Development of some Neural Network Models for Non-negative Matrix Factorization: Dimensionality Reduction

Prasun Dutta

Indian Statistical Institute

January 2025

INDIAN STATISTICAL INSTITUTE

DOCTORAL THESIS

Development of some Neural Network Models for Non-negative Matrix Factorization: Dimensionality Reduction

Author:
Prasun Dutta

Supervisor:
Professor Rajat K. De

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

Machine Intelligence Unit

Indian Statistical Institute
Kolkata, India

January 2025

To my family

Acknowledgements

A PhD thesis represents more than just a culmination of original research for earning a degree; it embodies the collective support, guidance and encouragement of many individuals. I am deeply grateful to all who have made this journey an unforgettable experience.

My heartfelt appreciation extends beyond words to Professor Rajat Kumar De, whose supervision and unwavering motivation sustained me through challenging times. I consider myself extremely fortunate to have had him as my mentor. His invaluable support and guidance over the years are truly indescribable.

I am also indebted to the Indian Statistical Institute (ISI) for its exceptional research infrastructure and the research fellowship that enabled me to conduct my work without financial constraints. I extend sincere thanks to all ISI faculty members for their continuous support and invaluable advice. Special gratitude goes to the members of the Research Fellow Advisory Committee, Computer and Communication Sciences Division, ISI, for their insightful feedback that significantly enhanced my research.

I am grateful to my colleagues and lab mates for fostering a supportive environment throughout my tenure. Acknowledgements are also due to the CSSC for providing computing facilities, the ISI library for its extensive resources, the office staff of the Machine Intelligence Unit for their friendly assistance and ISI authorities for their various other supports. I owe a debt of gratitude to the Institute of Data Engineering, Analytics and Science Foundation, Technology Innovation Hub, ISI, Kolkata, for their crucial infrastructure and financial support.

My deepest thanks go to my beloved family for their unwavering guidance, love, encouragement and unconditional support throughout my journey. I am especially grateful to my wife, Srijani, whose dedication to raising our daughter, Anishka, alleviated many of my responsibilities and allowed me to focus on my research. To Anishka, I apologize for the countless hours I should have spent with you.

Lastly, I want to express my gratitude to all those whom I may have inadvertently overlooked, for their kind wishes and support. Your contributions have been invaluable.

(Prasun Dutta)

January 2025

Abstract

Recent research has been driven by the abundance of data, leading to the development of systems that enhance understanding across various fields. Effective machine learning algorithms are crucial for managing high-dimensional data, with dimension reduction being a key strategy to improve algorithm efficiency and decision-making. Non-negative Matrix Factorization (NMF) stands out as a method that transforms large datasets into interpretable, lower-dimensional forms by decomposing a matrix with non-negative elements into a pair of non-negative factors. This approach addresses the curse of dimensionality by dimensionally reducing data while preserving meaningful information.

Dimension reduction techniques rely on extracting high-quality features from large datasets. Machine learning algorithms offer a solution by learning and optimizing feature representations, which often outperform manually crafted ones. Artificial Neural Networks (ANNs) emulate human brain processing and excel in handling complex and nonlinear data relationships. Deep neural network models learn hierarchical patterns from data without explicit human intervention, making them ideal for large datasets.

Traditional NMF technique employs block coordinate descent to update input matrix factors, whereas, we aim for simultaneous update. Our research work attempts to combine the strengths of NMF and neural networks to develop novel architectures that optimize low-dimensional data representation. We introduce five novel neural network architectures for NMF, accompanied by tailored objective functions and learning strategies to enhance the low rank approximation of input matrices in our thesis.

In this thesis, first of all, n^2MFn^2, a model based on shallow neural network architecture, has been developed. An approximation of the input matrix has been ensured by the formulation of an appropriate objective function and adaptive learning scheme. Activation functions and weight initialization strategies have also been adjusted to adapt to the circumstances. On top of this shallow model, two deep neural network models, named DN3MF and MDSR-NMF, have been designed. To achieve the robustness of the deep neural network framework, the models have been designed as a two stage architecture, viz., pre-training and stacking. To find the closest realization of the conventional NMF technique as well as the closest approximation of the input, a novel neural network architecture has been proposed in MDSR-NMF. Finally, two deep learning models, named IG-MDSR-NMF and IG-MDSR-RNMF, have been developed to imitate the human-centric learning strategy while guaranteeing a distinct pair of factor matrices that yields a better approximation of the input matrix. In IG-MDSR-NMF and IG-MDSR-RNMF the layers not only receive the hierarchically processed input from

the previous layer but also refer to the original data whenever needed to ensure that the learning path is correct. A novel kind of non-negative matrix factorization technique known as Relaxed NMF has been developed for IG-MDSR-RNMF, in which only one factor matrix meets the non-negativity requirements while the other one does not. This novel NMF technique allows the model to generate the best possible low dimensional representation of the input matrix while the confrontation of maintaining a pair of non-negative factors is removed.

Contents

List of Figures

List of Tables

List of Publications

1. Dutta, P. and De, R.K., 2024. DN3MF: Deep Neural Network for Non-negative Matrix Factorization towards Low Rank Approximation. Pattern Analysis and Applications, 27(4), pp. 112. Springer.
   doi: https://doi.org/10.1007/s10044-024-01335-3

2. Dutta, P. and De, R.K., 2023. MDSR-NMF: Multiple deconstruction single reconstruction deep neural network model for non-negative matrix factorization. Network: Computation in Neural Systems, 34(4), pp. 306-342. Taylor & Francis.
   doi: 10.1080/0954898X.2023.2257773

3. Dutta, P. and De, R.K., 2024. Input Guided Multiple Deconstruction Single Reconstruction neural network models for Matrix Factorization. arXiv, 2405.1344.
   doi: 10.48550/arXiv.2405.13449

4. Dutta, P. and De, R.K., 2022, December. n2MFn2: Non-negative Matrix Factorization in A Single Deconstruction Single Reconstruction Neural Network Framework for Dimensionality Reduction. In 2022 International Conference on High Performance Big Data and Intelligent Systems (HDIS) (pp. 79-84). IEEE.
   doi: 10.1109/HDIS56859.2022.9991646

5. Dutta, P. and De, R.K., 2022, December. A neural network model for matrix factorization: dimensionality reduction. In 2022 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE) (pp. 1-6). IEEE.
   doi: 10.1109/CSDE56538.2022.10089284

Chapter 1

Introduction

1.1 Introduction

The current era of science is somewhat driven by analyses of extensive datasets generated by high throughput technology. This huge amount of raw data is undoubtedly very rich in information content. The scenario has led to the emergence of systems approaches to advance our understanding of science. However, one of the main problems is handling the complexity of this information. Simultaneously, this rapid growth of data has come with the problem of the curse of dimensionality. The conventional techniques of data analysis are being challenged by this rapidly growing volume/dimension of data. Suitable machine learning algorithms need to be developed/used for handling such high-dimensional data. One way to deal with the high volume of data demands dimension reduction for reducing space and time complexity of the algorithms, and better decision making. Over the years, researchers have developed several dimension reduction techniques. Deep learning, a modern-day machine learning methodology, is quite capable of finding hidden structures from very large data sets in an incremental approach.

Non-negative Matrix Factorization (NMF) refers to a group of algorithms in multivariate analysis and linear algebra. NMF is popular for its effectiveness in feature extraction and uses the pervasiveness of the non-negative input data matrix to extract sparse and significant features. In NMF, multivariate data is decomposed into two constituent parts based on the required number of features. The original data matrix

along with the two learned constituent matrices follows non-negativity criteria. As the output coefficients are all positive, each data point may be represented as the sum of vectors in the basis multiplied by certain coefficients. As a result, the output basis is effectively a breakdown of given data points into small parts. These components can be added together to recreate the given data points. This additive parts-based representation of the given data points differentiates NMF from other low rank approximation techniques and makes the results easier to interpret. NMF has been used effectively in many fields, such as image processing, computer vision, recommender systems, text mining and audio signal processing, among others.

An artificial neural network, or ANN, addresses complex problems by modelling them in a similar way to information processing as done by the human brain. ANNs are superior to conventional machine learning algorithms in a number of ways. One of which is the ease of handling complicated and nonlinear data relationships. ANNs are capable of producing outputs that are not limited to the supplied input, i.e., they can learn on their own. Additionally, ANNs can adjust themselves to work with insufficient data. ANNs can process a variety of data formats, including texts, pictures and sequences, to mention a few. Their adaptability allows them to be used for diverse real-world scenarios. Neural networks are capable of adapting to the changes in the distribution of data over time. In dynamic contexts where the relationships between variables may evolve, this flexibility of adaptation to the environment is crucial. From the input, deep neural networks can automatically learn hierarchical feature representations, eliminating the need for manual feature engineering. Deep architectures can exhibit a degree of fault tolerance and redundancy, i.e., they can continue to function rather effectively even if some neurons or connections fail.

The success of a dimension reduction technique depends on the quality of features extracted from the hefty dataset. However, for many tasks, it is difficult to know about the features to be extracted. One solution to this problem is to apply machine learning algorithms to discover not only the mapping from representation to output but also the representation of itself. Learned representations often result in much better performance than that of hand-designed representations. Deep learning allows models to learn from experience. NMF is an iterative algorithm. The effectiveness of the NMF algorithm would have increased if the traditional iterative procedure could be clubbed

with the benefits of deep learning. Over the years researchers have tried to develop algorithms for NMF using neural networks.

Here, in this thesis, we have developed five models of novel neural networks for NMF, and have designed appropriate objective functions and learning methodology ensuring the best possible low rank approximation of the input matrix. The main objective of the thesis is to develop neural network models for NMF so that hefty high-dimensional datasets can be processed to a part-based, sparse and meaningful low rank representation of the same. Initially, a shallow neural network architecture based model, named n^2MFn^2, has been designed. Appropriate objective function and adaptive learning rules have been formulated to ensure the best possible approximation of the input matrix. Weight initialization techniques and activation functions have also been modified to fit the problem. The following two models, namely DN3MF and MDSR-NMF, have been built on top of this shallow model using the notion of deep neural network architecture. In deep neural network models, a two-stage approach, namely, pre-training and stacking, has been used to achieve the robustness of the models. Novel neural network architecture has been conceived to generate the nearest possible approximation of the input along with the closest realization of the traditional NMF technique. Following this, two other deep learning models, viz., IG-MDSR-NMF and IG-MDSR-RNMF, have been designed to mimic the human-centric learning approach while ensuring a unique pair of factor matrices resulting in the closest approximation of the input matrix. In IG-MDSR-RNMF, a new type of non-negative matrix factorization technique has been formulated, called Relaxed NMF, where only one factor matrix adheres to the non-negativity criteria whereas the other one does not.

1.2 Literature survey

Over the years researchers have developed numerous techniques to find a low dimensional representation of high-dimensional data and overcome the problems of curse of dimensionality. Some of these techniques are traditional machine learning techniques necessitating human intervention and some are employing present-day deep learning techniques. There are also several procedures which fuse the benefits of traditional machine learning techniques with that of deep learning. In the following sections, some of these techniques have been described in brief.

1.2.1 Traditional dimension reduction techniques

Dimension reduction techniques can be broadly divided into two categories, viz., linear dimension reduction techniques and non-linear dimension reduction techniques. The methods using linear transformations for reducing dimension are called linear dimension reduction methods. Linear dimension reduction techniques are applicable when data lies on a linear subspace. Some of the well-known linear dimension reduction methods are Principal Component Analysis (PCA) [84, 46], Singular Value Decomposition (SVD) [35, 116], Independent Component Analysis (ICA) [14, 49, 50], Canonical Correlation Analysis (CCA) [38], Multi-dimensional Scaling (MDS) [108, 61, 91, 16, 6], Factor Analysis (FA) [101, 37], Linear Discriminant Analysis (LDA) [31, 86], Latent Semantic Analysis (LSA) [18], and Locality Preserving Projections (LPP) [44, 43], among others. Some of these most popular linear dimensionality reduction techniques, viz., PCA, MDS, LDA, CCA, and LPP use orthogonal projections for interpreting low dimensional views of high-dimensional data.

On the other hand, when data do not lie in a linear subspace, non-linear transformation methods need to be applied. These non-linear transformation methods are also known as manifold learning methods. Here, it is assumed that the data are embedded in non-linear low dimensional manifolds which lie in the higher-dimensional space. Some of the well known non-linear dimension reduction methods include Locally Linear Embedding (LLE) [88], Hessian Locally Linear Embedding (HLLE) [22], Kernel Principal Component Analysis (KPCA) [92], Nonlinear Principal Component Analysis (NLPCA) [33], Self-organizing Map (SOM) [58, 59], t-Distributed Stochastic Neighbor Embedding (t-SNE) [78], Isometric Feature Mapping (Isomap) [105], Generative Topographic Mapping (GTM) [5], Autoencoder (AE) [64, 7, 45], and Spectral Embedding (Laplacian Eigenmaps) [3].

One of the most famous and oldest dimension reduction techniques is PCA [84, 46], which is a statistical procedure. An orthogonal transformation is applied to the input data to convert a set of values of correlated variables into a set of values of linearly uncorrelated variables, called principal components. This transformation ensures the maximum possible variance along the first principal component, and after that, each succeeding component being orthogonal to the preceding components has the highest possible variance. There are several variations to the model, viz., incremental PCA

(IPCA) [1], sparse PCA (SPCA) [132], kernel PCA (KPCA) [92] and nonlinear PCA (NLPCA) [33] to name a few.

Some other dimension reduction methods, like MDS, FA and SVD, are highly related to PCA. MDS [108, 61, 91, 16, 6] is used to visually represent distances or dissimilarities among sets of objects to analyze the similarity or dissimilarity in data. MDS preserves the closeness of data points with respect to one another while projecting them in lower dimensions. Another popular statistical method for dimension reduction is FA [101, 37]. Here it is assumed that the correlations between two observed variables are the effect of several unobserved latent variables. These latent variables are called factors. FA tries to capture the maximum variability in data using a minimum number of variables by discarding the correlated variables, keeping only one of them.

Researchers have tried to deal with the dimension reduction problem in many ways; one of them is factorization/matrix decomposition. SVD [35, 116], a technique from linear algebra is popular for factorization, where a given matrix is reduced to its constituent parts. Mathematically, SVD is defined as, $\mathbf{M}_{mn} = \mathbf{U}_{mm}\mathbf{\Sigma}_{mn}\mathbf{V}^*_{nn}$, where $\mathbf{M}$ is the input matrix having real or complex values, $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices, and $\mathbf{\Sigma}$ is a diagonal matrix consisting of the singular values of $\mathbf{M}$. The problem of dimension reduction is solved by replacing $\mathbf{M}$ by $\mathbf{U}$. Another popular dimension reduction technique called LSA [18] is designed mainly for text document classification. LSA is an unsupervised linear mapping technique based on SVD. The main idea of LSA is that words of similar meaning should appear in similar pieces in the text.

ICA [14, 49, 50] is a very widely used machine learning technique that tries to find the independent components/factors from a multivariate input signal by maximizing the statistical independence of the estimated components. Another dimension reduction technique, called CCA [38], tries to identify and measure the associations among two sets of variables. CCA tries to connect these two sets of variables by finding linear combinations of variables that maximally correlate. It identifies orthogonal linear combinations of the variables within a set, which can explain the variability both within and between sets. LDA [31, 86] projects data in such a manner that the variance between data is minimized and the distance between the means of the classes is maximized. LDA tries to express dependent variables in terms of a linear combination of other features.

Nowadays a technique called manifold learning has gained popularity to deal with the problem of nonlinear dimensionality reduction. Isomap [105] is one of the earliest known approaches to manifold learning. Isomap can be described as a combination of Floyd–Warshall and MDS algorithm. Isomap projects data to a lower-dimensional representation maintaining geodesic distances among all points. Dimensionality reduction using Laplacian Eigenmaps [3] is performed using a non-linear embedding called spectral embedding. Laplacian Eigenmaps preserves locality rather than local linearity, i.e., it maps nearby inputs to nearby outputs.

Another unsupervised dimensionality reduction technique LLE [88] represents a data point as a linear combination of its neighbors preserving the original non-linear geometric feature structure. LLE can be described as a series of local PCAs, which are globally compared to identify the best non-linear embedding. When the local linear structure is identified using a Hessian-based quadratic form then the technique is called HLLE [22]. Conceptually, HLLE can be described as a modification of the Laplacian Eigenmap framework.

One common disadvantage of Isomap, Laplacian eigenmaps and LLE techniques is that these techniques only consider the neighbourhood of the training data to find the lower-dimensional representation of a data point, and thus they extrapolate very poorly. LPP [44, 43] overcomes this issue by constraining the projections as a linear projection of the input vectors.

t-Distributed Stochastic Neighbor Embedding (t-SNE) [78] is one of the widely used stochastic neighbour embedding methods for dimensionality reduction. t-SNE is mainly used to visualize high dimensional data in lower dimensional space. The algorithm first computes probability between two points in higher dimensional space in such a manner that similar points are assigned higher probability while dissimilar points are assigned lower probability. t-SNE then chooses a low dimensional embedding such that it produces a similar distribution, i.e. t-SNE tries to preserve the pairwise similarities.

SOM [58, 59] is an unsupervised neural network model mainly used for data visualization. SOM projects higher dimensional data into lower dimensional space using topological similarity. The probabilistic variant of SOM is GTM [5], which uses the expectation-maximization algorithm for learning.

The concept of autoencoder, a deep learning model, used for dimension reduction, has evolved gradually over the years [64, 7, 45]. An autoencoder is an unsupervised artificial neural network model that learns to encode the input and then learns to reconstruct the same. Thus, an autoencoder consists of an encoder module followed by a decoder module, and the middlemost layer acts as the bottleneck of the system, thus learning the latent representation of the input data. There exist several variations of this traditional autoencoder model, namely stacked autoencoders, variational autoencoders, denoising autoencoders, sparse autoencoders, adversarial autoencoders, and Wasserstein autoencoders, among others.

Some other popular dimension reduction methods are Dictionary Learning methods, techniques based on Random Projection, and Non-negative Matrix Factorization.

Dictionary learning is a branch of signal processing and machine learning. It is a type of representation learning aiming towards a sparse representation of the input data in terms of its basic elements, called atoms, and a linear combination of them. These atoms compose a dictionary. A good dictionary is characterized by its sparseness. This is why dictionary learning is popularly known as sparse dictionary learning or sparse coding [83]. Mathematically, for a given dataset $\mathbf{X}_{mn}$ the technique tries to find a dictionary $\mathbf{D}_{mp}$ and a representation $\mathbf{R}_{pn}$, such that $||\mathbf{X} - \mathbf{DR}||_F$ is optimized and $\mathbf{R}$ satisfies the sparsity criteria.

For a set of points in Euclidean space, a technique, called random projection [53], is used in mathematics and statistics for dimension reduction. A given matrix $\mathbf{X}_{mn}$ is projected to a lower k-dimensional subspace using the formula $\mathbf{X}_{kn} = \mathbf{R}_{km}\mathbf{X}_{mn}$, where $\mathbf{R}$ is a random matrix whose columns have unit length. There are two common variations to this technique, namely Gaussian random projection, where $\mathbf{R}$ is generated using a Gaussian distribution and sparse random projection, where a sparse random matrix is used.

Non-negative Matrix Factorization (NMF) [65, 66] is a traditional matrix factorization technique, which can be used for representing a hefty dataset in lower dimensional representation. NMF is used to decompose a matrix, comprising non-negative elements, into a product of two factor matrices, such that both of them contain non-negative elements only. Most of the dimensionality reduction techniques suffer from the fact that they produce feature vectors with negative components and hence the

applicability of such methods is narrowed down. NMF overcomes this issue by imposing the non-negativity constraint and hence the interpretability of the outcome gets enhanced. NMF obtains a “parts-based" low dimensional representation of a dataset.

1.2.2 Non-negative Matrix Factorization

Non-negative Matrix Factorization is used to decompose a matrix comprising non-negative elements into a product of two factor matrices, such that both of these factor matrices contain non-negative elements only. That is, a matrix $\mathbf{X}_{m\times n}$ is decomposed into

$$\mathbf{X} \approx \mathbf{WH} \tag{1.2.1}$$

where, $\mathbf{W}_{m\times k}$ is the first factor matrix, called basis matrix or feature matrix, and $\mathbf{H}_{k\times n}$ is the second factor matrix, called coefficient matrix or activation matrix. The cost function D, a scalar error measure based on the Euclidean distance, is defined as

$$D(\mathbf{X}, \mathbf{WH}) = ||\mathbf{X} - \mathbf{WH}||_F^2 \tag{1.2.2}$$

The goal is to minimize the error D under the non-negativity constraint, i.e., minimize $D(\mathbf{X}, \mathbf{WH})$ with respect to $\mathbf{W}, \mathbf{H}$, subject to elements of $\mathbf{W}, \mathbf{H} \geq 0$. In this context, the algorithms mainly use multiplicative update rules based on the gradient descent technique. Instead of jointly updating both $\mathbf{W}$ and $\mathbf{H}$, the algorithms update one matrix assuming that the other matrix is constant, and vice-versa. This scheme is called block-coordinate descent and is given by

$$\mathbf{W}(t+1) \leftarrow \mathbf{W}(t) - \eta_{\mathbf{W}(t)} \circ \nabla_{\mathbf{W}(t)} D(\mathbf{X}, \mathbf{W}(t)\mathbf{H}(t)) \tag{1.2.3}$$

and

$$\mathbf{H}(t+1) \leftarrow \mathbf{H}(t) - \eta_{\mathbf{H}(t)} \circ \nabla_{\mathbf{H}(t)} D(\mathbf{X}, \mathbf{W}(t)\mathbf{H}(t)), \tag{1.2.4}$$

where $\circ$ denotes the Hadamard product (element-by-element product), $\nabla_{\mathbf{W}}$ and $\nabla_{\mathbf{H}}$ are the gradient operators with respect to $\mathbf{W}$ and $\mathbf{H}$ respectively. The respective learning rates are denoted by $\eta_{\mathbf{W}}$ and $\eta_{\mathbf{H}}$, and t denotes the iteration count. To make the expressions more readable, we drop the iteration count variable t from now onward.

Update rules based on Euclidean distance are

$$\mathbf{W} \leftarrow \mathbf{W} + \eta_{\mathbf{W}} \circ (\mathbf{X}\mathbf{H}^T - \mathbf{W}\mathbf{H}\mathbf{H}^T) \tag{1.2.5}$$

and

$$\mathbf{H} \leftarrow \mathbf{H} + \eta_{\mathbf{H}} \circ (\mathbf{W}^T\mathbf{X} - \mathbf{W}^T\mathbf{W}\mathbf{H}) \tag{1.2.6}$$

The learning rates are defined in such a manner that any subtraction disappears from the update rule to satisfy the non-negativity criteria. That is,

$$\eta_{\mathbf{W}} = \mathbf{W} \oslash (\mathbf{W}\mathbf{H}\mathbf{H}^T) \tag{1.2.7}$$

and

$$\eta_{\mathbf{H}} = \mathbf{H} \oslash (\mathbf{W}^T\mathbf{W}\mathbf{H}) \tag{1.2.8}$$

Here, $\oslash$ denotes element by element division. Thus the updated rules become

$$\mathbf{W} \leftarrow \mathbf{W} \circ ((\mathbf{X}\mathbf{H}^T) \oslash (\mathbf{W}\mathbf{H}\mathbf{H}^T)) \tag{1.2.9}$$

and

$$\mathbf{H} \leftarrow \mathbf{H} \circ ((\mathbf{W}^T\mathbf{X}) \oslash (\mathbf{W}^T\mathbf{W}\mathbf{H})) \tag{1.2.10}$$

NMF algorithms can be divided into four categories [119], viz. basic NMF, constrained NMF, structured NMF and generalized NMF. When only the non-negativity criteria are imposed, those algorithms are called basic NMF. In constrained NMF, some additional constraints are used as regularization. Sparse NMF [82, 47], orthogonal NMF [73, 20], discriminant NMF [51, 126, 60] and NMF on manifold algorithms [11, 10] are some of the popular constrained NMF techniques. When the standard factorization formulations are modified, the algorithms are called structured NMF. Weighed NMF [56, 79, 130], convolutive NMF [97, 98] and nonnegative matrix trifactorization techniques [124] fall under the umbrella of structured NMF algorithms. In generalized NMF, the NMF technique is extended in a broader sense. Some generalized NMF models are semi-NMF [21], nonnegative tensor factorization [120, 40, 93], nonnegative matrix-set factorization [72, 71], and kernel NMF [128, 9, 75], among others.

1.2.3 Neural network based approaches for NMF

NMF [65, 66] is a typical example of an iterative algorithm. Over the years, researchers have tried to increase the effectiveness of the NMF algorithm by clubbing its benefits with that of deep learning. Here we present some of the current state-of-the-art approaches developed by researchers using NMF.

Trigeorgis et al. have devised a model named semi-NMF [109] that automatically learns the attribute hierarchy of a dataset as well as the representations that are suitable for clustering. Deep Semi-NMF [110], a deep neural network-based variant of semi-NMF, and Deep WSF [110], a semi-supervised version of the approach, have also been developed. Deep WSF makes use of prior knowledge to some extent for each dataset attribute.

Ye et al. have attempted to tackle the community detection problem by learning the hierarchical mappings between the original network and the final community assignment using a Deep Autoencoder-like NMF (DANMF) [122] model. The NMF-based noise reduction approach has inspired the algorithm. Song et al. have presented a layer-wise feature learning approach based on stacked NMF layers [100]. Sparsity constraints in a multi layer NMF model have been used by Guo et al. to learn localised features [36].

Nonsmooth Nonnegative Matrix Factorization (nsNMF) [125] is a deep autoencoder like architecture that learns both part-based and hierarchical features, and produces more localised and less overlapped feature representations. Yang et al. have developed DAutoED-ONMF, a deep autoencoder network for Orthogonal NMF (ONMF) [121], to hierarchically extract features from source datasets while utilising the benefits of shallow ONMF model to achieve superior learning capability. Zhao et al. have designed a deep NMF model that deep factors basis image matrices to learn the underlying basis images and attempts to find patterns in complex data [131].

Using graph regularisation, the Deep Grouped NMF (DGNMF) [127] model learns different level attributes of data while preserving local information. Shu et al. have developed Deep Semi-NMF-EP to successfully represent high-dimensional data while retaining data elasticity using graph regularizers [94]. An NMF-based feature extraction technique incorporated in CNN has been developed by Lee et al. [70].

The researchers have used a stacked autoencoder incorporated in the NMF framework to achieve task-specific nonlinear dimension reduction [129]. To get a more discriminative, robust and generalised feature representation as well as dimension reduction, Tong et al. have integrated both global and central loss functions of the soft label constraint matrix in the objective function of the model [107].

Shu et al. [96] have attempted to cluster data by learning an optimum adaptive graph based on the local neighbourhood relationship among samples at each layer of the deep network rather than the predetermined fixed graph. The adaptive graph regularizer has been incorporated into the deep matrix factorization framework, called adaptive graph regularized deep semi-nonnegative matrix factorization (AGRDSNMF). Thus, the AGRDSNMF technique not only uses a deep framework to uncover hidden features but also fully exploits prior knowledge of data. A new scRNA-seq data representation approach for scRNA-seq data clustering, called Robust Graph regularised Non-Negative Matrix Factorization with Dissimilarity and Similarity constraints (RG-NMF-DS), has been introduced by Shu et al. [95]. It tackles the issue of high dimensionality and noise in scRNA-seq data by finding their embedded manifold structure using regularisation.

The majority of the research studies described above have applications on image datasets [109, 110, 125, 127, 131, 129, 94, 107, 36]. However, in addition to image datasets, Yang et al. [121] have shown applications in textual databases and networks. On the other hand, the research work presented by Ye et al. [122] deals with applications on networks for community discovery. Acoustic signals have been used as the application area by Lee et al. [70]. Song et al. [100] have shown application on document classification. Shu et al. [96] have demonstrated their application in the image as well as textual datasets. scRNA-seq data have been used for clustering by the RG-NMF-DS model [95].

1.3 Motivation of the thesis

Most of the dimensionality reduction techniques suffer from the fact that they may produce feature vectors with negative components and hence the applicability of such methods is limited. Non-negative Matrix Factorization [65, 66], a popular dimension

reduction method, overcomes this issue by imposing non-negativity constraints and hence the interpretability of the outcome gets enhanced. We have used NMF to dimensionally reduce a given dataset over unconstrained dimension reduction techniques in order to exploit the advantages of non-negative feature characteristics, as follows:

- Sparsity in the feature matrix can be enforced through non-negativity. Unconstrained decomposition normally results in non-zero factors even though the given attribute(s) does (do) not contribute to a signal, whereas in the case of non-negative techniques, zero factors are generated. Thus, sparse representations are one of the evident characteristics of non-negative decomposition. When we intend to uncover distinct feature sets or sample relationships, sparse representations are beneficial.

- Non-negativity assures that factors do not counterbalance one other. For example, if one of the factors overcorrects a signal, another factor may attempt to counter-correct to compensate. When factors can only be positive or zero, they cannot counter-balance and only additive signals can be explained.

- On top of the above advantages, there are theoretical connections between NMF and k-means, providing strong support and theoretical foundations for NMF-based clustering [19, 20, 21, 74]. It has been proved that NMF is equivalent to a relaxed k-means clustering yielding a soft partitioning [29]. It has also been shown that Orthogonal NMF amounts to k-means clustering [19, 74]. Hence, NMF can not only be used as a dimension reduction method, it can also be used as a clustering algorithm.

The aspiration of simulating the factorization behaviour of the traditional NMF technique ensuring the outcome of a unique pair of factor matrices of the reconstructed input matrix has motivated us for the progressive development of the models. We have fused the advantages of conventional iterative learning with those of deep learning in a way that resembles the trait of human learning. Traditional NMF technique uses a block coordinate descent scheme to update the factors of the input matrix. This limitation has been solved by updating both factors simultaneously using neural network architecture. While learning, humans always attempt to disintegrate the concepts

into smaller fragments and try to learn hierarchically referring back to the original details frequently ensuring the correctness of the learning. We have attempted to simulate such human-specific characteristics throughout our design and development of the models.

1.4 Scope of the thesis

This thesis is a comprehensive attempt to fuse the advantages of the traditional NMF technique and deep learning aiming toward dimension reduction. In this direction, five neural network models have been developed. The models have been built on top of the previous one overcoming any shortcomings of the predecessor. The current chapter defines the problem and sheds some light on the developed models. The next chapter describes the datasets, experimental procedures and validation strategies used to establish the effectiveness and superiority of the models. Chapters 3 to 7 constitute the contributory part of the thesis, explaining and establishing the models. Finally, Chapter 8 concludes the thesis with some directions for future work. The development of different neural network architectures over the upcoming chapters is summarized graphically in Figure 1.1.

1.4.1 Chapter 2 - Description of datasets, experimental procedure and evaluators

To establish the effectiveness and superiority of the novel models, developed in the thesis, various types of experiments have been performed on a number of datasets over different other well-known dimension reduction techniques. The results have also been justified using several metrics. The datasets have been described in Chapter 2 along with the experimental setup and experimental procedure.

