

Dimensionality Reduction for Data Visualization and Classification

Suchismita Das

Electronics and Communication Sciences Unit
Indian Statistical Institute
203 B. T. Road
Kolkata - 700 108, India

Supervisor: Prof. Nikhil R. Pal

Electronics and Communication Sciences Unit
Indian Statistical Institute
203 B. T. Road
Kolkata - 700 108, India



A thesis submitted to the Indian Statistical Institute
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Computer Science

July 2023

To my family...

Acknowledgements

Firstly, I ardently believe that writing a mere acknowledgment over a piece of paper cannot convey how deeply obliged and thankful I am to those who have extended their enormous support, thorough guidance, tireless encouragement, and continued motivation throughout my Ph.D. journey. Over the past few years, the voyage has been quite exciting, and I would like to convey my profound gratitude to those who have helped me to grow each day and have invigorated me to be focused, especially my supervisor Prof. Nikhil Ranjan Pal, all my teachers, family members, friends, and colleagues. I owe my earnest regards to you all.

I am extremely grateful to my supervisor, Prof. Nikhil Ranjan Pal, for his unwavering guidance, relentless support, and keen supervision. His extensive knowledge of the subject matter, timely suggestions, and unfaltering encouragement have shaped my Ph.D. journey and have made it possible for me to complete my thesis. I would like to express my boundless gratitude to Prof. Pal for scrutinizing different aspects of the thesis along with his minute observations and corrections wherever needed which have immensely contributed towards the betterment of this thesis.

I am indebted to my friends and colleagues at ISI for their helpful discussions and friendly advice, especially Dr. Koustav Nag, Dr. Suvra Jyoti Choudhury, Sunanda Das, Koushik Sarkar, Rupan Panja, Dr. Bikash Santra, Dr. Sandip Paul. I gratefully acknowledge my lab mates Dr. Koustav Nag and Dr. Suvra Jyoti Choudhury in particular, for their never-ending support. I would also like to thank all the faculty members and office staff of the Electronics and Communication Sciences Unit (ECSU) and other Units of ISI for their support and cooperation.

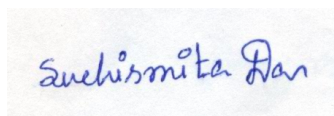
I am thankful to the members of the Research Fellow Advisory Committee, Computer and Communication Sciences Division (RFAC-CCSD) for providing constructive suggestions, invaluable advice in evaluation meetings, and their unceasing support.

Last, but not the least, I feel abundantly blessed to have a supportive family. Their absolute love, bottomless cooperation, and indomitable patience during the entire tenure have kept me steady and motivated in my Ph.D. journey. Special thanks to my father Prof. A. N. Das, mother Arundhuti Das, mother-in-law Anita Chakraborty and spouse Jit Chakraborty for their unshakable support and tremendous encouragement towards achieving my dreams.

I am really thankful to Dr. Himadri Mukherjee for the support he extended.

I would like to thank all my well-wishers from the bottom of my heart, including

those I might have missed mentioning and who have been part of this amazing journey.
Thank you !!



Suchismita Das

Abstract

In this thesis, we identify a few gaps in the existing methods of dimensionality reduction for data visualization and classification and propose some solutions to those as summarized below.

Most of the data visualization methods do not learn any explicit function to project high dimensional data to a lower dimension. To overcome the difficulty associated with the absence of an explicit map, in Chapter 2, we propose a framework to estimate explicit maps for data visualization in a supervised setting. The quality of output of any regression-type system depends on the quality of the target data. However, even for simple data, sometimes the target data for visualization may be severely distorted. We present a framework that can significantly correct such distortions in the output for data visualization.

For any supervised data visualization method the availability of target data is indispensable, which limits the applicability of such methods. Another problem with most of the methods is that they always produce some output given any input, even when the test input is far from the “sampling window” of the training data. In Chapter 3, using a fuzzy rule-based system (FRBS), we propose an unsupervised approach to learn explicit maps for data visualization that addresses the previously mentioned issues. The proposed method can project out-of-sample instances in a straightforward manner. It can also refuse to project an out-of-sample instance when it is far away from the sampling window of the training data. We have demonstrated the generality of the proposed framework using different objective functions for learning the FRBS.

When a data set has significant differences between its class and cluster structure, features selected considering only the discrimination between classes would lead to poor clustering performance. Similarly, features selected considering only the preservation of cluster structures would lead to poor classification performance. To address this issue, in Chapter 4, we propose a neural network-based feature selection method that focuses both on class discrimination and structure preservation. For large datasets, to reduce the computational overhead we propose an effective sample-based method.

When a data set has class-specific characteristics, selecting a single feature subset for the entire data set may not characterize the data correctly, although the classifier performance may be satisfactory. To address this, in Chapter 5, we have proposed class-specific feature selection (CSFS) schemes using feature modulators embedded in a fuzzy rule-based classifier. The parameters of the modulators are tuned by minimizing

a loss function comprising classification error and a regularizer to make the modulators completely select or reject features in a class-specific manner. Our method is free from the hazards of most of the existing CSFS methods, which suffer due to the use of one-vs-all strategy. We have extended the CSFS scheme so that it can monitor class-specific redundancy between selected features. We note here that data from a particular class may have multiple clusters and different clusters may be effectively defined by different subsets of features. To address this, finally, our CSFS framework is generalized to a rule-specific feature selection framework.

List of Publications

- **S. Das** and N. R. Pal. “Supervised learning of explicit maps with ability to correct distortions in the target output for manifold learning”, *Information Sciences*, vol. 614, pp. 311–324, 2022, (Digital Object Identifier: <https://doi.org/10.1016/j.ins.2022.06.069>).
- **S. Das** and N. R. Pal. “An unsupervised fuzzy rule-based method for structure preserving dimensionality reduction with prediction ability”, *Artificial Intelligence Applications and Innovations: 15th IFIP WG 12.5 International Conference, AIAI 2019, Springer*, pp. 413–424, 2019, (Digital Object Identifier: https://doi.org/10.1007/978-3-030-19823-7_35).
- **S. Das** and N. R. Pal. “Nonlinear Dimensionality Reduction for Data Visualization: An Unsupervised Fuzzy Rule-based Approach”, *IEEE Transactions on Fuzzy Systems*, vol. 30, no. 7, pp. 2157–2169, 2021, (Digital Object Identifier: 10.1109/TFUZZ.2021.3076583).
- **S. Das** and N. R. Pal. “Feature selection simultaneously preserving both class and cluster structures”, *arXiv preprint arXiv:2307.03902 (2023)*, (Digital Object Identifier: <https://doi.org/10.48550/arXiv.2307.03902>).
- **S. Das** and N. R. Pal. “Understanding the classes better with class-specific and rule-specific feature selection, and redundancy control in a fuzzy rule based framework”, *Intelligent Data Engineering and Automated Learning: 23rd International Conference, IDEAL 2022, Springer*, pp. 338–347, 2022, (Digital Object Identifier: https://doi.org/10.1007/978-3-031-21753-1_33).

Contents

Contents	vii
List of Figures	xi
List of Tables	xix
List of Algorithms	xxi
1 Introduction	1
1.1 Categorization of dimensionality reduction approaches	2
1.1.1 Feature Extraction	2
1.1.2 Feature Selection	3
1.2 Target tasks of dimensionality reduction methods	5
1.2.1 Unsupervised	5
1.2.2 Supervised	6
1.3 Pattern recognition concepts	8
1.3.1 Regression	8
1.3.2 Clustering	9
1.4 Computational intelligence tools	10
1.4.1 Fuzzy Rule-Based System	10
1.4.2 Multi-layer Perceptron	11
1.5 Scope and organization of the thesis	14
1.5.1 Chapter 2: Supervised learning of explicit maps with ability to correct distortions in the target output for manifold learning [1]	14
1.5.2 Chapter 3: Nonlinear dimensionality reduction for data visual- ization: An unsupervised fuzzy rule-based approach [2,3]	15
1.5.3 Chapter 4: Feature selection simultaneously preserving both class and cluster structures [4]	16
1.5.4 Chapter 5: Understanding the classes better with class-specific and rule-specific feature selection, and redundancy control in a fuzzy rule based framework [5]	16
1.5.5 Chapter 6: Conclusions, limitations, and future scopes	17

2	Explicit maps for manifold learning	19
2.1	Introduction	19
2.2	Proposed Method	22
2.3	Experiment	27
2.3.1	Data Set	27
2.3.2	Generation of Regression Target	27
2.3.3	Experimental Setting	29
2.4	Results and Discussion	30
2.4.1	Results with Targets Generated using t -SNE and ISOMAP . . .	35
2.5	Implementation with Neural Networks	37
2.6	Conclusion	43
3	Nonlinear dimensionality reduction for visualization	47
3.1	Introduction	47
3.2	Proposed Method	49
3.2.1	Initial Rule Extraction	51
3.2.2	The Objective Function and its Optimization	53
3.2.3	Scalability Analysis	53
3.3	Experimentation and Results	55
3.3.1	Data Set Description	55
3.3.2	Experiment Settings	56
3.3.3	Results and Comparisons	58
3.3.4	Impact of Initial Rules	63
3.3.5	Validation of Predictability for Out-of-sample Points	73
3.3.6	Declining to Project the Outputs	85
3.3.7	Generalized Nature of the Proposed Framework	85
3.4	Conclusion and Discussion	100
3.4.1	Limitations and scope of improvement	100
4	Feature selection preserving class & cluster structures	103
4.1	Introduction	103
4.2	Proposed Method	104
4.3	Experimentation and Results	108
4.3.1	Feature selection for conventional classification datasets	108
4.3.2	Band selection in hyperspectral images	116
4.4	Conclusion and Discussions	123
5	Class-specific and Rule-specific feature selection	125
5.1	Introduction	125
5.2	Proposed Method	128
5.2.1	Fuzzy Rule-based Classifiers	128
5.2.2	Feature Selection	129
5.2.3	Monitoring Redundancy	130
5.2.4	Exploiting Substructures Within a Class.	131

CONTENTS

ix

5.3

Experiments and Results

132

5.3.1

Experiment 1

132

5.3.2

Experiment 2

136

5.3.3

Experiment 3

138

5.4

Conclusion

139

6

Conclusions and future scopes

141

6.1

Conclusions

141

6.2

Limitations and future scopes

145

Bibliography

149

List of Figures

1.1	The components of MA model	11
1.2	Architecture of multi-layer perceptron network with two hidden layers	12
2.1	Data sets	27
2.2	LLE representation: Swiss Roll with $\epsilon =$ (a)15, (b)20, (c)25; C Curve with $\epsilon =$ (d)15, (e)20, (f)25; COIL with $\epsilon =$ (g)2, (h)3, (i)4.	28
2.3	Results of polynomial map-based regression on Swiss Roll data set. (a) Regression target (generated by LLE with $\epsilon = 25$). (b) Regression outputs without regularizer. Regularized regression outputs ($k = 10, \beta = 200$) : (c) Gaussian-based weights ($\sigma = 0.5$), (d) inverse quadratic-based weights ($\gamma = 10$), (e) inverse sub-quadratic-based weights ($\gamma = 100$).	31
2.4	Results of polynomial map-based regression on the Swiss Roll data set. (a) Regression target (generated by LLE with $\epsilon = 15$). (b) Regression outputs without regularizer. Regularized regression outputs ($k = 10$) : (c) Gaussian-based weights ($\beta = 500, \sigma = 0.5$), (d) Gaussian-based weights ($\beta = 1000, \sigma = 0.5$), (e) inverse quadratic based weights ($\beta = 1000, \gamma = 30$), and (f) inverse sub-quadratic based weights ($\beta = 500, \gamma = 100$).	32
2.5	Outputs of LE (Gaussian) method on the Swiss Roll data for different choices of k : (a) $k = 25$, (b) $k = 15$, and (c) $k = 10$	32
2.6	Results of polynomial map based regression on C Curve data set. (a) Regression target (generated by LLE with $\epsilon = 25$). (b) Regression outputs without regularizer. Regularized regression outputs ($k = 10$) : (c) Gaussian-based weights ($\beta = 100, \sigma = 1$), (d) inverse quadratic-based weights ($\beta = 100, \gamma = 30$), (e) inverse sub-quadratic-based weights ($\beta = 100, \gamma = 50$). (f) Output by LE (Gaussian) with $k = 10$	33
2.7	Results of polynomial map-based regression on COIL data set. (a) Regression target (generated by LLE with $\epsilon = 3$). (b) Regression outputs without regularizer. Regularized regression outputs ($k = 10$) : (c) Gaussian-based weights ($\beta = 100, \sigma = 1.45$), (d) Gaussian-based weights ($\beta = 200, \sigma = 1.45$), (e) inverse quadratic-based weights ($\beta = 200, \gamma = 13$), (f) inverse of sub-quadratic-based weights ($\beta = 200, \gamma = 13$).	34
2.8	LE (Gaussian based) Output on COIL dataset.	34

2.9	Gaussian Based Regularized ($\beta = 200$) Ridge Regression Outputs on COIL data set with $\theta =$ (a) 0, (b) 1, (c) 10, (d) 100.	35
2.10	Inverse quadratic based regularized ($\beta = 200$) ridge regression outputs on COIL data set with $\theta =$ (a) 0, (b) 1, (c) 10, (d) 100.	36
2.11	Inverse sub-quadratic based regularized ($\beta = 200$) ridge regression outputs on COIL data set with $\theta =$ (a) 0, (b) 1, (c) 10, (d) 100.	37
2.12	Results of polynomial map based regression on Swiss Roll data set with t -SNE as the target. (a) Regression target (generated by t SNE with perplexity=20). (b) Regression outputs without regularizer. Regularized regression outputs ($k = 5, \beta = 1000$) : (c) Gaussian based weights ($\sigma = 1$), (d) inverse quadratic based weights ($\gamma = 10$), (e) inverse sub-quadratic based weights ($\gamma = 10$).	38
2.13	Results of polynomial map based regression on Swiss Roll data set with ISOMAP as the target. (a) Regression target (generated by ISOMAP with $k = 25$). (b) Regression outputs without regularizer. Regularized regression outputs ($k = 5, \beta = 1000$) : (c) Gaussian based weights ($\sigma = 1$), (d) inverse quadratic based weights ($\gamma = 10$), (e) inverse sub-quadratic based weights ($\gamma = 10$).	39
2.14	Results of polynomial map based regression on COIL data set with t SNE as the target. (a) Regression target (generated by t SNE with perplexity=3). (b) Regression outputs without regularizer. Regularized regression outputs ($k = 5, \beta = 200$) : (c) Gaussian based weights ($\sigma = 1.45$), (d) inverse quadratic based weights ($\gamma = 13$), (e) inverse sub-quadratic based weights ($\gamma = 13$).	40
2.15	Results of polynomial map based regression on COIL data set with ISOMAP as the target. (a) Regression target (generated by ISOMAP with $k = 1$). (b) Regression outputs without regularizer. Regularized regression outputs ($k = 10, \beta = 200$) : (c) Gaussian based weights ($\sigma = 1.45$), (d) inverse quadratic based weights ($\gamma = 13$), (e) inverse sub-quadratic based weights ($\gamma = 13$).	41
2.16	Results of neural net-based regression on Swiss Roll data set. (a) Regression target (generated by LLE with $\epsilon = 25$). (b) Regression outputs without regularizer. Regularized regression outputs ($k = 10$) : (c) Gaussian-based weights ($\beta = 0.75, \sigma = 0.3$), (d) Gaussian ($\beta = 0.5, \sigma = 0.4$). . . .	42
2.17	Results of neural net-based regression on C Curve data set. (a) Regression target (generated by LLE with $\epsilon = 25$). (b) Regression outputs without regularizer. Regularized regression outputs ($k = 10$) : (c) Gaussian-based weights ($\beta = 1, \sigma = 0.2$), (d) Gaussian-based weights ($\beta = 0.25, \sigma = 0.3$).	42

2.18	Results of neural net-based regression on COIL data set. (a) Regression target (generated by LLE with $\epsilon = 3$). (b) Regression outputs without regularizer. Regularized regression outputs ($k = 5$): (c) Gaussian-based weights ($\beta = 0.5, \sigma = 2$), (d) inverse quadratic-based weights ($\beta = 0.5, \gamma = 0.01$), (e) inverse sub-quadratic-based weights ($\beta = 0.5, \gamma = 0.01$).	44
3.1	Data sets	55
3.2	Variation of average final error values with number of rules and initial spreads.	56
3.3	Visualization of the Swiss Roll data with (a) proposed method, (b) Sammon's projection, (c) ISOMAP, (d) LLE, (e) t -SNE, (f) UMAP, and (g) Diffusion net.	59
3.4	Visualization of the S Curve data with (a) original input space, (b) proposed method, (c) Sammon's projection, (d) ISOMAP, (e) LLE, (f) t -SNE, (f) UMAP, and (g) Diffusion net.	60
3.5	Visualization of the Helix data with (a) proposed method, (b) Sammon's projection, (c) ISOMAP, (d) LLE, (e) t -SNE, (f) UMAP, and (g) Diffusion net.	61
3.6	Visualization of the Frey face data set with the proposed method.	62
3.7	Visualization of the Frey-face data set with Sammon's Projection.	62
3.8	Visualization of the Frey-face data set with ISOMAP	63
3.9	Visualization of the Frey-face data set with LLE.	64
3.10	Visualization of the Frey-face data set with t -SNE.	64
3.11	Visualization of the Frey-face data set with UMAP.	65
3.12	Visualization of the Frey-face data set with diffusion net.	65
3.13	Visualization of the first object of the COIL Data Set with (a) proposed method, (b) Sammon's projection, (c) ISOMAP, (d) LLE.	66
3.14	Visualization of the first object of the COIL Data Set with (a) t -SNE, (b) UMAP, and (c) diffusion net.	67
3.15	Visualization of the USPS data set with the proposed method.	68
3.16	Visualization of the USPS data set with Sammon's Projection.	68
3.17	Visualization of the USPS data set with ISOMAP	69
3.18	Visualization of the USPS data set with LLE.	69
3.19	Visualization of the USPS data set with t -SNE.	70
3.20	Visualization of the USPS data set with UMAP.	70
3.21	Visualization of the USPS data set with diffusion net.	71
3.22	Visualization of the Swiss Roll data using proposed method with rule antecedents defined by (a) cluster centroids, (b) random values from the smallest hyperbox containing the training data.	72
3.23	Visualization of the Frey face data set using the proposed method with random rule antecedents.	72

3.24	Visualization of the Swiss Roll data set by proposed method with rule antecedents defined by random values within zero and one: (a) best run (b) worst run.	73
3.25	For experiment 1 on validation of predictability with the Swiss Roll data : (a) Training Set, (b) Test Set	74
3.26	For experiment 1 on validation of predictability with the Swiss Roll data proposed method output for (a) Training Set, (b) Test Set.	74
3.27	For experiment 1 on validation of predictability with the Swiss Roll data : (a) ISOMAP output for Training Set, and (b) Out of sample extension of ISOMAP for Test Set.	75
3.28	For experiment 1 on validation of predictability with the Swiss Roll data : (a) LLE output for Training Set, and (b) Out of sample extension of LLE for Test Set.	76
3.29	For experiment 1 on validation of predictability with the Swiss Roll data : (a) Autoencoder encoded output for Training Set, and (b) Autoencoder encoded output for Test Set.	76
3.30	For experiment 1 on validation of predictability with the Swiss Roll data : (a) 2nd Order SNPPE output for Training Set, (b) 2nd Order SNPPE output for Test Set.	77
3.31	For experiment 1 on validation of predictability with the Swiss Roll data : (a) 3rd Order SNPPE output for Training Set, (b) 3rd Order SNPPE output for Test Set.	77
3.32	Visualization of the training and test set (black rectangle borders) for Frey face data set using the proposed method	78
3.33	Visualization of the training set using ISOMAP and test set (black rectangle borders) with its Out of sample extension for Frey face data set	78
3.34	Visualization of the training set using LLE and test set (black rectangle borders) with its Out of sample extension for Frey face data set	79
3.35	Visualization of the training set using Autoencoder encoded output and test set (black rectangle borders) for Frey face data set	79
3.36	Visualization of the training set using 2nd Order SNPPE output and test set (black rectangle borders) for Frey face data set	80
3.37	Visualization of the training set using 3rd Order SNPPE output and test set (black rectangle borders) for Frey face data set	81
3.38	For experiment 2 on validation of predictability with the Swiss Roll data, (a) Training Set, (b) Training Set with Test Set.	81
3.39	For experiment 2 on validation of predictability with the Swiss Roll data, proposed method output for (a) Training Set, and (b) Training Set with Test Set.	82
3.40	For experiment 2 on validation of predictability with the Swiss Roll data, (a) ISOMAP output for Training Set, and (b) Out of sample extension of ISOMAP for Training Set and Test Set.	82

3.41	For experiment 2 on validation of predictability with the Swiss Roll data, (a) LLE output for Training Set, and (b) Out of sample extension of LLE for Training Set and Test Set.	83
3.42	For experiment 2 on validation of predictability with the Swiss Roll data, autoencoder encoded output for (a) Training Set, and (b) Training Set and Test Set.	83
3.43	For experiment 2 on validation of predictability with the Swiss Roll data, 2nd Order SNPPE output for (a) Training Set, and (b) Training Set and Test Set.	84
3.44	For experiment 2 on validation of predictability with the Swiss Roll data, 3rd Order SNPPE output for (a) Training Set, and (b) Training Set and Test Set.	84
3.45	For experiment 3 on validation of predictability with leave one out sets composed of the first object of the COIL data set, proposed method outputs corresponding to (a) Set 1, (b) Set 2, (c) Set 3, (d) Set 4, (e) Set 5, and (f) Set 6.	86
3.46	For experiment 3 on validation of predictability with leave one out sets composed of the first object of the COIL data set, proposed method outputs corresponding to (a) Set 7, (b) Set 8, (c) Set 9, (d) Set 10, (e) Set 11, and (f) Set 12.	87
3.47	For experiment 3 on validation of predictability with leave one out sets composed of the first object of the COIL data set, proposed method outputs corresponding to (a) Set 13, (b) Set 14, (c) Set 15, (d) Set 16, (e) Set 17, and (f) Set 18.	88
3.48	For experiment 3 on validation of predictability with leave one out sets composed of the first object of the COIL data set, proposed method outputs corresponding to (a) Set 19, (b) Set 20, (c) Set 21, (d) Set 22, (e) Set 23, and (f) Set 24.	89
3.49	For experiment 3 on validation of predictability with leave one out sets composed of the first object of the COIL data set, proposed method outputs corresponding to (a) Set 25, (b) Set 26, (c) Set 27, (d) Set 28, (e) Set 29, and (f) Set 30.	90
3.50	For experiment 3 on validation of predictability with leave one out sets composed of the first object of the COIL data set, proposed method outputs corresponding to (a) Set 31, (b) Set 32, (c) Set 33, (d) Set 34, (e) Set 35, and (f) Set 36.	91
3.51	For experiment 3 on validation of predictability with leave one out sets composed of the first object of the COIL data set, proposed method outputs corresponding to (a) Set 37, (b) Set 38, (c) Set 39, (d) Set 40, (e) Set 41, and (f) Set 42.	92

3.52	For experiment 3 on validation of predictability with leave one out sets composed of the first object of the COIL data set, proposed method outputs corresponding to (a) Set 43, (b) Set 44, (c) Set 45, (d) Set 46, (e) Set 47, and (f) Set 48.	93
3.53	For experiment 3 on validation of predictability with leave one out sets composed of the first object of the COIL data set, proposed method outputs corresponding to (a) Set 49, (b) Set 50, (c) Set 51, (d) Set 52, (e) Set 53, and (f) Set 54.	94
3.54	For experiment 3 on validation of predictability with leave one out sets composed of the first object of the COIL data set, proposed method outputs corresponding to (a) Set 55, (b) Set 56, (c) Set 57, (d) Set 58, (e) Set 59, and (f) Set 60.	95
3.55	For experiment 3 on validation of predictability with leave one out sets composed of the first object of the COIL data set, proposed method outputs corresponding to (a) Set 61, (b) Set 62, (c) Set 63, (d) Set 64, (e) Set 65, and (f) Set 66.	96
3.56	For experiment 3 on validation of predictability with leave one out sets composed of the first object of the COIL data set, proposed method outputs corresponding to (a) Set 67, (b) Set 68, (c) Set 69, (d) Set 70, (e) Set 71, and (f) Set 72.	97
3.57	(a) Training Set (blue circles) and Test Set (red circles), (b) Histogram of maximum rule firings for Training Set (blue region) and Test Set (red region), and (c) Proposed method Output for Training Set (blue circles) and Test Set (red circles) without exercising rejection option.	98
3.58	Visualization of Swiss Roll data with (a) original input space, and (b) proposed method with Sammon's Objective.	98
3.59	Visualization of S Curve data with (a) original input space, and (b) proposed method with Sammon's objective.	99
3.60	Visualization of Helix data with (a) original input space, and (b) proposed method with Sammon's objective.	99
4.1	Proposed network for feature selection	105
4.2	Variations of OCA with the number of selected bands for (a) Indian pines training set, (b) Indian pines test set, (c) Pavia university training set, (d) Pavia university test set, (e) Salinas training set, (f) Salinas test set.	119
4.3	Thematic maps of the Indian pines dataset (a) with ground truth labels, (b) with labels from classification (SVM) considering 70 bands selected by BSMLP, (c) with labels from classification (SVM) considering 140 bands selected by BSMLP.	120
4.4	Thematic maps resulting from classifications (SVM) of the Indian pines dataset with 70 bands selected by BSMLP _{struct} considering (a) $\beta = 0$, (b) $\beta = 20$, (c) $\beta = 50$	124

5.1	Variations of the class-specific feature modulation parameters $\lambda_{j,k}$ s for class 1 over iterations.	133
5.2	Variations of the class-specific feature modulation parameters $\lambda_{j,k}$ s for class 2 over iterations.	134
5.3	Variations of the class-specific feature modulation parameters $\lambda_{j,k}$ s for class 3 over iterations.	134
5.4	Variations of the global feature modulation parameters λ_j s over iterations.	135

List of Tables

3.1	Computational complexity of various methods.	55
3.2	Average of the final error values for different initializations	73
4.1	Summary of datasets.	109
4.2	Performance comparison for E. coli, training set for different choices of β and Q	111
4.3	Performance comparison for E. coli, test set for different choices of β and Q	111
4.4	Performance comparison for Glass, training set for different choices of β and Q	112
4.5	Performance comparison for Glass, test set for different choices of β and Q	112
4.6	Performance comparison for Ionosphere, training set for different choices of β and Q	113
4.7	Performance comparison for Ionosphere, test set for different choices of β and Q	114
4.8	Performance comparison for Sonar, training set for different choices of β and Q	114
4.9	Performance comparison for Sonar, test set for different choices of β and Q	115
4.10	Performance comparison for AR10P, training set for different choices of β and Q	115
4.11	Performance comparison for AR10P, test set for different choices of β and Q	115
4.12	Performance comparison for Indian Pines training set, with the number of bands= 70.	120
4.13	Performance comparison for Indian Pines test set, with the number of bands= 70.	121
4.14	Performance comparison for Pavia University training set, with the number of bands= 35.	121
4.15	Performance comparison for Pavia University test set, with the number of bands= 35.	121

4.16	Performance comparison for Salinas training set, with the number of bands= 70.	122
4.17	Performance comparison for Salinas test set, with the number of bands= 70.	122
5.1	Description of the dataset Synthetic1	132
5.2	Performance on the dataset, Synthetic1	135
5.3	Feature correlation matrix with the whole Synthetic2 data	136
5.4	Feature correlation matrix with the class 1 of Synthetic2 data	136
5.5	Feature correlation matrix with the class 2 of Synthetic2 data	137
5.6	Feature correlation matrix with the class 3 of Synthetic2 data	137
5.7	Class-specific feature selection on Synthetic2 data set	137
5.8	Description of the dataset Synthetic3	138
5.9	Features subsets selected for Synthetic3	139

List of Algorithms

1	Geodesic c -means algorithm (GCM)	52
2	Unsupervised Fuzzy Rule-based Dimensionality Reduction	54

Chapter 1

Introduction

Dimensionality reduction methods transform high-dimensional data to a lower-dimension, preserving the “relevant” information as much as possible. It is often an important and necessary step when we need to deal with high dimensional real-world data such as digital photographs, video sequences, functional magnetic resonance imaging (fMRI) scans, mass spectrometric data, chemical data, biomedical data, text data, meteorological data, and financial data [6–11]. Even for data having moderate dimensions, dimensionality reduction is often required for data visualization to get a better understanding of the characteristics of the data. Dimensionality reduction methods can improve the performance of the decision-making systems, enhance their generalization ability, and mitigate the curse of dimensionality, and other undesired properties of high-dimensional data like the need for extra storage space and high processing time [12]. The key idea of dimensionality reduction is, given a P -dimensional data set, we try to find a Q ($Q < P$) dimensional representation of the data so that the target problem can be solved satisfactorily using the Q -dimensional representation. Sometimes, the target problem is known; such as regression, classification, clustering, or visualization, while in some other cases, no target problem may be specified. In the latter case, dimensionality reduction can be done by preserving some properties of the high-dimensional data in its lower-dimensional representation. Over the last few decades, with an increase in the number of fields adopting machine learning (ML) based models, the importance of dimensionality reduction methods has also increased. The performances of ML models are greatly influenced by the data representations they use. Dimensionality reduction, being a means of providing data representations effective for learning the models, is in great demand [13]. One may wonder that in this era of AI-ML (Artificial Intelligence and Machine learning) when large-scale models with billions of parameters trained on massive data sets are not uncommon [14–17], is dimensionality reduction still relevant? Yes, it is still relevant because many problems are not big data problems, for many critical problems, such as healthcare, visualization is very important. For such applications, users demand an interpretable system with explainability. Dimensionality reduction can help to achieve these goals. The research community is showing interest in exploring dimensionality reduction in connection with emerging ML tools

such as deep neural networks [18], convolutional neural networks [19], deep belief networks [20], transformers [13, 21], deep generative models [22], and explainable artificial intelligence [23]. Under this scenario, research for addressing the issues which are not solved or solved with some limitations by the state-of-art dimensionality reduction methods carries great relevance. In this thesis, we consider the dimensionality reduction methods primarily designed for visualization and classification. We have identified some gaps and drawbacks in some of the existing body of work in dimensionality reduction. We propose some solutions for those gaps and drawbacks in this thesis. In this context, we primarily use two computational intelligence tools, neural networks and fuzzy systems [24–33]. In subsequent sections, we discuss some concepts related to dimensionality reduction that are relevant to this thesis.

1.1 Categorization of dimensionality reduction approaches

Based on the reduction approach, dimensionality reduction schemes can be divided into two broad categories: feature extraction and feature selection. We now formally define dimensionality reduction. Let us denote the input space by $\mathcal{X} \subseteq \mathbb{R}^P$, where, P is the original dimension of the data. Let the feature set corresponding to input space be $\{x_1, x_2, \dots, x_P\}$. Let us denote the output space, i.e, the space where the data are projected in lower dimension by $\mathcal{Y} \subseteq \mathbb{R}^Q$, where, Q is the output dimension or reduced dimension of the data and $Q < P$. Let the feature set corresponding to the output space be $\{y_1, y_2, \dots, y_Q\}$. Dimensionality reduction is a mapping $\phi : \mathcal{X} \mapsto \mathcal{Y}$. For a point $\mathbf{x} \in \mathcal{X}$, its projection in lower dimension, $\mathbf{y} \in \mathcal{Y}$ is given by $\mathbf{y} = \phi(\mathbf{x})$.

1.1.1 Feature Extraction

Feature extraction *generally* constructs a smaller number of new features from the original features. Feature extraction is more appropriate when the user does not care about the physical meaning of the extracted features and focuses principally on preserving some information e.g., variance and/or structure, present in the original data. We use the word ‘generally’ because in a broader sense feature extraction can generate features in a dimension higher than that of the input [34, 35]. For example, for an image of size $M \times N$, the pixel values provide a feature vector in $\mathbb{R}^{MN}$. Now, for each pixel, using the neighborhood information we can generate another $M \times N$ features making the image represented by a vector in $\mathbb{R}^{2MN}$. However, in this thesis, feature extraction always refers to dimensionality reduction. Hence, we shall assume $Q < P$. In feature extraction, new features y_k s, where, $k \in \{1, 2, \dots, Q\}$, are constructed by either *linear* or *non-linear* combinations of the original features x_j s, where, $j \in \{1, 2, \dots, P\}$. When y_k s are generated by *linear combinations* of x_j s, the mapping ϕ is *linear*. Similarly, when y_k s are generated by *non-linear combinations* of x_j s, the mapping ϕ is *non-linear*. Examples of linear methods include principal component analysis (PCA) [36], canonical correlation analysis [37], linear discriminant analysis (LDA) [38], factor analysis [39], locality preserving projections [40]. Note that these linear methods are not suitable

for capturing information if the data set has non-linear structures. Some of the non-linear feature extraction methods are Sammon’s projection [41], ISOMAP [42], Locally linear embedding (LLE) [43], Laplacian Eigenmaps (LE) [44], autoencoder based methods [24], t-distributed stochastic neighbor embedding (*t*-SNE) [45], and Uniform Manifold Approximation and Projection (UMAP) [46, 47]. Linear methods are extensively used due to their simple geometric interpretations and computational properties [48]. However, non-linear methods find applications in real-world datasets that often have non-linear structures. Apart from linear and non-linear, feature extraction methods can also be categorized based on the parametric nature of the mapping function ϕ . Some of the feature extraction methods are non-parametric, i.e., they map the original input data into the lower dimensional output space implicitly. Given an input dataset $\mathbf{X}$, the non-parametric methods learn the corresponding lower dimensional output $\mathbf{Y}$, without learning the mapping ϕ explicitly. Sammon’s projection, ISOMAP, LLE are examples of non-parametric feature extraction methods. On the other hand, there are some feature extraction methods that learn ϕ and compute the lower dimensional representations $\mathbf{Y}$ as $\mathbf{y} = \phi(\mathbf{x})$, $\forall \mathbf{x} \in \mathbf{X}$, $\mathbf{y} \in \mathbf{Y}$. These feature extraction methods are called parametric methods or explicit mapping methods. PCA, LDA, and the autoencoder based method are examples of parametric feature extraction methods. Needless to say, parametric methods can directly map unseen or new points to the lower dimensional space through the available mapping ϕ whereas, non-parametric methods have to recompute the projections with the augmented data, i.e., previously available points along with the new points. Naturally, non-parametric methods are computationally intensive compared to parametric methods in handling out-of-sample points. There are also other advantages of parametric methods that we will discuss in Chapter 2 in a relevant context. Whether ϕ be an explicit or implicit mapping function, to learn it an objective function ψ is optimized. Usually, ψ is a function of the input dataset $\mathbf{X}$, and output dataset $\mathbf{Y}$, if $\mathbf{Y}$ is available. If the objective function ψ does not contain any local optima, the solution space is said to be convex, and hence the corresponding feature extraction method is called a convex feature extraction method. Similarly, when the objective function ψ contains local optima, the corresponding feature extraction method is non-convex. PCA, LDA, ISOMAP, and LLE are examples of convex methods whereas, Sammon’s projection, autoencoder based method, *t*-SNE, UMAP are examples of non-convex methods. Non-convex methods suffer from the presence of local optima in the search space causing difficulty in finding the global optima. However, some of the studies [10, 45] have shown that non-convex methods such as autoencoder based method [49], *t*-SNE [45] perform better than convex methods. In some cases, optimization of convex objectives is infeasible for large real datasets [45]. In Chapters 2 and 3 of our thesis, we deal with feature extraction based dimensionality reduction methods.

1.1.2 Feature Selection

Feature selection chooses a subset of the original features to represent the data faithfully for some intended task. The selected feature set, $\{y_1, y_2, \dots, y_Q\} \subseteq \{x_1, x_2, \dots, x_P\}$, is

4 1.1. CATEGORIZATION OF DIMENSIONALITY REDUCTION APPROACHES

a subset of the original feature set. It is obvious that for feature selection, the mapping ϕ is linear. We can represent the selected features as the following.

$$\begin{aligned} y_1 &= \theta_{1,1}x_1 + \theta_{1,2}x_2 + \cdots + \theta_{1,P}x_P \\ y_2 &= \theta_{2,1}x_1 + \theta_{2,2}x_2 + \cdots + \theta_{2,P}x_P \\ &\vdots \\ y_Q &= \theta_{Q,1}x_1 + \theta_{Q,2}x_2 + \cdots + \theta_{Q,P}x_P \end{aligned} \tag{1.1.1}$$

given, $\theta_{k,j} \in \{0, 1\}$, $\sum_j \theta_{k,j} = 1$, $\sum_k \theta_{k,j} = 1$, $\forall k, j$.

In feature selection, the aim is to learn $\{y_1, y_2, \dots, y_Q\}$. Note that, learning $\{y_1, y_2, \dots, y_Q\}$ implies learning the coefficients $\theta_{k,j}$ s. So by default, the feature selection methods are explicit mapping methods. When a data set is projected by feature selection, the reduced space maintains the identity of the original features. If the original features carry some meaning, the selected features are also meaningful. This adds to the interpretability of the underlying system that uses the selected features. Feature selection methods require some criterion to measure the relevance of each feature. Once a feature selection criterion is decided, a procedure must be followed to find a subset of the most relevant features. The procedures can be broadly classified into three groups: filter, wrapper, and embedded methods. Filter methods rank the features and select some highly ranked ones. These methods do not take any feedback from the predictor system where its output is used. Having chosen a subset without any guidance from the underlying predictor system, the filter methods may not produce optimum subsets with respect to the predictor system [12, 50]. Fisher score or F score [51] and mutual information [52, 53], normalized mutual information [53–56] are examples of filtering criterion for filter based feature selection methods. In wrapper methods, the feature selection criterion is the performance of the predictor system. The subset of features that yields the highest score from the predictor is selected. To find the optimum feature subset, actually, all possible feature subsets need to be offered to the predictor system. For P input features, to find the best subset, $(2^P - 1)$ subsets need to be evaluated. This makes the wrapper method infeasible when P is even moderately high [12, 50]. To overcome this difficulty different searching strategies such as best first search, branch and bound, sequential search, hill climbing, simulated annealing, and genetic algorithms have been used [12, 57]. On the other hand, embedded methods perform feature selection in an integrated manner with the primary learning task i.e., they learn the predictor system simultaneously while choosing the best subset for the predictor system. Usually, embedded methods are more efficient than the other two families of methods [12, 50, 57]. Some popular approaches of the embedded category are: pruning based, decision tree based, and regularization based. Recursive feature elimination with support vector machines (SVM) is an example of a pruning based method; CART, C4.5, ID3 are examples of decision tree based methods; and least absolute shrinkage and selection operator (LASSO), Elastic net are the examples of regularization based embedded methods [12, 58]. There are several embedded methods based on neural networks, fuzzy systems, and evolutionary computing [25–27, 30, 31, 33, 50, 59–61]. Note

that, filter and wrapper methods choose subsets according to some score based on the feature selection criterion, however, embedded methods optimize an objective function to learn the predictor system as well as the required set of features. The objective functions of the embedded feature selection methods can also be convex or non-convex, although primarily non-convex. LASSO is an example convex method [12]. In Chapters 4 and 5 of our thesis, we deal with feature selection for classification based on neural networks and fuzzy systems using the embedded framework.

1.2 Target tasks of dimensionality reduction methods

As discussed at the beginning of this chapter, when we reduce the dimensionality of the data, there may or may not be an associated target problem. When there is no specified target problem, the dimensionality reduction is done to get rid of redundant or noisy information mainly to reduce the computational burden and the storage requirement. However, in many cases, dimensionality reduction is a pre-processing step before a pattern recognition task. In those cases, the aim of the dimensionality reduction is to help the underlying pattern recognition task. The pattern recognition task at hand may be a supervised or an unsupervised one. In the next subsections, we define unsupervised and supervised pattern recognition and briefly elaborate on the two target problems that we mainly deal with, in this thesis. We also present overviews of the approaches in which the dimensionality reduction methods produce data representations to facilitate the target applications.

1.2.1 Unsupervised

For an unsupervised pattern recognition task, the input data are not labeled or we can say that no target outputs are provided. To recognize the patterns or structures in the input data, no supervision is provided, and hence the corresponding task is called an unsupervised task. Data visualization, compression, denoising, clustering, association, mining, and information retrieval are some prominent unsupervised target tasks of dimensionality reduction methods [62–65]. Among these unsupervised tasks, in this thesis, we have explored and proposed dimensionality reduction methods for data visualization. So, We briefly discuss it in the following.

Data visualization

Data visualization require representation of the information present in the data effectively through graphical means. The agenda for data visualization is to recognize patterns/structures in the data through visual inspection. Clearly, we can not plot more than three dimensions graphically. There are two broad approaches to data visualization. In the first approach, the data is kept intact and some mechanisms are utilized to display more than two dimensions graphically. Examples are parallel coordinates, tree maps, and Chernoff face [66, 67]. The other approach is the dimensionality reduction approach where high dimensional data are reduced to three or two dimensions with a

view to keep the necessary information intact to the extent possible. Several dimensionality reduction methods [11, 41, 45, 68–73] have been adapted as data visualization tools. These dimensionality reduction methods are applied to reduce high dimensional input data into two or three-dimensional data which are then graphically represented in a scatter plot. In data visualization, we would like to discover the underlying structures and uncover the patterns present in the original data. To serve this purpose, the dimensionality reduction methods should preserve some properties of the original data in its lower dimensional representation. It is worth mentioning that linear methods are less used as data visualization tools because most of the real datasets are expected to have complex nonlinear structures. A linear method like PCA preserves the variance of the data in the lower dimensional representation through linear projections. However, it fails to project the nonlinear structures in the original data faithfully in the lower dimensions.

The nonlinear dimensionality reduction methods for data visualization can be broadly categorized on the property of the data they preserve in the lower dimensional projections. A group of dimensionality reduction methods aims at preserving inter-point distances. Multi-Dimensional Scaling (MDS) [74], Sammons projection [41], curvilinear component analysis [75], and curvilinear distance analysis [76] are examples of distance preserving methods. Spectral dimensionality reduction methods such as LLE [43], ISOMAP [42], LE [44], and maximum variance unfolding (MVU) [77] preserve some properties on the spectrum of the neighborhood graph of the original data. These methods learn the underlying manifold of the data through the neighborhood graph [70]. Stochastic neighborhood embedding (SNE) [78], t -SNE [45] are methods whose objective is to preserve the neighborhood distribution of the original data in the lower dimensional space. Another group of methods: self-organizing map, and generative topographic map [79] focus on preserving the ‘geometric structure’ or topology of the original data in the projected map. There are several other dimensionality reduction methods that use different objectives for preserving relevant information to facilitate visualization of the input data [28, 47, 49, 80–83].

1.2.2 Supervised

For some pattern recognition tasks, the input data are labeled, i.e., the corresponding target outputs are provided. The objective of supervised learning is to estimate the parameters of a decision-making system using the labeled training data. This kind of task is said to be a supervised learning task. All the supervised pattern recognition tasks can be broadly divided into two: classification and regression or function approximation. In our thesis, we have explored and proposed dimensionality reduction methods for the classification tasks, hence, we discuss the classification problem next.

Classification

For a classification problem, the input consists of a set of instances where each instance is associated with a label called class. The task is to train a decision making

system that can assign a class to an unseen instance whose label is not known [84]. Mathematically, we could define the classification problem as follows. Let the set of input instances be given by, an $n \times P$ matrix $\mathbf{X} = \{\mathbf{x}_i \in \mathbb{R}^P\}_{i=1}^n$. Here, $\mathbf{x}_i$ is a P dimensional row vector of the form, $\mathbf{x}_i = (x_{i1}, x_{i2}, \dots, x_{iP})$. The collection of class labels of $\mathbf{X}$ be $\mathbf{Z} = \{z_i \in \{1, 2, \dots, K\}\}_{i=1}^n$, where, z_i is the class label corresponding to $\mathbf{x}_i$. Let the input space associated with the input data $\mathbf{X}$ be $\mathcal{X}$. In classification, a function $\Upsilon : \mathcal{X} \mapsto \{1, 2, \dots, K\}$ is learnt to predict the class label of an instance $\hat{\mathbf{x}}$ with unknown class label as $\Upsilon(\hat{\mathbf{x}})$. Many of the problems, addressed by machine learning or pattern recognition can be posed as classification problems. Speech recognition [85], handwriting recognition [86], bio-metric identification [87], document classification [88], image classification [89], sentiment analysis [90] are examples of real world applications of classification. Dimensionality reduction, when applied as a preprocessing step in a classification task, is expected to improve the classification performance in addition to providing its usual benefits. Note that, for visualization, generally feature extraction based dimensionality reduction methods are adapted, but in classification, both feature extraction and feature selection based methods are widely used to reduce the data dimensionality. Non-parametric classifiers such as k nearest neighbors and probabilistic neural networks perform poorly under low sampling density. These classifiers benefit directly from the dimensionality reduction of the data as it increases the sampling density [91]. For some classifiers, the objective is to generate a latent space via a dimensionality reduction method that preserves the class discriminatory information maximally. Linear discriminant analysis (LDA) is a popular feature extraction based linear dimensionality reduction technique. LDA aims to find a projection that best discriminates the classes by maximizing between-class scatter and minimizing within-class scatter in the projection [92]. Several extensions of LDA were proposed to overcome the drawbacks of LDA. Regularized LDA [93], LDA with generalized singular value decomposition [94], null space LDA, orthogonal LDA [95] are examples of such LDA extensions. Note that, all of these are dimensionality reduction methods (feature extraction) that exploit class-label information to ensure class separability. The widely used non-negative matrix factorization (NMF) has been adapted to supervised versions by many. There are two broad approaches. The first approach [96–98] defines the objective function, taking into account a loss function involving class label as used in classification along with the NMF loss. The second approach [99–101] specifically involves the LDA objective i.e, minimizing within-class scatter and maximizing between-class scatter with the NMF loss. Supervised versions of common manifold learning algorithms [102–107] have been proposed to exploit the class labels to improve classification performance. Feature selection based dimensionality reduction methods are being used extensively for classification tasks. In feature selection, the identity of the features remains as it is, making the associated classifier more comprehensible. We have already mentioned that in a wrapper based feature selection, the feature selection criterion is the performance of the underlying predictor system. For a classification task, the predictor system is a classifier. Wrapper based methods use a classification score to decide the optimum subset. Clearly, wrapper based methods target at finding

the best performance of a classifier by choosing the optimum feature subset for it. The embedded feature selection methods integrate the process of learning the classifier with the process of learning the optimum feature subset for it. So, both the wrapper and embedded feature selection methods reduce data dimensionality with the objective of improving the associated classification performance. Filter based methods set a filtering criterion for selecting or eliminating features and do not take any feedback from the predictor system that will use the selected features. However, for a classification task, the filtering criterion may involve the class labels to improve the classification performance. Fisher score or F score is one such filtering criterion. It assigns a high score to a feature that shows similar values for the instances from the same class and different values for the instances from different classes.

1.3 A few other relevant pattern recognition concepts

Let us briefly discuss some topics from pattern recognition so that the subsequent contents of this thesis are easy to follow.

1.3.1 Regression

Regression analysis is a statistical technique for modeling relationships between some variables. The aim of regression is to approximate a function that maps the independent variables, called the predictors or explanatory variables to dependent variables, called the response or target variables [108,109]. Mathematically we can define regression as,

$$y_i = g(\mathbf{x}_i, \mathbf{b}) + e_i; i = 1, 2, \dots, n.$$

Here, $\{\mathbf{x}_i, y_i; i = 1, 2, \dots, n\}$, the collection has n observations. $\mathbf{x}_i$ represents the values of predictors or explanatory variables for the i^{th} observation. y_i is the response or target variable for the i^{th} observation, e_i is the error in estimating the response y_i . $g()$ is the regression model. If $g()$ is linear, it is defined using parameters $\mathbf{b} = (b_0, b_1, b_2, \dots, b_P)$. If $\mathbf{x}_i$ is a single variable, the corresponding regression is called simple regression and if $\mathbf{x}_i$ is a set of multiple variables, the corresponding regression is called multiple regression [110]. Similarly, if the response is composed of a single variable y_i , the regression is called single variate regression and when the response has multiple components $(y_{i1}, y_{i2}, \dots, y_{iQ})$, the regression is called multi-variate regression [111]. The most common practice is to take $g()$ as a linear equation. This type of regression is called linear regression. Assuming P predictors or explanatory variables, the linear regression model is expressed as follows.

$$\hat{y}_i = b_0 + b_1 x_{i1} + b_2 x_{i2} + \dots + b_P x_{iP} = \mathbf{b} \mathbf{x}_i^T$$

$$e_i = y_i - \hat{y}_i$$

Here, $\mathbf{x}_i = (x_{i1}, x_{i2}, \dots, x_{iP})$ is a P dimensional row vector representing explanatory variables for the i^{th} observation. When the model $g()$ is non-linear, the corresponding regression is called nonlinear regression. $g()$ could be a polynomial function, an

exponential function, and so on. The work in Chapter 2 of our thesis involves the polynomial regression as well as neural networks for learning an explicit map for dimensionality reduction. We have discussed the formulation of polynomial regression in detail in Chapter 2.

Feature selection can be done by keeping the structures (clusters) present in the data in view. In Chapter 4, we have proposed a method that tries to select features maintaining both class and cluster structures. Hence for the sake of completeness, next, we briefly introduce clustering.

1.3.2 Clustering

Clustering is an indispensable tool for artificial intelligence and its allied fields such as image processing, pattern recognition, and data mining. Clustering is used to solve problems of biology, banking, finance, marketing, weather and atmospheric science, web search, recommendation, healthcare, urban development, privacy protection, and so on [112,113]. clustering is an unsupervised task. Clustering refers to the partitioning of an unlabeled dataset $\mathbf{X}$ into some sets, $\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_{n_c}$, such that members from the same set are similar and members from the different sets are dissimilar; $\mathbf{X} = \bigcup_{i=1}^{n_c} \mathbf{X}_i$; $\mathbf{X}_i \cap \mathbf{X}_j = \emptyset$ [112]. Such clustering is known as crisp/hard clustering. There are other families of clustering algorithms where a data point may belong to more than one cluster. Such clustering where the sets may overlap, is known as soft clustering [114,115].

Based on the approaches, clustering algorithms can be assigned to the following categories [113,116].

- Partitional Clustering
- Hierarchical Clustering
- Density-Based Clustering
- Grid-Based Clustering
- Model-Based Clustering

The c -means (more commonly known as K -means) clustering [36], Gaussian mixture model (GMM) [116] and fuzzy c -means clustering (FCM) [117] are the most widely used clustering algorithms. The c -means and FCM are partitional clustering algorithms, whereas, GMM is a model-based clustering algorithm [116]. The c -means clustering is a hard clustering technique whereas GMM and FCM are soft clustering techniques [114] as they produce a soft partition of the data.

In this thesis, our proposed methods primarily use two computational intelligence tools, neural networks and fuzzy systems. This motivates us to briefly discuss the computational intelligence tools.

1.4 An introduction to some relevant computational intelligence tools

Computational intelligence (CI) tools or algorithms may be defined as a set of “nature-inspired” algorithms used for modeling complex problems and generally, they can deal with uncertain, imprecise, and incomplete knowledge [118,119]. The three most prominent pillars of CI are evolutionary computation, fuzzy logic, and artificial neural networks. The methods based on all three fields are extensively used for reducing data dimension by feature selection or extraction. There are several dimensionality reduction methods, based on both feature extraction and feature selection approaches that employ evolutionary computation [61,120–124], fuzzy logic [29–33], and artificial neural networks [24–28].

Now we will discuss two widely used computational intelligence tools- fuzzy rule based system (FRBS) and multi-layer perceptron (MLP) which is a type of artificial neural network. The following discussion will provide some foundation for the subsequent chapters as we have proposed models based on FRBS and MLP in our thesis.

1.4.1 Fuzzy Rule-Based System

FRBSs are widely used for building interpretable systems for complex problems in various fields such as control engineering, robotics, and bioinformatics. FRBSs can address uncertainty, imprecision, and non-linearity, the common aspects of a real-world problem. The FRBS is an extension of the classical rule-based system, composed of “If F_A Then F_C ” type rules, where the antecedent F_A and the consequent F_C both are fuzzy sets [125,126]. Let us define here, what fuzzy sets are. A classical set is based on bivalent logic. Any instance would be a member or not be a member of a classical set. In other words, for a classical set, memberships of its elements can be either 1 or 0. Fuzzy sets introduce degrees or grades of membership. A member of a fuzzy set has a membership value in $[0, 1]$. In a fuzzy set, a membership function defines the grade of memberships of its elements. So formally we may define that a membership function on X is any function $\mu(x) : X \mapsto [0, 1]; x \in X$. We formally define a fuzzy set A as $A = \{\langle x, \mu(x) \rangle | x \in X\}$ [127]. The fuzzy set theory allows modeling the linguistic variables and rules with natural language. With respect to rule structures there are three types of FRBSs: Mamdani Assilian (MA) type, Takagi Sugeno Kang (TSK) type, and classifier type [125,126].

In the MA model, both antecedent propositions and consequent propositions of a rule are defined by linguistic variables. A rule here is of the following form.

$$\text{If } x_1 \text{ is } F_{A_1} \text{ and } x_2 \text{ is } F_{A_2} \text{ and } \dots x_P \text{ is } F_{A_P} \text{ Then } y \text{ is } F_C.$$

where, x_i s and y are linguistic variables and F_{A_i} s and F_C are linguistic values. The components of a MA model are shown in Fig. 1.1. A crisp input is fuzzified to linguistic values through the fuzzification process. The fuzzified input is processed in the inference engine according to a set of ‘If Then’ fuzzy rules. The fuzzy rules are stored

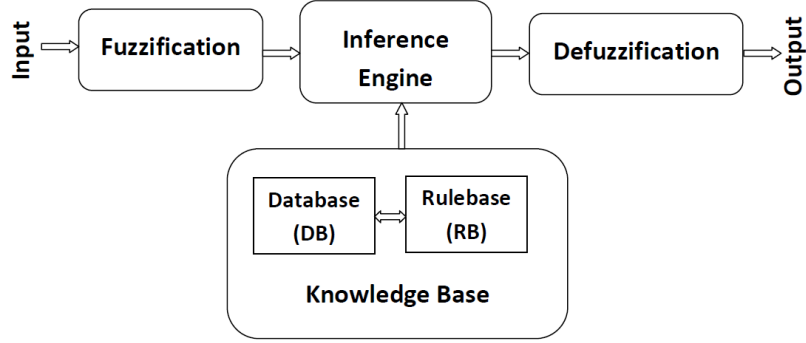


Figure 1.1: The components of MA model

in rulebase (RB) and the corresponding definitions of the fuzzy sets and parameters of the membership functions are stored in a database (DB). RB and DB together form the knowledge base. The outputs from the inference engine are also linguistic values. They are defuzzified to crisp values through the defuzzification process. The TSK model is almost similar to the MA model, except here, the consequent does not involve any fuzzy set rather the consequent is a function of the input variables. For the TSK model, a rule is of the following form.

If x_1 is F_{A_1} and x_2 is F_{A_2} and $\dots x_P$ is F_{A_P} Then $y = h(x_1, x_2, \dots, x_P)$.

Since there is a function at the consequent part for the TSK model, the output y is a crisp value, no need to defuzzify it. However, for a given input, different rules will produce different outputs which need to be aggregated to obtain the final rule-base output. For classification rules, also the antecedent remains the same but the consequents are class labels. Thus the classification rules take the following form.

If x_1 is F_{A_1} and x_2 is F_{A_2} and $\dots x_P$ is F_{A_P} Then class is k .

We note that the TSK model can also be used for classifications.

A TSK-type FRBS is used in the work proposed in Chapter 3 of our thesis. The proposed FRBS generates a lower dimensional representation of input data preserving the underlying manifold structure in an unsupervised learning setting. A brief working principle of a TSK-type FRBS would be discussed in Chapter 3. We have also explored FRBS as a classifier with an embedded feature selection mechanism in our work for Chapter 5 of this thesis. The working details of FRBS as a classifier would be discussed in Chapter 5.

1.4.2 Multi-layer Perceptron

An artificial neural network (ANN) is a tool that mimics the brain to some extent for performing a task. An ANN comprises simple computing units called ‘neurons’ or

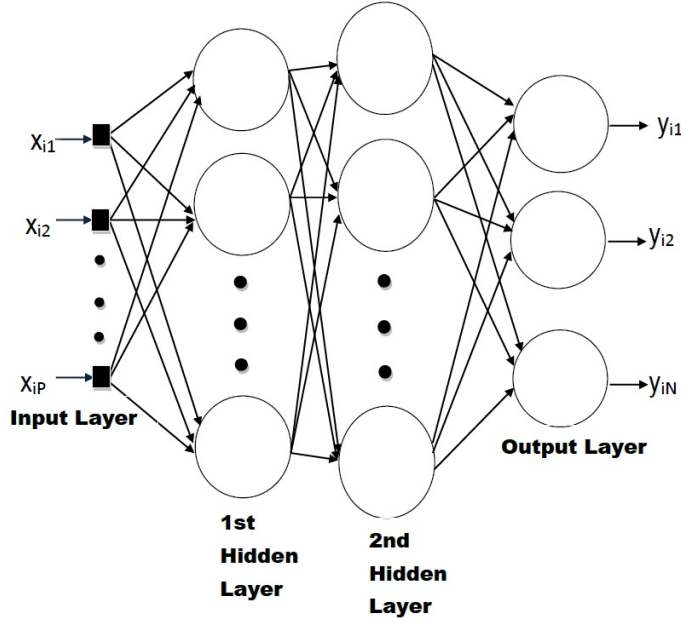


Figure 1.2: Architecture of multi-layer perceptron network with two hidden layers

‘processing units’ that are interconnected to build a massive network. ANN can learn from its environment and store the knowledge as the brain does [128]. A multi-layer perceptron (MLP) is an ANN having layered architecture and of feedforward type. It has one or more hidden layers between the input and output layers [128–130]. The architecture of an MLP is shown in Fig.1.2. Let us discuss how an MLP learns from a given input-output data set. Here we are considering the supervised learning paradigm. So, the input data are provided with the corresponding target output. Let the input data be $\mathbf{X} = \{\mathbf{x}_i \in \mathbb{R}^P\}_{i=1}^n$, where, $\mathbf{x}_i = (x_{i1}, x_{i2}, \dots, x_{iP})$, a P dimensional row vector. The target data be $\mathbf{T} = \{\mathbf{t}_i \in \mathbb{R}^Q\}_{i=1}^n$, where, $\mathbf{t}_i = (t_{i1}, t_{i2}, \dots, t_{iQ})$, an Q dimensional row vector. t_i is the output corresponding to $\mathbf{x}_i$. For a function approximation task, $\mathbf{t}_i$ would be the value of the function corresponding to the i^{th} input instance $\mathbf{x}_i$. For a classification task $\mathbf{t}_i$ would be the class label vector of $\mathbf{x}_i$. Essentially the input layer has P nodes, the same as the input dimension and the output layer has Q nodes, the same as the dimension of the target outputs. Let the output obtained from the neural network be $\mathbf{Y} = \{\mathbf{y}_i \in \mathbb{R}^Q\}_{i=1}^n$, where, $\mathbf{y}_i = (y_{i1}, y_{i2}, \dots, y_{iQ})$, a Q dimensional row vector. The input layer is a fan-out layer. As the i^{th} instance $\mathbf{x}_i$ is applied to the MLP, it fans out the input as it is. As per Fig.1.2, let us consider that the MLP has two hidden layers with H_1 and H_2 number of nodes, respectively. Each circle in the hidden layers and the output layer (Fig.1.2) is a neuron or processing unit. At each neuron, the weighted inputs from the outputs of the previous layer are summed and the sum is then applied to a special function called the activation function. The value produced by the activation function is the output of a neuron. The outputs of the neurons in a

layer are the input to each neuron of the next layer. Let the activation function of the neuron be represented by $\mathcal{S}()$. Some of the popular activation functions are sigmoidal, hyperbolic tangent, radial basis functions, and rectified linear unit (ReLU) [131]. With $\mathbf{x}_i$ as the input to the MLP, the input to the k^{th} node of the first hidden layer is given by,

$$s_{ik,1} = \sum_{j=1}^P w_{jk,1} x_{ij} + w_{k,1}; \quad k = 1, 2, \dots, H_1. \quad (1.4.1)$$

$w_{jk,1}$ is the synaptic weight between the j^{th} input node and the k^{th} neuron of the first hidden layer. $w_{k,1}$ is the bias term for the k^{th} neuron of the first hidden layer. The k^{th} neuron of the first hidden layer, passes the input $s_{ik,1}$ through an activation function. Considering the sigmoidal activation function, the output of the k^{th} neuron of the first hidden layer is given by.

$$h_{ik,1} = \frac{1}{1 + \exp(-s_{ik,1})}; \quad k = 1, 2, \dots, H_1. \quad (1.4.2)$$

Similarly, the input and output of the l^{th} neuron of the second hidden layer are respectively given by

$$s_{il,2} = \sum_{k=1}^{H_1} w_{kl,2} h_{ik,1} + w_{l,2}; \quad l = 1, 2, \dots, H_2; \quad (1.4.3)$$

$$h_{il,2} = \frac{1}{1 + \exp(-s_{il,2})}; \quad l = 1, 2, \dots, H_2. \quad (1.4.4)$$

where, $w_{kl,2}$ is the synaptic weight between the k^{th} node of the first hidden layer and the l^{th} neuron of the second hidden layer and $w_{l,2}$ is the bias term for the l^{th} neuron of the second hidden layer. If there were more hidden layers, the inputs, and outputs would be calculated in a similar way. For the MLP shown in Fig.1.2, the final output signal $\mathbf{y}_i = (y_{i1}, y_{i2}, \dots, y_{iQ})$ is calculated as

$$s_{im} = \sum_{l=1}^{H_2} w_{lm} h_{il,2} + w_m; \quad m = 1, 2, \dots, Q; \quad (1.4.5)$$

$$y_{im} = \frac{1}{1 + \exp(-s_{im})}; \quad m = 1, 2, \dots, Q. \quad (1.4.6)$$

Thus, the instantaneous error for the i^{th} instance is

$$E_i = \sum_{m=1}^N (t_{im} - y_{im})^2. \quad (1.4.7)$$

The loss in (1.4.7) is the square-loss function. There are many different loss functions that are used according to requirements and applications [132]. The synaptic weights of an MLP are initialized randomly. During the learning phase, the weights are updated

to reduce the loss. The back-propagation learning is one such learning paradigm that is extremely popular for training an MLP [128–130]. MLPs are used in various pattern recognition problems such as classification, regression, sequence generation, and so on. In the study covered in Chapter 2 of this thesis, we have proposed a method for learning an explicit map for dimensionality reduction employing MLP as a function approximator. We have proposed an embedded feature selection scheme that simultaneously discriminates classes and preserves the cluster structure using an MLP in Chapter 4 of our thesis.

1.5 Scope and organization of the thesis

As we have mentioned previously, our thesis broadly deals with the dimensionality reduction methods primarily designed for visualization and classification. In Chapters 2 and 3, we have proposed feature extraction based methods, and in Chapters 4 and 5, we have proposed feature selection based methods. To be specific, in Chapters 2 and 3, we have addressed some issues related to some common manifold learning algorithms, a special category under the feature extraction methods [133]. As the common manifold learning algorithms are widely used for data visualization, the application area for Chapters 2 and 3 is data visualization. The works in Chapters 4 and 5 have proposed feature selection methods for classification tasks but the methods in addition to focusing solely on classification performance, handle some other important aspects. In the following, we present chapter-wise brief discussions on the problem statements and our contributions to solving those problems.

1.5.1 Chapter 2: Supervised learning of explicit maps with ability to correct distortions in the target output for manifold learning [1]

Most manifold learning algorithms [42, 43, 77, 134] are non-parametric, they use unsupervised approaches, and hence lack prediction capability. Some of these methods have been adapted for out of sample points, but often they require the original training data. We propose a framework to estimate an explicit map for manifold learning in a supervised setting. We consider outputs from an unsupervised manifold learning method as the target in our supervised learning framework. However, even for simple data, often the outputs (target) get severely distorted. For high-dimensional data, it is difficult to assess and correct if the target data are distorted. To address these problems our proposed method in this chapter contribute in the following way.

1. We propose a supervised framework for learning an explicit map for manifold learning-based dimensionality reduction. Even if some of the target output data are distorted versions of the underlying manifold structure and they violate the actual neighborhood relationships for some regions, our model is capable of removing the distortion and restoring the actual neighborhood structure to a great extent. In this context, three variants of a structure-preserving regularizer have been proposed.

2. The proposed framework is a general one. We have explored the polynomial function and MLP to learn the explicit maps but other machine learning tools can also be used.
3. For high-dimensional data like image data, the model complexity of polynomial function having degree three or more is enormous. Our model exploits the spatial characteristics of image pixels and reduces the model complexity significantly.

