LibTopoART.IAccess_TopopART_C< _AccessType >, 83, 84
 - LibTopoART.IAccess_TopopART_R< _AccessType >, 87, 88
- LibTopoART.TopoART_C, 138, 139
- LibTopoART.TopoART_R, 144–146
- RecallStep
 - LibTopoART.Fast_TopopART_AM, 37
 - LibTopoART.IAccessAssociativeRecall< _AccessType >, 90
- REMOVED_EDGE_CANDIDATE
 - LibTopoART, 13
- REMOVED_NODE_CANDIDATE
 - LibTopoART, 13
- ResetAdaptationState
 - LibTopoART.Fast_TopopART_base, 43
 - LibTopoART.TopoART, 132
- Save
 - LibTopoART.Fast_TopopART_base, 44
 - LibTopoART.ITopoART_base, 117
 - LibTopoART.TopoART, 132
- SaveText
 - LibTopoART.Fast_TopopART_base, 44
 - LibTopoART.ITopoART_base, 118
 - LibTopoART.TopoART, 132
- TopoART
 - LibTopoART.TopoART, 129
- TopoART_C
 - LibTopoART.TopoART_C, 136, 137
- TopoART_C_prediction
 - LibTopoART.TopoART_C_prediction, 139
- TopoART_R
 - LibTopoART.TopoART_R, 143, 144
- TopoART_R_prediction
 - LibTopoART.TopoART_R_prediction< _ElementType >, 147
- Verbose
 - LibTopoART, 13
- VerbosityLevel
 - LibTopoART, 13