1.4.2 Chapter 3 - Non-negative Matrix Factorization Neural Network (n^2MFn^2) [24, 25]

A shallow neural network model, called Non-negative Matrix Factorization Neural Network (n^2MFn^2), has been developed aiming towards low rank approximation for

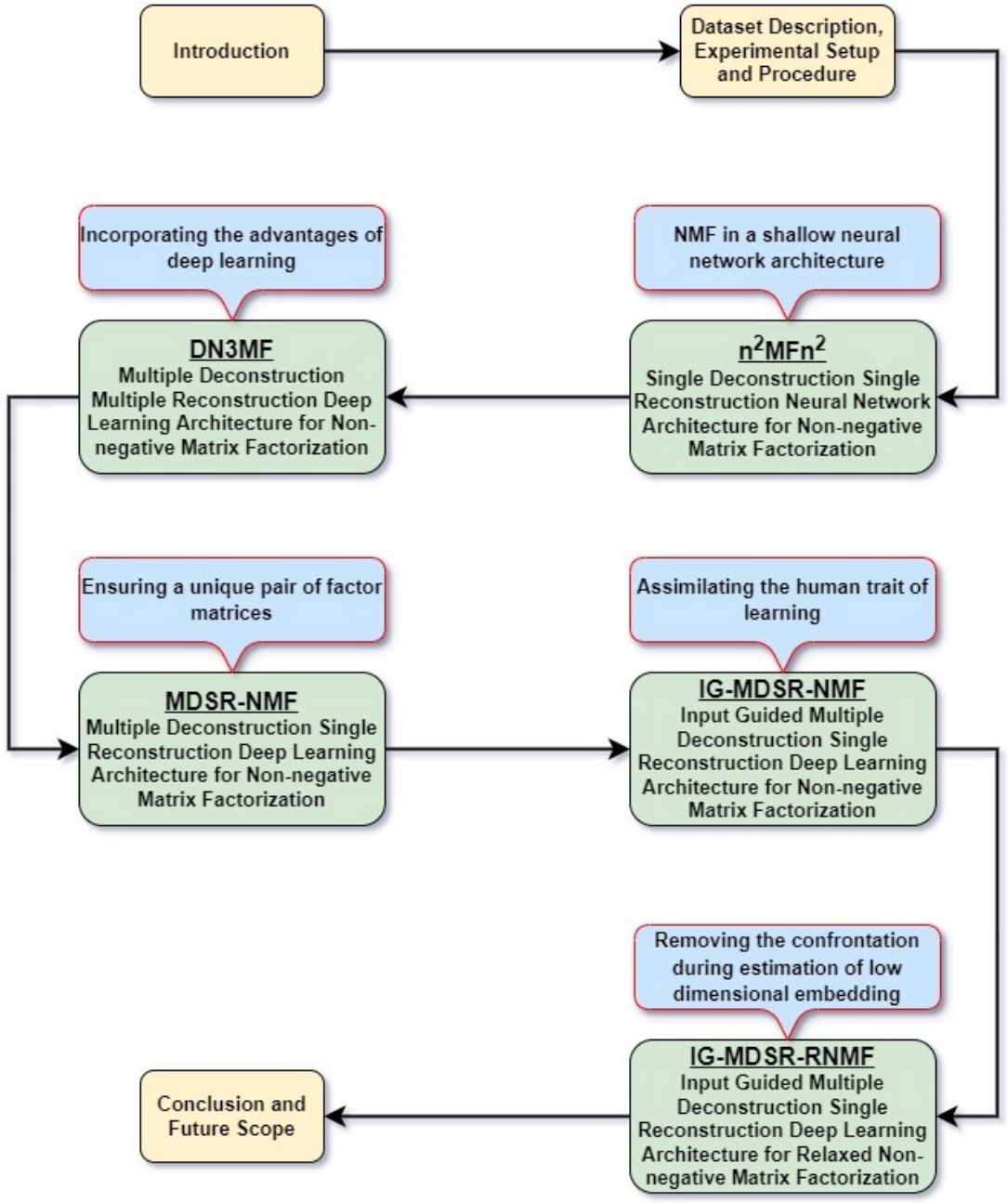

FIGURE 1.1: Graphical summarization of the thesis

non-negative matrix factorization under neural network framework. The architecture consists of a single deconstruction layer and a single reconstruction layer. Hence the architecture is categorized as a single deconstruction single reconstruction neural network model. The architecture has a single hidden layer, constructed in such a way that serves as the bottleneck layer of the model. With the help of hierarchical learning, the pervasiveness of the non-negative input data has been processed to produce a part-based, sparse, and meaningful representation. A modification of the He initialization

technique to initialize weights maintaining the non-negativity criteria of the model, has also been proposed. A necessary modification of the ReLU activation function has been made to suppress all neurons in a layer from adjusting their weights simultaneously. Regularization has been used in the design of the objective function of the model to minimize the risk of overfitting. To demonstrate the competency of n^2MFn^2, the results have been analysed and compared to those of six other leading dimension reduction techniques on five popular datasets in terms of local structure preservation of data in low rank embedding, as well as in the context of downstream analyses involving classification and clustering. It has also been tested and demonstrated that low dimensional embedding performs better than the original data, which supports the necessity of dimension reduction. The statistical significance of the findings has also been determined. The analysis of the same has justified the effectiveness and superiority of the model over some others. Additionally, the computational complexity and convergence analysis of the model have been discussed.

1.4.3 Chapter 4 - Deep Neural Network for Non-negative Matrix Factorization (DN3MF) [27]

Continuing with the aim of dimension reduction and mitigating the disadvantages of shallow neural network architecture while incorporating the advantages of deep neural network architectures, a deep learning model, called Deep Neural Network for Non-negative Matrix Factorization (DN3MF), has been developed for the task of NMF. There are two stages of the model, namely, pretraining and stacking. The pretraining stage is accomplished by a shallow neural network architecture and the stacking stage is a deep neural network architecture. Non-negative input data have been processed using hierarchical learning to generate part-based sparse and meaningful representation. The novel design of DN3MF ensures the non-negativity requirement of the model. The use of Xavier initialization technique solves the exploding or vanishing gradient problem. The objective function of the model has been designed employing regularization, ensuring the best possible approximation of the input matrix. A novel adaptive learning mechanism has been developed to accomplish the objective of the model. The superior performance of DN3MF has been established by comparing the results obtained by the model with that of seven other well-established dimension reduction algorithms on five well-known datasets in terms of preservation of the local structure of data in low

rank embedding, and in the context of downstream analyses using classification and clustering. Furthermore, low dimensional embedding has been evaluated and shown to outperform the original data, supporting the need for dimension reduction. The statistical significance of the results has also been established. The outcome clearly demonstrates DN3MF's superiority over compared dimension reduction approaches in terms of both statistical and intrinsic property preservation standards. The comparative analysis of all eight dimensionality reduction algorithms including DN3MF with respect to the computational complexity and a pictorial depiction of the convergence analysis for both stages of DN3MF have also been presented.

1.4.4 Chapter 5 - Multiple Deconstruction Single Reconstruction Deep Neural Network Model for Non-negative Matrix Factorization (MDSR-NMF) [26]

DN3MF is able to diminish the shortcomings of a shallow neural network architecture but fails to produce a unique pair of factor matrices. To address this issue, a novel deep-learning architecture, named MDSR-NMF, has been designed with multiple deconstruction and single reconstruction layers for non-negative matrix factorization aimed at low rank approximation. This design ensures that the reconstructed input matrix has a unique pair of factor matrices. The two-stage approach, namely, pretraining and stacking, aids in the robustness of the architecture. The sigmoid function has been adjusted in such a way that it fulfils the non-negativity criteria and also helps to alleviate the data-loss problem. Xavier initialization technique aids in the solution of the exploding or vanishing gradient problem. The objective function involves a regularizer that ensures the best possible approximation of the input matrix. The superior performance of MDSR-NMF, over eight well-known dimension reduction methods, has been demonstrated extensively using five datasets with an emphasis on maintaining the local structure of the data through low rank embedding and considering the implications for classification and clustering as a part of downstream analyses. The requirement of dimension reduction is further supported by experiments showing that reduced dimensional embedding has outperformed the original data. Furthermore, the statistical significance of the results has been demonstrated. Computational complexity and convergence analysis have also been presented to establish the model.

1.4.5 Chapter 6 - Input Guided Multiple Deconstruction Single Reconstruction neural network for Non-negative Matrix Factorization (IG-MDSR-NMF) [28]

Referring back to the original text in the course of hierarchical learning is a common human trait that ensures the right direction of learning. The model in this chapter has been developed based on the concept of Non-negative Matrix Factorization inspired by this idea. The aim is to deal with high-dimensional data by discovering its low rank approximation by determining a unique pair of factor matrices. The model, named Input Guided Multiple Deconstruction Single Reconstruction neural network for Non-negative Matrix Factorization (IG-MDSR-NMF), ensures the non-negativity constraints of both factors. The competency of preserving the local structure of data in its low rank embedding produced by the model has been appropriately verified. The superiority of low dimensional embedding over that of the original data justifying the need for dimension reduction has been established. The primacy of the model has also been validated by comparing its performance with that of nine other established dimension reduction algorithms on five popular datasets. Additionally, the statistical significance of the results has been determined. Moreover, the computational complexity of the model and convergence analysis have also been presented testifying to the supremacy of the model.

1.4.6 Chapter 7 - Input Guided Multiple Deconstruction Single Reconstruction neural network for Relaxed Non-negative Matrix Factorization (IG-MDSR-RNMF) [28]

The model IG-MDSR-NMF ensures the non-negativity constraints of both factor matrices. In contrast, Input Guided Multiple Deconstruction Single Reconstruction neural network for Relaxed Non-negative Matrix Factorization (IG-MDSR-RNMF) has introduced a novel idea of factorization with only the basis matrix adhering to the non-negativity criteria. This relaxed version helps the model to learn more enriched low dimensional embedding of the original data matrix. The ability of the model to maintain the local structure of data in its low rank embedding has been suitably verified. It has been demonstrated that low dimensional embedding performs better than the original

data, which supports the necessity of dimension reduction. By comparing the performance of the model on five widely used datasets with ten other well-known dimension reduction techniques, the model's superiority has also been confirmed. Furthermore, the statistical significance of the data has been shown. Additionally, the model's computational complexity and convergence analysis have been provided, attesting to its superiority.

1.4.7 Chapter 8 - Conclusions and Scope of Further Research

Finally, we offer closing thoughts on the neural network models developed in this thesis and the outcomes they produced in Chapter 8. We provide an understanding of the constraints included in every designed architecture. In this chapter, we also provide a quick overview of the future directions of the thesis.

1.5 Conclusions

In this chapter of the thesis, we have described the scope and problem statement of the thesis. The chapter also covers some traditional and current state-of-the-art methodologies to solve the problem of dealing with high dimensional data. A brief explanation of the traditional Non-negative Matrix Factorization technique has also been discussed. Finally, the outline of the thesis along with a brief introduction of the proposed models have been provided. To establish the proposed models a number of experiments have been performed. These experimental procedures together with the description of the datasets and performance metrics have been delineated in the following chapter.

Chapter 2

Description of datasets, experimental procedure and evaluators

2.1 Introduction

The problem statement and scope of the thesis have been covered in Chapter 1. A few current state-of-the-art and conventional techniques for handling high-dimensional data have also been described in Chapter 1. Additionally, a brief description of the conventional Non-negative Matrix Factorization method and proposed methodologies has also been covered. In order to demonstrate the effectiveness and superiority of the models developed in the thesis, several experiments have been carried out. This chapter outlines these experimental strategies along with the description of datasets and performance measures (evaluators).

The efficacy of dimensionality reduction with the models (n^2MFn^2, DN3MF, MDSR-NMF, IG-MDSR-NMF and IG-MDSR-RNMF) has been studied in two ways on five popular datasets. First, the extent to which these models can preserve the local structure of data after dimension reduction, has been compared with that of six different dimension reduction approaches. The quality of the low rank representation has also been assessed against the original dataset to justify the need for dimension reduction. Second, the discriminating ability of reduced feature space obtained by these models

has been compared with that of six different dimension reduction techniques in terms of classification and clustering. Three of the six dimension reduction methods are traditional dimension reduction methodologies; one is a classic NMF technique and the other two are current state-of-the-art neural network based NMF implementation techniques. The same testing procedure has been followed for all the five models, developed in this thesis, to determine their efficacy over others.

While comparing the performance of the designed models, along with six other dimension reduction techniques, each designed model has also been compared with the previously developed models. That is, for the first designed model, n^2MFn^2, the performance of n^2MFn^2 has been tested with that of six other dimension reduction techniques. For DN3MF, along with six other dimension reduction techniques, the performance of DN3MF has also been compared with n^2MFn^2. That is, the efficacy of DN3MF has been tested with a total of seven dimension reduction techniques. Similarly, the efficiency of MDSR-NMF has been established by comparing its performance with n^2MFn^2, DN3MF and six other models as mentioned above, i.e., a total of eight dimension reduction techniques. In the same way, while working with IG-MDSR-NMF, the performance of IG-MDSR-NMF has been compared to that of nine other dimension reduction techniques including MDSR-NMF and others. Finally, for IG-MDSR-RNMF, a total of ten dimension reduction techniques have been used to compare its performance.

The remaining part of the chapter is organized as follows. Data sources are narrated in Section 2.2. The technique of data preparation has been described in Section 2.3. Section 2.4 delineates the details of the experimental setup. The methodology to test the extent of local structure preservation is described in Section 2.5. The experimental procedure for evaluating the discriminating ability of dimensionally reduced datasets, in terms of classification and clustering, has also been described in Section 2.5. Different performance metrics used to justify the efficacy of the proposed models have also been described in Section 2.5. Finally 2.6 brings the chapter to a conclusion.

2.2 Data sources

Four popular datasets, viz., Gastrointestinal Lesions in Regular Colonoscopy (GLRC) dataset [81], Online News Popularity (ONP) dataset [30], Parkinson's Disease Classification (PDC) dataset [90] and Student Performance (SP) dataset [15] have been downloaded from the UCI machine learning repository [23]. The MovieLens dataset has been acquired from the GroupLens research lab website. GroupLens is a research lab at the Department of Computer Science and Engineering at the University of Minnesota. The effectiveness of dimensionality reduction by the models has been evaluated using these datasets.

2.2.1 Gastrointestinal Lesions in Regular Colonoscopy (GLRC) dataset

The dataset [81] consists of ground truth and features extracted from a colonoscopic video database of gastrointestinal lesions. Expert image inspection and histology have been used for modelling the ground truth. The dataset has 76 samples. There are 698 features in all, with 2D textural properties of the lesions accounting for the first 422 attributes, 2D colour features of the lesions accounting for the next 76 and 3D shape features of the lesions accounting for the last 200. Each sample used two distinct types of lighting. We have considered the kind of light as a feature while performing the computation. Thus, the data matrix has $2 \times 76 = 152$ rows and $1 + 698 = 699$ columns. The dataset consists of three types of lesions: hyperplasic, adenoma and serrated adenoma. Hyperplastic lesions are benign, while, adenoma and serrated adenoma lesions are malignant. As a result, we treat the dataset as a two-class (benign and malignant) problem [52].

2.2.2 Online News Popularity (ONP) dataset

The dataset [30] contains multiple sets of features extracted from Mashable articles published between January 7, 2013 and January 7, 2015. The collection includes 39644 entries for online articles. Each item is defined by a total of 60 features. URL is one of them, which we have not added because of its uniqueness to each article. The other 59 features are numeric, based on the article's structure and content, with the last one

being the number of days between article publication and dataset acquisition. As a consequence, the dataset size is 39644×59. The samples in the dataset are divided into two categories namely, popular and unpopular, based on the number of shares of each article [62, 63].

2.2.3 Parkinson's Disease Classification (PDC) dataset

A study employing voice recordings of 252 individuals was undertaken at the Department of Neurology at Cerrahpaşa Faculty of Medicine, Istanbul University, Istanbul, Turkey [90]. There were 188 patients with Parkinson's disease, while the remaining 64 did not have the ailment. As a consequence, the samples are divided into two groups. Each sample is described by 754 features. Time-frequency features, mel-frequency cepstral coefficients, wavelet transform-based features, vocal fold features and wavelet transform features with a configurable Q factor were employed. These characteristics have provided clinically relevant information for Parkinson's disease assessment. We have not considered the patient identification number as a feature. The experiment was repeated three times for each individual. Thus, the data matrix size is 756×753.

2.2.4 Student Performance (SP) dataset

The dataset [15] includes student data collected from two Portuguese secondary schools in the Alentejo region of Portugal in 2005 and 2006. It uses data from two sources: student grade reports and student replies to a series of questionnaires. The features include different student grades, as well as demographic, social and school-related information. Here, we have considered the Math performance of 395 students. These data are represented by 29 different features. The dataset has a feature identifying the student's school. Each student receives three unique grades: G1, G2 and G3 representing the first period grade, second period grade and final grade respectively. Grades are simply numerical values ranging from 0 to 20. G1 and G2 have been viewed as features, whereas G3 has been used as a target attribute. Thus, each student has been represented by 32 features, resulting in a database size of 395×32. We have designed the problem as a two-class problem based on G3, with a student passing if G3 is more than or equal to 10 and failing otherwise.

2.2.5 MovieLens dataset

The MovieLens datasets were prepared by the members of the GroupLens Research Project at the University of Minnesota [39]. This dataset comprises 100,000 ratings for 1682 movies from 943 people and the remaining entries are unavailable. Thus, the dataset comprises 943 samples, each having 1682 features. The ratings range from 1 to 5, with 1 being the lowest rating and 5 being the highest. Every user has rated at least twenty movies. The dataset also includes basic demographic information such as the users' age, gender, employment and zip code. The data was gathered over seven months, from September 19, 1997, to April 22, 1998, using the MovieLens website (movielens.umn.edu). Users with less than 20 ratings or with missing demographic information were removed from this dataset during the cleaning process. We have considered gender as the classifying attribute, hence the dataset is classified as a two-class problem.

The following table (Table 2.1) summarizes the datasets described above.

TABLE 2.1: Summary of the datasets used for experimentation.

Dataset	No. of samples	No. of features	doi/URL
GLRC	152	699	10.24432/C5V02D
ONP	39644	59	10.24432/C5NS3V
PDC	756	753	10.24432/C5MS4X
SP	395	32	10.24432/C5TG7T
MovieLens	943	1682	https://grouplens.org/

2.3 Data preparation

Consider a given data matrix, $\mathbf{U} = [u_{pi}]_{m\times n'}$. We process $\mathbf{U}$ to generate a matrix $\mathbf{X} = [x_{pi}]_{m\times n}$ with each element being non-negative. We use a methodology similar to the folding data method described in [55] to carry out this task. This approach uses two columns of $\mathbf{X}$ to represent each column of $\mathbf{U}$. That is, i^{th} and $(n'+i)^{th}$ columns of $\mathbf{X}$ correspond to i^{th} column of $\mathbf{U}$. Entries in every column of $\mathbf{U}$ can be either positive or negative. Positive values from i^{th} column of $\mathbf{U}$ are kept in i^{th} column of $\mathbf{X}$, whereas the absolute form of the negative values is stored in $(n'+i)^{th}$ column of $\mathbf{X}$. The remaining empty cells in i^{th} and $(n'+i)^{th}$ columns of $\mathbf{X}$ are filled with zeros. To obtain the original elements of the i^{th} column of $\mathbf{U}$, subtract the elements of the $(n'+i)^{th}$ column of $\mathbf{X}$ from

the elements of its i^{th} column. As a result, the number of columns in $\mathbf{X}$ is exactly twice that of $\mathbf{U}$, i.e., $n = 2n'$. Furthermore, it should be noted that in this manner, exactly half of the elements of $\mathbf{X}$ are zero, resulting in a sparse matrix. Each row of $\mathbf{X}$ is now used as input to the model.

2.4 Experimental setup

The proposed models n^{2}MFn^2, DN3MF and MDSR-NMF have been implemented from scratch in Python (version 3.7) using some basic libraries such as numpy. IG-MDSR-NMF and IG-MDSR-RNMF have been implemented in Keras (version 2.13.1). Different software libraries, such as Scikit-learn, TensorFlow and Keras have been employed as and when needed to program several other existing dimensionality reduction techniques. Plots of various figures have been generated using Python programming language.

The data matrices have been preprocessed before using them as input to the models. Data matrices have been normalised using the Z score normalisation technique. If a classification/clustering performance score generates an error, i.e., fails to produce a valid output for any reason, the lowest possible value of that metric is assigned in that place during computation. This step ensures that the classification/clustering performance scores for various dimension reduction algorithms for a given dataset are consistent.

2.4.1 Z score normalisation

The values of a feature are scaled using this method to have a mean of 0 and a standard deviation of 1. To compute this, for each feature value, subtract the mean of the feature from it, and then divide the result by the standard deviation of the feature.

$$New\_value = \frac{(x - \mu)}{\sigma} \tag{2.4.1}$$

where x represents the original value, the mean of the feature is represented by μ and σ denotes the standard deviation of the corresponding feature.

2.5 Experimental procedure

Six state-of-the-art dimension reduction techniques have been used to compare the performance of the models. They include Autoencoder (AE) (with one hidden layer, the number of nodes in the hidden layer being the dimension of the transformed space), Principal Component Analysis (PCA), Uniform Manifold Approximation and Projection (UMAP), traditional NMF, Semi-NMF [109] and Deep Semi-NMF (DS-NMF) [110]. Let these 6 dimensionality reduction techniques, including n^2MFn^2, be placed in set T and a dimension reduction technique (i.e., an element of T) be denoted by $\dot{T}$. So, for n^2MFn^2, $|T| = 7$. As mentioned above, for every progressive development of the methods, each new method is also compared with the previously developed methods. Thus, for DN3MF, $|T| = 8$, for MDSR-NMF, $|T| = 9$, for IG-MDSR-NMF, $|T| = 10$, and for IG-MDSR-RNMF, $|T| = 11$.

A dataset $\mathbf{X}$, containing m samples, each being described by n' features, is reduced to a dimension r ($r < n'$) determined using a random factor f $(0 < f < 1)$ and computed as $r = \lfloor n' \times f \rfloor$. For a certain value of f, the dataset $\mathbf{X}$ is dimensionally reduced using all $\dot{T}$ in T. If we want to reduce to a specific dimension r, the factor f is calculated accordingly. Thus, there are $|T|$ dimensionally transformed datasets for a value of f, each having the same r value, denoted by $\mathbf{X}_r(\dot{T})$. The effectiveness of the proposed model will now be illustrated on these transformed datasets.

The performance of dimension reduction by the proposed models has been demonstrated in two parts. Firstly, the quality of dimension reduction by the models has been quantified by comparing their ability to retain the local structure of data and by justifying the need for dimension reduction over the original data. Secondly, the effectiveness of the dimensionally reduced dataset is explored for downstream analyses, like classification and clustering. Each of these experiments has been performed for 5 randomly chosen f values and the mean values of the results have been presented for analysis. Additionally, the corresponding p-values have also been computed to establish the statistical significance of the results.

2.5.1 Quantifying the quality of low dimensional embedding

The quality of low dimensional embedding by the proposed models has been investigated in two ways, viz., studying the ability to preserve the local structure of data by using trustworthiness metric and comparing the effectiveness of dimension reduction by classification/cluster performance metrics compared to the original data.

2.5.1.1 Local structure preservation

The ability to preserve the local structure of data after dimension reduction by the proposed models over that of other dimension reduction approaches has been computed and compared using the trustworthiness score.

Trustworthiness

Trustworthiness is a metric to measure the extent of local structure retention in the latent space representation of the data with reference to the original data [111, 112, 85]. The value of trustworthiness lies between 0 and 1, and is defined as

$$T(k) = 1 - \frac{2}{nk(2n - 3k - 1)} \sum_{i=1}^{n} \sum_{j \in \mathcal{N}_i^k} \max(0, (r(i,j) - k)) \tag{2.5.1}$$

Here, for each sample i, $\mathcal{N}_i^k$ is the set of its k nearest neighbours in the output space, and every sample j is its $r(i,j)^{th}$ nearest neighbour in the input space. In other words, any unexpected nearest neighbour in the output space is penalised in proportion to their rank in the input space. The higher the trustworthiness score, the better the low rank representation, i.e., the better the dimension reduction technique is.

2.5.1.2 Decision making: Comparison with the original data

The efficacy of dimension reduction by the proposed models has been judged by performing classification and clustering on low dimensional embedding produced by them as well as on the original data, and then quantifying the performances using different classification and cluster validity metrics. This study demonstrates why the dimension

reduction is necessary highlighting the fact that the usability of the data increases with the low rank representation of the same.

Classification

The reduced datasets by the proposed models have been classified using four well-known classification methods: K-Nearest Neighbor (KNN), Multilayer Perceptron (MLP), Naive Bayes (NB) and Quadratic Discriminant Analysis (QDA). Four classification performance metrics have been utilized to examine the quality of the classification performed by the aforementioned classifiers. These measures are Accuracy (ACC), Cohen-Kappa score (CKS), F1 score (FS) and Matthews Correlation Coefficient (MCC). Thus, we get a classification performance score by performing classification using a classifier and validating the outcome using a classification performance measure. The same procedure has been followed over the original data as well, and thus, we get a similar performance score for the original data. These performance scores have also been compared to demonstrate the superiority of the dimensionally reduced dataset over the original data, establishing the necessity of dimension reduction.

Clustering

Likewise, the reduced datasets by the proposed models have been clustered using four well-known clustering techniques: Mini Batch k-Means (MBkM), Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH), Gaussian Mixture Models (GMM) and Fuzzy c-Means (FcM). To assess the quality of the results produced by them, four cluster validity indices have been used. These indices are Adjusted Mutual Information score (AMI), Adjusted Rand index (ARI), Jaccard index (JI) and Normalized Mutual Information score (NMI). Hence, we get a cluster validity score by performing clustering using a clustering algorithm and validating the outcome employing a cluster evaluation index. Furthermore, we get a similar performance score by following the same procedure over the original data. It has also been established that dimension reduction is necessary by comparing these performance scores to show that the dataset with reduced dimension is superior to the original one.

2.5.2 Downstream analyses and statistical significance: Comparison with other models

By performing classification and clustering on the low dimensional embedding generated by the proposed models and also that generated by the other $|T|$ dimension reduction techniques, the effectiveness of dimension reduction has been assessed. Each reduced dataset $\mathbf{X}_r(\dot{T})$ has been classified and clustered using the aforesaid four well-known classification and clustering methods. As previously mentioned, different metrics measuring classification and cluster performances have been used to quantify the same. Thus, for each $\mathbf{X}_r(\dot{T})$, we get a classification/clustering performance score by performing classification/clustering using a classification/clustering algorithm and validating the outcome using a classification/clustering performance metric. This part of the experimentation aims to determine the superiority of dimension reduction by the proposed models using different types of classification and clustering algorithms.

The classification/clustering performance results by the proposed models and that of other dimension reduction algorithms have also been tested for statistical significance. For this, pairwise p-values have been computed. Each of the other dimension reduction algorithms considered for competing with the proposed model has been paired with the proposed model for p-value calculation.

For example, when we are justifying the efficacy of n^{2}MFn^2, for a dataset, for each classification/cluster technique and each classification/cluster performance measure we have a set of 5 values corresponding to 5 randomly chosen f values. Similarly, there are other 6 sets of classification/cluster performance measures for each of the competing dimension reduction techniques, where each set has 5 elements. Thus, there is a total of $1 + 6 = 7$ sets of results. The cardinality of each set is 5. For statistical significance testing in terms of p-values, the set of results of n^{2}MFn^2 has been compared with the set of results of AE to get a single p-value. Similarly, the set of results of n^{2}MFn^2 has also been compared to that of PCA, UMAP, NMF, Semi-NMF and DS-NMF. Thus for n^{2}MFn^2, there are 6 p-values for each classification/cluster technique and each classification/cluster performance measure. As there are 4 classification/clustering algorithms, a total of $6 \times 4 = 24$ p-values will be generated for each classification/cluster performance metric.

A p-value below a predefined threshold is considered an indicator of statistical significance [17]. Here we have taken the threshold as 0.05. That is, when a p-value is less than 0.05, we say the two sets of results are independent of each other. We have counted the number of significant p-values against each classification/cluster performance metric for a dataset, and have reported the same in tabular format. Thus, for n^2MFn^2, the entries of the table represent the count of significant p-values from a total of 24 cases. Similarly, for DN3MF the total number of cases is $7 \times 4 = 28$, and $8 \times 4 = 32$ for MDSR-NMF. The same counts for IG-MDSR-NMF and IG-MDSR-RNMF are $9 \times 4 = 36$ and $10 \times 4 = 40$ respectively.

2.5.3 Performance metrics (evaluators)

The performance metrics used to justify the efficacy of the proposed models have been described hereunder.

Confusion Matrix

Confusion matrix is used in machine learning to assess the performance of a classification model. A summarization of the performance of a machine learning model on a set of test data can be represented using a confusion matrix. It is a way to present the statistics of accurate and inaccurate predictions of the model. The following four metrics are depicted in a confusion matrix.

- When a positive data point is correctly predicted by the model, this is known as a true positive (TP).
- When a negative data point is correctly predicted by the model, this is known as a true negative (TN).
- When a positive data item is mispredicted by the model, it results in false positives (FP).
- When a negative data item is mispredicted by the model, it results in false negatives (FN).

The following classification performance evaluation metrics (ACC, CKS, F1S and MCC) have been defined using these four measures (TP, TN, FP and FN).

2.5.3.1 Accuracy (ACC)

A classifier's accuracy reveals its ability to distinguish between classes. In other words, accuracy measures how frequently the model is accurate, and is defined as

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.5.2}$$

Accuracy is not a particularly reliable criterion to assess a classifier's performance for an imbalanced class [106].

2.5.3.2 F1 Score (FS)

F1 score is defined as the harmonic mean of precision and recall [8]. That is,

$$FS = \frac{2}{\frac{1}{precision} + \frac{1}{recall}} \tag{2.5.3}$$

Precision may be thought of as a measure of quality, i.e., the correctness of positive predictions. Precision is also referred to as positive predictive value (PPV), and is defined as

$$precision = \frac{TP}{TP + FP} \tag{2.5.4}$$

Recall attempts to determine if the model can discover all the instances of the positive class, i.e., recall being a measure of quantity. Recall is also known as sensitivity or true positive rate (TPR).

$$recall = \frac{TP}{TP + FN} \tag{2.5.5}$$

F1 score combines the traits of accuracy and recall, making it a stronger metric when the class distribution is uneven.

2.5.3.3 Cohen-Kappa Score (CKS)

A statistical measure of inter-rater agreement is defined by the Cohen-Kappa score [13, 99, 104]. A zero or lower values indicate no agreement, or random labeling, and a higher positive value indicates a good agreement. Cohen-Kappa score is computed as

$$CKS = \frac{p_0 - p_e}{1 - p_e} \tag{2.5.6}$$

where p_0 (the observed agreement ratio) is the empirical probability of agreement on the label assigned to any sample. In essence, p_0 is defined by equation (2.5.2) and is nothing but the accuracy measure. The expected agreement when both annotators assign labels at random is denoted by p_e. p_e is estimated using a per-annotator empirical prior over the class labels [2]. p_e is calculated as

$$p_e = \frac{(TN + FP)(TN + FN) + (FN + TP)(FP + TP)}{(TP + TN + FP + FN)^2} \tag{2.5.7}$$

CKS is sensitive to imbalanced data [106], and is widely used for measuring the performance of a classifier dealing with binary as well as multi-class problems [4, 32].

2.5.3.4 Matthews Correlation Coefficient (MCC)

The Matthews Correlation Coefficient computes the agreement between the predicted and actual classes, accounting for both true and false positives and negatives, to evaluate the quality of binary and multiclass classifications. MCC is viewed as an equal measure as it accounts for true/false positives/negatives [54]. Better agreement is implied by a higher MCC score, indicating that the model is also able to maintain the original dataset's class attributes in the altered dataset as well. MCC is sensitive to data imbalances [106]. MCC is calculated as

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{2.5.8}$$

Mutual Information score (MI)

The similarity between two labels of the same data is measured by Mutual Information. The Mutual Information between clusterings U and V is given by

$$MI(U,V) = \sum_{i=1}^{|U|}\sum_{j=1}^{|V|}\frac{|U_i \cap V_j|}{N} log \frac{N|U_i \cap V_j|}{|U_i||V_j|} \tag{2.5.9}$$

where the number of samples in cluster U_i is denoted by $|U_i|$, and the number of samples in cluster V_j by $|V_j|$. A permutation of the class or cluster label values has no effect on the score value, indicating that this measure is independent of the absolute values of the labels. Additionally, this metric is symmetric, meaning that substituting V (i.e., pred_label) for U (i.e., true_label) will provide the same score value. Thus, without taking permutations into account, Mutual Information is a function that assesses how well the two assignments agree. This metric is available in two different normalised versions: Normalized Mutual Information (NMI) and Adjusted Mutual Information (AMI) [115, 114, 80].

2.5.3.5 Normalized Mutual Information score (NMI)

NMI is a normalization of the MI score to scale the results between 0 (no mutual information) and 1 (perfect correlation). This function uses a generalised mean of $H(U)$ and $H(V)$ to normalise mutual information. NMI is defined as

$$NMI(U,V) = \frac{MI(U,V)}{mean(H(U),H(V))} \tag{2.5.10}$$

where

$$H(U) = -\sum_{i=1}^{|U|}\frac{|U_i|}{N} log \frac{|U_i|}{N} \tag{2.5.11}$$

and

$$H(V) = -\sum_{i=1}^{|V|}\frac{|V_i|}{N} log \frac{|V_i|}{N} \tag{2.5.12}$$

This metric is independent of the absolute values of the labels, i.e., a permutation of the class or cluster label values has no effect on the score value in any way. This measure is not adjusted for chance i.e., the effect of result agreement solely due to chance is not corrected (i.e. corrected the effect of result agreement solely due to chance).

2.5.3.6 Adjusted Mutual Information score (AMI)

AMI is an adjustment of the MI score to account for chance. It accounts for the fact that the MI is generally higher for two clusterings with a larger number of clusters, regardless of whether there is actually more information shared. For two clustering U and V, the AMI is given by

$$AMI(U,V) = \frac{MI(U,V) - E(MI(U,V))}{avg(H(U),H(V)) - E(MI(U,V))} \tag{2.5.13}$$

where, $H(U)$ and $H(V)$ are defined above, and $E(MI(U,V))$ is the expected mutual information between two random clustering. A permutation of the class or cluster label values has no effect on the score value, indicating that this measure is independent of the absolute values of the labels.

2.5.3.7 Adjusted Rand Index (ARI)

By considering all sample pairs, and counting pairs that are allocated in the same or different clusters in the predicted and true clustering, the Rand Index (RI) calculates a similarity measure between two clusterings [48]. RI is calculated as

$$RI = \frac{number\ of\ agreeing\ pairs}{total\ number\ of\ pairs} \tag{2.5.14}$$

Finding a value for random labelling that is close to 0.0 is not guaranteed by the Rand index. Such a baseline is provided by the Adjusted Rand index, which accounts for randomness. RI score is "adjusted for chance" into the ARI score [102, 12] as defined below

$$ARI = \frac{RI - E(RI)}{max(RI) - E(RI)} \tag{2.5.15}$$

2.5.3.8 Jaccard Index (JI)

The Jaccard Index (JI), also known as the Jaccard similarity coefficient, is a metric used to compare the similarity of the set of predicted labels for a sample to the corresponding set of true labels. JI is defined as

$$JI = \frac{\mathbf{Y} \cap \hat{\mathbf{Y}}}{\mathbf{Y} \cup \hat{\mathbf{Y}}} \tag{2.5.16}$$

where $\mathbf{Y}$ is the set of ground truth labels and $\hat{\mathbf{Y}}$ represents the set of predicted labels.

2.6 Conclusions

In this chapter, we have briefly described different datasets used to justify the efficacy of the proposed models. Experimental setup and experimental procedures have also been described in detail. This chapter has also covered different performance measures used throughout the scope of this thesis to establish the proposed models. The next chapter will describe our first contributory work of the thesis, n^2MFn^2, a shallow neural network architecture developed with the aim of dimension reduction.