1.5.2 Chapter 3: Nonlinear dimensionality reduction for data visualization: An unsupervised fuzzy rule-based approach [2, 3]

Dimensionality reduction methods that are intended to be used as data visualization tools in real-world problems should satisfy at least the following desirable characteristics: **D1**: preserve the *local geometry* of the original data in the lower dimension; **D2**: can make a faithful projection of test data points in a *straightforward* manner without requiring the training data; **D3**: unsupervised in nature; **D4**: can refuse to project test points which are coming from an area far from the “sampling window” of the training data, and **D5**: are interpretable/understandable at least to some extent. Our proposed methods in Chapter 2 although satisfy **D1** and **D2**, but they do not satisfy **D3**, **D4**, and **D5**. In this chapter, we propose a general framework for the unsupervised fuzzy rule-based dimensionality reduction which enjoys the following advantages.

1. It is an unsupervised system, yet it learns an explicit parametric system for projecting high dimensional data into a lower dimensional space. Thus, out-of-sample instances can be projected in a straightforward manner.
2. It can refuse to project an out-of-sample instance, when it is far away from the sampling window of the training data, i.e., the system is equipped with an appropriate reject option. Most machine learning systems cannot do so.
3. The system is reasonably transparent as it is based on fuzzy reasoning, where each rule-antecedent represents a small hyper-ellipsoidal region in the input space and its consequent is a local linear function of the inputs.
4. Because of the structure of fuzzy reasoning, the generalization ability of the proposed system is very good.
5. The proposed model is very robust to the initial condition.
6. It provides a general framework for fuzzy rule-based unsupervised dimensionality reduction. We can use different objective functions to preserve different characteristics of the original data in the projected space. For example, in [2], we have used conventional Sammons’ Stress [41] as the objective function, and in [3] we have used a modified form of Sammons’ Stress involving geodesic distances among the input points.

Finally, to the best of our knowledge, this is the first unsupervised method based on fuzzy rules for nonlinear manifold learning. The proposed method performs better or its performance is comparable to several state of-the-art non-fuzzy methods.

1.5.3 Chapter 4: Feature selection simultaneously preserving both class and cluster structures [4]

Based on whether the class label information is used or not, feature selection methods can be broadly classified as supervised and unsupervised. Supervised feature selection methods use the class label information and aim at selecting features that make different classes discernible. Generally, these methods do not consider or ensure the separability of the clusters, that may be present in the data, explicitly. On the other hand, unsupervised feature selection schemes are generally used when the underlying problem is not a classification problem, and naturally, class information is either not available or not considered. These methods usually focus on preserving data distribution, e.g. cluster structures. However, for a dataset when class and cluster structures are not identical or similar, the available supervised and unsupervised feature selection methods cannot guarantee reasonably high classification scores and good cluster separability simultaneously. For a practical system, the data samples represented with the selected features may be used in a classifier and also for clustering. For example, given gene expression data for a set of cancers, users may want to develop a diagnostic system (classifier) as well as to find subtypes (clustering), if present, of a given cancer. We could not find any feature selection method that focuses both on class and cluster separability. To bridge this gap, we propose a feature selection method, governed by an objective function that consists of classification error and a regularizer to preserve the cluster structure. With a proper choice of the multiplier to the structure-preserving regularizer, we choose a set of features that is good enough in terms of both classification performance and structure preservation.

1.5.4 Chapter 5: Understanding the classes better with class-specific and rule-specific feature selection, and redundancy control in a fuzzy rule based framework [5]

Recently, several studies [135–142] have claimed that using class-specific feature subsets provides certain advantages over using a single feature subset for representing the data for a classification problem. Unlike traditional feature selection methods, the class-specific feature selection methods select an “optimal” feature subset for each class. Typically class-specific feature selection (CSFS) methods use one-versus-all (OVA) split of the data set that leads to issues such as class imbalance, decision aggregation, and high computational overhead. In this chapter, we propose a CSFS scheme embedded in a fuzzy rule-based classifier (FRBC) that does not use the OVA strategy. Our main contributions in this study are summarised as follows.

1. We propose an embedded CSFS method that is not based on the OVA strategy

like most of the existing CSFS schemes. Thus, our method is free from the weaknesses of the OVA strategy.

2. We propose a classifier framework that is defined by class-specific fuzzy rules involving class-specific feature subsets.
3. Our method can monitor the level of redundancy in the selected features.
4. We also propose a generalized version of our class-specific feature selection method that not only chooses different subsets for different classes but also, chooses different subsets for different rules within a class if different substructures are present.

1.5.5 Chapter 6: Conclusions, limitations, and future scopes

In Chapter 6, we conclude our thesis by mentioning the limitations of the works done and the potential areas for future works.

Chapter 2

Supervised learning of explicit maps with ability to correct distortions in the target output for manifold learning [1]

2.1 Introduction

A manifold is a topological space that locally resembles Euclidean space [143]. In the real world, many high dimensional data sets may essentially lie on low dimensional manifolds; i.e., the low dimensional manifold is embedded in a high dimensional space. A class of non-linear feature extraction methods primarily used for visualization is categorized as manifold learning. Manifold learning aims to identify the intrinsic lower-dimensional structure or manifold on or around which the high dimensional data points lie. Manifold learning algorithms are of great importance as they serve several application areas such as aligning and clustering of gene network [144], 3D body pose recovery [145], face recognition [146], and data visualization [147]. In manifold learning for data visualization, the objective is to produce a low-dimensional (usually two or three) representation of the high-dimensional data by preserving the local neighborhood (local geometry). Local Linear Embedding (LLE) [43], ISOMAP [42], Laplacian Eigenmaps (LE) [134], Maximum Variance Unfolding (MVU) [77], Hessian LLE (HLLE) [81], and Local Tangent Space Analysis (LTSA) [82] are some popular manifold learning methods. However, these popular manifold learning methods are non-parametric, i.e., they map the original input data into a lower dimensional output space implicitly. The absence of an explicit map leads to various limitations. *First*, the cost associated with projecting new/unseen test points is not $O(1)$ as in the case of explicit mapping methods. To generate outputs for the test points, an implicit method repeats its whole process considering the training and the test points together leading to enhanced computational cost. Moreover, this demands the availability of the

training data at the time of projecting a new test point. *Second*, for implicit methods generalization ability cannot be analyzed. *Third*, no provision for constructing inverse mapping functions in the case of implicit methods. For explicit maps, sometimes it is possible to determine an inverse map at least locally. The availability of an inverse map may determine pre-image for novel output points and help to get deeper insight regarding the mapping. With the help of the inverse map, it is possible to obtain a reconstructed version of the original inputs from the lower dimensional outputs. The dissimilarity between the original and reconstructed input helps to analyze the information preservation capability of the underlying mapping method [10, 70].

To avoid the difficulties associated with the embedding of new points, for many of the implicit manifold learning methods, out-of-sample extensions have been proposed [70, 148–150]. But all of those extensions are not necessarily based on explicit maps. Some of the out-of-sample extension methods need all the training points to predict the outcome for new instances. These methods do not provide much improvement in the computational and space requirements. On the other hand, explicit maps, store the information extracted from the training data in the trained model. Storing model parameters is generally much more space-efficient than storing the entire training set. In [70] a general framework for finding explicit maps to deal with out-of-sample mapping is proposed. In general, learning an explicit mapping function could be supervised or unsupervised. A supervised approach requires the target or desired output. In the considered problem the target output is the lower dimensional representation of the high dimensional input produced by the concerned manifold learning algorithm. When the target data are not at hand, the explicit map is to be learned in an unsupervised framework. Examples of unsupervised explicit manifold learning methods are [3, 49, 83, 151]. The authors of [151] proposed a framework for training a feed-forward neural network with the objective function of the method t-SNE [45]. The study in [83] presents a polynomial function-based explicit map which is learned by minimizing the objective function of LLE. A “generalized autoencoder” was proposed in [49] which incorporates the concept of manifold learning in the autoencoder framework. The work in [3] explores a generalized fuzzy rule-based framework for manifold learning.

When low-dimensional target outputs corresponding to high-dimensional input points are given, the task of learning an explicit map simply becomes a problem of “function approximation through regression”. In other words, given the input data matrix $\mathbf{X} \subset \mathbb{R}^P$ and its corresponding lower-dimensional output data matrix $\mathbf{Y} \subset \mathbb{R}^Q, Q < P$, the task at hand is to approximate the function $\phi : \mathbb{R}^P \mapsto \mathbb{R}^Q$. $\mathbb{R}^P$ and $\mathbb{Y}$ are the input space and the output space respectively. One may question if such a method would perform well on future data points. As long as the input data points are generated from a time-invariant distribution, such a method would do a good job. In many practical scenarios, we may easily generate $\mathbf{Y}$ for a given input data using a standard manifold learning scheme but may not be interested to repeat the whole procedure with $\mathbf{X} \cup \mathbf{X}^{\text{New}}$, where $\mathbf{X}^{\text{New}}$ is the set of new input points. The procedure would be more cumbersome if the new input points appear one by one separated by some time interval but the lower-dimensional representation corresponding to a new point

is to be produced as soon as it appears. To solve this problem, a possible way is to train a parametric model with the aid of the input-output data set pair $(\mathbf{X}, \mathbf{Y})$. The parametric model realizes the map ϕ explicitly and hence directly can estimate $\mathbf{Y}^{\text{New}}$. There have been a few attempts [29, 152] using neural networks and fuzzy logic to learn such mappings for generating lower dimensional representation minimizing Sammon's stress function [41]. Note that Sammon's stress function preserves inter-point distances with more emphasis on short distances which leads to the preservation of some sort of local structure but not really in a manifold learning sense. The methods proposed in [28, 153] illustrate supervised learning of explicit maps for manifold learning. In [153], following [83] the mapping ϕ is considered a polynomial function. The target output $\mathbf{Y}$ is generated using the graph embedding framework proposed in [154]. The coefficients of the polynomial are estimated by minimizing the square loss with $L1$ -regularizer. In the autoencoder-based model in [28], the encoder is trained in a supervised framework with the diffusion map [155] embedding of the input data as the target output $\mathbf{Y}$. Similarly, the decoder is trained to reconstruct the high-dimensional input from the low-dimensional diffusion map embedding.

In this work, we consider a supervised framework to learn an explicit map that embeds the high-dimensional input data into a lower-dimensional space preserving the local neighborhood structure of the input. In other words, through supervised learning, we aim to learn an explicit map that approximates the underlying manifold corresponding to an input data set. In practice, the explicit map should be realized through a nonlinear function to unfold the nonlinear input structure if any. As mentioned previously, [83] and [153] employed a polynomial-based regression model to build an explicit map for manifold learning. Unlike MLP or fuzzy rule-based models, a polynomial regression-based model leads to a convex objective function eliminating the chances of obtaining a local minima as the solution. This motivates us to consider first the polynomial model. However, the proposed approach is applicable to models like neural networks and fuzzy systems also. When considering a supervised model an important factor is the appropriateness or correctness of the target output data. Many of the manifold learning algorithms are based on graph spectral decomposition of the neighborhood graphs. These methods sometimes introduce some structural distortions or disagreement of the neighborhood relationships between input and output, due to their nature or improper choice of the neighborhood size [156]. So, when taking the outputs of those methods as the target outputs in a regression-based model, even for the best fit model those distortions/ disagreements would propagate in the output generated by the model. This immediately raises a question, if that is true (which can be known only for synthetic data) how can we fix the distortion? To address this we have added a structure-preserving regularizer term to the usual square error loss function of the regression model. As regularizers, we have chosen the loss function of the method Laplacian Eigenmap [134] called the Laplacian with three variants of edge weighting functions of the neighborhood graph or Laplacian graph to rectify the structural distortions and restore the disturbed neighborhood relations, if any. The use of the Laplacian matrix to characterize local structures for manifold learning is quite well established [157–159].

The literature on the use of hypergraph Laplacian for various tasks that exploit neighborhood relations is quite rich. For example, in [160] a hypergraph Laplacian is used for image classification using a transductive learning framework. On the other hand, Wu et al. in [161] used hypergraph Laplacian for image click prediction which is then used for reranking of images. The overall objective function of our system including the regularizer is convex and has a direct analytical solution. A unique characteristic of this approach is that even when the target data have distortion (often it is the case), the proposed system can rectify that to a great extent along with providing a prediction system. In the case of image data, the dimension of the inputs is generally high which not only causes computational burden but also demands a large number of training instances. To address this issue, we have exploited the spatial characteristics of image pixels, which reduces the number of learnable parameters significantly. The proposed framework is quite general where in place of the polynomials, other explicit models can be used. In this study, we have also implemented the framework with an MLP network and shown its effectiveness on synthetic and image data sets. We can use other machine learning models like radial basis function networks with the proposed framework.

2.2 Proposed Method

Let the input data be denoted by an $n \times P$ matrix, $\mathbf{X} = \{\mathbf{x}_i \in \mathbb{R}^P\}_{i=1}^n$. Here, $\mathbf{x}_i$ is a P -dimensional row vector of the form $\mathbf{x}_i = (x_{i1}, x_{i2}, \dots, x_{iP})$. We assume that the input points $\mathbf{x}_i$ s essentially lie in a Q -dimensional manifold embedded in a P -dimensional space, where $Q < P$. Since we consider a supervised setting, the target outputs of our system, that is, the Q -dimensional representations corresponding to $\mathbf{X}$ are available. Let the target output data be $\mathbf{Y} = \{\mathbf{y}_i \in \mathbb{R}^Q\}_{i=1}^n$, a matrix of size $n \times Q$. $\mathbf{y}_i$ is a Q -dimensional row vector of the form $\mathbf{y}_i = (y_{i1}, y_{i2}, \dots, y_{iQ})$. The relation between the input and output data points may be linear or non-linear. Even data sets without labels or targets may have nonlinear structures, i.e., the data structure cannot be well understood using linear projections of the high dimensional data to lower dimensions. Of course, there are real-life data sets on which linear approaches may work pretty well. However, non-linear function based models such as polynomials, neural networks, and fuzzy systems are more general and can handle both linear and non-linear structures. Hence, we consider the non-linear function that maps $\mathbf{x}_i$ to $\mathbf{y}_i$ as an l^{th} degree polynomial function. Let the estimated value of $\mathbf{y}_i$ by the polynomial function be $\hat{\mathbf{y}}_i$. The m^{th} component, $\hat{y}_{im}$, of $\hat{\mathbf{y}}_i$ is given by,

$$\hat{y}_{im} = \sum_{\substack{k=(k_1, k_2, \dots, k_P) s.t. \\ k_r \in \mathbb{I} \\ 0 \leq k_r \leq l, r=1 \text{ to } P \\ 0 \leq \sum_{r=1}^P k_r \leq l}} b_{km} x_{i1}^{k_1} x_{i2}^{k_2} \dots x_{iP}^{k_P} \quad (2.2.1)$$

where $\mathbb{I}$ is the set of non-negative integers, b_{km} s are coefficients of the polynomial function. Considering all Q components of $\hat{\mathbf{y}}_i$, we can extend equation (2.2.1) to the

following.

$$\hat{\mathbf{y}}_i = \left(\sum b_{k1} x_{i1}^{k_1} x_{i2}^{k_2} \cdots x_{iP}^{k_P}, \sum b_{k2} x_{i1}^{k_1} x_{i2}^{k_2} \cdots x_{iP}^{k_P}, \cdots, \sum b_{kQ} x_{i1}^{k_1} x_{i2}^{k_2} \cdots x_{iP}^{k_P} \right) \quad (2.2.2)$$

The summation conditions in equation (2.2.2) are the same as the summation condition in equation (2.2.1). They are omitted to keep the equation neat. An l^{th} degree polynomial function involves terms from the first degree to l^{th} degree and a constant term. Let T be the total number of possible terms of an l^{th} degree polynomial. So we can write a matrix $\mathbf{B}$ of size $T \times Q$ which is a collection of all coefficients b_{km} s, for $m = 1$ to Q and for all possible values of k , involved in the l^{th} degree polynomial map. We can effectively express (2.2.2) as the following.

$$\hat{\mathbf{y}}_i = \left(1, x_{i1}, x_{i2}, \cdots, x_{iP}, \cdots, x_{i1}^l, x_{i1}^{l-1} x_{i2}, \cdots, x_{i1}^{l-1} x_{iP}, \cdots, x_{iP}^l \right) \mathbf{B} \quad (2.2.3)$$

$$= \mathbf{a}_i \mathbf{B} \quad (2.2.4)$$

Let us define, $\mathbf{A} = \{\mathbf{a}_i\}_{i=1}^n$, a $n \times T$ matrix and $\hat{\mathbf{Y}} = \{\hat{\mathbf{y}}_i\}_{i=1}^n$, a $n \times Q$ matrix. Then,

$$\hat{\mathbf{Y}} = \mathbf{A} \mathbf{B} \quad (2.2.5)$$

Under the supervised settings, the optimum values of the model parameters, i.e., the coefficients of the polynomial map, $\mathbf{B}$ are obtained by minimizing the square error involving the target outputs and the estimated outputs.

$$\mathbf{B}^* = \arg \min_{\mathbf{B}} E_{ls}(\mathbf{B}) \quad (2.2.6)$$

$$= \arg \min_{\mathbf{B}} \frac{1}{2} \sum_{i=1}^n \|\mathbf{y}_i - \hat{\mathbf{y}}_i(\mathbf{B})\|_{L_2}^2 \quad (2.2.7)$$

$$= \arg \min_{\mathbf{B}} \frac{1}{2} \sum_{i=1}^n \|\mathbf{y}_i - \mathbf{a}_i \mathbf{B}\|_{L_2}^2 \quad (2.2.8)$$

We can also express E_{ls} in terms of $\mathbf{Y}$ and $\hat{\mathbf{Y}}$.

$$E_{ls} = \frac{1}{2} \|\mathbf{Y} - \hat{\mathbf{Y}}\|_F^2 \quad (2.2.9)$$

$$= \frac{1}{2} \|\mathbf{Y} - \mathbf{A} \mathbf{B}\|_F^2 \quad (2.2.10)$$

As discussed earlier when target outputs $\mathbf{Y}$ are generated by any spectral manifold learning algorithm or by any other manifold learning method there are chances of having structural distortions, which can damage the neighborhood relationships in the lower-dimensional representation. Unless the data are synthetically generated and effectively lie in two or three dimensions, we would not be able to know if there are such distortions. Any regression model that minimizes E_{ls} alone, at its best can learn the target data perfectly. Consequently, the role of the target data becomes very important. However, if

the target data suffer from any undesirable distortion, the regression model may not be able to correct those distortions. For manifold learning, as we shall see, such distortions happen. There could be many possible causes such as limitations of the model, improper choice of neighborhood, and dependence on the initial condition. Our objective is to deal with the distorted targets properly (without knowing explicitly which targets are distorted) so that the identified regression model is not significantly affected by the distorted targets and the regression outputs corresponding to the distorted targets are corrected.

To address this problem, we propose the addition of a suitable local structure-preserving regularizer with the square error term. Thus, in addition to the targets, we also exploit the information on the distribution of the inputs to modify the loss function. The regularizer that we use has the following form.

$$E_{reg} = \frac{1}{4} \sum_{i=1}^n \sum_{j=1}^n \|\hat{\mathbf{y}}_i - \hat{\mathbf{y}}_j\|_{L2}^2 w_{ij}. \quad (2.2.11)$$

This expression in (2.2.11) is used in [134] as an objective function for manifold learning. An edge weight of the neighborhood graph w_{ij} is a function of $\mathbf{x}_i$ and $\mathbf{x}_j$ and captures the local geometry of the data. The collection of w_{ij} s for $i = 1$ to n and $j = 1$ to n , defines an $n \times n$ matrix $\mathbf{W}$ with $(i, j)^{th}$ entry as w_{ij} . In [134], w_{ij} is realized using a Gaussian function and a simple 1-0 function. We have explored three variants of w_{ij} . We use the Gaussian function, given in (2.2.12) along with two other functions: the inverse quadratic, given in (2.2.13), and inverse sub-quadratic, given in (2.2.14).

$$\begin{aligned} w_{ij} &= \exp \left\{ -\frac{\|\mathbf{x}_i - \mathbf{x}_j\|_{L2}^2}{2\sigma^2} \right\} & \text{if } \mathbf{x}_j \in \mathfrak{N}(\mathbf{x}_i) \\ &= 0 & \text{otherwise} \end{aligned} \quad (2.2.12)$$

$$\begin{aligned} w_{ij} &= \frac{1}{1 + \gamma \|\mathbf{x}_i - \mathbf{x}_j\|_{L2}^2} & \text{if } \mathbf{x}_j \in \mathfrak{N}(\mathbf{x}_i) \\ &= 0 & \text{otherwise} \end{aligned} \quad (2.2.13)$$

$$\begin{aligned} w_{ij} &= \frac{1}{\sqrt{1 + \gamma \|\mathbf{x}_i - \mathbf{x}_j\|_{L2}^2}} & \text{if } \mathbf{x}_j \in \mathfrak{N}(\mathbf{x}_i) \\ &= 0 & \text{otherwise} \end{aligned} \quad (2.2.14)$$

$\mathfrak{N}(\mathbf{x}_i)$ is a defined neighborhood around $\mathbf{x}_i$. For all three variants of $\mathbf{W}$, defined in (2.2.12), (2.2.13), and (2.2.14), the value of w_{ij} corresponding to two input points, $\mathbf{x}_i$ and $\mathbf{x}_j$ is the maximum (equals to 1) when they are coincident. As the distance between $\mathbf{x}_i$ and $\mathbf{x}_j$ increases, the value of w_{ij} falls off gradually and becomes zero when $\mathbf{x}_i$ and $\mathbf{x}_j$ are not within the predefined neighborhood. Due to this nature, the regularizer error function E_{reg} in (2.2.11), tries to place the nearby input points close and distant input points far apart in the generated lower-dimensional map. The overall objective

function that we minimize is the following.

$$\begin{aligned} E &= \alpha E_{ls} + \beta E_{reg} \\ &= \frac{\alpha}{2} \|\mathbf{Y} - \hat{\mathbf{Y}}\|_F^2 + \frac{\beta}{4} \sum_{i=1}^n \sum_{j=1}^n \|\hat{\mathbf{y}}_i - \hat{\mathbf{y}}_j\|_{L2}^2 w_{ij} \end{aligned} \quad (2.2.15)$$

The error function in (2.2.15) is convex. Using the first-order necessary condition of minimum and with some algebraic manipulation, the solution of minimization of (2.2.15) with respect to the model parameters $\mathbf{B}$ is,

$$\mathbf{B}^{opt} = (\alpha \mathbf{A}' \mathbf{A} + \beta \mathbf{A}' \mathbf{L} \mathbf{A})^{-1} (\alpha \mathbf{A}' \mathbf{Y}) \quad (2.2.16)$$

Here, $\mathbf{L} = \mathbf{D} - \mathbf{W}$, where $\mathbf{D}$ is a diagonal matrix with entries $d_{ii} = \sum_{j=1}^n w_{ij}$ for $i = 1$ to n . To avoid over-fitting a common practice is to add an $L2$ regularizer on the model parameters, i.e., the sum of squares of the model parameters to the main objective function. Adding the $L2$ regularizer to (2.2.15), the overall loss function becomes

$$E = \alpha E_{ls} + \beta E_{reg} + \theta E_{L2} \quad (2.2.17)$$

$$= \frac{\alpha}{2} \|\mathbf{Y} - \hat{\mathbf{Y}}\|_F^2 + \frac{\beta}{4} \sum_{i=1}^n \sum_{j=1}^n \|\hat{\mathbf{y}}_i - \hat{\mathbf{y}}_j\|_{L2}^2 w_{ij} + \frac{\theta}{2} \|\mathbf{B}\|_F^2 \quad (2.2.18)$$

The objective function in (2.2.17) is still a convex function with its optimal solution as follows:

$$\mathbf{B}^{opt} = (\alpha \mathbf{A}' \mathbf{A} + \beta \mathbf{A}' \mathbf{L} \mathbf{A} + \theta \mathbf{I})^{-1} (\alpha \mathbf{A}' \mathbf{Y}) \quad (2.2.19)$$

$\mathbf{I}$ is an Identity matrix of size $T \times T$.

For solving Eqn. 2.2.19 or 2.2.16, the major computational steps and their corresponding complexities are as follows:

1. Generation of the matrix $\mathbf{L}$ has the computational complexity $O(n^2 P)$.
2. Computational complexity for generating the matrix $(\alpha \mathbf{A}' \mathbf{A} + \beta \mathbf{A}' \mathbf{L} \mathbf{A} + \theta \mathbf{I})$ is $O(nT^2 + n^2 T)$.
3. Inversion of the matrix $(\alpha \mathbf{A}' \mathbf{A} + \beta \mathbf{A}' \mathbf{L} \mathbf{A} + \theta \mathbf{I})$ or $(\alpha \mathbf{A}' \mathbf{A} + \beta \mathbf{A}' \mathbf{L} \mathbf{A})$ has the computational complexity $O(T^3)$, where T is the total number of terms in the l^{th} degree polynomial and $T = \binom{P+l}{l}$.
4. Computational complexity for generating the matrix $(\alpha \mathbf{A}' \mathbf{Y})$ is $O(nT)$.
5. The final multiplication (multiplication of the matrices in the above two steps) for computation of $\mathbf{B}^{opt}$ involves $O(T^2)$.

For $T \gg n$ and for n and T of comparable order, the overall computational complexity is $O(T^3)$ and for $n \gg T$, the overall computational complexity is $O(n^2 T)$.

We note here that, in [159] also, a Laplacian regularized polynomial regression is proposed for wireless sensor network localization. But there is an essential difference between our approach and the method in [159]. The authors estimated an explicit polynomial function for mapping high-dimensional signal space to a two-dimensional physical space. Formalizing in our notation, the signal space is characterized so that the i th high-dimensional vector $\mathbf{x}_i$ consists of the geodesic distances of the i^{th} node from P anchor nodes and $\mathbf{y}_i$ is a two-dimensional vector representing the physical location. So, x_{iz_r} is the geodesic distance between i th node and r th anchor node or in general say z_r th node. The expression in (2.2.11) is used as the regularizer with w_{ij} defined as the following.

$$w_{ij} = \exp \left\{ -\frac{g_{ij}^2}{2\sigma^2} \right\}, \quad (2.2.20)$$

where, $g_{ij} = x_{ij}$ if j is an anchor node. Here, $g_{ij} \neq \|\mathbf{x}_i - \mathbf{x}_j\|_{L2}^2$. So, the two differences from (2.2.12) are: i) w_{ij} does not exploit any neighborhood to define it and ii) w_{ij} is not a function of $\|\mathbf{x}_i - \mathbf{x}_j\|_{L2}^2$. The regularizer term in [159] ensures to place $\mathbf{y}_i$ s in accordance with the global geodesic distances where $\mathbf{y}_i$ s are actually a nonlinear function of geodesic distances. In our formulation, the regularizer defined by (2.2.11) associates a penalty on placement of a pair $(\mathbf{y}_i, \mathbf{y}_j)$ according to $\|\mathbf{x}_i - \mathbf{x}_j\|_{L2}^2$ if $\mathbf{x}_i$ and $\mathbf{x}_j$ are neighbors. Moreover, [159] uses semi-supervised learning, while ours uses completely supervised learning.

Remark 1: One may think that our method is a kind of ‘knowledge distillation’. In knowledge distillation, the knowledge is transferred from a bigger model (teacher model) to a smaller one (student model) without significant loss [162, 163]. Keeping this interpretation in mind, we may view the unsupervised method as the bigger or the teacher model and the trained model as the smaller or student model because the knowledge of the unsupervised method is transferred to it. Moreover, the trained model is computationally efficient and can do the job of the unsupervised method for a given problem without much loss. So we may view it as distillation learning. On the other hand, for our problem, although the objective of the unsupervised method is good, it may not always produce faithful outputs and our trained model tries to fix that. Thus, the trained model is an efficient realization of the teacher model, but it can also improve the performance over that of the teacher, which may not be the typical case for ‘knowledge distillation’.

Remark 2: One may draw a connection between the proposed method and learning with inaccurate supervision, a class of weak supervision. Inaccurate supervision deals with learning when some of the training instances have incorrect labels [164–166]. In our problem also we are dealing with learning when the targets of some of the training data points may be severely distorted. Thus, it may be viewed as learning with inaccurate supervision. However, we are dealing with a regression type problem. Unlike class labels, for regression problems, for no training data instance, we can say that the target is correct or wrong. The truths about the targets are not binary; the target of every instance is likely to have some error. However, most targets may be accurate enough to solve the problem under consideration, while for the rest it may not be the case. This

makes the present problem different from the conventional weakly supervised learning, in particular learning with inaccurate supervision.

2.3 Experiment

2.3.1 Data Set

To demonstrate the effectiveness of the proposed method, we have considered two synthetic data sets and one real-world data set. As our work deals with nonlinear mapping and local structure preservation of the data, we have picked up the data sets that are commonly used for manifold learning studies. The synthetic data sets,

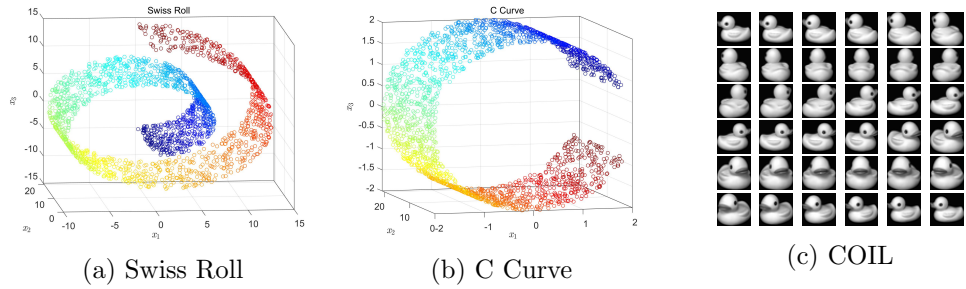


Figure 2.1: Data sets

as shown in Fig.2.1, are three-dimensional data but contained completely within a two-dimensional space. Each of these synthetic data sets consists of 2000 points. The Swiss Roll (Fig.2.1a) data set is generated using the ‘scikit-learn’ package [167] (version 0.19.2) of Python. The C Curve data set (Fig.2.1b) is generated by the equations: $x_1 = 2 \cos t$; $x_3 = 2 \sin t$, where t takes 2000 random values from the interval $(0.5, 2\pi - 0.5)$, and x_2 takes 2000 random values from the interval $(0, 21)$. The two synthetic data sets are topologically equivalent to a 2- D flat sheet. The real-world data set considered here is the COIL [168] data set. It consists of images of 20 different objects. For each object, 72 images are captured from 72 equally spaced angles i.e, each image is captured from five degrees apart views. We have considered the first object of the COIL data set (we shall refer to it as COIL). Each image is of resolution 32×32 . Thus, the input dimension is 1024, and the number of instances is 72. We refer to the consecutive views by consecutive numbers from 1 to 72. Figure 2.1c, shows 36 instances of the COIL data set chosen alternatively from the 72 instances.

2.3.2 Generation of Regression Target

As discussed in Sec.2.2, our work is based on supervised learning. We aim to generate an explicit non-linear map for manifold learning-based dimensionality reduction provided some target outputs are available. We also aim to rectify the structural defects present in the target outputs if any. We have extensively studied our method using the target

outputs generated by the LLE method. Later we shall also use targets generated by t -SNE and ISOMAP. *With the previously-mentioned data sets, we have tried different neighborhood sizes to generate targets with significant structural distortions.* Note that, with appropriate choices of parameters, all three algorithms, ISOMAP, t -SNE, and LLE can produce better projections. We have used distorted targets to investigate how effective the proposed method is to recover from the distortion. The LLE outputs for Swiss Roll, C Curve, and COIL data sets with three different choices of neighborhoods are shown in Fig.2.2.

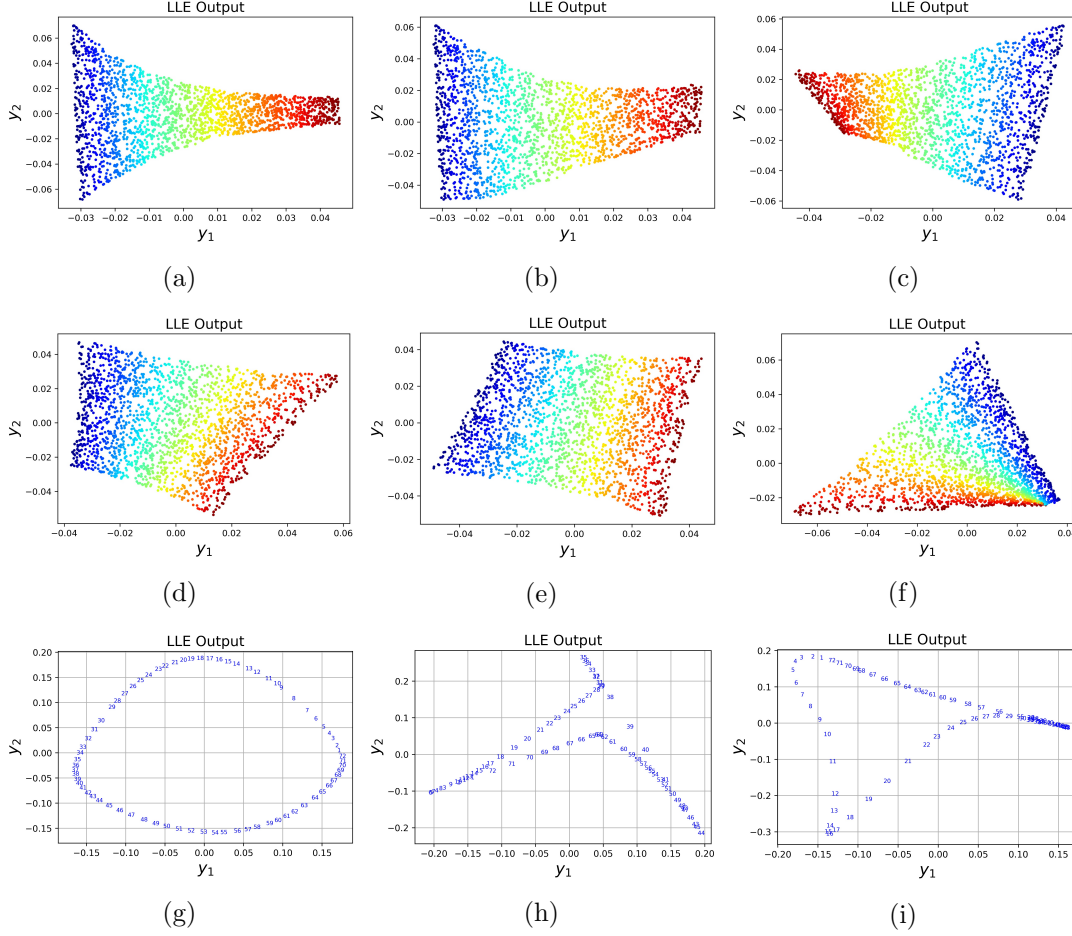


Figure 2.2: LLE representation: Swiss Roll with $\epsilon =$ (a)15, (b)20, (c)25; C Curve with $\epsilon =$ (d)15, (e)20, (f)25; COIL with $\epsilon =$ (g)2, (h)3, (i)4.

We have defined the neighborhood size in terms of the number of nearest neighbors, ϵ . For Swiss Roll and C Curve data sets ϵ is chosen as 15, 20, and 25. Ideally, the unfolded version of each of the two data sets should be approximately a two-dimensional rectangle. In the case of the Swiss Roll data set for $\epsilon = 15$ and 25, (Figs. 2.2a, 2.2c) it has unfolded quite asymmetrically. For the C Curve data set with $\epsilon = 25$, the lower-

dimensional output (Fig. 2.2f) is squeezed at one side. For these three outputs, we have investigated how effective the polynomial mapping, along with the regularizers, is in reducing the distortion to recover the desired structure. Figures 2.2g, 2.2h, and 2.2i show the results of the LLE outputs of the COIL data set for $\epsilon = 2, 3$, and 4, respectively. For COIL data set as mentioned in Sub-sec.2.3.1, instances are images of an object captured from five degrees apart consecutive views (See Fig. 2.1c), so its ideal two-dimensional embedding should be a circle formed of the instances where the instances with consecutive views should be placed next to each other.

In Figs. 2.2g, 2.2h, and 2.2i, each instance has been marked by its corresponding number as described in Sub-sec. 2.3.1. As seen in Fig. 2.2g, the LLE with $\epsilon = 2$ has embedded the input points in two-dimensions in a nearly circular form preserving the neighborhood structure. For $\epsilon = 3$ and 4, the LLE outputs, shown in Figs. 2.2h and 2.2i, clearly fail to preserve the neighborhood structure. As seen from Figs. 2.2h and 2.2i a section of points have almost merged to form spike-like extensions where local structures are not preserved. For example, in Fig. 2.2h, 15, 16, and 17 are nearest neighbors of 72 but in reality 1 and 71 are the two nearest neighbors of 72. We have done experiments on COIL data set with target outputs produced by LLE corresponding to $\epsilon = 3$ (Fig. 2.2h) to investigate whether such severe distortion in the targets can be corrected by the proposed method.

2.3.3 Experimental Setting

As discussed in Sec.2.2, our model employs a polynomial function to realize the required non-linear explicit map. We have used a polynomial function of degree 3 for the synthetic data sets. The form of $\mathbf{a}_i$, in this case, is the following.

$$(1, x_{i1}, x_{i2}, \dots, x_{iP}, x_{i1}^2, x_{i1}x_{i2}, \dots, x_{i1}x_{iP}, x_{i2}^2, x_{i2}x_{i3}, \dots, x_{i2}x_{iP}, \dots, x_{iP}^2, x_{i1}^3, x_{i1}^2x_{i2}, \dots, x_{i1}x_{iP}^2, \dots, x_{iP}^3) \quad (2.3.1)$$

In the case of the COIL data set, we have taken a polynomial function of degree 2 with certain restrictions. Here we have only 72 instances and the number of features $P = 1024$, even using a polynomial function of degree 2, we shall have too many parameters to be estimated. Hence, we use a judiciously chosen restriction to consider the product terms in the regression model. Let x_{ir} represent the r^{th} pixel of the i^{th} image. The terms of the form $x_{iu}x_{iv}$ are considered only if u and v are neighboring pixels based on a predefined neighborhood. We have defined such a setup because the interaction between two non-neighboring pixels may not be as useful as that between two neighboring pixels. The interaction between the neighboring pixels is very important to characterize an image as neighboring pixels define boundaries between two distinct regions, i.e., edges. Thus, the use of pixel pairs over a spatial neighborhood is likely to constitute good features to model a non-linear mapping from high-dimensional input (image) space to a lower-dimensional output space. In this investigation, we consider spatial neighborhoods of size 3×3 . If we do not consider product terms like $x_{iu}x_{iv}$ if u and v are non-neighboring pixels, it reduces the number of coefficients involved significantly. With input dimension, $P = 1024$, a polynomial function of degree 2 which includes

all the product terms, consists of 525,825 terms, and in turn, it needs to estimate $525,825 \times Q$ coefficients, which is computationally difficult and expensive. Moreover, for a reliable estimate of such a larger number of parameters we need a large training set. In our setup with a 3×3 neighborhood, the total number of terms reduces to 5,892, which is practically a useful one. To reduce the chances of overfitting, for COIL we have additionally explored the L2 regularized model with the loss function in (2.2.17).

The weights of the least square error term E_{ls} and the regularization error term E_{reg} are α and β respectively. We have set $\alpha = 1$ for all the experiments. In each experiment, we have also generated one result using only the least square error term E_{ls} shown in (2.2.9), i.e., without the regularizers. This helps us to understand the benefit of the proposed regularizers by comparing the results of unregularized and regularized cases. Different values of β have been tried to investigate the impact of the regularization term. The value of σ in (2.2.12) and γ in (2.2.13) and (2.2.14) decide the range of the input distance over which w_{ij} takes significant values. In other words, σ and γ control the limit on the input distance which is approximately preserved in the lower-dimensional representation. We have explored the effect of different values of σ and γ . For all three forms of the regularizer as defined in (2.2.12), (2.2.13), and (2.2.14), a neighborhood $\mathfrak{N}(\mathbf{x})$ has to be defined. We have defined $\mathfrak{N}(\mathbf{x})$ as the k nearest neighbors of $\mathbf{x}$. For all the experiments involving the regularizers, we have set $k = 10$.

2.4 Results and Discussion

Figures 2.3 and 2.4 show results with the Swiss Roll data set. The LLE output with the $\epsilon = 25$ shown in Fig. 2.3a is the regression target for the results shown in Fig. 2.3. Note that, the horizontal ends of the LLE output in Fig. 2.3a are skewed, in particular, the left end is heavily squeezed. Figure 2.3b shows that when no regularizer is applied, our model approximates the target output more or less as it is. Figs. 2.3c, 2.3d, and 2.3e are the output produced by the three different regularizers defined using (2.2.12), (2.2.13), and (2.2.14), respectively. For all three variants of regularizers, we find that the quality of the lower-dimensional output data improved in terms of skewness.

Figure 2.4 shows the results of our experiments on Swiss Roll using the regression target in Fig. 2.4a, which is generated by LLE with $\epsilon = 15$. In this case, the target output, shown in Fig. 2.4a, gets gradually compressed from the left to the right. As mentioned earlier, the ideal output should be a rectangle on the 2D plane, but the target output under consideration shows a significant change in width between the left and the right ends. The output without any regularizer is shown in Fig. 2.4b, which reveals a further increase in the (left to right) width-variation-degradation. Application of the regularizers in all four cases (Figs. 2.4c (Gaussian, $\sigma = 0.5, \beta = 500$), 2.4d (Gaussian, $\sigma = 0.5, \beta = 1000$), 2.4e (Inverse quadratic $\gamma = 30, \beta = 1000$), 2.4f (Inverse Sub-quadratic $\gamma = 100, \beta = 500$)) have controlled the instilled distortions to some extent. In the case of the Gaussian-based regularizer, the improvement is better than that with the inverse quadratic and inverse sub-quadratic-based regularizers.

Since our regularizer is the objective function of the LE method [134], one may

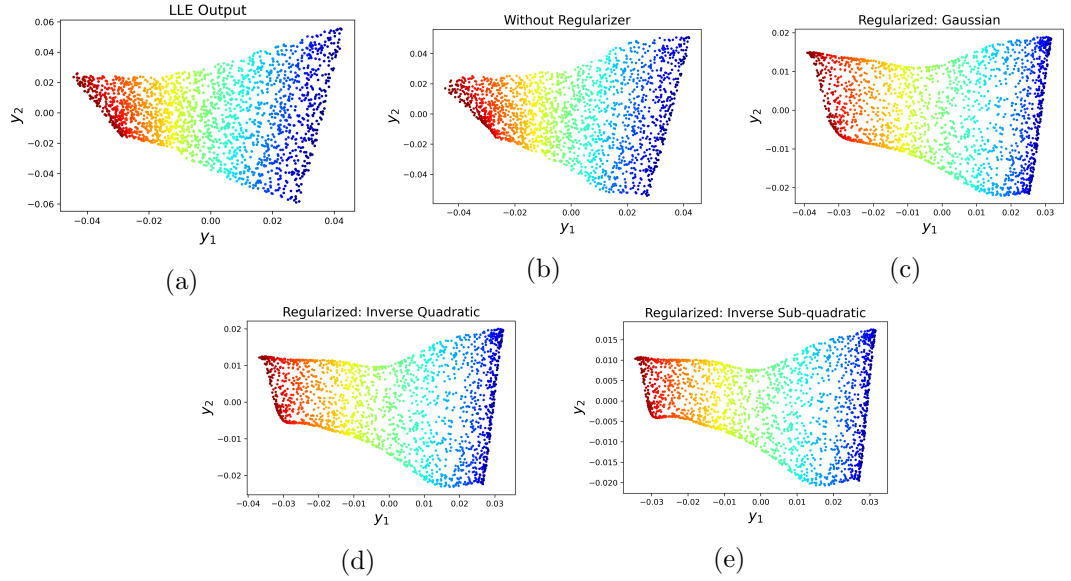


Figure 2.3: Results of polynomial map-based regression on Swiss Roll data set. (a) Regression target (generated by LLE with $\epsilon = 25$). (b) Regression outputs without regularizer. Regularized regression outputs ($k = 10, \beta = 200$): (c) Gaussian-based weights ($\sigma = 0.5$), (d) inverse quadratic-based weights ($\gamma = 10$), (e) inverse sub-quadratic-based weights ($\gamma = 100$).

be interested to see the output of this method. To demonstrate this in Fig. 2.5 we depict the output of the Gaussian kernel-based LE method (implemented using the class ‘manifold.SpectralEmbedding’ of ‘sklearn’ (scikit-learn) package [167] (version 0.19.2) of python) with neighborhood 25 (Fig. 2.5a), 15 (Fig. 2.5b) and 10 (Fig. 2.5c). In these three cases, the method could not produce the desirable unfolding of the Swiss Roll.

We report the results of our experiments on the C Curve data set in Fig. 2.6. Here also the target output displayed in Fig. 2.6a is generated by the LLE method with $\epsilon = 25$. Like Swiss Roll, for the C Curve also, the ideal two-dimensional embedding is a rectangle on the 2D plane. However, the generated target output shown in Fig. 2.6a is close to a triangle as one side of the desired rectangle is squashed to a point. The output without a regularizer, shown in Fig. 2.6b almost replicates the target. The three outputs corresponding to the three regularizers are shown in Figs. 2.6c, 2.6d, and 2.6e. They demonstrate that each of the three regularizers has improved the quality of the embedding by expanding the squashed region. Among the three, the Gaussian-based regularizer has produced the best result followed by inverse sub-quadratic followed by inverse quadratic. For the C Curve data set, we also find that the Gaussian kernel-based LE method with $\epsilon = 10$ produces an undesirable result (Fig. 2.6f).

Figure 2.7 depicts the results of the experiments on the COIL data with the target output as the corresponding LLE output (Fig. 2.7a) with $\epsilon = 3$. From Figs. 2.7a

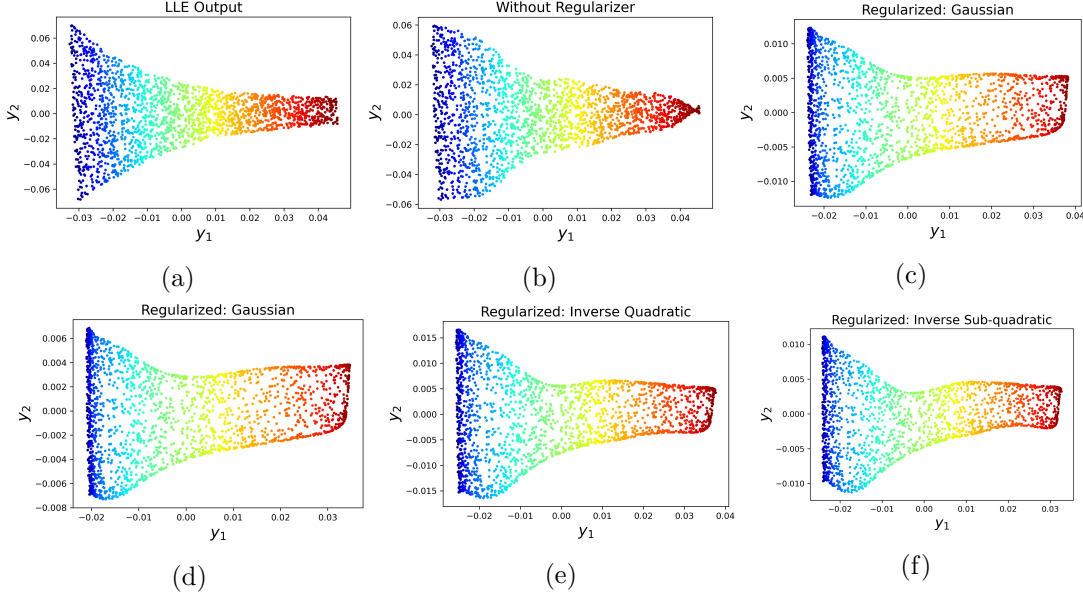


Figure 2.4: Results of polynomial map-based regression on the Swiss Roll data set. (a) Regression target (generated by LLE with $\epsilon = 15$). (b) Regression outputs without regularizer. Regularized regression outputs ($k = 10$): (c) Gaussian-based weights ($\beta = 500, \sigma = 0.5$), (d) Gaussian-based weights ($\beta = 1000, \sigma = 0.5$), (e) inverse quadratic based weights ($\beta = 1000, \gamma = 30$), and (f) inverse sub-quadratic based weights ($\beta = 500, \gamma = 100$).

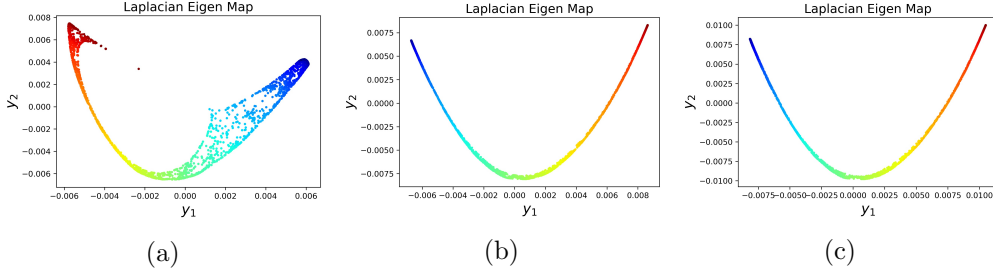


Figure 2.5: Outputs of LE (Gaussian) method on the Swiss Roll data for different choices of k : (a) $k = 25$, (b) $k = 15$, and (c) $k = 10$.

and 2.7b we note that the polynomial mapping without the regularizer reproduces the target output almost as it is. As can be observed in Figs. 2.7c and 2.7d, the Gaussian regularizers have restored the original neighborhood relationship in the output. However, the inverse quadratic (Fig. 2.7e) and inverse sub-quadratic (Fig. 2.7f) regularizers could not do as good a job as by the Gaussian regularizer.

We can also see from Figs. 2.7c and 2.7d that the Gaussian-based regularizer has significantly corrected the local structural distortions. The cluttered points in the three

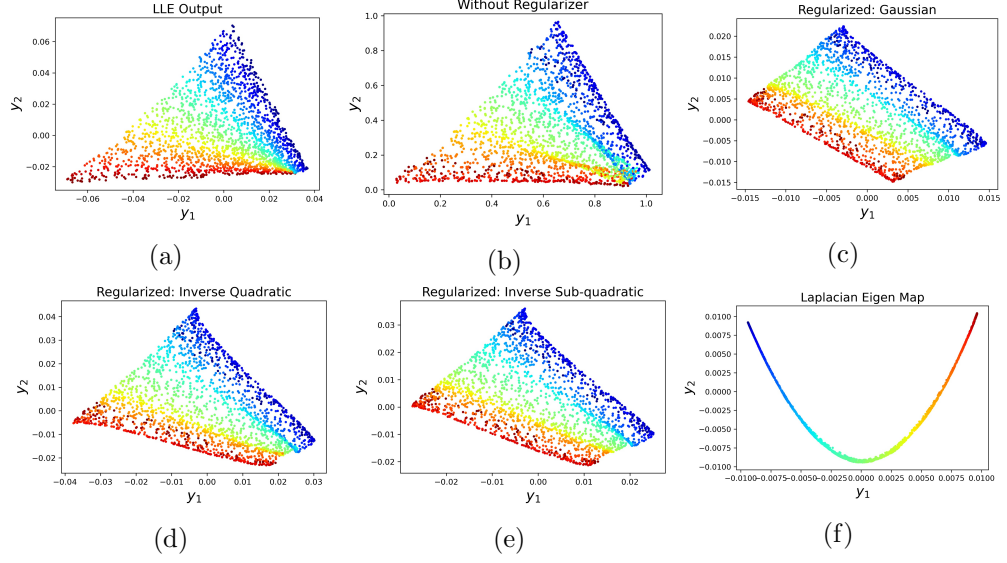


Figure 2.6: Results of polynomial map based regression on C Curve data set. (a) Regression target (generated by LLE with $\epsilon = 25$). (b) Regression outputs without regularizer. Regularized regression outputs ($k = 10$): (c) Gaussian-based weights ($\beta = 100, \sigma = 1$), (d) inverse quadratic-based weights ($\beta = 100, \gamma = 30$), (e) inverse sub-quadratic-based weights ($\beta = 100, \gamma = 50$). (f) Output by LE (Gaussian) with $k = 10$.

spike-like extensions present in the target output are placed continuously following the original topology. The inverse quadratic-based regularizer has also improved the output as revealed by Fig. 2.7e. In the case of the inverse sub-quadratic-based regularizer (Fig. 2.7f) the improvement due to the regularizer is not significant.

For the COIL dataset, the Gaussian kernel-based LE output shown in Fig. 2.8 embeds the data in the desired manner. As mentioned earlier, we have also considered an $L2$ -regularized version of the model. We have explored the effect of the addition of $L2$ -regularizer to our proposed regularized polynomial regression framework. The results of the COIL data set with Gaussian, inverse quadratic, and inverse sub-quadratic-based regularizers in the $L2$ regularized framework are shown in Figs. 2.9, 2.10, and 2.11, respectively. For all the results shown in Figs. 2.9, 2.10, and 2.11, the coefficient of the structure restoring regularizer E_{reg} , i.e., β is set to 200. As defined in (2.2.17), θ is the coefficient of the $L2$ regularization loss. We depict the results in Figs. 2.9, 2.10, 2.11 with $\theta = 0, 1, 10$, and 100, respectively. Even with $\theta = 100$, the output map remains almost the same as that with $\theta = 0$.

Note that, for all three data sets, for all three variants of the regularizers that we have used, the generated embeddings are scaled down by some factor. This is because the regularizer term (2.2.11) does not impose any constraint on the magnitude. The function (2.2.11) tries to put the proximate points in the input space nearby each

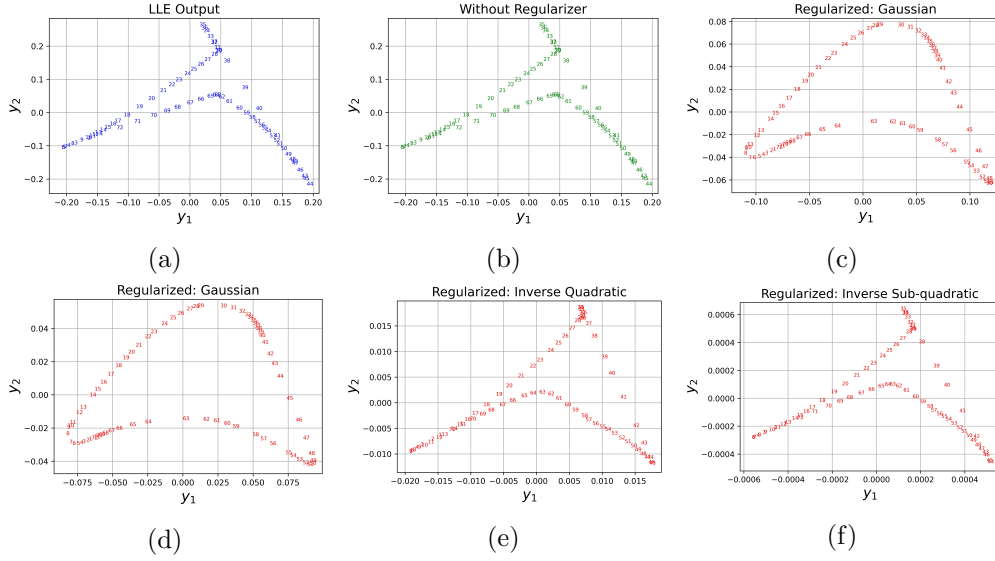


Figure 2.7: Results of polynomial map-based regression on COIL data set. (a) Regression target (generated by LLE with $\epsilon = 3$). (b) Regression outputs without regularizer. Regularized regression outputs ($k = 10$): (c) Gaussian-based weights ($\beta = 100, \sigma = 1.45$), (d) Gaussian-based weights ($\beta = 200, \sigma = 1.45$), (e) inverse quadratic-based weights ($\beta = 200, \gamma = 13$), (f) inverse of sub-quadratic-based weights ($\beta = 200, \gamma = 13$).

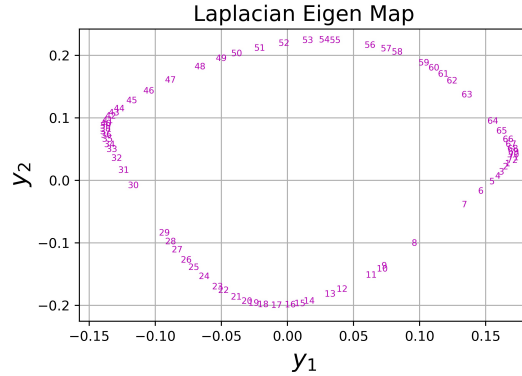


Figure 2.8: LE (Gaussian based) Output on COIL dataset.

other in the projected space, but on the other hand, (2.2.11) is also reduced when the projected points are squeezed into a small hyper-box. This may be avoided using some constraints.

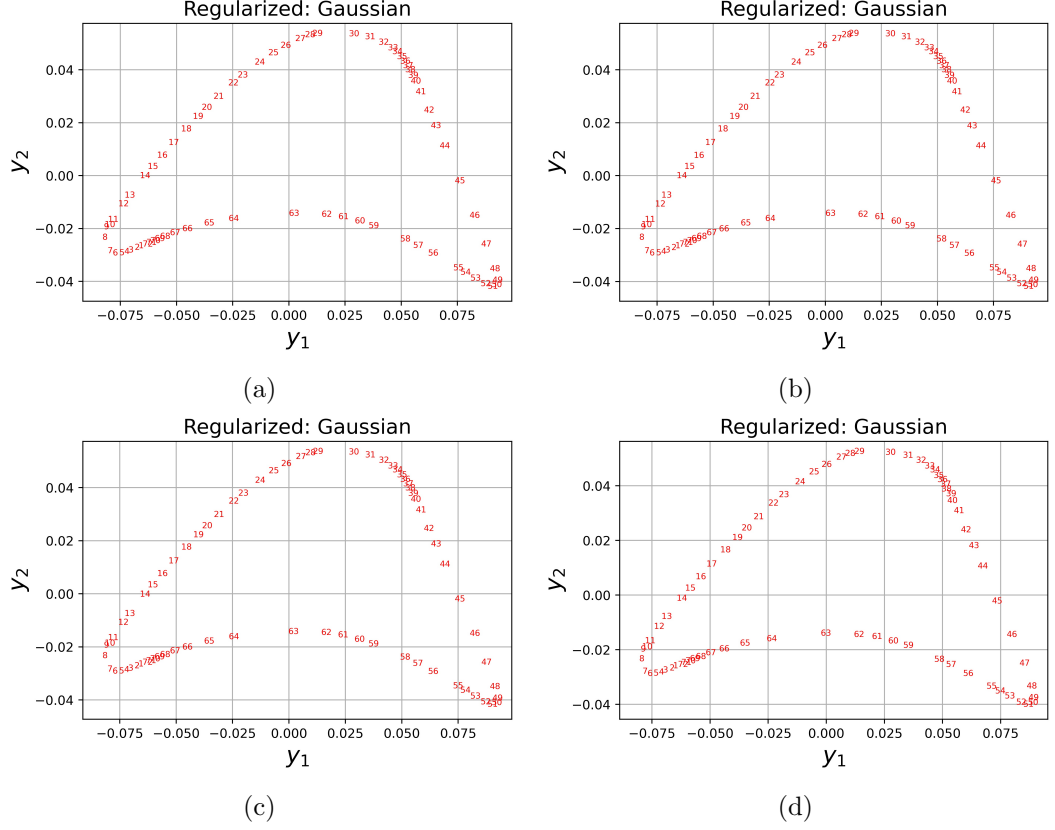


Figure 2.9: Gaussian Based Regularized ($\beta = 200$) Ridge Regression Outputs on COIL data set with $\theta =$ (a) 0, (b) 1, (c) 10, (d) 100.

2.4.1 Results with Targets Generated using t -SNE and ISOMAP

In the previously mentioned experiments, we have considered the manifold learning method LLE for generating targets. However, in place of LLE, we can choose any other manifold learning method. To show the effect of the target generation method on the output, we have considered targets generated by two popular manifold learning methods: t -SNE and ISOMAP on one synthetic data (Swiss Roll) and one real-world data (COIL). As already mentioned, here also we have generated distorted outputs intentionally to check the effectiveness of the proposed method. Figs. 2.12 and 2.13 show results on Swiss Roll with targets generated by t -SNE and ISOMAP respectively.

The target in Fig. 2.12a appears as groups of some clutter points. Here, points are placed very close to some of their neighbors but much far from their other neighbors, disturbing the uniform local neighborhood structure of the original Swiss Roll. The target shown in Fig. 2.13a also shows an undesired non-uniformity. The polynomial regression-based output (without the regularizers) for the t -SNE-based target, shown in Fig. 2.12b, spreads the cluttered projected points to move towards the de-

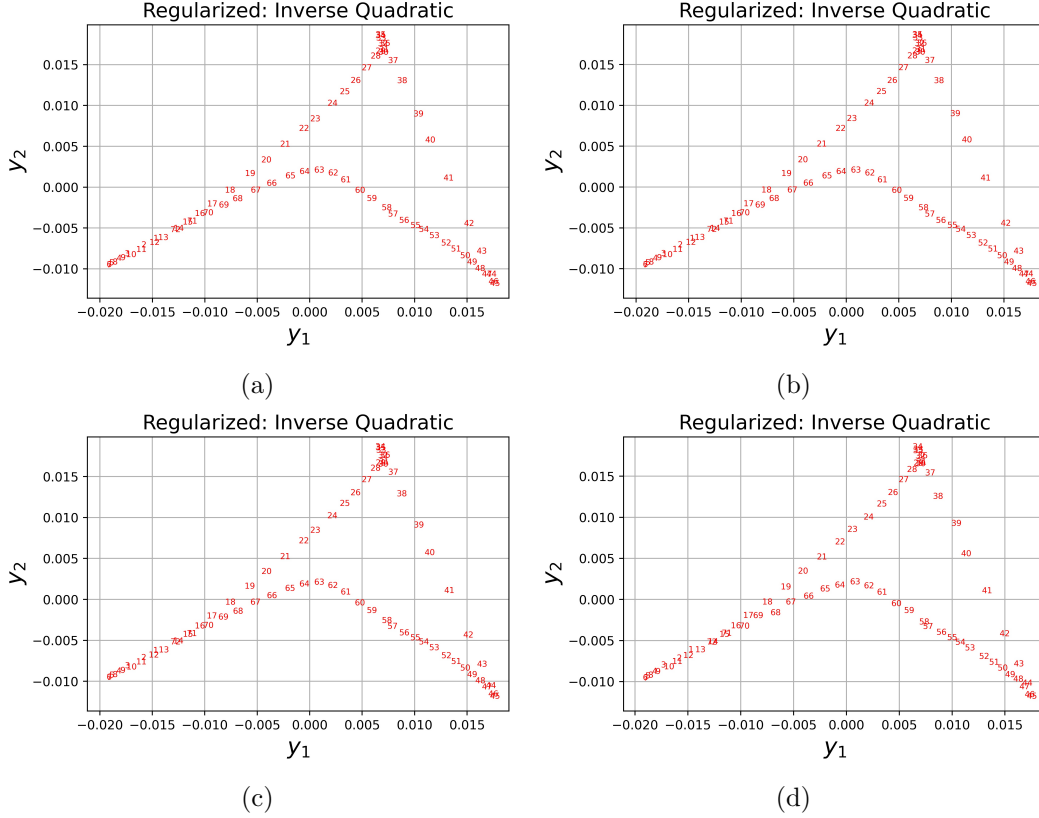


Figure 2.10: Inverse quadratic based regularized ($\beta = 200$) ridge regression outputs on COIL data set with $\theta =$ (a) 0, (b) 1, (c) 10, (d) 100.

sired uniformity to some extent. The polynomial regression-based output (without the regularizers) for the ISOMAP-based target, depicted in Fig.2.13b, does not improve much in terms of smoothness from its respective target. However, the outputs with the regularizers for both t -SNE and ISOMAP-based targets as shown in Figs. 2.12c, 2.12d, 2.12e and Figs. 2.13c, 2.13d, 2.13e are improved in terms of the local neighborhood structures and unfolding of the representation.