Chapter 3

Non-negative Matrix Factorization Neural Network (n^2MFn^2)

3.1 Introduction

In the previous chapter, we have presented an overview of various datasets used to support the effectiveness of the models developed in, along with comparisons, in the thesis. Additionally, a thorough explanation of the experimental setup and procedures has also been delineated.

Nowadays, the curse of dimensionality is an issue brought on by the explosive growth of data. In order to analyse the ever-increasing volume of data, dimension reduction is necessary for lowering complexity in terms of time and space. The majority of dimensionality reduction strategies have the drawback of producing feature vectors with negative components, which reduces the applicability of the same. By enforcing non-negativity criteria, Non-negative Matrix Factorization [65, 66], a conventional dimension reduction technique solves this problem and improves the interpretability of the outcome. On the other hand, neural networks are capable of learning from examples without human intervention. Complex nonlinear interactions between dependent and independent variables can also be implicitly detected by neural networks [113]. Our objective is to fuse the advantages of traditional machine learning models with those of neural networks for NMF. With this objective, initially, in this chapter, we

have designed a shallow neural network architecture n^2MFn^2 for dimension reduction [24, 25]. We have used a neural network based model to manipulate the ubiquity of nonnegative input data to generate part-based, sparse and meaningful representations. A modification of the He initialization technique has been proposed to initialize the weights while maintaining the non-negative criterion of the model. A necessary modification of the ReLU activation function has been made to satisfy the architectural constraints. Regularization has been used in the objective function of the model to produce an optimal approximation of the input matrix. To demonstrate the efficacy of the proposed model, we have analyzed and compared the results with six well known dimension reduction methods on five popular datasets. We have also discussed the computational complexity and convergence analysis of the model.

The rest of the chapter is organised as follows. Section 3.2 describes the motivation behind the architecture and learning of n^2MFn^2. The detailed design and derivation of respective learning rules have been presented in Section 3.3. Subsequently, Section 3.4 depicts the results following the experimentation procedure described in Chapter 2, with an adequate analysis. The convergence analysis of n^2MFn^2 and the analysis of computational complexity have been presented in Sections 3.5 and 3.6. Finally, Section 3.7 brings the chapter to a conclusion.

3.2 Motivation behind Architecture and Learning

We aim to fuse the advantages of the traditional iterative NMF technique with that of the neural network architecture. The traditional NMF technique uses a block coordinate descent scheme to update the factors of the input matrix. In this chapter, we develop a neural network model, called Non-negative Matrix Factorization Neural Network (n^2MFn^2), for the task of NMF, where this approach for updating the factors has been overcomed by modifying both the factors simultaneously.

The architecture of the model primarily includes a single hidden layer designed in such a manner that it acts as the slender layer of the system. The activity of the neural network model is divided into two phases, namely, deconstruction and reconstruction.

Hence, the architecture can be referred to as the Single Deconstruction Single Reconstruction (SDSR) framework. The slender layer output and the weight matrix connecting the slender and the output layers of the model are designed to be the two non-negative factors of the model. As the weight matrix has to follow the non-negativity criteria, to maintain the consistency of the model with respect to non-negativity, a modification of the popular He weight initialization technique [41], called Modified He initialization technique (mHe), has been derived.

Modified He initialization technique (mHe): First, the elements of the weight matrix are initialized by the original He initialization technique and hence, the initial weight values lie in $[-\varepsilon_1, +\varepsilon_2]$. Then the weight values are transformed in such a way that the negative values disappear and the new interval of values becomes $[0, (\varepsilon_1 + \varepsilon_2)]$. With this transformation, the standard deviation remains unaltered, but the previous mean 0 is changed to $(0 + \varepsilon_1)$, i.e., ε_1. Now, the transformed values of the elements of the weight matrix are multiplied by ε_1, forcing them to be very small positive fractions, as ε_1 is a fraction itself. Hence, in this case, the initial values of the elements of the weight matrix lie in $[0, (\varepsilon_1 + \varepsilon_2)\varepsilon_1]$.

The objective function has been designed to reduce over-fitting using L1 regularization/Lasso regularization. The novel regularizer ensures the best possible approximation of the input data matrix. Moreover, the regularizing parameter has been chosen in such a way that it has a controlled effect on the regularizer when n^2MFn^2 tries to regenerate the input. The learning rules of the architecture have been derived maintaining the constraints of the model. The update rules tune each element of the weight matrix individually to satisfy the constraint. In the gradient descent approach, the rule for the updation of the weight values is the same but the amount of change in each weight is dynamically decided. In n^2MFn^2, the rule of deciding the value of the adaptive learning rate is fixed but it has been designed in a way that it guides each weight value individually to satisfy the non-negativity property. Thus, the weight-specific learning rate is basically fine-tuning the weight update process satisfying the non-negativity criteria. Choosing a fixed learning rate *a priori*, satisfying the non-negativity criteria, is difficult, and hence, an adaptive learning rate is preferred. Furthermore, a good adaptive algorithm converges significantly quicker than basic back-propagation with a randomly chosen fixed learning rate [87].

The Rectified Linear Unit (ReLU) activation function replaces all values less than zero with zero. During the realization of the concept, this would lead to the problem of division by zero. Hence, the ReLU activation function has been modified to meet the non-negativity criteria of the architecture. The ReLU activation function does not activate all neurons at the same time. Thus, the sparsity of the network is maintained and it prevents all the neurons in a layer from synchronously optimizing their weights.

3.3 n^2MFn^2

In this section, we develop an artificial neural network model, called Non-negative Matrix Factorization Neural Network (n^2MFn^2), for the task of non-negative matrix factorization towards dimensionality reduction. We now describe the architecture of the model followed by its learning algorithm.

3.3.1 Architecture

The architecture of n^2MFn^2, as depicted in Figure 3.1, consists of an input layer, a single hidden layer and an output layer. Following the procedures outlined in Chapter 2 Section 2.3, we process the given data matrix $\mathbf{U} = [u_{pi}]_{m \times n'}$ and get a matrix $\mathbf{X} = [x_{pi}]_{m \times n}$ with each element being non-negative. At this point, the model uses each row of $\mathbf{X}$ as input. Thus, the input layer receives the input signal from a dataset consisting of m samples; each of which is described by n features. The hidden layer, comprising r nodes, is designed in such a manner that it acts as the slender layer of the system, extracting $r < n'$ features. The output layer tries to get back the original data from the extracted features.

The model is designed in such a way that the mapping from the input layer to the hidden layer deconstructs the data to its latent representation. Hence, we refer to this phase as the deconstruction phase. The model then tries to reconstruct the input from the latent representation, i.e., the output of the slender layer. In other words, the mapping from the hidden layer to the output layer reconstructs the input data from its latent representation. We call this phase of the model as reconstruction phase. Since there is only a single hidden layer in n^2MFn^2, the architecture is termed as a Single Deconstruction Single Reconstruction (SDSR) neural network framework.

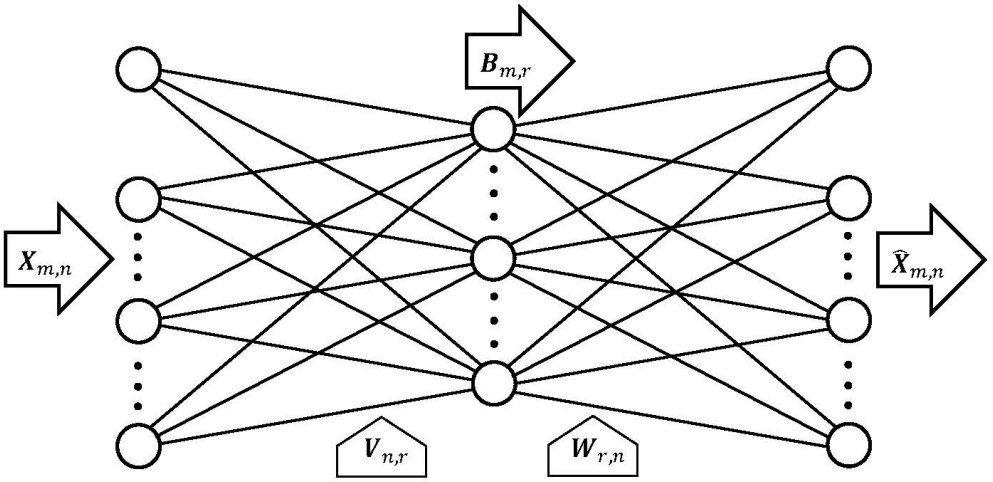

FIGURE 3.1: The proposed model: n^2MFn^2.

Two types of activation functions have been used in n^2MFn^2. Firstly, the activation of input nodes is an identity function. That is, the output of i^{th} input node is the same as its input. In other words, the input layer activation function passes on the input to the next layer without any processing. Secondly, a modified version of the Rectified Linear Unit (ReLU) activation function ψ has been used in the hidden and output layer nodes of the model to satisfy the non-negativity constraint. Mathematically, ψ is defined as,

$$\psi(x) = \begin{cases} x, & \text{if } x > 0 \\ \epsilon, & \text{otherwise} \end{cases} \tag{3.3.1}$$

where $\epsilon > 0$ is a small number specified by the user. Here we have considered $\epsilon = 0.001$ to avoid the problem of division by zero in the course of execution of the algorithm. Thus, the output of the hidden layer for all the samples can be written in matrix form $\mathbf{B} = [b_{pl}]_{m\times r}$, and is given by

$$\mathbf{B} = \psi(\mathbf{Y}) \tag{3.3.2}$$

The term $\psi(\mathbf{Y})$ is a matrix of order $m \times r$, for which each element is obtained by applying the activation function ψ on the corresponding element of $\mathbf{Y}$. Here $\mathbf{Y} = [y_{pl}]_{m\times r}$ is defined as

$$\mathbf{Y} = \mathbf{XV} \tag{3.3.3}$$

where $\mathbf{V} = [v_{il}]_{n\times r}$ is the weight matrix between the input and the hidden layers. This concludes the deconstruction phase of the model and hence marks the beginning of the

reconstruction phase. The output layer generates $\widehat{\mathbf{X}} = [\hat{x}_{pj}]_{m\times n}$, and is defined as

$$\widehat{\mathbf{X}} = \psi(\mathbf{Z}) \tag{3.3.4}$$

where $\mathbf{Z} = [z_{pj}]_{m\times n}$ is computed as

$$\mathbf{Z} = \mathbf{BW} \tag{3.3.5}$$

Here, $\mathbf{W} = [w_{lj}]_{r\times n}$ is the weight matrix between the hidden and the output layers. There are no restrictions on the weight values in $\mathbf{V}$, while $\mathbf{W}$ follows the non-negativity constraint. That is, each element of the weight matrix $\mathbf{W}$ has to be non-negative. The non-negativity of $\mathbf{W}$ is achieved by estimating the hyper-parameter of the respective learning rule as described below. The hidden layer output $\mathbf{B}$ and weight matrix $\mathbf{W}$ are the two non-negative factors of the regenerated input matrix $\widehat{\mathbf{X}}$.

3.3.2 Learning

Learning in an artificial neural network is a process of estimating weight parameters, based on the input and output of the network, to meet certain objectives. The objective of n^{2}MFn^2 is to minimize $\|\mathbf{X} - \widehat{\mathbf{X}}\|_F$ with respect to $\mathbf{V}$ and $\mathbf{W}$ subject to $\mathbf{VW} = \mathbf{I}$, where $\mathbf{I} = [\delta_{ij}]_{n\times n}$ is the Identity matrix of order n. Thus, the cost function Φ is defined as

$$\Phi = \frac{1}{mn}\sum_{p=1}^{m}\sum_{j=1}^{n}\frac{1}{2}(x_{pj} - \widehat{x}_{pj})^2 + \frac{\lambda}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{1}{2}(\sum_{l=1}^{r} v_{il}w_{lj} - \delta_{ij})^2 \tag{3.3.6}$$

In the above equation, the first term on the right-hand side has been used to measure the average squared reconstruction error between the given input $\mathbf{X}$ and the reconstructed output $\hat{\mathbf{X}}$. The second term on the right-hand side is the regularization term and λ is the Lagrange's multiplier (regularizing parameter). The regularizer of the form has been designed to satisfy the condition $\mathbf{VW} = \mathbf{I}$, given in the objective function of the model and to assist n^{2}MFn^2 to find an $\hat{\mathbf{X}}$ as close as possible to $\mathbf{X}$, i.e., $\hat{\mathbf{X}} \simeq \mathbf{X}$. The measure of reconstruction error has been used as guidance to the learning of the model so that the model can produce a meaningful representation of the input in the lower dimensional space. From equations (3.3.2) to (3.3.5), ignoring the activation function ψ, we get $\hat{\mathbf{X}} = \mathbf{BW}$, where $\mathbf{B} = \mathbf{XV}$. Hence, we can write $\hat{\mathbf{X}} = \mathbf{XVW}$. Now,

we try to regulate **VW** in such a manner, that the product tends to **I**, and thus, n^2MFn^2 will be able to produce the best possible approximation of **X**, i.e., **XI** = **X**. The term δ_{ij} (Kronecker delta) is defined as

$$\delta_{ij} = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{otherwise} \end{cases} \tag{3.3.7}$$

Thus, the problem boils down to learning (estimating) **V** and **W**, such that Φ becomes the minimum. This is achieved by the following iterative procedure

$$v_{il}(t+1) = v_{il}(t) + \Delta v_{il}(t), i = 1, 2, ..., n \text{ and } l = 1, 2, ..., r \tag{3.3.8}$$

and

$$w_{lj}(t+1) = w_{lj}(t) + \Delta w_{lj}(t), l = 1, 2, ..., r \text{ and } j = 1, 2, ..., n \tag{3.3.9}$$

t being iteration count. In order to minimize Φ with respect to **V** and **W**, we adopt gradient descent technique, i.e.,

$$\Delta v_{il}(t) = -\eta_{v_{il}(t)} \frac{\partial \Phi}{\delta v_{il}(t)} \tag{3.3.10}$$

and

$$\Delta w_{lj}(t) = -\eta_{w_{lj}(t)} \frac{\partial \Phi}{\delta w_{lj}(t)} \tag{3.3.11}$$

Here $\eta_{v_{il}(t)}$ and $\eta_{w_{lj}(t)}$ are the learning rates corresponding to $v_{il}(t)$ and $w_{lj}(t)$. It is to be noted that $\eta_{v_{il}(t)}$, $\eta_{v_{il}}(t)$ and $\eta_{v_{il}}$ are synonymous and will be used interchangeably for ease of depiction. Similarly $\eta_{w_{lj}(t)}$, $\eta_{w_{lj}}(t)$ and $\eta_{w_{lj}}$ are synonymous. Thus, the weight matrices **V** and **W** are learned (estimated) iteratively by

$$\mathbf{V}(t+1) = \mathbf{V}(t) - \eta_{\mathbf{V}(t)} \circ \nabla_{\mathbf{V}(t)} \Phi \tag{3.3.12}$$

and

$$\mathbf{W}(t+1) = \mathbf{W}(t) - \eta_{\mathbf{W}(t)} \circ \nabla_{\mathbf{W}(t)} \Phi \tag{3.3.13}$$

Here, $\circ$ denotes the Hadamard product and the matrices $\eta_{\mathbf{V}(t)}$ and $\eta_{\mathbf{W}(t)}$ are two hyper-parameters of the model, called learning rates corresponding to **V** and **W**. The terms $\nabla_{\mathbf{V}(t)}$ and $\nabla_{\mathbf{W}(t)}$ are the gradient operators with respect to the weight matrices **V** and

$\mathbf{W}$ respectively. It is to be noted that $\nabla_{\mathbf{V}(t)}$, $\nabla_{\mathbf{V}}(t)$ and $\nabla_{\mathbf{V}}$ are synonymous and will be used interchangeably for ease of depiction. Similarly, $\nabla_{\mathbf{W}(t)}$, $\nabla_{\mathbf{W}}(t)$ and $\nabla_{\mathbf{W}}$ are synonymous.

Now, we calculate the derivatives of Φ with respect to v_{il} and w_{lj} i.e., $\nabla_{\mathbf{V}}\Phi = [\frac{\partial\Phi}{\delta v_{il}(t)}]_{n\times r}$ and $\nabla_{\mathbf{W}}\Phi = [\frac{\partial\Phi}{\delta w_{lj}(t)}]_{r\times n}$. Let us drop t (iteration count) for simplicity. Thus,

$$
\begin{aligned}
\frac{\partial\Phi}{\delta v_{il}} &= \frac{\partial}{\delta v_{il}}\Big(\frac{1}{mn}\sum_{p=1}^{m}\sum_{j=1}^{n}\frac{1}{2}(x_{pj}-\widehat{x}_{pj})^2 + \frac{\lambda}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{1}{2}\Big(\sum_{l'=1}^{r}v_{il'}w_{l'j}-\delta_{ij}\Big)^2\Big)\\
&= \frac{1}{2mn}\sum_{p=1}^{m}\sum_{j=1}^{n}\frac{\partial}{\delta v_{il}}(x_{pj}-\widehat{x}_{pj})^2 + \frac{\lambda}{2n^2}\sum_{j=1}^{n}\frac{\partial}{\delta v_{il}}\Big(\sum_{l'=1}^{r}v_{il'}w_{l'j}-\delta_{ij}\Big)^2\\
&= -\frac{1}{mn}\sum_{p=1}^{m}\sum_{j=1}^{n}(x_{pj}-\widehat{x}_{pj})\frac{\partial\widehat{x}_{pj}}{\delta v_{il}} + \\
&\qquad\qquad \frac{\lambda}{n^2}\sum_{j=1}^{n}\Big(\sum_{l'=1}^{r}v_{il'}w_{l'j}-\delta_{ij}\Big)\frac{\partial}{\delta v_{il}}\sum_{l'=1}^{r}v_{il'}w_{l'j}\\
&= -\frac{1}{mn}\sum_{p=1}^{m}\sum_{j=1}^{n}(x_{pj}-\widehat{x}_{pj})\frac{\partial\widehat{x}_{pj}}{\delta z_{pj}}\frac{\partial z_{pj}}{\delta v_{il}} + \frac{\lambda}{n^2}\sum_{j=1}^{n}\Big(\sum_{l'=1}^{r}v_{il'}w_{l'j}-\delta_{ij}\Big)w_{lj}\\
&= -\frac{1}{mn}\sum_{p=1}^{m}\sum_{j=1}^{n}(x_{pj}-\widehat{x}_{pj})\frac{\partial z_{pj}}{\delta v_{il}} + \frac{\lambda}{n^2}\sum_{j=1}^{n}\Big(\sum_{l'=1}^{r}v_{il'}w_{l'j}-\delta_{ij}\Big)w_{lj} \qquad (3.3.14)\\
&= -\frac{1}{mn}\sum_{p=1}^{m}\sum_{j=1}^{n}(x_{pj}-\widehat{x}_{pj})\frac{\partial z_{pj}}{\delta b_{pl}}\frac{\partial b_{pl}}{\delta v_{il}} + \frac{\lambda}{n^2}\sum_{j=1}^{n}\Big(\sum_{l'=1}^{r}v_{il'}w_{l'j}-\delta_{ij}\Big)w_{lj}\\
&= -\frac{1}{mn}\sum_{p=1}^{m}\sum_{j=1}^{n}(x_{pj}-\widehat{x}_{pj})w_{lj}\frac{\partial b_{pl}}{\delta v_{il}} + \frac{\lambda}{n^2}\sum_{j=1}^{n}\Big(\sum_{l'=1}^{r}v_{il'}w_{l'j}-\delta_{ij}\Big)w_{lj}\\
&= -\frac{1}{mn}\sum_{p=1}^{m}\sum_{j=1}^{n}(x_{pj}-\widehat{x}_{pj})w_{lj}\frac{\partial b_{pl}}{\delta y_{pl}}\frac{\partial y_{pl}}{\delta v_{il}} + \frac{\lambda}{n^2}\sum_{j=1}^{n}\Big(\sum_{l'=1}^{r}v_{il'}w_{l'j}-\delta_{ij}\Big)w_{lj}\\
&= -\frac{1}{mn}\sum_{p=1}^{m}\sum_{j=1}^{n}(x_{pj}-\widehat{x}_{pj})w_{lj}\frac{\partial y_{pl}}{\delta v_{il}} + \frac{\lambda}{n^2}\sum_{j=1}^{n}\Big(\sum_{l'=1}^{r}v_{il'}w_{l'j}-\delta_{ij}\Big)w_{lj}\\
&= -\frac{1}{mn}\sum_{p=1}^{m}\sum_{j=1}^{n}(x_{pj}-\widehat{x}_{pj})w_{lj}x_{pi} + \frac{\lambda}{n^2}\sum_{j=1}^{n}\Big(\sum_{l'=1}^{r}v_{il'}w_{l'j}-\delta_{ij}\Big)w_{lj}
\end{aligned}
$$

and

$$\begin{aligned}
\frac{\partial \Phi}{\partial w_{lj}} &= \frac{\partial}{\partial w_{lj}}\left(\frac{1}{mn}\sum_{p=1}^{m}\sum_{j=1}^{n}\frac{1}{2}(x_{pj}-\widehat{x}_{pj})^2+\frac{\lambda}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{1}{2}(\sum_{l'=1}^{r}v_{il'}w_{l'j}-\delta_{ij})^2\right)\\
&= \frac{1}{2mn}\sum_{p=1}^{m}\frac{\partial}{\delta w_{lj}}(x_{pj}-\widehat{x}_{pj})^2+\frac{\lambda}{2n^2}\sum_{i=1}^{n}\frac{\partial}{\delta w_{lj}}(\sum_{l'=1}^{r}v_{il'}w_{l'j}-\delta_{ij})^2\\
&= -\frac{1}{mn}\sum_{p=1}^{m}(x_{pj}-\widehat{x}_{pj})\frac{\partial \widehat{x}_{pj}}{\delta w_{lj}}+\\
&\qquad \frac{\lambda}{n^2}\sum_{i=1}^{n}(\sum_{l'=1}^{r}v_{il'}w_{l'j}-\delta_{ij})\frac{\partial}{\delta w_{lj}}\sum_{l'=1}^{r}v_{il'}w_{l'j} \qquad (3.3.15)\\
&= -\frac{1}{mn}\sum_{p=1}^{m}(x_{pj}-\widehat{x}_{pj})\frac{\partial \widehat{x}_{pj}}{\delta z_{pj}}\frac{\partial z_{pj}}{\delta w_{lj}}+\frac{\lambda}{n^2}\sum_{i=1}^{n}(\sum_{l'=1}^{r}v_{il'}w_{l'j}-\delta_{ij})v_{il}\\
&= -\frac{1}{mn}\sum_{p=1}^{m}(x_{pj}-\widehat{x}_{pj})\frac{\partial z_{pj}}{\delta w_{lj}}+\frac{\lambda}{n^2}\sum_{i=1}^{n}(\sum_{l'=1}^{r}v_{il'}w_{l'j}-\delta_{ij})v_{il}\\
&= -\frac{1}{mn}\sum_{p=1}^{m}(x_{pj}-\widehat{x}_{pj})b_{pl}+\frac{\lambda}{n^2}\sum_{i=1}^{n}(\sum_{l'=1}^{r}v_{il'}w_{l'j}-\delta_{ij})v_{il}
\end{aligned}$$

Using equations 3.3.14 and 3.3.15, $\nabla_{\mathbf{V}}$ and $\nabla_{\mathbf{W}}$ can be written as

$$\nabla_{\mathbf{V}}\Phi = -\frac{1}{mn}(((\mathbf{X}-\widehat{\mathbf{X}})\mathbf{W}^T)^T\mathbf{X})^T+\frac{\lambda}{n^2}((\mathbf{VW}-\mathbf{I})\mathbf{W}^T) \qquad (3.3.16)$$

and

$$\nabla_{\mathbf{W}}\Phi = -\frac{1}{mn}(\mathbf{B}^T(\mathbf{X}-\widehat{\mathbf{X}}))+\frac{\lambda}{n^2}(\mathbf{V}^T(\mathbf{VW}-\mathbf{I})) \qquad (3.3.17)$$

As stated before, the activation function ψ is used only to replace the non-positive values of the argument by ϵ. Otherwise, the value of the function ψ is the same as its argument. For simplicity, ignoring ψ in equations 3.3.4, we can rewrite equation 3.3.17 as

$$\nabla_{\mathbf{W}}\Phi = -\frac{1}{mn}(\mathbf{B}^T(\mathbf{X}-\mathbf{BW}))+\frac{\lambda}{n^2}(\mathbf{V}^T(\mathbf{VW}-\mathbf{I})) \qquad (3.3.18)$$

Using equations 3.3.16 and 3.3.18, the learning rules, defined in equations 3.3.12 and 3.3.13, become

$$\begin{aligned}
\mathbf{V}(t+1) = \mathbf{V}(t)-\eta_{\mathbf{V}(t)}\circ\Bigg(&-\frac{1}{mn}(((\mathbf{X}-\widehat{\mathbf{X}})\mathbf{W}(t)^T)^T\mathbf{X})^T+\\
&\frac{\lambda}{n^2}((\mathbf{V}(t)\mathbf{W}(t)-\mathbf{I})\mathbf{W}(t)^T)\Bigg) \qquad (3.3.19)
\end{aligned}$$

and

$$\mathbf{W}(t+1) = \mathbf{W}(t) - \eta_{\mathbf{W}(t)} \circ \left(-\frac{1}{mn}(\mathbf{B}^T(\mathbf{X} - \mathbf{B}\mathbf{W}(t))) + \frac{\lambda}{n^2}(\mathbf{V}(t)^T(\mathbf{V}(t)\mathbf{W}(t) - \mathbf{I})) \right) \quad (3.3.20)$$

As defined above, the elements in $\mathbf{V}$ are unrestricted, whereas those in $\mathbf{W}$ have to be non-negative. In order to meet this criterion, we choose the value of $\eta_{\mathbf{W}}$ in such a manner that the negative terms arising from the computation in equation 3.3.20 get dismissed. Thus we choose $\eta_{\mathbf{W}}$ as

$$\eta_{\mathbf{W}} = (mn\mathbf{W}) \oslash (\mathbf{B}^T\mathbf{B}\mathbf{W}) \quad (3.3.21)$$

Here, $\oslash$ denotes Hadamard division. Now, using equation 3.3.21, equation 3.3.20 becomes,

$$\mathbf{W}(t+1) = \left((\mathbf{W}(t) \oslash (\mathbf{B}^T\mathbf{B}\mathbf{W}(t))) \circ (\mathbf{B}^T\mathbf{X}) \right) - \frac{m}{n} \left((\mathbf{W}(t) \oslash (\mathbf{B}^T\mathbf{B}\mathbf{W}(t))) \circ \lambda(\mathbf{V}(t)^T(\mathbf{V}(t)\mathbf{W}(t) - \mathbf{I})) \right) \quad (3.3.22)$$

As defined above, the elements in $\mathbf{X}$, $\mathbf{B}$ and $\mathbf{W}$ are all positive. Hence those in the first part on the right hand side of the equation 3.3.22 are positive. The second part of the equation 3.3.22 contains the expression $(\mathbf{V}\mathbf{W} - \mathbf{I})$, which will gradually vanish because of the constraint defined in the objective function of the model. That is, the value of $\mathbf{V}\mathbf{W}$ will tend towards $\mathbf{I}$ over the iterations. Even after the above, during back-propagation of the neural network, some of the elements of the weight matrix $\mathbf{W}$ may become negative. Those negative values are replaced with 0.001 to maintain the consistency of the model with respect to the non-negativity criterion of the weight matrix $\mathbf{W}$.

λ is the regularizing parameter used in the objective function of n^2MFn^2. The value of λ decides the amount by which we want to penalize the model's flexibility. We have used ridge regression in the model and it is used to overcome the overfitting of the model. The ridge regression technique works by preventing coefficient values from becoming too high. For $\lambda = 0$, the effect of the penalty term goes away and as $\lambda \to \infty$, the effect of the shrinkage penalty grows. Thus, we have chosen $\lambda = 0.1$ to have a

controlled effect on the regularizer when n^{2}MFn^2 tries to regenerate the input. For simplicity, all the elements of the hyper-parameter matrix $\eta_{\mathbf{V}}$ have been set to 0.1, i.e., $\eta_{\mathbf{V}} = [0.1]_{n \times r}$. Hence, equations 3.3.19 and 3.3.22 are the update rules for $\mathbf{V}$ and $\mathbf{W}$ respectively.

3.4 Experimental Results, Analysis and Discussion

n^{2}MFn^2 model has been evaluated using five real-world datasets described in Chapter 2. The datasets can be classified into two groups based on their dimensions. ONP, PDC and SP datasets have more samples than features ($m > n'$), whereas GLRC and MovieLens datasets consist of fewer samples than features ($m < n'$).

Here the performance of n^{2}MFn^2 has been presented and justified in two ways. Firstly, the quality of dimension reduction by n^{2}MFn^2 has been evaluated by comparing its ability to preserve the local structure of data. The need for dimension reduction, i.e., the efficacy of the low rank embedding in contrast to the original data has also been analyzed and established. Secondly, the discriminating ability of the dimensionally reduced dataset is explored for downstream analyses, like classification and clustering. The statistical significance of the results obtained by n^{2}MFn^2 with respect to other dimension reduction techniques has also been studied.

In n^{2}MFn^2 model, the elements of the weight matrix $\mathbf{V}$ have been initialized using He initialization technique [41] and the elements of $\mathbf{W}$ have been initialized using Modified He initialization technique developed here. The number of training epochs is decided dynamically. Training stops on reaching predefined stopping criteria based on the difference in the cost values of two consecutive epochs.

3.4.1 Quantifying the quality of low dimensional embedding

The quality of low dimensional embedding by n^{2}MFn^2 has been investigated in two ways, viz., studying the ability to preserve the local structure of data by using trustworthiness score and by comparing the effectiveness of dimension reduction by classification/cluster performance measures compared with the original data.

3.4.1.1 Local structure preservation

The ability to preserve the local structure of data after dimension reduction by n^2MFn^2 over that of six other dimension reduction approaches has been computed and compared using the trustworthiness score values. The outcome of the same is depicted in the spider/star plot (Figure 3.2).

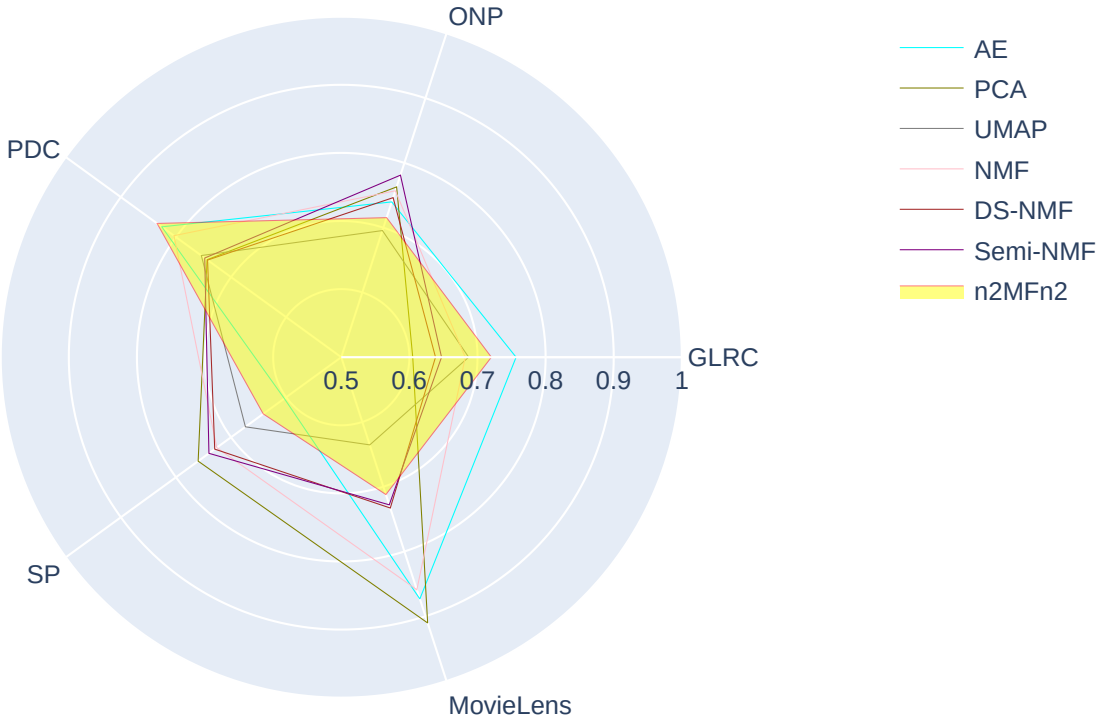

FIGURE 3.2: Trustworthiness scores of seven dimension reduction techniques including n^2MFn^2.

There are five axes corresponding to five datasets. The trustworthiness score of a dimension reduction technique for a particular dataset is a point on that axis. Thus, for a dimension reduction technique, there are five points on five axes corresponding to five datasets. These points can be considered as vertices of a polygon. Thus, in Figure 3.2, there are seven polygons for seven dimension reduction techniques. The area covered by a polygon justifies the efficacy of a dimension reduction method over all the datasets together. The higher the area, the better is the performance of the algorithm. From the depiction, we can note that n^2MFn^2 has beaten other dimension reduction techniques for the PDC dataset, and for GLRC the trustworthiness score of n^2MFn^2 is better than most of the others. The area bounded by the polygon of n^2MFn^2 is shown in a shaded colour in Figure 3.2. We compute the area of the polygon by adding individual trustworthiness scores of the dimension reduction techniques for all five datasets. It can be

TABLE 3.1: Sum of trustworthiness scores of seven dimension reduction techniques including n^2MFn^2 on five datasets.