We have mentioned earlier that for COIL, the ideal two-dimensional representation of its underlying manifold is a circle. t -SNE with perplexity parameter= 3 generates a 2-D representation of COIL (Fig.2.14a) which deviates from forming a closed circular shape although the neighborhood relations are preserved. The polynomial regression-based outputs for COIL without the regularizers, as shown in Fig.2.14b, mimic the target. On the other hand, the regularized outputs on COIL, displayed in Figs. 2.14c, 2.14d, and 2.14e have restored the neighborhood structures and formed closed circular structures. Fig. 2.15 shows results on COIL using targets generated by ISOMAP. As seen from Fig.2.15a, the target output is far from the ideal 2-D representation which is a circle. Even the regularizers in all three cases (Figs. 2.15c, 2.15d, 2.15e) have failed to

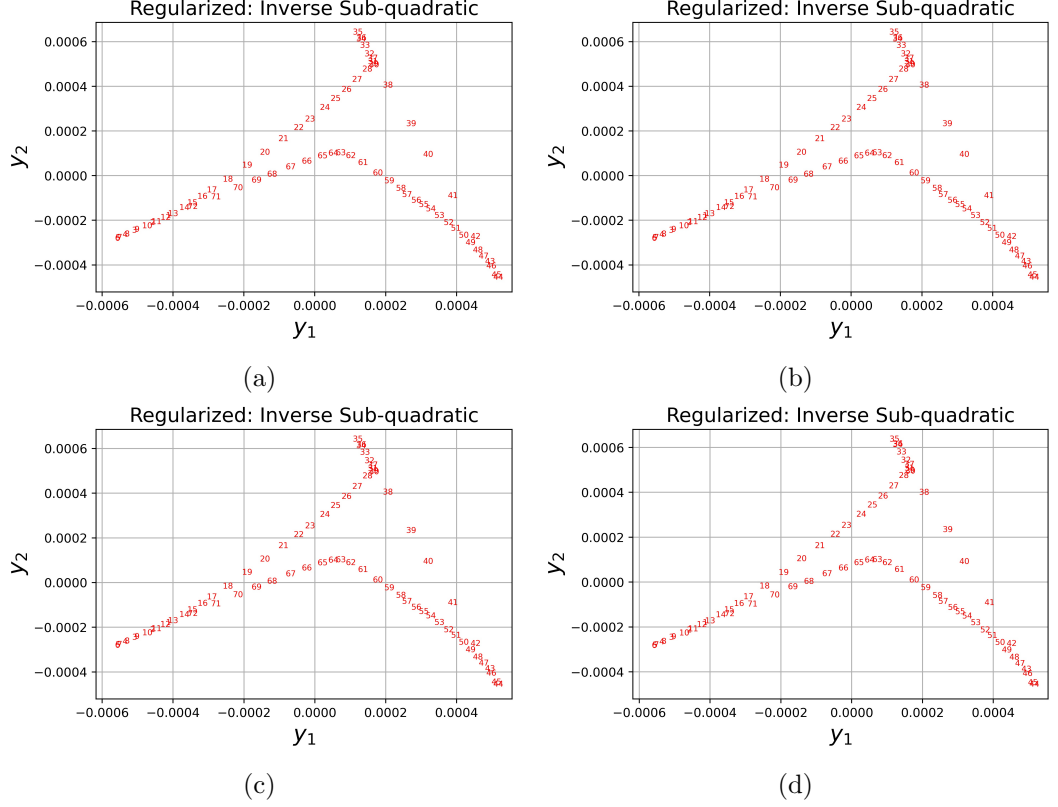


Figure 2.11: Inverse sub-quadratic based regularized ($\beta = 200$) ridge regression outputs on COIL data set with $\theta =$ (a) 0, (b) 1, (c) 10, (d) 100.

restore the actual neighborhood structure. However, the inverse sub-quadratic-based regularizer has reconstructed (see Fig. 2.15e) the target towards the right direction significantly.

2.5 Implementation with Neural Networks

As mentioned in Sec.2.1, the regularized regression framework proposed here is generic in nature. In the previous experiments, the explicit mapping function f was a polynomial function. In this section, we realize f with a neural network, in particular, a multi-layer perceptron (MLP) having a single hidden layer. We choose an MLP considering its universal approximation capability. Here also to learn the network parameters we have minimized (2.2.15), where $\hat{\mathbf{Y}}$ in (2.2.15) is produced by the MLP. To avoid under-fit or over-fit, an optimum number of hidden nodes (n_h) is to be chosen for the MLP. Note that computational complexity of the MLP-based implementation of the proposed method is $O(nPn_h)$ [10]. For Swiss Roll data set, based on the average validation error (the squared error E_{ls} , averaged over five runs on the validation set), we

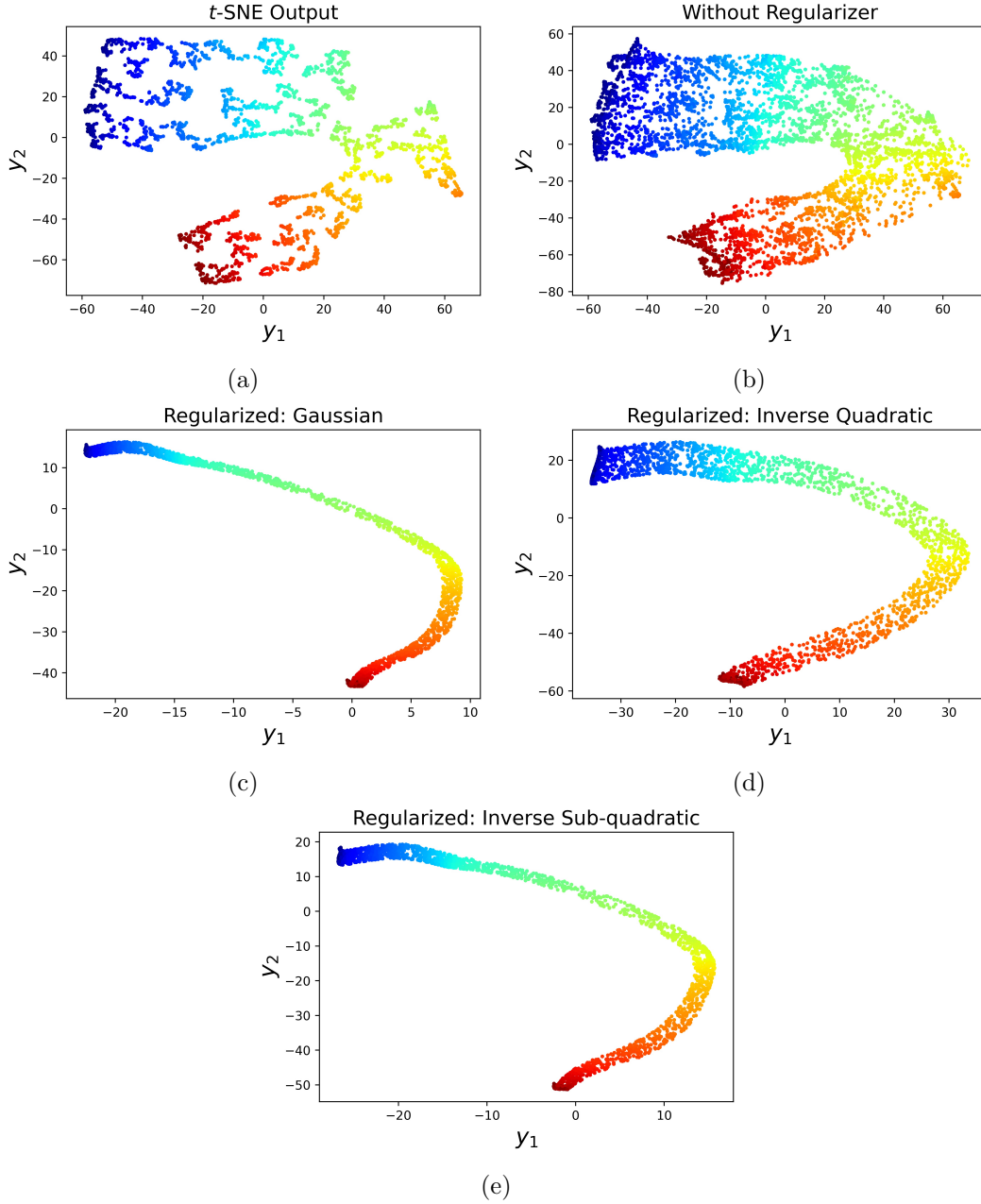


Figure 2.12: Results of polynomial map based regression on Swiss Roll data set with t -SNE as the target. (a) Regression target (generated by t SNE with perplexity=20). (b) Regression outputs without regularizer. Regularized regression outputs ($k = 5, \beta = 1000$) : (c) Gaussian based weights ($\sigma = 1$), (d) inverse quadratic based weights ($\gamma = 10$), (e) inverse sub-quadratic based weights ($\gamma = 10$).

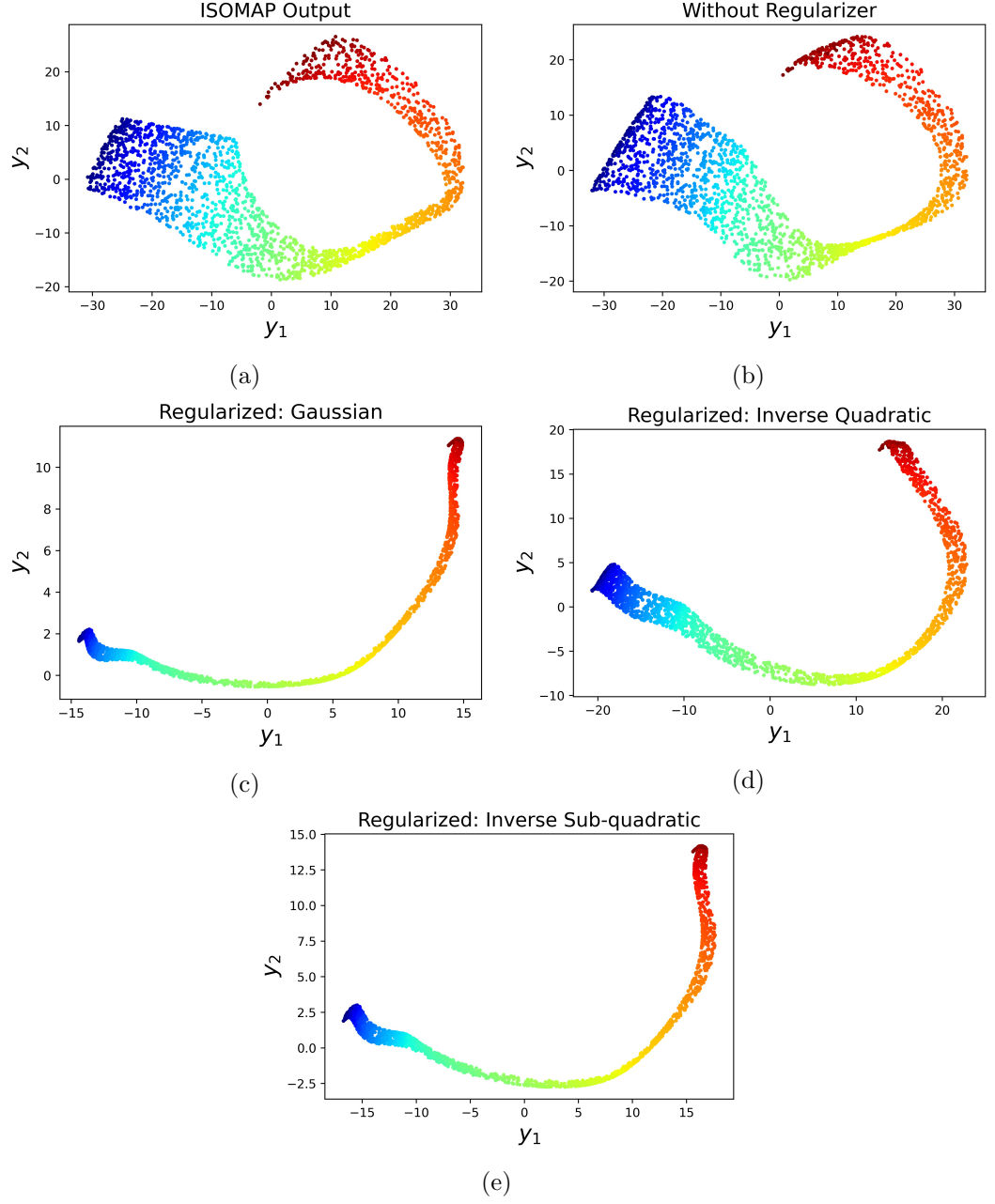


Figure 2.13: Results of polynomial map based regression on Swiss Roll data set with ISOMAP as the target. (a) Regression target (generated by ISOMAP with $k = 25$). (b) Regression outputs without regularizer. Regularized regression outputs ($k = 5, \beta = 1000$): (c) Gaussian based weights ($\sigma = 1$), (d) inverse quadratic based weights ($\gamma = 10$), (e) inverse sub-quadratic based weights ($\gamma = 10$).

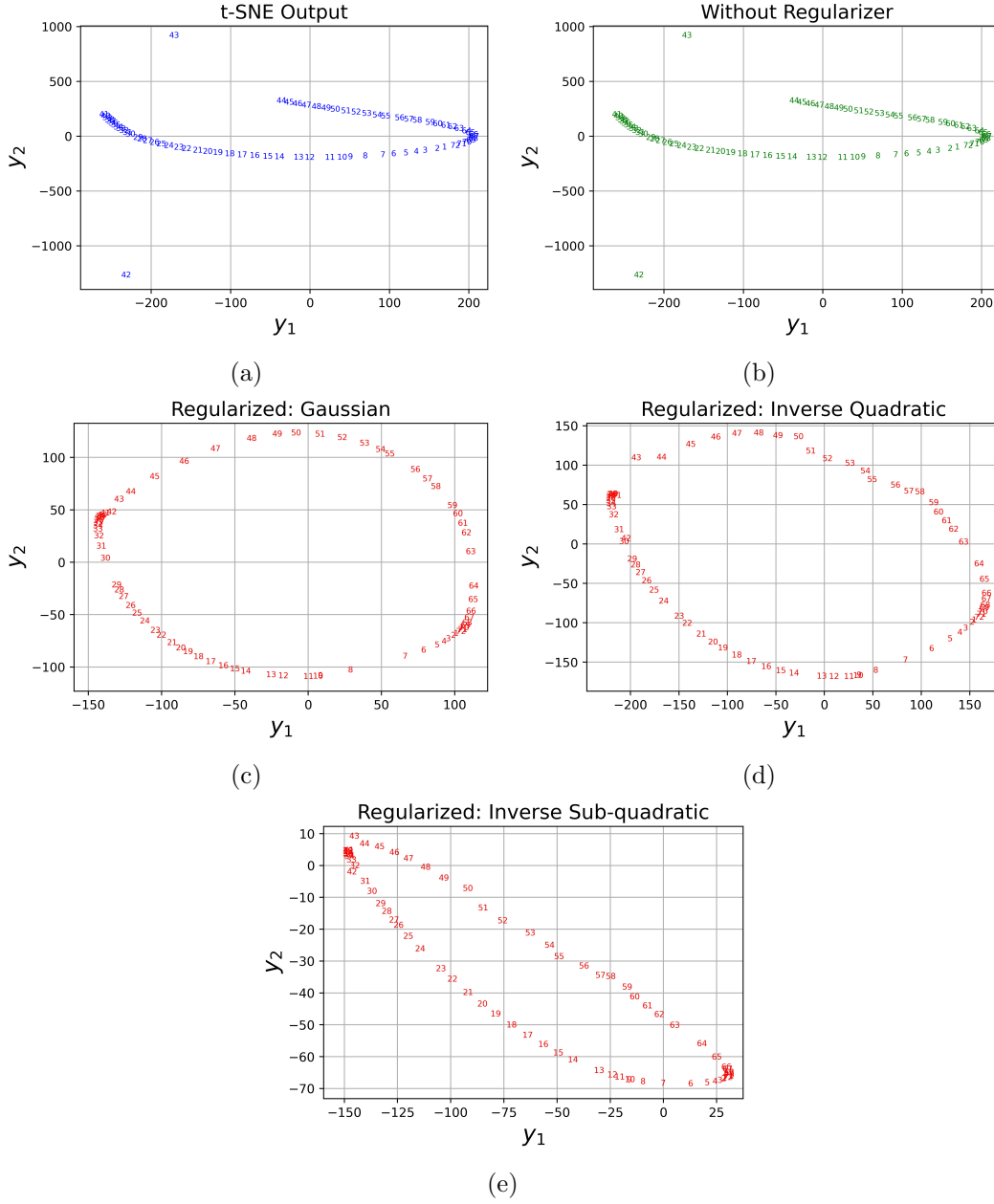


Figure 2.14: Results of polynomial map based regression on COIL data set with tSNE as the target. (a) Regression target (generated by tSNE with perplexity=3). (b) Regression outputs without regularizer. Regularized regression outputs ($k = 5, \beta = 200$) : (c) Gaussian based weights ($\sigma = 1.45$), (d) inverse quadratic based weights ($\gamma = 13$), (e) inverse sub-quadratic based weights ($\gamma = 13$).

have chosen $n_h = 70$. The two synthetic data sets considered, i.e., Swiss Roll and C

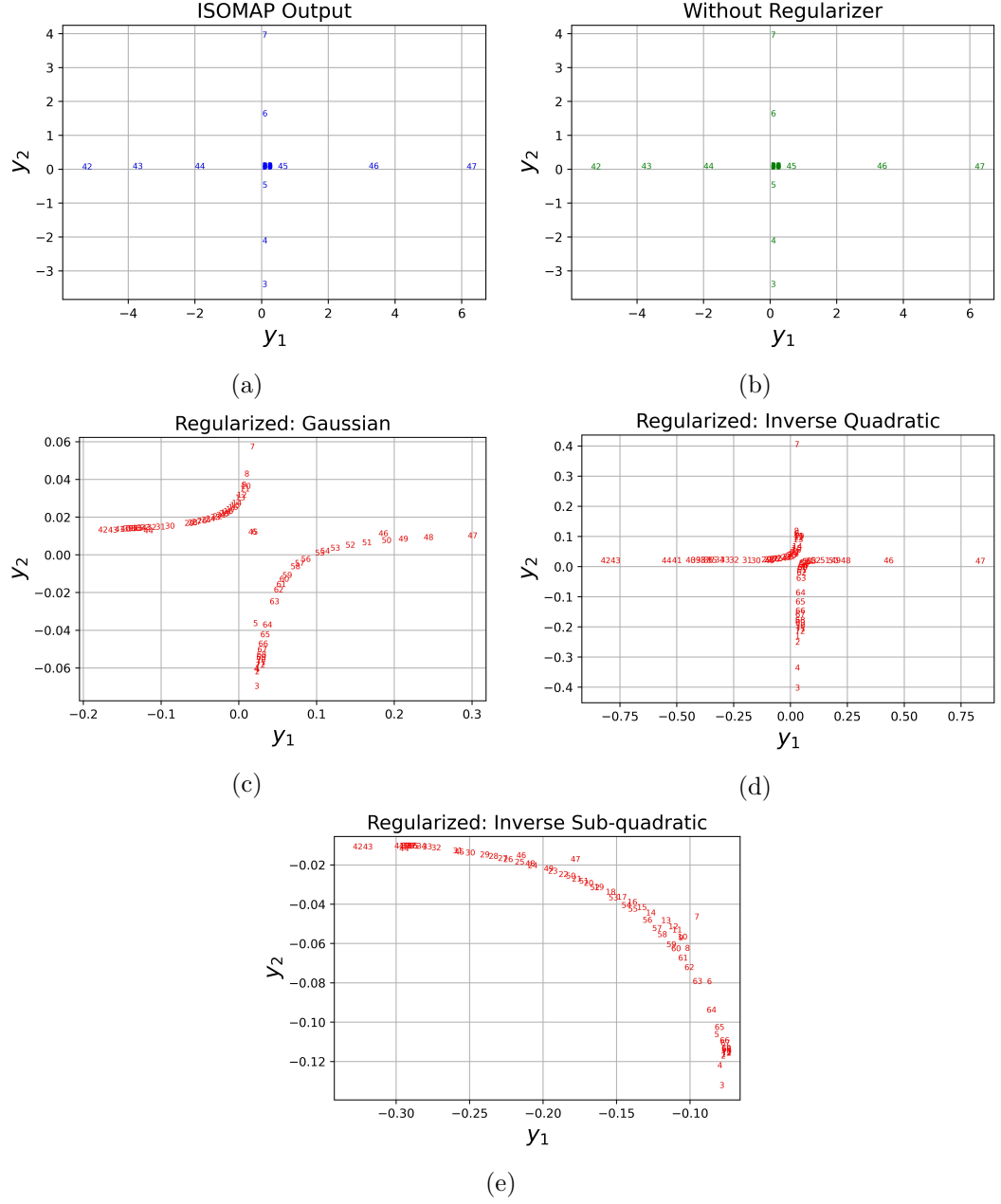


Figure 2.15: Results of polynomial map based regression on COIL data set with ISOMAP as the target. (a) Regression target (generated by ISOMAP with $k = 1$). (b) Regression outputs without regularizer. Regularized regression outputs ($k = 10, \beta = 200$): (c) Gaussian based weights ($\sigma = 1.45$), (d) inverse quadratic based weights ($\gamma = 13$), (e) inverse sub-quadratic based weights ($\gamma = 13$).

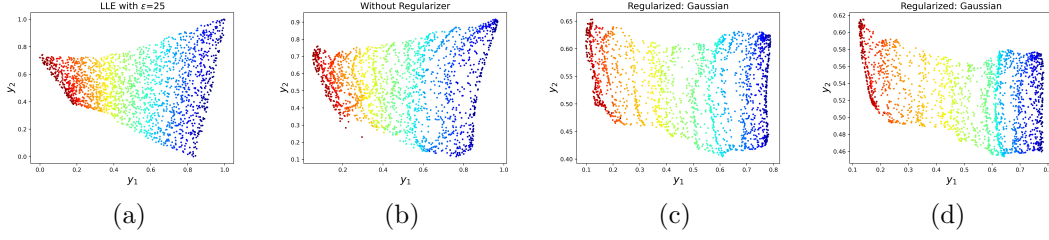


Figure 2.16: Results of neural net-based regression on Swiss Roll data set. (a) Regression target (generated by LLE with $\epsilon = 25$). (b) Regression outputs without regularizer. Regularized regression outputs ($k = 10$): (c) Gaussian-based weights ($\beta = 0.75, \sigma = 0.3$), (d) Gaussian ($\beta = 0.5, \sigma = 0.4$).

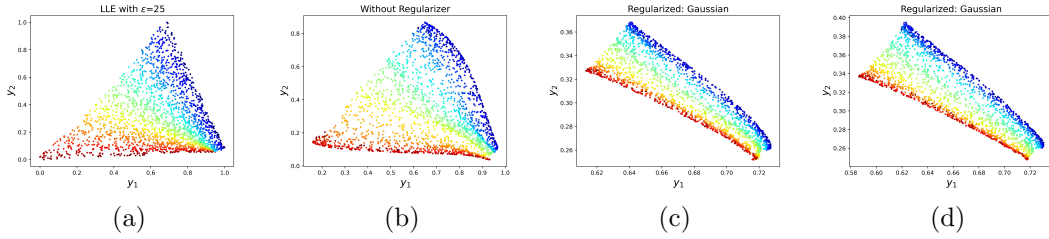


Figure 2.17: Results of neural net-based regression on C Curve data set. (a) Regression target (generated by LLE with $\epsilon = 25$). (b) Regression outputs without regularizer. Regularized regression outputs ($k = 10$): (c) Gaussian-based weights ($\beta = 1, \sigma = 0.2$), (d) Gaussian-based weights ($\beta = 0.25, \sigma = 0.3$).

Curve have the same input dimension, and the same number of instances, and both are topologically equivalent to a 2-D flat sheet. So, for the C Curve also the same value of n_h , i.e., 70 should provide reasonably good results. We note here that use of a validation set may suggest a different optimal choice of n_h , but this is not the primary issue here. For the COIL data set, similar experiments with average validation error suggested $n_h = 100$. Figs. 2.16, 2.17, and 2.18 depict the results produced by the neural net-based implementation corresponding to Swiss Roll, C Curve, and COIL data sets respectively. For the Swiss Roll, the results depicted in Fig. 2.16, correspond to the target output generated by the LLE output with $\epsilon = 25$, which is shown in Fig. 2.16a. Similarly, Fig. 2.17a displays the target for the C Curve data set. Since we have used sigmoidal activation functions for the output layer nodes, in both cases we have normalized the target data to lie in between $[0, 1]$ to facilitate neural net-based learning. Also in both the cases we have explored all the three regularizers defined in (2.2.12), (2.2.13), and (2.2.14) in the regularization loss term (2.2.11). For Swiss Roll, as seen from Figs. 2.16c and 2.16d, the skewness present in the left end of the target (Fig. 2.16a), which is propagated in the unregularized learning of the neural network (Fig. 2.16b), has been corrected by the proposed MLP based method. Also, the target output shows a left to right width variation which should not be there in the

desirable embedding. Figs. 2.16c and 2.16d show almost rectangular shapes recovering from those width-variation distortions. In Fig. 2.17a, the target data of the C Curve, exhibits a significant distortion where one side of the desirable rectangular shape has squeezed almost to a point. As the results in Fig. 2.17c and 2.17d reveal, with the application of Gaussian-based regularizers, the squeezed end of the shape has opened up to some extent. For these data sets, we may also apply the inverse quadratic or inverse sub-quadratic-based regularizers defined in (2.2.13) and (2.2.14) respectively. For the COIL data, the unregularized output shown in Fig. 2.18b has transferred the distortions of the neighborhood relationships of the target output (Fig. 2.18a) and added some more distortions. The outputs obtained by applying the Gaussian, inverse quadratic, and inverse sub-quadratic-based regularizers are shown in Figs. 2.18c, 2.18d, and 2.18e, respectively. In all three cases, the regularized outputs show that the actual neighborhood relationships are restored. Although due to the application of the regularizers, the generated outputs are squeezed (for display purposes, we scale up the figures), in terms of preserving neighborhood relations, they represent the input data far better than the target output data generated by LLE.

2.6 Conclusion

We have proposed a general supervised framework for manifold learning-based dimensionality reduction that can correct distortions in the target data, if any. The method estimates an explicit map for the concerned dimensionality reduction task. This improves the computational complexity in the test phase as it can be directly used to project new test data points. It also helps to analyze the projection system better. The use of supervised methods demands some target data. For supervised manifold learning problems, the target data are generally obtained using an unsupervised manifold learning method. The goal here is to produce a lower-dimensional representation that preserves the local neighborhood structure. It is almost impossible to guarantee that a given unsupervised method of manifold learning will preserve the neighborhood structure for all data sets or even for the same data set the neighborhood structure will be preserved over the entire data. As we have demonstrated that even for a simple three-dimensional synthetic data set the neighborhood preservation may not be uniform over the entire data set - for a part of the data set the algorithm may do an outstanding job while for some other parts, there may be significant distortions from the desired projection. On the other hand, except for synthetic data sets, it is practically very difficult, if not impossible, to assess without post-processing whether the actual topology of the data is preserved in the lower dimension or not. The proposed approach, not only tries to find a faithful mapping from the input dimension to the lower dimension but also tries to remove any distortion of local neighborhood structure injected in the target data. To achieve this we have proposed a set of suitable regularizers that are added to the loss function of the regression model. Due to the regularizer, the method rectifies the structural distortion present in the target data to a great extent. As the regression model, we have considered polynomials and multi-layer perceptron networks.

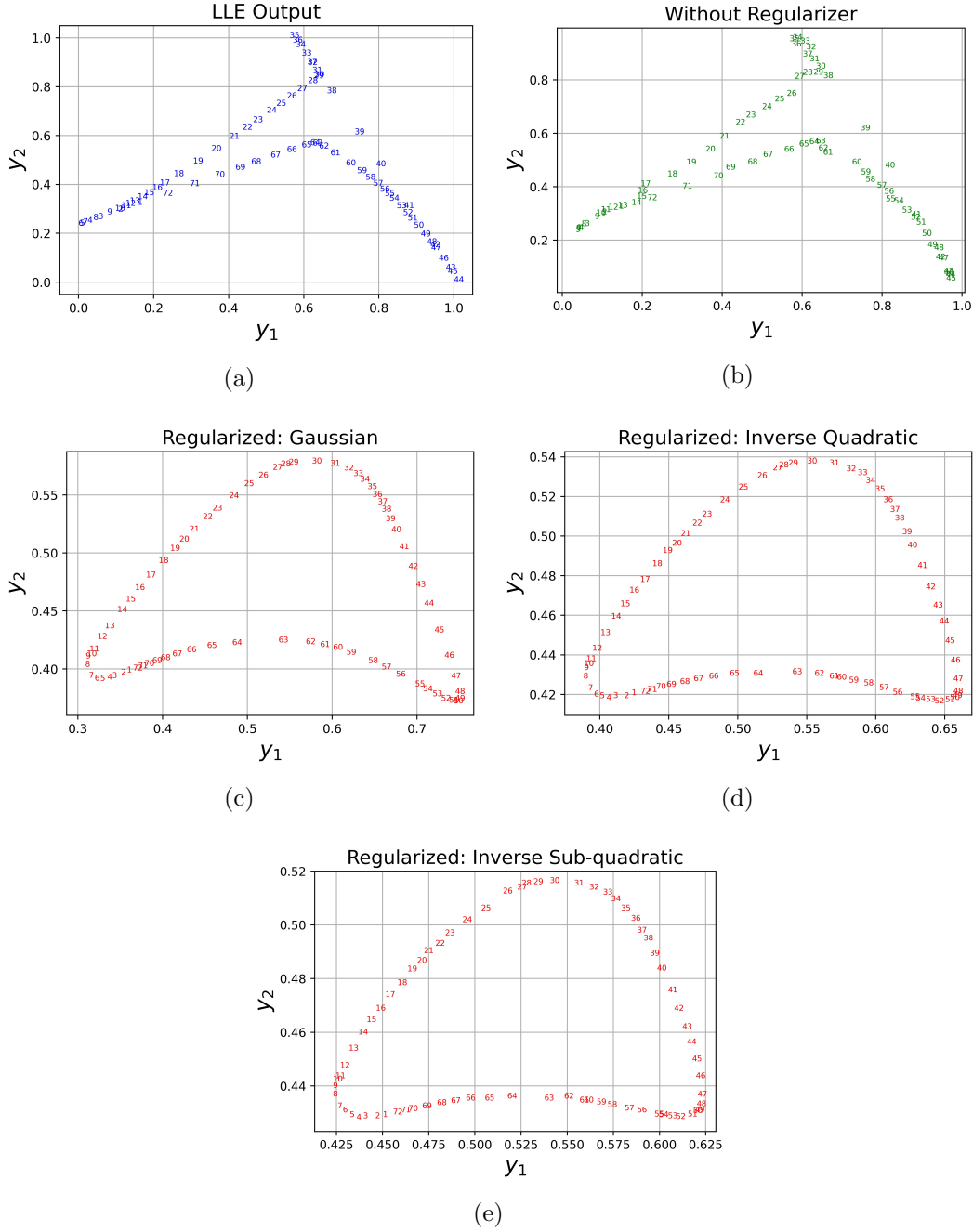


Figure 2.18: Results of neural net-based regression on COIL data set. (a) Regression target (generated by LLE with $\epsilon = 3$). (b) Regression outputs without regularizer. Regularized regression outputs ($k = 5$): (c) Gaussian-based weights ($\beta = 0.5, \sigma = 2$), (d) inverse quadratic-based weights ($\beta = 0.5, \gamma = 0.01$), (e) inverse sub-quadratic-based weights ($\beta = 0.5, \gamma = 0.01$).

We explored three variants of the regularizers. We inspected a limited set of values as the multipliers for the regularizer terms, however, since we are using a supervised learning set up, all these parameters can be chosen using cross validation. The method is validated on synthetic as well as on image data sets considering three different algorithms for the generation of the target data. For image data, typically the input dimension is high, which leads to polynomials with a large number of coefficients even for degrees two or three. Thus, for a faithful estimation of the parameters not only we need a large training set but also it will demand a huge computational overhead. To address this issue, we have exploited the spatial characteristics of image pixels, which reduces the number of learnable parameters significantly. Although we have considered polynomials and multi-layer perceptron networks, the framework is quite general and other tools such as radial basis functions network can also be used.

Chapter 3

Nonlinear dimensionality reduction for data visualization: An unsupervised fuzzy rule-based approach [2, 3]

3.1 Introduction

In Chapter 2, we have discussed the issues associated with the common manifold learning-based data visualization methods for being non-parametric, i.e., not having an explicit function to map higher dimensional points to lower dimensions. We have also pointed out that some of these non-parametric methods have been extended/adapted so that they can handle out-of-sample points. However, several of these extended versions need the training data to deal with out-of-sample points. To overcome the issues, in Chapter 2 we have proposed a method that learns explicit maps through supervision. One weakness of any supervised system is it depends on the target data. We dealt with the dependency of the projection on the quality of the target data in the context of manifold learning in Chapter 2. Our method proposed in Chapter 2 predicts as well as eliminates distortions in the target data to a great extent in the output. However, like any other supervised learning method, the availability of the target data is indispensable, which limits the applicability of such methods. This brings out the importance of learning explicit maps for manifold learning in an unsupervised setting. Another problem with the already proposed out-of-sample extension methods is that they *always* produce some output given *any* input, even when the test input is far from the sampling window of the training data. In literature, these issues are not well addressed in the context of manifold learning or in general non-linear dimensionality reduction-based data visualization methods. Considering these issues, we summarize the *desirable* characteristics of dimensionality reduction-based data visualization methods as follows. These methods **D1:** are unsupervised in nature; **D2:** can preserve the

local geometry of the data in the lower dimension; **D3**: can make a faithful projection of test data points in a *straightforward* manner without requiring the training data; **D4**: can refuse to project test points which are coming from an area far from the “sampling window” of the training data, and **D5**: are interpretable/understandable at least to some extent.

Motivated by these goals we decided to explore fuzzy rule-based systems (FRBSs) for realizing dimensionality reduction-based data visualization models. FRBSs are parametric models that are extensively used in different machine learning tasks such as feature selection [30, 31, 169], control [170, 171], classification [172, 173], and forecasting [174, 175]. An FRBS learns a function from a given set of training points and directly predicts outputs for test points. FRBSs can model a non-linear relationship between input and output. Moreover, they are easy to understand and develop. They provide rules that are “explainable”/“comprehensible” at least to some extent. However, the literature on FRBS is not rich in data visualization applications. Recently, Lughofer and Nikzad-Langerodi [176] proposed an interesting dimensionality reduction technique preserving local neighborhood structure to design fuzzy rule based systems. This method constructs a latent variable space having reduced dimensionality using a local structure preserving variant of the partial least square method. Then the FRBS is designed in that latent variable space to predict the target variables of the time-series data. However, in [176] the fuzzy system is not used to perform the dimensionality reduction task. Apart from the work in [29], which is a *supervised* method, we could not find any investigation employing FRBSs for data visualization applications through dimensionality reduction. In this context, it is worth noting that there are many fuzzy-rough set based methods for feature selection [177, 178]. One can use any feature selection method, including the fuzzy-rough set based methods to select two or three features and use them for visualization. But such feature selection methods neither are intended for nor can do manifold learning where the local neighborhood structure needs to be preserved. Moreover, these methods are also not fuzzy rule-based methods.

We decided to use a Takagi Sugeno (TS) [179] type fuzzy rule based system because the antecedent of a fuzzy rule represents a small hyper-ellipsoidal region in the input space (i.e., captures the local geometry) and its consequent represents a local linear model for the projected data. This is exactly what manifold learning demands- projections preserving the local structure. Moreover, the TS model enjoys the universal approximation property [180]. Identification of fuzzy rule based systems for classification/regression problems demands either experts to provide the rules or the use of labeled data where each instance in the training data should have a target output. However, in our problem, neither we have the output data nor experts who can provide such rules. This makes the problem challenging. Here, our objective is: given an unlabelled high-dimensional data set, $\mathbf{X}$, we want to find a lower dimensional representation, $\mathbf{Y}$, of $\mathbf{X}$ so that the local neighborhood structure of $\mathbf{X}$ is preserved in $\mathbf{Y}$. Due to the absence of target data, traditional approaches to extract fuzzy rule based systems cannot be used. We propose to use a suitable objective function to design the

fuzzy system. The novelty of the scheme lies in the fact that it extracts the desired fuzzy rules for non-linear projection of data without using any target data. Moreover, it provides a *general* framework, which enjoys the following advantages:

1. Most of the unsupervised methods for dimensionality reduction cannot project out-of-sample test data points in a *straightforward* manner, but ours can.
2. This is one of the few methods of nonlinear data projection that has the *reject* option.
3. Our method enjoys some level of interpretability and because of the structure of the fuzzy reasoning, it is not likely to make a poor generalization.
4. The proposed model is found to be very robust to the initial condition.
5. It provides a *general* framework for unsupervised dimensionality reduction. We can use different objective functions to preserve different characteristics of the original data in the projected space. For example, in [2], we have used Sammons' Stress [41] as the objective function.
6. A modified form of the geodesic c-means clustering ensures that the cluster centers are always on the manifold and thereby providing better initial rules.
7. The proposed method performs better than or comparable to several non-fuzzy methods.
8. Finally, to the best of our knowledge, this is the first unsupervised method based on fuzzy rules for nonlinear manifold learning.

3.2 Proposed Method

We want to develop a system to project high dimensional data for visualization. The system should have at least the desirable attributes D1 to D5 listed in Section 3.1. To realize this, we consider an FRBS as the most appropriate modeling framework. Although a Mamdani-Assilian (MA) model could be used, considering greater flexibility we use the TS model. Since this is an *unsupervised* method the main challenge is to define a suitable objective function. The second challenge is how to generate the initial rule base. Typically, we use a clustering algorithm to generate the initial rules. But when the data points lie on a lower dimensional manifold that is embedded in a higher dimension, i.e., the intrinsic dimension of the data is lower than the original dimension of the data, the *c*-means or the fuzzy *c*-means [117, 181] algorithm may not be useful as these algorithms may (usually will) find cluster centers that are significantly away from the manifold. We denote the input dataset by $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n$, an $n \times P$ matrix and the corresponding projected dataset by $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^n$, an $n \times Q$ matrix ($Q < P$). Here, $\mathbf{x}_i$ is a P -dimensional row vector of the form $\mathbf{x}_i = (x_{i1}, x_{i2}, \dots, x_{iP})$ and similarly, $\mathbf{y}_i$ is a Q -dimensional row vector of the form $\mathbf{y}_i = (y_{i1}, y_{i2}, \dots, y_{iQ})$. *Note that only $\mathbf{X}$*

is given and $\mathbf{Y}$ has to be estimated along with other system parameters. Let the j^{th} input variable be x_j and the m^{th} output variable be y_m . Let, there be n_c rules for each output variable. For the TS model, the k^{th} rule for the m^{th} output variable is of the form:

$$\begin{aligned} R_{km} : & \text{ If } x_1 \text{ is } F_{k1} \text{ AND } x_2 \text{ is } F_{k2} \text{ AND } \dots \text{ AND } x_P \text{ is } F_{kP} \\ & \text{ then } y_m^k = a_{km0} + \sum_{j=1}^P a_{kmj} x_j \end{aligned} \quad (3.2.1)$$

where $k = 1, 2, \dots, n_c$; $m = 1, 2, \dots, Q$; F_{kj} is the k^{th} fuzzy set (linguistic value) defined on the j^{th} input feature and a_{kmj} 's are consequent parameters. Let us define the matrix $\mathbf{A} = (A_1, A_2, \dots, A_{n_c})^T$ where, $A_k = (a_{k10}, a_{k11}, \dots, a_{k1P}, a_{k20}, a_{k21}, \dots, a_{k2P}, \dots, a_{kQ0}, a_{kQ1}, \dots, a_{kQP})$.

Let the firing strength of the rule R_{km} for the point $\mathbf{x}_i$ be $\alpha_{k,i}$; $k = 1, 2, \dots, n_c$. Note that, for an antecedent, ' x_1 is F_{k1} AND x_2 is F_{k2} AND $\dots$ AND x_P is F_{kP} ' there are Q consequents, ' $y_m^k = a_{km0} + \sum_{j=1}^P a_{kmj} x_j$ '; $m = 1, 2, \dots, Q$; resulting in Q rules. But the firing strength for each of these Q rules remains the same ($\alpha_{k,i}$). The system output is computed as:

$$y_{im} = \sum_{k=1}^{n_c} (\alpha_{k,i} / \sum_{p=1}^{n_c} \alpha_{p,i}) y_{im}^k \quad (3.2.2a)$$

$$= \sum_{k=1}^{n_c} (\alpha_{k,i} / \sum_{p=1}^{n_c} \alpha_{p,i}) \left(a_{km0} + \sum_{j=1}^P a_{kmj} x_{ij} \right) \quad (3.2.2b)$$

where, $i = 1, 2, \dots, n$; $m = 1, 2, \dots, Q$. Typically, for designing a FRBS, the target outputs for the training data are known and the rule base parameters are estimated by minimizing the square error defined by the target outputs and the estimated outputs. But for us, the target output is *not* available. So we need to define a suitable objective function that can help to learn the manifold of the input data $\mathbf{X}$ as well as the parameters of the rule base. One promising approach would be to preserve the geodesic neighborhood relationship, i.e., the geodesic distance structure of the manifold, on to the projected lower dimension. Note here, in differential geometry, a geodesic is a curve defining the shortest path between a pair of points on a surface or in a Riemannian manifold. The length of the geodesic defines the geodesic distance between the two points. Thus, it is the shortest distance between two points while moving along the surface. The geodesic generalizes the notion of a "straight line" on a Riemannian manifold. Mathematically, the geodesic distance $d(p, q)$ between two points p and q over a Riemannian manifold is defined as $d(p, q) = \inf_{\gamma} L(\gamma)$, where, γ s are the piece-wise smooth curve segments from p to q and $L(\gamma)$ represents length of the curve γ [182]. Let $gd_{ii'}^{\mathbf{X}}$ be the geodesic distance between $\mathbf{x}_i$ and $\mathbf{x}_{i'}$, $\mathbf{x}_i, \mathbf{x}_{i'} \in \mathbb{R}^P$ and $ed_{ii'}^{\mathbf{Y}}$ be the Euclidean distance between $\mathbf{y}_i$ and $\mathbf{y}_{i'}$, $\mathbf{y}_i, \mathbf{y}_{i'} \in \mathbb{R}^Q$; $\mathbf{y}_i, \mathbf{y}_{i'}$ are the projections corresponding to $\mathbf{x}_i$ and $\mathbf{x}_{i'}$, respectively. A good objective function to estimate $\mathbf{y}_i$ s is (3.2.3)

$$E = \sum_{i=1}^{n-1} \sum_{i'=i+1}^n (gd_{ii'}^{\mathbf{X}} - ed_{ii'}^{\mathbf{Y}})^2 / gd_{ii'}^{\mathbf{X}}. \quad (3.2.3)$$

The objective (3.2.3) was introduced by Yang in [183]. Note that (3.2.3) is similar to Sammon's stress function which uses Euclidean distance in both high and low dimensional spaces. In place of (3.2.3), we can use other functions also. Next, we address the issue of identification of the rule base.

3.2.1 Initial Rule Extraction

When both input and output data are provided there are many ways of generating an initial rule base and its refinement [29, 30, 180, 184–186]. Some of the popular methods use evolutionary algorithms [186] or clustering [29, 30, 180] of input-output data. Here we do not have the output data. So we cluster the input data into a predefined number of clusters and translate each cluster into a rule (to be precise, to a rule antecedent).

The proposed method being a fuzzy rule-based scheme, the obvious choice seems to be the fuzzy c -means (FCM) clustering algorithm. But we did not do so for two reasons: First, for the FCM, the cluster centers may not fall on the data manifold. Since our idea is to use a cluster centroid to model a set of points in the neighborhood of the centroid, it is better to have the centroid on the data manifold. Second, the geodesic distance is not defined in terms of an inner-product induced norm and hence the FCM convergence theory does not hold for the geodesic distance-based FCM. So we use a slightly modified c -means called Geodesic c -Means (GCM) algorithm. Here we cluster the input, aiming to obtain some representative points on the input manifold. To extract information regarding the input manifold structure, we use geodesic distance as the dissimilarity measure for clustering. Like the conventional c -means, data points are assigned to a cluster using the minimum distance (here minimum geodesic distance) criterion and the cluster centroids are computed as the mean vector of the points in a cluster. Since such centroids may not lie on the manifold, we use an extra step. For each computed centroid, we find the input data point closest to the centroid and use that data point as a cluster centroid. We note here that the use of geodesic distance in the c -means clustering algorithm has been investigated in other studies [187–189] but their approaches are different from ours. In [187] the authors introduced a class of geodesic distance which took into account local density information and employed that geodesic distance in the c -means clustering algorithm. The study in [188] incorporated geodesic distance in a soft kernel c -means algorithm. The authors in [189] integrated the geodesic distance measure into the fuzzy c -means clustering algorithm. The algorithmic description of our clustering algorithm (GCM) is given in Algorithm 1.

To estimate the geodesic distance we have followed an approach similar to that in ISOMAP [42]. First, we construct a neighborhood graph of the input data points based on the Euclidean distance. For approximating geodesic distance, every edge is assigned a weight/cost which is basically the Euclidean distance between the pair of points on which the edge is incident. For any point, the geodesic distances to its neighboring points are approximated by the Euclidean distance. The geodesic distance between two points that are not neighbors, i.e., not connected by an edge is approximated by computing the length of the shortest path on the neighborhood graph. We compute the shortest path distance using the Floyd-Warshall algorithm [190].

The k^{th} cluster centroid, $\mathbf{v}_k = (v_{k1}, v_{k2}, \dots, v_{kP})$, obtained by GCM is translated into the antecedent of the k^{th} rule, R_{km} as follows.

$$\begin{aligned} &\text{If } x_1 \text{ is "close to } v_{k1}\text{" AND } x_2 \text{ is "close to } v_{k2}\text{"} \\ &\text{AND } \dots x_P \text{ is "close to } v_{kP}\text{"}. \end{aligned} \tag{3.2.4}$$

Algorithm 1: Geodesic c -means algorithm (GCM)

```

1 Determine Geodesic distances  $\mathbf{gd}$  between pairs of points in  $\mathbf{X}$ .
2 Let the number of clusters be  $n_c$ . Initialize, cluster centroids  $\{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{n_c}\}$ 
   with  $n_c$  random input points i.e,  $n_c$  random points from  $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ 
3 Initialize sets  $S_k = \phi$ ;  $k = 1, 2, \dots, n_c$ .
4 while  $iteration < MaximumIteration$  do
5     for  $i = 1$  to  $n$  do
6          $k^* = \arg_k \min \mathbf{gd}(\mathbf{x}_i, \mathbf{v}_k)$ 
7          $S_{k^*} = S_{k^*} \cup \mathbf{x}_i$ 
8     end
9     for  $k = 1$  to  $n_c$  do
10         $\hat{\mathbf{v}}_k = 1/|S_k| \sum_{\mathbf{x}_i \in S_k} \mathbf{x}_i$ 
11         $i^* = \arg_i \min \mathbf{ed}(\hat{\mathbf{v}}_k, \mathbf{x}_i)$ 
12         $\mathbf{v}_k = \mathbf{x}_{i^*}$ 
13    end
14 end
15 return  $\{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{n_c}\}$ 

```

The fuzzy set F_{kj} in (3.2.1) is defined and interpreted as “close to v_{kj} ”; $j = 1, 2, \dots, P$. Thus the k^{th} rule given in (3.2.1) becomes

$$R_{km} : \text{If } x_1 \text{ is “close to } v_{k1}\text{” AND } x_2 \text{ is “close to } v_{k2}\text{” AND} \\ \dots \text{ AND } x_P \text{ is “close to } v_{kP}\text{” then } y_m^k = \sum_{j=0}^P a_{kmj} x_j \quad (3.2.5)$$

where, $x_0 = 1$, $k = 1, 2, \dots, n_c$; $m = 1, 2, \dots, Q$. We model ‘ x_j is “close to v_{kj} ”’ using a Gaussian membership function with the center at v_{kj} . To exploit the differentiability property, we use Gaussian membership functions. This also makes the antecedent of the k^{th} fuzzy rule to model a hyper-ellipsoidal volume centering at $\mathbf{v}_k$ in the input space. For the i^{th} point, the membership to the set “ x_j is close to v_{kj} ” is computed as

$$\mu_{kj,i} = \exp \left\{ -(x_{ij} - v_{kj})^2 / 2\sigma_{kj}^2 \right\}. \quad (3.2.6)$$

Here, σ_{kj} is the spread of the Gaussian membership function of the fuzzy set F_{kj} centered at v_{kj} . Although any T -norm [191] can be used, we use the product T -norm to aggregate $\mu_{kj,i}$ ’s to obtain the firing strength $\alpha_{k,i}$.

$$\alpha_{k,i} = \prod_{j=1}^P \mu_{kj,i}. \quad (3.2.7)$$

We denote the matrix of cluster centroids as $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{n_c})^T = [v_{kj}]_{n_c \times P}$. Similarly, the matrix of spreads $\mathbf{\Sigma} = (\sigma_1, \sigma_2, \dots, \sigma_{n_c})^T = [\sigma_{kj}]_{n_c \times P}$. An initial estimate of σ_{kj} can be taken as the standard deviation of j^{th} feature in the k^{th} cluster or can be initialized using some other method. The set of consequent parameters A is initialized randomly. Note that these are just the initial choices that will be refined during the training.

3.2.2 The Objective Function and its Optimization

To learn the lower dimensional representations $\mathbf{Y}$ of the given input $\mathbf{X}$, the error function defined in (3.2.3) is minimized with respect to the rule base parameters $\mathbf{V}$, $\mathbf{\Sigma}$, and $\mathbf{A}$. Using (3.2.6) and (3.2.7) in (3.2.2b) we obtain the following relation.

$$y_{im} = \sum_{k=1}^{n_c} \frac{\left(\prod_{j=1}^P \exp \left\{ -\frac{(x_{ij}-v_{kj})^2}{2\sigma_{kj}^2} \right\} \right) \left(\sum_{j=0}^P a_{kmj} x_{ij} \right)}{\sum_{p=1}^{n_c} \prod_{j=1}^P \exp \left\{ -\frac{(x_{ij}-v_{pj})^2}{2\sigma_{pj}^2} \right\}} \quad (3.2.8)$$

Here, $x_{i0} = 1$ and $\mathbf{y}_i = (y_{i1}, y_{i2}, \dots, y_{iQ})$. Hence $\mathbf{y}_i$ is dependent on v_{kj}, σ_{kj} , and $a_{kmj}; \forall k = 1$ to $n_c, j = 1$ to P and $m = 1$ to Q which form $\mathbf{V}, \mathbf{\Sigma}$ and $\mathbf{A}$, respectively. It is evident that we can write the lower dimensional vectors $\mathbf{y}_i$ s as a function of the rule base parameters $\mathbf{V}, \mathbf{\Sigma}$, and $\mathbf{A}$.

$$\mathbf{y}_i = f(\mathbf{V}, \mathbf{\Sigma}, \mathbf{A}; \mathbf{x}_i) \quad (3.2.9)$$

Replacing $ed_{ii'}^{\mathbf{Y}}$ in (3.2.3) as $\|\mathbf{y}_i - \mathbf{y}_{i'}\|$ and then using (3.2.9) we get

$$\begin{aligned} E &= \sum_{i=1}^{n-1} \sum_{i'=i+1}^n \left(gd_{ii'}^{\mathbf{X}} - \|\mathbf{y}_i - \mathbf{y}_{i'}\| \right)^2 / gd_{ii'}^{\mathbf{X}} \\ &= \sum_{i=1}^{n-1} \sum_{i'=i+1}^n \left(gd_{ii'}^{\mathbf{X}} - \|f(\mathbf{V}, \mathbf{\Sigma}, \mathbf{A}; \mathbf{x}_i) - f(\mathbf{V}, \mathbf{\Sigma}, \mathbf{A}; \mathbf{x}_{i'})\| \right)^2 / gd_{ii'}^{\mathbf{X}}. \end{aligned} \quad (3.2.10)$$

The optimum values of $\mathbf{V}, \mathbf{\Sigma}, \mathbf{A}$ are obtained as

$$(\mathbf{V}_{opt}, \mathbf{\Sigma}_{opt}, \mathbf{A}_{opt}) = \arg \min_{\mathbf{V}, \mathbf{\Sigma}, \mathbf{A}} (E). \quad (3.2.11)$$

To minimize the error function E in (3.2.11), a momentum based gradient descent approach [128] is used here. The overall process is summarized in Algorithm 2.

3.2.3 Scalability Analysis

Scalability is a desirable property of any algorithm. The error function (3.2.3) of our method considers all the inter-point distances, i.e., a total of $n(n-1)/2$ distances. During the training phase, to update the rule parameters, in each iteration, $n(n-1)/2$ values need to be computed. So for t iterations, the computational complexity of the proposed method is $O(n^2t)$ for a fixed input dimension. In Table 3.1, we have compared the computational complexity of the proposed method during the training phase, with the computational complexities of six popular dimensionality reduction methods for data visualization. Note that, LLE and ISOMAP perform convex optimization whereas the other methods including ours perform non-convex optimization with iterative steps. Similar to our method, Sammon's projection and t -SNE also involve inter-point distances in their objective functions. Hence, they have the same complexity as that of the proposed method. The computational complexity of UMAP is bounded by the approximate nearest neighbor search which is empirically equal to $O(n^{1.14})$ [47]. Considering a total of t iterations, computational complexity would be $O(n^{1.14}t)$. Since UMAP

Algorithm 2: Unsupervised Fuzzy Rule-based Dimensionality Reduction

-
- 1 Determine Geodesic distance $gd^{\mathbf{X}}$ between every pair of input data points in $\mathbf{X}$.
 - 2 Cluster the input data $\mathbf{X}$ into n_c clusters using the Geodesic distance-based c -means clustering algorithm (Algorithm 1). Let the obtained cluster centroids be $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{n_c}$; $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{n_c})^T$.
 - 3 Translate each cluster ($\mathbf{v}_k$) into a rule as explained in (3.2.5) and (3.2.6). Initialize spreads of the Gaussian membership functions (described in (3.2.6)) by Σ . Here we use a predefined initial value for $\sigma_{kj} \forall k, j$.
 - 4 Let the gradients of the objective function E in (3.2.3) with respect to $\mathbf{V}, \Sigma$ and $\mathbf{A}$ be $\nabla_{\mathbf{V}}E, \nabla_{\Sigma}E$ and $\nabla_{\mathbf{A}}E$, respectively.
 - 5 Initialize the consequent parameters $\mathbf{A}$ randomly and set the iteration count $t = 1$. $\Delta\mathbf{V}^t, \Delta\Sigma^t$, and $\Delta\mathbf{A}^t$ are updates at time step t corresponding to centers, spreads, and consequent parameters. $\Delta\mathbf{V}^0, \Delta\Sigma^0$, and $\Delta\mathbf{A}^0$ are set to zeros.
 - 6 **while** $t < \text{MaximumIteration}$ **do**
 - 7 Calculate $\Delta\mathbf{V}^t, \Delta\Sigma^t, \Delta\mathbf{A}^t$ and update $\mathbf{V}^t, \Sigma^t$ and $\mathbf{A}^t$ as follows:

$$\Delta\mathbf{V}^t = -\nabla_{\mathbf{V}}E + \alpha\Delta\mathbf{V}^{t-1}; \Delta\Sigma^t = -\nabla_{\Sigma}E + \alpha\Delta\Sigma^{t-1}; \Delta\mathbf{A}^t = -\nabla_{\mathbf{A}}E + \alpha\Delta\mathbf{A}^{t-1} \quad (3.2.12)$$
 - 8 $\mathbf{V}^t = \mathbf{V}^{t-1} + \eta\Delta\mathbf{V}^t; \Sigma^t = \Sigma^{t-1} + \eta\Delta\Sigma^t; \mathbf{A}^t = \mathbf{A}^{t-1} + \eta\Delta\mathbf{A}^t \quad (3.2.13)$
 - 9 $\alpha > 0$ and $\eta > 0$ are momentum and learning coefficient, respectively.
 - 10 **end**
 - 11 **return** outputs $\mathbf{Y}$ and fuzzy rule base parameters $\mathbf{V}, \Sigma, \mathbf{A}$.
-

involves the estimation of the nearest neighbors, it needs to store the inter-point distances. Hence, its space complexity is $O(n^2)$. We emphasize that being an explicit mapping method, during the test phase the proposed method has a computational complexity of $O(1)$. In [28] authors did not discuss the complexity of training diffusion net. Diffusion net involves computation of diffusion map (complexity $O(n^3)$ [10]) and training an autoencoder (complexity $O(nwt)$, where w is the number of weights in the autoencoder [10]). Except for Diffusion net, the other compared methods in their original form cannot predict out-of-sample points. They exhibit the same computational complexity in the test phase as in the training phase. However, their extensions have lower complexities for projecting test data. In terms of space requirement, like most of the other methods, our method also exhibits a complexity of $O(n^2)$. It is also important to understand the dependency of an algorithm with the dimension of the input feature vectors (P). In our method, the number of rule parameters varies linearly with the input dimension. So both the computational and space complexity scale up linearly with the number of input features.

Table 3.1: Computational complexity of various methods.

Method	Complexity	
	Computational	Space
Proposed Method	$O(n^2t)$	$O(n^2)$
Sammon's Projection [10]	$O(n^2t)$	$O(n^2)$
ISOMAP [10]	$O(n^3)$	$O(n^2)$
LLE [10]	$O(pn^2)$	$O(pn^2)$
t-SNE [45]	$O(n^2t)$	$O(n^2)$
UMAP [47]	$O(n^{1.14}t)$	$O(n^2)$
Diffusion Net [28]	$O(n^3 + nwt)$	$O(n^2)$

Here n = size of the training set, t = number of iterations, p = ratio of nonzero to total number of elements in the sparse matrix related to LLE, and w = number of weights in the network

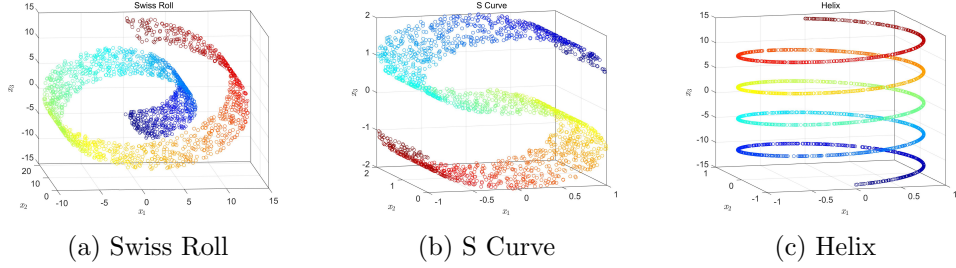


Figure 3.1: Data sets

3.3 Experimentation and Results

3.3.1 Data Set Description

For our experiments, we consider three synthetic data sets and three real-world data sets. We choose these data sets as they are commonly used in manifold learning studies [41, 43, 45, 83]. The synthetic data sets are: Swiss Roll, S Curve, and Helix as shown in Figs. 3.1a, 3.1b, 3.1c, respectively. All of these are three-dimensional data but contained completely within a two-dimensional space. Each of these synthetic data sets consists of 2000 points. The first two data sets (Swiss Roll was already used in the previous chapter) are generated using the 'scikit-learn' package [167] (version 0.19.2) of python. Helix data set is generated by the equations: $x_3 = t/\sqrt{2}$; $x_1 = \cos x_3$; and $x_2 = \sin x_3$. The variable t is varied from -20 to 20 with steps of 0.02 . The real-world data sets considered are Frey face [192], COIL [168], and USPS handwritten digits [192]. Frey face consists of 1965 gray scale images each of size 20×28 pixels, i.e., the input dimension is 560. These are images of the face of a single person with different face orientations and facial emotions. The COIL data set was used in Chapter 2. Here also,

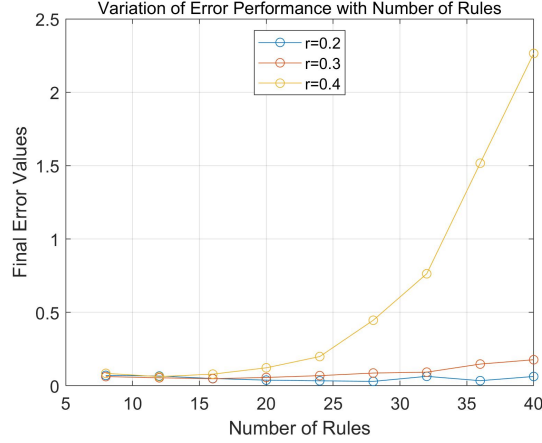


Figure 3.2: Variation of average final error values with number of rules and initial spreads.

we have considered the first object of the COIL data set. Each image is of resolution 32×32 . Thus, the input dimension is 1024, and the number of instances = 72. USPS handwritten digits data set contains 1100 images for each of the handwritten digits: 0 - 9. Each image is of size 16×16 , resulting in inputs of dimension 256. We have considered the images of the digit 0 only.

3.3.2 Experiment Settings

We compare the results obtained by the proposed method on the above-mentioned data sets with six other data visualization methods: Sammon’s projection, ISOMAP, LLE, t -SNE, UMAP, and diffusion net. The synthetic data sets are used directly without any preprocessing. For image data sets, pixel values are divided by 255 to have their values in $[0, 1]$. We compare the predictability of the proposed method with that of the out-of-sample extension of ISOMAP and LLE [148], autoencoder [24], and simplified neighborhood preserving polynomial embedding (SNPPE) [83]. For this, we use the Swiss roll and the Frey face data sets. Following [83] we choose the number of nearest neighbors, $\epsilon = 1\%$ of the training samples (rounded to the nearest integer). For the first object of the COIL data set (we shall refer to it as COIL), the number of instances is 72. So we use $\epsilon = 5$ instead of $\epsilon = 1$ to construct a reasonable graph. To choose the number of fuzzy rules, n_c , and the spreads of the Gaussian memberships σ s, we perform the following experiment. We run our method on the Swiss Roll dataset with the number of rules (n_c) = 8, 12, 16 and so on till 40 and repeat this experiment for three different spread initializations of the Gaussian membership functions: $\sigma = r \times \text{feature-specific range}$, for $r = 0.2, 0.3$, and 0.4. For each combination of n_c and σ five runs are performed and the average of the final error values (Eqn. (3.2.3)) over the five runs is computed. Fig. 3.2 shows the plot of average final error values vs. the number of rules. We observe that for all three initialization of spread, when the number of rules=20, we

get an acceptable low average error value. But again we should consider that to model a data set with a higher number of points satisfactorily, more rules may be necessary. In the case of the Swiss Roll dataset, the number of input points is 2000. Since 20 is 1% of 2000, n_c is set as 1% of the number of input instances (rounded to the nearest integer). Here also for the COIL data set to avoid under-fit, we choose $n_c = 5$ instead of $n_c = 1$. With $r = 0.2$ or 0.3 , low average final error values are obtained for all the values of n_c . So, along with Swiss Roll for all three synthetic data sets, we conduct experiments with these two choices of σ . However, for real data sets, we have to choose higher values of σ to avoid many cases of no rule firing. For 0s of USPS data set (we shall refer to it as USPS) and Frey face we initialize σ s as 0.4 times the feature-specific range. For COIL data set, σ s are initialized as 0.5 times the feature-specific range. In all cases, consequent parameters are initialized with random values in $(-0.5, 0.5)$.

For both synthetic and real-world data sets, we repeat the experiments five times for each choice of the spread. For the three synthetic data sets, there are two initial choices for spread. So ten runs are conducted giving ten rule-based systems for each synthetic data set. On the other hand, for each real-world data set, we use only one choice of the initial spread and thereby generate five rule-based systems. Then based on the minimum value of the objective function in (3.2.3), we choose the best rule-based system for a given data set. We do *not* look at any test data and the best result can always be chosen in an automatic manner without any human intervention based on (3.2.3) as this gives the training error.

To optimize the objective function (3.2.3) we use the *'train.MomentumOptimizer'* class of the *'TensorFlow'* framework [193] with learning rate = 0.1 and momentum = 0.5. For a fair comparison, for the comparing methods, we apply the synthetic data sets directly and the real-world data sets are used after feature-wise zero-one normalization. For computing Sammon's projection, we use the MATLAB implementation of multi-dimensional scaling [74], *mdscale* with the parameter 'Criterion' set to 'sammon'. The dimension of the output is set to two. For other parameters, the default values are used. The ISOMAP, LLE, and *t*-SNE are implemented respectively using the classes 'manifold.Isomap', 'manifold.LocallyLinearEmbedding', and 'manifold.TSNE' of the 'scikit-learn'(sklearn) package [167] (version 0.19.2) of python. UMAP is implemented using the python API 'umap' [46]. The diffusion net is implemented with the available code [194]. For LLE, ISOMAP, UMAP, and diffusion net we use the same neighborhood size (number of nearest neighbors) as used by the proposed method. For *t*-SNE, there is no provision to set the neighborhood size directly. The 'perplexity' [45] parameter of *t*-SNE provides an indirect measure of the effective number of neighbors. So we set the 'perplexity' parameter to the number of nearest neighbors used in the proposed method. For these methods, all other parameters are kept at their default values as supported by the respective routines. For out-of-sample testing, except for SNPPE, the other three comparing methods are realized using the Matlab Toolbox for Dimensionality Reduction (drtoolbox) by Laurens van der Maaten [195]. The results of SNPPE are generated using the code provided by the authors of SNPPE. In these cases also the neighborhood size, when required, is set to the same value as used by

the proposed method. All the other parameters are kept at their default values.