Dimension reduction techniques	Sum of trustworthiness scores
AE	**4.55550328220533**
PCA	4.38647684863791
UMAP	4.13297187263103
NMF	4.50973409888627
DS-NMF	4.22833659492512
Semi-NMF	4.29147059452664
n^2MFn^2	4.34404888837219

observed from Table 3.1 that the sum of trustworthiness scores of n^2MFn^2 has beaten three out of six dimension reduction techniques compared with.

3.4.1.2 Decision making: Comparison with the original data

The efficacy of dimension reduction by n^2MFn^2 has been judged by performing classification and clustering on low dimensional embedding produced by them as well as on the original data, and then quantifying the performances using different classification and cluster validity metrics. This study demonstrates why the dimension reduction is necessary highlighting the fact that the usability of the data increases with the low rank representation of the same.

Classification

Figures 3.3-3.7 presents the performance of n^2MFn^2 and original data in terms of classification. For the GLRC (Figure 3.3) and PDC (Figure 3.5) datasets, n^2MFn^2 generated low rank embedding has outperformed the original data for all four classifiers in terms of all four metrics. For the ONP dataset, the same count is two out of four for all four classification evaluators (Figure 3.4). For FS, CKS and MCC performance metrics, n^2MFn^2 has performed better than the original dataset for three out of four classification algorithms for the MovieLens dataset, and the same count is two when the evaluator is ACC (Figure 3.7). In the case of the SP dataset, the performance metric of original data is better than the low rank embedding produced by n^2MFn^2 on all occasions (Figure 3.6).

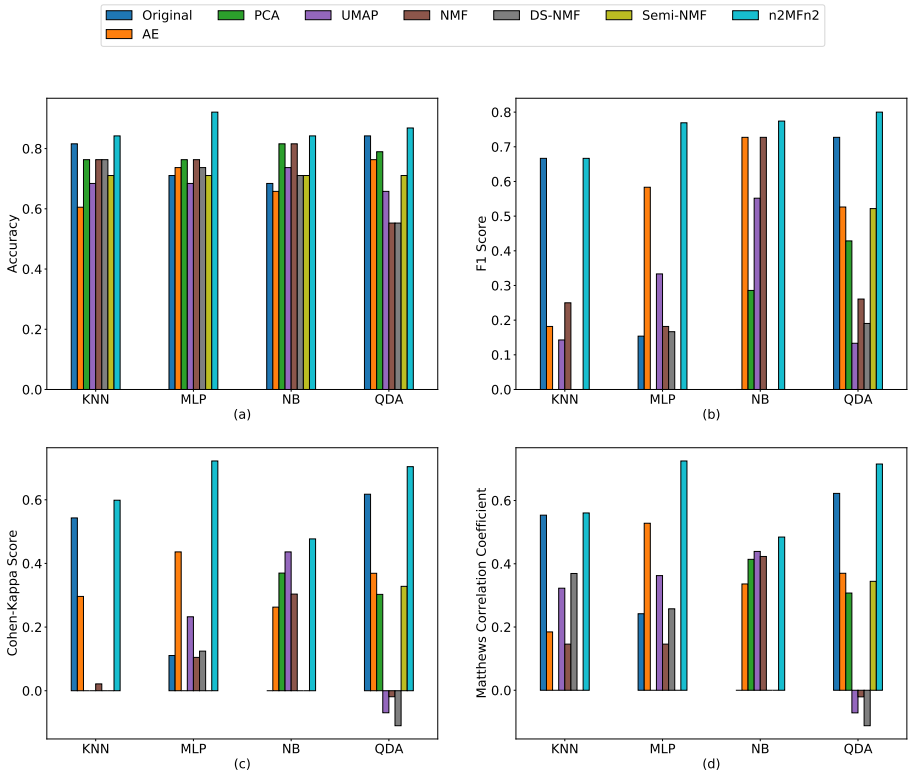

FIGURE 3.3: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the GLRC dataset by n^2MFn^2 and six other dimension reduction techniques along with the original data.

Thus, it is evident that in most of the cases, n^2MFn^2 projected data have performed better than the original data in terms of classification. This justifies the need for dimension reduction along with the ability to produce low rank embedding preserving elemental characteristics of data.

Clustering

The performance comparison of clustering done on the low dimensional embedding produced by n^2MFn^2 and the original data has been illustrated in Figures 3.8-3.12. For the ONP (Figure 3.9), PDC (Figure 3.10) and SP (Figure 3.11) datasets, for all four cluster validity indexes, n^2MFn^2 has performed better than the original data with respect to all four clustering algorithms. For the GLRC (Figure 3.8) dataset, the performance score is three out of four in favour of n^2MFn^2 for all four cluster evaluators. When the cluster validity index is ARI, n^2MFn^2 has performed better than the original data for

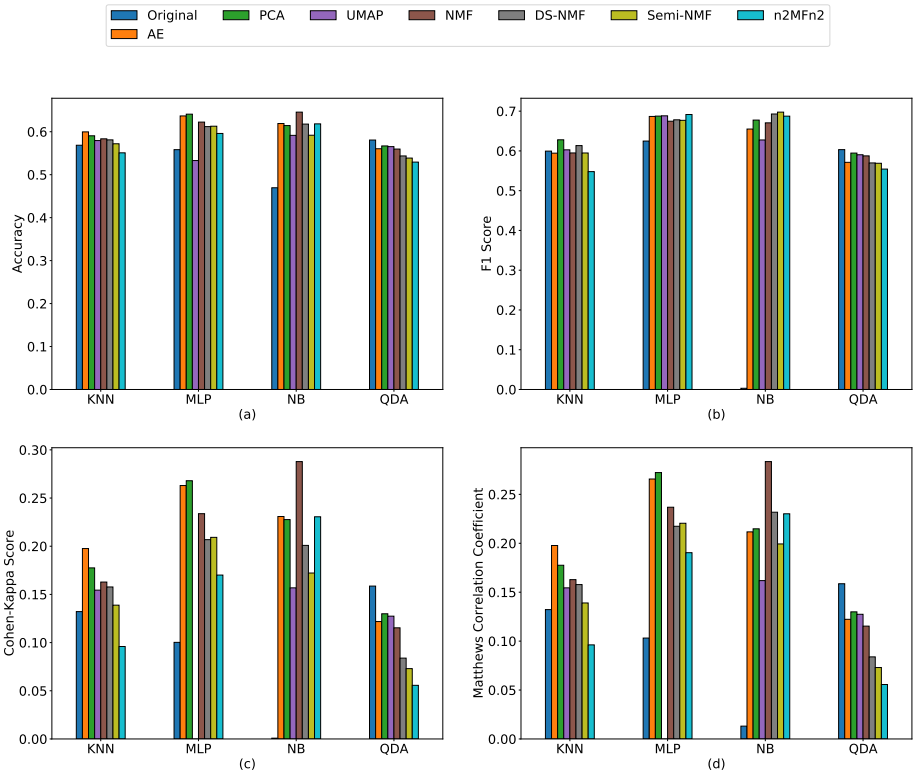

FIGURE 3.4: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the ONP dataset by n^2MFn^2 and six other dimension reduction techniques along with the original data.

three out of four clustering algorithms in the case of the MovieLens dataset, and for other evaluators the same count is four out of four (Figure 3.12).

Hence, it is established in terms of clustering performance that the low rank embedding generated by n^2MFn^2 is much better in preserving the fundamental properties of the original data. Thus the need for dimension reduction is also justified.

3.4.2 Downstream analyses and statistical significance: Comparison with other models

By performing classification and clustering on the low dimensional embedding generated by n^2MFn^2 and also that generated by the other six dimension reduction techniques, the effectiveness of dimension reduction has been assessed. Several metrics measuring classification and cluster performances have been used to quantify the same. Pairwise p-values have also been calculated to support the superiority of n^2MFn^2 over

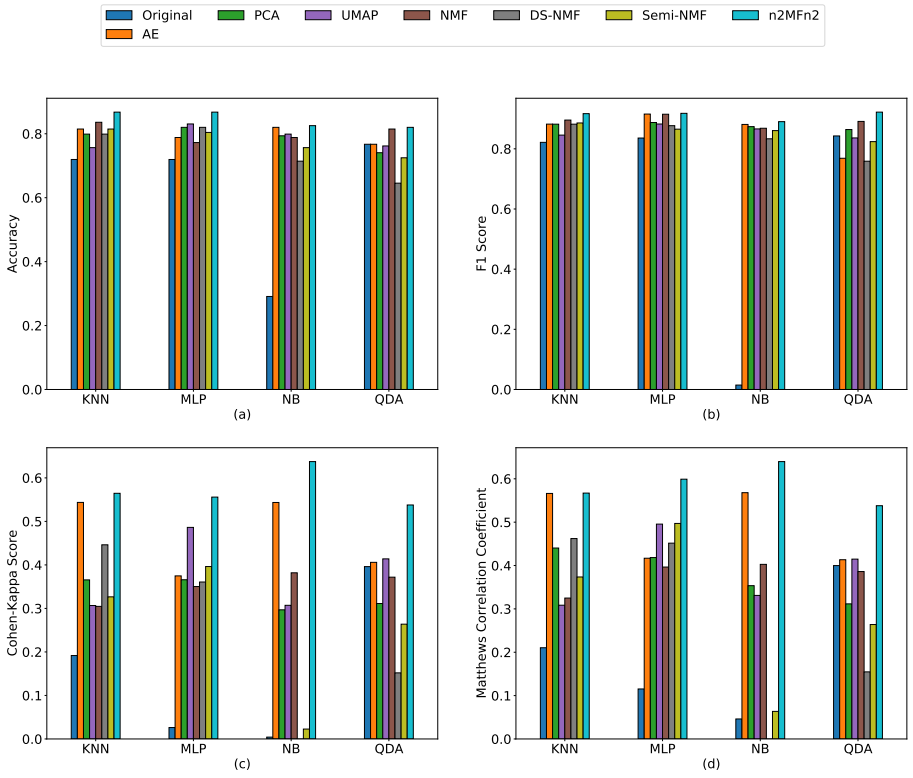

FIGURE 3.5: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the PDC dataset by n^2MFn^2 and six other dimension reduction techniques along with the original data.

other dimension reduction algorithms in terms of producing output from an independent set of observations. A p-value less than a particular threshold justifies the statistical significance of the results. Here, we have taken the threshold value to be 0.05. Thus, for a dataset, a classification/clustering algorithm and a classification/cluster validity index, six p-values have been computed comparing the performance of n^2MFn^2 with that of six other dimension reduction algorithms. As there are four classification/cluster algorithms, there will be a total of $6 \times 4 = 24$ p-values for each classification/cluster validity index against each dataset. This part of the experimentation aims to determine the superiority of dimension reduction by n^2MFn^2 using different types of classification and clustering algorithms.

Classification

While working with the n^2MFn^2 model for classification, the outcome has been depicted by Figures 3.3-3.7. The summary of the count of statistically significant p-values

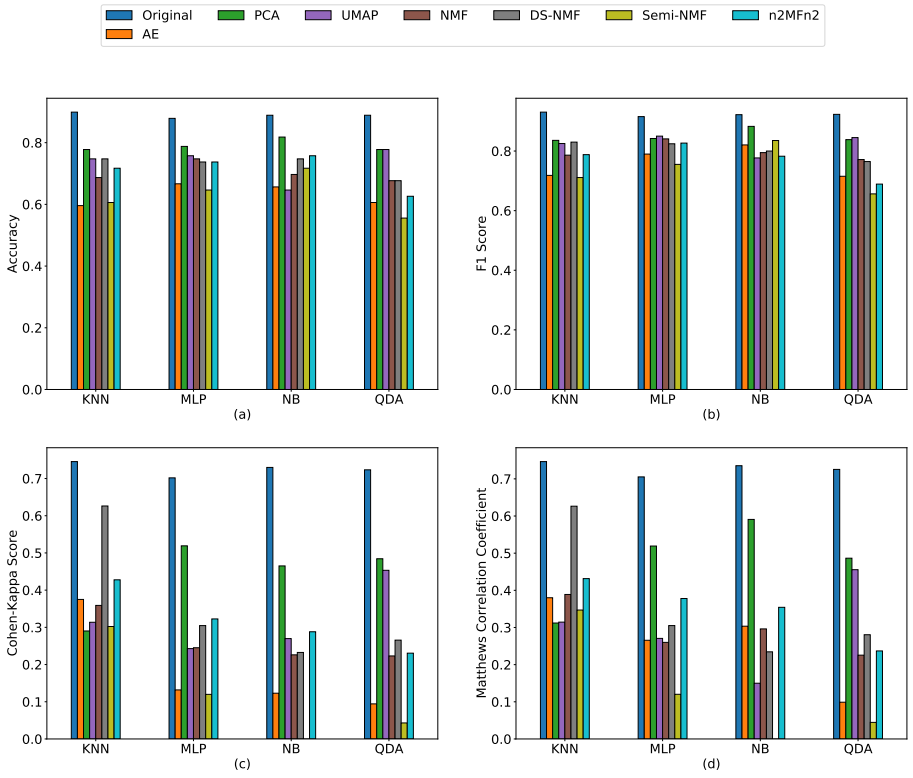

FIGURE 3.6: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the SP dataset by n^2MFn^2 and six other dimension reduction techniques along with the original data.

with respect to n^2MFn^2 has also been presented in Table 3.2.

For four classification techniques, n^2MFn^2 has always achieved the highest accuracy score for GLRC (Figure 3.3(a)), PDC (Figure 3.5(a)) and MovieLens (Figure 3.7(a)) datasets and none for ONP (Figure 3.4(a)) and SP (Figure 3.6(a)) datasets. The same statistics hold when the classification performance metrics are Cohen-Kappa score (Figures 3.3(c), 3.4(c), 3.5(c), 3.6(c) and 3.7(c)) and Matthews Correlation Coefficient score (Figures 3.3(d), 3.4(d), 3.5(d), 3.6(d) and 3.7(d)). When the F1 score is used as the classification performance indicator, the outcome favouring n^2MFn^2 is the same as that of the Cohen-Kappa score for GLRC (Figure 3.3(b)), PDC (Figure 3.5(b)) and MovieLens (Figure 3.7(b)) datasets. For the ONP dataset (Figure 3.4(b)), the count favouring n^2MFn^2 is only one, and for the SP dataset, this value is zero (Figure 3.6(b)).

The preceding description clearly shows that in the majority of situations, for GLRC, PDC and MovieLens datasets and four types of classifiers, the accuracy score of the transformed dataset using n^2MFn^2 has surpassed the others. For ONP and SP datasets,

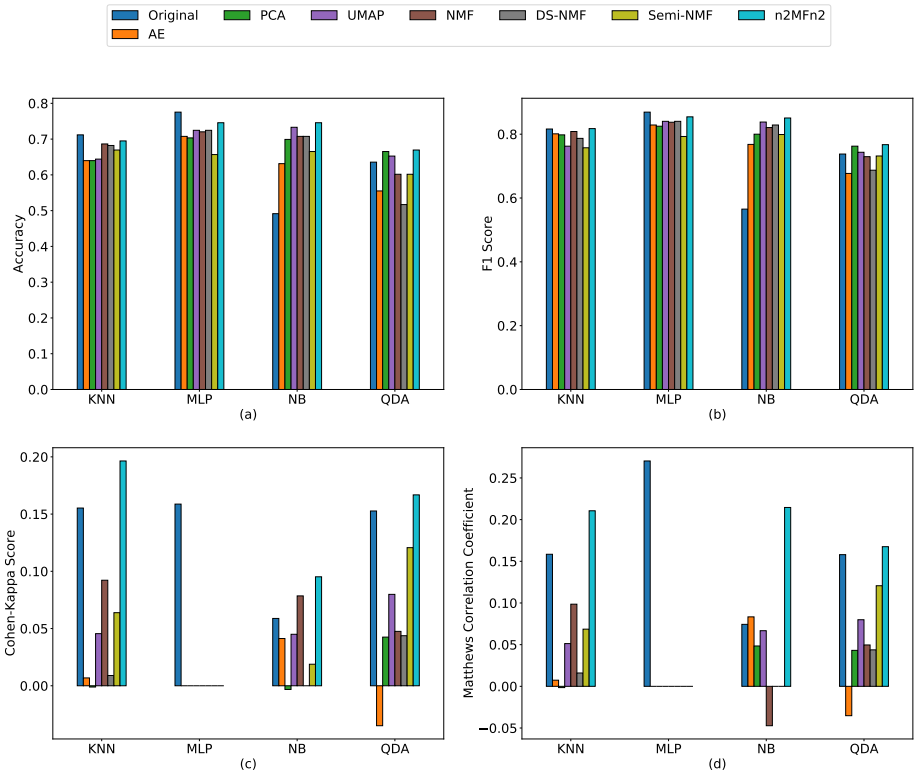

FIGURE 3.7: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the MovieLens dataset by n^2MFn^2 and six other dimension reduction techniques along with the original data.

n^2MFn^2 generated low rank embedding has failed over other dimension reduction techniques. Accuracy quantifies how often the model is correct. Together with accuracy, we have computed the F1 score, i.e., the harmonic mean of precision and recall. Figures 3.3-3.7 show that n^2MFn^2 has outperformed other models in terms of F1 score in most of the situations except SP dataset. Thus, the supremacy of n^2MFn^2 is justified by its Accuracy and F1 score. The pictorial illustrations show that n^2MFn^2 has resulted in greater positive Cohen-Kappa scores and outperformed the others in all of the situations for GLRC, PDC and MovieLens datasets. As a consequence, it is possible to conclude that n^2MFn^2 is able to maintain and learn the inherent qualities of the input, resulting in higher scores for GLRC, PDC and MovieLens datasets. Whereas for ONP and SP datasets, n^2MFn^2 has performed poorly. Matthews Correlation Coefficient assesses the quality of binary and multiclass classifications. A higher MCC score implies better agreement, which means that the model can preserve the class properties of the original dataset in the altered dataset as well. Figures 3.3-3.7 show that n^2MFn^2 has outperformed the other models in terms of MCC score for GLRC, PDC and MovieLens

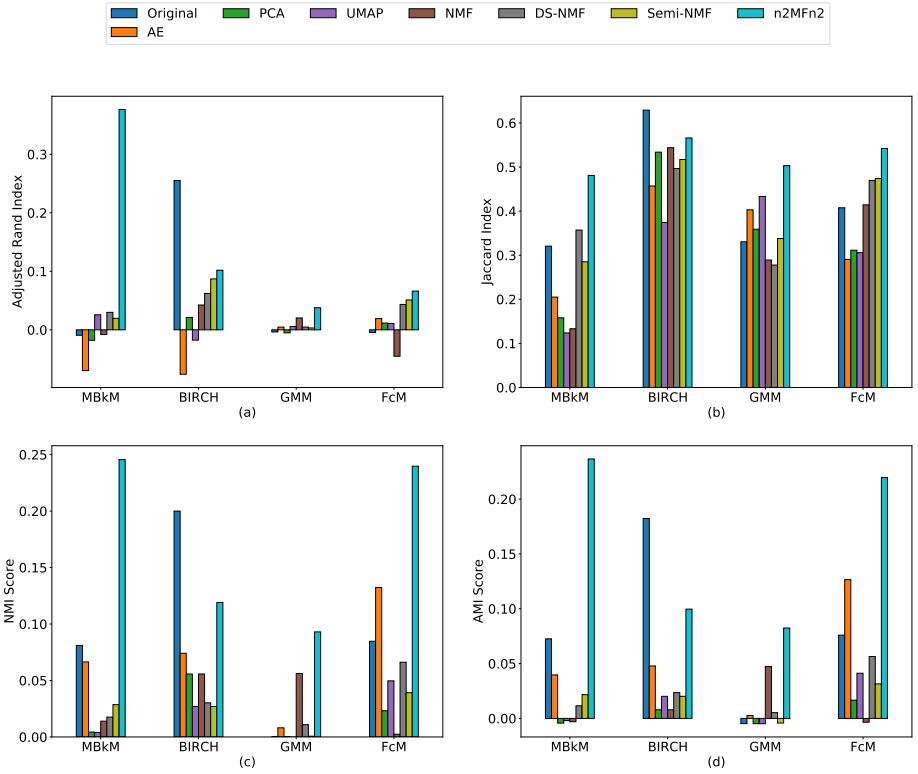

FIGURE 3.8: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the GLRC dataset by n^2MFn^2 and six other dimension reduction techniques along with the original data.

TABLE 3.2: The summary of the count (out of 24) of statistically significant p-values for each classification performance metric against each dataset with respect to n^2MFn^2.

Dataset	ACC	FS	CKS	MCC
GLRC	24	23	23	22
ONP	18	20	19	20
PDC	22	22	23	23
SP	10	11	13	11
MovieLens	23	22	11	09

datasets. The preceding discussion demonstrates the advantage of n^2MFn^2 over the other dimension reduction algorithms in terms of both statistical and intrinsic property preservation metrics. Thus, for GLRC, PDC and MovieLens datasets, n^2MFn^2 has suppressed others but for ONP and SP datasets the model performance is comparable with others.

For each classification performance index against each dataset, out of a total of 24 p-values, the count of p-values less than the assumed threshold (0.05) value is presented in Table 3.2. Tables A.1-A.20 of Appendix A depicts the actual p-values of n^2MFn^2

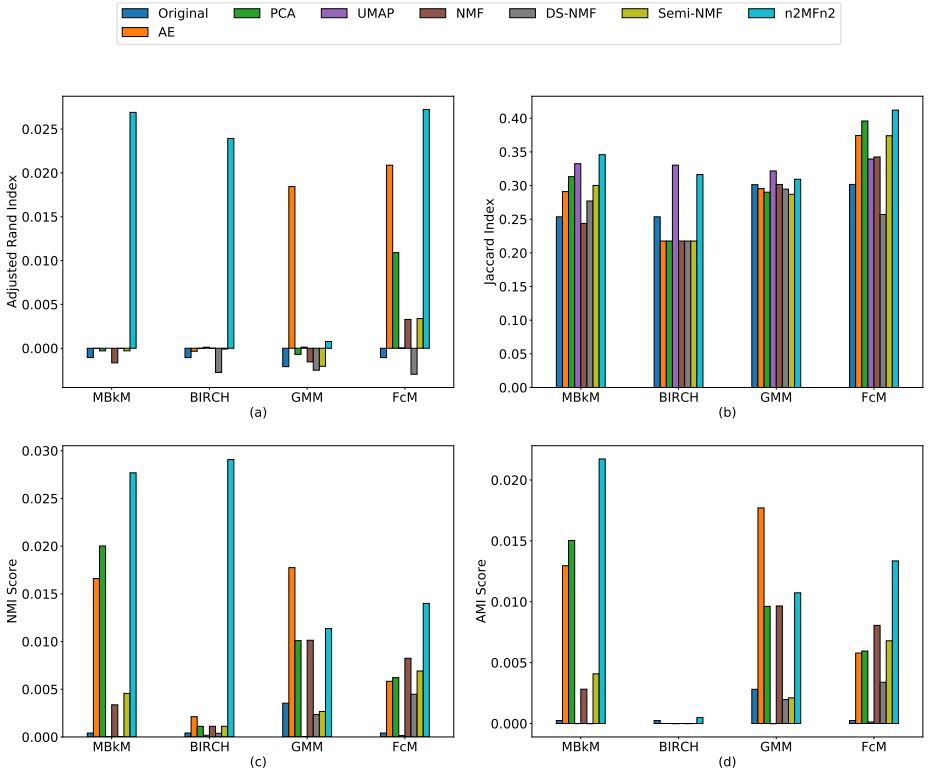

FIGURE 3.9: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the ONP dataset by n^2MFn^2 and six other dimension reduction techniques along with the original data.

over six other dimension reduction techniques for each of the five datasets, each of the four classifiers and each of the four classification performance evaluators. The above statistics indubitably quantify the quality of low rank embedding produced by n^2MFn^2 over others.

Clustering

For clustering purposes with the n^2MFn^2 model, Figures 3.8-3.12 present the outcome. Table 3.3 provides an overview of the count of statistically significant p-values for n^2MFn^2 for clustering.

n^2MFn^2 has achieved the highest performance score for the Adjusted Rand index for the GLRC (Figure 3.8(a)), PDC (Figure 3.10(a)) and SP (Figure 3.11(a)) datasets for all four clustering approaches considered here. This count is three out of four for the ONP (Figure 3.9(a)) dataset and one out of four for the MovieLens (Figure 3.12(a))

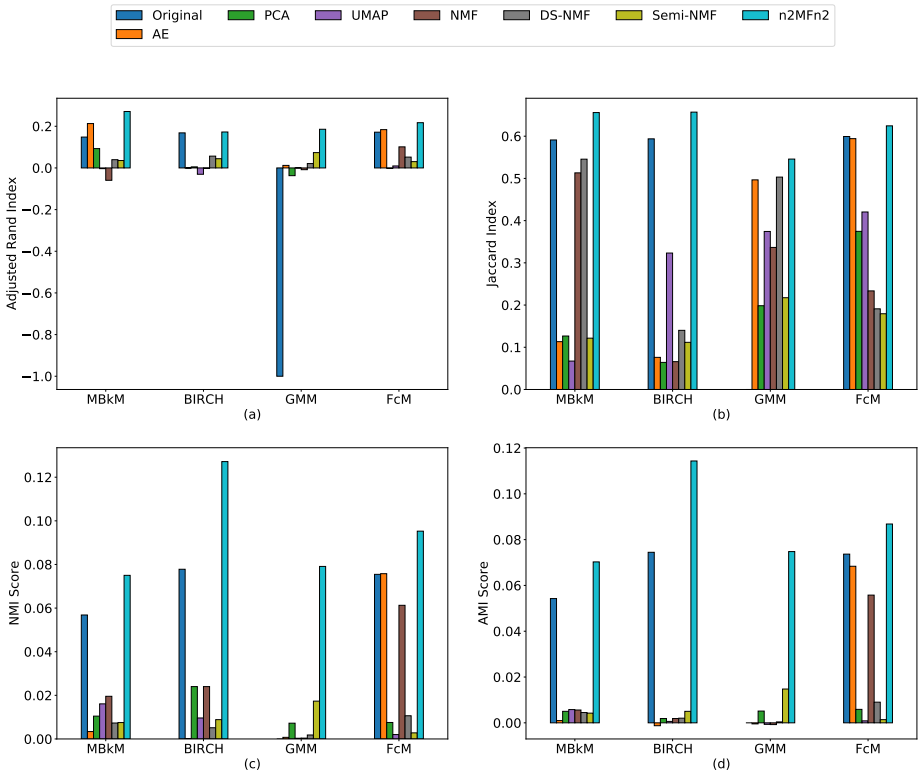

FIGURE 3.10: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the PDC dataset by n^2MFn^2 and six other dimension reduction techniques along with the original data.

dataset. When using the Jaccard Index as the cluster validity estimator, n^2MFn^2 has outperformed the others in four out of four clustering algorithms on all except the ONP dataset, for which the count is two (Figures 3.8(b), 3.9(b), 3.10(b), 3.11(b), 3.12(b)). For both NMI and AMI scores, n^2MFn^2 projected transformed space has outperformed others three out of four times for the ONP (Figures 3.9(c), 3.9(d)) dataset and for other four datasets considered here the count is four out of four.

Adjusted Rand Index (ARI) evaluates the similarity of two data clusterings. Figures 3.8-3.12 show that n^2MFn^2 has outperformed other dimension reduction techniques across five datasets and four clustering algorithms in terms of ARI score. The Jaccard Index is used to determine the similarity of two sets. n^2MFn^2 has outperformed the rest in terms of the Jaccard Index. Thus, it may be argued that n^2MFn^2 has learnt the fundamental features of the input and mapped them to a low rank representation well. NMI is defined as the normalisation of the Mutual Information score to scale the outcomes in $[0, 1]$. This metric is unadjusted for chance. The AMI score, on the other hand, is invariant to the permutation of the class or cluster label. Figures 3.8-3.12 show

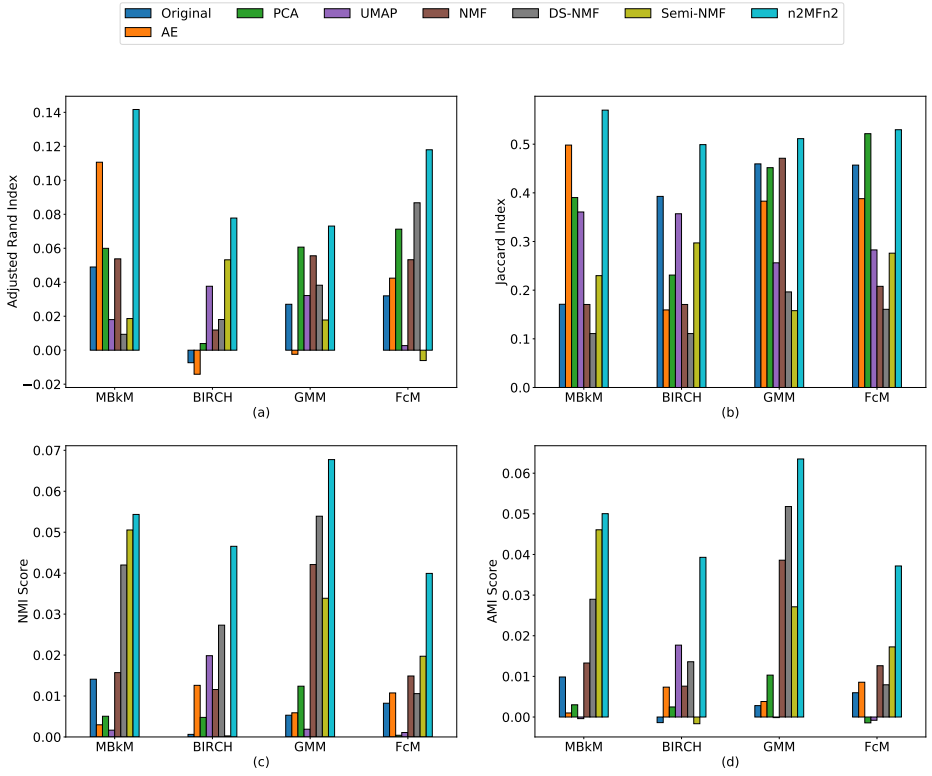

FIGURE 3.11: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the SP dataset by n^2MFn^2 and six other dimension reduction techniques along with the original data.

TABLE 3.3: The summary of the count (out of 24) of statistically significant p-values for each cluster performance metric against each dataset with respect to n^2MFn^2.

Dataset	ARI	JI	NMI	AMI
GLRC	24	23	22	22
ONP	16	21	21	22
PDC	23	24	23	23
SP	19	18	21	21
MovieLens	23	23	22	24

that n^2MFn^2 has outperformed other dimension reduction strategies in terms of both NMI and AMI scores. The improved performance of n^2MFn^2 demonstrates that the low rank representation of the datasets using n^2MFn^2 has been able to maintain the inherent qualities of the original data better than the other approaches considered here.

Out of the total of 24 p-values for each cluster validity index against each dataset, Table 3.3 displays the count of p-values that fall below the assumed threshold value, i.e., 0.05. The aforementioned tally unequivocally demonstrates how good low rank embedding produced by n^2MFn^2 is compared to others.

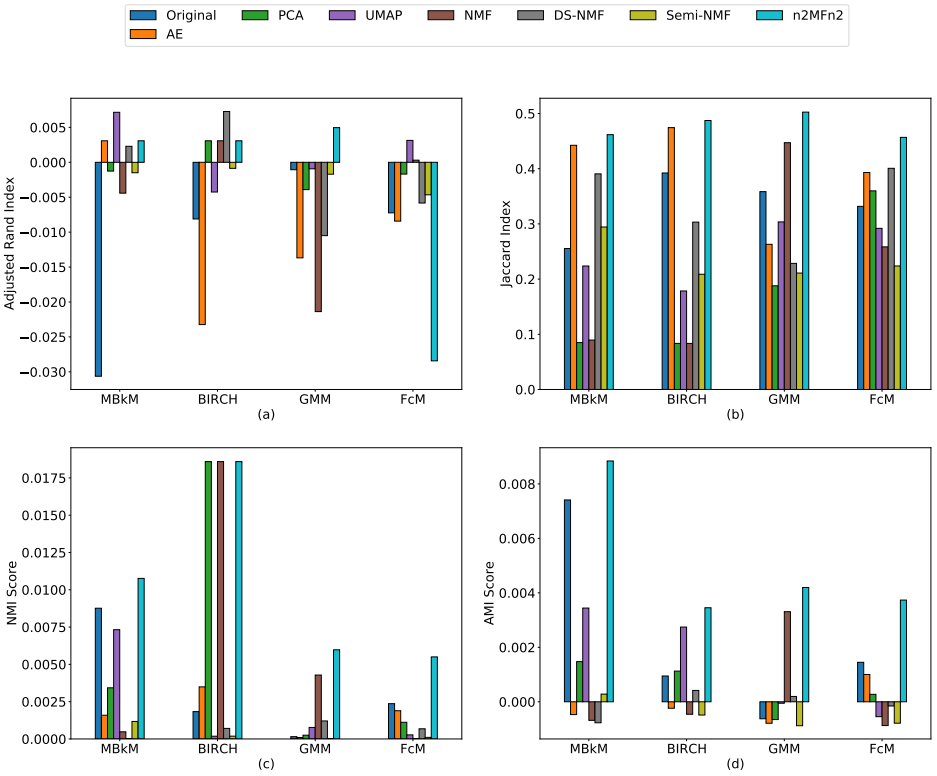

FIGURE 3.12: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the MovieLens dataset by n^{2}MFn^2 and six other dimension reduction techniques along with the original data.

3.4.3 Discussion

The ability to preserve the local structure of data by n^{2}MFn^2 over others has been studied and discussed using the trustworthiness score. The performance of n^{2}MFn^2 has been compared individually with other six dimension reduction methods over all five datasets, categorized into two sets based on the relation between the number of attributes and samples, as discussed above. It has been observed that n^{2}MFn^2 has showcased better performance than three out of six other methods in dimension reduction in terms of preservation of the granular relationship of data.