3.3.3 Results and Comparisons

Figures 3.3a, 3.4a, and 3.5a depict the projections made by the proposed method. These figures reveal that the proposed method successfully unfolds the non-linear structures present in the three data sets, Swiss Roll, S Curve, and Helix data respectively to a nearly linear structure. For all three data sets, Sammon’s projection (Figs. 3.3b, 3.4b, 3.5b) cannot unfold the original data to its intrinsic linear structure but it tries to preserve the global geometric structure. ISOMAP tries to preserve the pairwise distances over the manifold by using the geodesic distance in the classical multidimensional scaling error function [42]. Preserving geodesic distance, ISOMAP is able to unfold the non-linear structure as shown in Figs. 3.3c and 3.4c. A drawback of the system is that large distances play a stronger role compared to small distances in its objective. As a result, in the case of the Helix data set, as seen in Fig. 3.5c, it cannot unfold the data as desired. LLE, t-SNE, UMAP, and diffusion net try to capture the local structure present in the original data, in the projected space. Figures 3.3d, 3.4d, and 3.5d reveal that the LLE can unfold the underlying structure to a reasonable extent but the quality of unfolding is far from that by the proposed method and ISOMAP. Similarly, t-SNE and UMAP although they have the same aim, are not able to unfold the data sets properly as revealed by Figs. 3.3e, 3.3f, 3.4e, 3.4f, 3.5e, and 3.5f. Figures 3.3g, 3.4g, 3.5g indicate that, apart from the Helix data set, (Fig. 3.5g), the diffusion net is not successful in capturing the variations along the intrinsic dimensions. For both Swiss Roll and S Curve, the diffusion net embedding captures variations along one intrinsic feature out of the two. In summary, of the seven methods, only the fuzzy rule-based method is successful for all three data sets and the next best performing method is ISOMAP.

The proposed method can also faithfully project real-world data sets in two dimensions. In Fig. 3.6, the two dimensional embedding of the Frey-face data set is shown. Figure 3.6 reveals that the gradual change in the face orientation from right to left is roughly related to the increase in the projected feature y_1 on the horizontal axis. Similarly, the change in expressed emotion varies roughly with the extracted feature (projection) y_2 . We observe that the smiling, neutral, and annoyed faces are mapped in the top, middle, and bottom regions, respectively. The results of the five comparing methods are placed in Figs. 3.7, 3.8, 3.9, 3.10, 3.11, 3.12. Sammon’s projection being a Euclidean distance preserving method has placed the faces expressing different emotions in overlapping regions as seen in Fig. 3.7. The other five methods which preserve neighborhood in some ways through their objective functions, have unfolded the high dimensional manifold comparatively better than Sammon’s method. All these five methods are successful in mapping the faces with different emotions apart to some extent, however, in the projections of LLE (Fig. 3.9), UMAP (Fig. 3.11), diffusion net (Fig. 3.12) the faces are cluttered in regions instead of unfolding with smooth variations of emotions. Comparing the results in Figs. 3.8, 3.9, 3.10, 3.11, and 3.12 with the result in Fig. 3.6 we can infer that the proposed method has mapped the variations of

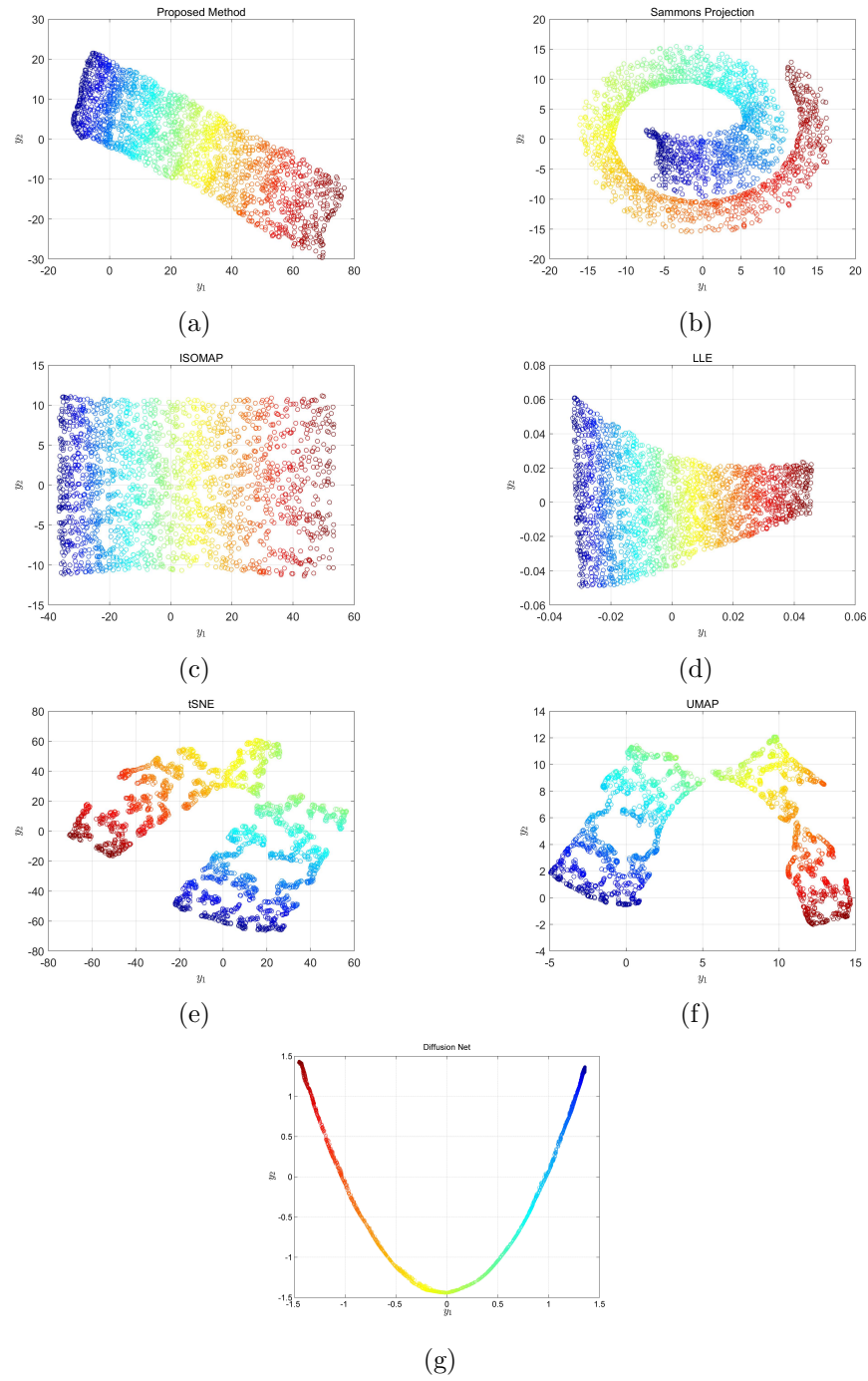


Figure 3.3: Visualization of the Swiss Roll data with (a) proposed method, (b) Sammon's projection, (c) ISOMAP, (d) LLE, (e) t -SNE, (f) UMAP, and (g) Diffusion net.

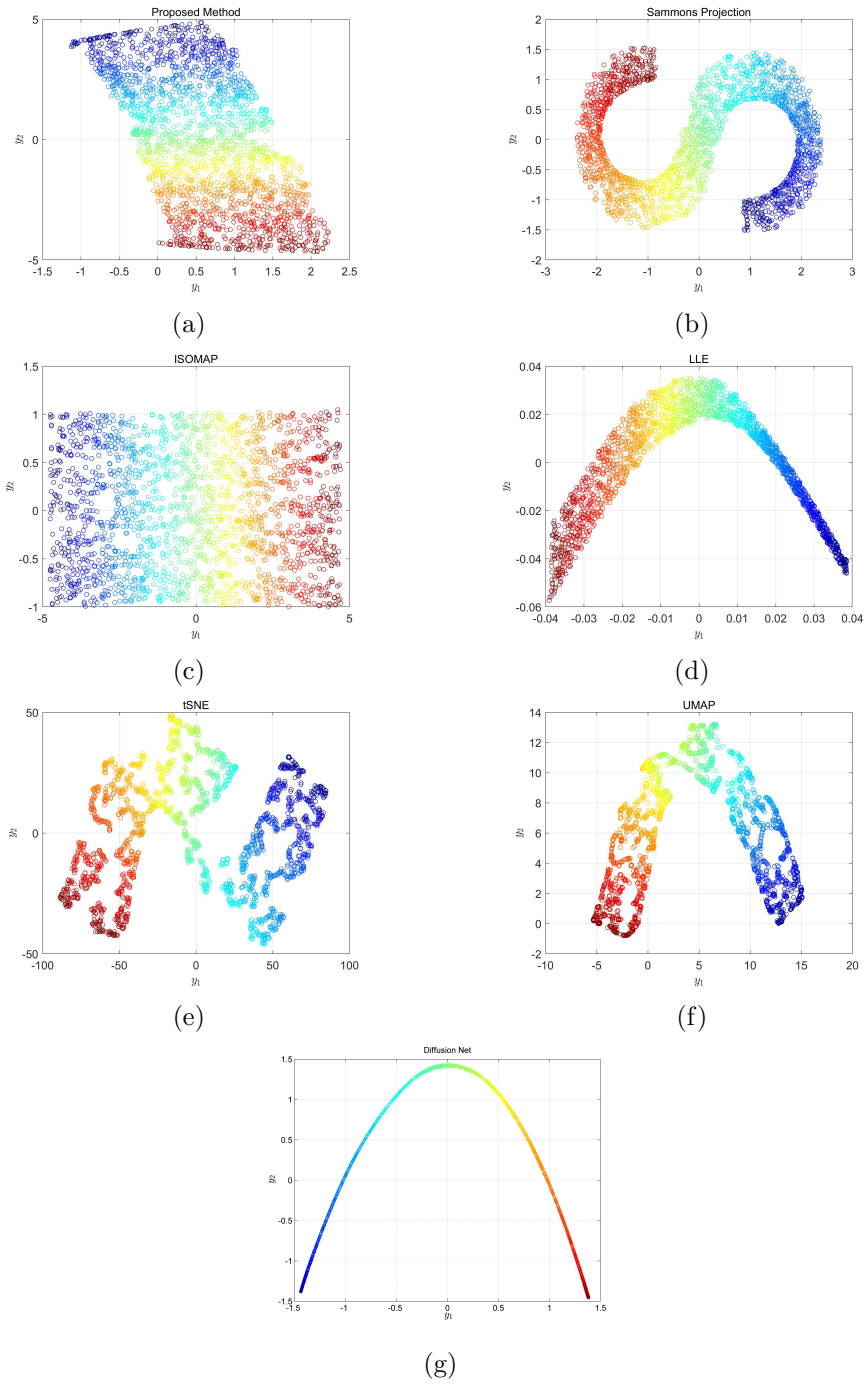


Figure 3.4: Visualization of the S Curve data with (a) original input space, (b) proposed method, (c) Sammon's projection, (d) ISOMAP, (e) LLE, (f) t -SNE, (f) UMAP, and (g) Diffusion net.

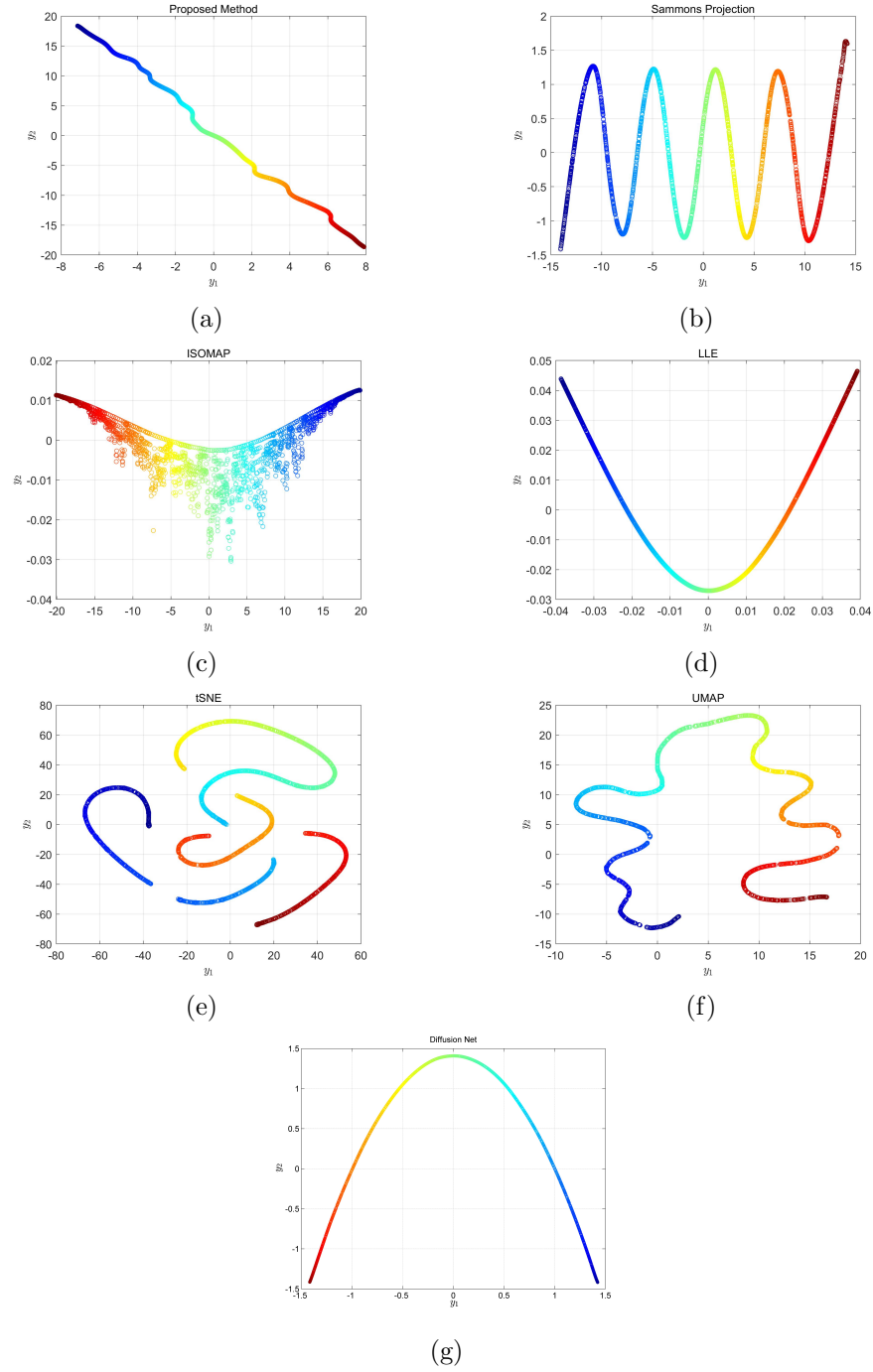


Figure 3.5: Visualization of the Helix data with (a) proposed method, (b) Sammon's projection, (c) ISOMAP, (d) LLE, (e) t -SNE, (f) UMAP, and (g) Diffusion net.

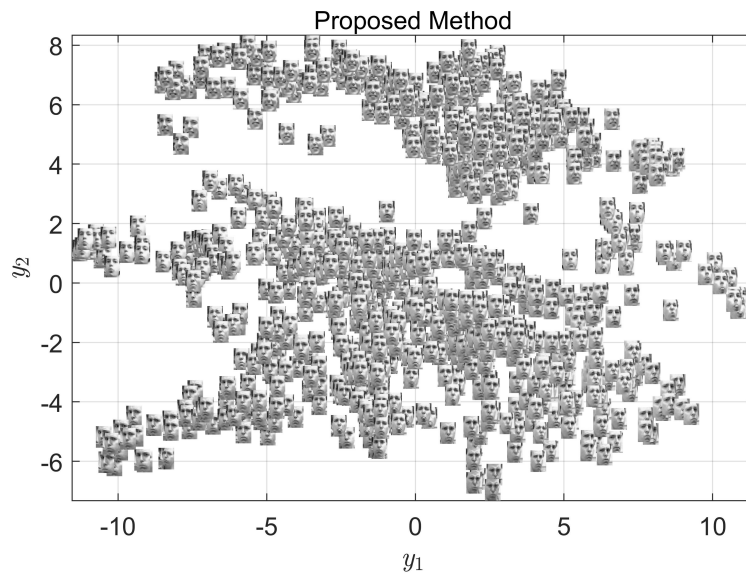


Figure 3.6: Visualization of the Frey face data set with the proposed method.

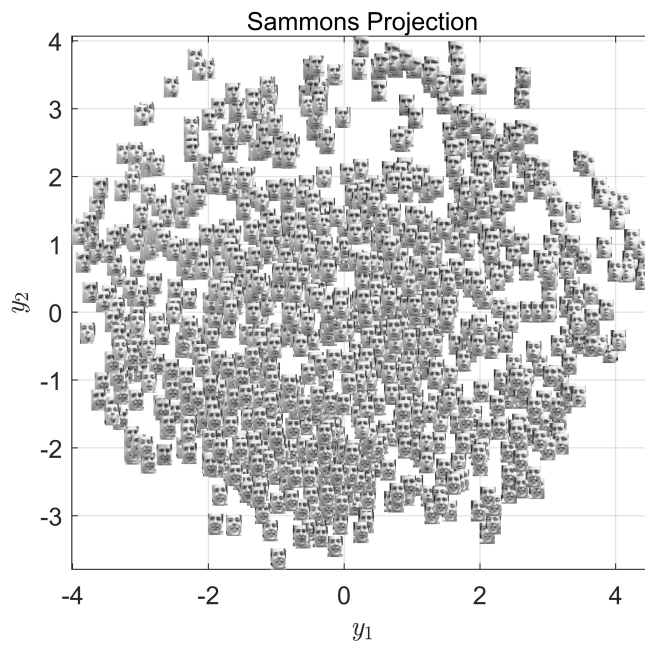


Figure 3.7: Visualization of the Frey-face data set with Sammon's Projection.

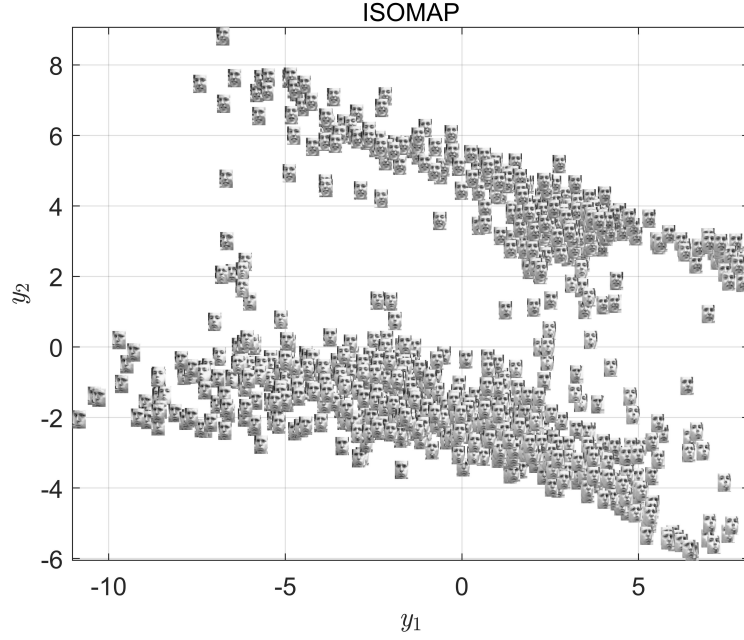


Figure 3.8: Visualization of the Frey-face data set with ISOMAP

Frey face data set comparatively in a more consistent and desired manner.

Figures 3.13 and 3.14 depict the two dimensional embeddings of the COIL data set by various methods. For this data set, the expected outcome is obtained by the proposed method (Fig. 3.13a) as well as by ISOMAP (Fig. 3.13c), t -SNE (Fig. 3.14a), and diffusion net (Fig. 3.14c). However, results obtained from Sammon’s projection, LLE, and UMAP shown in Figs. 3.13b, 3.13d, and Fig. 3.14b, respectively cannot recover the manifold faithfully on the two dimensional space.

Figure 3.15 shows a visualization of the handwritten character 0s of the USPS data set by the proposed method. In this case also, the proposed method successfully projects zeros with different orientations and line widths in different regions in a continuous manner. The projections by the other six methods shown in Figs. 3.16, 3.17, 3.18, 3.19, 3.20, and 3.21 reveal that LLE (Fig. 3.18), t -SNE (Fig. 3.19), UMAP (Fig. 3.20), and diffusion net (Fig. 3.21) could not represent the data in a way so that the orientation and line width of the character vary smoothly in the projected space. In all three cases of the real-world datasets, the proposed method successfully learns the approximate two dimensional manifold on which the original data lie.

3.3.4 Impact of Initial Rules

Our method initializes the rule antecedents using the centroids of the clusters obtained by Algorithm 1. To show the impact of the initial rules we have initialized the rule antecedents using uniformly distributed random values generated from the smallest

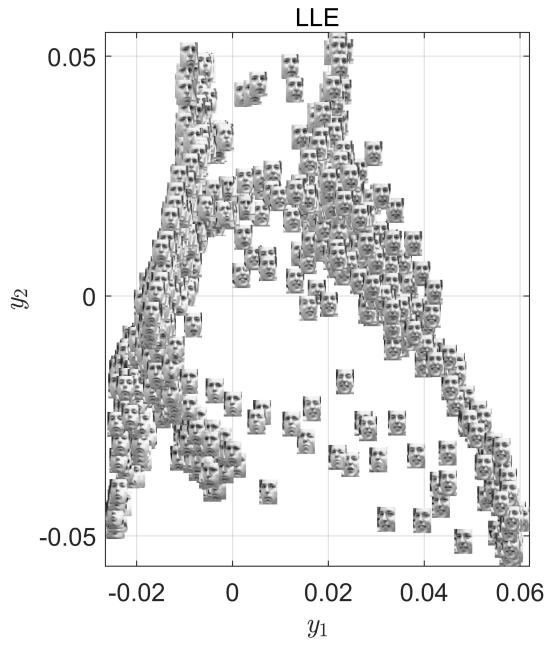


Figure 3.9: Visualization of the Frey-face data set with LLE.

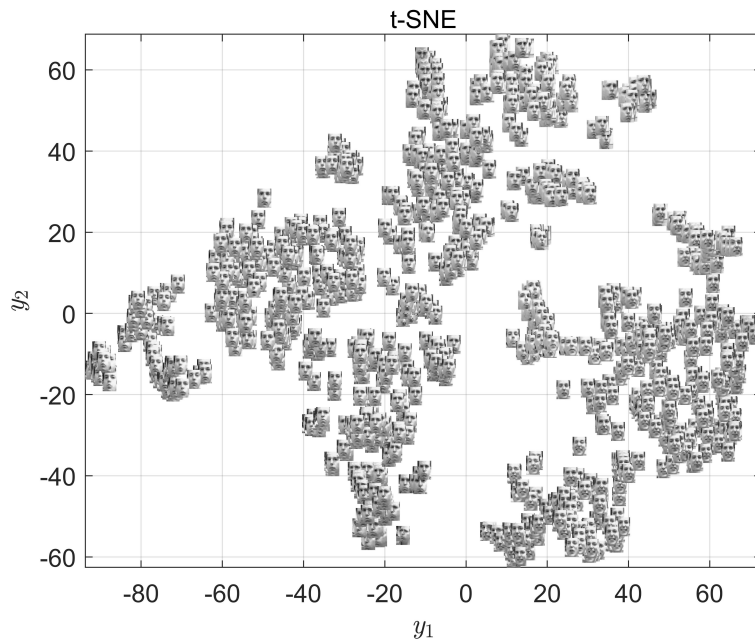


Figure 3.10: Visualization of the Frey-face data set with t -SNE.

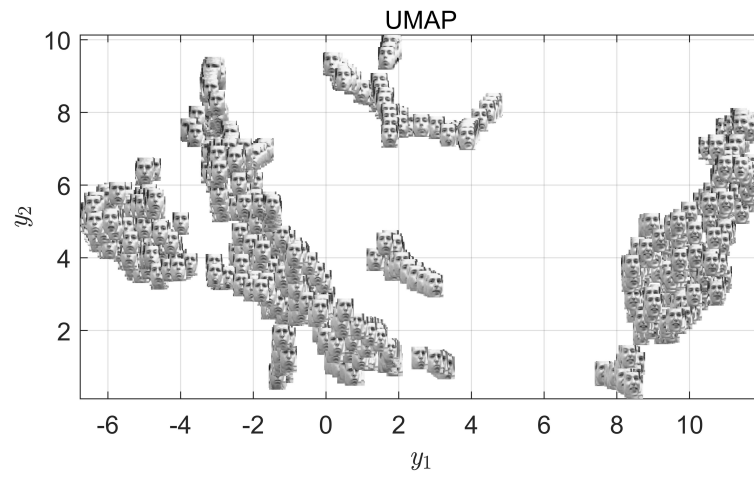


Figure 3.11: Visualization of the Frey-face data set with UMAP.

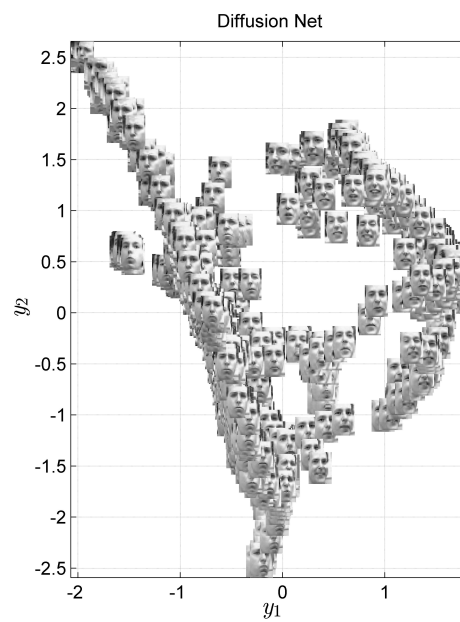


Figure 3.12: Visualization of the Frey-face data set with diffusion net.

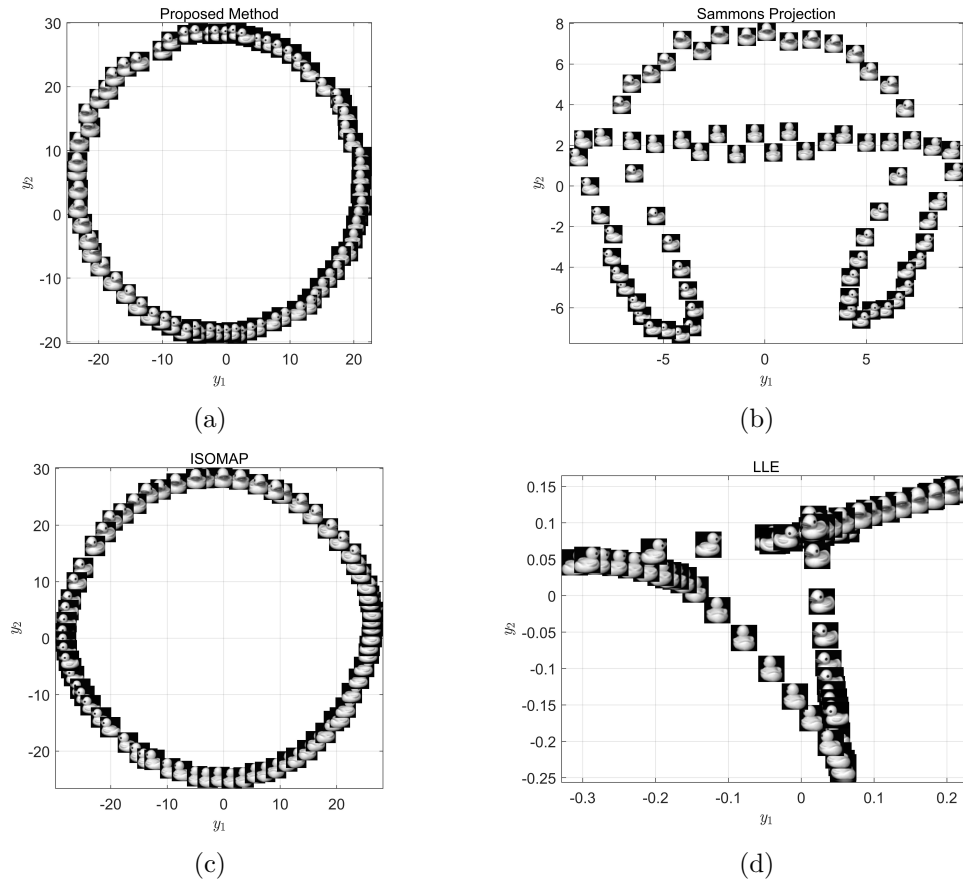


Figure 3.13: Visualization of the first object of the COIL Data Set with (a) proposed method, (b) Sammon's projection, (c) ISOMAP, (d) LLE.

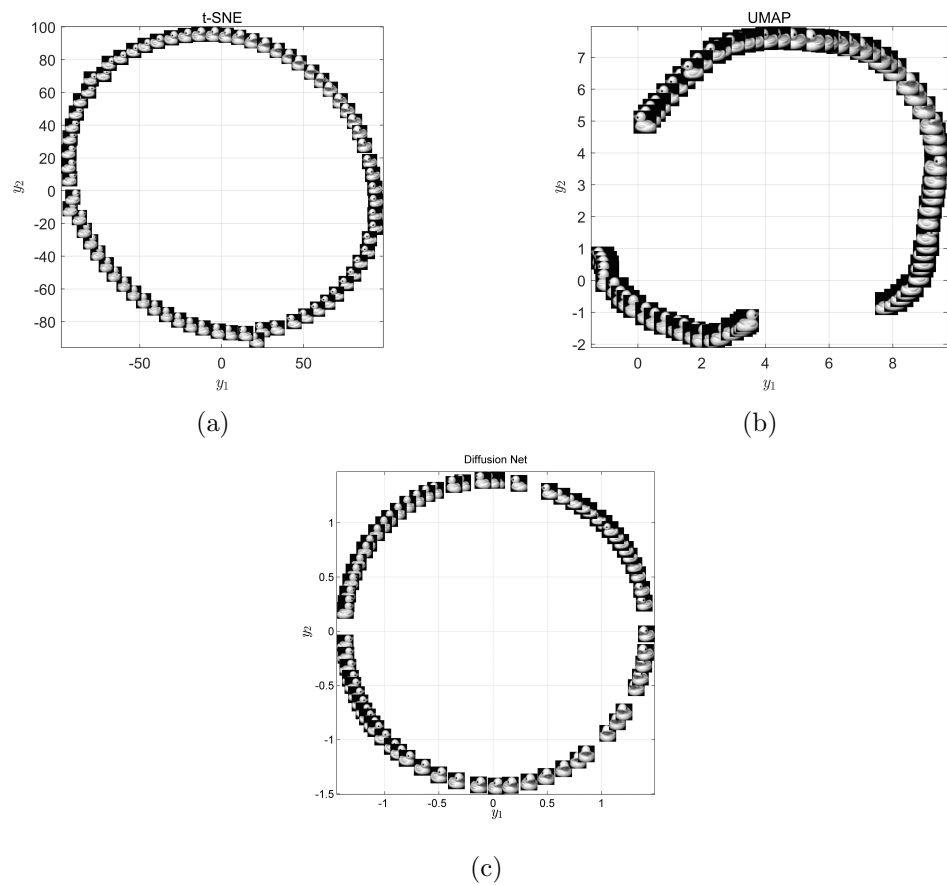


Figure 3.14: Visualization of the first object of the COIL Data Set with (a) *t*-SNE, (b) UMAP, and (c) diffusion net.

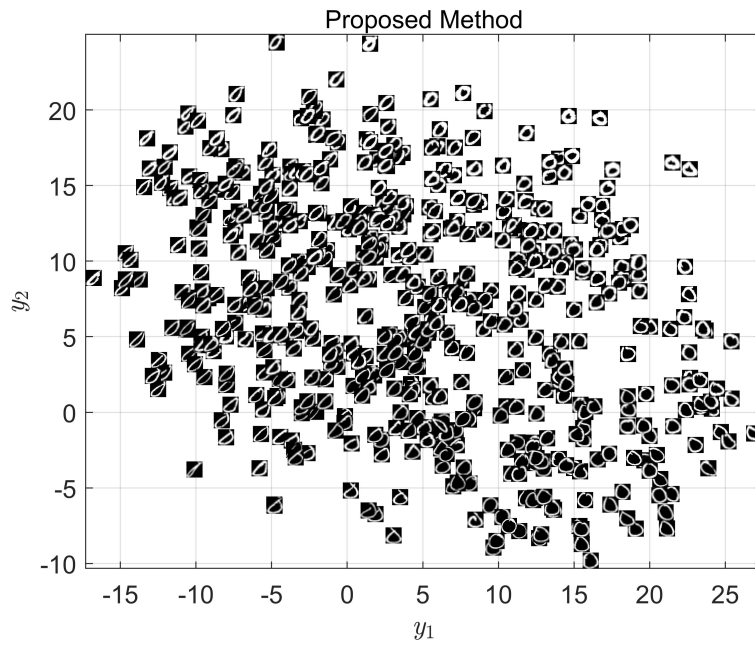


Figure 3.15: Visualization of the USPS data set with the proposed method.

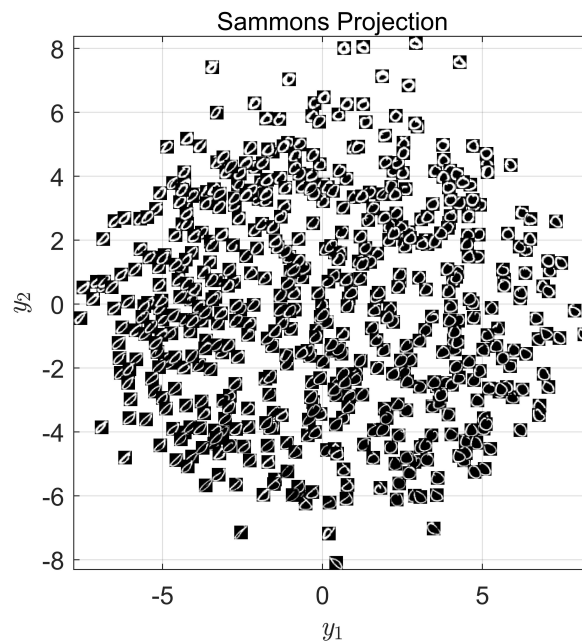


Figure 3.16: Visualization of the USPS data set with Sammon's Projection.

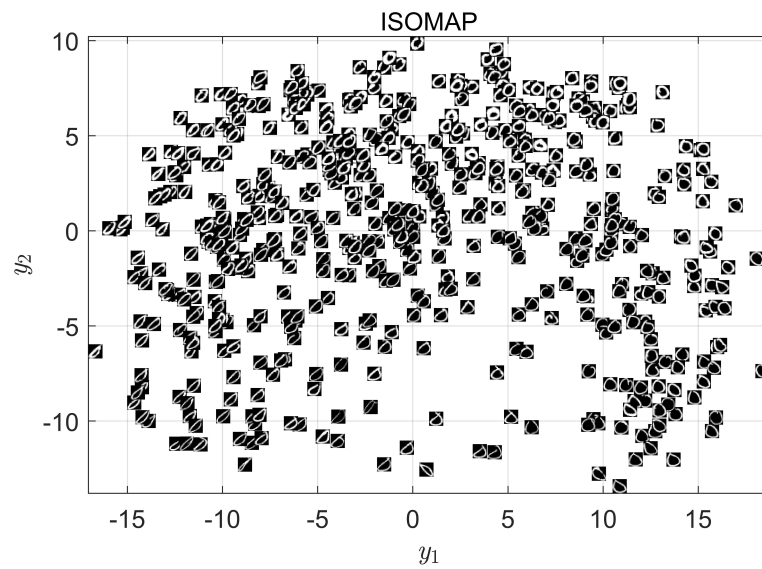


Figure 3.17: Visualization of the USPS data set with ISOMAP

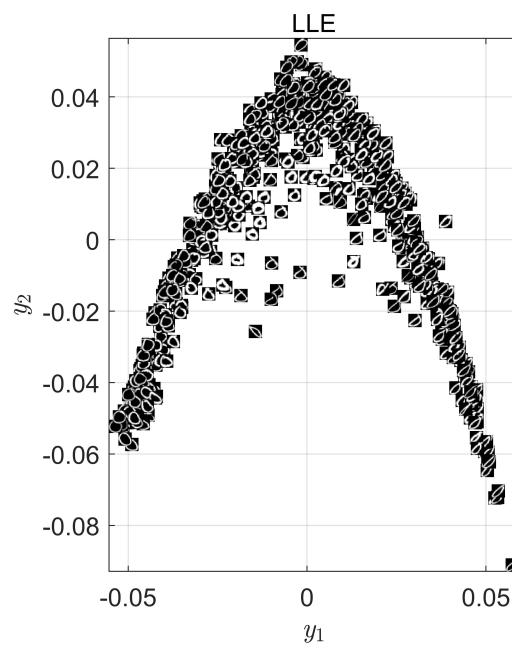


Figure 3.18: Visualization of the USPS data set with LLE.

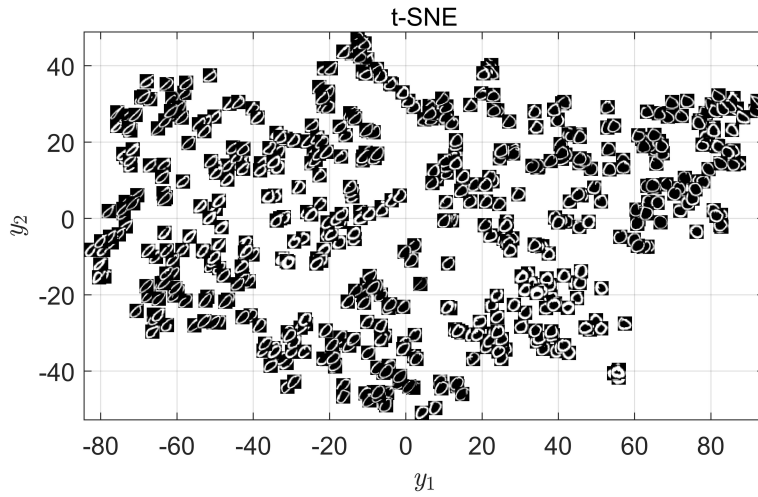


Figure 3.19: Visualization of the USPS data set with t -SNE.

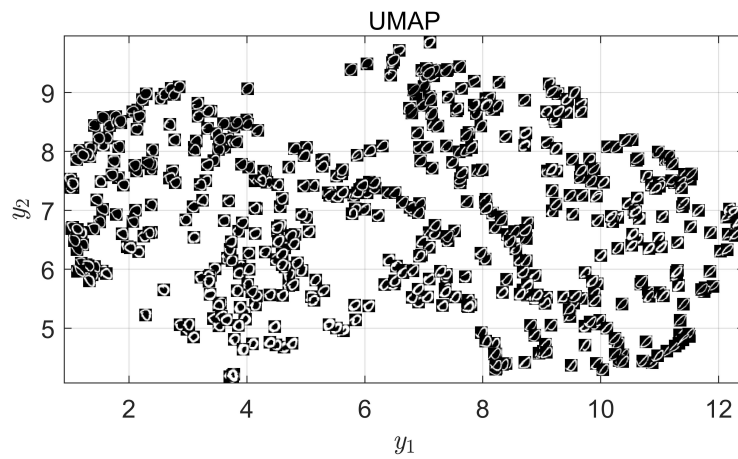


Figure 3.20: Visualization of the USPS data set with UMAP.

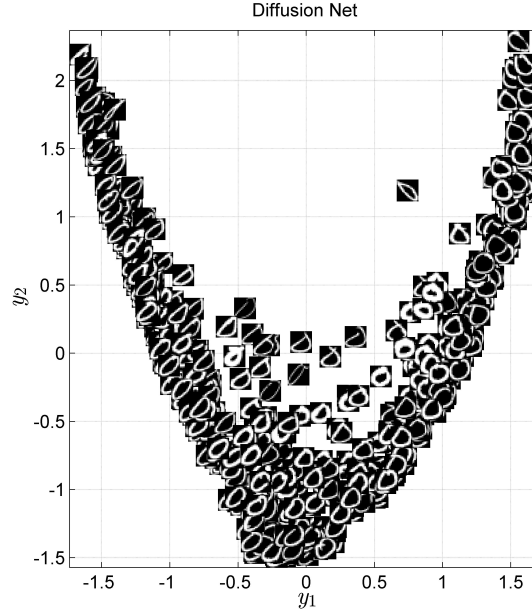


Figure 3.21: Visualization of the USPS data set with diffusion net.

hyperbox containing the training data. The obtained projection for the Swiss Roll data is shown in Fig. 3.22b. Figure 3.22a shows the result obtained by the proposed method with rule antecedents defined by the cluster centroids. From Figs. 3.22a and 3.22b it is evident that with or without judicious initialization of rule antecedents, the proposed method can unfold the non-linear structure satisfactorily. This observation is also true in the case of the Frey face data set. Fig. 3.23 shows the visualization of Frey face data set using the proposed method with random initialization of the antecedents as mentioned above. To investigate further, we have computed the average final error (equation (3.2.3)) over the ten runs on the Swiss Roll dataset and five runs on the Frey Face dataset (performed as explained in sub-section 3.3.2) with both types of initialization of the rule antecedents. Table 3.2 shows that the averages of the final error values are less in the case of initialization of the rule antecedents by cluster centroids compared to those by random initialization for both data sets. This suggests that although in both cases the algorithm ends up in useful local minima, for the cluster-based initial rules, generally the minima are superior. We have also done 10 experiments on the Swiss Roll data with rules whose Gaussian membership centers are initialized with random values in $[0,1]$ hypercube. The two results of those experiments are shown in Fig. 3.24. Even in this case, the results are reasonably good, but not as good as the cases with judiciously initialized rules. In a few cases, the system could not unfold. Thus, the system is quite robust with respect to the initial choice of rules but certainly, cluster-based rules help.

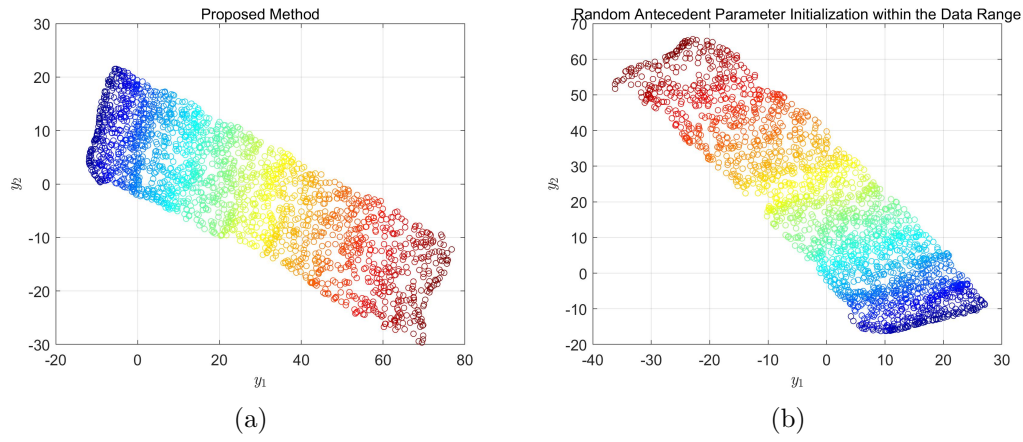


Figure 3.22: Visualization of the Swiss Roll data using proposed method with rule antecedents defined by (a) cluster centroids, (b) random values from the smallest hyperbox containing the training data.

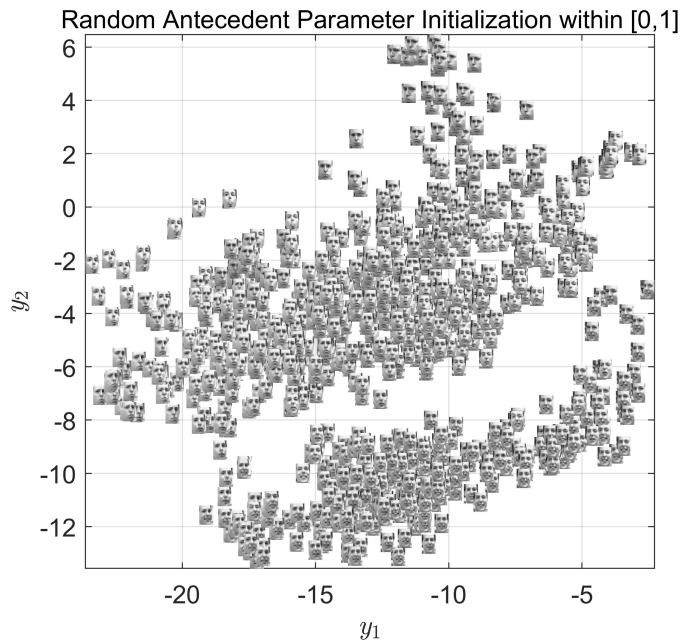


Figure 3.23: Visualization of the Frey face data set using the proposed method with random rule antecedents.

Table 3.2: Average of the final error values for different initializations

Data Set	Initial rule antecedents defined by	
	Cluster centroids	Random
Swiss Roll	0.04740	0.06490
Frey face	0.50459	0.50500

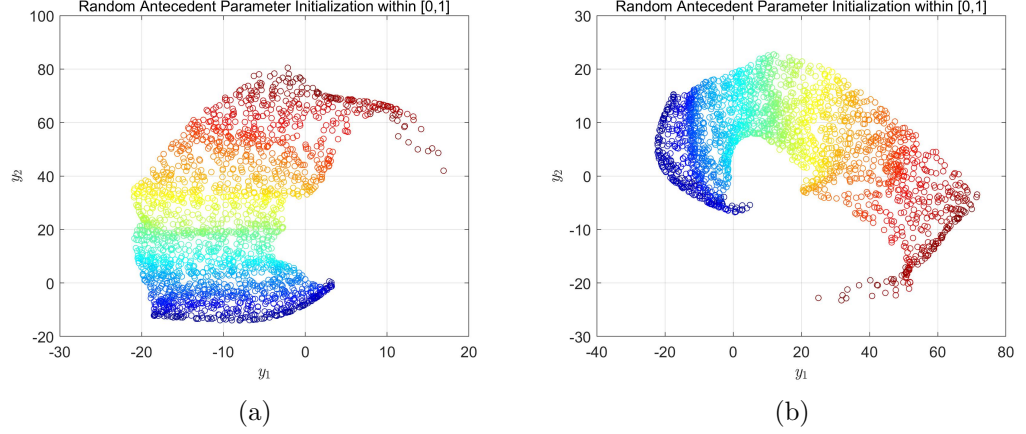


Figure 3.24: Visualization of the Swiss Roll data set by proposed method with rule antecedents defined by random values within zero and one: (a) best run (b) worst run.

3.3.5 Validation of Predictability for Out-of-sample Points

One of the key advantages of utilizing an FRBS is that it is parametric and thereby providing an explicit mapping function from the high dimensional inputs to a lower dimensional one. So, after the system is trained it is possible to predict the lower dimensional data for new test points. As mentioned earlier, to compare the predictability of the proposed method we consider four methods: out of sample extension of ISOMAP and LLE [148] and two parametric methods: autoencoder [24] and SNPPE [83]. Note that SNPPE involves a polynomial mapping. Following [83] we use the 2nd order and 3rd order polynomial-based SNPPE. To assess the predictability of the system, we have designed three experiments: (i) randomly dividing the data set into training and test sets; (ii) using a contiguous portion of the manifold as the test set and rest as the training set; and (iii) leave one out validation. We have performed the first test on the Swiss Roll and Frey face data sets. We use 75% of the Swiss Roll data set, i.e., 1500 randomly selected points to form the training set and the rest 25%, i.e., 500 points to form the test set. The training and test data sets are shown in Fig. 3.25a and Fig. 3.25b respectively. While partitioning the data set, the associated label (color) vectors are also partitioned accordingly. Figure 3.26a and Fig. 3.26b are the results corresponding to the training and test data sets by the proposed method. From Fig.

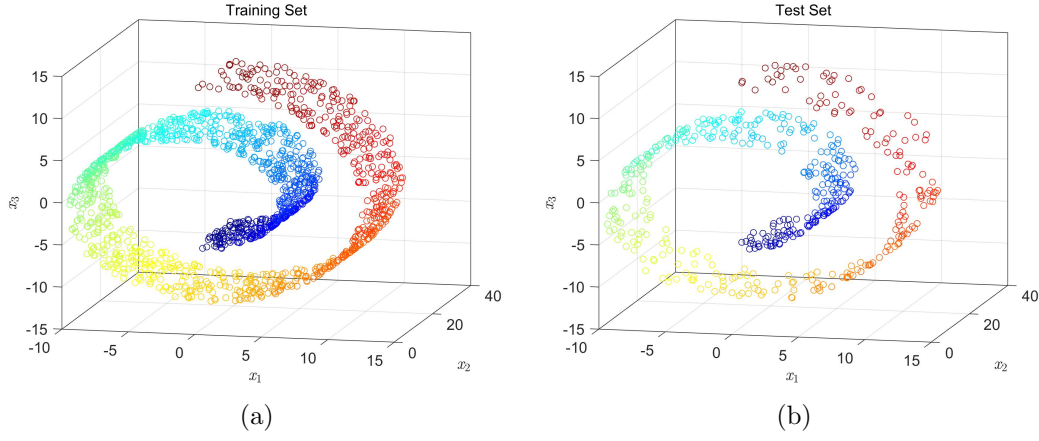


Figure 3.25: For experiment 1 on validation of predictability with the Swiss Roll data : (a) Training Set, (b) Test Set

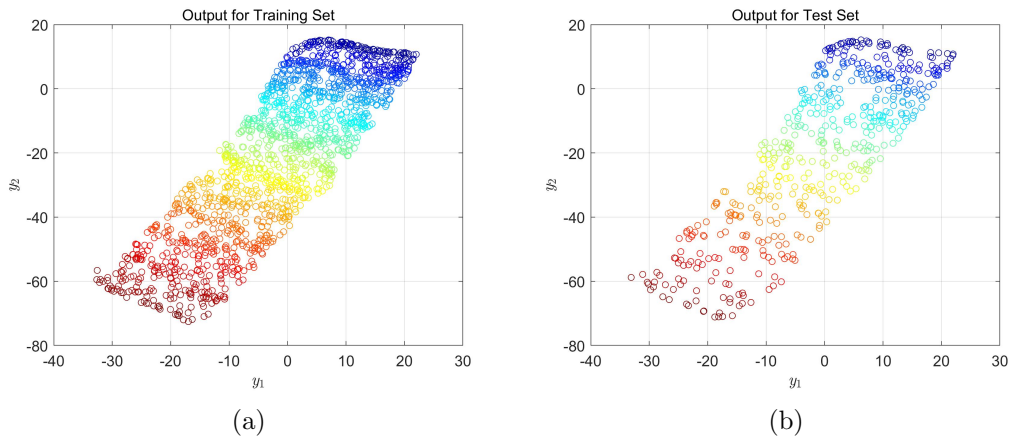


Figure 3.26: For experiment 1 on validation of predictability with the Swiss Roll data proposed method output for (a) Training Set, (b) Test Set.

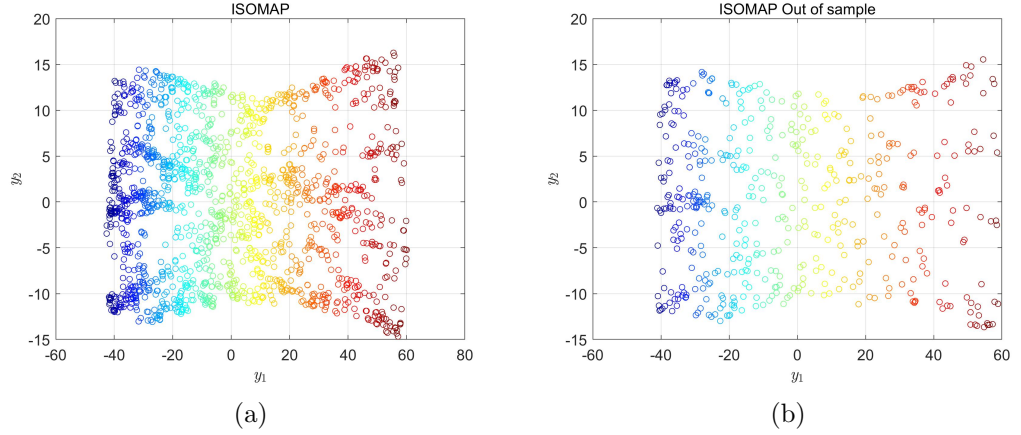


Figure 3.27: For experiment 1 on validation of predictability with the Swiss Roll data : (a) ISOMAP output for Training Set, and (b) Out of sample extension of ISOMAP for Test Set.

3.26b it is evident that the predicted locations determined by the proposed method for the test points are at their expected positions in the lower dimension. Figs. 3.27, 3.28, 3.29, 3.30, and 3.31 show the projections of the training and test sets by ISOMAP, LLE, autoencoder, 2nd order SNPPE, and 3rd order SNPPE, respectively. Except for autoencoder, the other compared methods successfully unfold the Swiss Roll structure in the training phase and appropriately predict the test points. From Fig. 3.29 we observe that the autoencoder-based method is unable to preserve the geometry of the training data. In Figs. 3.26b, 3.27b, 3.28b, 3.30b, and 3.31b the color assignment to the projected test points by the label vector shows that the test data set is unfolded in the desired manner in all five cases.

For the Frey face data set, we have selected 15 random instances as the test set and the rest of the points as the training set. Figure 3.32 displays the projected output by the proposed method for the training and test data together where the images with black rectangular borders correspond to the test instances. For all comparing methods for this data, we follow the same representation scheme for the training and test data. In Figs. 3.32, 3.33, 3.34 we can see that the test instances are projected to the regions where the training instances have the same/similar facial expression and orientation. Thus, for high dimensional image data sets also the proposed method, as well as LLE and ISOMAP, are successful in prediction. The autoencoder-based lower dimensional projection, shown in Fig. 3.35 clustered different facial expressions and different face orientations in different regions. However, this method is unable to unfold the intrinsic structure properly because in some cases it placed faces with opposite orientations (left and right) closer to each other compared to the faces with the same orientations. A similar placement has occurred for opposite expressions (e.g. happy and annoyed) and neutral expressions. Figures 3.36 and 3.37 reveal that SNPPE based methods neither unfolded the higher dimensional data nor predicted the positions of the test points in

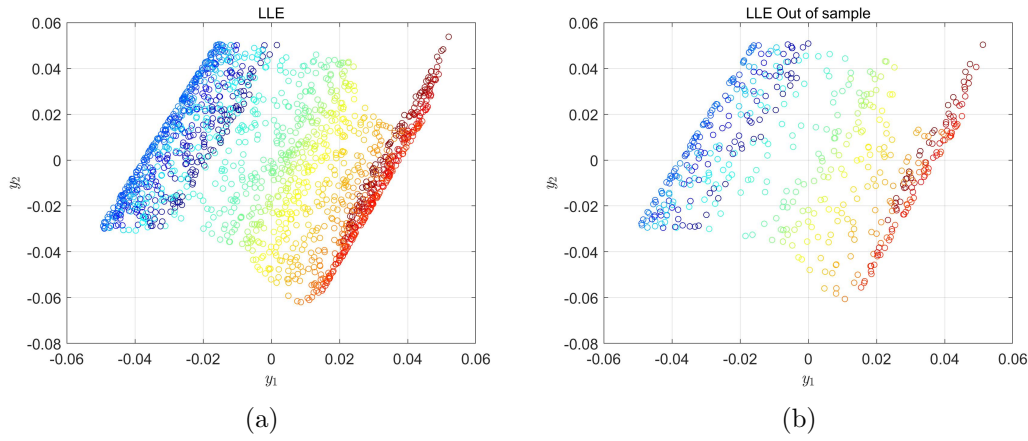


Figure 3.28: For experiment 1 on validation of predictability with the Swiss Roll data : (a) LLE output for Training Set, and (b) Out of sample extension of LLE for Test Set.

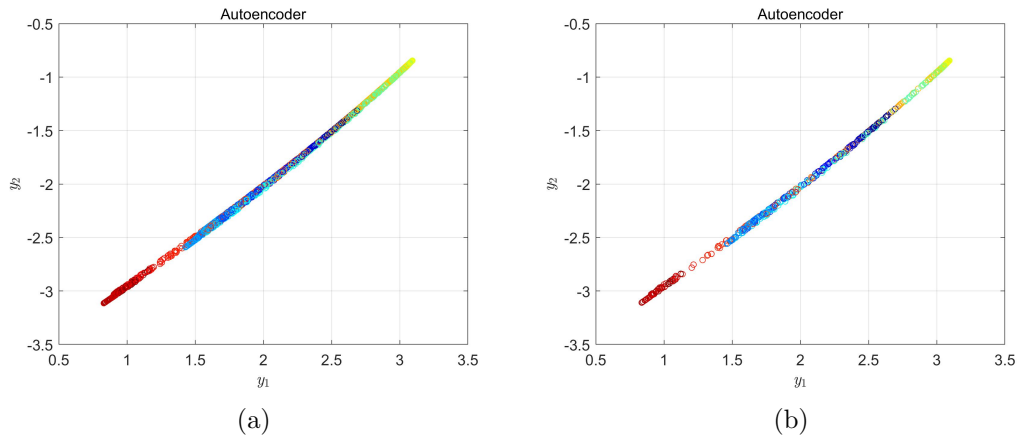


Figure 3.29: For experiment 1 on validation of predictability with the Swiss Roll data : (a) Autoencoder encoded output for Training Set, and (b) Autoencoder encoded output for Test Set.

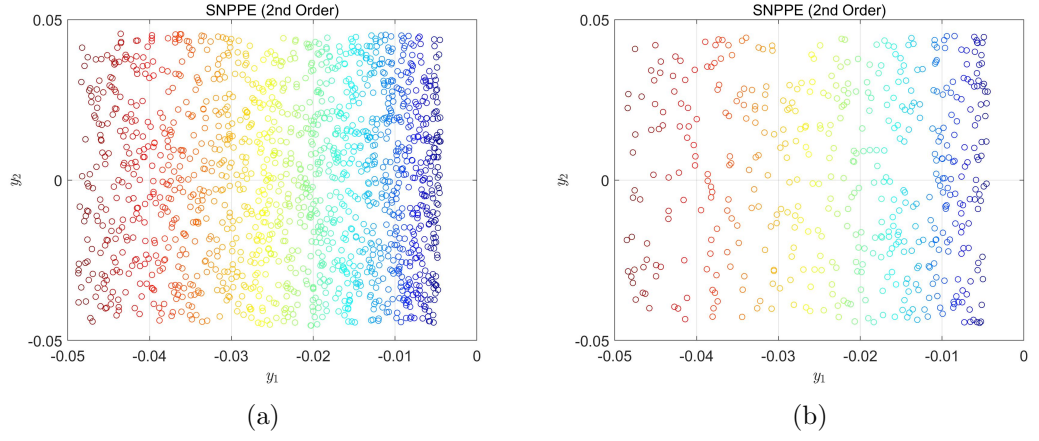


Figure 3.30: For experiment 1 on validation of predictability with the Swiss Roll data : (a) 2nd Order SNPPE output for Training Set, (b) 2nd Order SNPPE output for Test Set.

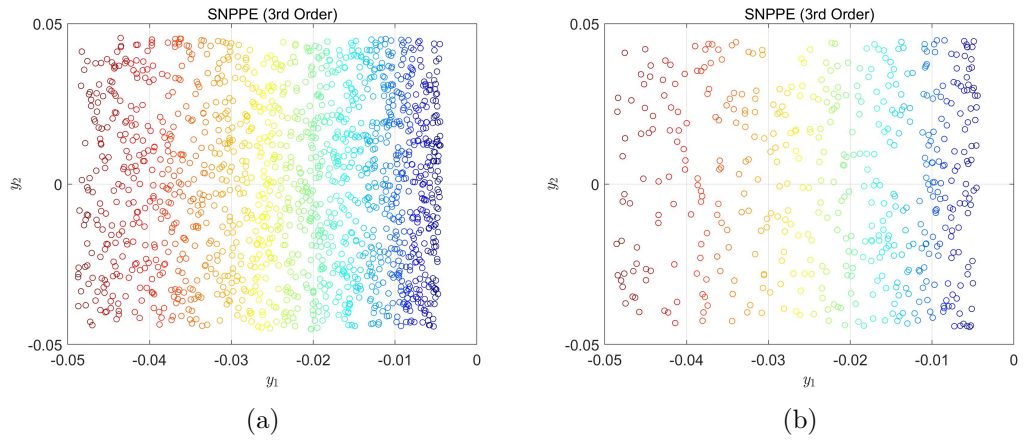


Figure 3.31: For experiment 1 on validation of predictability with the Swiss Roll data : (a) 3rd Order SNPPE output for Training Set, (b) 3rd Order SNPPE output for Test Set.

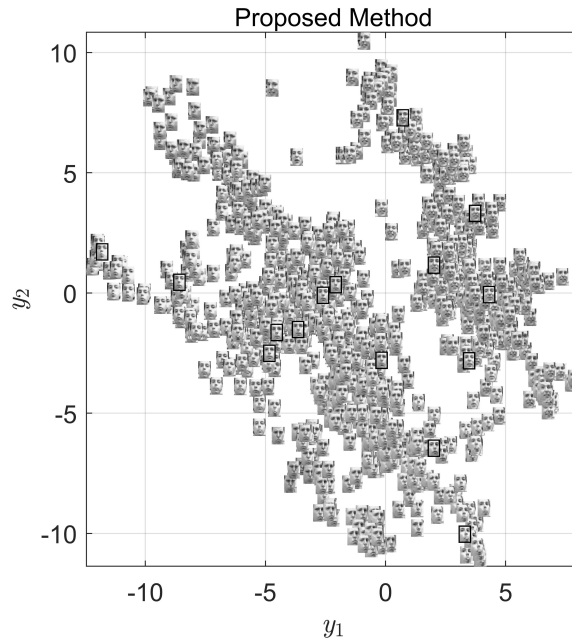


Figure 3.32: Visualization of the training and test set (black rectangle borders) for Frey face data set using the proposed method

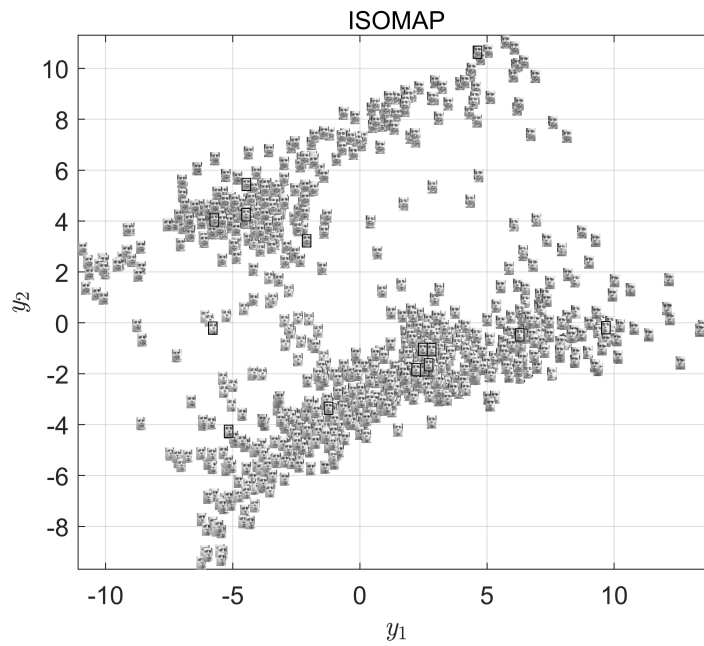


Figure 3.33: Visualization of the training set using ISOMAP and test set (black rectangle borders) with its Out of sample extension for Frey face data set

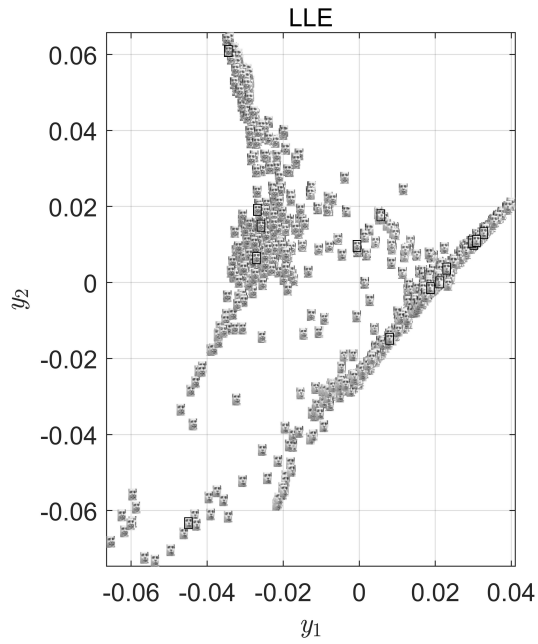


Figure 3.34: Visualization of the training set using LLE and test set (black rectangle borders) with its Out of sample extension for Frey face data set

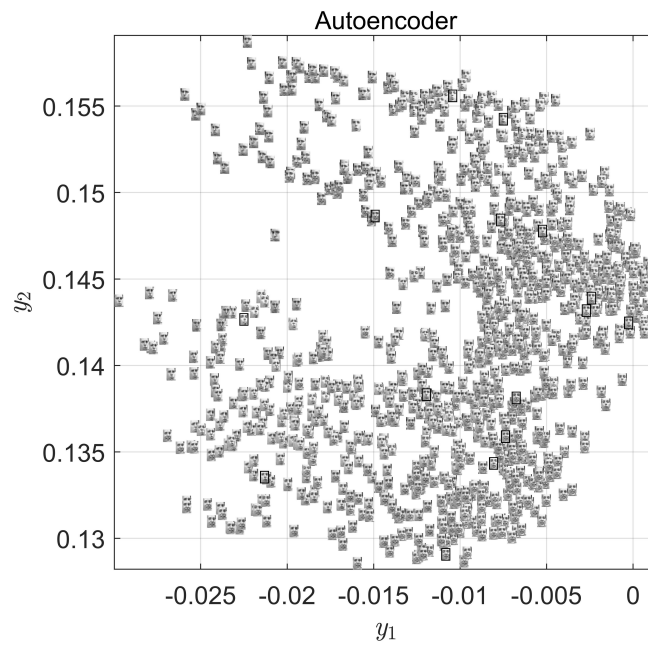


Figure 3.35: Visualization of the training set using Autoencoder encoded output and test set (black rectangle borders) for Frey face data set

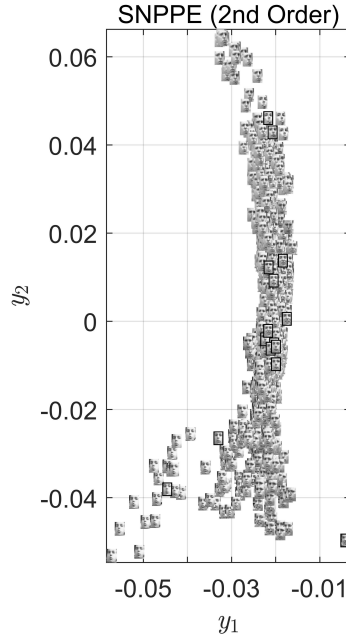


Figure 3.36: Visualization of the training set using 2nd Order SNPPE output and test set (black rectangle borders) for Frey face data set

a desirable manner.