Different types of classification techniques have been considered to validate the performance of low rank approximation by the n^{2}MFn^2 model. For example, KNN is a non-parametric approach; whereas NB algorithm is a probabilistic classifier; MLP is a feed-forward artificial neural network; and QDA is a generative model-based classifier. We have also used two different types of clustering algorithms. MBkM, FcM and GMM

are centroid-based clustering algorithms. On the other hand, BIRCH is a hierarchical clustering methodology.

For five datasets, four classification algorithms and four classification performance measures, a total of $5 \times 4 \times 4 = 80$ performance scores are there for each dimension reduction algorithm. It can be seen that n^2MFn^2 projected datasets have performed better than the original data on 51 out of 80 occasions. On the other hand, n^2MFn^2 has secured the highest rating of 49 times out of 80 when comparing the performance with other dimension reduction algorithms. n^2MFn^2 has almost succeeded in all cases for GLRC, PDC and MovieLens datasets, whereas for ONP and SP datasets the performance is comparable. Thus, the supremacy of n^2MFn^2 over the others is comparable.

The low rank embeddings produced by n^2MFn^2 for different datasets are not only capable of outperforming the original and other dimensionally reduced datasets produced by different dimension reduction methods. They are also proven to be statistically significant in terms of the comparative p-values they produce. The overall count of statistically significant results related to n^2MFn^2 for all classifiers and classification metrics is 92 out of 96 $(4 \times 4 \times 6)$ for the GLRC dataset. The same count for the PDC, ONP, SP and MovieLens datasets are 77, 90, 45 and 65 respectively. Thus the efficacy of n^2MFn^2 is established over other dimension reduction algorithms in terms of producing statistically significant low dimensional embedding.

Similar to classification, four clustering algorithms and four cluster validity metrics have been used over five datasets to justify the competence of n^2MFn^2. For clustering, when comparing the performance against the original data, out of 80 possible cases, n^2MFn^2 has registered better performance 75 times. The count of supremacy for n^2MFn^2 with respect to other dimension reduction algorithms stands out to be 72 out of 80. In light of the preceding discussion, it is clear that in the majority of circumstances, n^2MFn^2 has proven to be superior to the other dimension reduction approaches considered here.

In addition to outperforming the original and other dimensionally reduced datasets produced by different dimension reduction methods, the low rank embeddings generated by n^2MFn^2 for various datasets have also been shown to be statistically significant based on the comparative p-values they produce. Considering all clustering algorithms and cluster validity indexes together, the total number of n^2MFn^2 related statistically

significant findings is 91 out of 96 $(4 \times 4 \times 6)$ for the GLRC dataset. The PDC, ONP, SP and MovieLens datasets have the same count of 80, 93, 79 and 92, respectively. Thus, it is proven that n^2MFn^2 is more effective than other dimension reduction algorithms in generating low dimensional embeddings that are statistically significant.

As previously stated, the datasets on which we have experimented are divided into two sets depending on the relation between the number of samples and attributes. n^2MFn^2 has demonstrated superiority for both types of datasets, proving its ability in dimension reduction, and invariance to the relationship between the number of samples and attributes. Furthermore, the number (r) of the reduced dimension is not limited by the number of samples or attributes. These properties distinguish n^2MFn^2 from several other widely used dimension reduction approaches. As a result, it is established that n^2MFn^2 is widely applicable and not limited by input dimension. As a consequence, n^2MFn^2 has outperformed the other six cutting-edge dimension reduction methods for various classification and clustering approaches on two separate categories of datasets.

The above two modes of experiments establish the superior performance of n^2MFn^2 not only competing with six other state-of-the-art dimension reduction techniques but also showcasing their comparative performance to preserve the local structure of data. The need for dimension reduction in contrast to working with the original data has also been demonstrated. The results have also been judged statistically and have been proven to be so.

3.5 Convergence Analysis

We have established the convergence of the proposed n^2MFn^2 model using experimental results. The convergence plots of n^2MFn^2 for all five datasets is shown in Figure 3.13. The plots illustrate the variation of the cost function Φ against iteration for all five datasets. Overall, the nature of the cost over time validates that the model converges. It can also be observed from the plots that the initial cost value for all the datasets starts from a high position and after a few initial epochs, the value of the cost function has almost reached a straight line parallel to the horizontal axis. That is, there are very

nominal changes in the cost value. Thus, we can conclude that the model has converged. Figure 3.13(a) depict the cost versus iteration plot for the GLRC dataset with $r = 69$. Similarly, Figure 3.13(b) and 3.13(c) represent the convergence plots for the ONP ($r = 30$) and PDC ($r = 252$) datasets respectively. The plot related to the SP dataset is portrayed in Figure 3.13(d) with $r = 15$ and Figure 3.13(e) depicts the convergence plot for the MovieLens dataset with $r = 269$.

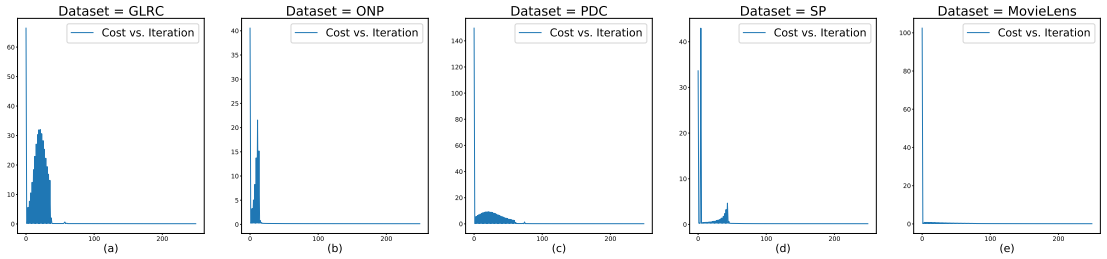

FIGURE 3.13: Cost vs. iteration plot of n^2MFn^2 for (a) GLRC, (b) ONP, (c) PDC, (d) SP and (e) MovieLens dataset.

3.6 Analysis of Computational Complexity

Computational complexity of n^2MFn^2 is calculated in terms of the number of operations done. n^2MFn^2 is a shallow neural network architecture with only three layers: input, hidden and output. The input, i.e., each row of $\mathbf{X}$, passes through the identity function at the input layer, hence the computational complexity is $\mathcal{O}(mn)$. The computational complexity for the next step, as defined in equation (3.3.3), is $\mathcal{O}(mnr)$. The value of $\mathbf{Y}$ is now passed via the activation function ψ (equation (3.3.2)) and the complexity of this operation is $\mathcal{O}(mr)$. The subsequent step, equation (3.3.5), necessitates $\mathcal{O}(mrn)$ operations. Finally, the output layer computes $\hat{\mathbf{X}}$ (equation (3.3.4)) with $\mathcal{O}(mn)$ complexity. As a result, $\mathcal{O}(mn + mnr + mr + mrn + mn)$ operations are required for the forward pass. We may express the computational complexity of the forward pass as $\mathcal{O}(mnr)$ after eliminating the lower order terms. Similar to this, the complexity of computing Φ (equation (3.3.6)) is $\mathcal{O}(mn + n^2)$. Updation of weights through backpropagation algorithm requires $\mathcal{O}(mnr + n^2r)$ operations that are defined in equations (3.3.19) and (3.3.22). As a result, an epoch has a computational complexity of $\mathcal{O}(mnr + n^2r)$. For such t_p epochs, the complexity is $\mathcal{O}(t_p(mnr + n^2r))$.

3.7 Conclusions

There exist a large number of methods for dimensionally reducing a big dataset. The benefits of NMF and artificial neural networks have been combined in this chapter. A shallow neural network model, called n^2MFn^2, has been developed for the task of NMF towards dimensionality reduction. n^2MFn^2 is composed of two phases: deconstruction and reconstruction. The reduced dimension r should be smaller than the number of features (n). The sample size m does not constrain the choice of r. The methods for initialisation of weights, transfer function, objective function and the learning algorithm have been designed in a manner that it supports optimal learning of n^2MFn^2.

Extensive experimentation on five well-known datasets has been performed to justify the efficacy of n^2MFn^2 over the original dataset as well as over six other notable dimension reduction strategies. Along with three conventional dimension reduction algorithms, three additional NMF-based dimension reduction strategies have been chosen for comparison. Four different classification/clustering algorithms and four different classification/clustering performance metrics have been used for downstream analyses. The statistical significance of the resultset has also been depicted. When compared with the original dataset in terms of classification n^2MFn^2 has performed better in three out of five datasets, equally good in one and not so good in the remaining dataset. In terms of clustering n^2MFn^2 have performed better for all five datasets. Thus, overall the need for dimension reduction has been established. In terms of classification, n^2MFn^2 has outperformed other dimension reduction techniques in three out of five datasets, while not performing that well in the remaining two. Whereas, for clustering, n^2MFn^2 has always performed better. Thus, the authority of n^2MFn^2 over the other dimension reduction strategies taken into consideration has been justified by the results obtained. The test of statistical significance between the resultset produced by n^2MFn^2 and that of other dimension reduction algorithms also justifies the efficacy of n^2MFn^2. Trustworthiness score has been used to assess the quality of local structure preservation in the low dimensional embedding by n^2MFn^2 in comparison with other dimension reduction techniques considered here. Experimental results show that the performance of n^2MFn^2 is competitive with others in terms of preserving the local structure of data. It is better than half of the methods compared with.

The present chapter portrays NMF in a shallow neural network architecture. However, deep neural network architecture can be used for the task of NMF. Deep learning framework should be used to incorporate the advantage of hierarchical learning during dimension reduction of huge datasets. ReLU activation function has been used in n^2MFn^2 to satisfy the non-negativity constraint of the problem, but ReLU activation function suffers from the data loss problem, because of its nature of discarding negative elements. Replacing ReLU with some other activation function has been incorporated in the next chapters.

Chapter 4

Deep Neural Network for Non-negative Matrix Factorization (DN3MF)

4.1 Introduction

In the previous chapter, a shallow neural network model, n^2MFn^2, has been developed for NMF aiming towards dimension reduction of large datasets. We have seen a mixed performance of n^2MFn^2 in contrast to the original dataset and other dimension reduction techniques. Some points have also been highlighted where possible modifications can be made over n^2MFn^2 for an enhanced model design.

In this chapter, we have attempted to combine the advantages of the traditional iterative procedure with the current deep learning framework. One of the major advantages of deep learning is hierarchical learning. Thus, the representation of data is learned in a layer wise manner. We have developed a deep learning model, named Deep Neural Network for Non-negative Matrix Factorization (DN3MF), for the task of NMF aiming towards low rank approximation of the data matrix [27]. There are two stages of the model, namely, pretraining and stacking. The pretraining stage is accomplished by a shallow neural network architecture and the stacking stage is a deep neural network architecture.

The performance of DN3MF is basically a composite effect of the two-stage training as well as the neural network's ability to hierarchical learning and optimise parameters. Each of these stages is divided into two phases, namely, deconstruction and reconstruction. The novel architecture of DN3MF guarantees the non-negativity criteria of the model. The use of sigmoid activation function helps to alleviate the data loss problem in contrast to ReLU. To satisfy the non-negativity criterion of the model sigmoid activation function has been judiciously modified. The exploding or vanishing gradient problem has been resolved by using the Xavier initialization approach. Regularisation has been used in the formulation of the model's objective function to ensure the best feasible approximation of the input matrix. The development of a unique adaptive learning mechanism has helped to achieve the objective of the model.

The superiority of DN3MF over seven well-known dimension reduction techniques has been demonstrated in two ways. The ability of DN3MF to preserve the local structure of data in low dimensional embedding has been judged over that of other dimension reduction algorithms. On the other hand, the quality of dimension reduction with respect to downstream analyses using classification and clustering has been performed. Different types of datasets, classification techniques, clustering algorithms and evaluation indexes have been used to support the efficiency of DN3MF. Moreover, the statistical significance of the results has also been thoroughly experimented with. The computational complexity of DN3MF along with the convergence analysis has also been presented.

The remaining parts of the chapter have been organised as follows. Section 4.2 discusses the motive behind the architecture and learning of DN3MF. Section 4.3 showcases the comprehensive design and derivation of the respective rules of learning. Section 4.4 illustrates the outcomes of the experimentation process presented in Chapter 2, including satisfactory analysis. Sections 4.5 and 4.6 provide the DN3MF convergence and computational complexity analyses, respectively. Finally, Section 4.7 brings the chapter to a conclusion.

4.2 Motivation behind Architecture and Learning

The deep Neural Network for Non-negative Matrix Factorization (DN3MF) model is a two-stage architecture developed for the task of NMF towards low rank approximation. Pretraining and stacking are the two stages of DN3MF. Pretraining is achieved using a shallow neural network architecture and stacking is performed using a deep neural network architecture. Each of the stages is divided into two phases, called deconstruction and reconstruction.

In NMF, we try to break down (deconstruct) a given matrix into two non-negative factor matrices. For both the pretraining and stacking stages of DN3MF, first, we deconstruct a given matrix into two factor matrices and then reconstruct the input matrix from these factor matrices. The novel architecture of the stacking stage of DN3MF has multiple deconstruction layers and the same number of reconstruction layers. Hence, the architecture can be referred to as the Multiple Deconstruction Multiple Reconstruction (MDMR) framework. It may be mentioned here that in a standard autoencoder, unlike in DN3MF, an encoded version of the data is generated. This is followed by decoding the encoded version to get an approximate version of the original data. In DN3MF, the input to the model is transformed into the latent space in the deconstruction phase. The latent space representation is used as the input to the reconstruction phase that tries to reconstruct the best possible approximation of the input while satisfying the non-negativity requirement.

Different architectural constraints have been addressed. To meet the non-negativity criteria, sigmoid activation function has been modified, and thus, the data loss problem has also been solved. Sigmoid function takes the input of any interval $(-\infty, +\infty)$ and maps the output to the interval (0, 1). There exist other activation functions, like ReLU, which also guarantee non-negativity by dropping the negative terms. In that sense, ReLU activation function suffers from the data loss problem, whereas sigmoid function does not. This modification of the sigmoid function is intended towards ensuring the non-negativity of the factors obtained by the model.

The exploding or vanishing gradient problem has been tackled by maintaining the variance of the activation same across every layer. Xavier initialization [34] technique has been used in this regard to initialize the weights of the shallow neural network.

The objective function has been designed following the same procedure of n^{2}MFn^2 as described in Chapter 3 Section 3.2. Momentum factor has been used to speed up the convergence of the gradient-based optimization. The learned weights in the pretraining stage are used as the initialization of weights of the stacking stage. The architecture and respective learning algorithm of the stages of DN3MF have been described below.

4.3 DN3MF

In this section, we design the two-stage architecture of DN3MF and formulate its learning rules in detail.

4.3.1 Pretraining Stage

The first stage of DN3MF is carried out with the help of a shallow neural network architecture. The primary aim of this stage is to determine the initial weight values for the stacking stage of the model. The given data matrix $\mathbf{U} = [u_{pi}]_{m\times n'}$ is processed using the procedures outlined in Chapter 2 Section 2.3 to produce a matrix $\mathbf{X} = [x_{pi}]_{m\times n}$ with each element being non-negative. Each row of $\mathbf{X}$ now serves as an input to the pertaining stage of the model. The design of the shallow model, followed by its learning has been described in the next two sections.

4.3.1.1 Architecture of the Pretraining Stage

The pretraining stage architecture of DN3MF is the same as the architecture of n^{2}MFn^2 as described in Chapter 3 Section 3.3.1. A dataset having m samples and each sample having n features, acts as the input to the model. The hidden layer has been designed to act as the slender layer of the model having r nodes and thus extracts $r < n'$ features. There is no restriction of r with respect to the number of samples m (Section 3.3.1, Chapter 3). The only difference with n^{2}MFn^2 is for the hidden and output layer nodes, a modified version of the sigmoid activation function σ has been used. Sigmoid activation function S is defined as

$$S(x) = \frac{1}{1+e^{-x}} \tag{4.3.1}$$

We have defined σ as,

$$\sigma(x) = \begin{cases} S(x), & \text{if } S(x) > 0 \\ \epsilon, & \text{otherwise} \end{cases} \tag{4.3.2}$$

where $\epsilon > 0$ is a user-specified small number. We have chosen $\epsilon = 0.001$ to avoid the problem of division by zero during the execution of the algorithm. The hidden layer output $\mathbf{B}$ and weight matrix $\mathbf{W}$ are the two non-negative factors of the regenerated input matrix $\widehat{\mathbf{X}}$.

4.3.1.2 Learning of the Pretraining Stage

The objective function of the model is the same as described in Chapter 3 Section 3.3.2 and is defined as

$$\Phi = \frac{1}{mn} \sum_{p=1}^{m} \sum_{j=1}^{n} \frac{1}{2} (x_{pj} - \widehat{x}_{pj})^2 + \frac{\lambda}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{1}{2} (\sum_{l=1}^{r} v_{il} w_{lj} - \delta_{ij})^2 \tag{4.3.3}$$

For minimising Φ with respect to $\mathbf{V}$ and $\mathbf{W}$, the weight matrices are modified iteratively. Adopting the gradient descent technique we get

$$\mathbf{V}(t+1) = \mathbf{V}(t) - \eta_{\mathbf{V}(t)} \circ \nabla_{\mathbf{V}(t)} \Phi \tag{4.3.4}$$

and

$$\mathbf{W}(t+1) = \mathbf{W}(t) - \eta_{\mathbf{W}(t)} \circ \nabla_{\mathbf{W}(t)} \Phi \tag{4.3.5}$$

Here t is the iteration count. The matrices $\eta_{\mathbf{V}(t)}$ and $\eta_{\mathbf{W}(t)}$ are two hyper-parameters of the model, called learning rates corresponding to $\mathbf{V}$ and $\mathbf{W}$. The terms $\nabla_{\mathbf{V}(t)}$ and $\nabla_{\mathbf{W}(t)}$ are the gradient operators with respect to the weight matrices $\mathbf{V}$ and $\mathbf{W}$ respectively.

Calculating the derivatives of Φ with respect to v_{il} and w_{lj} for $i = 1, 2, ..., n$, $l = 1, 2, ..., r$ and $j = 1, 2, ..., n$, we get

$$\frac{\partial \Phi}{\partial v_{il}} = -\frac{1}{mn} \sum_{p=1}^{m} \sum_{j=1}^{n} (x_{pj} - \widehat{x}_{pj}) \widehat{x}_{pj} (1 - \widehat{x}_{pj}) w_{lj} b_{pl} (1 - b_{pl}) x_{pi} + \frac{\lambda}{n^2} \sum_{j=1}^{n} \left((\sum_{l'=1}^{r} v_{il'} w_{l'j} - \delta_{ij}) w_{lj} \right) \tag{4.3.6}$$

and

$$\frac{\partial \Phi}{\partial w_{lj}} = -\frac{1}{mn}\sum_{p=1}^{m}(x_{pj}-\widehat{x}_{pj})\widehat{x}_{pj}(1-\widehat{x}_{pj})b_{pl} + \frac{\lambda}{n^2}\sum_{i=1}^{n}\left((\sum_{l'=1}^{r} v_{il'}w_{l'j} - \delta_{ij})v_{il}\right) \tag{4.3.7}$$

Using equations (4.3.6) and (4.3.7), $\nabla_{\mathbf{V}}$ and $\nabla_{\mathbf{W}}$ can be written as

$$\nabla_{\mathbf{V}}\Phi = -\frac{1}{mn}\mathbf{X}^T((((\mathbf{X}-\widehat{\mathbf{X}})\circ\widehat{\mathbf{X}}\circ(1-\widehat{\mathbf{X}}))\mathbf{W}^T)\circ(\mathbf{B}\circ(1-\mathbf{B}))) + \frac{\lambda}{n^2}(\mathbf{VW}-\mathbf{I})\mathbf{W}^T \tag{4.3.8}$$

and

$$\nabla_{\mathbf{W}}\Phi = -\frac{1}{mn}\mathbf{B}^T((\mathbf{X}-\widehat{\mathbf{X}})\circ\widehat{\mathbf{X}}\circ(1-\widehat{\mathbf{X}})) + \frac{\lambda}{n^2}\mathbf{V}^T(\mathbf{VW}-\mathbf{I}) \tag{4.3.9}$$

Now, rewriting the learning rules, defined in equations (4.3.4) and (4.3.5), using equations (4.3.8) and (4.3.9), we get

$$\mathbf{V}(t+1) = \mathbf{V}(t) - \eta_{\mathbf{V}(t)}\circ\left(-\frac{1}{mn}\mathbf{X}^T((((\mathbf{X}-\widehat{\mathbf{X}})\circ\widehat{\mathbf{X}}\circ(1-\widehat{\mathbf{X}}))\mathbf{W}^T)\circ(\mathbf{B}\circ(1-\mathbf{B}))) + \frac{\lambda}{n^2}(\mathbf{VW}-\mathbf{I})\mathbf{W}^T\right) \tag{4.3.10}$$

and

$$\mathbf{W}(t+1) = \mathbf{W}(t) - \eta_{\mathbf{W}(t)}\circ\left(-\frac{1}{mn}\mathbf{B}^T((\mathbf{X}-\widehat{\mathbf{X}})\circ\widehat{\mathbf{X}}\circ(1-\widehat{\mathbf{X}})) + \frac{\lambda}{n^2}\mathbf{V}^T(\mathbf{VW}-\mathbf{I})\right) \tag{4.3.11}$$

According to the architecture of the model, the elements in $\mathbf{V}$ are unrestricted, but the elements of $\mathbf{W}$ should be non-negative. To satisfy this criterion, we choose $\eta_{\mathbf{W}}$ in such a manner that the negative terms arising from the computation in equation (4.3.11) get dismissed. We choose $\eta_{\mathbf{W}}$ as

$$\eta_{\mathbf{W}} = (mn\mathbf{W}) \oslash (\mathbf{B}^T(\widehat{\mathbf{X}}\circ\widehat{\mathbf{X}} + \mathbf{X}\circ\widehat{\mathbf{X}}\circ\widehat{\mathbf{X}})) \tag{4.3.12}$$

Here, $\oslash$ denotes Hadamard division. Now, using equation (4.3.12), equation (4.3.11) becomes,

$$\mathbf{W}(t+1) = ((\mathbf{W}(t) \oslash (\mathbf{B}^T(\widehat{\mathbf{X}} \circ \widehat{\mathbf{X}} + \mathbf{X} \circ \widehat{\mathbf{X}} \circ \widehat{\mathbf{X}}))) \circ \mathbf{B}^T(\mathbf{X} \circ \widehat{\mathbf{X}} + \widehat{\mathbf{X}} \circ \widehat{\mathbf{X}} \circ \widehat{\mathbf{X}})) - \frac{m}{n}((\mathbf{W}(t) \oslash (\mathbf{B}^T(\widehat{\mathbf{X}} \circ \widehat{\mathbf{X}} + \mathbf{X} \circ \widehat{\mathbf{X}} \circ \widehat{\mathbf{X}}))) \circ \lambda(\mathbf{V}(t)^T(\mathbf{V}(t)\mathbf{W}(t) - \mathbf{I}))) \quad (4.3.13)$$

It is to be noted that the elements in $\mathbf{X}$, $\mathbf{B}$, $\mathbf{W}$ and $\widehat{\mathbf{X}}$ are all positive. Hence the first part on the right-hand side of the equation (4.3.13) is positive. The second part of the equation (4.3.13) contains the expression $(\mathbf{VW} - \mathbf{I})$. As per the objective of the model, this term will gradually vanish over the iterations because the value of $\mathbf{VW}$ will tend towards $\mathbf{I}$. Even after the above, during back-propagation of the network, if some of the elements of $\mathbf{W}$ become negative then those negative values are replaced with 0.001. Hence, the consistency of the model with respect to the non-negativity criterion is maintained.

λ is the regularizing parameter used in the objective function of DN3MF. The value of λ decides the amount by which we want to penalize the model's flexibility. We have used ridge regression in the model and it is used to overcome the overfitting of the model. The ridge regression technique works by preventing coefficient values from becoming too high. For $\lambda = 0$, the effect of the penalty term goes away and as $\lambda \to \infty$, the effect of the shrinkage penalty grows. Thus, we have chosen $\lambda = 0.1$ to have a controlled effect on the regularizer when DN3MF tries to regenerate the input. All the elements of the hyper-parameter matrix $\eta_{\mathbf{V}}$ have been set to 0.1, i.e., $\eta_{\mathbf{V}} = [0.1]_{n \times r}$. Hence, the update rules for $\mathbf{V}$ and $\mathbf{W}$ are given by equations (4.3.10) and (4.3.13) respectively.

To speed up the convergence of the gradient based optimisation technique we use the momentum factor. It is a method to accelerate learning in low curvature directions while remaining stable in high-curvature directions. While small values of α do not have much effect in the process of learning, large values ($\alpha > 1$) may lead to lower importance in minimizing the objective function. Thus, we have considered $\alpha \in (0, 1)$, in general, and $\alpha = 0.9$ in particular for updating the weight matrix $\mathbf{V}$. Hence, we can rewrite equation (4.3.4) incorporating the momentum factor $\alpha_{\mathbf{V}}$ as

$$\mathbf{V}(t+1) = \mathbf{V}(t) - \eta_{\mathbf{V}(t)} \circ \nabla_{\mathbf{V}(t)}\Phi + \alpha_{\mathbf{V}} \circ \nabla_{\mathbf{V}(t-1)}\Phi \quad (4.3.14)$$

Let there be s successive shallow models in the pretraining stage of DN3MF. The first shallow network takes $\mathbf{X}^{(0)}$ as its input. The output $\mathbf{X}^{(1)} = [x^{(1)}_{pi_1}]_{m\times r_1}$ of the slender layer of this first shallow model is used as the input for the second shallow model. Similarly, the slender layer output of the second shallow model acts as the input for the third shallow model and so on. Thus the slender layer output of the s^{th} shallow model is $\mathbf{X}^{(s)} = [x^{(s)}_{pi_s}]_{m\times r_s}$. The term r_s is the required reduced dimension, and thus, concludes the first stage of DN3MF. Training of these s shallow networks is performed one after another. For this purpose, the target output of the nodes in the reconstruction layers of these s shallow networks are $\mathbf{X}^{(0)}$, $\mathbf{X}^{(1)}$, ..., $\mathbf{X}^{(s-1)}$. Following the properties of the shallow architecture, the entries in the weight matrices $\mathbf{V}^{(1)}, \mathbf{V}^{(2)}, ..., \mathbf{V}^{(s)}$ are unrestricted and the entries in the weight matrices $\mathbf{W}^{(1)}, \mathbf{W}^{(2)}, ..., \mathbf{W}^{(s)}$ are non-negative in nature. Figure 4.1 depicts the g^{th} $(1 \leq g \leq s)$ shallow model.

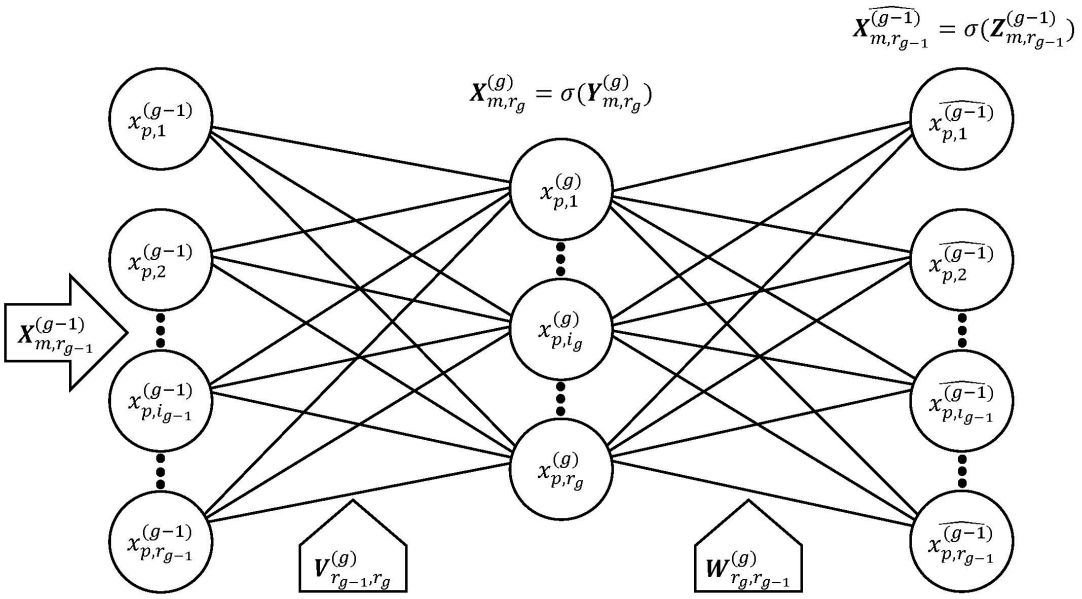

FIGURE 4.1: Pretraining stage architecture of DN3MF.

4.3.2 Stacking Stage

A deep neural network architecture is used in the second stage of DN3MF. The learned weights of the pre-trained shallow models are used to initialise the weight values of the deep model. This stage fine-tunes the weight values. The architecture of the deep model along with its learning has been presented in the following sections.

4.3.2.1 Architecture of the Stacking Stage

The stacking stage of the model uses s pretrained shallow neural network models stacked together forming a deeper network architecture. As described in Chapter 2 Section 2.3, the input data matrix $\mathbf{U} = [u_{pi}]_{m \times n'}$ is processed to form the matrix $\mathbf{X}^{(0)} = [x^{(0)}_{pi_0}]_{m \times r_0}$, where $r_0 = n$. This matrix $\mathbf{X}^{(0)}$ is used as the input to the deep neural network model. The model uses the same set of activation functions, viz., the identity function and the modified version of the sigmoid activation function σ, as defined earlier. The architecture of stacked pre-trained shallow models is illustrated in Figure 4.2.

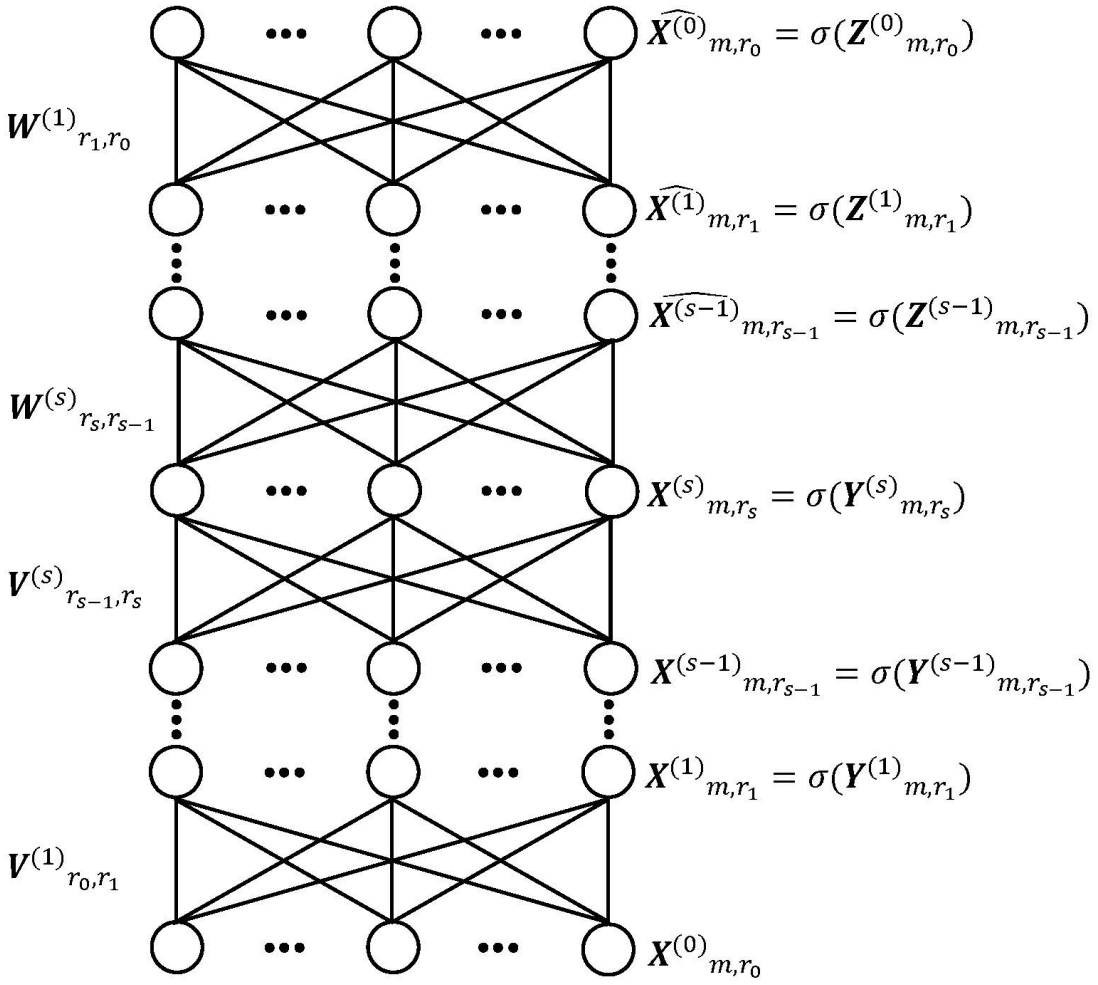

FIGURE 4.2: Stacking stage architecture of DN3MF.