The second test is also done with the Swiss Roll data set. A contiguous portion of the data set consisting of fifty points has been removed and the remaining 1950 points serve as the training data set as shown in Fig. 3.38a. The removed portion serves as the test set (Fig. 3.38b). Figure 3.39a displays the output of the proposed method corresponding to the training set. Figure 3.39b shows the test set outputs of the proposed method along with the training set. The test set points are demarcated with the plus (+) symbol in both original and projected spaces. For the comparing methods, we represent the results in the same manner. Figures 3.40a, 3.41a, 3.42a, 3.43a, and 3.44a show the projections of the training data for ISOMAP, LLE, autoencoder, 2nd order SNPPE, and 3rd order SNPPE, respectively. Similarly, Figs. 3.40b, 3.41b, 3.42b, 3.43b, and 3.44b show the projections of the test set (demarcated with the plus (+) symbol) along with the training set for ISOMAP, LLE, autoencoder, 2nd order SNPPE, and 3rd order SNPPE, respectively. These results reveal that the proposed method, as well as the comparing methods except for the autoencoder, have appropriately interpolated the test points.

The third test uses the leave one out validation on the COIL data set. Each of the 72 instances is considered the test instance and the remaining 71 instances are used as the training set. For each of the 72 training-test sets, outputs are generated. Results of all 72 tests are shown in Figs. 3.45, 3.46, 3.47, 3.48, 3.49, 3.50, 3.51, 3.52, 3.53, 3.54, 3.55, 3.56. The training points are demarcated in green and the test point is

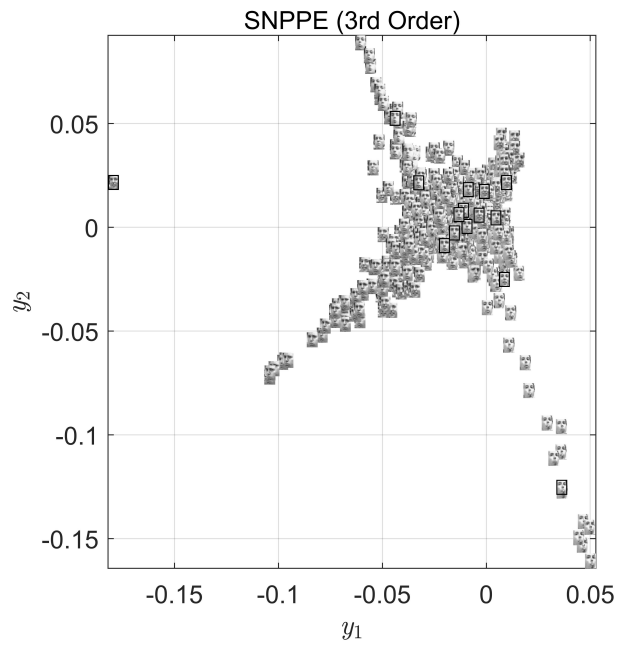


Figure 3.37: Visualization of the training set using 3rd Order SNPPE output and test set (black rectangle borders) for Frey face data set

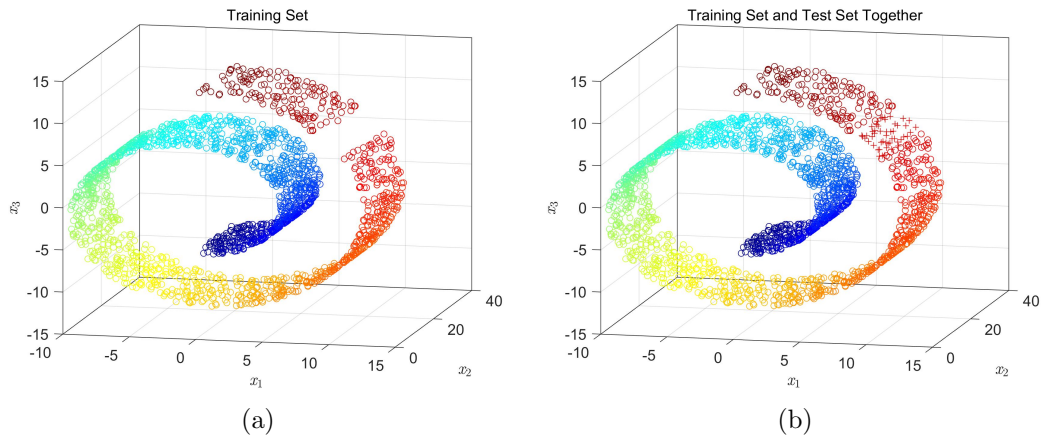


Figure 3.38: For experiment 2 on validation of predictability with the Swiss Roll data, (a) Training Set, (b) Training Set with Test Set.

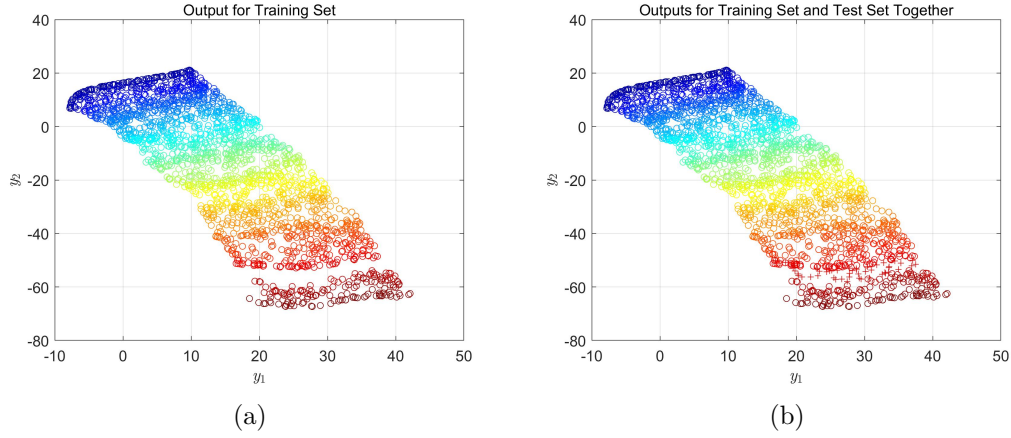


Figure 3.39: For experiment 2 on validation of predictability with the Swiss Roll data, proposed method output for (a) Training Set, and (b) Training Set with Test Set.

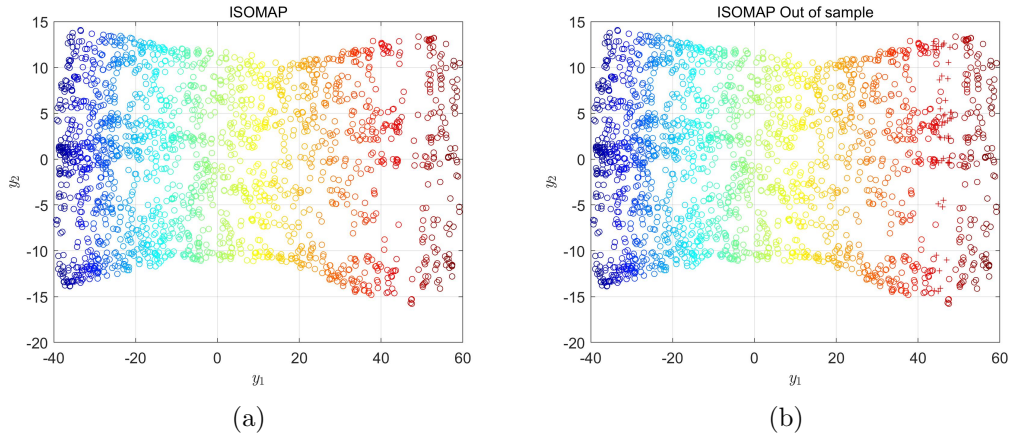


Figure 3.40: For experiment 2 on validation of predictability with the Swiss Roll data, (a) ISOMAP output for Training Set, and (b) Out of sample extension of ISOMAP for Training Set and Test Set.

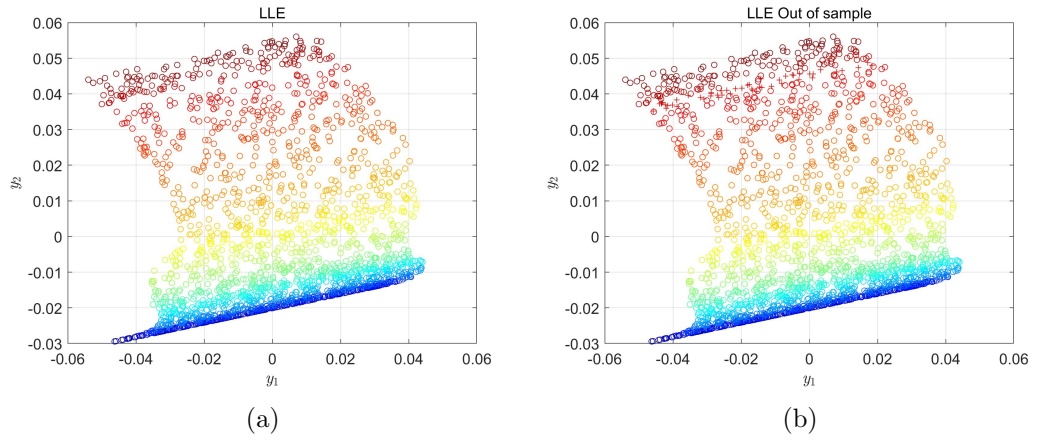


Figure 3.41: For experiment 2 on validation of predictability with the Swiss Roll data, (a) LLE output for Training Set, and (b) Out of sample extension of LLE for Training Set and Test Set.

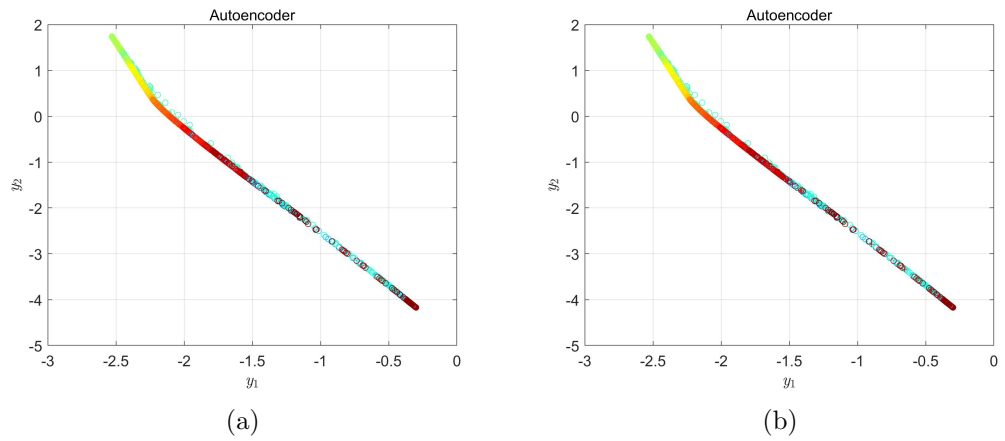


Figure 3.42: For experiment 2 on validation of predictability with the Swiss Roll data, autoencoder encoded output for (a) Training Set, and (b) Training Set and Test Set.

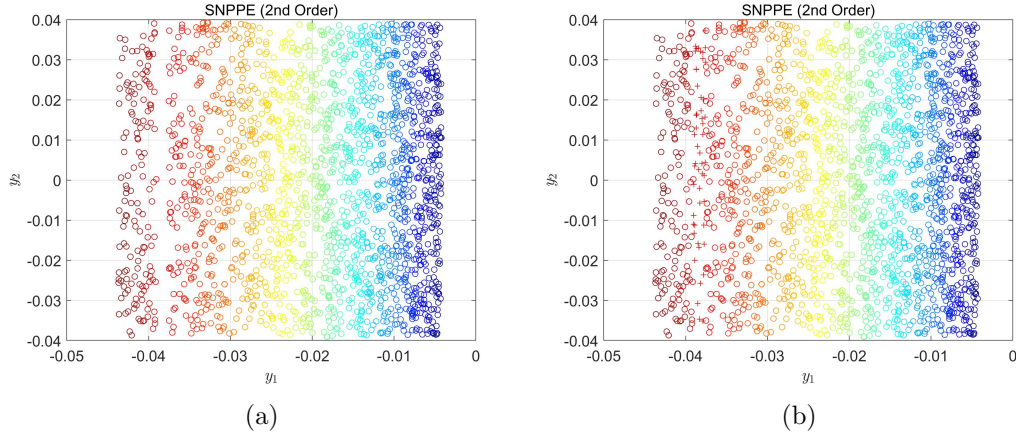


Figure 3.43: For experiment 2 on validation of predictability with the Swiss Roll data, 2nd Order SNPPE output for (a) Training Set, and (b) Training Set and Test Set.

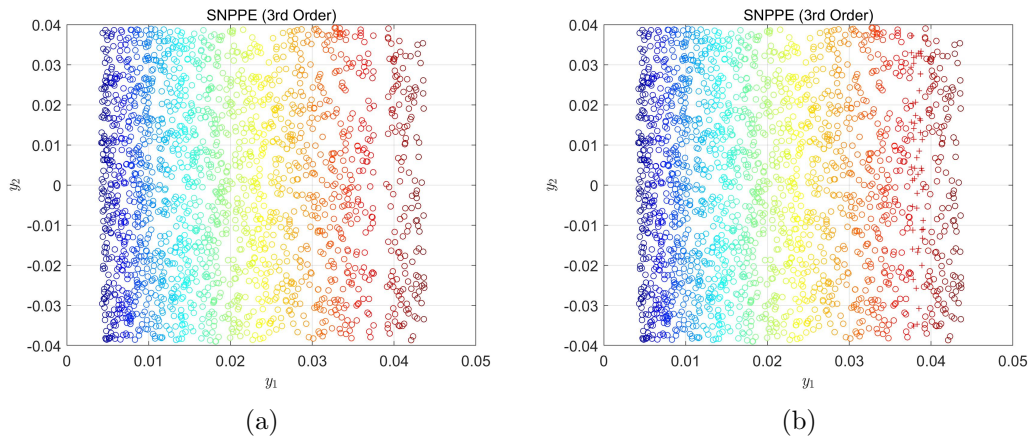


Figure 3.44: For experiment 2 on validation of predictability with the Swiss Roll data, 3rd Order SNPPE output for (a) Training Set, and (b) Training Set and Test Set.

demarcated in red. Instead of showing the images of the object, the consecutive viewing angles of the images are represented by consecutive numbers. In most cases, it is found that the test object is placed in the desired position, i.e., in between images corresponding to input images captured at plus-minus five degrees of the viewing angle of the given test point.

3.3.6 Declining to Project the Outputs

The proposed method is capable of identifying the test cases where the system may not produce reliable outputs. In a TS type fuzzy rule-based system, the output is computed by (2.2.2). The rule having the maximum firing strength mainly characterizes the output. For test points far away from the training data, no fuzzy rule is expected to fire strongly. Consequently, for those points, the outputs obtained may not be reliable. Figure 3.57a shows a set of training points and test points for the Swiss Roll data set. The training points are depicted in blue, while the test points are in red. Test points are located away from the training points. If the firing strength of a test point is very low compared to the firing strength of the training set, then the system should decline to project for such a test point. In order to decide the threshold on the firing strength, we consider the histogram of the maximum firing strengths of rules as shown in Fig. 3.57b. Here also the training and test firing strengths are indicated with blue and red colors, respectively. Figure 3.57b shows that the maximum rule firing strengths of the test points are noticeably lower than those of the training points. This information can be used to decide when the system should reject its output (refuse to make a decision). For example, if the maximum firing strength is less than 0.15, the system output can be rejected. Note that this threshold can be decided based on the training data only. In Fig. 3.57c we have depicted the projected outputs for the training and test data corresponding to Fig. 3.57a without exercising the rejection option. It is clear that the prediction by the proposed method for the test points (red circles) is inappropriate and should be discarded. Note that except Diffusion net, other compared methods do not have any provision for rejecting outputs.

3.3.7 Generalized Nature of the Proposed Framework

The proposed model gives a general framework for learning an unsupervised FRBS for manifold learning or data projection. Instead of using the objective function in (3.2.3), we can use different objectives to preserve different characteristics of the data. For example, the objective function of Sammon's projection [41] can be used to preserve the global geometric structure as in [2]. Figure 3.58 shows the Swiss Roll data and its corresponding lower dimensional output using the proposed method with Sammon's stress [41] as the objective function for learning. Similarly, Figs. 3.59 and 3.60 depict the output produced by the FRBS using Sammon's stress [41] as the objective function for the S Curve and Helix data, respectively. In both cases, the projection reveals what Sammon's method is expected to do. For these experiments, the other settings are kept the same as described in subsection 3.3.2. This system can predict for out-of-sample

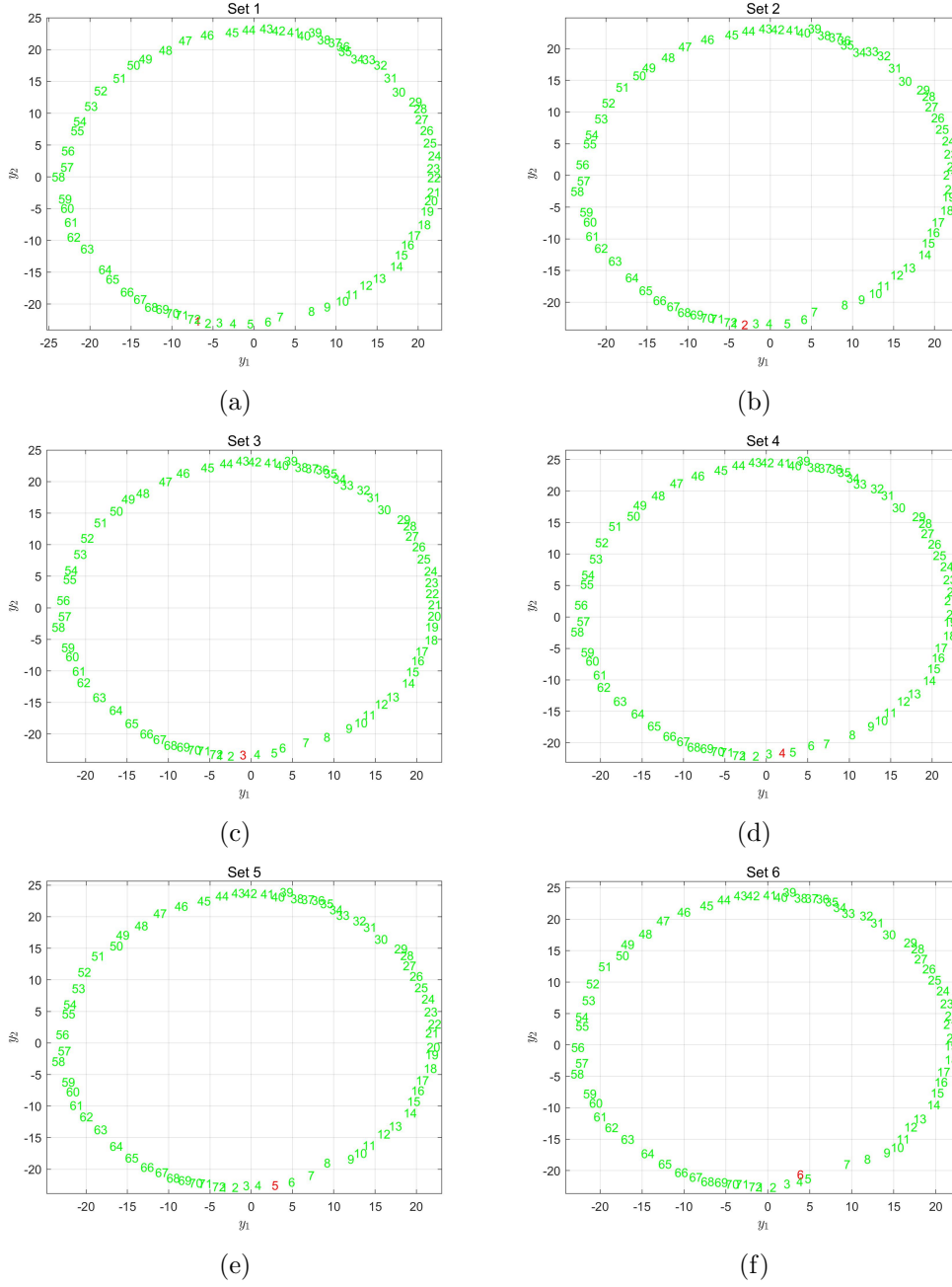
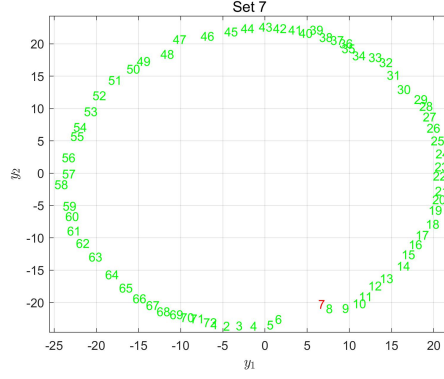
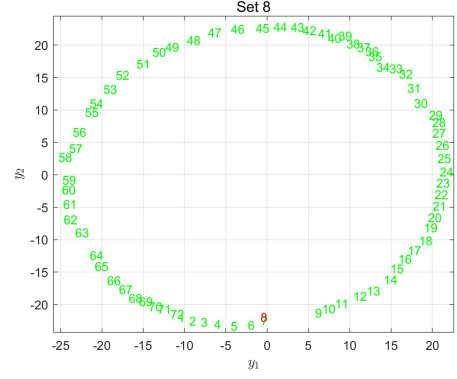


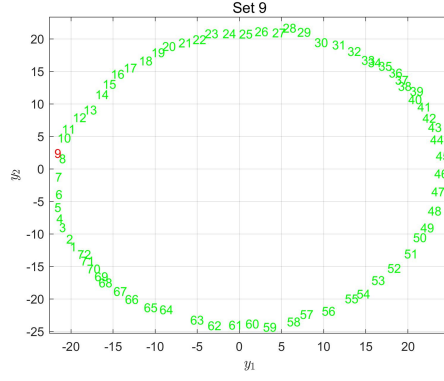
Figure 3.45: For experiment 3 on validation of predictability with leave one out sets composed of the first object of the COIL data set, proposed method outputs corresponding to (a) Set 1, (b) Set 2, (c) Set 3, (d) Set 4, (e) Set 5, and (f) Set 6.



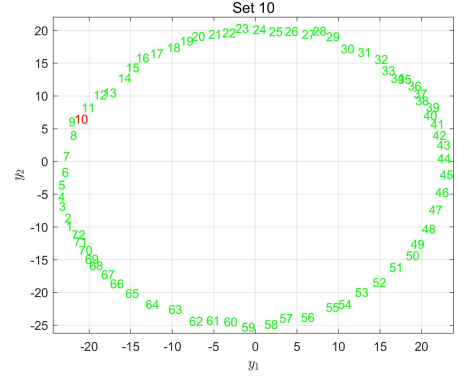
(a)



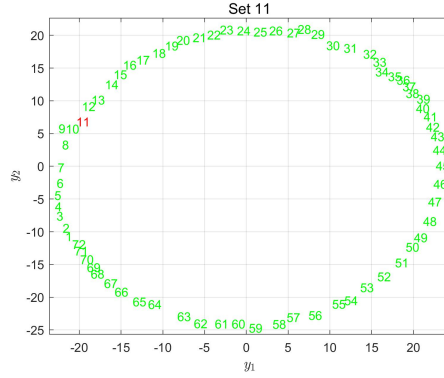
(b)



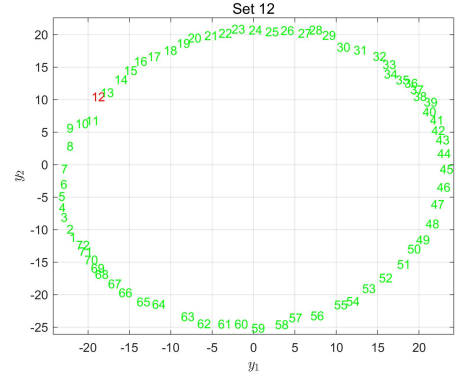
(c)



(d)



(e)



(f)

Figure 3.46: For experiment 3 on validation of predictability with leave one out sets composed of the first object of the COIL data set, proposed method outputs corresponding to (a) Set 7, (b) Set 8, (c) Set 9, (d) Set 10, (e) Set 11, and (f) Set 12.

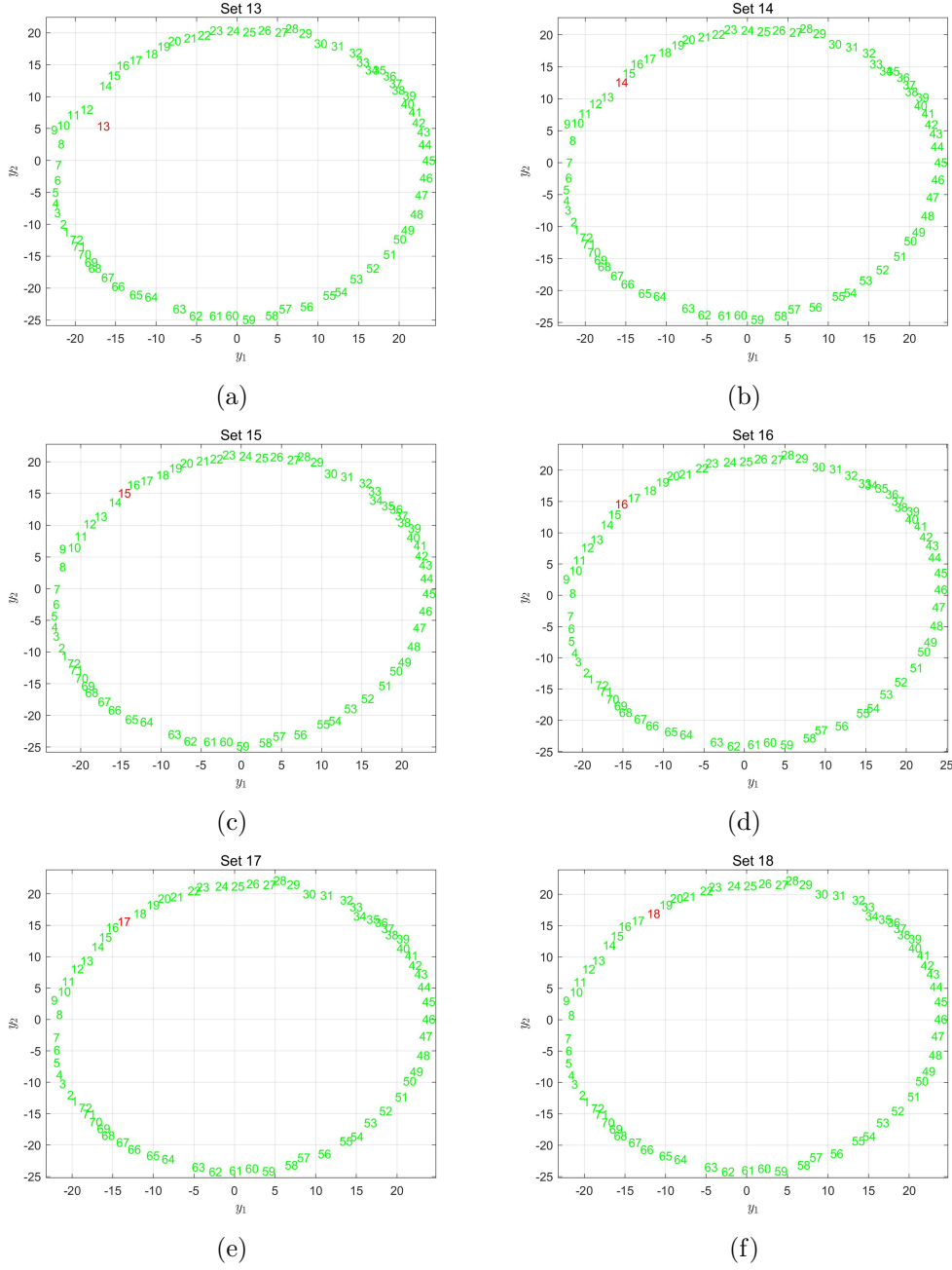


Figure 3.47: For experiment 3 on validation of predictability with leave one out sets composed of the first object of the COIL data set, proposed method outputs corresponding to (a) Set 13, (b) Set 14, (c) Set 15, (d) Set 16, (e) Set 17, and (f) Set 18.

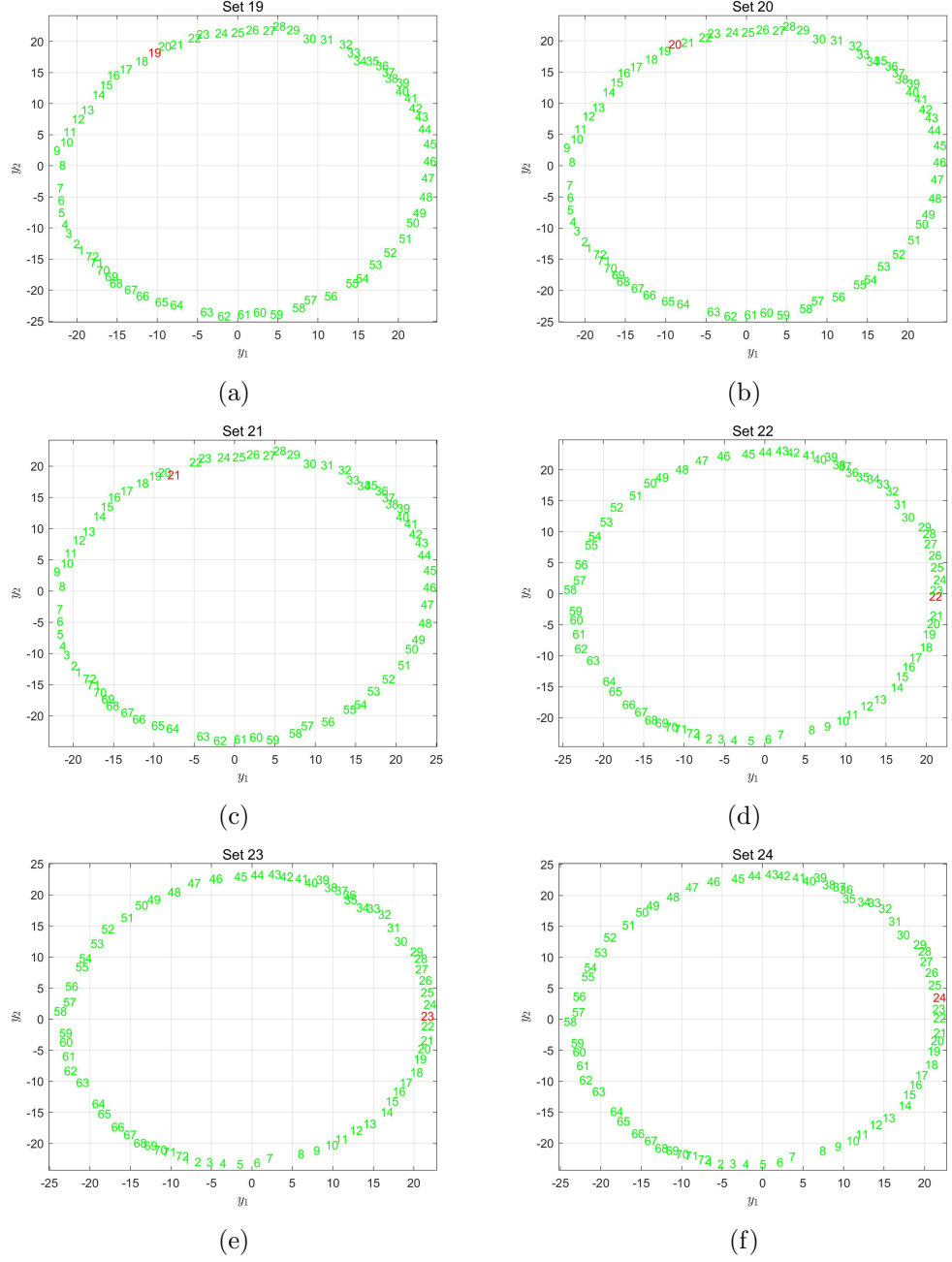


Figure 3.48: For experiment 3 on validation of predictability with leave one out sets composed of the first object of the COIL data set, proposed method outputs corresponding to (a) Set 19, (b) Set 20, (c) Set 21, (d) Set 22, (e) Set 23, and (f) Set 24.

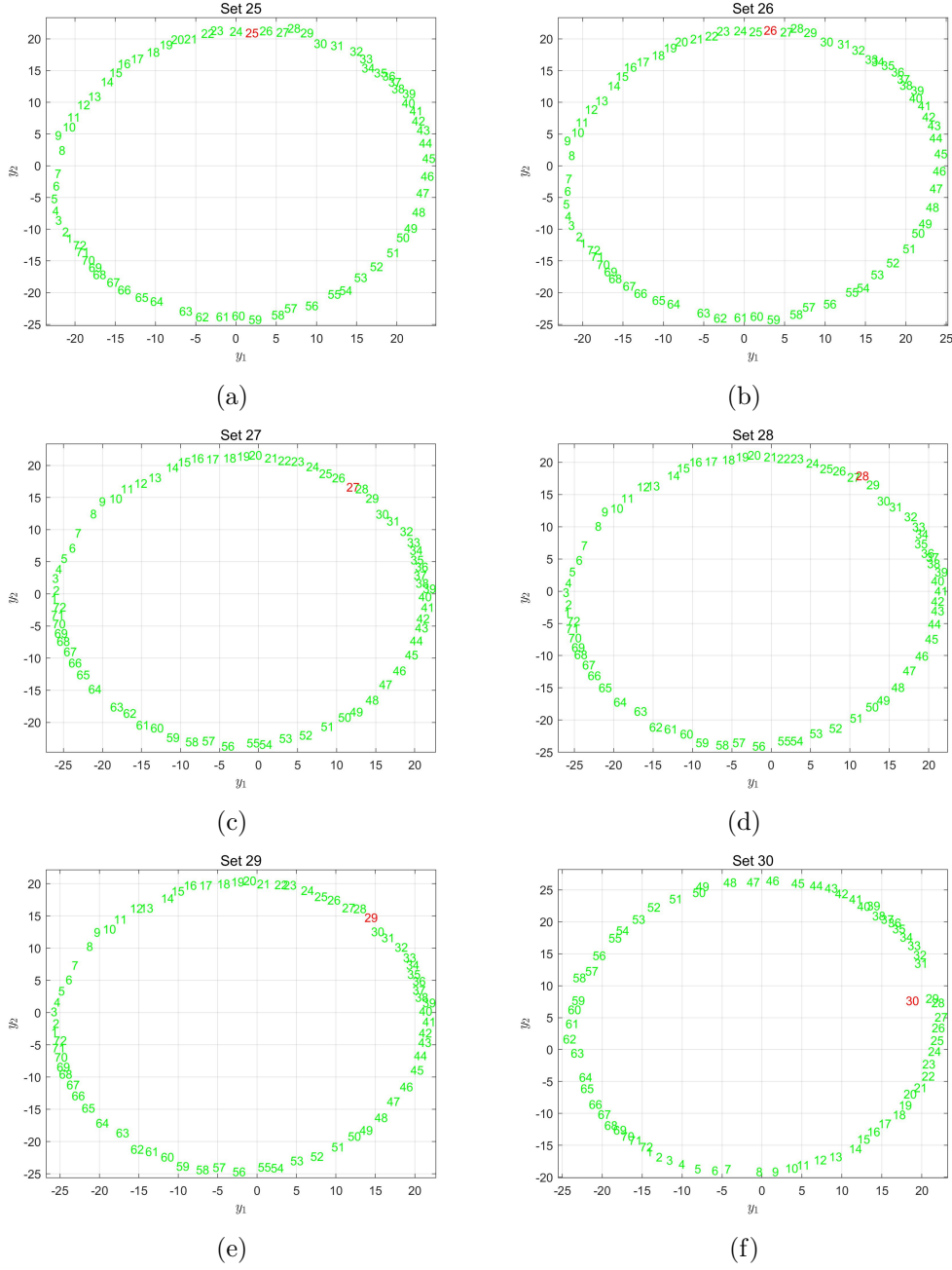
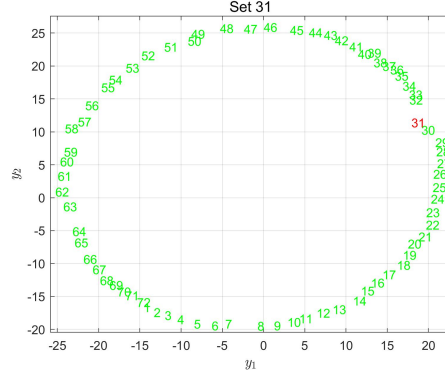
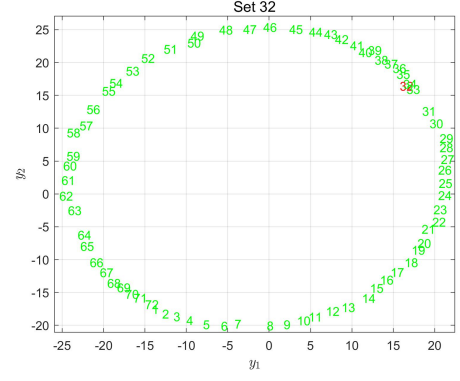


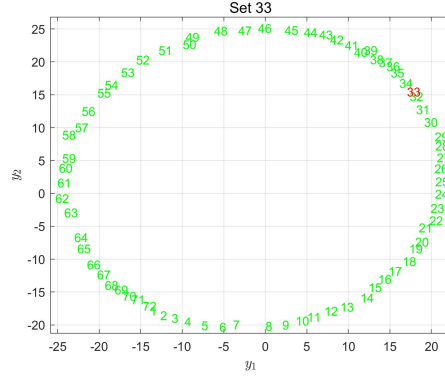
Figure 3.49: For experiment 3 on validation of predictability with leave one out sets composed of the first object of the COIL data set, proposed method outputs corresponding to (a) Set 25, (b) Set 26, (c) Set 27, (d) Set 28, (e) Set 29, and (f) Set 30.



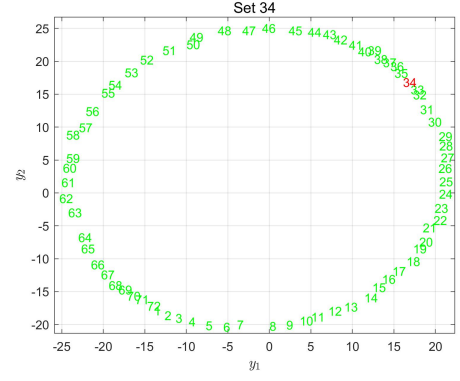
(a)



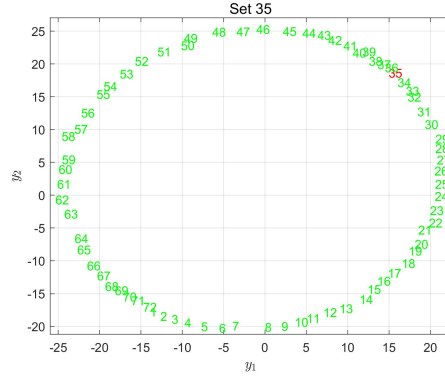
(b)



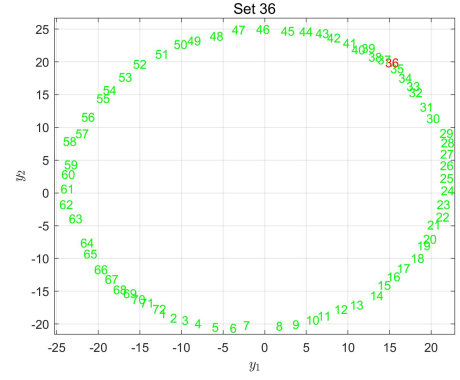
(c)



(d)



(e)



(f)

Figure 3.50: For experiment 3 on validation of predictability with leave one out sets composed of the first object of the COIL data set, proposed method outputs corresponding to (a) Set 31, (b) Set 32, (c) Set 33, (d) Set 34, (e) Set 35, and (f) Set 36.

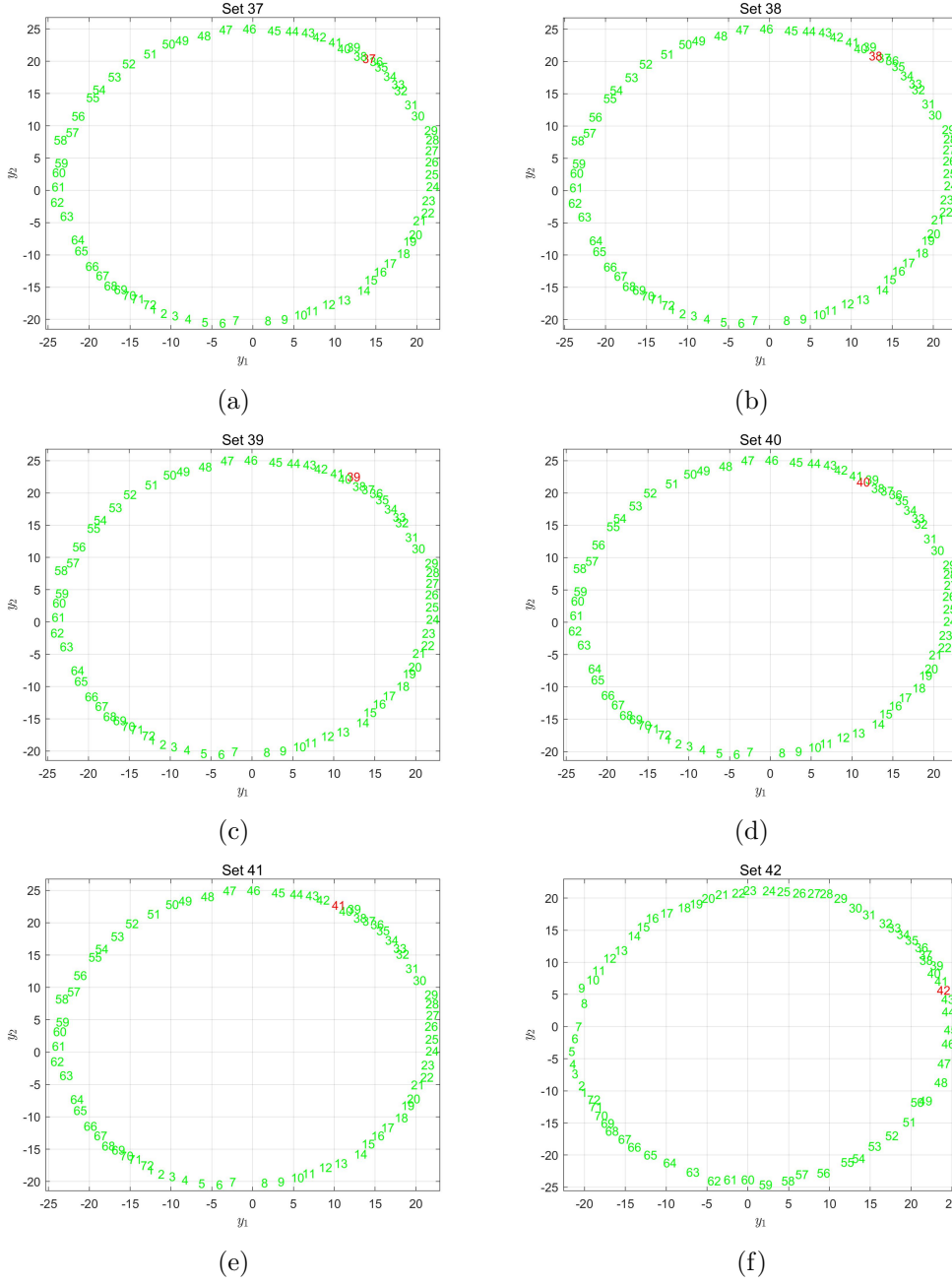
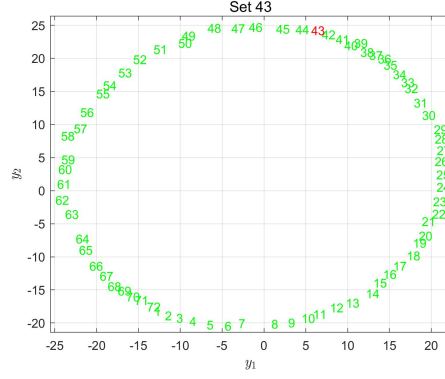
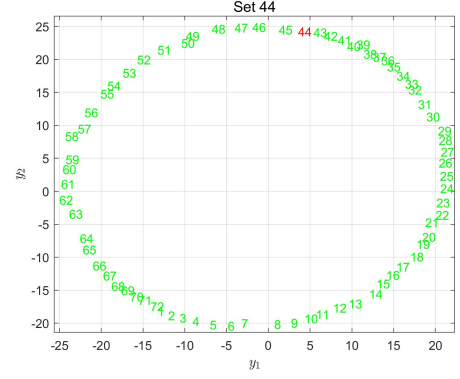


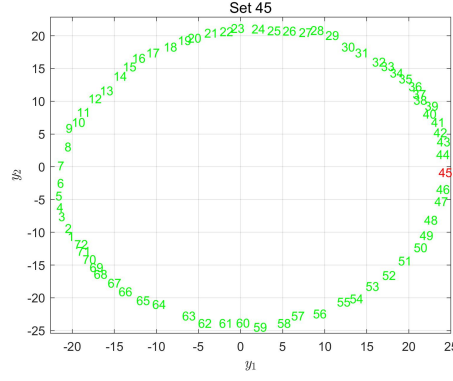
Figure 3.51: For experiment 3 on validation of predictability with leave one out sets composed of the first object of the COIL data set, proposed method outputs corresponding to (a) Set 37, (b) Set 38, (c) Set 39, (d) Set 40, (e) Set 41, and (f) Set 42.



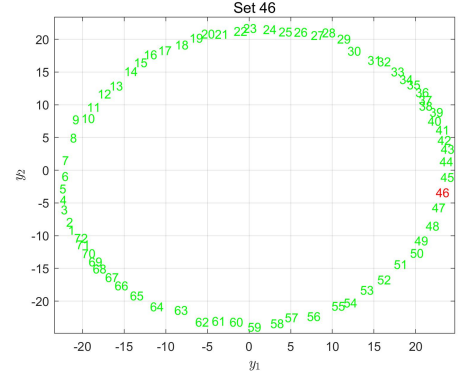
(a)



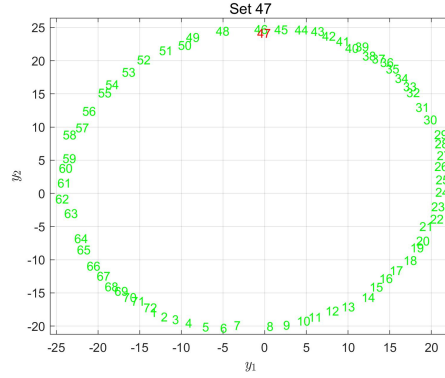
(b)



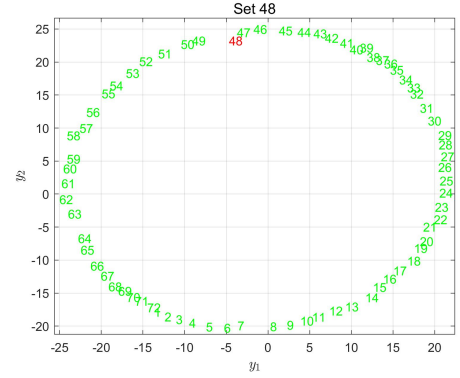
(c)



(d)



(e)



(f)

Figure 3.52: For experiment 3 on validation of predictability with leave one out sets composed of the first object of the COIL data set, proposed method outputs corresponding to (a) Set 43, (b) Set 44, (c) Set 45, (d) Set 46, (e) Set 47, and (f) Set 48.

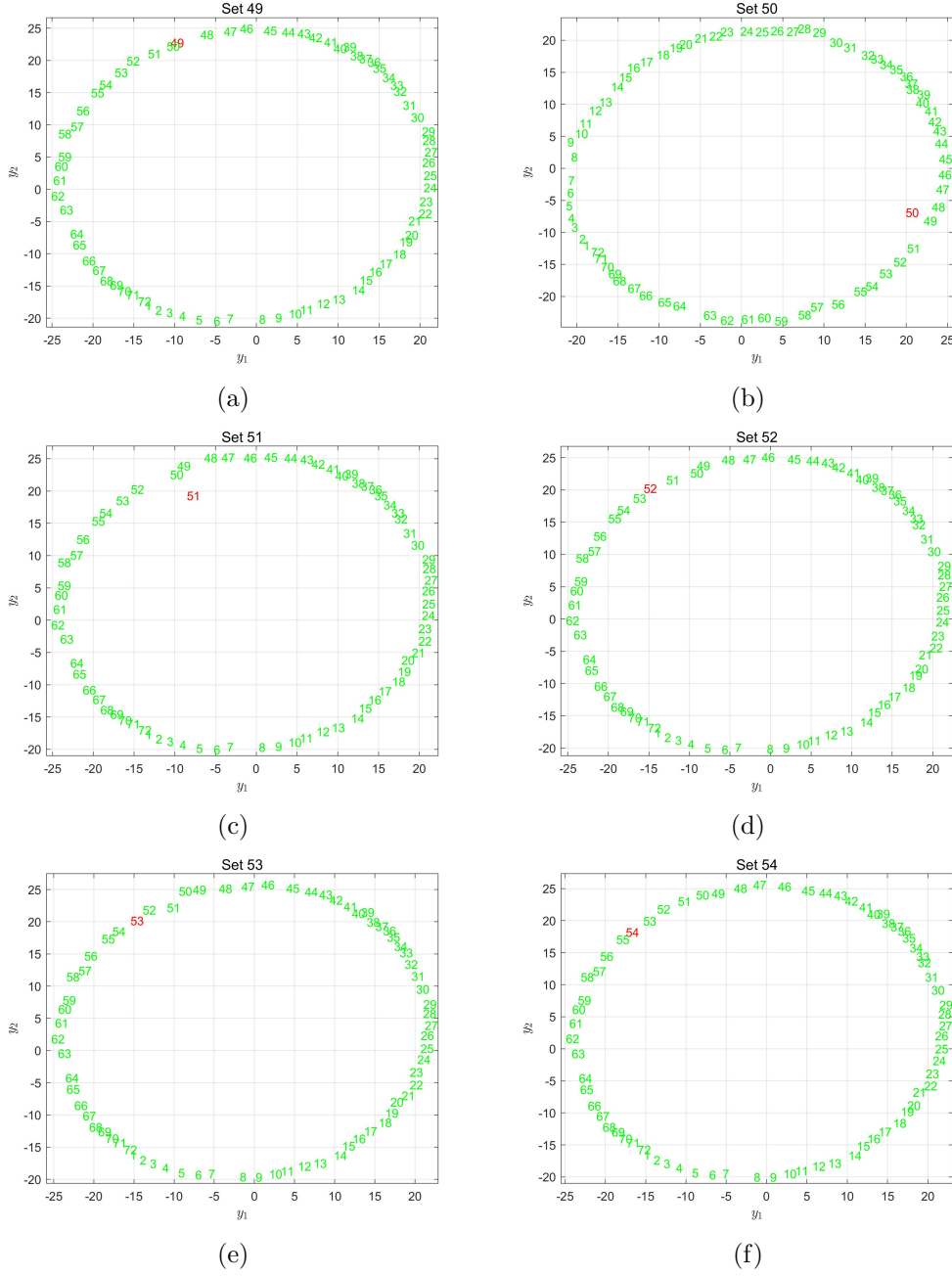
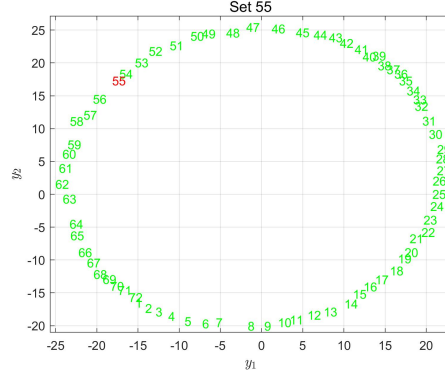
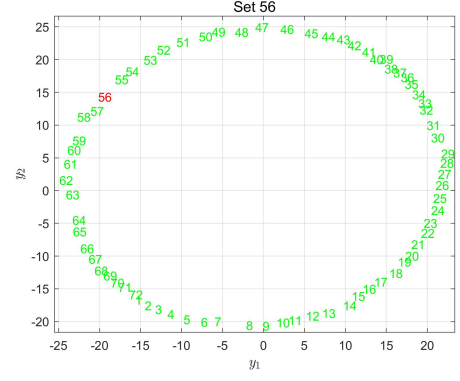


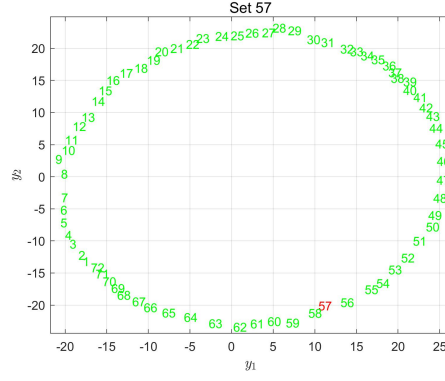
Figure 3.53: For experiment 3 on validation of predictability with leave one out sets composed of the first object of the COIL data set, proposed method outputs corresponding to (a) Set 49, (b) Set 50, (c) Set 51, (d) Set 52, (e) Set 53, and (f) Set 54.



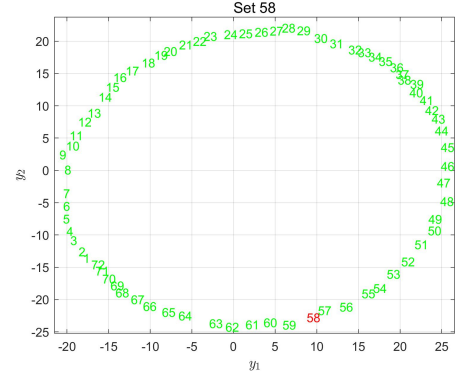
(a)



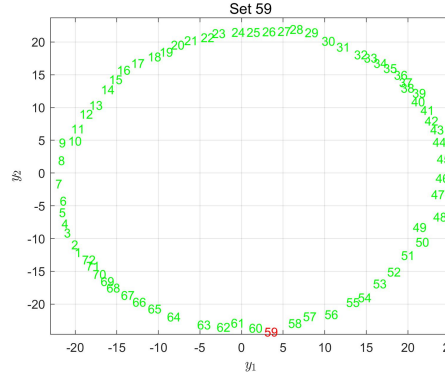
(b)



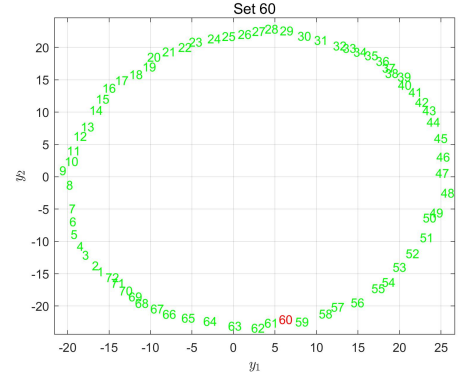
(c)



(d)



(e)



(f)

Figure 3.54: For experiment 3 on validation of predictability with leave one out sets composed of the first object of the COIL data set, proposed method outputs corresponding to (a) Set 55, (b) Set 56, (c) Set 57, (d) Set 58, (e) Set 59, and (f) Set 60.

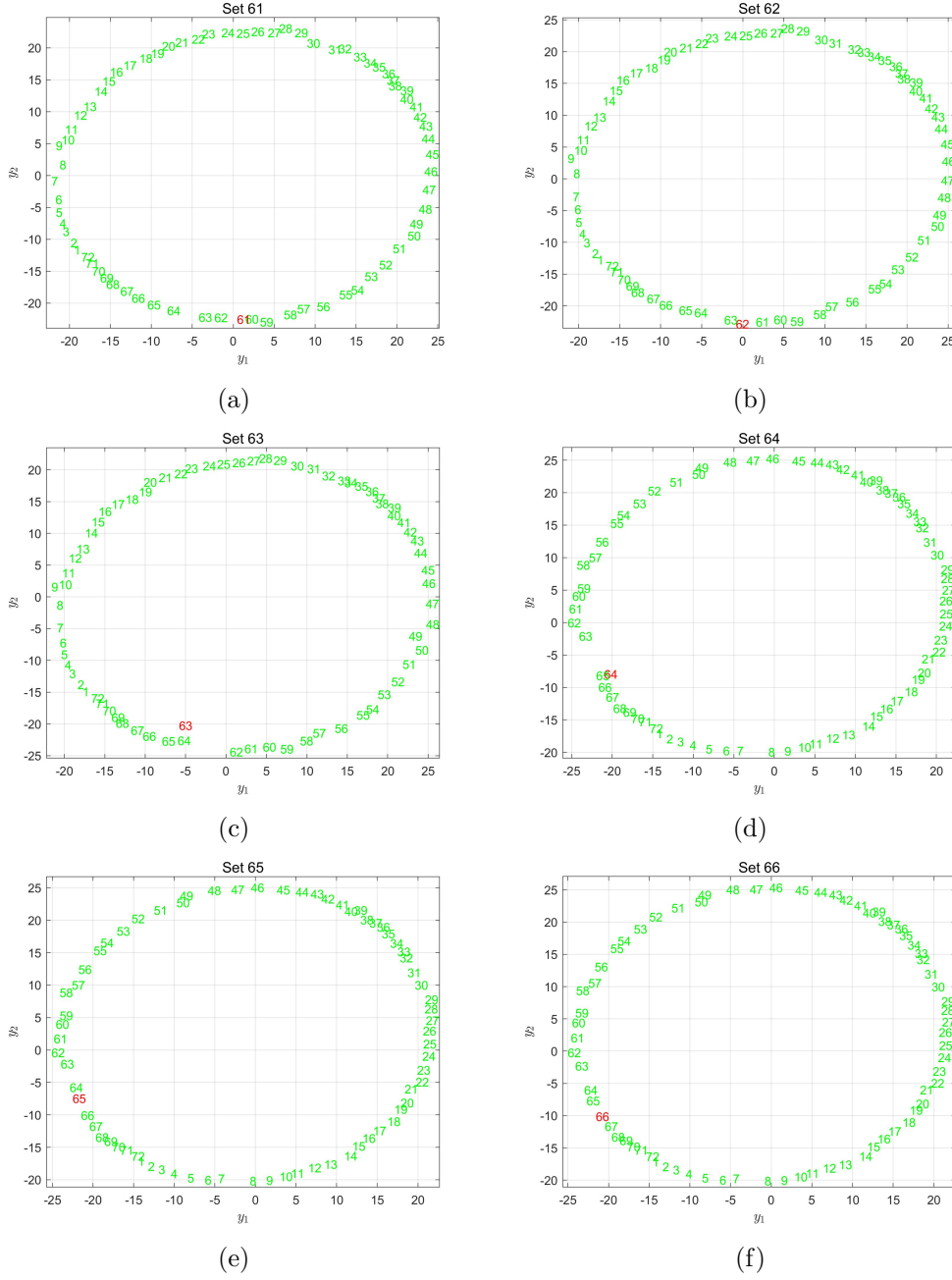


Figure 3.55: For experiment 3 on validation of predictability with leave one out sets composed of the first object of the COIL data set, proposed method outputs corresponding to (a) Set 61, (b) Set 62, (c) Set 63, (d) Set 64, (e) Set 65, and (f) Set 66.

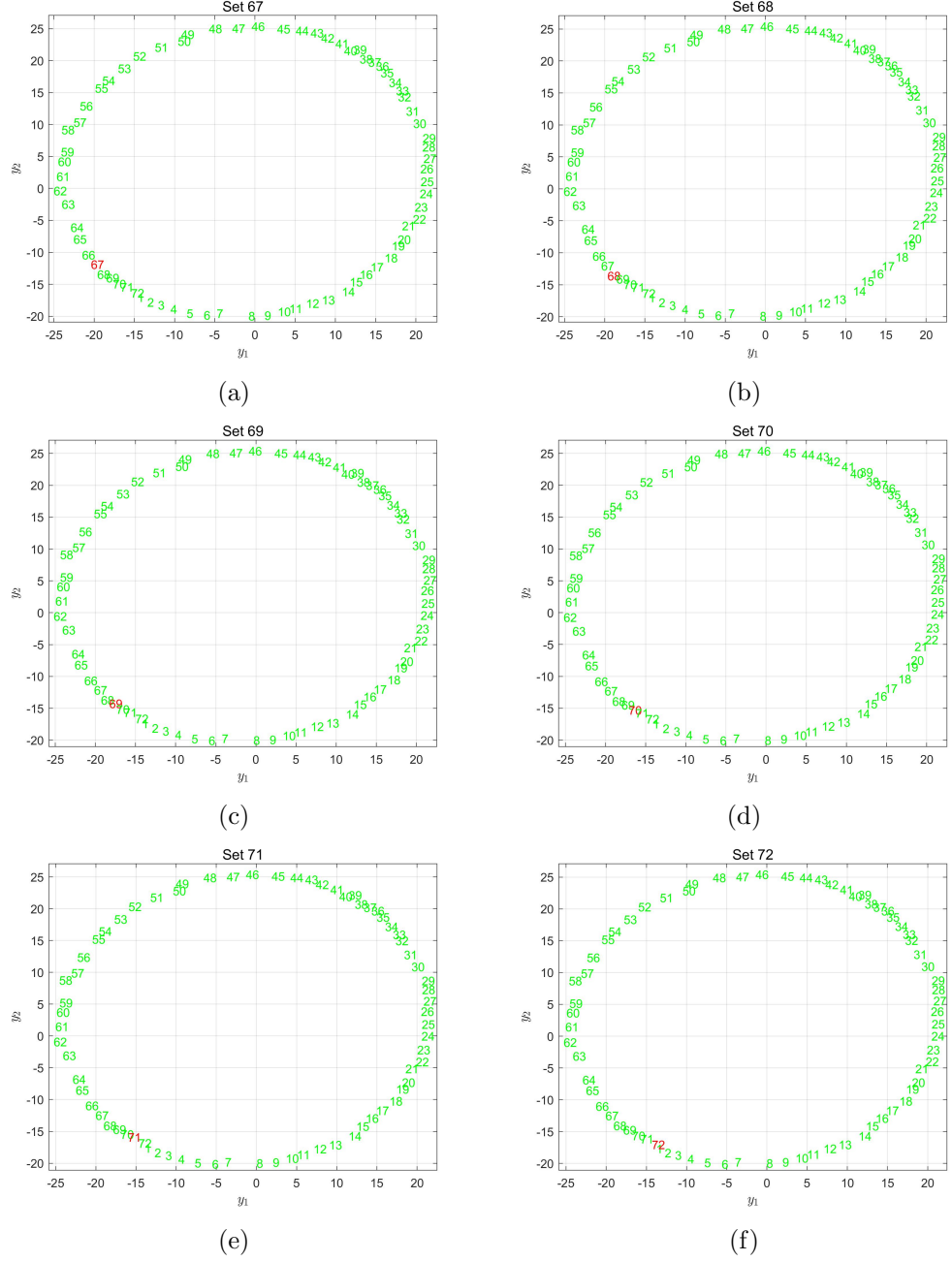


Figure 3.56: For experiment 3 on validation of predictability with leave one out sets composed of the first object of the COIL data set, proposed method outputs corresponding to (a) Set 67, (b) Set 68, (c) Set 69, (d) Set 70, (e) Set 71, and (f) Set 72.

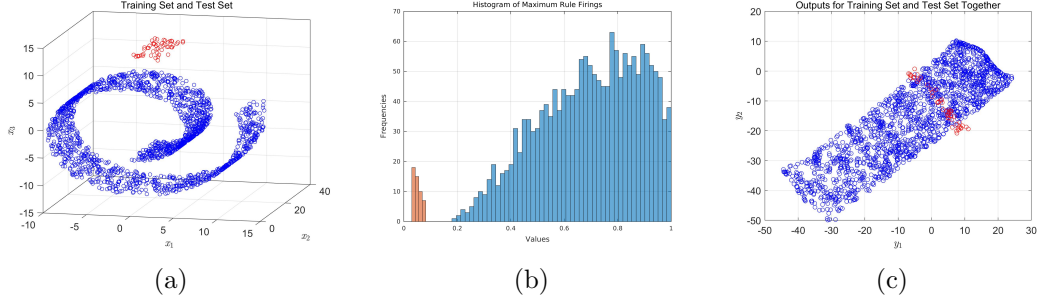


Figure 3.57: (a) Training Set (blue circles) and Test Set (red circles), (b) Histogram of maximum rule firings for Training Set (blue region) and Test Set (red region), and (c) Proposed method Output for Training Set (blue circles) and Test Set (red circles) without exercising rejection option.

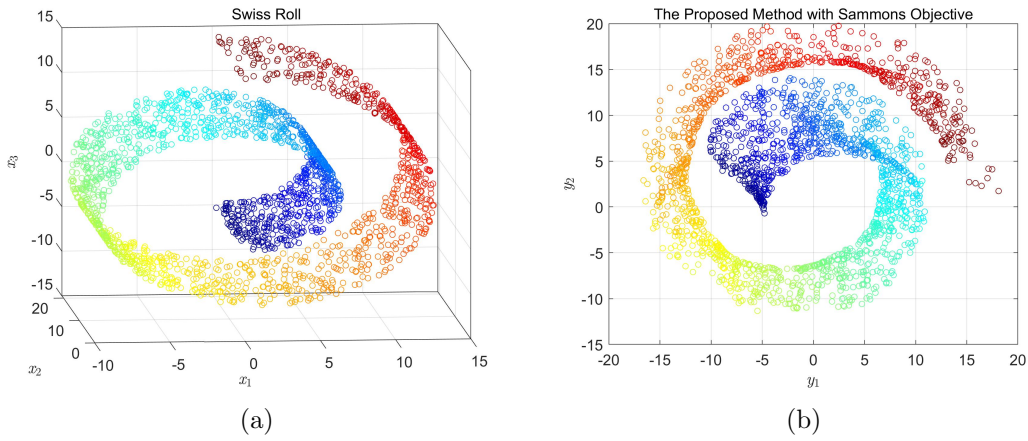


Figure 3.58: Visualization of Swiss Roll data with (a) original input space, and (b) proposed method with Sammon's Objective.

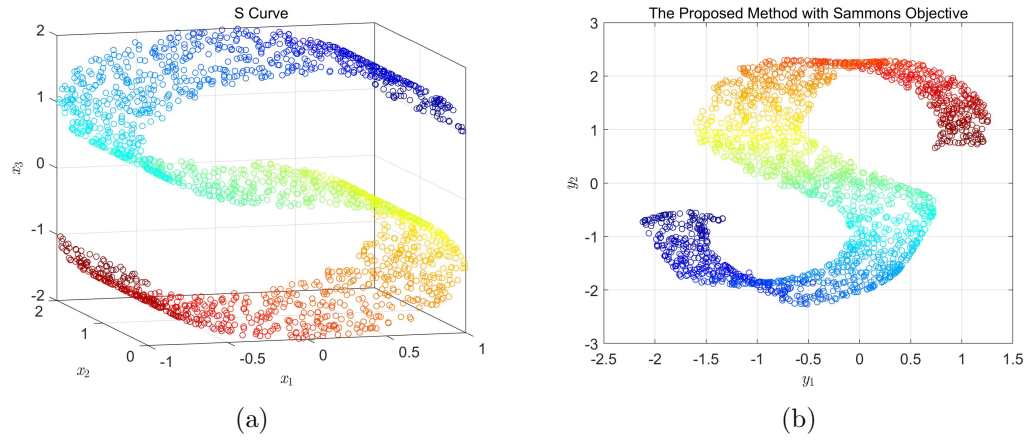


Figure 3.59: Visualization of S Curve data with (a) original input space, and (b) proposed method with Sammons's objective.

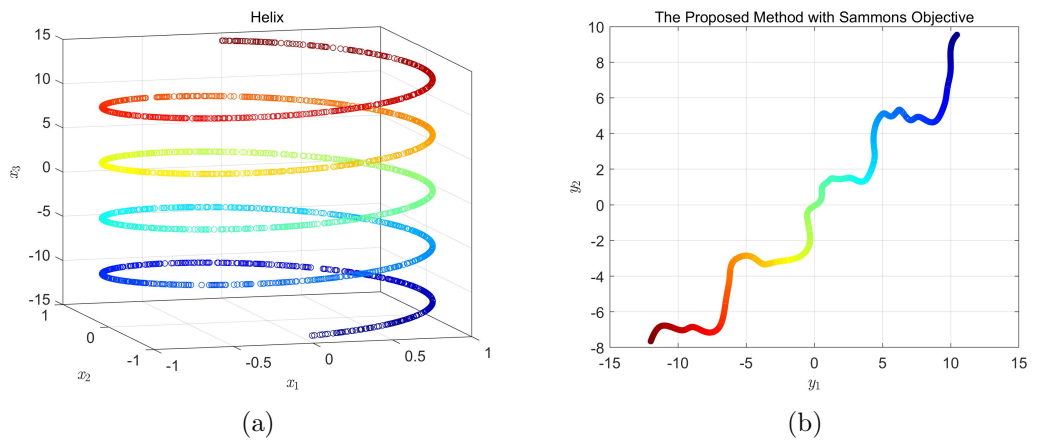


Figure 3.60: Visualization of Helix data with (a) original input space, and (b) proposed method with Sammons's objective.

points and that is an advantage over the usual Sammon’s method. We also note that unlike the fuzzy system in [29], the proposed system does not require any target output for learning.

3.4 Conclusion and Discussion

Fuzzy rule-based systems have been used for various machine learning tasks but unfortunately, FRBSs are almost unexplored for manifold learning or data visualization. Here, we have proposed a framework for unsupervised learning of a Takagi-Sugeno FRBS for the projection of high dimensional data. Without using any target data, the parameters of the FRBS are estimated so that the FRBS can project the high dimensional data preserving the local neighborhood structure.

Some advantages of the proposed scheme are listed next. (i) It is an unsupervised system, yet it learns a parametric system for explicit mapping of high dimensional data into a lower dimensional space. Thus, out-of-sample instances can be projected in a straightforward manner. (ii) It can refuse to project an out-of-sample instance, when it is far away from the sampling window of the training data, i.e, the system is equipped with an appropriate reject option without requiring the training data. Most machine learning systems cannot do so. (iii) The system is reasonably transparent as it is based on fuzzy reasoning, where each rule-antecedent represents a small hyper-ellipsoidal region in the input space and its consequent is a local linear function of the inputs. (iv) It provides a general framework that can use different objective functions. (v) The learning algorithm is found to be quite robust with respect to initialization. (vi) The scheme can be easily extended for Mamdani-Assilian model of fuzzy systems.

We have compared the performance of the proposed system with Sammon’s method, ISOMAP, LLE, t -SNE, UMAP, and Diffusion net on several data sets. The proposed system is the only one which performs consistently well on all data sets tested. The predictability of the proposed system is validated using three types of experiments involving both synthetic and real-world data sets.

3.4.1 Limitations and scope of improvement

Like several other data projection methods, both the time and space complexities of the proposed method are quadratic functions of the number of data points, n . So for a very large n , the computational overhead will be high. The use of GPUs can reduce the effective complexities significantly. Moreover, a proper sampling scheme [152] that produces a subset of the original data, which adequately represents the original data distribution may be used. The firing strength of a rule is computed as the product of the memberships of the atomic antecedent clauses in the rule. So for a very high-dimensional data, it is possible that for some instances no rule fires with a significant strength leading to an undesirable situation. However, this issue is true for *any* FRBS, it is not special to the system proposed here. Like several other manifold learning schemes, the proposed scheme is sensitive to the number of neighbors that is used to

construct the neighborhood graph for computing geodesic distances. In our study, the number of fuzzy rules has been chosen empirically. The choice of the number of fuzzy rules is related to the number of clusters we choose for clustering the training data. We did not explore the possibilities of using cluster validity indices or other schemes to decide on the optimal number of clusters as this was not the primary objective of this study. Moreover, a cluster validity index helps to find the ‘optimal’ number of clusters in the pattern recognition sense. But here even if the data do not have any cluster in the pattern recognition sense, we can use clustering to find the initial rule-base. Our experiments with the Swiss-Roll and other data sets suggest that the performance of the proposed method is not affected much by the choice of number of rules over a wide range of choices, hence we decided the number of rule empirically based on a limited number of experiments. We note here that unlike supervised methods, the cross-validation scheme cannot be used here. However, starting from an initial number of rules, we can increase the number of rules one by one and compute the average loss function over, say 5, repeats of the algorithm. The minimum number of rules for which the loss becomes less than a threshold, can be used as the number of rules.

The clustering algorithm we use, GCM, depends on the construction of a neighborhood graph using the input data points based on the Euclidean distance. For very high dimensional data, the GCM algorithm may suffer from the curse of dimensionality as Euclidean distance does not behave in the desirable manner for a very high dimensional space [196]. However, it is worth noting that methods like LLE, Laplacian Eigenmap, and ISOMAP, which use neighborhoods defined using Euclidean distance also suffer from the same problem for high dimensional data. For non-image data, a possible approach to address this problem is to project the data onto a space having moderate dimension using an unsupervised feature selection scheme or using a parametric feature extraction scheme that does not influence much the intrinsic manifold structure but eliminates redundant features or information. For image data, if the size of the image is big, it will also result in high dimensional inputs. But here we cannot use feature (pixel) selection as that may not be effective. However, we may use feature embedding techniques like an auto-encoder to reduce the size of the image. Then we can develop the proposed fuzzy rule-based system in the latent space / reduced space. We have restricted our work to data sets lying on a single manifold. Through neighborhood graph construction we can only approximate the geodesic distance over a single manifold or overlapping manifolds. But if the data lie in two (or more) widely separated manifolds, using a reasonable number of nearest neighbors we may not get a connected neighborhood graph. We have not proposed any technique to approximate geodesic distance over a multi-component neighborhood graph. These issues are kept for future investigation.

Chapter 4

Feature selection simultaneously preserving both class and cluster structures [4]

4.1 Introduction

In Chapters 2 and 3, we have explored feature extraction based dimensionality reduction methods. The other category of methods that reduce data dimension is feature selection methods. Feature selection approaches are preferable to feature extraction approaches when building a comprehensible system. As we have already mentioned in Subsec. 1.1.2, in feature selection the projected space maintains the identity of the original features and their semantics, if any, and hence can make the underlying system comprehensible [197]. In this chapter (Chapter 4) and in the next chapter (Chapter 5), we have explored feature selection schemes. In Chapters 2 and 3, we have dealt with input data both with and without target outputs, and the intended task was visualization. In the current chapter (Chapter 4) and the next chapter (Chapter 5), we deal with labeled data, specifically, data having class labels. Naturally, classification is the primary task for the feature selection schemes in Chapter 4 and Chapter 5. However, the feature selection schemes introduced in these two chapters set some additional goals along with selecting class discriminative features. Feature selection methods can be broadly classified on the basis of the utilization of class label information. There are three categories: supervised, semi-supervised and unsupervised [197, 198]. The supervised feature selection method exploits the label information to find out the relevant features which can distinguish samples of different classes [199, 200]. Semi-supervised feature selection is used when some labeled samples along with plenty of unlabelled samples are present [201, 202]. Both labeled and unlabelled data are used to modify a hypothesis obtained from the labeled data [203]. Unsupervised feature selection is much more difficult as it needs to find out the useful features in the absence of any label or target information [197, 204]. Different criteria have been chosen to select a subset of original features in different unsupervised feature selection studies. Some of them are:

preserving the data distribution such as manifold structure [205,206], preserving cluster structure [207,208], and preserving data similarity [209,210]. It is noteworthy that in the case of unsupervised feature selection, some methods try to preserve the “structure” or “geometry” of the data in some sense. Contrarily supervised feature selection methods in most cases do not set any explicit criteria to preserve the structure of the data. They only pay heed to separating the classes as much as possible with different measures exploiting class information such as Fisher score [51,211], Laplacian score [212], mutual information [52,53], normalized mutual information [53–56], ReliefF [213], class correlation [214], and classifier score [215]. We should note here that the feature selection criteria are not always led by a single objective. Feature selection methods often follow a criterion that consists of two or more objectives. The study in [204] proposes a criterion named ‘maximum projection and minimum redundancy’ which is governed by two goals: projecting data into a feature subspace with minimum reconstruction error and minimum redundancy. The studies in [216–219] claim that both global structure and local structure should be preserved in the projected space as both of them may carry important discriminating information and hence, they have proposed feature selection schemes that focus both on global and local structure preservation. The investigation in [220] claims to preserve dual global structures. Going through various feature selection schemes having multiple objectives we found that whenever class label is available, no work in feature selection explicitly focused preserving structural information along with class information although both of these are important discriminative information and may have positive impact on the generalization ability of the classifier. Suppose, for a data set, the class and cluster structures are substantially different. Exploiting only the class labels, it may not be possible to preserve the cluster structures in the projected space. For a practical system, even when the primary task is classification, we may need to cluster the samples in the space defined by the selected features. For example, fuzzy rule based classifiers are often designed by clustering the training data for each class and translating each cluster into a rule [30,33,59]. We could not find any feature selection method that focuses both on class and cluster separability together. To bridge this gap, in this study we propose a feature selection method that selects features preserving class and cluster-structure information simultaneously. We employ a multi-layer perceptron (MLP) based neural network to develop an embedded feature selection scheme. The training of the proposed MLP based feature selection method is governed by both class discriminating and cluster (structure) preserving objectives. The philosophy is quite general and can be easily extended to other networks such as radial basis function network.

4.2 Proposed Method

Let us denote the input data by an $n \times P$ matrix, $\mathbf{X} = \{\mathbf{x}_i \in \mathbb{R}^P\}_{i=1}^n$. Here, $\mathbf{x}_i$ is a P dimensional row vector of the form, $\mathbf{x}_i = (x_{i1}, x_{i2}, \dots, x_{iP})$. The collection of class labels of $\mathbf{X}$ be $\mathbf{Z} = \{z_i \in \{1, 2, \dots, C\}\}_{i=1}^n$, where, z_i is the class label corresponding to $\mathbf{x}_i$. We aim to select a subset of size Q from the original set of features such

that the selected subset performs reasonably well in terms of the classification task as well as in clustering. In other words, if we design a classifier using the selected features, the performance of the classifier would be comparable to a classifier designed using all features. Similarly, if we cluster the data in the reduced dimension as well as in the original dimension we expect to get a similar partition matrix. Here, we propose a neural network-based framework to select features. Neural networks have been explored for feature selection [221–223] as well as for classification [224–227]. In our proposed model the neural network simultaneously selects features and learns a classifier as we follow an embedded method for feature selection. Moreover, our proposed network preserves structural information and class discriminatory information simultaneously, whereas, the feature selection networks in [221, 223] solve classification problems, consider class label information in their loss function but not any structural information. Note here, the work in [222] considers a system identification problem. To build the neural network-based embedded feature selector, we employ the multi-layer perceptron (MLP) based framework used in [25–27]. The basic framework is shown in Fig. 4.1. As seen in Figure 4.1, preceding the input layer of the MLP, there is a

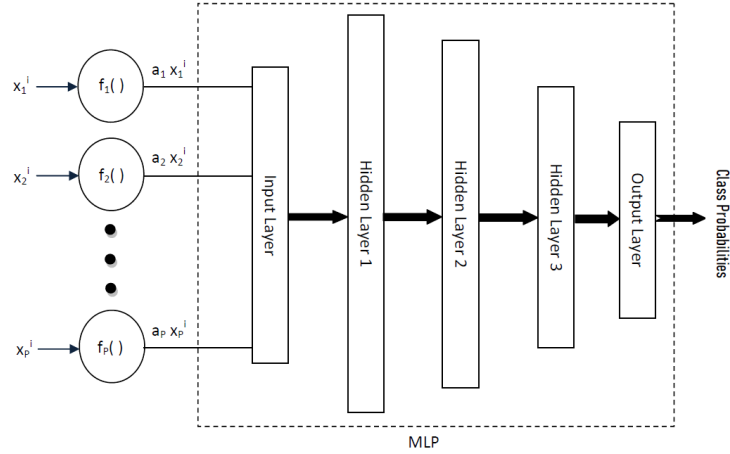


Figure 4.1: Proposed network for feature selection

layer consisting of P nodes. Before entering to the input layer of the MLP, the j^{th} feature passes through the node $f_j(\cdot)$. These nodes act as attenuating gates that allow or block features from contributing to the output of the neural network effectively. For the i^{th} instance, its j^{th} feature x_{ij} on passing the gate node $f_j(\cdot)$ becomes $a_j x_{ij}$; i.e., $f_j(x_{ij}) = a_j x_{ij}$. The parameters a_j s are to be learned for feature selection. In MLP, a weighted sum of the values available at the input nodes is applied to the hidden nodes of the first hidden layer. Zero value at an input node implies that the corresponding feature is not considered. When training of the MLP-based framework is complete, a_j s for the selected features become close to 1, effectively allowing them to contribute to the classifier. Whereas, for poor or rejected features a_j s become close

to 0, effectively making them not contribute to the classifier. In [25–27], this framework was explored for classification-oriented feature selection, group feature selection, and redundancy-controlled feature selection. Here, we explore this framework for simultaneous structure-preserving and class-discriminating feature selection. Next, we elaborate on the MLP-based framework and the proposed objective functions to train the network.

We denote the P nodes before the input layer of the MLP as $f_j()$ s for $j = 1, 2, \dots, P$ where $f_j()$ is a gate or modulator function applied on the j^{th} feature, x_j . Now, we have to design $f_j()$ in such a way,

$$f_j(x_j) = a_j x_j = \begin{cases} x_j & \text{if } x_j \text{ is a useful feature.} \\ 0 & \text{otherwise.} \end{cases} \quad (4.2.1)$$

In our framework, the factor, a_j is learnable. We implement a_j as a smooth continuous function, $a_j = \exp(-\lambda_j^2)$. Clearly, when $\lambda_j = 0$, the value of $\exp(-\lambda_j^2) = 1$ and when $\lambda_j \rightarrow \pm\infty$, the value of $\exp(-\lambda_j^2) = 0$. By adding suitable regularizer terms to the objective function we design our learning system in such a way that, over the learning process, the gate parameters, λ_j s for useful features drop close to zero and that for derogatory or indifferent features rise to high values. So, in our learning system, the learnable parameters are λ_j s and the neural network weights are learned together, i.e., the loss function is minimized with respect to both λ_j s and the neural network weights.

Now, we have to define a suitable loss function for selecting features along with learning the embedded classifier. Our aim is to select features that are reasonably good for classification as well as for clustering. To satisfy this requirement, we take the loss function as a combination of two losses E_{class} and E_{struct} . E_{class} is considered for learning the classes and E_{struct} is considered for preserving structural information. At this moment let us consider the network for selecting features for efficient classification only. A suitable loss function to impose class discrimination is the cross-entropy loss [228]. We define, E_{class} as the cross-entropy loss involving actual and predicted class labels.

$$E_{class} = -\frac{1}{n} \sum_{i=1}^n \sum_{k=1}^C t_k^i \log(p_k(\mathbf{x}_i)) \quad (4.2.2)$$

Here, t_k^i is k^{th} element of the one-hot encoded label of the sample $\mathbf{x}_i$ or in other words, t_k^i is the k^{th} element of the vector $\mathbf{t}^i \in \{0, 1\}^C$ such that

$$t_k^i = \begin{cases} 1 & \text{if } k = z_i \\ 0 & \text{otherwise} \end{cases} \quad (4.2.3)$$

In (4.2.2), $p_k(\mathbf{x}_i)$ is the predicted probability (by the MLP classifier) of $\mathbf{x}_i$ being in the k^{th} class. As already discussed above, for effective feature selection, the magnitude of λ_j s for the selected features should drop to almost zero and for rejected features should

rise to high values. To ensure this condition we add the following regularizer.

$$\begin{aligned} E_{select} &= \frac{1}{P} \sum_{j=1}^P a_j(1 - a_j) \\ &= \frac{1}{P} \sum_{j=1}^P \exp(-\lambda_j^2)(1 - \exp(-\lambda_j^2)) \end{aligned} \quad (4.2.4)$$

In a feature selection framework, a constraint for selecting a fixed number of features is necessary. The following regularizer tries to keep the number of the selected features close or equal to Q .

$$E_Q = \frac{1}{Q^2} \left\{ \left(\sum_{j=1}^P a_j \right) - Q \right\}^2 = \frac{1}{Q^2} \left\{ \left(\sum_{j=1}^P \exp(-\lambda_j^2) \right) - Q \right\}^2 \quad (4.2.5)$$

So, the overall loss function for the selection of features with our framework for classification purposes is the following.

$$E = E_{class} + \alpha_1 E_{select} + \alpha_2 E_Q \quad (4.2.6)$$

Here, $\alpha_1 \geq 0, \alpha_2 \geq 0$ are scalar multipliers for adjusting the importance of E_{select} and E_Q in the overall error function E . Note that E_{class} is also dependent on λ_j s as the attenuated features get into the network.

Now let us focus on our original agenda of selecting features that perform satisfactorily both for classification and clustering. To preserve structural information of the data in the lower dimensional space formed by the selected Q features, we consider Sammon's stress [41] as a loss function. The Sammon's stress is the loss function for a non-linear mapping named Sammon's mapping that is able to capture complex non-linear structures in data, as a result, also preserves cluster structure. The lower the value of Sammon's stress, the better the lower dimensional representations in capturing the original inter-point distances or structures of the original data. We can define Sammon's stress involving the original input space and selected feature space as the following.

$$E_{sammons} = \frac{1}{\left(\sum_{i,l=1}^n d_{il}^{\mathbf{X}} \right)} \sum_{i=1}^{n-1} \sum_{l=i+1}^n \frac{\left(d_{il}^{\mathbf{X}} - d_{il}^{\hat{\mathbf{X}}} \right)^2}{d_{il}^{\mathbf{X}}} \quad (4.2.7)$$

$d_{il}^{\mathbf{X}}$ is the distance between $\mathbf{x}_i$ and $\mathbf{x}_l$. $\hat{\mathbf{X}} = \{\hat{\mathbf{x}}_i = (a_1 x_{i1}, a_2 x_{i2}, \dots, a_P x_{iP})^T \in \mathbb{R}^P\}_{i=1}^n$. So, $d_{il}^{\hat{\mathbf{X}}}$ is the distance between $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{x}}_l$. As discussed earlier, at the end of the training of our embedded system, a_j s will be close to 0 or 1 depending on whether the corresponding features are rejected or selected. Therefore, for a trained system $d_{il}^{\hat{\mathbf{X}}}$ would signify the distance between i^{th} and l^{th} instances in the latent space formed by the implicitly selected Q features. So considering $E_{sammons}$ in Equation (4.2.7) as an regularizer, the resultant overall loss function is given by.

$$E_{tot} = E_{class} + \beta E_{sammons} + \alpha_1 E_{select} + \alpha_2 E_Q \quad (4.2.8)$$

$\beta \geq 0$ is a scalar multiplier that controls the trade-off between the class information and the structural information in the feature selection process. Note that, the computational complexity for the loss function in Equation (4.2.7) is $O(n^2)$. For a large n , the overhead of computing Equation (4.2.7) and hence Equation (4.2.8) is intensive. As the weight update at each iteration will involve computing Equation (4.2.7), the overall computation cost would be high. For small and moderate n , we use Equation (4.2.8) as the loss function to be minimized. However, for large n to avoid the high computational cost we modify Equation (4.2.7) as follows.

$$E_{struct} = \frac{1}{(\sum_{\mathbf{x}_i, \mathbf{x}_l \in S_t} d_{il}^{\mathbf{X}})} \sum_{\mathbf{x}_i \in S_t} \sum_{\mathbf{x}_l \in S_t; \mathbf{x}_l \neq \mathbf{x}_i} \frac{(d_{il}^{\mathbf{X}} - d_{il}^{\hat{\mathbf{X}}})^2}{d_{il}^{\mathbf{X}}} \quad (4.2.9)$$

Here S_t is a randomly selected subset of $\mathbf{X}$ at the t^{th} iteration. Different S_t s are chosen at different iterations and hence different sets of inter-point distances are preserved. Since the considered MLP is trained over a large number of iterations, the use of Equation (4.2.9) is expected to result in almost the same effect as that by Equation (4.2.7). We have to choose $|S_t|$ such that Equation (4.2.9) is computationally manageable and at the same time it should be large enough to make E_{struct} an effective substitute of $E_{sammons}$. Adding Equation (4.2.9) to Equation (4.2.6) we propose the following loss function for our system.

$$E_{tot} = E_{class} + \beta E_{struct} + \alpha_1 E_{select} + \alpha_2 E_Q \quad (4.2.10)$$

E_{tot} is minimized with respect to the gate parameters λ_j s and the weights of the network to find their optimal values. In each iteration, we update P gate parameters and w weights, where w is the total number of weights of the MLP. Computational complexity for training the MLP is $O(n(P + w))$ [10].

4.3 Experimentation and Results

The feature selection framework proposed in this chapter is generic but it can be adapted to solve specialized problems. We have studied the proposed framework for general datasets as well as for solving a special problem: band selection of hyperspectral images. We present the results of band selection for HSIs in a different subsection, Subsec. 4.3.2. We present the results of feature selection for the conventional classification problem in the following subsection (Subsec. 4.3.1).

4.3.1 Feature selection for conventional classification datasets

We have used five publicly available datasets that are very commonly used for classification and clustering. The first four datasets are downloaded from UCI machine learning repository [229]. AR10P is downloaded from the open-source feature selection repository of Arizona State University [230]. The details of the number of features,

Table 4.1: Summary of datasets.

Name	Number of features P	Number of classes C	Number of instances n
E. coli	7	8	336
Glass	9	6	214
Ionosphere	34	2	351
Sonar	60	2	208
AR10P	2400	10	130

number of classes, and number of instances for the five datasets are summarized in Table 4.1. The datasets are used directly without any pre-processing. The datasets are partitioned into training and test sets as approximately 90% and 10% of the total number of instances, respectively. To implement our proposed feature selection scheme we use the neural network shown in Fig. 4.1 with the number of hidden layers, $n_H = 1$. The input and output layers have P and C nodes respectively, where P is the number of features and C is the number of classes corresponding to the considered dataset. The number of hidden nodes in the hidden layer is 8 (20 for AR10P data set). To get stable feature selection results, the network weights are initialized in a certain way. To set the initial weights of the proposed network, we take the following steps. First, we consider the usual MLP part of our network (i.e. without feature selection), depicted by the portion within the dotted rectangle in Fig. 4.1, and initialize its weights randomly. Next, we train the usual MLP with the cross-entropy loss defined in Equation (4.2.2) with the training set until convergence. The weights of the converged network are used as the initial weights of the proposed network. The gate parameter λ_j s are initialized with values drawn randomly from a normal distribution with mean = 2 and spread = $\frac{1}{\sqrt{P}}$. The initial values of λ_j s are chosen around 2 to effectively make the gates almost closed initially. As the learning progresses the λ_j s are updated in a way to allow the useful features to affect the network. For the proposed system, to select a subset of Q features, the gate parameters λ_j s are sorted in ascending order after the training is over, and the Q features corresponding to the top Q , λ_j s are selected. The network weights as well as the gate parameters λ_j s are learned using the adaptive gradient algorithm, ‘*train.AdagradOptimizer*’ routine of the ‘*TensorFlow*’ framework [193]. For all experiments, both α_1 and α_2 of the error functions in Equations (4.2.6) and (4.2.8) are set as 1. The total number of iterations for training the network is set to 20000.

The five datasets we consider here, have the number of instances $n < 400$, which is not so large. Therefore, we use (4.2.8) as the overall loss function to train the MLP based architecture for selecting features that are reasonably good for clustering and classification. When $\beta = 0$ in (4.2.8), effectively, the error function that governs the learning of our MLP based embedded feature selection scheme is (4.2.6). The corresponding feature selection scheme now only considers classification. Let us name this method as feature selection with MLP (FSMLP). When $\beta \neq 0$ in (4.2.8), our method takes structure preservation into account along with classification. Let us

name the corresponding method as FSMLP_{struct}. To understand the importance of adding the structure preserving regularizer (4.2.7), we perform feature selection with FSMLP and compare with FSMLP_{struct} having different β values. We explore three values of β s 0.1, 1, and 10. Although the exact value of the β that is optimum for a particular dataset for a particular number of selected features Q cannot be decided from these three values, we investigate the effect of three widely different β s to see the role of the weight to the structure preserving regularizer, i.e. β on the performance of the selected features. We compare with three other methods namely, Independent Component Analysis (ICA)-based feature selection [231], F-score based filter method [51], and mutual information based filter method [52, 53]. The performance of both FSMLP and FSMLP_{struct} is dependent on the initial weights of the network. So, we repeat the initialization of the network weights and gate parameters λ_j s five times and run each of the schemes, FSMLP and FSMLP_{struct} five times with the five initializations. To check the effectiveness of the methods in selecting features that perform well in classification and clustering simultaneously, we compute the classification scores of the support vector machine (SVM) and MLP classifiers as well as several structure-preserving indices: Sammon's stress (SS) [41], normalized mutual information (NMI) [54–56], adjusted rand index (ARI) [232], and Jaccard Index (JI) [233]. As the measure of classification performance, we use the overall classification accuracy (OCA) for both the classifiers. The optimal hyper-parameters of SVM are determined through five-fold cross-validation using grid search. For MLPs, we use the same number of hidden nodes as used by the FSMLP or FSMLP_{struct}. Note that here the test set is not only unseen to the SVM classifier but unseen to the feature selection methods also. SS, defined in Equation (4.2.7) uses the original inter-point distances $d_{il}^{\mathbf{X}}$ s and latent space inter-point distances $d_{il}^{\tilde{\mathbf{X}}}$ s. Here to compute $d_{il}^{\tilde{\mathbf{X}}}$, we use the lower dimensional data formed by the selected Q features. We use NMI, ARI, and JI as the structure-preserving performance metrics by supplying the cluster labels obtained from clustering the data in the original space (using all features) as the true label and the cluster labels obtained from clustering the data in the reduced space formed by the selected Q features as the predicted cluster label. So, NMI, ARI, and JI measure how the cluster assignments in the original space and in the selected space agree, effectively giving a measure for the preservation of the original cluster structure in the selected space. We know that the maximum value for NMI or ARI or JI is 1. Here, the value of each of these three measures being close to 1 indicates that the cluster structure in the original space is preserved in the selected space. As the clustering algorithm we use, the Fuzzy C Means (FCM) algorithm [117] with the fuzzy exponent $m = 2$. We set the number of clusters for FCM algorithm as the number of classes. We use two values for the number of the selected features, Q . $Q = 0.35 \times P$ and $Q = 0.5 \times P$, where these values are rounded up to the nearest integers using the ceiling function. We report the average of each performance index, computed using the five subsets of features obtained in the five different runs.

Tables 4.2 and 4.3 summarize the performances of the proposed method and other compared methods for training and test sets, respectively for the E. coli dataset. We tabulate the four previously mentioned structure preserving measures and one classi-

Table 4.2: Performance comparison for E. coli, training set for different choices of β and Q .

Method	SS		NMI		ARI		JI		SVM OCA		NN OCA	
	Q=3	Q=4	Q=3	Q=4	Q=3	Q=4	Q=3	Q=4	Q=3	Q=4	Q=3	Q=4
ICA	0.2323	0.2146	0.4250	0.4220	0.2643	0.2617	0.4250	0.4220	0.6788	0.6656	0.4338	0.4795
F Score	0.1547	0.0648	0.5417	0.6124	0.3520	0.5073	0.5417	0.6124	0.6325	0.7815	0.6377	0.6517
Mutual Info	0.0726	0.0204	0.6061	0.7511	0.4993	0.6884	0.6061	0.7511	0.6823	0.6719	0.6066	0.4834
FSMLP	0.1391	0.1098	0.4356	0.4751	0.2705	0.3167	0.4356	0.4751	0.8391	0.8828	0.5464	0.4987
FSMLP _{struct} , $\beta = 0.1$	0.1391	0.1098	0.4356	0.4751	0.2705	0.3167	0.4356	0.4751	0.8391	0.8828	0.5179	0.4987
FSMLP _{struct} , $\beta = 1$	0.0902	0.0391	0.5338	0.6138	0.3868	0.4873	0.5338	0.6138	0.9000	0.9007	0.5755	0.5623
FSMLP _{struct} , $\beta = 10$	0.0766	0.0280	0.5812	0.7137	0.4497	0.6362	0.5812	0.7137	0.8675	0.8768	0.5603	0.5020

Table 4.3: Performance comparison for E. coli, test set for different choices of β and Q .

Method	SS		NMI		ARI		JI		SVM OCA		NN OCA	
	Q=3	Q=4	Q=3	Q=4	Q=3	Q=4	Q=3	Q=4	Q=3	Q=4	Q=3	Q=4
ICA	0.2562	0.2332	0.6191	0.6068	0.2858	0.2775	0.6191	0.6068	0.5588	0.6471	0.4529	0.4706
F Score	0.1389	0.0682	0.7124	0.7150	0.4579	0.4552	0.7124	0.7150	0.5882	0.6471	0.5882	0.5824
Mutual Info	0.0763	0.0426	0.7047	0.8047	0.4541	0.6114	0.7047	0.8047	0.4545	0.5909	0.5647	0.4706
FSMLP	0.1217	0.0976	0.7341	0.7416	0.5284	0.5449	0.7341	0.7416	0.6824	0.7000	0.4941	0.4706
FSMLP _{struct} , $\beta = 0.1$	0.1217	0.0976	0.7341	0.7416	0.5284	0.5449	0.7341	0.7416	0.6824	0.7000	0.4706	0.4706
FSMLP _{struct} , $\beta = 1$	0.0949	0.0382	0.6910	0.8179	0.4215	0.6669	0.6910	0.8179	0.7235	0.8118	0.5000	0.5059
FSMLP _{struct} , $\beta = 10$	0.0932	0.0346	0.6921	0.8571	0.4171	0.7029	0.6921	0.8571	0.7412	0.8588	0.5412	0.4706

fication score for two choices of the number of selected features (approximately 35% and 50% of the original dimension) i.e., $Q = 3$, and $Q = 4$ in Tables 4.2 and 4.3, respectively. As we have already discussed in Sec. 4.2 the lesser the value of SS, the better the projected space (formed by selected features) preserves the original pairwise distances and hence the structure of the original data. We observe in Table 4.2, the mutual information based method shows the lowest value of SS, and the second lowest is FSMLP_{struct} with $\beta = 10$ for both $Q = 3$ and $Q = 4$. Actually, the SS values for the mutual information based method and FSMLP_{struct} with $\beta = 10$ are almost the same, equal up to two places after decimal points in both choices of Q . The SS values achieved by ICA, the F score based method and FSMLP are comparatively higher. So, the mutual information based method and FSMLP_{struct} with $\beta = 10$ preserve the original pairwise distances most in the projected space. They are also expected to preserve the structures most. The values of the other three structure preserving measures i.e., NMI, ARI, and JI confirm the same. We know that the higher the values of NMI, ARI, and JI, the closer are the cluster structures of the projected space to the original cluster structures. The highest values of the NMI, ARI, and JI are obtained by the mutual information based method, followed by the FSMLP_{struct} with $\beta = 10$. So, in cluster structure preservation, mutual information based method and FSMLP_{struct} with $\beta = 10$ perform better than the other three methods and even better than the other two models trained by FSMLP_{struct} with $\beta = 0.1$ and 1. β is the weight of the regularizer $E_{sammons}$ in (4.2.7). Although SS and $E_{sammons}$ are not exactly same, under

Table 4.4: Performance comparison for Glass, training set for different choices of β and Q .

Method	SS		NMI		ARI		JI		SVM OCA		NN OCA	
	Q=4	Q=5	Q=4	Q=5	Q=4	Q=5	Q=4	Q=5	Q=4	Q=5	Q=4	Q=5
ICA	0.2699	0.2216	0.5066	0.5416	0.3343	0.3688	0.5066	0.5416	0.6771	0.6823	0.3729	0.4260
F Score	0.1284	0.1011	0.5961	0.6343	0.5137	0.5750	0.5961	0.6343	0.6979	0.7396	0.4531	0.4771
Mutual Info	0.2091	0.1159	0.5073	0.6113	0.3321	0.5688	0.5073	0.6113	0.6823	0.6719	0.5156	0.4365
FSMLP	0.1968	0.1593	0.5512	0.6028	0.4473	0.5235	0.5512	0.6028	0.6302	0.7094	0.3688	0.3479
FSMLP _{struct} , $\beta = 0.1$	0.1968	0.0747	0.5512	0.7501	0.4473	0.6873	0.5512	0.7501	0.6302	0.7510	0.3688	0.3802
FSMLP _{struct} , $\beta = 1$	0.0494	0.0623	0.7732	0.7965	0.7124	0.7665	0.7732	0.7965	0.7740	0.7708	0.4198	0.3490
FSMLP _{struct} , $\beta = 10$	0.0309	0.0406	0.8551	0.7394	0.8507	0.6792	0.8551	0.7394	0.8125	0.7667	0.3698	0.4042

Table 4.5: Performance comparison for Glass, test set for different choices of β and Q .

Method	SS		NMI		ARI		JI		SVM OCA		NN OCA	
	Q=4	Q=5	Q=4	Q=5	Q=4	Q=5	Q=4	Q=5	Q=4	Q=5	Q=4	Q=5
ICA	0.3039	0.2333	0.6536	0.6063	0.3409	0.2979	0.6536	0.6063	0.4545	0.4545	0.4182	0.5000
F Score	0.1393	0.1258	0.7028	0.7744	0.5746	0.6861	0.7028	0.7744	0.5455	0.6364	0.5091	0.5364
Mutual Info	0.2034	0.1372	0.7805	0.7728	0.6358	0.6655	0.7805	0.7728	0.4545	0.5909	0.5909	0.4818
FSMLP	0.2027	0.1669	0.7366	0.7425	0.5053	0.5380	0.7366	0.7425	0.5727	0.6727	0.4364	0.3818
FSMLP _{struct} , $\beta = 0.1$	0.2027	0.0806	0.7366	0.7885	0.5053	0.6190	0.7366	0.7885	0.5727	0.6636	0.4364	0.4273
FSMLP _{struct} , $\beta = 1$	0.0406	0.0636	0.8265	0.8300	0.6951	0.7355	0.8265	0.8300	0.6818	0.6727	0.4455	0.4091
FSMLP _{struct} , $\beta = 10$	0.0215	0.0372	0.9022	0.7950	0.8670	0.6210	0.9022	0.7950	0.6727	0.6818	0.4091	0.4545

the influence of E_{select} , it is expected that the higher the value of β lesser the value of SS would be. Table 4.2 reconfirms that. The SS values become lesser as β increases from 0.1 to 10. SS values of FSMLP (which is basically FSMLP_{struct} with $\beta = 0$) and FSMLP_{struct} with $\beta = 0.1$ are the same for both the choices of Q . Except for NN OCA, the methods FSMLP and FSMLP_{struct} with $\beta = 0.1$ have the other ten measures same. It proves that for the E.coli dataset, $\beta = 0.1$ does not give any effective weight to the structure preservation term and chooses the same subset as that by FSMLP. For the classification performance measure OCA, FSMLP_{struct} with $\beta = 1$, achieves the highest value, followed by FSMLP_{struct} with $\beta = 10$. The mutual information based method and FSMLP_{struct} with $\beta = 10$ have all the structure preserving measures either almost equal or of comparable values, however for OCA, FSMLP_{struct} with $\beta = 10$ is better than mutual information based method with a margin more than 18%. For E. coli data, the test set follows the observed trends in the training set with the following exceptions. First, for $Q = 3$, the values of NMI, ARI, and JI have not increased as β increases from 0.1 to 10. Second, for $Q = 4$, FSMLP_{struct} with $\beta = 10$ beats all the methods including mutual information based method. Analyzing the performances over train and test sets, for E. coli data FSMLP_{struct} with $\beta = 10$ is the winner among the other six models.

Tables 4.4 and 4.5 report the performance of the Glass dataset on its training and test sets, respectively.

The chosen numbers of features for the Glass data are 4 and 5. The expected nature

Table 4.6: Performance comparison for Ionosphere, training set for different choices of β and Q .

Method	SS		NMI		ARI		JI		SVM OCA		NN OCA	
	Q=12	Q=17	Q=12	Q=17	Q=12	Q=17	Q=12	Q=17	Q=12	Q=17	Q=12	Q=17
ICA	0.1791	0.0926	0.8261	0.8817	0.8766	0.9250	0.8261	0.8817	0.9492	0.9714	0.8787	0.8889
F Score	0.1753	0.0914	0.9035	0.9313	0.9497	0.9621	0.9035	0.9313	0.9397	0.9746	0.8660	0.9022
Mutual Info	0.1776	0.0803	0.6829	0.8645	0.7838	0.9250	0.6829	0.8645	0.9746	0.9651	0.8952	0.9048
FSMLP	0.1721	0.0895	0.6292	0.7330	0.7246	0.8192	0.6292	0.7330	0.9606	0.9733	0.8971	0.9117
FSMLP _{struct} , $\beta = 0.1$	0.1675	0.0878	0.6689	0.7415	0.7641	0.8243	0.6689	0.7415	0.9638	0.9733	0.8883	0.9054
FSMLP _{struct} , $\beta = 1$	0.1505	0.0776	0.7307	0.7867	0.8051	0.8582	0.7307	0.7867	0.9632	0.9568	0.8737	0.8844
FSMLP _{struct} , $\beta = 10$	0.1437	0.0766	0.7894	0.8111	0.8533	0.8723	0.7894	0.8111	0.9683	0.9644	0.8425	0.8870

of decreasing SS with increasing β is clearly observed for $Q = 5$ for both the training and test set. For $Q = 4$, the Glass data also follows the characteristics of the E. coli data of having the same values for FSMLP and FSMLP_{struct} with $\beta = 0.1$ in all the ten measures for both training and test set. For $Q = 4$, from $\beta = 0.1$ onwards, increasing β s produce decreasing SS values and increasing NMI, ARI, and JI values for both training and test datasets. We observe from the Tables 4.4 and 4.5, for $Q = 5$, as the β increases from 0 (FSMLP) to 0.1, and then to 1, NMI, ARI, and JI values are increased for both training and test datasets. However, at $\beta = 10$, NMI, ARI, and JI values are decreased compared to $\beta = 0.1$ and 1. We can conclude that, for $Q = 4$, FSMLP_{struct} with $\beta = 10$ gives the best structure preserving performance among the considered models and for $Q = 5$, FSMLP_{struct} with $\beta = 1$ is best in structure preservation. In terms of the classification performance measure OCA, FSMLP_{struct} with $\beta = 10$ and FSMLP_{struct} with $\beta = 1$ show the highest OCA values for the training set and test set respectively, with $Q = 4$. On the other hand, for $Q = 5$, FSMLP_{struct} with $\beta = 10$ show the highest OCA values for the training set and FSMLP_{struct} with $\beta = 1$ show the highest OCA values for the test set. Inspecting all the performance measure values, we may conclude that for the Glass dataset, both FSMLP_{struct} with $\beta = 10$ and FSMLP_{struct} with $\beta = 1$ are comparatively better in simultaneously preserving both class and cluster structures than the other methods.

The performances of the Ionosphere dataset are recorded in Tables 4.6 and 4.7 for training and test sets respectively. For the Ionosphere data set, the number of selected features, Q is set to 12 and 17. Here, in all the cases, whenever the β is increasing, SS is decreasing and the other structure preserving indices NMI, ARI, and JI are increasing consistently. Unlike, E. coli and Glass data set, here when β increases from 0 (in FSMLP) to 0.1, the structure preserving metrics including SS shifted in the desired direction in most of the cases and remained the same in some cases. Except for SS, in the other three structure preserving measures, ICA and F score based methods have performed better than FSMLP and FSMLP_{struct} for all the cases. Classification performance is good for almost all methods for the Ionosphere data set. In the training set, for both $Q = 12$ and $Q = 17$, an accuracy of 97.46% is reached by mutual info and F score based methods, however, FSMLP and FSMLP_{struct}

Table 4.7: Performance comparison for Ionosphere, test set for different choices of β and Q .

Method	SS		NMI		ARI		JI		SVM OCA		NN OCA	
	Q=12	Q=17	Q=12	Q=17	Q=12	Q=17	Q=12	Q=17	Q=12	Q=17	Q=12	Q=17
ICA	0.1652	0.0964	0.7351	0.7351	0.7782	0.7782	0.7351	0.7351	0.9143	0.9429	0.8571	0.7886
F Score	0.1952	0.1003	0.4325	0.4778	0.3411	0.4152	0.4325	0.4778	0.9429	0.9429	0.8057	0.8229
Mutual Info	0.2186	0.0972	0.5287	0.6541	0.4959	0.6774	0.5287	0.6541	0.9429	0.9429	0.8229	0.8171
FSMLP	0.1801	0.0913	0.6518	0.8284	0.6841	0.8682	0.6518	0.8284	0.9371	0.9143	0.8629	0.8057
FSMLP _{struct} , $\beta = 0.1$	0.1722	0.0885	0.7434	0.8284	0.7714	0.8682	0.7434	0.8284	0.9371	0.9314	0.8229	0.7886
FSMLP _{struct} , $\beta = 1$	0.1437	0.0726	0.7764	0.8597	0.8265	0.8884	0.7764	0.8597	0.8971	0.9257	0.8057	0.8286
FSMLP _{struct} , $\beta = 10$	0.1310	0.0720	0.7960	0.9146	0.8454	0.9328	0.7960	0.9146	0.9029	0.9257	0.8114	0.8171

Table 4.8: Performance comparison for Sonar, training set for different choices of β and Q .

Method	SS		NMI		ARI		JI		SVM OCA		NN OCA	
	Q=21	Q=30	Q=21	Q=30	Q=21	Q=30	Q=21	Q=30	Q=21	Q=30	Q=21	Q=30
ICA	0.7830	0.4963	0.0004	0.0276	-0.0040	0.0382	0.0004	0.0276	0.7112	0.8075	0.6417	0.7711
F Score	0.3614	0.2029	0.1167	0.1778	0.1159	0.1926	0.1167	0.1778	0.8289	0.9733	0.8128	0.8321
Mutual Info	0.2247	0.0950	0.4601	0.6601	0.5662	0.7585	0.4601	0.6601	0.9733	0.9465	0.8086	0.8332
FSMLP	0.1880	0.0910	0.3735	0.5293	0.4432	0.6104	0.3735	0.5293	0.9273	0.9765	0.8064	0.8203
FSMLP _{struct} , $\beta = 0.1$	0.1314	0.0779	0.4917	0.5587	0.5698	0.6358	0.4917	0.5587	0.9775	0.9722	0.8075	0.8171
FSMLP _{struct} , $\beta = 1$	0.0620	0.0285	0.5743	0.6838	0.6620	0.7696	0.5743	0.6838	0.9872	0.9840	0.7968	0.8193
FSMLP _{struct} , $\beta = 10$	0.0481	0.0201	0.6983	0.7473	0.7751	0.8318	0.6983	0.7473	0.9936	0.9968	0.7305	0.8118

models have reached more than 96% accuracy in every case. For the test set, all the structure preserving indices are better for FSMLP and FSMLP_{struct} than ICA, F score, and mutual information based methods, although in terms of classification score OCA, the F score and mutual information based methods have performed marginally better than FSMLP and FSMLP_{struct} models. This may have happened because FSMLP_{struct} makes a compromise between preserving cluster structure and classifier loss. For the Ionosphere dataset, our proposed models are not the winner.

For the Sonar data, the summary of the performance on the training and test data sets in terms of the five measures for two choices of the number of selected features are included in Tables 4.8 and 4.9. We set, $Q = 21$ and 30 for the Sonar data set. In the case of the Sonar data set, not only with increasing β , all the structure preserving indices improve, in case of the training set, FSMLP_{struct} with $\beta = 10$ are significantly better than ICA, F score, and mutual information based methods, and FSMLP in all five scores for both the choices of Q . For the test set, in some cases, FSMLP_{struct} with $\beta = 1$ is better than FSMLP_{struct} with $\beta = 10$. For the Sonar data set, clearly, the proposed method performed extremely well in terms of classification and clustering performance.

Tables 4.10 and 4.11 summarize the performance of the proposed method and other compared methods for training and test sets, respectively for the AR10P data set. The original number of features, P for the AR10P data set is 2400, which is comparatively

Table 4.9: Performance comparison for Sonar, test set for different choices of β and Q .

Method	SS		NMI		ARI		JI		SVM OCA		NN OCA	
	Q=21	Q=30	Q=21	Q=30	Q=21	Q=30	Q=21	Q=30	Q=21	Q=30	Q=21	Q=30
ICA	0.7556	0.4968	0.2513	0.1993	0.1546	0.2383	0.2513	0.1993	0.7143	0.5714	0.6286	0.4667
F Score	0.3629	0.1941	0.2906	0.3029	0.3519	0.3536	0.2906	0.3029	0.5714	0.8571	0.6286	0.6952
Mutual Info	0.2144	0.0955	0.2906	0.5411	0.3519	0.6378	0.2906	0.5411	0.8095	0.8095	0.5524	0.6286
FSMLP	0.1967	0.0977	0.3918	0.5288	0.4633	0.5850	0.3918	0.5288	0.8190	0.8857	0.6286	0.6762
FSMLP _{struct} , $\beta = 0.1$	0.1436	0.0840	0.4026	0.5540	0.4368	0.6155	0.4026	0.5540	0.7714	0.8762	0.6667	0.6952
FSMLP _{struct} , $\beta = 1$	0.0679	0.0302	0.6905	0.6824	0.7223	0.7146	0.6905	0.6824	0.7619	0.8381	0.6571	0.6381
FSMLP _{struct} , $\beta = 10$	0.0473	0.0198	0.5001	0.7156	0.5547	0.7490	0.5001	0.7156	0.8000	0.7810	0.7048	0.6762

Table 4.10: Performance comparison for AR10P, training set for different choices of β and Q .

Method	SS		NMI		ARI		JI		SVM OCA		NN OCA	
	Q=40	Q=60	Q=40	Q=60	Q=40	Q=60	Q=40	Q=60	Q=40	Q=60	Q=40	Q=60
ICA	0.7619	0.7107	0.2807	0.2449	0.2222	0.2066	0.2807	0.2449	1	1	0.6957	0.7436
F Score	0.7933	0.7484	0.1693	0.1709	0.0550	0.0438	0.1693	0.1709	1	1	0.8188	0.8940
Mutual Info	0.7805	0.7309	0.1613	0.1863	0.0429	0.0558	0.1613	0.1863	0.9915	0.9915	0.8376	0.8821
FSMLP	0.7640	0.7109	0.2296	0.2108	0.1038	0.0943	0.2296	0.2108	0.9983	1	0.9470	0.9419
FSMLP _{struct} , $\beta = 25$	0.7566	0.7061	0.3828	0.5251	0.3907	0.5654	0.3828	0.5251	1	1	0.9162	0.9094
FSMLP _{struct} , $\beta = 50$	0.7567	0.7059	0.5307	0.5089	0.5575	0.5268	0.5307	0.5089	1	1	0.8718	0.9043
FSMLP _{struct} , $\beta = 100$	0.7564	0.7061	0.4849	0.5830	0.4852	0.6353	0.4849	0.5830	1	1	0.8872	0.8957

Table 4.11: Performance comparison for AR10P, test set for different choices of β and Q .

Method	SS		NMI		ARI		JI		SVM OCA		NN OCA	
	Q=40	Q=60	Q=40	Q=60	Q=40	Q=60	Q=40	Q=60	Q=40	Q=60	Q=40	Q=60
ICA	0.7739	0.7256	0.8255	0.8222	0.0031	-0.0523	0.8255	0.8222	0.6154	0.6154	0.2615	0.3385
F Score	0.7364	0.6856	0.9072	0.9152	0.3645	0.4155	0.9072	0.9152	0.8462	0.8462	0.7538	0.8615
Mutual Info	0.7614	0.7112	0.8451	0.8451	0.0122	0.0122	0.8451	0.8451	0.9231	0.8462	0.8154	0.8462
FSMLP	0.7838	0.7213	0.9051	0.9152	0.3686	0.4155	0.9051	0.9152	0.8462	0.7538	0.8308	0.6615
FSMLP _{struct} , $\beta = 25$	0.7795	0.7346	0.9223	0.9146	0.4820	0.4380	0.9223	0.9146	0.4615	0.6154	0.6000	0.6000
FSMLP _{struct} , $\beta = 50$	0.7815	0.7352	0.9050	0.9050	0.3782	0.3782	0.9050	0.9050	0.4615	0.6154	0.5385	0.6308
FSMLP _{struct} , $\beta = 100$	0.7804	0.7356	0.9056	0.9136	0.3557	0.4727	0.9056	0.9136	0.4615	0.6154	0.5231	0.5077

higher than that of the other four data sets used in this sub-section. The two choices of the number of selected features here are 40 and 60 and these are not approximately 35% and 50% of the original dimension like in previous cases. The study in [27], proposed a feature selection scheme for redundancy control in features. They reported an average number of selected features of 58.9 without practicing redundancy control and an average number of selected features in the range of 22.8 to 44.2 when practicing redundancy control for AR10P data set. Hence, we choose the number of selected features Q as 40 and 60. From the classification scores shown in Table 4.10, we note that for all the methods for both the choices of Q , classification scores in training set

are more than 99%. In the training set, we observe that for $\text{FSMLP}_{\text{struct}}$, as β increases SS decreases in almost all the cases. But for the test set, this is not true. For the other structure-preserving measures for the training set, $\text{FSMLP}_{\text{struct}}$ with $\beta = 50$ is best among all the methods for $Q = 40$ and $\text{FSMLP}_{\text{struct}}$ with $\beta = 100$ is best among all the methods for $Q = 60$. In the test set, all the methods have performed almost the same in terms of the structure-preserving measures. The classification performances of $\text{FSMLP}_{\text{struct}}$ are very poor in the test set for AR10P data. The significant differences in training and test OCA values for $\text{FSMLP}_{\text{struct}}$ indicate poor generalization of the system. This problem may be addressed by choosing optimal number of nodes for $\text{FSMLP}_{\text{struct}}$ through cross-validation.

Results from the five data sets clearly establish the benefit of introducing the proposed structure preserving regularizer term, E_{sammons} in the overall loss function (4.2.8) of the MLP based embedded feature selection scheme. The classification performance using MLP networks is generally worse than that of the corresponding SVM classifiers. This is also true for the features selected by other feature selection methods in most of the cases. As mentioned earlier, the number of hidden nodes used for every dataset is the same as that used for the FSMLP network. Thus, the number of hidden nodes used for testing the performance of the selected features is not likely to be the best choice because the number of features used in this case is much less than that used by the FSMLP networks. This may be a possible cause for the poor performance of MLP for all feature selection methods. This issue needs further investigation. Next we shall consider the band (channel) selection problem for hyperspectral satellite images.

4.3.2 Band selection in hyperspectral images

Hyperspectral imaging is an advanced imaging technique where the objects are characterized by their spectral reflectances over hundreds of narrow and contiguous bands from the visible to the near-infrared region [234–236]. A hyperspectral image (HSI) represents an image scene by a three-dimensional cube where the third dimension is essentially the spectral information [237]. HSIs may have hundreds of bands to achieve higher spectral resolution, however, this leads to some difficulties. To overcome the difficulties, band selection is performed on HSIs. We employ our proposed feature selection scheme as a band selection scheme for HSI data sets. Let our considered hyperspectral image I be of dimension, $H \times W \times P$ where, H , W , and P are the height, width, and number of spectral bands of the image respectively. We can represent the pixels of I as $\mathbf{x}_i \in \mathbb{R}^P : i = 1, 2, \dots, H \times W$. Let, there be total n pixels annotated with C land cover classes. Without any loss of generality, we index the n pixels, as $\mathbf{x}_i$; $i = 1, 2, \dots, n$ which are having class labels. Our input data for land cover classification problem be $\mathbf{X} = \{\mathbf{x}_i = (x_{i1}, x_{i2}, \dots, x_{iP}) \in \mathbb{R}^P\}_{i=1}^n$. The collection of class labels of $\mathbf{X}$ be $\mathbf{Z} = \{z_i \in \{1, 2, \dots, C\}\}_{i=1}^n$, where, z_i is the class label corresponding to $\mathbf{x}_i$. We aim to select a subset of size Q from the original set of bands such that the selected subset performs reasonably well for land cover classification as well as in clustering.

We have performed the experiments with three benchmark HSI datasets for land cover classification problems- Indian pines, Pavia University, and Salinas [238]. We

have used the corrected version of the Indian pines and Salinas datasets having the number of bands 200 and 204 respectively. The Pavia University dataset uses 103 bands. The pre-processing of the datasets is the same as done in [239], following the code available in [240]. For any dataset, its pixel values are scaled to $[0, 1]$ using the expression $(x - \min(x))/(\max(x) - \min(x))$, where, x is a pixel value. The max and min are computed over the entire HSI. The data are then mean normalized across each channel by subtracting channel-wise means. The datasets are partitioned into training and test datasets. For band selection, only the training datasets are fed to the model. For measuring performance both training and test datasets are used. For splitting the datasets into training and test subsets, we drop the pixels of the unknown land-cover type. Let $\mathbf{X}$ be the set of pixels with known land-cover type. To obtain the training and test sets, let us divide $\mathbf{X}$ into two subsets $\mathbf{A}$ and $\mathbf{B}$ such that $\mathbf{A} \cup \mathbf{B} = \mathbf{X}$, $\mathbf{A} \cap \mathbf{B} = \phi$, and $\mathbf{A}$ and $\mathbf{B}$ contain, respectively, 75% and 25% pixels of $\mathbf{X}$. We use $\mathbf{B}$ as the test set. Note that, all three datasets suffer from the class imbalance problem. To avoid the learning difficulty raised by class imbalance, in the training set, we consider the same number of instances from each class. For this, from the subset $\mathbf{A}$, we randomly select (without replacement) 200 pixels per class. If a class has less than 200 instances in $\mathbf{A}$, we oversample the class by synthetic minority oversampling technique (SMOTE) [241] to gather 200 points. For band selection also, we use the same neural network (Fig. 4.1) with the number of hidden layers, $n_H = 3$. All the hyperparameters of the network, namely, number of hidden layers, number of nodes, type of activation functions and number of iterations for training are set according to [239], following the code available in [240]. The numbers of hidden nodes in the three hidden layers are 500, 350, and 150 respectively. Here the number of input nodes of the MLP is equal to the number of bands (P). The network weights and the gate parameters λ_j s are initialized in the same way as done for the datasets, described in Subsec. 4.3.1. The subset of Q bands is also selected in the same way as described in Subsec. 4.3.1. Here also the adaptive gradient algorithm is used as the learning algorithm and implemented with the ‘*train.AdagradOptimizer*’ routine of the ‘*TensorFlow*’ framework [193]. For all experiments of the current sub-section, α_1 and α_2 of the error functions in Equations (4.2.6) and (4.2.10) are set as 5 and 1 respectively. The total number of iterations for training the network is set to 50000.

Experiment I

Here the proposed feature selection scheme is used for band selection of remote sensing images with land cover types as the class labels. To accommodate the complexity of high dimensional HSI data, we have used a deeper network than that used for the datasets in Subsec. 4.3.1. Since image data or to be specific HSI datasets are widely different in nature from the typical classification datasets used in Subsec. 4.3.1, before stepping into a dual objective of classification and clustering, we first check the performance of the band selection model for land cover classification. In this experiment (Experiment I), we consider the band selection for land cover classification as the target problem and compare the performance of our method with other band selection methods. So,

we train our MLP-embedded band selection method by taking Equation (4.2.6) as the loss function. Here we denote the proposed method as BSMLP (Band Selection MLP). Figure 4.2 shows the classification performances of BSMLP, ICA based band selection [242], F-score [51], and mutual information [52, 53] based filtering methods. To measure the classification performance we compute OCA as done in Subsec. 4.3.1. Figure 4.2a and Figure 4.2b depict the results on the Indian pines dataset for the training dataset and test dataset respectively. Both for training and test set, BSMLP shows OCA values higher or comparable to the other band selection methods. For less number of selected bands (say up to the number of selected bands=100), the OCA of BSMLP is greater than that of other band selection methods with quite a good margin for both training (Figure 4.2a) and test sets (Figure 4.2b). As the number of selected bands increases further the margin of difference in OCA between BSMLP and other band selection methods, as expected, decreases finally converging at 200 (all bands) which is natural.

The same observation is true for the Pavia university dataset as seen from its graphical results in Figure 4.2c (training set) and Figure 4.2d (test set). So, BSMLP is the winner among the compared methods for the Indian pines and Pavia university datasets.

For the Salinas dataset, we note from Figure 4.2e (training set) and Figure 4.2f (test set) that the OCA values of BSMLP and mutual information-based filter method are almost the same. The F-score-based filter method also shows similar values of OCAs for the number of selected bands= 40 and beyond. The ICA-based method shows comparatively low values of OCA. So, for the Salinas dataset, both BSMLP and the filter method based on mutual information are best among the four methods. Considering all the results, BSMLP is demonstrated to be a better performing band selection method for the land cover classification task.

In Figure 4.3, we demonstrate thematic maps of the entire spatial region captured in the Indian pines dataset. We use the class labels obtained from the SVM classifier (trained on the considered training set) on the representations by 70 bands (Figure 4.3b), and 140 bands (Figure 4.3c) selected by BSMLP. We also show the mapping by the ground truth labels in Figure 4.3a. Figure 4.3 reveals that BSMLP is capable of selecting useful bands that can broadly capture the land cover types. Moreover, with an increase in the number of selected bands, the accuracy of land cover classification also increases.

Experiment II

In Experiment II, we use the loss function proposed in (4.2.10). Note that, in (4.2.10), as a structure preserving regularizer, E_{struct} defined in (4.2.9) is used. Let us name this proposed band selection method considering structure-preserving regularizer as BSMLP_{struct}. The number of training instances of the Indian pines and Salinas data set is 3200 and that of Pavia university is 1800. Both of the number of training instances, n are high. Computation of the $E_{sammons}$ in (4.2.7) would involve computing $(3200)^2$ or $(1800)^2$ distances. Adding $E_{sammons}$ to the overall loss function would cause

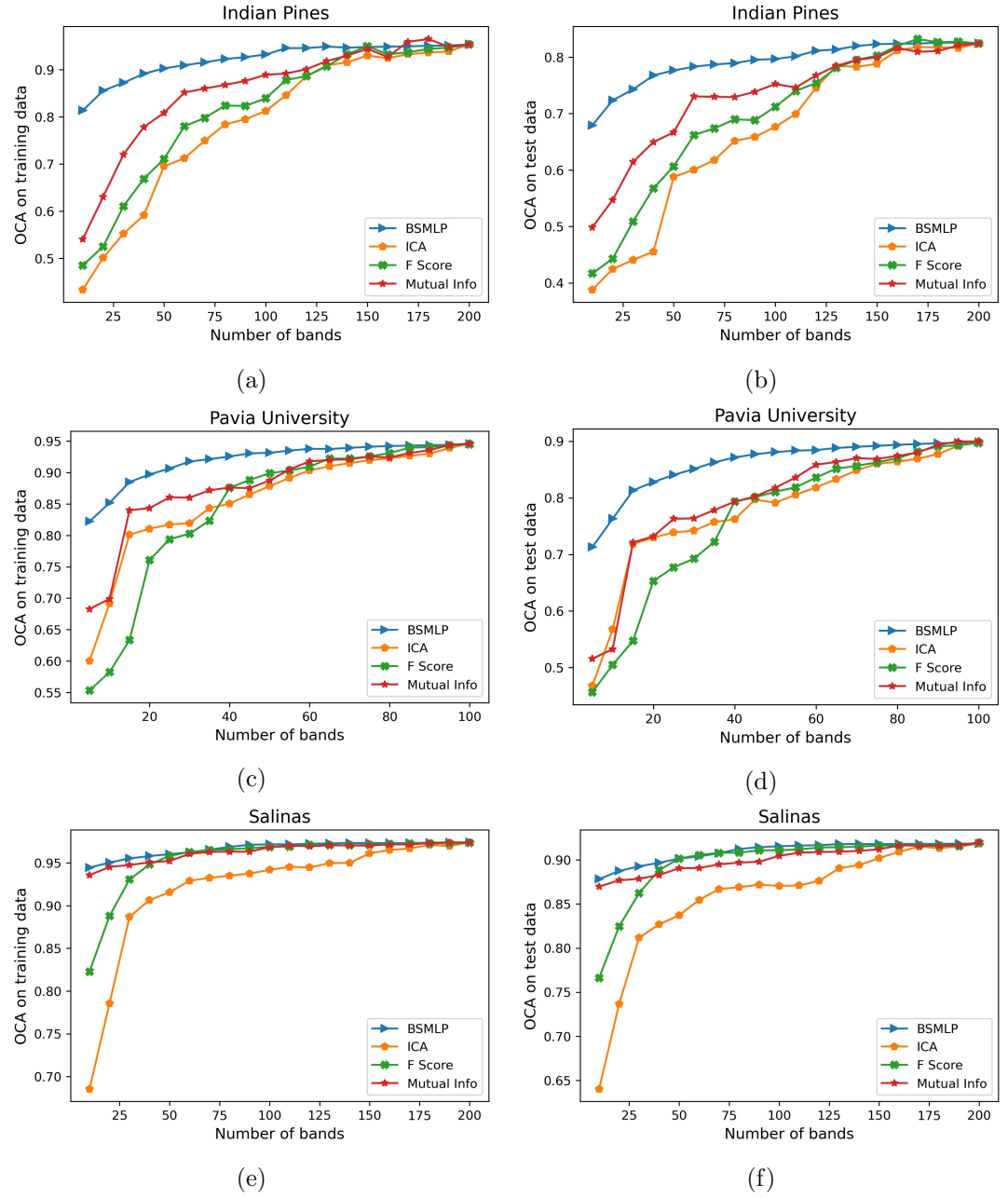


Figure 4.2: Variations of OCA with the number of selected bands for (a) Indian pines training set, (b) Indian pines test set, (c) Pavia university training set, (d) Pavia university test set, (e) Salinas training set, (f) Salinas test set.