The model can be divided into two phases, namely, the deconstruction phase and the reconstruction phase. In the deconstruction phase, the input data is interpreted in lower dimensional representation in a step-wise approach and in the reconstruction

phase, the aim is to regenerate the input data from the latent representation of data in a step-by-step fashion. The architecture has the input layer, followed by s deconstruction layers and on top of them, there are s reconstruction layers. As the stacked model has multiple deconstruction layers and multiple reconstruction layers, the architecture is described as multiple deconstruction multiple reconstruction deep learning architecture. The input layer, having $r_0 = n$ nodes, receives $\mathbf{X}^{(0)} = [x^{(0)}_{pi_0}]_{m\times r_0}$ as its input and with the identity function as the activation function of the layer, generates the output same as its input. The input layer is followed by the first deconstruction layer having r_1 nodes, where $r_1 < r_0$. The term $\mathbf{V}^{(1)} = [v^{(1)}_{i_0 i_1}]_{r_0\times r_1}$ is the weight matrix between the input and the first deconstruction layer. $\mathbf{V}^{(1)}$ is initialized by the trained $\mathbf{V}^{(1)}$ in the first pretrained shallow model. The first deconstruction layer generates output $\mathbf{X}^{(1)} = [x^{(1)}_{pi_1}]_{m\times r_1}$, where $\mathbf{X}^{(1)} = \sigma(\mathbf{Y}^{(1)})$ and $\mathbf{Y}^{(1)} = [y^{(1)}_{pi_1}]_{m\times r_1}$. The term $\mathbf{Y}^{(1)}$ is calculated as $\mathbf{Y}^{(1)} = \mathbf{X}^{(0)}\mathbf{V}^{(1)}$. Now, $\mathbf{X}^{(1)}$ acts as the input to the second deconstruction layer and follows the same procedure as the previous layer. Thus, for d^{th} $(1 \leq d \leq s)$ deconstruction layer,

$$\mathbf{X}^{(d)} = \sigma(\mathbf{Y}^{(d)}) \tag{4.3.15}$$

where $\mathbf{X}^{(d)} = [x^{(d)}_{pi_d}]_{m\times r_d}$ is the output of d^{th} deconstruction layer and $\mathbf{Y}^{(d)} = [Y^{(d)}_{pi_d}]_{m\times r_d}$, and

$$\mathbf{Y}^{(d)} = \mathbf{X}^{(d-1)}\mathbf{V}^{(d)} \tag{4.3.16}$$

where $\mathbf{V}^{(d)}$ is the weight matrix between the $(d-1)^{th}$ and d^{th} deconstruction layer of the model.

The weight matrix $\mathbf{V}^{(d)}$ is initialized by the trained $\mathbf{V}^{(d)}$ in the d^{th} pretrained shallow model. It is to be noted that $r^{d-1} > r^d$. Thus, for $d = s$, the final deconstruction layer is the slenderest layer of the model in terms of the number of nodes in that layer. This slenderest layer acts as the bottleneck layer of the model and generates $\mathbf{X}^{(s)} = [x^{(s)}_{pi_s}]_{m\times r_s}$ as its output. Thus, the model achieves the targeted reduced dimension r_s, the number of nodes in this layer. This slenderest layer concludes the deconstruction phase of the model and marks the beginning of the reconstruction phase of the model.

The slenderest layer is followed by the first reconstruction layer of the model with $\mathbf{X}^{(s)}$ acting as the input for the same and $\mathbf{W}^{(s)} = [w^{(s)}_{j_s j_{s-1}}]_{r_s\times r_{s-1}}$ is the weight matrix between these two layers. The weight matrix $\mathbf{W}^{(s)}$ is initialized by the trained $\mathbf{W}^{(s)}$ in the s^{th} pretrained shallow model. The first reconstruction layer generates output

$\widehat{\mathbf{X}^{(s-1)}} = [\widehat{x^{(s-1)}_{pj_{s-1}}}]_{m\times r_{s-1}}$, where $\widehat{\mathbf{X}^{(s-1)}} = \sigma(\mathbf{Z}^{(s-1)})$ and $\mathbf{Z}^{(s-1)} = [z^{(s-1)}_{pj_{s-1}}]_{m\times r_{s-1}}$. The term $\mathbf{Z}^{(s-1)}$ is calculated as $\mathbf{Z}^{(s-1)} = \mathbf{X}^{(s)}\mathbf{W}^{(s)}$. The second reconstruction layer receives $\widehat{\mathbf{X}^{(s-1)}}$ as its input and follows the same procedure as the previous layer. Thus, for e^{th} $(1 \leq e \leq s)$ reconstruction layer,

$$\widehat{\mathbf{X}^{(s-e)}} = \sigma(\mathbf{Z}^{(s-e)}) \tag{4.3.17}$$

where $\widehat{\mathbf{X}^{(s-e)}} = [\widehat{x^{(s-e)}_{pj_{s-e}}}]_{m\times r_{s-e}}$ is the output of the e^{th} reconstruction layer. $\mathbf{Z}^{(s-e)} = [z^{(s-e)}_{pj_{s-e}}]_{m\times r_{s-e}}$ is calculated as

$$\mathbf{Z}^{(s-e)} = \begin{cases} \mathbf{X}^{(s-e+1)}\mathbf{W}^{(s-e+1)}, \text{ for } e = 1 \\ \widehat{\mathbf{X}^{(s-e+1)}}\mathbf{W}^{(s-e+1)}, \text{ for } 1 < e \leq s \end{cases} \tag{4.3.18}$$

where $\mathbf{X}^{(s-e+1)} = [x^{(s-e+1)}_{pj_{s-e+1}}]_{m\times r_{s-e+1}}$, for $e = 1$, is the output of the slenderest layer, $\widehat{\mathbf{X}^{(s-e+1)}} = [\widehat{x^{(s-e+1)}_{pj_{s-e+1}}}]_{m\times r_{s-e+1}}$, for $1 < e \leq s$, is the output of $(e-1)^{th}$ reconstruction layer. $\mathbf{W}^{(s-e+1)}$, for $e = 1$, is the weight matrix between slenderest layer and e^{th} reconstruction layer of the model, $\mathbf{W}^{(s-e+1)}$ for $1 < e \leq s$, is the weight matrix between $(e-1)^{th}$ and e^{th} reconstruction layers of the model. $\mathbf{W}^{(s-e+1)}$ is initialized by the trained $\mathbf{W}^{(s-e+1)}$ in the $(s-e+1)^{th}$ pretrained shallow model. We impose the restriction $r_{s-e+1} < r_{s-e}$. Thus, for $e = s$, the final reconstruction layer, also known as the output layer of the model, generates $\widehat{\mathbf{X}^{(0)}} = [\widehat{x^{(0)}_{pj_0}}]_{m\times r_0}$ as its output and thus the architecture of the stacking stage tries to regenerate the input to the model, i.e., $\mathbf{X}^{(0)}$. As mentioned before, the elements in the weight matrices $\mathbf{V}^{(1)}, \mathbf{V}^{(2)}, ..., \mathbf{V}^{(s)}$ are unrestricted and the elements in the matrices $\mathbf{W}^{(s)}, ..., \mathbf{W}^{(2)}, \mathbf{W}^{(1)}$ are non-negative. The stacked model is finetuned maintaining these non-negativity constraints. The slenderest layer output $\mathbf{X}^{(s)}$ is one of the two non-negative factors of the regenerated input matrix $\widehat{\mathbf{X}^{(0)}}$. The combination of the weight matrices $\mathbf{W}$ along with the activation function σ constitutes the other non-negative factor of $\widehat{\mathbf{X}^{(0)}}$. Thus, we can write

$$\widehat{\mathbf{X}^{(0)}} = \sigma(\sigma(...\sigma(\sigma(\mathbf{X}^{(s)}\mathbf{W}^{(s)})\mathbf{W}^{(s-1)})...)\mathbf{W}^{1}) \tag{4.3.19}$$

4.3.2.2 Learning of the Stacking Stage

The objective of the model is to minimize $\|\mathbf{X}^{(0)} - \widehat{\mathbf{X}^{(0)}}\|_F$ with respect to $\mathbf{V}^{(1)}, \mathbf{V}^{(2)}, ..., \mathbf{V}^{(s)}$, $\mathbf{W}^{(s)}, ..., \mathbf{W}^{(2)}, \mathbf{W}^{(1)}$ subject to $\prod_{d=1}^{s} \mathbf{V}^{(d)} \prod_{e=1}^{s} \mathbf{W}^{(s-e+1)} = \mathbf{I}$, where $\mathbf{I} = [\delta_{ij}]_{r_0 \times r_0}$ is the Identity matrix of order r_0. Thus, the cost function Φ is defined as

$$\Phi = \frac{1}{mn} \sum_{p=1}^{m} \sum_{j=1}^{n} \frac{1}{2} (x_{pj}^{(0)} - \widehat{x_{pj}^{(0)}})^2 + \frac{\lambda}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{1}{2} (a_{ij} - \delta_{ij})^2 \tag{4.3.20}$$

Similar to the pretraining stage, the first term of equation (4.3.20) measures the reconstruction error and the second term acts as the regularizer. λ is the regularizing parameter and δ_{ij} is the Kronecker delta. In order for the model to learn and provide a meaningful representation of the input in the lower dimensional space, reconstruction error has been used as guidance. The term $\mathbf{A} = [a_{ij}]_{r_0 \times r_0}$ is defined as

$$a_{ij} = [\sum_{j_1=1}^{r_1} [...[\sum_{i_s=1}^{r_s} [\sum_{i_{s-1}=1}^{r_{s-1}} [...[\sum_{i_1=1}^{r_1} v_{i_0 i_1}^{(1)} v_{i_1 i_2}^{(2)}]...] v_{i_{s-1} i_s}^{(s)}] w_{j_s j_{s-1}}^{(s)}]...] w_{j_1 j_0}^{(1)}] \tag{4.3.21}$$

Here, it is to be noted that $n = r_0$, $i_s = j_s$, $i = i_0$ and $j = j_0$. For minimizing Φ (equation (4.3.20)) with respect to $\mathbf{V}^{(d)}$ and $\mathbf{W}^{(e)}$, for $1 \leq d \leq s$ and $1 \leq e \leq s$, $\mathbf{V}^{(d)}$ and $\mathbf{W}^{(e)}$ are iteratively modified as

$$\mathbf{V}^{(d)}(t+1) = \mathbf{V}^{(d)}(t) + \Delta \mathbf{V}^{(d)}(t) \tag{4.3.22}$$

$$\mathbf{W}^{(e)}(t+1) = \mathbf{W}^{(e)}(t) + \Delta \mathbf{W}^{(e)}(t) \tag{4.3.23}$$

Adopting the gradient descent technique, we get

$$\mathbf{V}^{(d)}(t+1) = \mathbf{V}^{(d)}(t) - \eta_{\mathbf{V}^{(d)}} \circ \nabla_{\mathbf{V}^{(d)}} \Phi \tag{4.3.24}$$

$$\mathbf{W}^{(e)}(t+1) = \mathbf{W}^{(e)}(t) - \eta_{\mathbf{W}^{(e)}} \circ \nabla_{\mathbf{W}^{(e)}} \Phi \tag{4.3.25}$$

Here, $\circ$ denotes the Hadamard product and the learning rates corresponding to $\mathbf{V}^{(d)}$ and $\mathbf{W}^{(e)}$ are denoted by matrices $\eta_{\mathbf{V}^{(d)}}$ and $\eta_{\mathbf{W}^{(e)}}$, the hyper-parameters of the model. The terms $\nabla_{\mathbf{V}^{(d)}}$ and $\nabla_{\mathbf{W}^{(e)}}$ denote gradient operators with respect to $\mathbf{V}^{(d)}$ and $\mathbf{W}^{(e)}$ respectively.

Now, calculating derivative of Φ with respect to $\mathbf{W}^{(e)}$, we get

$$\nabla_{\mathbf{W}^{(e)}}\Phi = \begin{cases} \frac{-1}{mn}\widehat{\mathbf{X}^{(e)}}^T\Theta_{\mathbf{W}^{(e)}} + \frac{\lambda}{n^2}(\prod_{q=1}^{s}\mathbf{V}^{(q)}\prod_{q'=1}^{s-e}\mathbf{W}^{(s-q'+1)})^T \\ (\mathbf{A}-\mathbf{I}), & \text{for } e=1 \\ \frac{-1}{mn}\widehat{\mathbf{X}^{(e)}}^T\Theta_{\mathbf{W}^{(e)}} + \frac{\lambda}{n^2}(\prod_{q=1}^{s}\mathbf{V}^{(q)}\prod_{q'=1}^{s-e}\mathbf{W}^{(s-q'+1)})^T \\ (\mathbf{A}-\mathbf{I})(\prod_{q'=s-e+2}^{s}\mathbf{W}^{(s-q'+1)})^T, & \text{for } 1<e<s \\ \frac{-1}{mn}\mathbf{X}^{(e)^T}\Theta_{\mathbf{W}^{(e)}} + \frac{\lambda}{n^2}(\prod_{q=1}^{s}\mathbf{V}^{(q)})^T \\ (\mathbf{A}-\mathbf{I})(\prod_{q'=s-e+2}^{s}\mathbf{W}^{(s-q'+1)})^T, & \text{for } e=s \end{cases} \tag{4.3.26}$$

where

$$\Theta_{\mathbf{W}^{(e)}} = \begin{cases} (\mathbf{X}^{(0)} - \widehat{\mathbf{X}^{(0)}}) \circ \widehat{\mathbf{X}^{(0)}} \circ (1 - \widehat{\mathbf{X}^{(0)}}), & \text{for } e=1 \\ (\Theta_{\mathbf{W}^{(e-1)}}\mathbf{W}^{(e-1)^T}) \circ \widehat{\mathbf{X}^{(e-1)}} \circ (1 - \widehat{\mathbf{X}^{(e-1)}}), & \text{otherwise} \end{cases} \tag{4.3.27}$$

Calculating derivative of Φ with respect to $\mathbf{V}^{(d)}$, we have

$$\nabla_{\mathbf{V}^{(d)}}\Phi = \frac{-1}{mn}\mathbf{X}^{(d-1)^T}\Theta_{\mathbf{V}^{(d)}} + \begin{cases} \frac{\lambda}{n^2}(\prod_{q=1}^{d-1}\mathbf{V}^{(q)})^T(\mathbf{A}-\mathbf{I}) \\ (\prod_{q'=1}^{s}\mathbf{W}^{(s-q'+1)})^T, & \text{for } d=s \\ \frac{\lambda}{n^2}(\prod_{q=1}^{d-1}\mathbf{V}^{(q)})^T(\mathbf{A}-\mathbf{I}) \\ (\prod_{q=d+1}^{s}\mathbf{V}^{(q)}\prod_{q'=1}^{s}\mathbf{W}^{(s-q'+1)})^T, & \text{for } 1<d<s \\ \frac{\lambda}{n^2}(\mathbf{A}-\mathbf{I}) \\ (\prod_{q=d+1}^{s}\mathbf{V}^{(q)}\prod_{q'=1}^{s}\mathbf{W}^{(s-q'+1)})^T, & \text{for } d=1 \end{cases} \tag{4.3.28}$$

where

$$\Theta_{\mathbf{V}^{(d)}} = \begin{cases} (\Theta_{\mathbf{W}^{(s)}}\mathbf{W}^{(s)^T}) \circ \mathbf{X}^{(s)} \circ (1 - \mathbf{X}^{(s)}), & \text{for } d=s \\ (\Theta_{\mathbf{V}^{(d+1)}}\mathbf{V}^{(d+1)^T}) \circ \mathbf{X}^{(d)} \circ (1 - \mathbf{X}^{(d)}), & \text{otherwise} \end{cases} \tag{4.3.29}$$

Using equations (4.3.26) - (4.3.29), we can write equations (4.3.24) and (4.3.25) as

$$\mathbf{V}^{(d)}(t+1) = \mathbf{V}^{(d)}(t) - \eta_{\mathbf{V}^{(d)}} \circ \begin{cases} \left(\frac{-1}{mn} \mathbf{X}^{(d-1)^T} \Theta_{\mathbf{V}^{(d)}} + \frac{\lambda}{n^2} (\mathbf{A} - \mathbf{I}) \right. \\ \left. \left(\prod_{q=d+1}^{s} \mathbf{V}^{(q)} \prod_{q'=1}^{s} \mathbf{W}^{(s-q'+1)} \right)^T \right), \text{ for } d = 1 \\ \left(\frac{-1}{mn} \mathbf{X}^{(d-1)^T} \Theta_{\mathbf{V}^{(d)}} + \frac{\lambda}{n^2} (\prod_{q=1}^{d-1} \mathbf{V}^{(q)})^T (\mathbf{A} - \mathbf{I}) \right. \\ \left. \left(\prod_{q=d+1}^{s} \mathbf{V}^{(q)} \prod_{q'=1}^{s} \mathbf{W}^{(s-q'+1)} \right)^T \right), \text{ for } 1 < d < s \\ \left(\frac{-1}{mn} \mathbf{X}^{(d-1)^T} \Theta_{\mathbf{V}^{(d)}} + \frac{\lambda}{n^2} (\prod_{q=1}^{d-1} \mathbf{V}^{(q)})^T (\mathbf{A} - \mathbf{I}) \right. \\ \left. \left(\prod_{q'=1}^{s} \mathbf{W}^{(s-q'+1)} \right)^T \right), \text{ for } d = s \end{cases} \tag{4.3.30}$$

and

$$\mathbf{W}^{(e)}(t+1) = \mathbf{W}^{(e)}(t) - \eta_{\mathbf{W}^{(e)}} \circ \begin{cases} \left(\frac{-1}{mn} \widehat{\mathbf{X}^{(e)}}^T \Theta_{\mathbf{W}^{(e)}} + \frac{\lambda}{n^2} (\prod_{q=1}^{s} \mathbf{V}^{(q)} \prod_{q'=1}^{s-e} \mathbf{W}^{(s-q'+1)})^T \right. \\ \left. (\mathbf{A} - \mathbf{I}) \right), \text{ for } e = 1 \\ \left(\frac{-1}{mn} \widehat{\mathbf{X}^{(e)}}^T \Theta_{\mathbf{W}^{(e)}} + \frac{\lambda}{n^2} (\prod_{q=1}^{s} \mathbf{V}^{(q)} \prod_{q'=1}^{s-e} \mathbf{W}^{(s-q'+1)})^T \right. \\ \left. (\mathbf{A} - \mathbf{I}) (\prod_{q'=s-e+2}^{s} \mathbf{W}^{(s-q'+1)})^T \right), \text{ for } 1 < e < s \\ \left(\frac{-1}{mn} \mathbf{X}^{(e)^T} \Theta_{\mathbf{W}^{(e)}} + \frac{\lambda}{n^2} (\prod_{q=1}^{s} \mathbf{V}^{(q)})^T \right. \\ \left. (\mathbf{A} - \mathbf{I}) (\prod_{q'=s-e+2}^{s} \mathbf{W}^{(s-q'+1)})^T \right), \text{ for } e = s \end{cases} \tag{4.3.31}$$

As per the definition of the model, the elements in $\mathbf{V}^{(d)}$ are unrestricted and the elements in $\mathbf{W}^{(e)}$ have to be non-negative. Following similar arguments given in the pretraining stage, the hyper-parameters λ, $\eta_{\mathbf{V}^{(d)}}$ and $\eta_{\mathbf{W}^{(e)}}$, for $1 \leq d \leq s$ and $1 \leq e \leq s$ is set to 0.1. Thus, the update rules for $\mathbf{V}^d$ and $\mathbf{W}^e$, for $1 \leq d \leq s$ and $1 \leq e \leq s$ are given by equations (4.3.30) and (4.3.31) respectively. Like pretraining, for fast convergence, we use momentum factors $\alpha_{\mathbf{V}^{(d)}} = 0.9$ and $\alpha_{\mathbf{W}^{(e)}} = 0.9$ and thus the equations (4.3.24)

and (4.3.25) becomes

$$\mathbf{V}^{(d)}(t+1) = \mathbf{V}^{(d)}(t) - \eta_{\mathbf{V}^{(d)}} \circ \nabla_{\mathbf{V}^{(d)}}\Phi + \alpha_{\mathbf{V}^{(d)}} \circ \nabla_{\mathbf{V}(t-1)^{(d)}}\Phi \tag{4.3.32}$$

$$\mathbf{W}^{(e)}(t+1) = \mathbf{W}^{(e)}(t) - \eta_{\mathbf{W}^{(e)}} \circ \nabla_{\mathbf{W}^{(e)}}\Phi + \alpha_{\mathbf{W}^{(e)}} \circ \nabla_{\mathbf{W}(t-1)^{(e)}}\Phi \tag{4.3.33}$$

4.4 Experimental Results, Analysis and Discussion

There are two parts to the presentation and justification of the performance of DN3MF. First, the degree to which DN3MF is able to maintain the local structure of data has been compared to assess the quality of its dimension reduction. Additionally, an analysis and decisiveness on the effectiveness of the low rank embedding in comparison to the original data has been made to justify the necessity of dimension reduction. Second, the discriminating power of the dimensionally reduced dataset has been investigated for downstream analyses, such as clustering and classification. It has also been investigated how statistically significant the outcomes of DN3MF are in comparison to other dimension reduction methods.

As stated earlier DN3MF is a two-stage model, namely, the pretraining stage and the stacking stage. The objective of DN3MF is to reduce an original n dimensional feature space to r dimensional transformed feature space. DN3MF has been designed to have s shallow networks. For the purpose of demonstration, we have considered $s = 2$. Hence, the pretraining stage consists of two shallow neural networks. The first one projects the original n dimensional feature space to r_1 dimensional feature space, where r_1 is defined as, $r_1 = n - \frac{n-r}{2}$. The second shallow network receives the r_1 dimensional feature vector as input and transforms to r dimensional feature space. In the stacking stage, the deep neural network architecture consists of the input layer, three hidden layers and the output layer. The number of nodes in these layers is n, r_1, r, r_1 and n respectively from input to output.

Xavier normal initialization technique [34] has been proven effective for neural networks using sigmoid type activation functions. For the pretraining stage of DN3MF, the elements of both the weight matrix $\mathbf{V}$ and $\mathbf{W}$ have been initialized using Xavier

initialization technique. The number of training epochs is decided dynamically. Training stops on reaching predefined stopping criteria based on the difference in the cost values of two consecutive epochs.

4.4.1 Quantifying the quality of low dimensional embedding

The ability of DN3MF to maintain the local structure of data has been studied using the trustworthiness metric and the efficiency of dimension reduction as measured by classification/cluster performance metrics has been compared with the original data in order to assess the quality of the low dimensional embedding.

4.4.1.1 Local structure preservation

The superiority of DN3MF over seven other dimension reduction techniques in maintaining the local structure of data after dimension reduction has been calculated and compared using the trustworthiness score. The spider/star plot illustrates the result of the same (Figure 4.3).

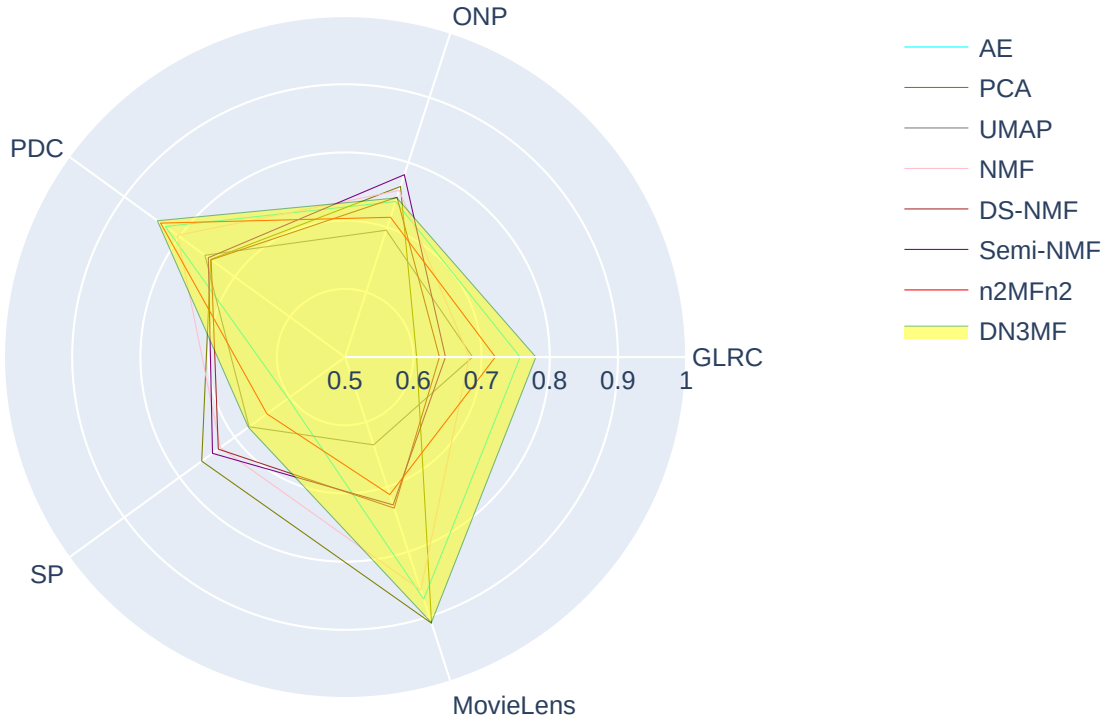

FIGURE 4.3: Trustworthiness scores of eight dimension reduction techniques including DN3MF.

TABLE 4.1: Sum of trustworthiness scores of eight dimension reduction techniques including DN3MF on five datasets.

Dimension reduction techniques	Sum of trustworthiness scores
AE	4.55550328220533
PCA	4.38647684863791
UMAP	4.13297187263103
NMF	4.50973409888627
DS-NMF	4.22833659492512
Semi-NMF	4.29147059452664
n^2MFn^2	4.34404888837219
DN3MF	**4.72960521062986**

Five datasets have been represented by five axes of the plot. A point on an axis represents the trustworthiness score of a dimension reduction technique for a given dataset. As a result, five points on five axes, representing five datasets, correspond to a dimension reduction method. These points can be thought of as the vertices of a polygon. Consequently, there are eight polygons for each of the eight dimension reduction strategies (Figure 4.3). The area of coverage by a polygon validates the effectiveness of the dimension reduction method across all datasets combined. A higher area indicates greater algorithmic performance. From the depiction, we can note that DN3MF has beaten other dimension reduction techniques for the GLRC, PDC and MovieLens datasets. For the ONP dataset, the trustworthiness score of DN3MF is better than most of the others. The region enclosed by the polygon related to DN3MF is depicted in Figure 4.3 using a shaded colour. By aggregating the individual trustworthiness ratings from each of the dimension reduction techniques for all five datasets, we are able to calculate the size of the polygon. Table 4.1 shows that DN3MF has the highest total trustworthiness score out of all the categories. As a result, we can conclude that DN3MF generates a low dimensional embedding of higher quality than the other dimension reduction techniques considered here.

4.4.1.2 Decision making: Comparison with the original data

The effectiveness of dimension reduction using DN3MF has been assessed by classifying and clustering both the original data and the low dimensional embedding generated by DN3MF and then quantifying the results using various classification/cluster

performance evaluators. This study shows that the low rank representation of the data improves its usability, which is one of the reasons why dimension reduction is required.

Classification

Figures 4.4-4.8 presents the performance of DN3MF and original data in terms of classification. For the GLRC (Figure 4.4) and PDC (Figure 4.6) datasets, DN3MF generated low rank embedding has outperformed the original data for all four classifiers in terms of all four metrics. For ONP and MovieLens datasets, for ACC, FS and MCC performance metrics, DN3MF has performed better than the original dataset for three out of four classification algorithms (Figures 4.5, 4.8). In terms of CKS, for the ONP dataset, the scoreline favouring DN3MF is three out of four and for the MovieLens dataset, the same count is two out of four. In the case of the SP dataset, the performance metric of original data is better than the low rank embedding produced by DN3MF on all occasions (Figure 4.7).

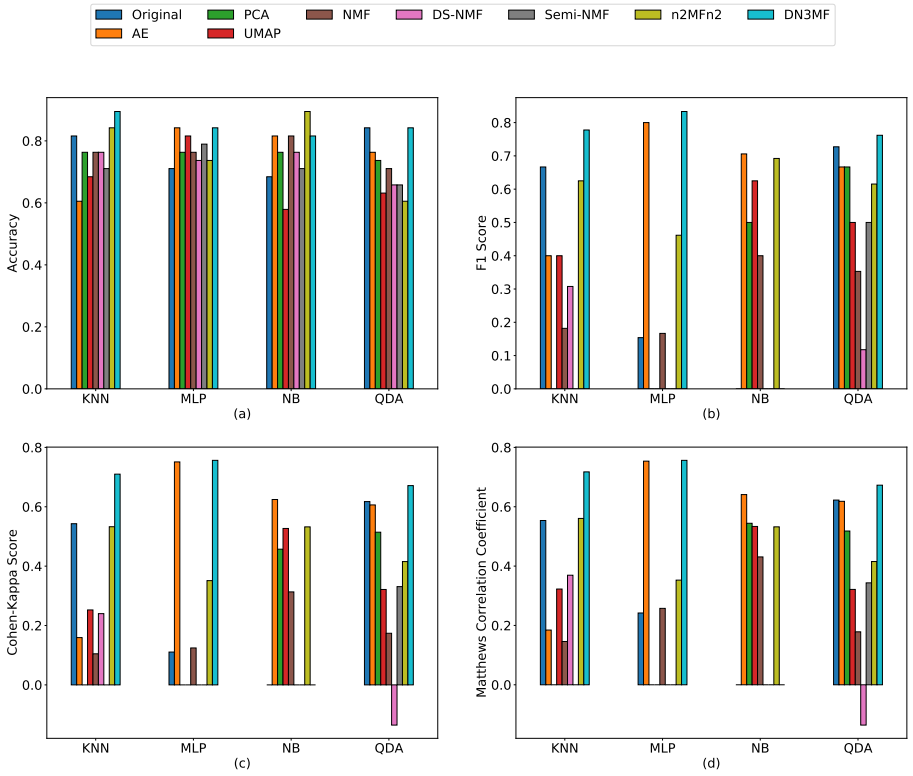

FIGURE 4.4: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the GLRC dataset by DN3MF and seven other dimension reduction techniques along with the original data.

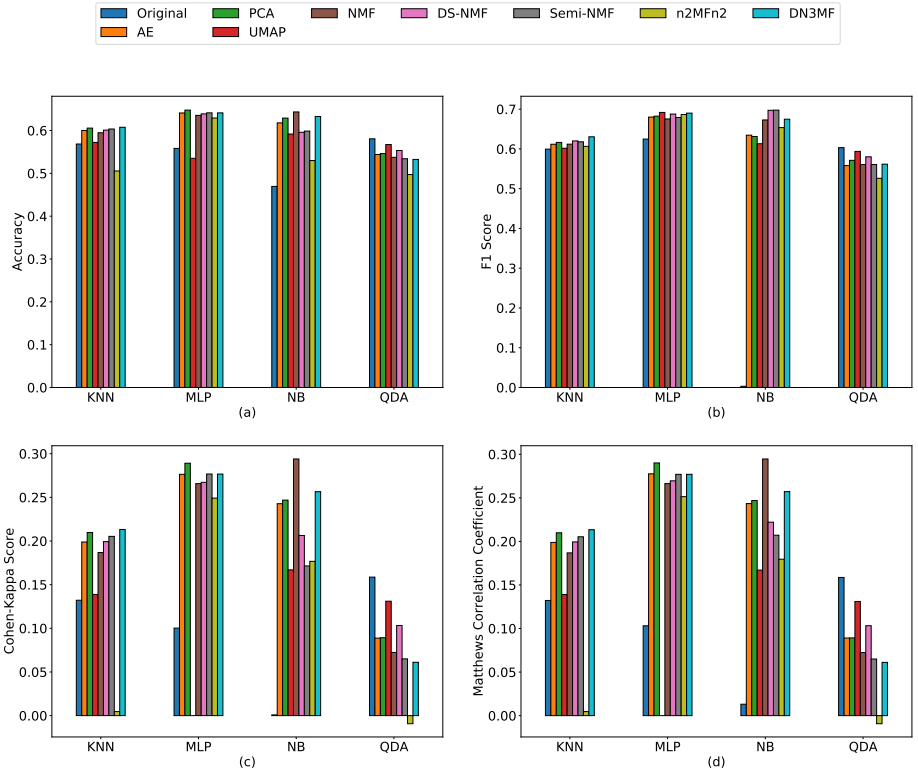

FIGURE 4.5: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the ONP dataset by DN3MF and seven other dimension reduction techniques along with the original data.

It is clear from these depictions that, with the exception of the SP dataset, majority of the time, DN3MF projected low rank embedding has outperformed the original in terms of classification. This supports the requirement for both dimension reduction and the capacity to generate low rank embeddings that preserve the fundamental properties of the data.

Clustering

The performance comparison of clustering done on the low dimensional embedding produced by DN3MF and the original data has been illustrated in Figures 4.9-4.13. For the ONP (Figure 4.10) dataset, for all four cluster validity indexes, DN3MF has performed better than the original data with respect to all four clustering algorithms. The same statistics for the GLRC (Figure 4.9) dataset are three out of four. For the ARI metric, the performance score is three out of four in favour of DN3MF for PDC (Figure 4.11), SP (Figure 4.12) and MovieLens (Figure 4.13) datasets. Similarly, for the NMI

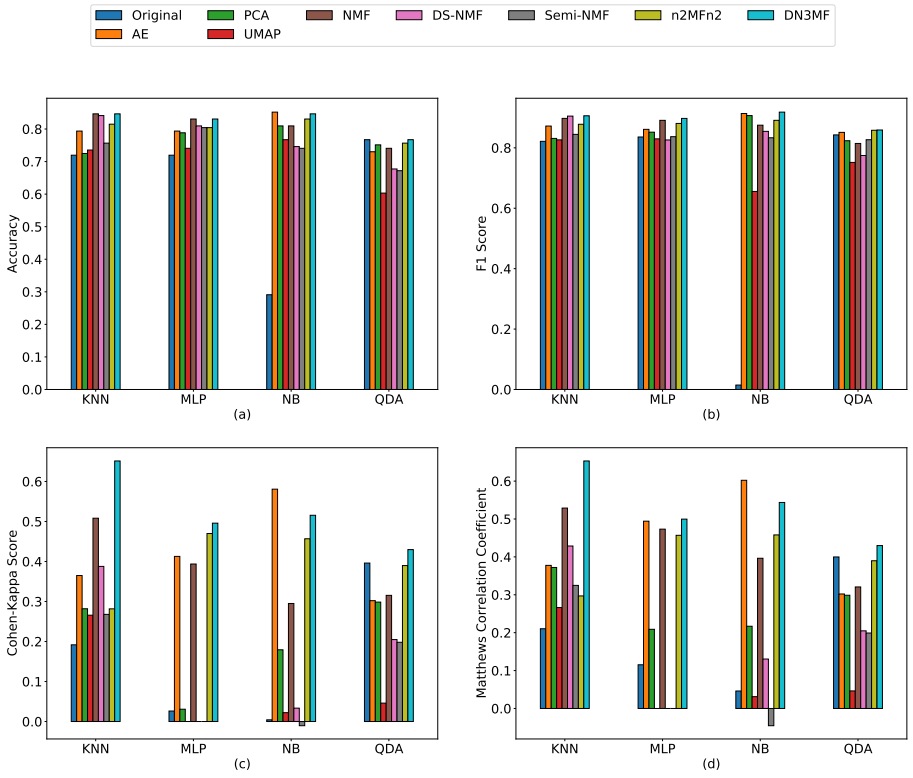

FIGURE 4.6: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the PDC dataset by DN3MF and seven other dimension reduction techniques along with the original data.

clustering validator, the count favouring DN3MF is four out of four for the same set of datasets. The same count with respect to the AMI metric is four, four and three out of four clustering algorithms and for the JI metric, the values are two, three and four respectively for the PDC, SP and MovieLens datasets.

As a result, it has been demonstrated that the low rank embedding produced by DN3MF performs significantly better in terms of maintaining the fundamental characteristics of the original data in terms of clustering. Therefore, dimension reduction is likewise necessary and warranted.

4.4.2 Downstream analyses and statistical significance: Comparison with other models

The efficiency of dimension reduction has been evaluated by conducting classification and clustering on the low dimensional embedding produced by DN3MF as well as

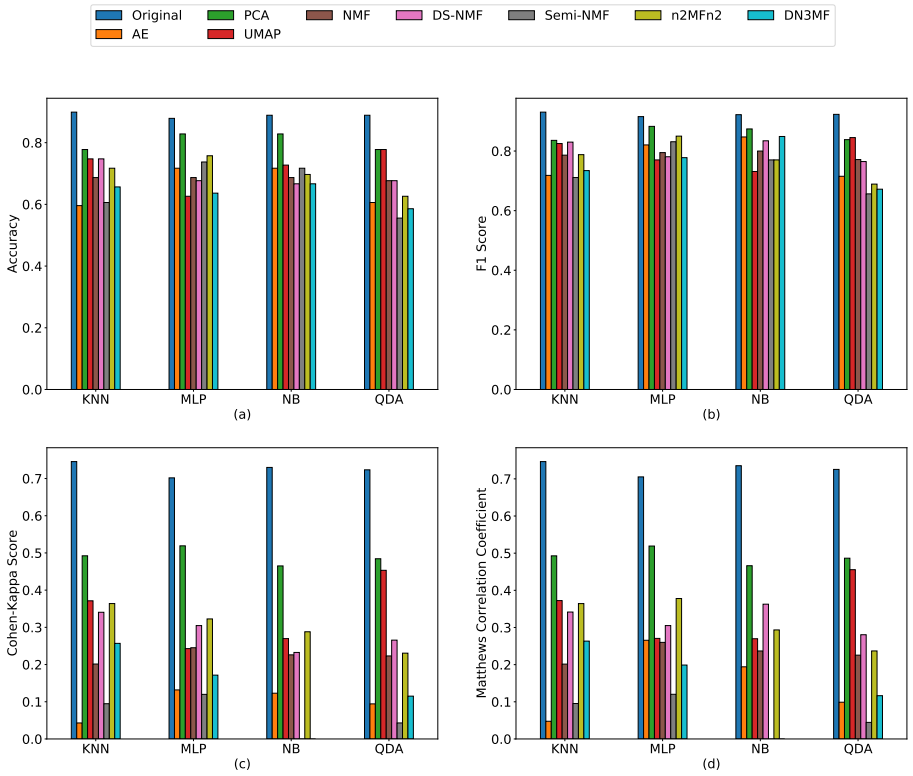

FIGURE 4.7: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the SP dataset by DN3MF and seven other dimension reduction techniques along with the original data.

that produced by the seven other dimension reduction methodologies. To quantify the same, a variety of measures assessing cluster and classification performances have been employed. In order to demonstrate the superiority of DN3MF over other dimension reduction algorithms in terms of generating output from an independent set of observations, pairwise p-values have also been computed. The statistical significance of the results is justified by a p-value below a certain threshold. In this case, 0.05 has been chosen as the threshold. In order to compare the performance of DN3MF with that of seven other dimension reduction techniques, seven p-values have been calculated for a dataset, a classification/clustering methodology and a classification/cluster validity index. There will be a total of $7 \times 4 = 28$ p-values for each classification/cluster validity index against each dataset because there are four classification/cluster methods. This portion of the experiment seeks to ascertain if dimension reduction using DN3MF is preferable when employing various classification and clustering techniques.

Classification

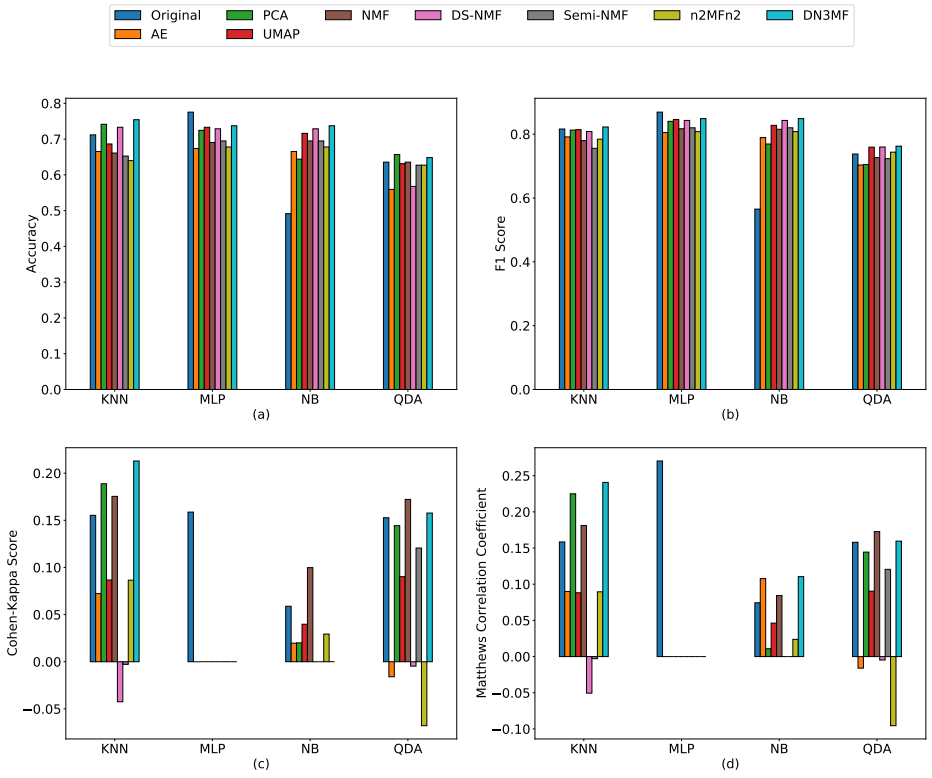

FIGURE 4.8: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the MovieLens dataset by DN3MF and seven other dimension reduction techniques along with the original data.

While working with the DN3MF model for classification, the outcome has been depicted by Figures 4.4-4.8. The summary of the count of statistically significant p-values with respect to DN3MF has also been presented in Table 4.2.

For four classification techniques, DN3MF has achieved the highest accuracy score on three out of four occasions for GLRC (Figure 4.4(a)), PDC (Figure 4.6(a)) and MovieLens (Figure 4.8(a)) datasets and once for the ONP (Figure 4.5(a)) dataset. The same statistics hold when the classification performance metric is MCC score (Figures 4.4(d), 4.5(d), 4.6(d) and 4.8(d)). DN3MF has surpassed the others in terms of the F1 score on PDC (Figure 4.6(b)) and MovieLens (Figure 4.8(b)) datasets using four out of four classification techniques, thrice for GLRC (Figure 4.4(b)) dataset and once for ONP (Figure 4.5(b)) dataset. When the Cohen-Kappa score is used as the classification performance indicator, the outcome favouring DN3MF is three out of four for the GLRC (Figure 4.4(c)) and PDC (Figure 4.6(c)) datasets. For the MovieLens (Figure 4.8(c)) dataset,

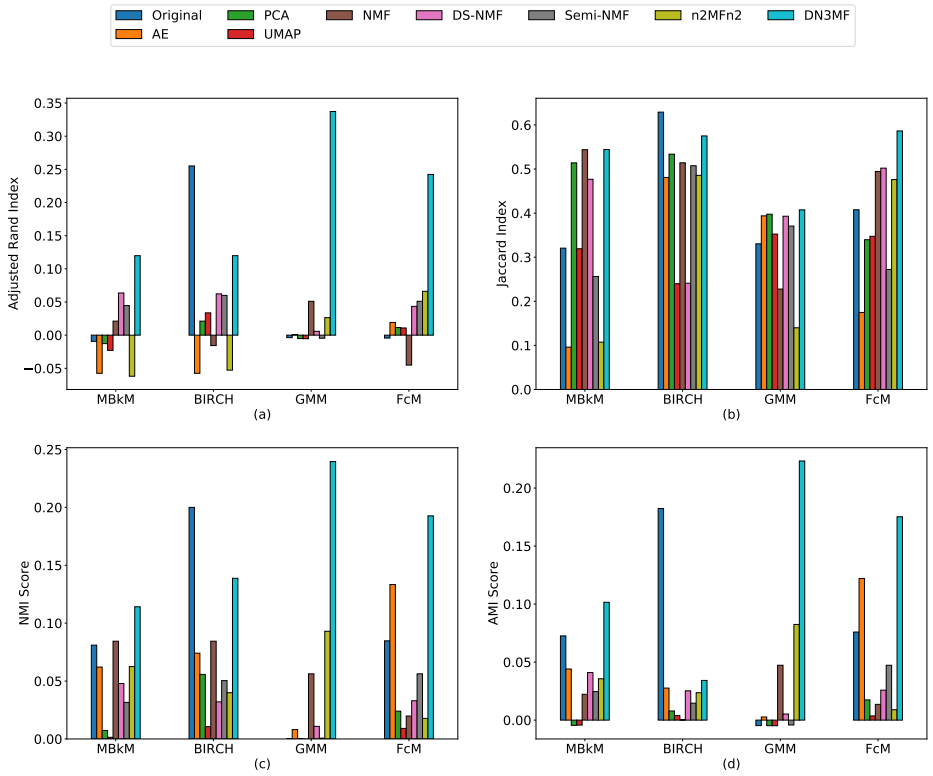

FIGURE 4.9: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the GLRC dataset by DN3MF and seven other dimension reduction techniques along with the original data.

the same value is two and for ONP (Figure 4.5(c)) that is one. DN3MF has failed on all four occasions in the case of the SP dataset.

As the above explanation makes evident, the transformed dataset using DN3MF has a higher accuracy score than the others in most cases for GLRC, PDC and MovieLens datasets with respect to four classifiers. For ONP and SP datasets the performance of DN3MF is comparable to the others. A model's accuracy indicates how frequently it is accurate. We have calculated the F1 score, or the harmonic mean of precision and recall, in addition to accuracy. Figures 4.4-4.8 demonstrate that, in the majority of scenarios, DN3MF has performed better than other models in terms of F1 score except ONP and SP datasets, where the performance of DN3MF is worthy of comparison. Therefore, the superiority of DN3MF is supported by its F1 score and accuracy. The visual representations demonstrate that DN3MF has either outperformed or has at per performance with respect to others in most cases for Cohen-Kappa scores. Higher Cohen-Kappa scores can infer that DN3MF is able to preserve and learn the intrinsic properties of the input. A higher MCC score denotes stronger agreement, indicating that the model

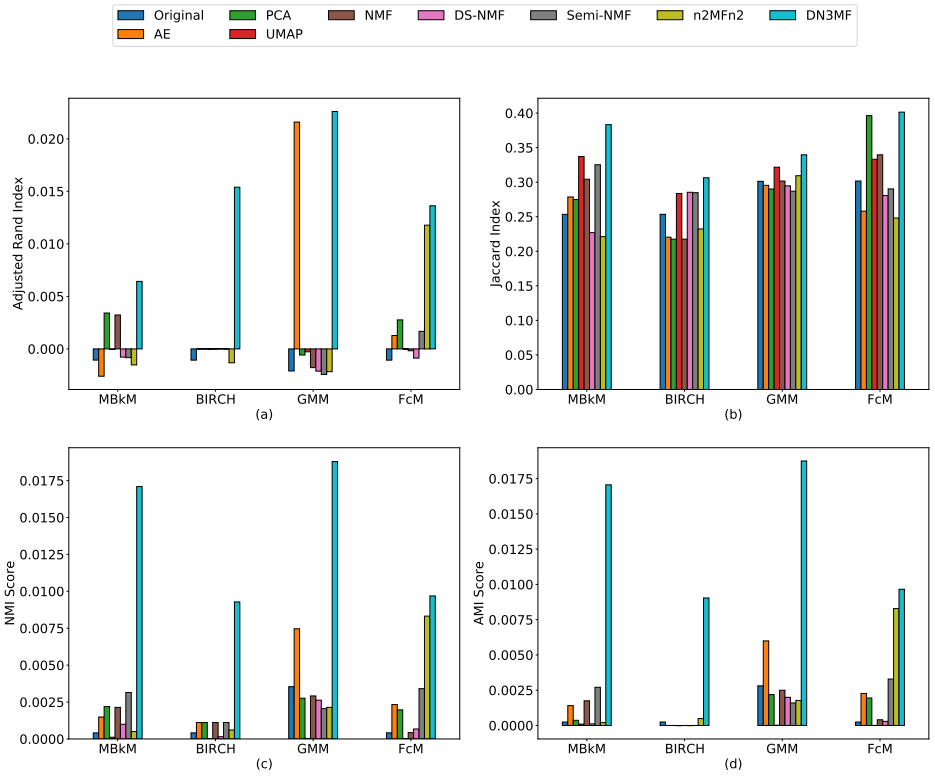

FIGURE 4.10: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the ONP dataset by DN3MF and seven other dimension reduction techniques along with the original data.

TABLE 4.2: The summary of the count (out of 28) of statistically significant p-values for each classification performance metric against each dataset with respect to DN3MF.

Dataset	ACC	FS	CKS	MCC
GLRC	21	24	25	25
ONP	19	15	20	20
PDC	25	25	25	25
SP	11	11	17	17
MovieLens	22	25	14	10

is also able to maintain the class properties of the original dataset in the transformed dataset. In terms of MCC score, DN3MF has done better than the other models, for GLRC, PDC and MovieLens datasets and for ONP and SP datasets the performances are comparable, as seen in Figures 4.4-4.8. The above explanation illustrates the comparable performance of DN3MF over other dimension reduction algorithms with respect to intrinsic property preservation criteria as well as statistical measures.

Table 4.2 presents the count of p-values less than the selected threshold (0.05) for each classification performance index against each dataset, out of a total of 28 p-values.

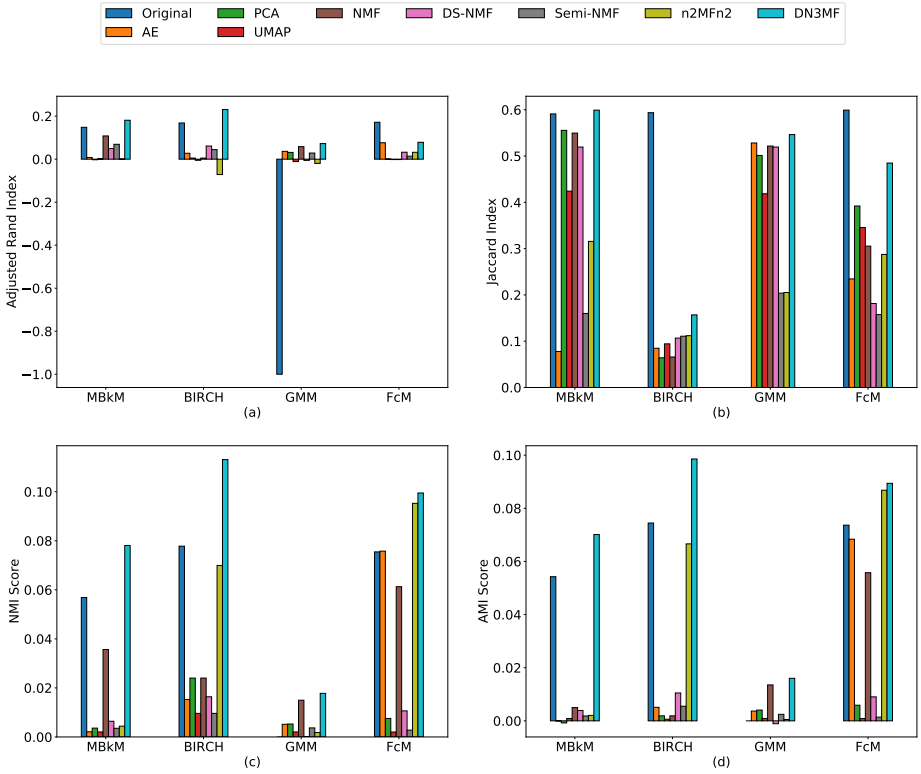

FIGURE 4.11: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the PDC dataset by DN3MF and seven other dimension reduction techniques along with the original data.

The aforementioned figures unequivocally demonstrate how good DN3MF's low rank embedding is compared to others.

Clustering

For clustering purposes with the DN3MF model, the Figures 4.9-4.13 present the outcome. Table 4.3 provides an overview of the count of statistically significant p-values for DN3MF for clustering.

DN3MF has achieved the highest performance score for the Adjusted Rand index for the GLRC (Figure 4.9(a)), ONP (Figure 4.10(a)) and PDC (Figure 4.11(a)) datasets. The same statistics hold for the other three cluster performance evaluators namely JI, NMI and AMI as well (Figures 4.9, 4.10, 4.11). For SP (Figure 4.12(a)) and MovieLens (Figure 4.13(a)) datasets, the count of supremacy favouring DN3MF with respect to ARI is two out of four and that is two and four for SP (Figure 4.12(b)) and MovieLens

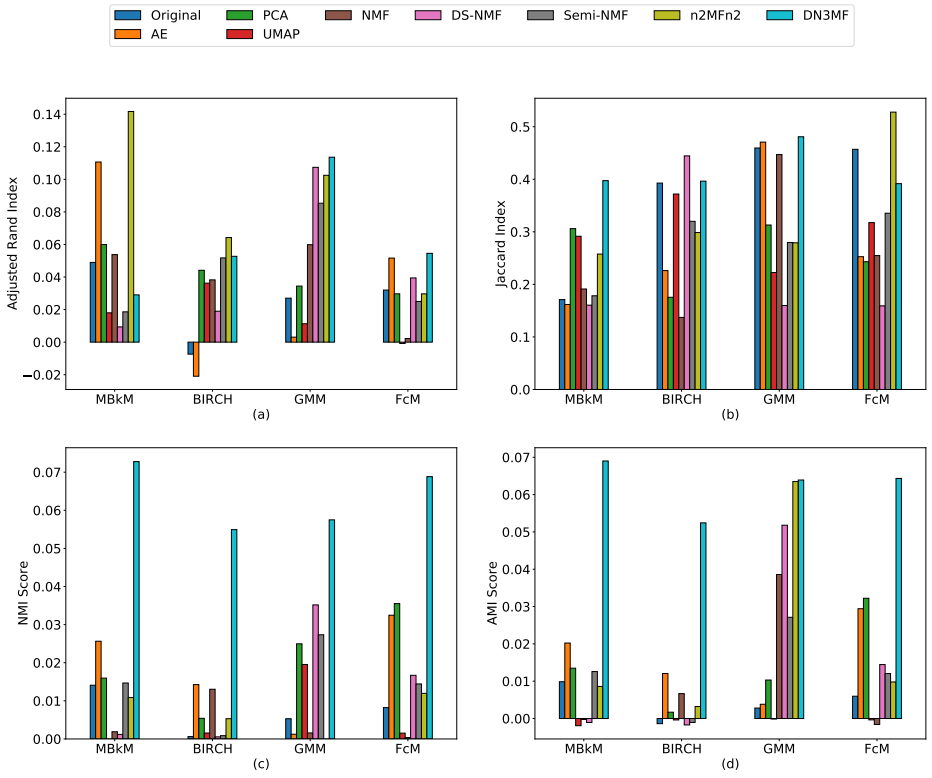

FIGURE 4.12: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the SP dataset by DN3MF and seven other dimension reduction techniques along with the original data.

(Figure 4.13(b)) datasets respectively with respect to Jaccard index. DN3MF has outperformed the others in four out of four clustering algorithms on the SP (Figure 4.12(c)) dataset when the cluster validity index is Normalized Mutual Information score and for the MovieLens (Figure 4.13(c)) dataset, the same count is three out of four. DN3MF projected transformed space has achieved the highest Adjusted Mutual Information score among the other dimension reduction techniques four times on the SP (Figure 4.12(d)) and MovieLens (Figure 4.13(d)) datasets.

Adjusted Rand Index compares two data clusters to see how comparable they are. In terms of ARI score, DN3MF has performed better than other dimension reduction methods across five datasets and four clustering algorithms, as shown in Figures 4.9-4.13. The Jaccard Index is used to compare two sets in terms of similarity. In terms of Jaccard Index, DN3MF has done better than the others. Therefore, one might claim that DN3MF has successfully learned the essential properties of the input and mapped them to a low rank representation. The Normalisation of Mutual Information score to scale the results in $[0, 1]$ is known as NMI. This measure is not adjusted for chance. In

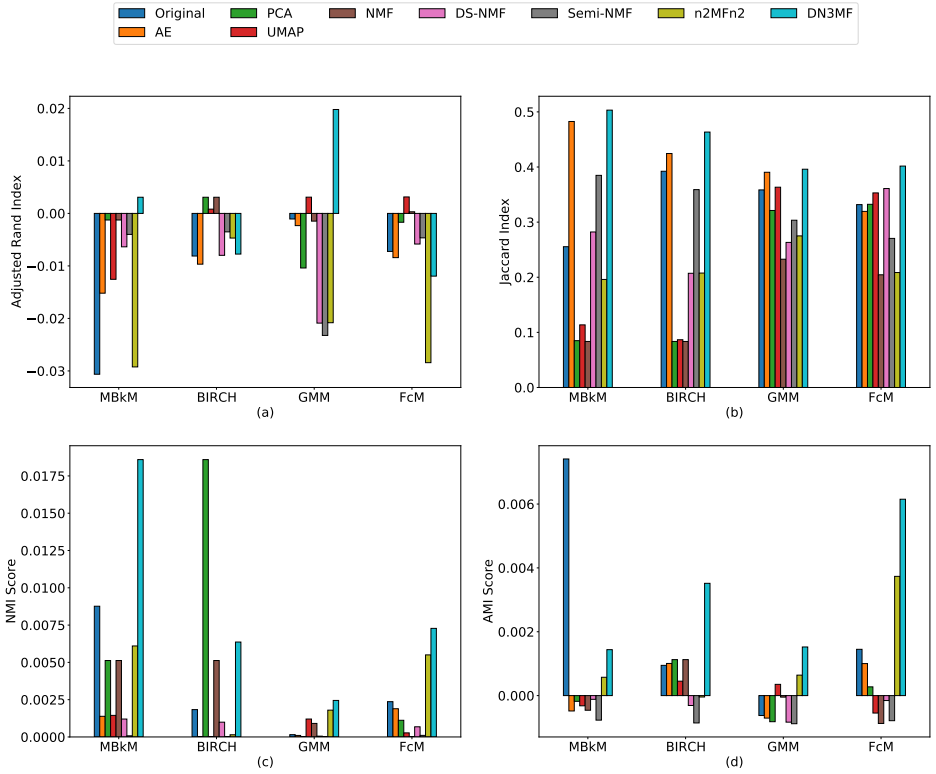

FIGURE 4.13: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the MovieLens dataset by DN3MF and seven other dimension reduction techniques along with the original data.

TABLE 4.3: The summary of the count (out of 28) of statistically significant p-values for each cluster performance metric against each dataset with respect to DN3MF.

Dataset	ARI	JI	NMI	AMI
GLRC	17	22	26	26
ONP	26	27	26	26
PDC	24	23	24	24
SP	11	06	09	10
MovieLens	20	24	24	23

contrast, the AMI score remains constant regardless of how the class or cluster label is arranged. Both NMI and AMI scores indicate that DN3MF has performed better than other dimension reduction techniques, as shown in Figures 4.9-4.13. The performance of DN3MF indicates that, in comparison to the other methods examined here, the low rank representation of the datasets using DN3MF has been able to preserve the intrinsic properties of the original data more successfully.

The count of p-values that are less than the chosen threshold (0.05), is shown in Table 4.3, out of a total of 28 p-values for each cluster validity index versus each dataset.

The previously given tally clearly shows how superior the low rank embedding produced by DN3MF is, over others.

4.4.3 Discussion

As mentioned in Section 3.4.3 of Chapter 3, we have used different types of datasets, classification techniques and clustering algorithms to justify the efficacy of dimension reduction by the proposed models.

DN3MF has been rigorously evaluated using the trustworthiness score metric across five datasets, comparing its performance with seven other dimension reduction techniques including n^2MFn^2. It has consistently demonstrated its effectiveness in preserving local data structures, outperforming or showing competitive performance with others in four out of five datasets.

Out of a total of 80 performance scores (for five datasets, four classification algorithms and four classification performance metrics), it is observed that on 55 instances, DN3MF projected datasets have outperformed the original data and in 41 cases it has outperformed other dimension reduction algorithms. For GLRC, PDC and MovieLens datasets, DN3MF has performed far better than the others in most of the cases. For ONP, DN3MF has recovered its performance over that of n^2MFn^2, and for SP dataset, the performance is still comparable to the others. Thus, it is undeniable that overall DN3MF is superior to the rest.

In addition to outperforming the original and other dimensionally reduced datasets produced by various dimension reduction methods for most of the cases, the low rank embeddings, generated by DN3MF for various datasets, have also been shown to be statistically significant based on the comparative p-values they have resulted in. For GLRC, the total number of statistically significant results related to DN3MF for all classifiers and classification metrics is 95 out of 112 $(4 \times 4 \times 7)$. The PDC, ONP, SP and MovieLens datasets have resulted in the corresponding counts of 74, 100, 56 and 71. Thus, it is proven that DN3MF is more effective than other dimension reduction algorithms in generating low dimensional embeddings that are statistically significant.

In a manner similar to classification, the competency of DN3MF has been demonstrated through the application of four clustering techniques and four cluster validity

metrics across five datasets. When comparing the clustering performance to the original data, DN3MF has reported a higher performance of 69 times out of 80. DN3MF's count of superiority over other dimension reduction algorithms is notable, coming in at 73 out of 80. Considering the previous discussion, it is evident that DN3MF has shown to be more effective than the other dimension reduction methods considered here in most cases.

Apart from demonstrating superior performance compared to the original and other dimensionally reduced datasets generated by the other dimension reduction techniques, the low rank embeddings produced by DN3MF for all the datasets have also been demonstrated to be statistically significant in terms of p-values. The overall number of DN3MF related statistically significant results for the GLRC dataset is 91 out of 112 $(4 \times 4 \times 7)$ for all clustering techniques and cluster validity indexes. The counts for the PDC, ONP, SP and MovieLens datasets are 105, 95, 36 and 91, respectively. Thus, it has been demonstrated that DN3MF produces low dimensional embeddings that are statistically significant and more effective than the other dimension reduction techniques.

Overall, the ability of DN3MF to produce statistically significant low-dimensional embeddings, regardless of dataset characteristics like sample size or number of attributes, highlights its robustness and broad applicability. It has consistently outperformed state-of-the-art techniques in both classification and clustering tasks, reinforcing its effectiveness in maintaining data structure fidelity. In conclusion, DN3MF emerges as a leading dimension reduction technique, adept at preserving local data structure integrity and delivering meaningful low-dimensional representations across diverse analytical scenarios.

4.5 Convergence Analysis

Here, we aim to establish the convergence of the proposed DN3MF model based on the experimental results. Figure 4.14 presents the convergence plots for five datasets, namely GLRC, ONP, PDC, SP and MovieLens respectively. Each row of Figure 4.14 consists of five plots. Each plot depicts the variation of the cost function Φ over iteration. Figures 4.14(a), (d), (g), (j) and (m) represent the convergence plot for the first

shallow neural network architecture which reduces the input feature space n to r_1 (Section 4.4). Figures 4.14(b), (e), (h), (k) and (n) represent the convergence plot for the second shallow neural network architecture which reduces r_1 dimensional feature space to r dimension. Finally, Figures 4.14(c), (f), (i), (l) and (o) represent the convergence plot for the deep model i.e., the stacking stage. The leftmost and middle column plots portray a one-step dimension reduction related cost analysis in the pretraining stage, whereas the rightmost column of plots represents the overall dimension reduction related cost analysis in the stacking stage. Figures 4.14(a) - 4.14(c) depict the cost versus iteration plots for the GLRC dataset with $r_1 = 398$ and $r = 97$. Similarly, Figures 4.14(d) - 4.14(f) represent the convergence plots for ONP dataset with $r_1 = 40$ and $r = 21$ and for PDC dataset with $r_1 = 472$ and $r = 192$, the convergence has been portrayed in Figures 4.14(g) - 4.14(i). The convergence plots for SP dataset with $r_1 = 23$ and $r = 15$ has been portrayed by Figures 4.14(j) - 4.14(l) and Figures 4.14(m) - 4.14(o) depicts the convergence plots for MovieLens dataset with $r_1 = 929$ and $r = 176$.

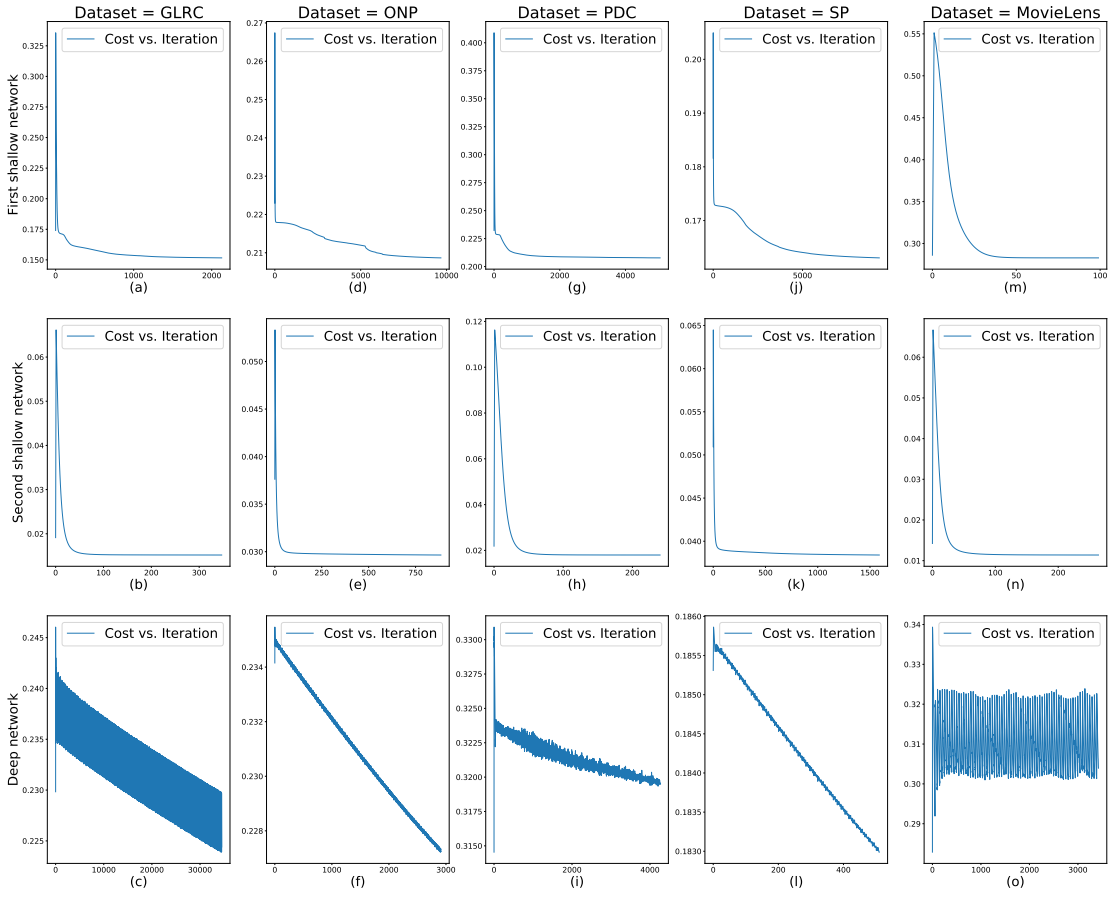

FIGURE 4.14: Cost vs. iteration plots of DN3MF for (a)-(c) GLRC, (d)-(f) ONP, (g)-(i) PDC, (j)-(l) SP and (m)-(o) MovieLens dataset for both pretraining and stacking stages of the model.