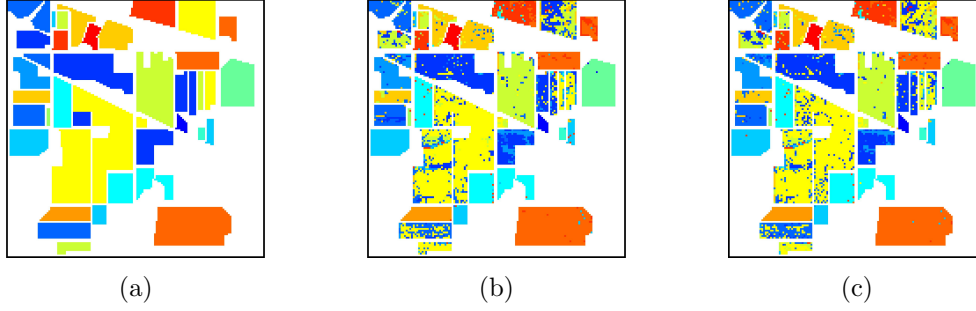


Figure 4.3: Thematic maps of the Indian pines dataset (a) with ground truth labels, (b) with labels from classification (SVM) considering 70 bands selected by BSMLP, (c) with labels from classification (SVM) considering 140 bands selected by BSMLP.

Table 4.12: Performance comparison for Indian Pines training set, with the number of bands= 70.

Method	SS	NMI	ARI	JI	OCA
ICA	0.6982	0.5437	0.3071	0.3335	0.7497
F Score	0.5325	0.5567	0.3015	0.0704	0.7978
Mutual Info	0.1966	0.5633	0.3232	0.2493	0.8600
BSMLP	0.0162	0.7994	0.6530	0.6827	0.9157
BSMLP _{struct} , $\beta=2$	0.0153	0.8027	0.6588	0.6894	0.9165
BSMLP _{struct} , $\beta=5$	0.0140	0.8051	0.6627	0.6945	0.9155
BSMLP _{struct} , $\beta=20$	0.0107	0.8188	0.6953	0.7216	0.9153
BSMLP _{struct} , $\beta=50$	0.0071	0.8400	0.7404	0.7549	0.9179

very intensive computation at each iteration. So, instead of $E_{sammons}$, its proposed approximation E_{struct} defined in (4.2.9) is used. In (4.2.9), $|S_t|$ is taken as 100. Varying the value of β in (4.2.10), we analyse its effect on the OCA, SS, NMI, ARI, and JI. We compute SS, NMI, ARI, and JI as described in Subsec. 4.3.1. We also use the same clustering algorithm with the same settings as used in Subsec. 4.3.1.

Tables 4.12 and 4.13 summarize the comparative results of BSMLP_{struct} with BSMLP and other band selection methods, ICA, F score, and mutual information based filter methods on the training and test datasets of Indian pines respectively. Similarly, Table 4.14 and 4.15 summarize the comparative results on the training and test datasets of Pavia university and Table 4.16 and 4.17 summarize the comparative results of training and test datasets of Salinas. In this experiment, we have fixed the number of selected bands Q , approximately to 35% of the original number of bands P . So, The number of selected bands is 70 for Indian pines and Salinas, and it is 35 for Pavia University. Tables 4.12 and 4.13 record the values of the structure-preserving indices and classification scores on Indian pines for different β values in BSMLP_{struct} (β values in Equation (4.2.10)). The considered β s for Indian pines, as well as those for Salinas, are 2, 5, 20, and 50. Note here that, BSMLP is basically BSMLP_{struct} with $\beta = 0$. We

Table 4.13: Performance comparison for Indian Pines test set, with the number of bands= 70.

Method	SS	NMI	ARI	JI	OCA
ICA	0.6748	0.5360	0.3052	0.2983	0.6175
F Score	0.5457	0.5518	0.2951	0.1643	0.6738
Mutual Info	0.2041	0.5787	0.3283	0.2395	0.7298
BSMLP	0.0158	0.8729	0.7949	0.8109	0.7871
BSMLP _{struct} , $\beta=2$	0.0148	0.8831	0.8210	0.8384	0.7871
BSMLP _{struct} , $\beta=5$	0.0137	0.8717	0.7881	0.8017	0.7881
BSMLP _{struct} , $\beta=20$	0.0102	0.8896	0.8346	0.8507	0.7882
BSMLP _{struct} , $\beta=50$	0.0066	0.8926	0.8360	0.8505	0.7850

Table 4.14: Performance comparison for Pavia University training set, with the number of bands= 35.

Method	SS	NMI	ARI	JI	OCA
ICA	0.3231	0.6941	0.5858	0.6340	0.8433
F Score	0.2386	0.6451	0.4654	0.4748	0.8233
Mutual Info	0.1387	0.8905	0.8769	0.8871	0.8717
BSMLP	0.1686	0.9072	0.8925	0.8863	0.9213
BSMLP _{struct} , $\beta=1$	0.1625	0.9169	0.9014	0.8952	0.9213
BSMLP _{struct} , $\beta=1.5$	0.1616	0.9233	0.9112	0.9033	0.9212
BSMLP _{struct} , $\beta=2$	0.1592	0.9243	0.9121	0.9044	0.9213
BSMLP _{struct} , $\beta=2.5$	0.1590	0.9198	0.9056	0.8993	0.9218

Table 4.15: Performance comparison for Pavia University test set, with the number of bands= 35.

Method	SS	NMI	ARI	JI	OCA
ICA	0.3215	0.5533	0.3777	0.2082	0.7573
F Score	0.3350	0.5091	0.3126	0.0886	0.7222
Mutual Info	0.1433	0.8911	0.8814	0.8990	0.7785
BSMLP	0.1584	0.9001	0.8930	0.9084	0.8628
BSMLP _{struct} , $\beta=1$	0.1568	0.9156	0.9131	0.9254	0.8650
BSMLP _{struct} , $\beta=1.5$	0.1573	0.9175	0.9149	0.9270	0.8672
BSMLP _{struct} , $\beta=2$	0.1556	0.9189	0.9166	0.9279	0.8666
BSMLP _{struct} , $\beta=2.5$	0.1562	0.9168	0.9143	0.9261	0.8648

observe in Tables 4.12 and 4.13 that both for training and test datasets as the value of β increases for BSMLP_{struct} (in the last five rows of the corresponding Tables) the value of SS becomes smaller. A similar trend is also observed for the Pavia University data set (here, β varies as 1, 1.5, 2, and 2.5) for training (Table 4.14) and test (Table 4.15) sets. For the Salinas dataset as revealed from Table 4.16 and Table 4.17, when β

Table 4.16: Performance comparison for Salinas training set, with the number of bands= 70.

Method	SS	NMI	ARI	JI	OCA
ICA	0.7159	0.7066	0.5385	0.5111	0.9325
F Score	0.0167	0.9420	0.9307	0.9291	0.9650
Mutual Info	0.0902	0.9207	0.8775	0.8860	0.9628
BSMLP	0.0072	0.9672	0.9589	0.9628	0.9654
BSMLP _{struct} , $\beta = 2$	0.0078	0.9678	0.9603	0.9640	0.9658
BSMLP _{struct} , $\beta = 5$	0.0072	0.9691	0.9626	0.9659	0.9648
BSMLP _{struct} , $\beta = 20$	0.0055	0.9710	0.9657	0.9683	0.9646
BSMLP _{struct} , $\beta = 50$	0.0039	0.9688	0.9630	0.9659	0.9649

Table 4.17: Performance comparison for Salinas test set, with the number of bands= 70.

Method	SS	NMI	ARI	JI	OCA
ICA	0.7027	0.6297	0.4144	0.3490	0.8670
F Score	0.0230	0.8584	0.7483	0.7615	0.9081
Mutual Info	0.0820	0.8791	0.7950	0.8230	0.8950
BSMLP	0.0078	0.9368	0.9183	0.9265	0.9077
BSMLP _{struct} , $\beta = 2$	0.0084	0.9392	0.9231	0.9305	0.9076
BSMLP _{struct} , $\beta = 5$	0.0079	0.9413	0.9275	0.9340	0.9069
BSMLP _{struct} , $\beta = 20$	0.0061	0.9366	0.9175	0.9258	0.9065
BSMLP _{struct} , $\beta = 50$	0.0044	0.9342	0.9136	0.9229	0.9073

is increased from 0 to 2, the value of SS is increased however, from $\beta = 2$ to $\beta = 50$ onward, the values of SS are decreased. As we increase the value of β , there is more stress to reduce the loss function Equation (4.2.9). In most cases, increasing β results in a drop in SS. This clearly suggests that the loss function Equation (4.2.9) that we use, is a computationally efficient substitute for the original SS defined in Equation (4.2.7). The exceptions for the Salinas dataset, while increasing β from 0 to 2 is possibly due to the fact that we do not use the entire training data in Equation (4.2.9) and use of $|S_t| = 100$ in Equation (4.2.9) is not adequate to capture the structure of the data faithfully for the Salinas dataset. As discussed earlier, setting the value of $|S_t|$ is crucial for approximating Equation (4.2.7) with Equation (4.2.9). We have set $|S_t| = 100$ for all three datasets empirically. However, choosing an optimum value of $|S_t|$ for each dataset is expected to avoid the occurred exceptions.

For the Pavia university dataset, we have set the values of β in BSMLP_{struct} as 1, 1.5, 2, and 2.5. Unlike Indian pines and Salinas datasets, for Pavia university, we restrict the β s to lower values. This is due to the fact that the number of selected bands for Pavia university is 35 and that for Indian pines and Salinas is 70. The smaller the number of bands, the lesser the importance (β) to be given to our structure preserving

regularizer in Equation (4.2.9) to obtain a desired balance between classification and clustering performance. Table 4.12 which contains the results for the Indian pines training data, clearly shows that both BSMLP and BSMLP_{struct} are better than ICA, F-score based, mutual information based methods in all four structure preserving metrics as well as in terms of the OCA. In Table 4.12 we observe that with increasing values of β there is a consistent improvement in the values of the four structure preserving metrics while the values of OCAs stay approximately at 91%. The results shown in Table 4.13 for the Indian pines test set also depict that BSMLP and BSMLP_{struct} perform better in terms of all the five metrics than the other three methods. Also with an increase in β all the structure-preserving metrics improve for BSMLP_{struct}, except the value of JI slightly decreases when β goes to 50 from 20. The classification metric OCA is around 78% with bands selected by BSMLP_{struct} for different choices of β s.

It is notable here that the test set is completely unseen in the process of band selection, yet the selected bands for the proposed method are providing fairly good results for structure preservation as well as for classification. As observed from Table 4.14 and Table 4.15 for Pavia university training and test sets respectively, the lowest (best) SS value among all the compared methods is achieved by mutual information based filter method. However, for the other four metrics i.e. NMI, ARI, JI and OCA; BSMLP and BSMLP_{struct} achieve better values. In the case of the Pavia university dataset with increasing β ; NMI, ARI, and JI are not consistently increasing but the results indicate that, it is possible to find a β , (here $\beta = 2$) where the structures are preserved better maintaining a good classification score. We note from Tables 4.16 and 4.17 that, for the Salinas dataset also BSMLP and BSMLP_{struct} are better than the other three methods in terms of all the five metrics used. All four structure preserving metrics scores for the BSMLP_{struct} are better than or comparable to those by BSMLP keeping the classification score OCA at approximately 96% for the training dataset and 90% for the test dataset.

Figure 4.4 illustrates thematic maps of the entire region captured in the Indian pines dataset represented with 70 bands selected by BSMLP_{struct} with $\beta = 0$, i.e., by the method BSMLP (Figure 4.4a), and BSMLP_{struct} considering $\beta = 20$ (Figure 4.4b), and $\beta = 50$ (Figure 4.4c). Figure 4.4 ensures that even with the increasing stress on the structure-preserving regularizer E_{struct} , our proposed band selection method BSMLP_{struct} is able to select bands that maintain a good land cover classification performance.

4.4 Conclusion and Discussions

To the best of our knowledge, a feature selection method that simultaneously cares about class discrimination and structure preservation is not available in the literature. In this study, we have tried to bridge this gap by proposing a neural network-based feature selection method that focuses both on class discrimination and structure preservation. To learn the proposed system, we use Sammon's stress as a regularizer to the classification loss. For datasets having a large number of instances, the computational

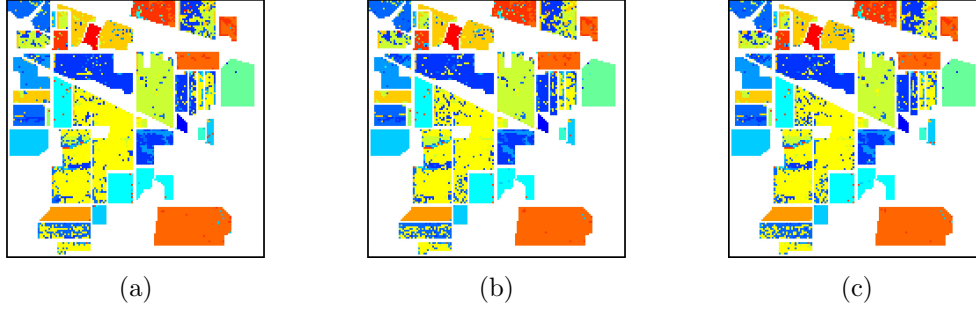


Figure 4.4: Thematic maps resulting from classifications (SVM) of the Indian pines dataset with 70 bands selected by BSMLP_{struct} considering (a) $\beta = 0$, (b) $\beta = 20$, (c) $\beta = 50$.

overhead associated with Sammon’s stress is very high. Consequently, as the structure-preserving regularizer, we use Sammon’s stress computed based on a sample of the original data (using dynamic sampling on each iteration during the adaptive gradient descent based learning). Using this regularizer in the experiments with datasets having a large number of instances, we have demonstrated that this regularizer is an effective and computationally efficient implementation of Sammon’s stress based structure-preserving regularizer. Our proposed feature selection scheme is generic. So we have investigated its effectiveness on datasets commonly used for assessing classifiers as well as for a specialized case: band selection in hyperspectral images(HSI). We have applied the feature selection scheme to five real-world datasets which are commonly used typically for assessing classification. In the context of band selection, we have applied our method to three well-known HSI datasets and compared performances with three other band selection methods. Based on our experiments, we conclude that the proposed feature selection method is able to produce reasonably good classification and clustering scores in the majority of the data sets, indicating that the proposed method is capable of selecting a subset of features that is good both for classification and clustering. Our scheme provides a mechanism to control the number of selected features. The proposed method is easily extendable to other networks like Radial Basis Function (RBF) network.

Chapter 5

Understanding classes better: class-specific and rule-specific feature selection along with redundancy control in a fuzzy rule-based framework [5]

5.1 Introduction

The feature selection schemes that we have introduced in Chapter 1 and explored in Chapter 4 choose a single “optimal” subset of the features for the entire dataset. Traditionally, this is how feature selection is done. However, some studies [135–142] have used class-specific approaches for selecting features, where for each class, a unique subset of the original features is selected. If there are C classes, in the class-specific approach, C subsets are chosen. In a traditional feature selection method, the selected feature subset is chosen based on the global characteristics of the data. It does not take into account any class-specific or local characteristics of the data which may be present. For example, there may exist a group of features that follows a distinct distribution for a specific class but varies randomly over the remaining classes. Such a group of features plays a significant role in distinguishing the specific class from other classes but may not be very useful for the C -class problem as a whole. There are application areas like the discovery of biomarkers for different diseases including cancers where disease(class)-specific attributes are of special significance which can help to understand the biology of the disease [141, 142]. Class-specific characteristics may also exist in the form of class-specific redundancy. Different sets of features could be redundant for different classes. Two features are called redundant if they have discriminatory power (or are important for a given task) and one feature is dependent on the other; for example, the absolute value of correlation between them or mutual information between them

is high. The two features form a set with redundancy as only one of them is needed for the task at hand. The class-specific feature selection (CSFS) methods in [135–142] have proposed suitable frameworks that exploit class-specific feature subsets to solve classification problems. They have shown that the classifiers built with the subsets chosen by class-specific methods performed better than or comparable to the classifiers built with subsets chosen by global feature selection methods. The CSFS may also enhance the transparency/explainability of the classification process associated with it [135]. Next, we briefly review the existing studies that deal with CSFS.

In [136], the authors proposed a CSFS scheme with one-against-all or one-versus-all (OVA) strategy-based multi-class support vector machine (SVM) classifiers. In this framework, the winning class is decided by aggregating outcomes of the C (number of classes), binary SVM classifiers. The authors deal with two issues: a) the same feature subset may not be optimal for all C binary SVMs, so they selected optimum features for individual OVA sub-problems; b) the reliability of all the classifiers may not be the same, so they proposed a reliability measure and applied the same for normalizing outputs from each binary classifier before aggregating them into the final outcome. The study in [139], proposed a general framework for extending any traditional feature selection method to a class-specific one. Basically, the authors of [139] proposed to convert a C -class classification problem into C binary classification problems and then to apply any traditional feature selection schemes to each of the C binary classification problems to generate C class-specific feature sets. The authors of [142], claimed that identifying different bio-markers for different cancer types may improve the diagnosis and treatments of the same. So, they applied their methods to some popular gene-expression data sets used for cancer identification. Their proposed scheme is based on pipe lining two modules: selection of features by maximum relevance binary particle swarm optimizer (MRBPSO) and class-dependent multi-category classification (CDMC). At first, the genes were (pre-)filtered using F-statistics. Here also, a C -class classification problem is converted into C binary classification problems. The optimum subsets for the binary classification problems are searched by a binary particle swarm optimizer (BPSO) which uses maximum relevance (MR) as the fitness function. The C class-dependent subsets learned by MRBPSO are then used in the C classifiers of the CDMC module. The authors explored fuzzy K -nearest neighbor classifiers and OVA-based multi-class SVM in the CDMC module. The method proposed in [137] is called Sparse-Modeling Based Approach for Class-Specific Feature Selection (SMBA-CSFS). In this work, each class of a dataset is modeled with its sparse representation using less number of features than the original. Since, instead of the whole data, each class is modeled, the representative feature sets are class-specific. Exploiting those C feature subsets an ensemble of C classifiers is built for performing the relevant classification task. The paper in [138] proposed CSFS for liver cancer classification. Here also, the original C class problem is decomposed to C -binary sub-problems using the OVA strategy. For each binary classification problem, the feature selection scheme in [138] applies two steps- information gain for initial filtering followed by a sequential forward feature selection. As a result, class-specific feature sets are generated. The

authors used K -nearest neighbors, decision tree, and naive Bayes classifiers to evaluate the selected class-specific features. In [140], the author proposed a CSFS method based on two goals: a) minimize the distances of samples from the same class and maximize distances between samples from different classes. These two objectives were combined to form a single objective specific to each class; as a result C class-specific feature sets were obtained. For classifying points from the test set, the author proposed a rule to aggregate results from C nearest neighbors classifiers defined in the C sub-spaces formed by those selected C class-specific subsets. The CSFS method in [141], involves elastic nets and probabilistic SVMs (PSVM). Again with OVA, C binary classification problems were generated. Elastic nets were used to select (class-specific) feature subsets for each of the binary classification problems. Then PSVMs were trained on those class-specific subsets and finally, from their outputs, the final class label was predicted. The study in [135] proposed a method based on genetic algorithms (GA) and an OVA strategy. The method in [135] converts a C -class classification problem into C binary classification problems and then uses GA to find the optimum feature set for each of the binary classification problems. For each candidate feature subset, its fitness is measured in terms of the classification score of a binary classifier on the relevant binary classification problem. The authors of [135] established the advantages of their CSFS method over the global feature selection methods not only by showing that their proposed CSFS scheme outperformed some popular global feature selection methods but also by interpreting the output in terms of the class-specific features. Therefore, they termed the feature selection scheme as an explainable feature selection method.

Note that, the majority of the CSFS methods [135, 136, 138, 139, 141, 142] follow the OVA strategy to select C class-specific feature subsets. OVA strategy-based CSFS methods have certain drawbacks. Generally, it leads to class imbalance demanding careful design of classifiers. The OVA strategy-based methods are computationally intensive and for testing we need an aggregation mechanism. Even, for the other CSFS methods [137, 140] which are not OVA strategy-based, to exploit the obtained class-specific feature subsets in the classification process, C classifiers are employed and require aggregation of C outputs for testing. Moreover, none of these methods consider the presence of - (i) class-specific redundancy, and (ii) within-class substructures in their frameworks.

Here, we propose a CSFS scheme embedded in a fuzzy rule-based classifier (FRBC) that does not use the OVA strategy. Our method selects class-specific feature subsets by learning a single FRBC and hence avoids the issues associated with the CSFS methods involving C classifiers. Moreover, we extend our framework to deal with- (i) CSFS with redundancy control, and (ii) rule-specific feature selection (RSFS) that can exploit the presence of substructures within a class. The rules provided by the FRBC are generally interpretable and more specific. Exploiting the class-specific local features, the proposed FRBC enjoys more transparency and interpretability than a standard classifier exploiting class-specific feature subsets. Our main contributions in this study are summarised as follows.

1. We propose a CSFS method that is not based on the OVA strategy like most of

the existing CSFS schemes. Thus, our method is free from the weaknesses of the OVA strategy.

2. We propose a classifier framework that is defined by class-specific fuzzy rules involving class-specific feature subsets.
3. Our method can monitor the level of redundancy in the selected features.
4. We also propose a general version of our class-specific feature selection method that not only chooses different subsets for different classes but also, chooses different subsets for different rules within a class if different substructures are present.

5.2 Proposed Method

We follow the same notations as those of the previous chapters i.e., the input data set is denoted by $\mathbf{X} = \{\mathbf{x}_i = (x_{i1}, x_{i2}, \dots, x_{iP})^T \in \mathbb{R}^P\}_{i=1}^n$. The collection of class labels of $\mathbf{X}$ is $\mathbf{Z} = \{z_i \in \{1, 2, \dots, C\}\}_{i=1}^n$, where, z_i is the class label corresponding to $\mathbf{x}_i$. For our purposes we represent the class label of $\mathbf{x}_i$ as $\mathbf{t}_i \in \{0, 1\}^C$, where $t_{ik} = 1$ if $z_i = k$ and $t_{ik} = 0$, otherwise. We denote the j th feature by x_j , the class label by z , and the target vector by $\mathbf{t}$. The set of original features is $\mathbf{f} = \{x_1, x_2, \dots, x_P\}$. Let the optimal class-specific subset of features for the k th class be $\mathbf{s}_k$. We need to find out $\mathbf{s}_k$ s where, $\mathbf{s}_k \subset \mathbf{f} \forall k \in \{1, 2, \dots, C\}$. We propose a CSFS mechanism embedded in a fuzzy rule-based classifier (FRBC). So next we discuss the FRBC.

5.2.1 Fuzzy Rule-based Classifiers

We employ the FRBC framework used in [30, 31]. Each class is represented by a set of rules. Let there be n_k rules for the k th class. The l th rule corresponding to the k th class, R_{kl} is given by

$$R_{kl} : \text{If } x_1 \text{ is } A_{1,kl} \text{ and } x_2 \text{ is } A_{2,kl} \text{ and } \dots x_P \text{ is } A_{P,kl} \text{ then } z \text{ is } k. \quad (5.2.1)$$

Here, $k \in \{1, 2, \dots, C\}$, $l \in \{1, 2, \dots, n_k\}$, and $A_{j,kl}$ is a linguistic value (fuzzy set) defined on the j th feature for the l th rule of the k th class. Let, α_{kl} be the firing strength of the rule R_{kl} . The rule firing strength is computed using the product T -norm [191] over the fuzzy sets $A_{1,kl}, A_{2,kl}, \dots, A_{P,kl}$. Let the membership to the fuzzy set $A_{j,kl}$ be $\mu_{j,kl}$. So, α_{kl} is given by the following.

$$\alpha_{kl} = \prod_{j=1}^P \mu_{j,kl}$$

The final output of the FRBC is of the form $\mathbf{o} = (o_1, o_2, \dots, o_C)$, where, o_k is the support for k th class, computed as

$$o_k = \max\{\alpha_{k1}, \alpha_{k2}, \dots, \alpha_{kn_k}\}.$$

Since “max” is not differentiable, a differentiable approximation of max, known as softmax can be used [30, 243]. We may define soft-max with the following formulation [30, 243].

$$\text{softmax}(\alpha_1, \alpha_2, \dots, \alpha_m, \theta) = \frac{\sum_{i=1}^m \alpha_i \exp(\theta \alpha_i)}{\sum_{i=1}^m \exp(\theta \alpha_i)}; \alpha_i \in [0, 1] \quad (5.2.2)$$

As θ tends to ∞ , the softmax function provides the maximum of α_i s. To learn an efficient classifier from the initial fuzzy rule-based system, the parameters defining the fuzzy sets $A_{j,kl}$ s can be tuned by minimizing the loss function,

$$E_{cl} = \sum_{i=1}^n \sum_{k=1}^C (o_{ik} - t_{ik})^2. \quad (5.2.3)$$

To extract the rules, following [30, 31], we cluster the training data of the k th class into n_k clusters. We note here that the k th class may not have n_k clusters in the pattern recognition sense. By clustering we just group the nearby points and then define a rule for each group. Let the centroid of the l th cluster of the k th class be $\mathbf{v}_{kl} = (v_{1,kl}, v_{2,kl}, \dots, v_{P,kl})$. The cluster centroid $\mathbf{v}_{kl}$ is then translated into P fuzzy sets, $A_{j,kl} = \text{“close to” } v_{j,kl} \forall j \in \{1, 2, \dots, P\}$. The fuzzy set ‘“close to” $v_{j,kl}$ ’ is modeled by a Gaussian membership function with mean $v_{j,kl}$.

5.2.2 Feature Selection

Following [27, 30, 31, 59, 60], we use feature modulators that inhibit the derogatory features and promote useful features to take part in the rules of the FRBC. For each feature, there is an associated modulator. We can have different choices of the modulator such as $M(\lambda_j) = \frac{1}{1+\exp(-\lambda_j)}$ or $M(\lambda_j) = \exp(-\lambda_j^2)$, [27, 31]. Here, we use the later one. Thus for each feature j , we use

$$M(\lambda_j) = \exp(-\lambda_j^2), \text{ where } j \in \{1, 2, \dots, P\}.$$

To select or reject a feature using the modulator function, the membership values associated with the j th feature are modified as

$$\hat{\mu}_{j,kl} = \mu_{j,kl}^{M(\lambda_j)} = \mu_{j,kl}^{\exp(-\lambda_j^2)} \forall k, l \quad (5.2.4)$$

Note that, $|\lambda_j| \approx 0$ makes $\hat{\mu}_{j,kl} \approx \mu_{j,kl}$. Similarly, when $|\lambda_j|$ is high (say, $|\lambda_j| \geq 2$), $\hat{\mu}_{j,kl} \approx 1$. The rule firing strength is now calculated as $\alpha_{kl} = \prod_{j=1}^P \hat{\mu}_{j,kl}$. So, when $\hat{\mu}_{j,kl} \approx \mu_{j,kl}$, j th feature influences the rule firing strength α_{kl} and in turn influences the classification process, whereas if $\hat{\mu}_{j,kl} \approx 1$ then the j th feature has no influence on the firing strength and hence no influence on the predictions by the FRBC. This would be true for any T-Norm as $T(x, 1) = x, x \in [0, 1]$. Thus, for useful features, $|\lambda_j|$ s should be made close to zero and for derogatory features, $|\lambda_j|$ s should be made high. The desirable values of λ_j s are obtained by minimizing E_{cl} defined in (5.2.3) with respect to λ_j s. The training begins with $\lambda_j = 1 + \text{Gaussian noise}$. $M(\lambda_j) \approx 0$ indicates a strong rejection of x_j , while $M(\lambda_j) \approx 1$ suggests a strong acceptance of x_j . However, training

may lead λ_j s such that $M(\lambda_j)$ takes a value in between 0 and 1. This implies that the corresponding feature influences the classification partially. This is not desirable in our case, as our primary goal is to select or reject features. To facilitate this, like in Chapter 4, we add a regularizer term E_{select} to E_{cl} such that E_{select} adds a penalty if any λ_j allows the corresponding feature partially. In [31], E_{select} is set as follows.

$$E_{select} = \frac{1}{P} \sum_{j=1}^P \exp(-\lambda_j^2)(1 - \exp(-\lambda_j^2)) \quad (5.2.5)$$

So, the overall loss function for learning suitable λ_j s becomes

$$E = E_{cl} + c_1 E_{select}. \quad (5.2.6)$$

Class-specific feature selection

So far we have not considered the selection of class-specific features. In the class-specific scenario, for each class, a different set of P modulators is engaged. So, a total of $C \times P$ feature modulators are employed. Consequently, for each class a different set of features, if appropriate, may get selected. Here, we represent the feature modulator for the j th feature of the k th class as $M(\lambda_{j,k}) = \exp(-\lambda_{j,k}^2)$, where $j \in \{1, 2, \dots, P\}; k \in \{1, 2, \dots, C\}$. The modulator value $M(\lambda_{j,k})$ modifies the membership values involved in the rules corresponding to the j th feature of the k th class as follows.

$$\hat{\mu}_{j,kl} = \mu_{j,kl}^{M(\lambda_{j,k})} = \mu_{j,kl}^{\exp(-\lambda_{j,k}^2)} \forall l = 1, 2, \dots, n_k. \quad (5.2.7)$$

For this problem, the E_{select} is changed to

$$E_{select} = \frac{1}{CP} \sum_{k=1}^C \sum_{j=1}^P \exp(-\lambda_{j,k}^2)(1 - \exp(-\lambda_{j,k}^2)) \quad (5.2.8)$$

We now minimize (5.2.6) with respect to $\lambda_{j,k}$ s to find the optimal $\lambda_{j,k}$ s.

5.2.3 Monitoring Redundancy

Suppose a data set has three useful features say x_1, x_2, x_3 such that each of x_2 and x_3 is strongly dependent on (say correlated with) x_1 then all the three features carry almost the same information and only one of them is enough. These three features form a redundant set of features. However, if we just use one of them and there is some error in measuring that feature, the system may fail to do the desired job. Therefore, complete elimination of redundancy is not good because then if there is a measurement error in the selected feature, the system will not be able to recover. So, controlled use of redundant features is desirable. For the global feature selection framework, redundancy control has been realized by adding the regularizer (5.2.9) to (5.2.6) [27, 31, 60]:

$$E_r = \frac{1}{P(P-1)} \sum_{j=1}^P \sum_{m=1, m \neq j}^P \sqrt{\exp(-\lambda_j^2) \exp(-\lambda_m^2) \rho^2(x_j, x_m)} \quad (5.2.9)$$

Here, $\rho()$ is Pearson's correlation coefficient (or it could be mutual information also), which is a measure of dependency between two features. When x_j and x_m are highly

correlated (positively or negatively), $\rho^2(x_j, x_m)$ is close to one (its highest value). In this case, to reduce the penalty E_r , the training process will adapt λ_j and λ_m in such a way that one of $\exp(-\lambda_j^2)$ and $\exp(-\lambda_m^2)$ is close to 0 and the other is close to 1. Note the (5.2.9) is not suitable for class-specific scenarios. Next, we change (5.2.9) for class-specific redundancy.

Class-specific redundancy

For class-specific redundancy, we compute $\rho_k(x_j, x_m)$ between features x_j and x_m considering only instances of the k th class. In the class-specific case, for each class, we have P feature modulators, $M(\lambda_{j,k}) = \exp(-\lambda_{j,k}^2)$, where $j \in \{1, 2, \dots, P\}; k \in \{1, 2, \dots, C\}$. So, (5.2.9) is modified as following.

$$E_{r_c} = \frac{1}{CP(P-1)} \sum_{k=1}^C \sum_{j=1}^P \sum_{m=1, m \neq j}^P \sqrt{\exp(-\lambda_{j,k}^2) \exp(-\lambda_{m,k}^2) \rho_k^2(x_j, x_m)} \quad (5.2.10)$$

Considering the class-specific redundancy, our new loss function for learning the system becomes:

$$E_{tot} = E_{cl} + c_1 E_{select} + c_2 E_{r_c}. \quad (5.2.11)$$

5.2.4 Exploiting Substructures Within a Class.

For some real-world problems, the data corresponding to a class may have distinct clusters and different clusters may lie even in different sub-spaces. For example, in a multi-cancer gene expression data set, each cancer may have several sub-types, where each sub-type is characterized by a different set of highly expressed genes/features. This generalizes the concept of class-specific feature selection further. To exploit such local substructures within a class while extracting rules, we need to use rule-specific feature modulators. Each rule of the k th class is assumed to represent a local structure or cluster present in the k th class. So, for the k th class there are $n_k \times P$ feature modulators. For the overall system there are $n_{rule} \times P$ feature modulators where, $n_{rule} (= \sum_{k=1}^C n_k)$ is the total number of rules. A modulator function is now represented by $M(\lambda_{j,kl})$ and the corresponding modulated membership is the following.

$$\hat{\mu}_{j,kl} = \mu_{j,kl}^{M(\lambda_{j,kl})} = \mu_{j,kl}^{\exp(-\lambda_{j,kl}^2)} \quad (5.2.12)$$

The regularizer, E_{select} is now modified as

$$E_{select} = \frac{1}{CP} \sum_{k=1}^C \frac{1}{n_k} \sum_{l=1}^{n_k} \sum_{j=1}^P \exp(-\lambda_{j,kl}^2) (1 - \exp(-\lambda_{j,kl}^2)) \quad (5.2.13)$$

In this framework, we do not consider redundancy. Using (5.2.12) for E_{cl} and (5.2.13) for E_{select} we define the loss function $E = E_{cl} + c_1 E_{select}$ for discovering rule-specific feature subset.

We note that in each iteration, for the CSFS framework, we update CP modulator parameters and for the rule-specific case, we update $n_{rule}P$ modulator parameters. For

both the cases, a total of $2n_{rule}P$ rule-base parameters (mean and spread of the Gaussian memberships) are updated. So, the computational complexity for the proposed method is $O(nn_{rule}P)$. Here, we have not included the cost of initialization of the rule-base as that can be done using different methods including random initialization as mentioned in Chapter 3.

5.3 Experiments and Results

We have done three experiments to validate the three main contributions of our proposed framework.

Experiment 1: We have shown the effectiveness of the proposed CSFS over the usual global feature selection using the proposed FRBC framework.

Experiment 2: We have demonstrated the significance of class-specific redundancy control using our approach.

Experiment 3: A data set having multiple sub-structures in different sub-spaces within a class has been considered to show the utility of our method.

We have not tuned the rule base parameters of the FRBC and have tuned only the feature modulators to select/reject features. For clustering, we have used the c -means algorithm. Each fuzzy set is modeled using a Gaussian membership function having two parameters: center and spread. The cluster centers are used as the centers of the Gaussian membership functions and their spreads have been set as 0.2 times the feature-specific range. To minimize the error functions, we use the optimizer, *train.GradientDescentOptimizer* from *TensorFlow* [193]. For all experiments, the learning rate is set to 0.2 and the stopping criterion is 2000 iterations. However, typically the loss function reduces very rapidly at the beginning of the training and it converges within a few hundred iterations. As mentioned in Sec.5.2 we denote the class-specific feature subset for class 1 as $\mathbf{s}_1$, for class 2 as $\mathbf{s}_2$ and so on.

5.3.1 Experiment 1

We use a three-class synthetic data set, Synthetic1, with six features having distributions as described in Table 5.1. Here, $\mathcal{N}(m, s)$ represents a normal distribution with

Table 5.1: Description of the dataset Synthetic1

Instances	Features						Class
	x_1	x_2	x_3	x_4	x_5	x_6	z
$\mathbf{x}_1 \cdots \mathbf{x}_{100}$	$\mathcal{N}(0, 0.5)$	$\mathcal{N}(0, 0.5)$	$\mathcal{U}(-10, 10)$	$\mathcal{U}(-10, 10)$	$\mathcal{U}(-10, 10)$	$\mathcal{U}(-10, 10)$	1
$\mathbf{x}_{101} \cdots \mathbf{x}_{200}$	$\mathcal{U}(-10, 10)$	$\mathcal{U}(-10, 10)$	$\mathcal{N}(0, 0.5)$	$\mathcal{N}(0, 0.5)$	$\mathcal{U}(-10, 10)$	$\mathcal{U}(-10, 10)$	2
$\mathbf{x}_{201} \cdots \mathbf{x}_{300}$	$\mathcal{U}(-10, 10)$	$\mathcal{U}(-10, 10)$	$\mathcal{U}(-10, 10)$	$\mathcal{U}(-10, 10)$	$\mathcal{N}(0, 0.5)$	$\mathcal{N}(0, 0.5)$	3

mean, m and standard deviation, s ; $\mathcal{U}(a, b)$ represents a uniform distribution over the interval (a, b) . Without loss of generality, we have assigned the first 100 points to class 1, the next 100 points to class 2, and the last 100 points to class 3. From Table 5.1

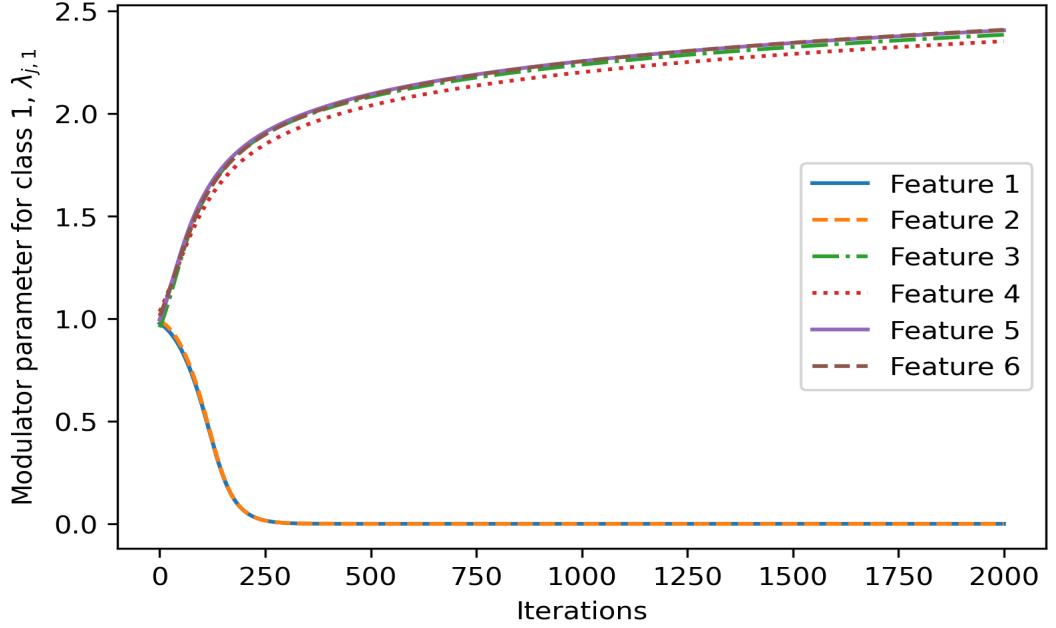


Figure 5.1: Variations of the class-specific feature modulation parameters $\lambda_{j,k}$ s for class 1 over iterations.

we can see that class 2 and class 3 are uniformly distributed over a given interval for features x_1 and x_2 . On the other hand, class 1 is clustered around $(0, 0)$ in the feature space formed by x_1 and x_2 . Hence the feature space formed by x_1 and x_2 , discriminates class 1 from the other two classes. Similarly, (x_3, x_4) and (x_5, x_6) discriminates class 2 and class 3 respectively, from the corresponding remaining classes. To understand the importance of CSFS, we perform global feature selection (GFS) and CSFS and compare their performances. We have also computed the performance of the FRBC with all features. The number of rules considered per class is one. We have conducted 5 runs for each of the FRBCs.

Figures 5.1, 5.2, and 5.3 depict the variations of the class-specific feature modulation parameters with iterations in run 1 of learning the FRBCs, for class 1, for class 2, and for class 3, respectively. As we have mentioned in Subsec. 5.2.2, when the absolute value of a feature modulator parameter (λ_j or $\lambda_{j,k}$) tends to zero, in our framework, the corresponding feature is selected. It is seen from Fig. 5.1, that for class 1, the absolute value of the modulator parameters for feature 1 (x_1) and feature 2 (x_2), i.e. $|\lambda_{1,1}|$ and $|\lambda_{2,1}|$ tend to zero i.e., our CSFS scheme effectively selects x_1 and x_2 for class 1 which are actually discriminative features for class 1. Similarly, from Figs. 5.2 and 5.3, it is evident that our framework selects x_3 and x_4 for class 2 and x_5 and x_6 for class 3, respectively. We have discussed previously that x_3 and x_4 are discriminative features for class 2 and x_5 and x_6 are discriminative features for class 3. So, our proposed CSFS successfully selects features that are most discriminative for a class in its respective

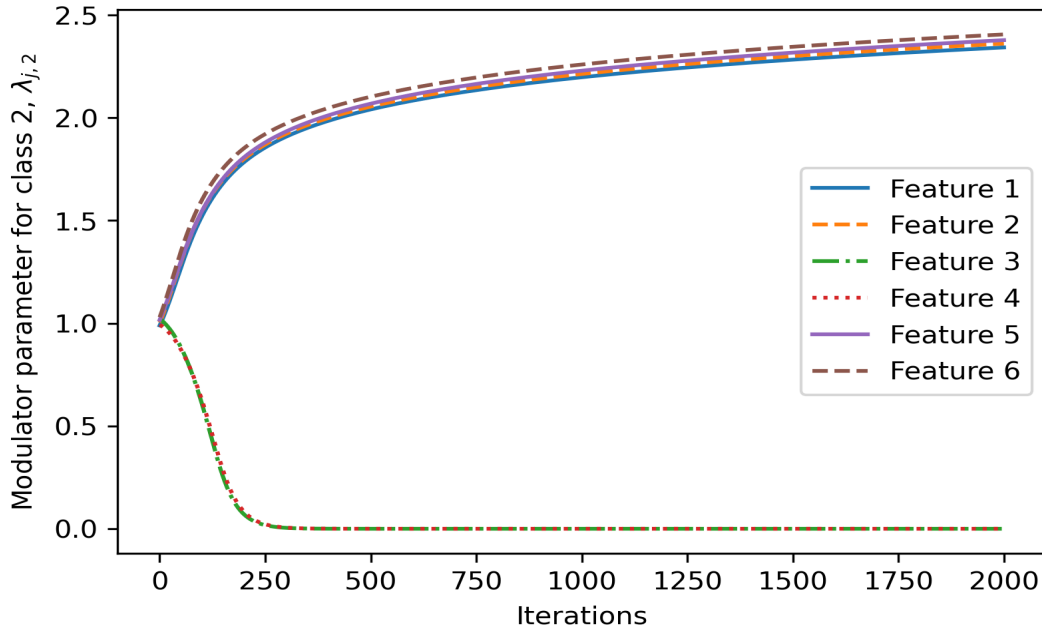


Figure 5.2: Variations of the class-specific feature modulation parameters $\lambda_{j,k}$ s for class 2 over iterations.

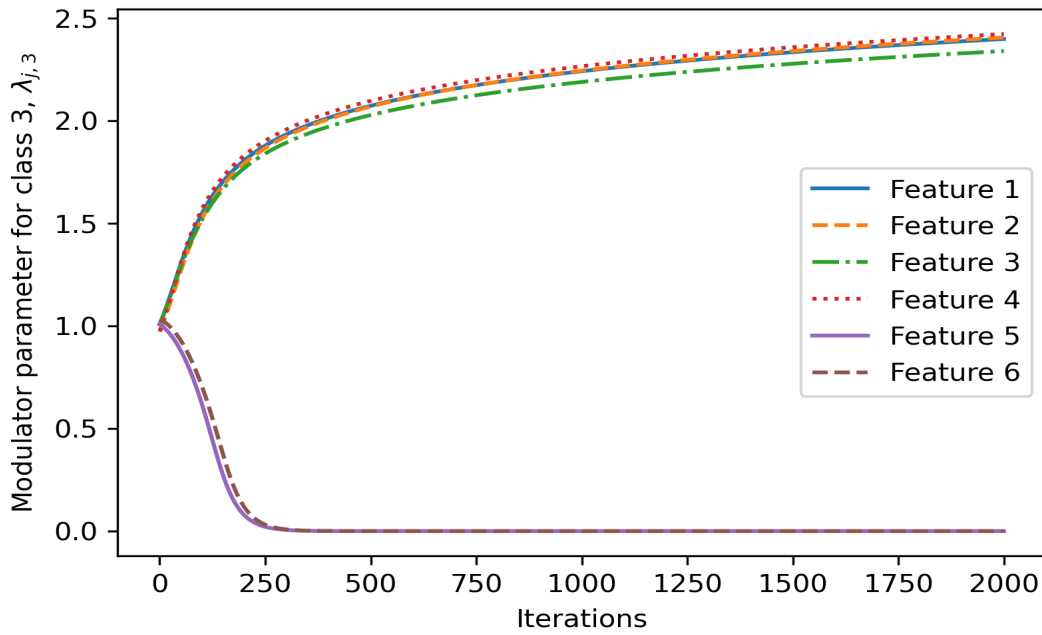


Figure 5.3: Variations of the class-specific feature modulation parameters $\lambda_{j,k}$ s for class 3 over iterations.

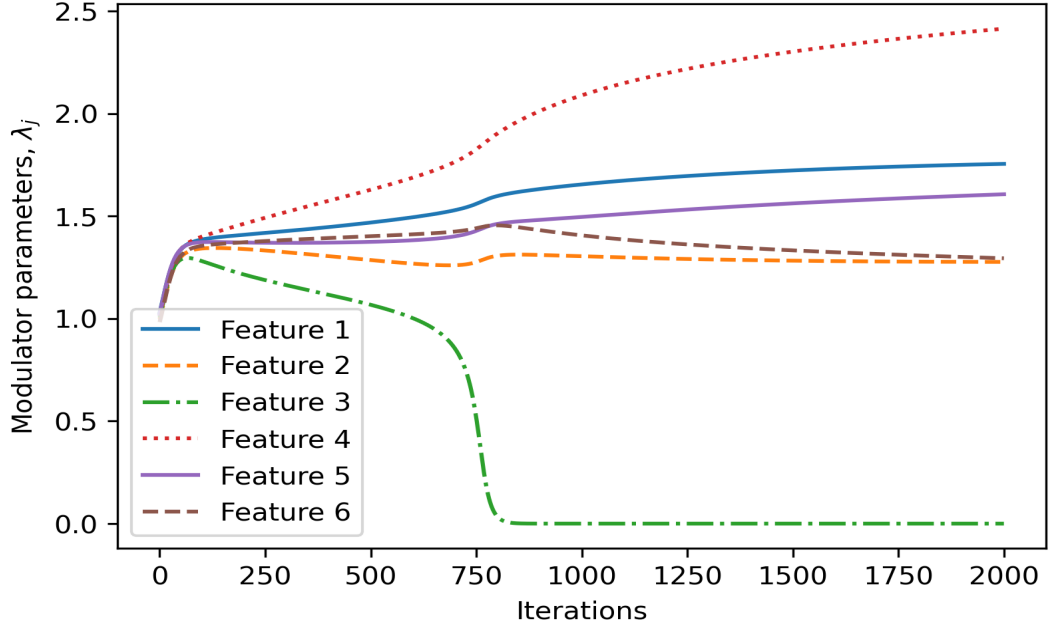


Figure 5.4: Variations of the global feature modulation parameters λ_j s over iterations.

class-specific subsets.

Figure 5.4 depicts the variations of the (global) feature modulation parameters in run 1 of learning the FRBCs, λ_j s versus iterations. In the traditional feature selection approach using FRBC, a single feature subset is selected. From Fig. 5.4, it is evident the traditional feature selection scheme using FRBC selects only x_3 .

In Table 5.2, we record the subsets of the selected features as well as classification accuracies for our proposed CSFS scheme and the global feature selection scheme using the same FRBC framework, and also the classification accuracy achieved in the FRBC using all features. In all five runs the proposed CSFS selects the characteristic features

Table 5.2: Performance on the dataset, Synthetic1

Run	Features selected		Avg. accuracy of FRBC (%)		
	Class-specific	Global	Class-specific	Global	All features
1-5	$s_1 : x_1, x_2$; $s_2 : x_3, x_4$; $s_3 : x_5, x_6$	x_3	98.7	45.3	62.1

for each class (i.e. x_1, x_2 for class 1, and so on). The FRBC with the class-specific selected features has achieved an average accuracy of 98.7% - in fact, each run achieved the same accuracy. In GFS, the selected subset is x_3 . The FRBC using a globally selected feature subset has achieved an accuracy of 45.3% in each of the five runs. One can argue that the class-specific model uses all six features and performs better than the global model, which uses one feature. But, when we learn the FRBC rules using all six features it has achieved an average accuracy of 62.1% over the five runs.

The importance of class-specific feature selection is clearly established through this experiment.

5.3.2 Experiment 2

For Experiment 2, we have considered another synthetic dataset, Synthetic2, produced by appending two additional features x_7 and x_8 to Synthetic1 data set. For class 1, x_7 and x_8 are generated as $x_1 + \mathcal{N}(0, 0.1)$ and $x_2 + \mathcal{N}(0, 0.1)$, respectively. For the other two classes, x_7 and x_8 are generated from $\mathcal{U}(-10, 10)$ and $\mathcal{U}(-10, 10)$, respectively. To illustrate the dependency of the variables we generate four feature correlation matrices. In all four Tables 5.3, 5.4, 5.5, and 5.6, any entry corresponding to the row x_j and column x_k is the Pearson correlation coefficient value between those two features. Table 5.3 shows the correlation among the eight features considering the whole data, which we do conventionally. From Table 5.3, we observe that no two features are strongly correlated; the maximum absolute correlation value between two features is 0.1323 which is low. Tables 5.4, 5.5, and 5.6 record correlation among the eight features

Table 5.3: Feature correlation matrix with the whole Synthetic2 data

variables	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
x_1	1	-0.0300	-0.0170	-0.0162	-0.0111	-0.0094	0.1323	0.0558
x_2	-0.0300	1	-0.0236	0.0227	0.1250	0.0144	0.0576	-0.0734
x_3	-0.0170	-0.0236	1	0.0308	0.0455	0.0422	0.0454	-0.0034
x_4	-0.0162	0.0227	0.0308	1	-0.0947	-0.0733	0.0305	-0.1059
x_5	-0.0111	0.1250	0.0456	-0.0947	1	-0.0142	-0.0484	-0.0654
x_6	-0.0094	0.0144	0.0422	-0.0733	-0.0142	1	-0.0581	-0.0534
x_7	0.1323	0.0576	0.0454	0.0305	-0.0484	-0.0581	1	0.0211
x_8	0.0558	-0.0734	-0.0034	-0.1059	-0.0654	-0.0534	0.0211	1

of the Synthetic2 data set considering only points from class 1, class 2, and class 3, respectively. In Table 5.4, which is calculated based on instances from class 1 of

Table 5.4: Feature correlation matrix with the class 1 of Synthetic2 data

variables	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
x_1	1	0.0475	-0.0685	-0.0409	-0.0483	-0.0553	0.9849	0.0649
x_2	0.0475	1	-0.1331	0.0291	-0.0443	-0.0385	0.0423	0.9859
x_3	-0.0685	-0.1331	1	-0.0705	0.0953	0.0870	-0.0877	-0.1157
x_4	-0.0409	0.0291	-0.0705	1	-0.1701	-0.1481	-0.0189	0.0146
x_5	-0.0483	-0.0443	0.0953	-0.1701	1	-0.0402	-0.0467	-0.0425
x_6	-0.0553	-0.0385	0.0870	-0.1481	-0.0402	1	-0.0469	-0.0344
x_7	0.9849	0.0423	-0.0877	-0.0189	-0.0467	-0.0469	1	0.0580
x_8	0.0649	0.9859	-0.1157	0.0146	-0.0425	-0.0344	0.0580	1

Synthetic2 data set, we observe that x_1 is highly correlated to x_7 with a correlation value of 0.9849 and x_2 is highly correlated to x_8 with a correlation value of 0.9859.

However, in Tables 5.5 and Table 5.6, which are calculated based on instances from class 2 and class 3 of Synthetic2, respectively, no feature pair has high correlation value.

We note that x_7 is dependent on x_1 and x_8 is dependent on x_2 for class 1 but the

Table 5.5: Feature correlation matrix with the class 2 of Synthetic2 data

variables	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
x_1	1	-0.0424	0.1045	-0.0470	-0.0524	-0.0141	0.2504	0.0732
x_2	-0.0424	1	0.1298	-0.0677	0.2713	0.0336	-0.0147	-0.0998
x_3	0.1045	0.1298	1	-0.0473	0.0970	-0.1600	0.1694	0.2603
x_4	-0.0470	-0.0677	-0.0473	1	-0.0912	-0.1553	0.0295	-0.0107
x_5	-0.0524	0.2713	0.0970	-0.0912	1	0.0195	-0.0891	-0.1383
x_6	-0.0141	0.0336	-0.1600	-0.1553	0.0195	1	-0.1136	-0.1115
x_7	0.2504	-0.0147	0.1694	0.0295	-0.0891	-0.1136	1	0.0578
x_8	0.0732	-0.0998	0.2603	-0.0107	-0.1383	-0.1115	0.0578	1

Table 5.6: Feature correlation matrix with the class 3 of Synthetic2 data

variables	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
x_1	1	-0.0212	-0.0296	-0.0179	0.1305	0.0788	0.0075	0.0350
x_2	-0.0212	1	-0.0417	0.0517	-0.1101	-0.0243	0.1352	-0.0569
x_3	-0.0296	-0.0417	1	0.1271	-0.0226	0.1592	0.0894	-0.0202
x_4	-0.0179	0.0517	0.1271	1	0.0363	0.0444	0.0594	-0.2123
x_5	0.1305	-0.1101	-0.0226	0.0363	1	-0.0002	-0.0261	-0.0059
x_6	0.0788	-0.0243	0.1592	0.0444	-0.0002	1	0.0716	0.1659
x_7	0.0075	0.1352	0.0894	0.0594	-0.0261	0.0716	1	-0.0152
x_8	0.0350	-0.0569	-0.0202	-0.2123	-0.0059	0.1659	-0.0152	1

remaining two classes are indiscernible among themselves considering features x_7 and x_8 . Clearly, features x_7, x_8 are also discriminatory for class 1. However, do x_7, x_8 add any information over x_1, x_2 for class 1? The answer is no, as for class 1, x_7 and x_8 are noisy versions of x_1 and x_2 , respectively. This feature-redundancy is specific to class 1. In Table 5.7 we have described the performances of the FRBCs in the CSFS framework without and with class-specific redundancy control. Here also, we have set

Table 5.7: Class-specific feature selection on Synthetic2 data set

Run	Without redundancy control		With class-specific redundancy control	
	Selected features	Acc.	Selected features	Acc.
1	$\mathbf{s}_1:x_1, x_2, x_7, x_8; \mathbf{s}_2:x_3, x_4; \mathbf{s}_3:x_5, x_6$	99.3	$\mathbf{s}_1:x_2, x_7; \mathbf{s}_2:x_3, x_4; \mathbf{s}_3:x_5, x_6$	99.3
2	$\mathbf{s}_1:x_1, x_2, x_7; \mathbf{s}_2:x_3, x_4; \mathbf{s}_3:x_5, x_6$	99.3	$\mathbf{s}_1:x_1, x_8; \mathbf{s}_2:x_3, x_4; \mathbf{s}_3:x_5, x_6$	99
3	$\mathbf{s}_1:x_1, x_2, x_7, x_8; \mathbf{s}_2:x_3, x_4; \mathbf{s}_3:x_5, x_6$	99.3	$\mathbf{s}_1:x_7, x_8; \mathbf{s}_2:x_3, x_4; \mathbf{s}_3:x_5, x_6$	99.3
4	$\mathbf{s}_1:x_1, x_2, x_7, x_8; \mathbf{s}_2:x_3, x_4; \mathbf{s}_3:x_5, x_6$	99.3	$\mathbf{s}_1:x_7, x_8; \mathbf{s}_2:x_3, x_4; \mathbf{s}_3:x_5, x_6$	99.3
5	$\mathbf{s}_1:x_1, x_2, x_7, x_8; \mathbf{s}_2:x_3, x_4; \mathbf{s}_3:x_5, x_6$	99.3	$\mathbf{s}_1:x_1, x_7; \mathbf{s}_2:x_3, x_4; \mathbf{s}_3:x_5, x_6$	99

the number of fuzzy rules per class as one and repeated the experiments five times with

each model. The term ‘Acc.’ in Table 5.7 refers to accuracy of the FRBC in percentage. For class 1, features x_1 and x_7 are heavily dependent. Hence, to avoid redundancy only one of them should be selected. The same argument is true for features x_2 and x_8 . Minimizing (5.2.11) which uses (5.2.10) to control class-specific redundancy, the FRBC has successfully chosen only one from x_1 and x_7 and one from x_2 and x_8 to include in $\mathbf{s}_1$ in all five runs (Table 5.7). On the other hand, we observe that without any redundancy control, the class-specific feature selection framework selects all the four discriminatory features to include in $\mathbf{s}_1$ in four runs. The best accuracy achieved by the CSFS framework without redundancy control and that of CSFS with class-specific redundancy control are the same and equal to 99.3% although the later selects only two features. This establishes the benefit of class-specific redundancy control.

5.3.3 Experiment 3

In experiment 3, we validate our proposed framework for handling the presence of different clusters in different sub-spaces within a class. We have generated Synthetic3, a two-class data having four features where each class is composed of two distinct clusters lying in two different sub-spaces. The data set Synthetic3 is described in Table 5.8. Each class has 200 points. For class 1, instances 1 to 100 create a distinct cluster

Table 5.8: Description of the dataset Synthetic3

Instances	Features				Class
	x_1	x_2	x_3	x_4	
$\mathbf{x}_1 \cdots \mathbf{x}_{100}$	$\mathcal{N}(0, 0.5)$	$\mathcal{N}(-5, 0.5)$	$\mathcal{U}(-10, 10)$	$\mathcal{U}(-10, 10)$	1
$\mathbf{x}_{101} \cdots \mathbf{x}_{200}$	$\mathcal{U}(-10, 10)$	$\mathcal{U}(-10, 10)$	$\mathcal{N}(0, 0.5)$	$\mathcal{N}(0, 0.5)$	1
$\mathbf{x}_{201} \cdots \mathbf{x}_{300}$	$\mathcal{U}(-10, 10)$	$\mathcal{N}(0, 0.5)$	$\mathcal{N}(-5, 0.5)$	$\mathcal{U}(-10, 10)$	2
$\mathbf{x}_{301} \cdots \mathbf{x}_{400}$	$\mathcal{N}(5, 0.5)$	$\mathcal{U}(-10, 10)$	$\mathcal{U}(-10, 10)$	$\mathcal{N}(-5, 0.5)$	2

around $(0, -5)$ in the feature space formed of x_1, x_2 and instances 101 to 200 create a distinct cluster around $(0, 0)$ in the feature space formed of x_3, x_4 . Similarly, class 2 also has two distinct clusters in the feature spaces formed of x_2, x_3 and x_1, x_4 respectively. To handle this dataset we employ our proposed rule-specific approach implemented using (5.2.12), and (5.2.13). The number of rules per class is set to two.

As observed from Table 5.9, the rule-specific feature selection is successful in identifying the two important sub-spaces, i.e., x_1, x_2 and x_3, x_4 for class 1 and x_2, x_3 and x_1, x_4 for class 2. We note that for both the classes the selected rule-specific subsets interchange between rules 1 and 2 because the cluster number assignment to different groups of points for a class varies. We also note from Table 5.9, using the class-specific feature selection method, in different runs, $\mathbf{s}_1$ comprises of x_1 or x_2 and $\mathbf{s}_2$ comprises of x_2 or x_1, x_2 . These subsets obviously do not characterize the classes correctly. The average accuracy of the FRBC using feature subsets selected by rule-specific, class-specific feature selection and using all features are 100%, 77.4%, and 87.5%, respectively. This

Table 5.9: Features subsets selected for Synthetic3

Run	Rule-specific		Class-specific	
	Class 1	Class 2	s_1	s_2
1	rule 1: x_1, x_2 ; rule 2: x_3, x_4	rule 1: x_2, x_3 ; rule 2: x_1, x_4	x_1	x_1, x_2
2	rule 1: x_1, x_2 ; rule 2: x_3, x_4	rule 1: x_1, x_4 ; rule 2: x_2, x_3	x_2	x_1, x_2
3	rule 1: x_3, x_4 ; rule 2: x_1, x_2	rule 1: x_1, x_4 ; rule 2: x_2, x_3	x_2	x_2
4	rule 1: x_3, x_4 ; rule 2: x_1, x_2	rule 1: x_1, x_4 ; rule 2: x_2, x_3	x_2	x_1, x_2
5	rule 1: x_3, x_4 ; rule 2: x_1, x_2	rule 1: x_2, x_3 ; rule 2: x_1, x_4	x_2	x_2

demonstrates the usefulness of our proposed RSFS framework.

Note that, here, we chose the number of rules based on the prior knowledge of the data sets because our goal was to demonstrate the utility and effectiveness of class-specific and rule-specific feature selection. For real world data sets where such knowledge may not be available we can choose the number of rules that provides the minimum classification error over a validation set.

5.4 Conclusion

In this work, first, we have proposed a class-specific feature selection scheme using feature modulators embedded in an FRBC. The parameters of the feature modulators are tuned by minimizing a loss function comprising of classification error and a regularizer to make the modulators completely select or reject features. This framework is used in [30, 31] for selecting globally useful features. We have generalized it to make it suitable for CSFS. Our proposed CSFS method does not employ an OVA strategy like most of the existing CSFS works and hence is free from the enhanced computational overload and hazards associated with the OVA based methods. We have extended the CSFS scheme so that it can monitor class-specific redundancy by adding a suitable regularizer. Finally, our CSFS framework is generalized to a rule-specific feature selection framework to handle the presence of multiple sub-space-based clusters within a class. There are problems like diagnosis of cancer subtypes or stages, where a particular type of cancer may have different stages, each of which is characterized by different groups of genes (maybe with or without overlap). The proposed rule-specific scheme can discover the genes that characterize the subtypes as well as provide human interpretable rules for diagnosis. For such applications, interpretability is important. The three approaches proposed are validated through three experiments on appropriate synthetic data sets.

Chapter 6

Conclusions and future scopes

6.1 Conclusions

In this thesis, we have addressed certain gaps and drawbacks in some of the existing dimensionality reduction methods, primarily aiming at data visualization and classification. Dimensionality reduction is generally done through feature extraction and feature selection. This thesis covers both approaches. In our study, we primarily use two computational intelligence tools, neural networks and fuzzy rule-based systems. In Chapter 2 and Chapter 3, we have proposed feature extraction-based dimensionality reduction methods. To be specific we have addressed some issues related to some common manifold learning algorithms, a particular category of feature extraction methods. In these two chapters, (Chapters 2 and 3), we have dealt with unlabeled data, and the primary task is data visualization. In Chapter 4 and Chapter 5, we have proposed feature selection based dimensionality reduction methods. In these chapters (Chapters 4 and 5), we deal with labeled data, specifically, data having class labels. Naturally, classification is the primary task for the feature selection schemes described in Chapters 4 and 5. However, the feature selection schemes introduced in these two chapters achieve some additional goals along while selecting class discriminative features.

In Chapter 2, we have proposed a general supervised framework for learning explicit maps for manifold learning-based dimensionality reduction methods. Note that, the popular manifold learning methods proposed in [42, 43, 77, 81, 82, 134] do not learn any explicit function for projecting high dimensional data to a lower dimensional one in their original form. Our proposed method equips an implicit manifold learning method with explicit maps and solves a few relevant problems. First, to generate output for test points, an implicit method repeats its whole process considering the training and the test points together demanding enhanced computational cost. Moreover, this demands the availability of the training data at the time of projecting a new test point. All these make such methods unsuitable for real-time applications. With the explicit maps learned according to our proposed method, the computational cost associated with projecting new/unseen test points is $O(1)$, and the requirement and hence overhead for making the training set available is eliminated. Second, for implicit methods,

generalization ability cannot be analyzed, which can be done for explicit maps. Third, there is no provision for constructing inverse mapping functions in the case of implicit methods. However, with explicit maps, sometimes it is possible to determine an inverse map at least locally. The availability of an inverse map may determine pre-image for novel output points and help to get deeper insight regarding the mapping. With the help of the inverse map, it is possible to obtain a reconstructed version of the original inputs from the lower dimensional outputs. The dissimilarity between the original and reconstructed input helps to analyze the information preservation capability of the underlying mapping method [10, 70].

The proposed method of learning explicit maps is a supervised one so, it needs target data. Here, the target data are the outputs from any one of the implicit manifold learning methods which are unsupervised. Our goal is to finally produce a lower dimensional representation that preserves the local neighborhood structure. It is almost impossible to guarantee that a given unsupervised method of manifold learning will preserve the neighborhood structure for all data sets or even for the same data set, the neighborhood structure will be preserved over the entire data set. Many of the manifold learning algorithms are based on graph spectral decomposition of the neighborhood graphs. These methods sometimes introduce some structural distortions or disagreement of the neighborhood relationships between input and output, due to their nature or improper choice of the neighborhood size [156]. So, when taking the outputs of those methods as the target outputs in a regression-based model, even for the best fit model those distortions/ disagreements would propagate in the output generated by the model. We have demonstrated in Subsections 2.3.2 and 2.4.1 that even for a simple three-dimensional synthetic data set the neighborhood preservation may not be uniform over the entire data set - for a part of the data set the algorithm may do an outstanding job while for some other part, there may be significant distortions from the desired projection. On the other hand, except for synthetic data sets, it is practically very difficult, if not impossible, to assess without post-processing whether the actual topology of the data is preserved in the lower dimension or not.

The proposed approach, not only tries to find a faithful mapping from the input dimension to a lower dimension but also tries to remove any distortion of local neighborhood structure, injected in the target data. To achieve this we have proposed a set of suitable regularizers that are added to the loss function of the regression model. Due to the regularizer, the method rectifies the structural distortion present in the target data to a reasonable extent in the projections made by our proposed system. As the regression model, we have considered polynomials and multi-layer perceptron networks. We explored three variants of the regularizers. The method is validated on synthetic as well as on image data set considering three different algorithms for the generation of the target data. For image data, typically the input dimension is high, which leads to polynomials with a large number of coefficients even for polynomials of degrees two or three. Thus, for a faithful estimation of the parameters not only we need large training set but also it will demand a huge computational overhead. To address this issue, we have exploited the spatial characteristics of image pixels, which reduces the number

of learnable parameters significantly. Although we have considered polynomials and multi-layer perceptron networks, the framework is quite general and other tools such radial basis functions network can also be used.

In Chapter 2 we have proposed a method that learns explicit maps for implicit manifold learning methods through supervision. For any other supervised learning based method including our method proposed in Chapter 2, the availability of the target data is indispensable, which limits the applicability of such methods. This brings out the importance of learning explicit maps for manifold learning in unsupervised learning settings. As discussed earlier in Chapter 3, some unsupervised data projection methods have been extended for dealing with out-of-sample points. A problem with such out-of-sample extension methods is that they *always* produce some output given *any* input, even when the test input is far from the sampling window of the training data. In other words, such methods cannot deal with the “open world” [244, 245] nature of the problem; they do not know when to refuse to make decisions.

In Chapter 3, we have proposed an unsupervised approach to learn an explicit map for manifold learning employing Takagi-Sugeno type fuzzy rule based system (FRBS) that addresses the previously mentioned issues in the context of manifold learning or in general non-linear dimensionality reduction based data visualization methods. FRBSs are used for various machine learning tasks but unfortunately, are almost unexplored for manifold learning or data visualization. The proposed method learns the FRBS that projects the high dimensional datasets into a lower dimensional space, (here we use a two-dimensional space as it is suitable for visualization) preserving the local neighborhood structure. Our method does not use any target data. We have experimented on three synthetic and three real world data sets. We have compared the results of the proposed method with several well-established manifold learning or data visualization methods namely, Sammon’s method [41], ISOMAP [42], LLE [43], *t*-SNE [45], UMAP [47], Diffusion net [28], autoencoder [24], and SNPPE [83]. The results from the experiments have established several advantages of the proposed method that we list in the following. (i) Comparing the visualizations produced by the proposed method with other methods for all the datasets, we observe that the proposed system is the only one which performs consistently well on all the data sets tested. (ii) With the proposed method the out-of-sample instances can be projected in a straightforward manner. The predictability of the proposed system is validated using three types of experiments involving both synthetic and real-world data sets. The quality of the produced outputs for out-of-sample points is better or comparable to the other methods. (iii) It can refuse to project an out-of-sample instance, when it is far away from the sampling window of the training data, i.e, the system is equipped with an appropriate reject option, without requiring the training data. Thus, it can deal with the “open-world” problem. Most machine learning systems cannot do so. (iv) The system is reasonably transparent as it is based on comprehensible fuzzy rules and fuzzy reasoning, where each rule-antecedent represents a small hyper-ellipsoidal region in the input space and its consequent is a local linear function of the inputs. (v) It provides a general framework that can use different objective functions. (vi) The learning algorithm is found to be quite robust

with respect to initialization.

To the best of our knowledge, a feature selection method that simultaneously cares about class discrimination and structure preservation is not available in the literature. In Chapter 4, we have tried to bridge this gap by proposing a neural network-based feature selection method that focuses both on class discrimination and structure preservation. To learn the proposed system, we use Sammon's stress as a regularizer to the classification loss. Any other cluster-structure preserving regularizer can also be used. Although Sammon's stress is quite effective, for large data sets (large number of instances) it is computationally expensive. To deal with this issue we have proposed a sample-based approximation of Sammon's stress. The experimental results indicate that the proposed sample-based regularizer is a computationally efficient approximation of Sammon's stress for data sets having a large number of instances. The approximation is also effective in terms of performance. Our proposed feature selection scheme is generic. So we have investigated its effectiveness for typical numerical datasets used for assessing classifiers as well as for a specialized case: band selection in hyperspectral images(HSI). We have applied the feature selection scheme on popular real-world general datasets and compared its performance with three feature selection schemes. In the context of band selection, we have applied our method to three well-known HSI datasets and compared performances with three band selection methods based on the previously mentioned feature selection schemes. Based on our experiments, we conclude that the proposed feature/band selection method is able to produce reasonably good classification and clustering scores in the majority of the data sets, proving that the proposed method is capable of selecting a subset of features that is good both for classification and clustering. Our scheme provides a mechanism to control the number of selected features. The proposed method is easily extendable to other networks like Radial Basis Function (RBF) network.

In Chapter 5, we have proposed class-specific feature selection (CSFS) schemes using feature modulators embedded in a fuzzy rule based classifier (FRBC). The parameters of the feature modulators are tuned by minimizing a loss function comprising of classification error and a regularizer to make the modulators completely select or reject features in a class-specific or (fuzzy) rule-specific manner. Our proposed CSFS method does not employ a one-versus-all (OVA) strategy like most of the existing CSFS works and hence it is free from the enhanced computational overload and data-imbalance hazard associated with the OVA based methods. We have extended the CSFS scheme so that it can monitor the level of redundancy in the selected features by a given class by adding a suitable regularizer. Finally, our CSFS framework is generalized to a rule-specific feature selection framework to handle the presence of multiple sub-space-based clusters within a class. The three approaches proposed are validated through three experiments on appropriate synthetic data sets.

It is important to note that for the first two contributory chapters (Chapter 2 and Chapter 3), the proposed dimensionality reduction schemes are for data visualization. So, for both cases the output dimension is taken to be two. The output dimension can also be taken as three, but beyond that we cannot visualize the lower-dimensional data.

Both for Chapter 2 and Chapter 3, although the lower dimensional data provide very useful information about the structure of the high dimensional data, these projected data are not interpretable. On the other hand, in Chapter 4 and Chapter 5, we select features keeping in view the clustering and classification problems. In these cases the data in the lower dimension retain the same semantic of the features in the original high dimension.

6.2 Limitations and future scopes

The methods we have proposed in Chapters 2, 3, 4, and 5 to bridge some gaps and drawbacks of some of the existing dimensionality reduction methods, also have some limitations. Next, we discuss those limitations and possible scopes of future works.

In Chapter 2, we have considered three variants of regularizers for adding to the loss function of the regression model to remove any distortion of the local neighborhood structure injected in the target data. As observed from Figs. 2.3, 2.4, 2.6, and 2.7, for all three data sets, for all three variants of the regularizers we have used, the generated embeddings are scaled down by some factor. This is because the regularizer term in (2.2.11) does not impose any constraint on the magnitude. The function (2.2.11) tries to put the proximate points in the input space nearby each other in the projected space, but on the other hand, (2.2.11) is also reduced when the projected points are squeezed into a small hyper-box. To avoid this, we should explore adding a proper constraint on the magnitude of the projected vectors in the loss function governing the regression. We have already mentioned that the work of Chapter 2 provides a general framework of a supervised model for learning explicit mappings associated with any implicit manifold learning method. We have explored polynomials and multi-layer perception based neural networks to model the explicit maps but we could use any non-linear functions for example, fuzzy rule based systems and radial basis function based neural networks. The three regularizers that we have explored for removing any distortion of local neighborhood structures present in the target data are based on the Laplacian regularizer (2.2.11) with three different edge weighting functions of the neighborhood graph as presented in (2.2.12), (2.2.13), and (2.2.14). We may explore the impact of other neighborhood preserving functions [246, 247] as regularizers in our proposed framework in future studies.

Like several other data projection methods, for the method proposed in Chapter 3, both the time and space complexities are quadratic functions of the number of data points, n . So for a very large n , the computational overhead will be high. The use of GPUs can reduce the effective complexities significantly. Moreover, a proper sampling scheme [152] that produces a subset of the original data, which adequately represents the original data distribution could be used. In our future study, we want to explore learning the proposed framework with representative samples of the training set, instead of the full training set. The firing strength of a fuzzy rule is computed as the T-norm of the memberships of the atomic antecedent clauses in the rule. In Chapter 3, we have used the product as the T-norm. So for a very high-dimensional data, it is

possible that for some instances no rule fires with a significant strength leading to an undesirable situation. Because of this limitation, in Chapter 3 we have used datasets with a maximum input dimension of 1024. However, in future studies, we could employ adaptive softmin, a scheme proposed in [33] to overcome the limitation associated with traditional T-norms encountered while dealing with very high-dimensional data sets in fuzzy rule based systems. Like several other manifold learning schemes, the proposed scheme is sensitive to the number of neighbors that is used to construct the neighborhood graph for computing geodesic distances. Following [83] we have chosen the number of nearest neighbors, $\epsilon = 1\%$ of the training samples (rounded to the nearest integer). However, this scheme may not be optimal for modeling the underlying manifold using the corresponding neighborhood graph for all data sets. In future studies, we plan to use an adaptive neighborhood selection mechanism as used in [248] to construct the neighborhood graph. In our study, the number of fuzzy rules has been chosen empirically. The choice of the number of fuzzy rules is related to the number of clusters we have chosen for clustering the training data. We have not explored the possibilities of using cluster validity indices or other schemes to decide on the optimal number of clusters as this was not the primary objective of this study. Moreover, a cluster validity index helps to find the ‘optimal’ number of clusters in the pattern recognition sense. But here even if the data do not have any cluster in the pattern recognition sense, we can use clustering to find the initial rule-base. However, for a given data set, there may be a desirable number of rules, which may simplify the system and improve its performance further. A detailed investigation to find out the optimal number of rules using a validation data set is in the future scope of this study. The geodesic c -Means clustering algorithm that we have used, depends on the construction of a neighborhood graph using the input data points based on the Euclidean distance. For very high dimensional data, the GCM algorithm may suffer from the curse of dimensionality as Euclidean distance does not behave properly in a very high dimensional space [196]. However, it is worth noting that methods like LLE, Laplacian Eigenmap, and ISOMAP, which use neighborhoods defined using Euclidean distance also suffer from the same problem for high dimensional data. For non-image data, a possible approach to address this problem can be to project the data onto a space having moderate dimensions using an unsupervised feature selection scheme or using a parametric feature extraction scheme that does not influence the intrinsic manifold structure much, but eliminates redundant features or information. For image data, if the size of the image is big, it will also result in high dimensional inputs. But here we cannot use feature (pixel) selection as that will not be effective. However, we may use feature embedding techniques like an auto-encoder to reduce the size of the image. Then we can develop the proposed fuzzy rule-based system in the latent space / reduced space. We have restricted our work to data sets lying on a single manifold. Through neighborhood graph construction we can only approximate the geodesic distance over a single manifold or overlapping manifolds. But if the data lie in two (or more) widely separated manifolds, using a reasonable number of nearest neighbors we may not get a connected neighborhood graph. We have not proposed any technique to approximate

geodesic distance over a multi-component neighborhood graph. These issues are kept for future investigation.

In Chapter 4, we have proposed a feature selection scheme that chooses features which perform reasonably well for both classification and clustering tasks. This feature selection method would be especially beneficial over the state of art feature selection methods if the data sets have significant differences between class and cluster structures. In future studies, we would perform investigations on this type of data set. We note that for this type of data set maximizing classification performance and maximizing clustering performance may be conflicting in nature. In our future studies, we plan to adopt the multi-objective approach based feature selection methods proposed in [120, 200, 249, 250] with classification performance and clustering performance as the two objectives.

Finally, in Chapter 5, we have proposed embedded feature selection schemes using a fuzzy rule based framework for selecting class-specific as well as rule-specific features. We have used different synthetic data sets to provide compelling evidence to demonstrate the power of the proposed schemes. However, we have not done any extensive studies with real-world data sets to show the effectiveness of our proposed scheme. In future studies, we want to study the performances of the proposed schemes on several relevant real-world data sets. As mentioned in the future scope of Chapter 3, which is also based on a fuzzy rule based system, we would employ the adaptive softmin [33] as T-norm for handling high dimensional data set and perform a detailed investigation to find out the optimal number of rules in our future studies.

Bibliography

- [1] S. Das and N. R. Pal, “Supervised learning of explicit maps with ability to correct distortions in the target output for manifold learning,” *Information Sciences*, vol. 614, pp. 311–324, 2022. vii, 14, 19
- [2] S. Das and N. R. Pal., “An unsupervised fuzzy rule-based method for structure preserving dimensionality reduction with prediction ability,” in *IFIP International Conference on Artificial Intelligence Applications and Innovations*. Springer, 2019, pp. 413–424. vii, 15, 47, 49, 85
- [3] S. Das and N. R. Pal ., “Nonlinear dimensionality reduction for data visualization: An unsupervised fuzzy rule-based approach,” *IEEE Transactions on Fuzzy Systems*, vol. 30, no. 7, pp. 2157–2169, 2021. vii, 15, 20, 47
- [4] S. Das and N. R. Pal, “Feature selection simultaneously preserving both class and cluster structures,” 2023. vii, 16, 103
- [5] S. Das and N. R. Pal., “Understanding the classes better with class-specific and rule-specific feature selection, and redundancy control in a fuzzy rule based framework,” in *Intelligent Data Engineering and Automated Learning–IDEAL 2022: 23rd International Conference, IDEAL 2022, Manchester, UK, November 24–26, 2022, Proceedings*. Springer, 2022, pp. 338–347. vii, 16, 125
- [6] A. M. Álvarez-Meza, J. A. Lee, M. Verleysen, and G. Castellanos-Dominguez, “Kernel-based dimensionality reduction using renyi’s α -entropy measures of similarity,” *Neurocomputing*, vol. 222, pp. 36–46, 2017. 1
- [7] A. Talwalkar, S. Kumar, and H. Rowley, “Large-scale manifold learning,” in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, 2008, pp. 1–8. 1
- [8] A. R. Koss, M. R. Canagaratna, A. Zaytsev, J. E. Krechmer, M. Breitenlechner, K. J. Nihill, C. Y. Lim, J. C. Rowe, J. R. Roscioli, F. N. Keutsch *et al.*, “Dimensionality-reduction techniques for complex mass spectrometric datasets: application to laboratory atmospheric organic oxidation experiments,” *Atmospheric chemistry and physics*, vol. 20, no. 2, pp. 1021–1041, 2020. 1

- [9] S.-H. Bae, J. Y. Choi, J. Qiu, and G. C. Fox, “Dimension reduction and visualization of large high-dimensional data via interpolation,” in *Proceedings of the 19th ACM international symposium on high performance distributed computing*, 2010, pp. 203–214. 1
- [10] L. Van Der Maaten, E. Postma, and J. Van den Herik, “Dimensionality reduction: A comparative review,” *J Mach Learn Res*, vol. 10, pp. 66–71, 2009. 1, 3, 20, 37, 54, 55, 108, 142
- [11] Y. Wang, K. Feng, X. Chu, J. Zhang, C.-W. Fu, M. Sedlmair, X. Yu, and B. Chen, “A perception-driven approach to supervised dimensionality reduction for visualization,” *IEEE transactions on visualization and computer graphics*, vol. 24, no. 5, pp. 1828–1840, 2018. 1, 6
- [12] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu, “Feature selection: A data perspective,” *ACM computing surveys (CSUR)*, vol. 50, no. 6, pp. 1–45, 2017. 1, 4, 5
- [13] Á. Huertas-García, A. Martín, J. Huertas-Tato, and D. Camacho, “Exploring dimensionality reduction techniques in multilingual transformers,” *arXiv preprint arXiv:2204.08415*, 2022. 1, 2
- [14] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, “Language models are few-shot learners,” *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020. 1
- [15] M. Tan and Q. Le, “Efficientnetv2: Smaller models and faster training,” in *International conference on machine learning*. PMLR, 2021, pp. 10 096–10 106. 1
- [16] S. Smith, M. Patwary, B. Norick, P. LeGresley, S. Rajbhandari, J. Casper, Z. Liu, S. Prabhumoye, G. Zerveas, V. Korthikanti *et al.*, “Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model,” *arXiv preprint arXiv:2201.11990*, 2022. 1
- [17] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, “A convnet for the 2020s,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 11 976–11 986. 1
- [18] S. W. Choi and B. H. Kim, “Applying pca to deep learning forecasting models for predicting pm2. 5,” *Sustainability*, vol. 13, no. 7, p. 3726, 2021. 2
- [19] Y. Liu and L. J. Durlofsky, “3d cnn-pca: A deep-learning-based parameterization for complex geomodels,” *Computers & Geosciences*, vol. 148, p. 104676, 2021. 2

- [20] D. Menaga and S. Revathi, "Probabilistic principal component analysis (ppca) based dimensionality reduction and deep learning for cancer classification," in *Intelligent Computing and Applications*. Springer, 2021, pp. 353–368. 2
- [21] R. Ran, T. Gao, and B. Fang, "Transformer-based dimensionality reduction," *arXiv preprint arXiv:2210.08288*, 2022. 2
- [22] J. Ding, A. Condon, and S. P. Shah, "Interpretable dimensionality reduction of single-cell transcriptome data with deep generative models," *Nature communications*, vol. 9, no. 1, pp. 1–13, 2018. 2
- [23] A. Björklund, J. Mäkelä, and K. Puolamäki, "Slisemap: Explainable dimensionality reduction," *arXiv preprint arXiv:2201.04455*, 2022. 2
- [24] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006. 2, 3, 10, 56, 73, 143
- [25] D. Chakraborty and N. R. Pal, "Selecting useful groups of features in a connectionist framework," *IEEE transactions on neural networks*, vol. 19, no. 3, pp. 381–396, 2008. 2, 4, 10, 105, 106
- [26] R. Chakraborty, C.-T. Lin, and N. R. Pal, "Sensor (group feature) selection with controlled redundancy in a connectionist framework," *International journal of neural systems*, vol. 24, no. 06, p. 1450021, 2014. 2, 4, 10, 105, 106
- [27] R. Chakraborty and N. R. Pal, "Feature selection using a neural framework with controlled redundancy," *IEEE transactions on neural networks and learning systems*, vol. 26, no. 1, pp. 35–50, 2014. 2, 4, 10, 105, 106, 115, 129, 130
- [28] G. Mishne, U. Shaham, A. Cloninger, and I. Cohen, "Diffusion nets," *Applied and Computational Harmonic Analysis*, vol. 47, no. 2, pp. 259–285, 2019. 2, 6, 10, 21, 54, 55, 143
- [29] N. R. Pal, V. K. Eluri, and G. K. Mandal, "Fuzzy logic approaches to structure preserving dimensionality reduction," *IEEE Transactions on Fuzzy Systems*, vol. 10, no. 3, pp. 277–286, 2002. 2, 10, 21, 48, 51, 100
- [30] Y.-C. Chen, N. R. Pal, and I.-F. Chung, "An integrated mechanism for feature selection and fuzzy rule extraction for classification," *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 4, pp. 683–698, 2012. 2, 4, 10, 48, 51, 104, 128, 129, 139
- [31] I.-F. Chung, Y.-C. Chen, and N. R. Pal, "Feature selection with controlled redundancy in a fuzzy rule based framework," *IEEE Transactions on Fuzzy Systems*, vol. 26, no. 2, pp. 734–748, 2018. 2, 4, 10, 48, 128, 129, 130, 139
- [32] A. Karami, "Application of fuzzy clustering for text data dimensionality reduction," *International Journal of Knowledge Engineering and Data Mining*, vol. 6, no. 3, pp. 289–306, 2019. 2, 10

- [33] G. Xue, Q. Chang, J. Wang, K. Zhang, and N. R. Pal, "An adaptive neuro-fuzzy system with integrated feature selection and rule extraction for high-dimensional classification problems," *IEEE Transactions on Fuzzy Systems*, 2022. 2, 4, 10, 104, 146, 147
- [34] O. Yilmaz, "Machine learning using cellular automata based feature expansion and reservoir computing." *Journal of Cellular Automata*, vol. 10, 2015. 2
- [35] D. Jung, J. Lee, and H. Park, "Feature expansion of single dimensional time series data for machine learning classification," in *2021 Twelfth International Conference on Ubiquitous and Future Networks (ICUFN)*. IEEE, 2021, pp. 96–98. 2
- [36] R. O. Duda, P. E. Hart *et al.*, *Pattern classification*. John Wiley & Sons, 2006. 2, 9
- [37] H. Hotelling, "Relations between two sets of variates," *Biometrika*, vol. 28, no. 3/4, pp. 321–377, 1936. 2
- [38] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of eugenics*, vol. 7, no. 2, pp. 179–188, 1936. 2
- [39] H. Akaike, "Factor analysis and aic," in *Selected Papers of Hirotugu Akaike*. Springer, 1987, pp. 371–386. 2
- [40] X. He and P. Niyogi, "Locality preserving projections," in *Advances in neural information processing systems*, 2004, pp. 153–160. 2
- [41] J. W. Sammon, "A nonlinear mapping for data structure analysis," *IEEE Transactions on computers*, vol. 100, no. 5, pp. 401–409, 1969. 3, 6, 15, 21, 49, 55, 85, 107, 110, 143
- [42] J. B. Tenenbaum, V. De Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *science*, vol. 290, no. 5500, pp. 2319–2323, 2000. 3, 6, 14, 19, 51, 58, 141, 143
- [43] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *science*, vol. 290, no. 5500, pp. 2323–2326, 2000. 3, 6, 14, 19, 55, 141, 143
- [44] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural computation*, vol. 15, no. 6, pp. 1373–1396, 2003. 3, 6
- [45] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. 11, pp. 2579–2605, 2008. 3, 6, 20, 55, 57, 143

- [46] L. McInnes, J. Healy, N. Saul, and L. Großberger, “Umap: Uniform manifold approximation and projection,” *Journal of Open Source Software*, vol. 3, no. 29, p. 861, 2018. [Online]. Available: <https://doi.org/10.21105/joss.00861> 3, 57
- [47] L. McInnes, J. Healy, and J. Melville, “Umap: Uniform manifold approximation and projection for dimension reduction,” *arXiv preprint arXiv:1802.03426*, 2018. 3, 6, 53, 55, 143
- [48] J. P. Cunningham and Z. Ghahramani, “Linear dimensionality reduction: Survey, insights, and generalizations,” *The Journal of Machine Learning Research*, vol. 16, no. 1, pp. 2859–2900, 2015. 3
- [49] W. Wang, Y. Huang, Y. Wang, and L. Wang, “Generalized autoencoder: A neural network framework for dimensionality reduction,” in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2014, pp. 490–497. 3, 6, 20
- [50] K. Nag and N. R. Pal, “A multiobjective genetic programming-based ensemble for simultaneous feature selection and classification,” *IEEE transactions on cybernetics*, vol. 46, no. 2, pp. 499–510, 2016. 4
- [51] Q. Song, H. Jiang, and J. Liu, “Feature selection based on fda and f-score for multi-class classification,” *Expert Systems with Applications*, vol. 81, pp. 22–27, 2017. 4, 104, 110, 118
- [52] J. R. Vergara and P. A. Estévez, “A review of feature selection methods based on mutual information,” *Neural computing and applications*, vol. 24, pp. 175–186, 2014. 4, 104, 110, 118
- [53] H. Peng, F. Long, and C. Ding, “Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 27, no. 8, pp. 1226–1238, 2005. 4, 104, 110, 118
- [54] P. A. Estévez, M. Tesmer, C. A. Perez, and J. M. Zurada, “Normalized mutual information feature selection,” *IEEE Transactions on neural networks*, vol. 20, no. 2, pp. 189–201, 2009. 4, 104, 110
- [55] M. Tesmer and P. A. Estévez, “Amifs: Adaptive feature selection by using mutual information,” in *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No. 04CH37541)*, vol. 1. IEEE, 2004, pp. 303–308. 4, 104, 110
- [56] J. R. Vergara and P. A. Estévez, “Cmim-2: an enhanced conditional mutual information maximization criterion for feature selection,” *Journal of Applied Computer Science Methods*, vol. 2, no. 1, pp. 5–20, 2010. 4, 104, 110
- [57] I. Guyon and A. Elisseeff, “An introduction to variable and feature selection,” *Journal of machine learning research*, vol. 3, no. Mar, pp. 1157–1182, 2003. 4

- [58] A. Jović, K. Brkić, and N. Bogunović, “A review of feature selection methods with applications,” in *2015 38th international convention on information and communication technology, electronics and microelectronics (MIPRO)*. Ieee, 2015, pp. 1200–1205. 4
- [59] D. Chakraborty and N. R. Pal, “A neuro-fuzzy scheme for simultaneous feature selection and fuzzy rule-based classification,” *IEEE Transactions on Neural Networks*, vol. 15, no. 1, pp. 110–123, 2004. 4, 104, 129
- [60] J. Wang, H. Zhang, J. Wang, Y. Pu, and N. R. Pal, “Feature selection using a neural network with group lasso regularization and controlled redundancy,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 3, pp. 1110–1123, 2020. 4, 129, 130
- [61] B. Xue, M. Zhang, W. N. Browne, and X. Yao, “A survey on evolutionary computation approaches to feature selection,” *IEEE Transactions on evolutionary computation*, vol. 20, no. 4, pp. 606–626, 2015. 4, 10
- [62] J. A. Lee and M. Verleysen, “Unsupervised dimensionality reduction: Overview and recent advances,” in *The 2010 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2010, pp. 1–8. 5
- [63] S. Tasoulis, N. G. Pavlidis, and T. Roos, “Nonlinear dimensionality reduction for clustering,” *Pattern Recognition*, vol. 107, p. 107508, 2020. 5
- [64] C. Ding, X. He, H. Zha, and H. D. Simon, “Adaptive dimension reduction for clustering high dimensional data,” in *2002 IEEE International Conference on Data Mining, 2002. Proceedings.* IEEE, 2002, pp. 147–154. 5
- [65] A. C. Kumar, “Analysis of unsupervised dimensionality reduction techniques,” *Computer science and information systems*, vol. 6, no. 2, pp. 217–227, 2009. 5
- [66] A. Gisbrecht and B. Hammer, “Data visualization by nonlinear dimensionality reduction,” *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 5, no. 2, pp. 51–73, 2015. 5
- [67] H. Chernoff, “The use of faces to represent points in k-dimensional space graphically,” *Journal of the American statistical Association*, vol. 68, no. 342, pp. 361–368, 1973. 5
- [68] M. Vlachos, C. Domeniconi, D. Gunopulos, G. Kollis, and N. Koudas, “Nonlinear dimensionality reduction techniques for classification and visualization,” in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2002, pp. 645–651. 6
- [69] H. Yin, “Nonlinear dimensionality reduction and data visualization: a review,” *International Journal of Automation and Computing*, vol. 4, pp. 294–303, 2007. 6

- [70] K. Bunte, M. Biehl, and B. Hammer, “A general framework for dimensionality-reducing data visualization mapping,” *Neural Computation*, vol. 24, no. 3, pp. 771–804, 2012. 6, 20, 142
- [71] B. Yang, M. Xiang, and Y. Zhang, “Multi-manifold discriminant isomap for visualization and classification,” *Pattern Recognition*, vol. 55, pp. 215–230, 2016. 6
- [72] E. Becht, L. McInnes, J. Healy, C.-A. Dutertre, I. W. Kwok, L. G. Ng, F. Ginhoux, and E. W. Newell, “Dimensionality reduction for visualizing single-cell data using umap,” *Nature biotechnology*, vol. 37, no. 1, pp. 38–44, 2019. 6
- [73] M. Becker, J. Lippel, A. Stuhlsatz, and T. Zielke, “Robust dimensionality reduction for data visualization with deep neural networks,” *Graphical Models*, vol. 108, p. 101060, 2020. 6
- [74] I. Borg and P. Groenen, “Modern multidimensional scaling: theory and applications,” *Journal of Educational Measurement*, vol. 40, no. 3, pp. 277–280, 2003. 6, 57
- [75] P. Demartines and J. Hérault, “Curvilinear component analysis: A self-organizing neural network for nonlinear mapping of data sets,” *IEEE Transactions on neural networks*, vol. 8, no. 1, pp. 148–154, 1997. 6
- [76] J. A. Lee, A. Lendasse, N. Donckers, M. Verleysen *et al.*, “A robust non-linear projection method.” in *ESAnn*, 2000, pp. 13–20. 6
- [77] K. Q. Weinberger and L. K. Saul, “An introduction to nonlinear dimensionality reduction by maximum variance unfolding,” in *AAAI*, vol. 6, 2006, pp. 1683–1686. 6, 14, 19, 141
- [78] G. E. Hinton and S. T. Roweis, “Stochastic neighbor embedding,” in *Advances in neural information processing systems*, 2003, pp. 857–864. 6
- [79] M. Pena, W. Barbakh, and C. Fyfe, “Topology-preserving mappings for data visualisation,” in *Principal Manifolds for Data Visualization and Dimension Reduction*. Springer, 2008, pp. 131–150. 6
- [80] B. Schölkopf, A. Smola, and K.-R. Müller, “Kernel principal component analysis,” in *International Conference on Artificial Neural Networks*. Springer, 1997, pp. 583–588. 6
- [81] D. L. Donoho and C. Grimes, “Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data,” *Proceedings of the National Academy of Sciences*, vol. 100, no. 10, pp. 5591–5596, 2003. 6, 19, 141
- [82] Z. Zhang and H. Zha, “Principal manifolds and nonlinear dimensionality reduction via tangent space alignment,” *SIAM journal on scientific computing*, vol. 26, no. 1, pp. 313–338, 2004. 6, 19, 141

- [83] H. Qiao, P. Zhang, D. Wang, and B. Zhang, "An explicit nonlinear mapping for manifold learning," *IEEE Transactions on Cybernetics*, vol. 43, no. 1, pp. 51–63, 2013. 6, 20, 21, 55, 56, 73, 143, 146
- [84] A. Singh, N. Thakur, and A. Sharma, "A review of supervised machine learning algorithms," in *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*. Ieee, 2016, pp. 1310–1315. 7
- [85] J.-T. Chien, "Linear regression based bayesian predictive classification for speech recognition," *IEEE transactions on speech and audio processing*, vol. 11, no. 1, pp. 70–79, 2003. 7
- [86] A. Biem, "Minimum classification error training for online handwriting recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 28, no. 7, pp. 1041–1051, 2006. 7
- [87] S. Arora, R. Mittal, H. Kukreja, and M. Bhatia, "An evaluation of denoising techniques and classification of biometric images based on deep learning," *Multimedia Tools and Applications*, vol. 82, no. 6, pp. 8287–8302, 2023. 7
- [88] S. Jiang, J. Hu, C. L. Magee, and J. Luo, "Deep learning for technical document classification," *IEEE Transactions on Engineering Management*, 2022. 7
- [89] W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: A comprehensive review," *Neural computation*, vol. 29, no. 9, pp. 2352–2449, 2017. 7
- [90] M. Trupthi, S. Pabboju, and G. Narasimha, "Improved feature extraction and classification—sentiment analysis," in *2016 International Conference on Advances in Human Machine Interaction (HMI)*. IEEE, 2016, pp. 1–6. 7
- [91] R. M. Lotlikar, "Dimensionality reduction for classification," 1999. 7
- [92] Q. Hou, Y. Wang, and C. Li, "Semisupervised linear discriminant analysis based on pairwise constraint propagation for hyperspectral images," *IEEE Geoscience and Remote Sensing Letters*, vol. 18, no. 10, pp. 1801–1805, 2020. 7
- [93] J. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos, "Regularization studies of linear discriminant analysis in small sample size scenarios with application to face recognition," *Pattern recognition letters*, vol. 26, no. 2, pp. 181–191, 2005. 7
- [94] P. Howland and H. Park, "Generalizing discriminant analysis using the generalized singular value decomposition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 8, pp. 995–1006, 2004. 7
- [95] J. Ye, T. Xiong, and D. Madigan, "Computational and theoretical analysis of null space and orthogonal linear discriminant analysis." *Journal of Machine Learning Research*, vol. 7, no. 7, 2006. 7

- [96] H. Lee, J. Yoo, and S. Choi, "Semi-supervised nonnegative matrix factorization," *IEEE Signal Processing Letters*, vol. 17, no. 1, pp. 4–7, 2009. 7
- [97] L. Jing, C. Zhang, and M. K. Ng, "Snmfca: Supervised nmf-based image classification and annotation," *IEEE transactions on image processing*, vol. 21, no. 11, pp. 4508–4521, 2012. 7
- [98] X. Shu, H. Lu, and L. Tao, "Joint learning with nonnegative matrix factorization and multinomial logistic regression," in *Proceedings of the 2013 International Conference on Image Processing, Melbourne, Australia*, 2013, pp. 15–18. 7
- [99] S. Zafeiriou, A. Tefas, I. Buciu, and I. Pitas, "Exploiting discriminant information in nonnegative matrix factorization with application to frontal face verification," *IEEE Transactions on Neural Networks*, vol. 17, no. 3, pp. 683–695, 2006. 7
- [100] I. Kotsia, S. Zafeiriou, and I. Pitas, "A novel discriminant non-negative matrix factorization algorithm with applications to facial image characterization problems," *IEEE Transactions on Information Forensics and Security*, vol. 2, no. 3, pp. 588–595, 2007. 7
- [101] Y. Lu, Z. Lai, Y. Xu, X. Li, D. Zhang, and C. Yuan, "Nonnegative discriminant matrix factorization," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 7, pp. 1392–1405, 2016. 7
- [102] X. Geng, D.-C. Zhan, and Z.-H. Zhou, "Supervised nonlinear dimensionality reduction for visualization and classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 35, no. 6, pp. 1098–1107, 2005. 7
- [103] Y. Zhang, Z. Zhang, J. Qin, L. Zhang, B. Li, and F. Li, "Semi-supervised local multi-manifold isomap by linear embedding for feature extraction," *Pattern Recognition*, vol. 76, pp. 662–678, 2018. 7
- [104] D. De Ridder, O. Kouropteva, O. Okun, M. Pietikäinen, and R. P. Duin, "Supervised locally linear embedding," in *Artificial Neural Networks and Neural Information Processing—ICANN/ICONIP 2003: Joint International Conference ICANN/ICONIP 2003 Istanbul, Turkey, June 26–29, 2003 Proceedings*. Springer, 2003, pp. 333–341. 7
- [105] S.-q. Zhang, "Enhanced supervised locally linear embedding," *Pattern Recognition Letters*, vol. 30, no. 13, pp. 1208–1218, 2009. 7
- [106] B. Raducanu and F. Dornaika, "A supervised non-linear dimensionality reduction approach for manifold learning," *Pattern Recognition*, vol. 45, no. 6, pp. 2432–2444, 2012. 7

- [107] R. Wu, Y. Yu, and W. Wang, "Scale: Supervised and cascaded laplacian eigenmaps for visual object recognition based on nearest neighbors," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 867–874. 7
- [108] H. Hyötyniemi, "Multivariate regression," Technical report, Helsinki University of Technology, Control Engineering . . . , Tech. Rep., 2001. 8
- [109] S. Chatterjee and J. S. Simonoff, *Handbook of regression analysis*. John Wiley & Sons, 2013. 8
- [110] P. Teetor, *R cookbook: Proven recipes for data analysis, statistics, and graphics*. " O'Reilly Media, Inc.", 2011. 8
- [111] S. KATIPAMULA, T. REDDY, and D. CLARIDGE, "Multivariate regression modeling," *Journal of solar energy engineering*, vol. 120, no. 3, pp. 177–184, 1998. 8
- [112] T. S. Madhulatha, "An overview on clustering methods," *IOSR Journal of Engineering*, vol. 2(4), pp. 719–725, 2012. 9
- [113] A. Ghosal, A. Nandy, A. K. Das, S. Goswami, and M. Panday, "A short review on different clustering techniques and their applications," *Emerging Technology in Modelling and Graphics: Proceedings of IEM Graph 2018*, pp. 69–83, 2020. 9
- [114] P. Giordani, M. B. Ferraro, F. Martella, P. Giordani, M. B. Ferraro, and F. Martella, *Introduction to Clustering*. Springer, 2020. 9
- [115] R. A. Aliev, W. Pedrycz, B. Guirimov, and O. H. Huseynov, "Clustering method for production of z-number based if-then rules," *Information Sciences*, vol. 520, pp. 155–176, 2020. 9
- [116] E. Hancer, B. Xue, and M. Zhang, "A survey on feature selection approaches for clustering," *Artificial Intelligence Review*, vol. 53, pp. 4519–4545, 2020. 9
- [117] J. C. Bezdek, R. Ehrlich, and W. Full, "Fcm: The fuzzy c-means clustering algorithm," *Computers & Geosciences*, vol. 10, no. 2-3, pp. 191–203, 1984. 9, 49, 110
- [118] R. Kruse, C. Borgelt, C. Braune, S. Mostaghim, M. Steinbrecher, F. Klawonn, and C. Moewes, *Computational intelligence*. Springer, 2011. 10
- [119] T. Hanne, R. Dornberger, T. Hanne, and R. Dornberger, *Computational intelligence*. Springer, 2017. 10
- [120] B. Xue, M. Zhang, and W. N. Browne, "Particle swarm optimization for feature selection in classification: A multi-objective approach," *IEEE Transactions on cybernetics*, vol. 43, no. 6, pp. 1656–1671, 2012. 10, 147

- [121] E. Hancer, B. Xue, and M. Zhang, “Differential evolution for filter feature selection based on information theory and feature ranking,” *Knowledge-Based Systems*, vol. 140, pp. 103–119, 2018. 10
- [122] Y. Xue, B. Xue, and M. Zhang, “Self-adaptive particle swarm optimization for large-scale feature selection in classification,” *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 13, no. 5, pp. 1–27, 2019. 10
- [123] L. Ali, I. Wajahat, N. Amiri Golilarz, F. Keshtkar, and S. A. C. Bukhari, “Lda-ga-svm: improved hepatocellular carcinoma prediction through dimensionality reduction and genetically optimized support vector machine,” *Neural Computing and Applications*, vol. 33, pp. 2783–2792, 2021. 10
- [124] P. Wang, B. Xue, J. Liang, and M. Zhang, “Feature clustering-assisted feature selection with differential evolution,” *Pattern Recognition*, vol. 140, p. 109523, 2023. 10
- [125] L. Magdalena, “Fuzzy rule-based systems,” *Springer handbook of computational intelligence*, pp. 203–218, 2015. 10
- [126] L. S. Riza, C. Bergmeir, F. Herrera, and J. M. Benítez, “frbs: Fuzzy rule-based systems for classification and regression in r,” *Journal of statistical software*, vol. 65, pp. 1–30, 2015. 10
- [127] J. Deng and Y. Deng, “Information volume of fuzzy membership function,” *International Journal of Computers Communications & Control*, vol. 16, no. 1, 2021. 10
- [128] S. Haykin, *Neural networks: a comprehensive foundation*. Prentice Hall PTR, 1998. 12, 14, 53
- [129] H. Ramchoun, M. J. Idrissi, Y. Ghanou, and M. Ettaouil, “Multilayer perceptron: Architecture optimization and training with mixed activation functions,” in *Proceedings of the 2nd international Conference on Big Data, Cloud and Applications*, 2017, pp. 1–6. 12, 14
- [130] L. Noriega, “Multilayer perceptron tutorial,” *School of Computing. Staffordshire University*, vol. 4, p. 5, 2005. 12, 14
- [131] V. Mudunuru, “Comparison of activation functions in multilayer neural networks for stage classification in breast cancer,” *Neural Parallel Sci. Comput. Arch*, vol. 24, pp. 83–96, 2016. 13
- [132] Y. Li, Y. Liu, and D. S. Williamson, “On loss functions for deep-learning based t60 estimation,” in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 486–490. 13

- [133] D. Lunga, S. Prasad, M. M. Crawford, and O. Ersoy, "Manifold-learning-based feature extraction for classification of hyperspectral data: A review of advances in manifold learning," *IEEE Signal Processing Magazine*, vol. 31, no. 1, pp. 55–66, 2013. 14
- [134] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering." in *Advances in neural information processing systems*, vol. 14, 2001. 14, 19, 21, 24, 30, 141
- [135] C. P. Ezenkwu, U. I. Akpan, and B. U.-A. Stephen, "A class-specific metaheuristic technique for explainable relevant feature selection," *Machine Learning with Applications*, vol. 6, p. 100142, 2021. 16, 125, 126, 127
- [136] G. de Lannoy, D. François, M. Verleysen *et al.*, "Class-specific feature selection for one-against-all multiclass svms." in *ESANN*. Citeseer, 2011. 16, 125, 126, 127
- [137] D. Nardone, A. Ciaramella, and A. Staiano, "A sparse-modeling based approach for class specific feature selection," *PeerJ Computer Science*, vol. 5, p. e237, 2019. 16, 125, 126, 127
- [138] R. Panthong and A. Srivihok, "Liver cancer classification model using hybrid feature selection based on class-dependent technique for the central region of thailand," *Information*, vol. 10, no. 6, p. 187, 2019. 16, 125, 126, 127
- [139] B. B. Pineda-Bautista, J. A. Carrasco-Ochoa, and J. F. Martínez-Trinidad, "General framework for class-specific feature selection," *Expert Systems with Applications*, vol. 38, no. 8, pp. 10 018–10 024, 2011. 16, 125, 126, 127
- [140] Y. Qian, "Class-specific guided local feature selection for data classification," in *2019 IEEE 4th International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)*. IEEE, 2019, pp. 645–649. 16, 125, 126, 127
- [141] L.-m. Yuan, Y. Sun, and G. Huang, "Using class-specific feature selection for cancer detection with gene expression profile data of platelets," *Sensors*, vol. 20, no. 5, p. 1528, 2020. 16, 125, 126, 127
- [142] W. Zhou and J. A. Dickerson, "A novel class dependent feature selection method for cancer biomarker discovery," *Computers in biology and medicine*, vol. 47, pp. 66–75, 2014. 16, 125, 126, 127
- [143] J. W. Cannon, "The recognition problem: what is a topological manifold?" *Bulletin of the American Mathematical Society*, vol. 84, no. 5, pp. 832–866, 1978. 19
- [144] N. D. Nguyen, I. K. Blaby, and D. Wang, "Maninetcluster: a novel manifold learning approach to reveal the functional links between gene networks," *BMC genomics*, vol. 20, pp. 1–14, 2019. 19

- [145] A. Elgammal and C.-S. Lee, “Inferring 3d body pose from silhouettes using activity manifold learning,” in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, vol. 2. IEEE, 2004, pp. II–II. 19
- [146] Z. Dong, S. Jia, C. Zhang, M. Pei, and Y. Wu, “Deep manifold learning of symmetric positive definite matrices with application to face recognition,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017. 19
- [147] J. Zhang, H. Huang, and J. Wang, “Manifold learning for visualizing and analyzing high-dimensional data,” *IEEE Intelligent Systems*, vol. 25, no. 04, pp. 54–61, 2010. 19
- [148] Y. Bengio, J.-f. Paiement, P. Vincent, O. Delalleau, N. L. Roux, and M. Ouimet, “Out-of-sample extensions for lle, isomap, mds, eigenmaps, and spectral clustering,” in *Advances in neural information processing systems*, 2004, pp. 177–184. 20, 56, 73
- [149] T.-J. Chin and D. Suter, “Out-of-sample extrapolation of learned manifolds,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 9, pp. 1547–1556, 2008. 20
- [150] L. K. Saul and S. T. Roweis, “Think globally, fit locally: unsupervised learning of low dimensional manifolds,” *Journal of machine learning research*, vol. 4, no. Jun, pp. 119–155, 2003. 20
- [151] L. van der Maaten, “Learning a parametric embedding by preserving local structure,” in *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*. Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA: PMLR, 16–18 Apr 2009, pp. 384–391. [Online]. Available: <http://proceedings.mlr.press/v5/maaten09a.html> 20
- [152] N. R. Pal and V. K. Eluri, “Two efficient connectionist schemes for structure preserving dimensionality reduction,” *IEEE Transactions on Neural Networks*, vol. 9, no. 6, pp. 1142–1154, 1998. 21, 100, 145
- [153] Y. Xia, Q. Lu, J. Feng, and H.-Y. Bae, “An explicit sparse mapping for non-linear dimensionality reduction,” in *International Conference on Rough Sets and Knowledge Technology*. Springer, 2014, pp. 149–157. 21
- [154] S. Yan, D. Xu, B. Zhang, H.-J. Zhang, Q. Yang, and S. Lin, “Graph embedding and extensions: A general framework for dimensionality reduction,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 29, no. 1, pp. 40–51, 2006. 21
- [155] R. R. Coifman and S. Lafon, “Diffusion maps,” *Applied and computational harmonic analysis*, vol. 21, no. 1, pp. 5–30, 2006. 21

- [156] S.-B. Chen, C. H. Ding, and B. Luo, "Linear regression based projections for dimensionality reduction," *Information Sciences*, vol. 467, pp. 74–86, 2018. 21, 142
- [157] M. Zhao, T. W. Chow, Z. Wu, Z. Zhang, and B. Li, "Learning from normalized local and global discriminative information for semi-supervised regression and dimensionality reduction," *Information Sciences*, vol. 324, pp. 286–309, 2015. 21
- [158] Y. Xue and L. Zhang, "Laplacian pair-weight vector projection for semi-supervised learning," *Information Sciences*, vol. 573, pp. 1–19, 2021. 21
- [159] H. Xu, "Semi-supervised manifold learning based on polynomial mapping for localization in wireless sensor networks," *Signal Processing*, vol. 172, p. 107570, 2020. 21, 26
- [160] J. Yu, D. Tao, and M. Wang, "Adaptive hypergraph learning and its application in image classification," *IEEE Transactions on Image Processing*, vol. 21, no. 7, pp. 3262–3272, 2012. 22
- [161] J. Yu, Y. Rui, and D. Tao, "Click prediction for web image reranking using multimodal sparse coding," *IEEE Transactions on Image Processing*, vol. 23, no. 5, pp. 2019–2032, 2014. 22
- [162] L. Wang and K.-J. Yoon, "Knowledge distillation and student-teacher learning for visual intelligence: A review and new outlooks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2021. 26
- [163] Y. Li, J. Yang, Y. Song, L. Cao, J. Luo, and L.-J. Li, "Learning from noisy labels with distillation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1910–1918. 26
- [164] Z.-H. Zhou, "A brief introduction to weakly supervised learning," *National science review*, vol. 5, no. 1, pp. 44–53, 2018. 26
- [165] Y.-F. Li, L.-Z. Guo, and Z.-H. Zhou, "Towards safe weakly supervised learning," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 1, pp. 334–346, 2019. 26
- [166] Z.-Y. Zhang, P. Zhao, Y. Jiang, and Z.-H. Zhou, "Learning from incomplete and inaccurate supervision," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, 2021. 26
- [167] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011. 27, 31, 55, 57
- [168] S. A. Nene, S. K. Nayar, and H. Murase, "Columbia object image library (coil-20)," Department of Computer Science, Columbia University, Tech. Rep. CUCS-005-96, 1996. 27, 55

- [169] W. Li, S. Zhai, W. Xu, W. Pedrycz, Y. Qian, W. Ding, and T. Zhan, “Feature selection approach based on improved fuzzy c-means with principle of refined justifiable granularity,” *IEEE Transactions on Fuzzy Systems*, 2022. 48
- [170] W. Pedrycz, *Fuzzy control and fuzzy systems*. Research Studies Press Ltd., 1993. 48
- [171] S.-K. Oh, W.-D. Kim, and W. Pedrycz, “Design of optimized cascade fuzzy controller based on differential evolution: Simulation studies and practical insights,” *Engineering Applications of Artificial Intelligence*, vol. 25, no. 3, pp. 520–532, 2012. 48
- [172] M. Pratama, W. Pedrycz, and E. Lughofer, “Evolving ensemble fuzzy classifier,” *IEEE Transactions on Fuzzy Systems*, vol. 26, no. 5, pp. 2552–2567, 2018. 48
- [173] X. Han, X. Zhu, W. Pedrycz, and Z. Li, “A three-way classification with fuzzy decision trees,” *Applied Soft Computing*, vol. 132, p. 109788, 2023. 48
- [174] P.-C. Chang and C.-H. Liu, “A tsf type fuzzy rule based system for stock price prediction,” *Expert Systems with applications*, vol. 34, no. 1, pp. 135–144, 2008. 48
- [175] X. Yang, F. Yu, and W. Pedrycz, “Long-term forecasting of time series based on linear fuzzy information granules and fuzzy inference system,” *International Journal of Approximate Reasoning*, vol. 81, pp. 1–27, 2017. 48
- [176] E. Lughofer and R. Nikzad-Langerodi, “Robust generalized fuzzy systems training from high-dimensional time-series data using local structure preserving pls,” *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 11, pp. 2930–2943, 2020. 48
- [177] Y. Lin, Y. Li, C. Wang, and J. Chen, “Attribute reduction for multi-label learning with fuzzy rough set,” *Knowledge-based systems*, vol. 152, pp. 51–61, 2018. 48
- [178] N. Mac Parthaláin, R. Jensen, and R. Diao, “Fuzzy-rough set bireducts for data reduction,” *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 8, pp. 1840–1850, 2019. 48
- [179] T. Takagi and M. Sugeno, “Fuzzy identification of systems and its applications to modeling and control,” *IEEE Transactions on Systems, Man, and Cybernetics*, no. 1, pp. 116–132, 1985. 48
- [180] N. R. Pal and S. Saha, “Simultaneous structure identification and fuzzy rule generation for takagi–sugeno models,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 38, no. 6, pp. 1626–1638, 2008. 48, 51
- [181] S. Mitra, W. Pedrycz, and B. Barman, “Shadowed c-means: Integrating fuzzy and rough clustering,” *Pattern recognition*, vol. 43, no. 4, pp. 1282–1291, 2010. 49

- [182] J. M. Lee, *Introduction to smooth manifolds*. Springer, 2001. 50
- [183] L. Yang, “Sammon’s nonlinear mapping using geodesic distances,” in *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, vol. 2. IEEE, 2004, pp. 303–306. 50
- [184] M. Sugeno and T. Yasukawa, “A fuzzy-logic-based approach to qualitative modeling,” *IEEE Transactions on fuzzy systems*, vol. 1, no. 1, p. 7, 1993. 51
- [185] L.-X. Wang and J. M. Mendel, “Generating fuzzy rules by learning from examples,” *IEEE Transactions on systems, man, and cybernetics*, vol. 22, no. 6, pp. 1414–1427, 1992. 51
- [186] O. Cordón and F. Herrera, “A two-stage evolutionary process for designing tsf fuzzy rule-based systems,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 29, no. 6, pp. 703–715, 1999. 51
- [187] N. Asgharbeygi and A. Maleki, “Geodesic k-means clustering,” in *2008 19th International Conference on Pattern Recognition*. IEEE, 2008, pp. 1–4. 51
- [188] J. Kim, K.-H. Shim, and S. Choi, “Soft geodesic kernel k-means,” in *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP’07*, vol. 2. IEEE, 2007, pp. II–429. 51
- [189] B. Feil and J. Abonyi, “Geodesic distance based fuzzy clustering,” in *Soft Computing in Industrial Applications*. Springer, 2007, pp. 50–59. 51
- [190] R. W. Floyd, “Algorithm 97: shortest path,” *Communications of the ACM*, vol. 5, no. 6, p. 345, 1962. 51
- [191] M. M. Gupta and J. Qi, “Theory of t-norms and fuzzy inference methods,” *Fuzzy sets and systems*, vol. 40, no. 3, pp. 431–450, 1991. 52, 128
- [192] S. Roweis. (2020) Data for matlab hackers, faces, frey face. [Online]. Available: https://cs.nyu.edu/~roweis/data/frey_rawface.mat 55
- [193] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean *et al.*, “Tensorflow: A system for large-scale machine learning,” in *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, 2016, pp. 265–283. 57, 109, 117, 132
- [194] G. Mishne. (2017) Diffusionnet - a geometric autoencoder. [Online]. Available: <https://github.com/gmishne/DiffusionNet> 57
- [195] L. van der Maaten. (2020) Matlab toolbox for dimensionality reduction. [Online]. Available: <https://lvdmaaten.github.io/drtoolbox/> 57

- [196] C. C. Aggarwal, A. Hinneburg, and D. A. Keim, “On the surprising behavior of distance metrics in high dimensional space,” in *International conference on database theory*. Springer, 2001, pp. 420–434. 101, 146
- [197] S. Solorio-Fernández, J. A. Carrasco-Ochoa, and J. F. Martínez-Trinidad, “A review of unsupervised feature selection methods,” *Artificial Intelligence Review*, vol. 53, no. 2, pp. 907–948, 2020. 103
- [198] R. Zhang, F. Nie, X. Li, and X. Wei, “Feature selection with multi-view data: A survey,” *Information Fusion*, vol. 50, pp. 158–167, 2019. 103
- [199] Z. Atashgahi, X. Zhang, N. Kichler, S. Liu, L. Yin, M. Pechenizkiy, R. Veldhuis, and D. C. Mocanu, “Supervised feature selection with neuron evolution in sparse neural networks,” *arXiv preprint arXiv:2303.07200*, 2023. 103
- [200] R. Jiao, B. Xue, and M. Zhang, “Solving multi-objective feature selection problems in classification via problem reformulation and duplication handling,” *IEEE Transactions on Evolutionary Computation*, 2022. 103, 147
- [201] S. An, M. Zhang, C. Wang, and W. Ding, “Robust fuzzy rough approximations with knn granules for semi-supervised feature selection,” *Fuzzy Sets and Systems*, p. 108476, 2023. 103
- [202] Z. Zeng, X. Wang, F. Yan, and Y. Chen, “Local adaptive learning for semi-supervised feature selection with group sparsity,” *Knowledge-Based Systems*, vol. 181, p. 104787, 2019. 103
- [203] G. Chandrashekar and F. Sahin, “A survey on feature selection methods,” *Computers & Electrical Engineering*, vol. 40, no. 1, pp. 16–28, 2014. 103
- [204] S. Wang, W. Pedrycz, Q. Zhu, and W. Zhu, “Unsupervised feature selection via maximum projection and minimum redundancy,” *Knowledge-Based Systems*, vol. 75, pp. 19–29, 2015. 103, 104
- [205] L. Du and Y.-D. Shen, “Unsupervised feature selection with adaptive structure learning,” in *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, 2015, pp. 209–218. 104
- [206] J.-S. Wu, M.-X. Song, W. Min, J.-H. Lai, and W.-S. Zheng, “Joint adaptive manifold and embedding learning for unsupervised feature selection,” *Pattern Recognition*, vol. 112, p. 107742, 2021. 104
- [207] Z. Li, J. Liu, Y. Yang, X. Zhou, and H. Lu, “Clustering-guided sparse structural learning for unsupervised feature selection,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 9, pp. 2138–2150, 2013. 104
- [208] C. Hou, F. Nie, X. Li, D. Yi, and Y. Wu, “Joint embedding learning and sparse regression: A framework for unsupervised feature selection,” *IEEE transactions on cybernetics*, vol. 44, no. 6, pp. 793–804, 2013. 104

- [209] Z. Zhao, L. Wang, H. Liu, and J. Ye, "On similarity preserving feature selection," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 3, pp. 619–632, 2011. 104
- [210] W. Gao, L. Hu, Y. Li, and P. Zhang, "Preserving similarity and staring decisis for feature selection," *IEEE Transactions on Artificial Intelligence*, vol. 2, no. 6, pp. 584–593, 2021. 104
- [211] Q. Gu, Z. Li, and J. Han, "Generalized fisher score for feature selection," *arXiv preprint arXiv:1202.3725*, 2012. 104
- [212] X. He, D. Cai, and P. Niyogi, "Laplacian score for feature selection," *Advances in neural information processing systems*, vol. 18, 2005. 104
- [213] R. J. Urbanowicz, M. Meeker, W. La Cava, R. S. Olson, and J. H. Moore, "Relief-based feature selection: Introduction and review," *Journal of biomedical informatics*, vol. 85, pp. 189–203, 2018. 104
- [214] J. Wang, J. Wei, and Z. Yang, "Supervised feature selection by preserving class correlation," in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, 2016, pp. 1613–1622. 104
- [215] T. M. Le, T. M. Vo, T. N. Pham, and S. V. T. Dao, "A novel wrapper-based feature selection for early diabetes prediction enhanced with a metaheuristic," *IEEE Access*, vol. 9, pp. 7869–7884, 2020. 104
- [216] N. Zhou, Y. Xu, H. Cheng, J. Fang, and W. Pedrycz, "Global and local structure preserving sparse subspace learning: An iterative approach to unsupervised feature selection," *Pattern Recognition*, vol. 53, pp. 87–101, 2016. 104
- [217] Y. Zhu, X. Zhang, R. Hu, and G. Wen, "Adaptive structure learning for low-rank supervised feature selection," *Pattern Recognition Letters*, vol. 109, pp. 89–96, 2018. 104
- [218] X. Liu, L. Wang, J. Zhang, J. Yin, and H. Liu, "Global and local structure preservation for feature selection," *IEEE transactions on neural networks and learning systems*, vol. 25, no. 6, pp. 1083–1095, 2013. 104
- [219] X. Zhu, S. Zhang, R. Hu, Y. Zhu *et al.*, "Local and global structure preservation for robust unsupervised spectral feature selection," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 3, pp. 517–529, 2017. 104
- [220] Q. Ye, X. Zhang, and Y. Sun, "Dual global structure preservation based supervised feature selection," *Neural Processing Letters*, vol. 51, pp. 2765–2787, 2020. 104
- [221] R. Setiono and H. Liu, "Neural-network feature selector," *IEEE transactions on neural networks*, vol. 8, no. 3, pp. 654–662, 1997. 105

- [222] G. Castellano and A. M. Fanelli, "Variable selection using neural-network models," *Neurocomputing*, vol. 31, no. 1-4, pp. 1–13, 2000. 105
- [223] A. Verikas and M. Bacauskiene, "Feature selection with neural networks," *Pattern recognition letters*, vol. 23, no. 11, pp. 1323–1335, 2002. 105
- [224] G. P. Zhang, "Neural networks for classification: a survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 30, no. 4, pp. 451–462, 2000. 105
- [225] S. Dreiseitl and L. Ohno-Machado, "Logistic regression and artificial neural network classification models: a methodology review," *Journal of biomedical informatics*, vol. 35, no. 5-6, pp. 352–359, 2002. 105
- [226] S. Agrawal and J. Agrawal, "Neural network techniques for cancer prediction: A survey," *Procedia Computer Science*, vol. 60, pp. 769–774, 2015. 105
- [227] A. Mabrouk, R. P. D. Redondo, and M. Kaye, "Deep learning-based sentiment classification: a comparative survey," *IEEE Access*, vol. 8, pp. 85 616–85 638, 2020. 105
- [228] Z. Zhang and M. R. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," in *32nd Conference on Neural Information Processing Systems (NeurIPS)*, 2018. 106
- [229] D. Dua and C. Graff, "UCI machine learning repository," 2017. [Online]. Available: <http://archive.ics.uci.edu/ml> 108
- [230] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu, "Feature selection: A data perspective," *ACM Computing Surveys (CSUR)*, vol. 50, no. 6, p. 94, 2018. 108
- [231] M. Prasad, A. Sowmya, and I. Koch, "Efficient feature selection based on independent component analysis," in *Proceedings of the 2004 Intelligent Sensors, Sensor Networks and Information Processing Conference, 2004.* IEEE, 2004, pp. 427–432. 110
- [232] D. Steinley, "Properties of the hubert-arable adjusted rand index." *Psychological methods*, vol. 9, no. 3, p. 386, 2004. 110
- [233] S. Fletcher, M. Z. Islam *et al.*, "Comparing sets of patterns with the jaccard index," *Australasian Journal of Information Systems*, vol. 22, 2018. 110
- [234] W. Sun, L. Zhang, B. Du, W. Li, and Y. M. Lai, "Band selection using improved sparse subspace clustering for hyperspectral imagery classification," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 8, no. 6, pp. 2784–2797, 2015. 116

- [235] W. Sun and Q. Du, “Hyperspectral band selection: A review,” *IEEE Geoscience and Remote Sensing Magazine*, vol. 7, no. 2, pp. 118–139, 2019. 116
- [236] A. Challa, G. Barman, S. Danda, and B. D. Sagar, “Band selection using dilation distances,” *IEEE Geoscience and Remote Sensing Letters*, 2021. 116
- [237] L. Li, J. Yin, X. Jia, S. Li, and B. Han, “Joint spatial–spectral attention network for hyperspectral image classification,” *IEEE Geoscience and Remote Sensing Letters*, vol. 18, no. 10, pp. 1816–1820, 2021. 116
- [238] M. Graña, M. Veganzons, and B. Ayerdi, “Hyperspectral remote sensing scenes,” 06 2021. [Online]. Available: http://www.ehu.eus/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes 116
- [239] A. Santara, K. Mani, P. Hatwar, A. Singh, A. Garg, K. Padia, and P. Mitra, “Bass net: Band-adaptive spectral-spatial feature learning neural network for hyperspectral image classification,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 9, pp. 5293–5301, 2017. 117
- [240] A. Santara, K. Mani, and A. Shukla, “Deep learning for land-cover classification in hyperspectral images,” 07 2016. [Online]. Available: <https://kgpml.github.io/Hyperspectral/> 117
- [241] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “Smote: synthetic minority over-sampling technique,” *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002. 117
- [242] H. Du, H. Qi, X. Wang, R. Ramanath, and W. E. Snyder, “Band selection using independent component analysis for hyperspectral image processing,” in *32nd Applied Imagery Pattern Recognition Workshop, 2003. Proceedings.* IEEE, 2003, pp. 93–98. 118
- [243] E. Jang, S. Gu, and B. Poole, “Categorical reparameterization with gumbel-softmax,” *arXiv preprint arXiv:1611.01144*, 2016. 129
- [244] J. Parmar, S. Chouhan, V. Raychoudhury, and S. Rathore, “Open-world machine learning: applications, challenges, and opportunities,” *ACM Computing Surveys*, vol. 55, no. 10, pp. 1–37, 2023. 143
- [245] V. Schwag, A. N. Bhagoji, L. Song, C. Sitawarin, D. Cullina, M. Chiang, and P. Mittal, “Analyzing the robustness of open-world machine learning,” in *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*, 2019, pp. 105–116. 143
- [246] X. He, D. Cai, S. Yan, and H.-J. Zhang, “Neighborhood preserving embedding,” in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, vol. 2. IEEE, 2005, pp. 1208–1213. 145

-
- [247] X. Chen, C. Wang, X. Lan, N. Zheng, and W. Zeng, “Neighborhood geometric structure-preserving variational autoencoder for smooth and bounded data sources,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 8, pp. 3598–3611, 2021. 145
 - [248] Z. Zhang, J. Wang, and H. Zha, “Adaptive manifold learning,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 2, pp. 253–265, 2011. 146
 - [249] P. Wang, B. Xue, J. Liang, and M. Zhang ., “Feature selection using diversity-based multi-objective binary differential evolution,” *Information Sciences*, 2023. 147
 - [250] P. Wang, B. Xue, J. Liang, and M. Zhang, “Differential evolution based feature selection: A niching-based multi-objective approach,” *IEEE Transactions on Evolutionary Computation*, 2022. 147