In the cost versus iteration plots for both shallow models, an initial spike in the error curve has been observed. This may be due to the fact that the initialization of weights was so near to optimal that the use of a fixed momentum factor had pushed the error up slightly for a brief number of iterations and after attaining some maximum, a steady decline in the error value is observed and finally the error curve has almost become a flat line. In the case of the stacked network, the small oscillation in the error value throughout the training may also be the effect of the fixed momentum factor. Overall, the declining nature of the cost over time clearly proves that both the shallow and the deep models are convergent in nature.

4.6 Analysis of Computational Complexity

The computational complexity of DN3MF has been derived in terms of number of operations performed. The upper bound of the complexity is expressed in terms of $\mathcal{O}$ notation. DN3MF is a two-stage model, viz., the pretraining stage and the stacking stage.

The computational complexity of the pertaining stage of DN3MF is the same as that of n^{2}MFn^2 and is given as $\mathcal{O}(t_p(mnr + n^2r))$ where, t_p denotes the number of epochs, (m, n) being the order of the input matrix $\mathbf{X}$ and r is the number of nodes in the slender layer. There are s successive shallow models in the pretraining stage of DN3MF. Hence the computational complexity of the pretraining stage of DN3MF is $\mathcal{O}(st_p(mnr_1 + n^2r_1))$, where, r_1 is the size of the slenderest layer of the first shallow model and $r_1 > r_2 > ... > r_s$.

In the stacking stage, the computational complexity of the forward pass can be similarly computed as $\mathcal{O}(mr_0r_1)$, where $r_0 = n$ and $r_0 > r_1 > ... > r_s$. The computational complexity of $\mathbf{A}$ (equation (4.3.21)) is $\mathcal{O}(r_0r_1r_0)$ i.e. $\mathcal{O}(n^2r_1)$. Therefore, the computation of Φ (equation (4.3.20)) involves $\mathcal{O}(mn + n^2r_1)$ operations. The upper bound of the number of operations in the backward pass is $\mathcal{O}(mnr_1 + n^2r_1)$. Thus, the overall computational complexity for an epoch of the stacking stage of DN3MF is $\mathcal{O}(mnr_1 + n^2r_1)$. For t_s epochs, the complexity is $\mathcal{O}(t_s(mnr_1 + n^2r_1))$. Hence, the computational complexity of DN3MF becomes $\mathcal{O}(st_p(mnr_1 + n^2r_1) + t_s(mnr_1 + n^2r_1))$, i.e., $\mathcal{O}((st_p + t_s)(m + n)nr_1)$.

4.7 Conclusions

There are quite a few techniques available to dimensionally reduce a huge dataset. In this chapter, we have developed DN3MF unifying the advantages of NMF and deep learning, towards dimension reduction. DN3MF is a two-stage model, namely, pretraining and stacking. The pretraining stage employs a sequence of shallow neural networks, whereas the stacking stage employs a deep neural network. Again, each of these stages is divided into two phases, viz., deconstruction and reconstruction. The naming of the phases resembles the objective of the model. The deep neural network architecture is an MDMR architecture in the sense that there is more than one deconstruction and more than one reconstruction layer. The performance of DN3MF is essentially a combination of the two-stage training and the neural network's ability to learn hierarchically and optimise parameters.

The constraints of DN3MF have been met through both the design and learning procedure of the model. The objective function has been developed employing an innovative regularizer, ensuring the best possible approximation of the input data matrix. Furthermore, the regularising parameter has been customised such that it has a controlled effect on the regularizer when DN3MF attempts to regenerate the input. The novel objective function helps the model to keep its focus on the generation of the best possible meaningful approximation of the input. The DN3MF learning process has been performed employing an adaptive learning approach.

The superiority of DN3MF has been demonstrated over seven other dimension reduction techniques including n^2MFn^2, through a considerable amount of experimentation. The meaningfulness of the low rank approximation produced by DN3MF over that of others has been judged with the help of the trustworthiness score. It can be observed that DN3MF has performed better than or at least at par with others for four out of five datasets and for the remaining one DN3MF performance is competitive. Overall the trustworthiness scores justify DN3MF. Also, the discriminating ability of the low dimensional embedding has been properly analysed employing both classification and clustering on five popular datasets. A total of 4 classification algorithms, 4 classification performance metrics and 4 clustering algorithms, 4 cluster validity indexes have been used in the course of experimentation. The hefty result set has been properly supported in terms of its statistical significance for better understanding. When compared

to the original dataset in terms of justifying the need for dimension reduction, for classification DN3MF is not so successful for one out of five datasets however for clustering DN3MF is successful. In this case, with respect to the previous model (n^2MFn^2) performance, the overall performance of DN3MF is better. When the performance of DN3MF is judged over other dimension reduction techniques in terms of classification, DN3MF has better performance for three out of four datasets, competitive performance in one and not-so-good performance for the remaining one. For clustering, overall DN3MF performance is better than others. In contrast with the previous model, DN3MF has much better performance. Thus, the results strongly indicate the superiority of DN3MF over the original dataset and other dimension reduction techniques in terms of both statistical and intrinsic property preservation principles. The convergence of DN3MF has been presented in terms of experimentation. An assessment of the computational complexity of DN3MF has also been demonstrated.

DN3MF has established itself better than the shallow model (n^2MFn^2) described in the previous chapter, in a number of ways. The overall aim of the thesis is to dimensionally reduce a huge dataset to overcome the curse of dimensionality problem. Conventional NMF produces two unique non-negative factor matrices for an input matrix. The slenderest layer output ($\mathbf{B}$) of DN3MF is one of the factors. The other factor is basically a combination of weights and activation functions associated with the layers above the slenderest layer up to the output layer of DN3MF. As DN3MF follows MDMR architecture, there is more than one reconstruction layer above the slenderest layer of the model, which is difficult to compute directly. On the other hand, as we have $\mathbf{B}$ and $\hat{\mathbf{X}}$ with us, it is easier to back-calculate the second factor matrix. However, the input matrix $\mathbf{X}$ is not necessarily a square matrix. Hence, $\hat{\mathbf{X}}$ and $\mathbf{B}$ are not always square. Non-square matrices are not invertible. In these cases, we need to compute the pseudo-inverse, which is not unique. Thus, multiple reconstruction layers fail to compute a unique $\mathbf{W}$, which is, in this case, the second factor matrix. This is a serious problem with respect to the overall aim of the model design. The following chapter will mainly focus on this issue.

Chapter 5

Multiple Deconstruction Single Reconstruction Deep Neural Network Model for Non-negative Matrix Factorization (MDSR-NMF)

5.1 Introduction

In the previous chapter, a deep learning based NMF model, named DN3MF, has been designed and established through rigorous experimentation. However, serious shortcomings of the model, i.e., failing to produce two unique factor matrices, have also been highlighted. In this chapter, we mainly try to solve the same through the design of a novel deep learning framework.

We have combined the benefits of the conventional iterative approach with the current deep learning framework in this research work. Hierarchical learning is a significant benefit of deep learning. As a result, data representation is learned layer by layer. For low rank approximation of the data matrix using NMF, we have developed a deep learning model, named Multiple Deconstruction Single Reconstruction Deep Neural Network Model for Non-negative Matrix Factorization (MDSR-NMF) [26]. Pretraining and stacking are the two stages of the model.

The pretraining stage of the model is accomplished by means of a shallow neural network architecture, comprising input, output and a single hidden layer [24, 25]. For the stacking stage, a unique deep learning architecture has been devised. The design of this stacking stage distinguishes itself from other current deep learning architectures. The number of layers between the input and the bottleneck layer in a traditional Autoencoder is the same as the number of layers between the bottleneck and the output layer. The number of layers leading from the input layer to the slenderest layer of the framework in the MDSR-NMF stacking stage architecture differs from the number of layers between the slenderest layer and the output layer of the framework. There is no restriction on the number of layers between the input and the slenderest layer of the MDSR-NMF framework. On the other hand, the slenderest layer connects directly to the output layer. As a result, the model is known as Multiple Deconstruction Single Reconstruction Deep Neural Network Model. This novel architecture ensures a unique pair of factor matrices of the reconstructed input matrix. Thus, MDSR-NMF simulates the factorization behaviour of traditional NMF techniques.

In MDSR-NMF, two-stage strategy, i.e., pretraining followed by model fine tuning, enhances the architecture's resilience. A shallow neural network architecture is used for pretraining, while a deep neural network architecture is used for stacking. Both architectures are divided into two phases, which are deconstruction and reconstruction. Various architectural restrictions have been handled during the design of the model. A Sigmoid function maps any data point inside the range $(0, 1)$, solving the data loss problem. The sigmoid activation function has been modified to suit the architecture's non-negativity criterion. The exploding or vanishing gradient problem has been addressed by keeping the variance of activation constant across all layers and to do so Xavier initialization [34] technique has been employed to initialise the weights of the neural network framework.

L1 regularization/Lasso regularisation has been used to design the objective function, which decreases the chance of over-fitting. The regularizer has been designed in such a manner that assists the model in achieving the closest approximation of the input matrix. The learning rules of the architecture have been derived while keeping the restrictions of the model in mind. Momentum factor has been utilised to accelerate learning. Using both classification and clustering, the superiority of MDSR-NMF over eight well-known dimension reduction techniques has been established. A total of five

datasets have been used to justify the efficacy of the model. The test with respect to the preservation of the local structure of data in low rank embedding has also been performed. The need for dimension reduction over the original has also been experimented with.

This is how the remainder of the chapter is structured. The inspiration behind MDSR-NMF's design and learning is explained in Section 5.2. In Section 5.3, the specific design and derivation of the corresponding learning rules are provided. Following the experimentation process outlined in Chapter 2, the findings are then presented in Section 5.4 along with a sufficient analysis. Sections 5.5 and 5.6 have been used to provide the convergence and the computational complexity analysis of MDSR-NMF. Section 5.7 finally brings the chapter to a conclusion.

5.2 Motivation behind Architecture and Learning

MDSR-NMF is a novel deep learning model developed for the task of NMF. There are two stages of MDSR-NMF, namely, pretraining and stacking. A shallow neural network architecture has been used for pretraining, while a deep neural network architecture has been employed for stacking. Each of these stages of the model is divided into two phases, viz., deconstruction and reconstruction. The input to the neural network is transformed into latent space during the deconstruction phase and the network attempts to recreate the input from its low rank representation during the reconstruction phase. In the stacking stage of MDSR-NMF, there are multiple deconstruction layers leading towards the slenderest layer of the network from the input but there is only one reconstruction layer between the slenderest and the output layers of the network. Hence, the model follows Multiple Deconstruction Single Reconstruction (MDSR) architecture.

Motivation behind the design of the deep architecture

The deep neural network architecture of the stacking stage comprises multiple deconstruction layers and a single reconstruction layer. The novel deep neural network architecture tries to learn the low rank representation of the input data in a stepwise manner

in the deconstruction phase of the model. Whereas in the reconstruction phase, it directly tries to reconstruct the given input from the latent representation in a single step. The aim of the model is to factorise a given input matrix into two constituent factor matrices using Non-negative Matrix Factorization (NMF) aiming towards dimension reduction. The input matrix and both factor matrices adhere to the non-negativity criterion. In MDSR-NMF, the input matrix $\mathbf{X}$ is factored to produce two factors namely, $\mathbf{B}$ and $\mathbf{W}$, where $\mathbf{B}$ is the output of the slenderest layer of the architecture and $\mathbf{W}$ is the weight matrix connecting the slenderest layer and the output layer of the model. These two factors are used to compute the regenerated matrix $\hat{\mathbf{X}}$ as $\hat{\mathbf{X}} = \mathbf{BW}$. Thus, we need to produce a unique pair of $\mathbf{B}$ and $\mathbf{W}$ as the output of the model.

The model tries to learn the low rank representation of the input in a cumulative approach using the advantage of hierarchical learning facilitated by the deep neural network architecture. Hence, there is more than one layer in the deconstruction phase of the model and a unique $\mathbf{B}$ is available as the output at the end of the deconstruction phase. The model tries to simulate the factorization behaviour of the traditional NMF technique by taking only one layer in the deconstruction phase of the model. If there were multiple layers in the reconstruction phase then finding a unique value of $\mathbf{W}$ with respect to $\mathbf{X}$ and $\mathbf{B}$ is not possible. Let there be k layers in the reconstruction phase. The weight matrices are denoted as $\mathbf{W}_1, \mathbf{W}_2, ..., \mathbf{W}_k$ respectively. Thus, ignoring the activation functions for the sake of simplicity, we can write

$$\hat{\mathbf{X}} = \mathbf{B}\mathbf{W}_1\mathbf{W}_2...\mathbf{W}_k \tag{5.2.1}$$

Now, from equation (5.2.1) we need to find a unique $\mathbf{W}$, which can be defined as $\mathbf{W} = \prod_{i=1}^{k} \mathbf{W}_i$. Thus, we can write

$$\mathbf{W} = \mathbf{B}^{-1}\hat{\mathbf{X}} \tag{5.2.2}$$

The input matrix $\mathbf{X}$ is not necessarily a square matrix. Hence, $\hat{\mathbf{X}}$ and $\mathbf{B}$ are also not square in nature. Non-square matrices are not invertible. In these cases, we need to compute the pseudo-inverse, which is not unique in nature. Thus, multiple reconstruction layers fail to compute a unique $\mathbf{W}$, whereas a single reconstruction layer ensures a unique pair of $\mathbf{B}$ and $\mathbf{W}$ as the output. Thus, the deep architecture has been designed in such a manner.

5.3 MDSR-NMF

Deep Neural Network Model for Non-negative Matrix Factorization. The architecture has been designed in this manner to simulate the factorization behaviour of the traditional NMF technique. By taking only one layer in the reconstruction step of the model, the model is able to synthesize a unique pair of constituent non-negative factor matrices. Multiple reconstruction layers fail to produce a unique pair of factor matrices because in that case, we need to compute pseudo-inverse and pseudo-inverse is not unique in nature. This section describes the architecture and learning algorithms of the stages of MDSR-NMF that we have developed in this article.

5.3.1 Pretraining stage

The first stage of MDSR-NMF is performed using a shallow neural network architecture. The main purpose of this stage is to find the initial values of the weights for the stacking stage of the model. The pretraining stage architecture and learning of MDSR-NMF is the same as that of DN3MF as described in Chapter 4 Section 4.3.1.1.

Let there be $s+1$ shallow models in the pretraining stage of MDSR-NMF. The first shallow network accepts $\mathbf{X} = \mathbf{X}^{(0)}$ as input, where, $\mathbf{X}_{m\times n}$ is the outcome of the processed given data matrix $\mathbf{U}_{m\times n'}$, eliminating non-negativity following the procedures outlined in Chapter 2 Section 2.3. The output $\mathbf{X}^{(1)} = [x^{(1)}_{pi_1}]_{m\times r_1}$ of the slender layer of this first shallow model, is the input to the second shallow model. Similarly, the slender layer output of the second shallow model serves as the input to the third shallow model, and so on. As a result, the slender layer output of the s^{th} shallow model is $\mathbf{X}^{(s)} = [x^{(s)}_{pi_s}]_{m\times r_s}$. The required reduced dimension is denoted by r_s. The only restriction on r_s is that it should be less than n', there is no restriction of r_s with respect to the number of samples m (Section 3.3.1, Chapter 3). These s shallow networks are trained one after another. The target outputs of the nodes in the reconstruction layers of these s shallow networks for this purpose are $\mathbf{X}^{(0)}$, $\mathbf{X}^{(1)}$, ..., $\mathbf{X}^{(s-1)}$. Following the properties of the shallow architectures, the elements in the weight matrices $\mathbf{V}^{(1)}, \mathbf{V}^{(2)}, ..., \mathbf{V}^{(s)}$ are unrestricted, while that in the weight matrices $\mathbf{W}^{(1)}, \mathbf{W}^{(2)}, ..., \mathbf{W}^{(s)}$ are non-negative.

As discussed above, the first s consecutive shallow models among $s+1$ shallow models of the pretraining stage of MDSR-NMF, reduce the given n dimensional input

to the required $r = r_s$ dimensional feature space in a step-by-step fashion. The remaining one shallow model takes $\mathbf{X}^{(0)}$ as input and produces $\widehat{\mathbf{X}^{(0)}}$ as output. The number of nodes in the slenderest layer of $(s+1)^{th}$ shallow model is r_s, the desired reduced feature space dimension. The weight matrices of this $(s+1)^{th}$ shallow network are denoted by $\mathbf{V}$ and $\mathbf{W}$. The elements of $\mathbf{V}$ are unconstrained and that of $\mathbf{W}$ are non-negative. As a result, this $(s+1)^{th}$ shallow network reduces the input $\mathbf{X}^{(0)}$ to $\mathbf{X}^{(s)}$ in order to regenerate the input. Therefore, the consecutive s shallow models and this $(s+1)^{th}$ shallow model both reduce the n dimensional input to r dimensional lower rank representation separately. With this, the first stage of MDSR-NMF is completed.

5.3.2 Stacking stage

The second stage of MDSR-NMF is performed using a deep neural network architecture. The initialization of the weight values of this model is done using the learned weights of the pre-trained shallow models. The job of this stage is to fine-tune the weights of the model. The following sections describe the motivation behind the design of the deep network and the architecture of the deep model followed by its learning.

5.3.2.1 Architecture of the deep model

The stacking stage of the model employs $(s+1)$ pre-trained shallow neural network models being stacked together to produce a deeper network architecture. As previously explained, the input data matrix $\mathbf{U} = [u_{pi}]_{m \times n'}$ is processed to generate the matrix $\mathbf{X}^{(0)} = [x^{(0)}_{pi_0}]_{m \times r_0}$, where $r_0 = n$. The deep neural network model receives $\mathbf{X}^{(0)}$ as input. The model employs the same set of activation functions as described in Chapter 4 Section 4.3.1.1, namely, the identity function and a modified version of the sigmoid activation function σ. Figure 5.1 depicts the architecture of the stacked model.

The task of the model may be broken down into two phases: deconstruction and reconstruction. The input data is interpreted in lower dimensional representation in the deconstruction phase. The goal of the reconstruction phase is to regenerate the input from this latent representation. The design consists of an input layer, s deconstruction layers and a single reconstruction layer on top. Hence, the design is known

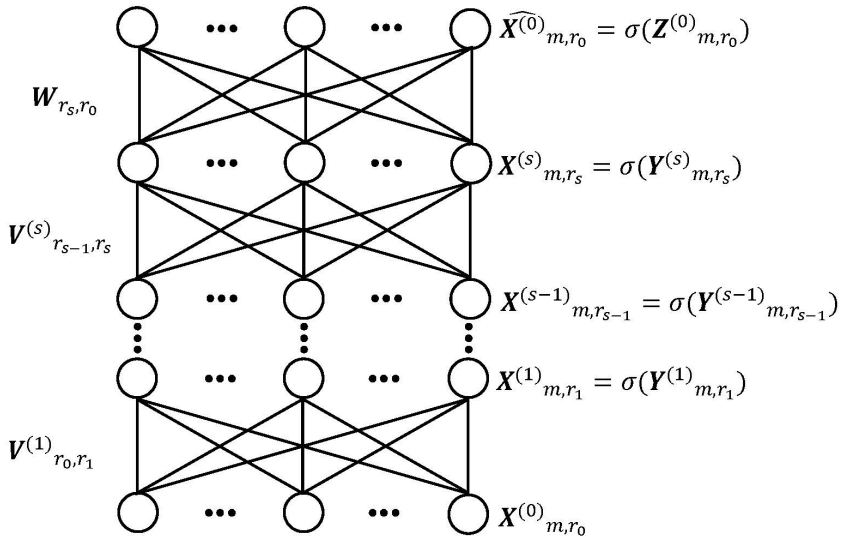

FIGURE 5.1: Stacking stage architecture of MDSR-NMF.

as multiple deconstruction single reconstruction deep learning architecture. The input layer, which has $r_0 = n$ nodes, gets $\mathbf{X}^{(0)} = [x^{(0)}_{pi_0}]_{m\times r_0}$ as input and produces the same output as its input using the identity function as the layer's activation function. The first deconstruction layer with r_1 nodes follows the input layer, where $r_1 < r_0$. The weight matrix between the input and the first deconstruction layer is denoted by $\mathbf{V}^{(1)} = [v^{(1)}_{i_0 i_1}]_{r_0\times r_1}$. The learned $\mathbf{V}^{(1)}$ in the first pretrained shallow model initialises $\mathbf{V}^{(1)}$. The first deconstruction layer produces output $\mathbf{X}^{(1)} = [x^{(1)}_{pi_1}]_{m\times r_1}$, where $\mathbf{X}^{(1)} = \sigma(\mathbf{Y}^{(1)})$ and $\mathbf{Y}^{(1)} = [y^{(1)}_{pi_1}]_{m\times r_1}$. $\mathbf{Y}^{(1)}$ is computed as $\mathbf{Y}^{(1)} = \mathbf{X}^{(0)}\mathbf{V}^{(1)}$. Now, $\mathbf{X}^{(1)}$ serves as the input to the second deconstruction layer, which operates in the same manner as the previous layer. As a result, for d^{th} $(1 \leq d \leq s)$ deconstruction layer,

$$\mathbf{X}^{(d)} = \sigma(\mathbf{Y}^{(d)}) \tag{5.3.1}$$

where $\mathbf{X}^{(d)} = [x^{(d)}_{pi_d}]_{m\times r_d}$ is the output of d^{th} deconstruction layer and

$$\mathbf{Y}^{(d)} = \mathbf{X}^{(d-1)}\mathbf{V}^{(d)} \tag{5.3.2}$$

Here, $\mathbf{V}^{(d)}$ is the weight matrix between $(d-1)^{th}$ and d^{th} deconstruction layers of the model.

The trained $\mathbf{V}^{(d)}$ in d^{th} pretrained shallow model initialises the weight matrix $\mathbf{V}^{(d)}$. It should be observed that $r^{d-1} > r^d$. As a result, for $d = s$, the last deconstruction layer

is the slenderest layer of the model in terms of node count. This slenderest layer serves as the bottleneck layer of the model, producing $\mathbf{X}^{(s)} = [x^{(s)}_{pi_s}]_{m\times r_s}$ as output. As a result, the model achieves the desired reduced dimension r_s, which is the number of nodes in this layer. This slenderest layer concludes the deconstruction phase and marks the beginning of the reconstruction phase.

The slenderest layer is followed by the single reconstruction layer of the model, with $\mathbf{X}^{(s)}$ serving as the input and $\mathbf{W} = [w_{j_s j_0}]_{r_s\times r_0}$ serving as the weight matrix between the slenderest and output layers. The learned $\mathbf{W}$ in the $(s+1)^{th}$ pretrained shallow model initialises $\mathbf{W}$. The reconstruction layer produces output $\widehat{\mathbf{X}^{(0)}} = [\widehat{x^{(0)}_{pj_0}}]_{m\times r_0}$ and therefore, the stacking stage architecture attempts to reproduce the input to the model, i.e., $\mathbf{X}^{(0)}$. In this case, $\widehat{\mathbf{X}^{(0)}} = \sigma(\mathbf{Z}^{(0)})$ and $\mathbf{Z}^{(0)} = [z^{(0)}_{pj_0}]_{m\times r_0}$. $\mathbf{Z}^{(0)}$ is computed as $\mathbf{Z}^{(0)} = \mathbf{X}^{(s)}\mathbf{W}$. As previously stated, the elements in the weight matrices $\mathbf{V}^{(1)}, \mathbf{V}^{(2)}, ..., \mathbf{V}^{(s)}$ are unconstrained and that of $\mathbf{W}$ are non-negative. The stacking model is fine-tuned keeping these non-negativity limitations in mind. One of the two non-negative components of the regenerated input matrix $\widehat{\mathbf{X}^{(0)}}$ is the slenderest layer output $\mathbf{X}^{(s)}$. The other non-negative component of $\widehat{\mathbf{X}^{(0)}}$ is the combination of the weight matrix $\mathbf{W}$ and the transfer function σ. As a result, we may write

$$\widehat{\mathbf{X}^{(0)}} = \sigma(\mathbf{X}^{(s)}\mathbf{W}) \tag{5.3.3}$$

5.3.2.2 Learning of the deep model

The objective of the model is to minimise $\|\mathbf{X}^{(0)} - \widehat{\mathbf{X}^{(0)}}\|_F$ with respect to $\mathbf{V}^{(1)}, \mathbf{V}^{(2)}, ..., \mathbf{V}^{(s)}$, $\mathbf{W}$, subject to $(\prod_{d=1}^{s}\mathbf{V}^{(d)})\mathbf{W} = \mathbf{I}$, where $\mathbf{I} = [\delta_{ij}]_{r_0\times r_0}$ is the Identity matrix of order r_0. As a result, the cost function Φ is defined as

$$\Phi = \frac{1}{2mn}\sum_{p=1}^{m}\sum_{j=1}^{n}(x^{(0)}_{pj} - \widehat{x^{(0)}_{pj}})^2 + \frac{\lambda}{2n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}(a_{ij} - \delta_{ij})^2 \tag{5.3.4}$$

Similar to the shallow model, the first term of Φ measures the reconstruction error and the second term acts as a regularizer. Here, λ is the regularising parameter and δ_{ij} (Kronecker delta) is defined in Chapter 3 equation (3.3.7). The term $\mathbf{A} = [a_{ij}]_{r_0\times r_0}$ is computed as

$$a_{ij} = [\sum_{i_s=1}^{r_s}[\sum_{i_{s-1}=1}^{r_{s-1}}[...[\sum_{i_1=1}^{r_1} v^{(1)}_{i_0 i_1}v^{(2)}_{i_1 i_2}]...]v^{(s)}_{i_{s-1}i_s}]w_{j_s j_0}] \tag{5.3.5}$$

It should be observed here that $n = r_0$, $i_s = j_s$, $i = i_0$ and $j = j_0$. For minimising Φ (equation (5.3.4)) with respect to $\mathbf{V}^{(d)}$ and $\mathbf{W}$, for $1 \leq d \leq s$, $\mathbf{V}^{(d)}$ and $\mathbf{W}$ are successively adjusted as follows:

$$\mathbf{V}^{(d)}(t+1) = \mathbf{V}^{(d)}(t) + \Delta\mathbf{V}^{(d)}(t) \tag{5.3.6}$$

$$\mathbf{W}(t+1) = \mathbf{W}(t) + \Delta\mathbf{W}(t) \tag{5.3.7}$$

Using the gradient descent approach, we obtain

$$\mathbf{V}^{(d)}(t+1) = \mathbf{V}^{(d)}(t) - \eta_{\mathbf{V}^{(d)}} \circ \nabla_{\mathbf{V}^{(d)}}\Phi \tag{5.3.8}$$

$$\mathbf{W}(t+1) = \mathbf{W}(t) - \eta_{\mathbf{W}} \circ \nabla_{\mathbf{W}}\Phi \tag{5.3.9}$$

Hadamard product is represented by '$\circ$' and the learning rates corresponding to $\mathbf{V}^{(d)}$ and $\mathbf{W}$ are represented by matrices $\eta_{\mathbf{V}^{(d)}}\Phi$ and $\eta_{\mathbf{W}}\Phi$, which are the hyper-parameters of the model. Gradient operators with respect to $\mathbf{V}^{(d)}$ and $\mathbf{W}$ are denoted by $\nabla_{\mathbf{V}^{(d)}}$ and $\nabla_{\mathbf{W}}$, respectively. Now, when we calculate derivative of Φ with respect to $\mathbf{W}$, we obtain

$$\nabla_{\mathbf{W}}\Phi = \frac{-1}{mn}{\mathbf{X}^{(s)}}^T\Theta_{\mathbf{W}} + \frac{\lambda}{n^2}(\prod_{q=1}^{s}\mathbf{V}^{(q)})^T(\mathbf{A}-\mathbf{I}) \tag{5.3.10}$$

where

$$\Theta_{\mathbf{W}} = (\mathbf{X}^{(0)} - \widehat{\mathbf{X}^{(0)}}) \circ \widehat{\mathbf{X}^{(0)}} \circ (1 - \widehat{\mathbf{X}^{(0)}}) \tag{5.3.11}$$

We also obtain the derivative of Φ with respect to $\mathbf{V}^{(d)}$.

$$\nabla_{\mathbf{V}^{(d)}}\Phi = \frac{-1}{mn}{\mathbf{X}^{(d-1)}}^T\Theta_{\mathbf{V}^{(d)}} + \begin{cases} \frac{\lambda}{n^2}(\prod_{q=1}^{d-1}\mathbf{V}^{(q)})^T(\mathbf{A}-\mathbf{I})\mathbf{W}^T, & \text{for } d = s \\ \frac{\lambda}{n^2}(\prod_{q=1}^{d-1}\mathbf{V}^{(q)})^T(\mathbf{A}-\mathbf{I})((\prod_{q=d+1}^{s}\mathbf{V}^{(q)})\mathbf{W})^T, & \text{for } 1 < d < s \\ \frac{\lambda}{n^2}(\mathbf{A}-\mathbf{I})((\prod_{q=d+1}^{s}\mathbf{V}^{(q)})\mathbf{W})^T, & \text{for } d = 1 \end{cases} \tag{5.3.12}$$

where

$$\Theta_{\mathbf{V}^{(d)}} = \begin{cases} (\Theta_{\mathbf{W}}\mathbf{W}^T) \circ \mathbf{X}^{(s)} \circ (1 - \mathbf{X}^{(s)}), & \text{for } d = s \\ (\Theta_{\mathbf{V}^{(d+1)}}{\mathbf{V}^{(d+1)}}^T) \circ \mathbf{X}^{(d)} \circ (1 - \mathbf{X}^{(d)}), & \text{otherwise} \end{cases} \tag{5.3.13}$$

Equations (5.3.10) - (5.3.13) are used to rewrite equations (5.3.8) and (5.3.9) as

$$\mathbf{V}^{(d)}(t+1) = \mathbf{V}^{(d)}(t) - \eta_{\mathbf{V}^{(d)}} \circ \begin{cases} \left(\frac{-1}{mn} \mathbf{X}^{(d-1)^T} \Theta_{\mathbf{V}^{(d)}} + \right. \\ \left. \frac{\lambda}{n^2} (\mathbf{A} - \mathbf{I})((\prod_{q=d+1}^{s} \mathbf{V}^{(q)})\mathbf{W})^T \right), & \text{for } d = 1 \\ \left(\frac{-1}{mn} \mathbf{X}^{(d-1)^T} \Theta_{\mathbf{V}^{(d)}} + \right. \\ \left. \frac{\lambda}{n^2} (\prod_{q=1}^{d-1} \mathbf{V}^{(q)})^T (\mathbf{A} - \mathbf{I})((\prod_{q=d+1}^{s} \mathbf{V}^{(q)})\mathbf{W})^T \right), & \text{for } 1 < d < s \\ \left(\frac{-1}{mn} \mathbf{X}^{(d-1)^T} \Theta_{\mathbf{V}^{(d)}} + \right. \\ \left. \frac{\lambda}{n^2} (\prod_{q=1}^{d-1} \mathbf{V}^{(q)})^T (\mathbf{A} - \mathbf{I})\mathbf{W}^T \right), & \text{for } d = s \end{cases} \tag{5.3.14}$$

and

$$\mathbf{W}(t+1) = \mathbf{W}(t) - \eta_{\mathbf{W}} \circ (\frac{-1}{mn} \mathbf{X}^{(s)^T} \Theta_{\mathbf{W}} + \frac{\lambda}{n^2} (\prod_{q=1}^{s} \mathbf{V}^{(q)})^T (\mathbf{A} - \mathbf{I})) \tag{5.3.15}$$

According to the formulation of the model, the elements in $\mathbf{V}^{(d)}$ are unconstrained, but the elements in $\mathbf{W}$ must be non-negative. The hyper-parameters of the model λ, $\eta_{\mathbf{V}^{(d)}}$ and $\eta_{\mathbf{W}}$, for $1 \leq d \leq s$, are set to 0.1. Thus, the update rules for $\mathbf{V}^d$ and $\mathbf{W}$, for $1 \leq d \leq s$ are given by equations (5.3.14) and (5.3.15) respectively. For speedy convergence of the model, we employ momentum factors $\alpha_{\mathbf{V}^{(d)}} = 0.9$ and $\alpha_{\mathbf{W}} = 0.9$. Accordingly, equations (5.3.8) and (5.3.9) becomes

$$\mathbf{V}^{(d)}(t+1) = \mathbf{V}^{(d)}(t) - \eta_{\mathbf{V}^{(d)}} \circ \nabla_{\mathbf{V}^{(d)}} \Phi + \alpha_{\mathbf{V}^{(d)}} \circ \nabla_{\mathbf{V}(t-1)^{(d)}} \Phi \tag{5.3.16}$$

$$\mathbf{W}(t+1) = \mathbf{W}(t) - \eta_{\mathbf{W}} \circ \nabla_{\mathbf{W}} \Phi + \alpha_{\mathbf{W}} \circ \nabla_{\mathbf{W}(t-1)} \Phi \tag{5.3.17}$$

5.4 Experimental Results, Analysis and Discussion

Two aspects have been used to demonstrate and justify the efficacy of MDSR-NMF. First, the degree to which MDSR-NMF has been able to maintain the local structure of the data has been used to assess the quality of dimension reduction. Additionally, an analysis and determination of the effectiveness of the low rank embedding produced

by MDSR-NMF in comparison to the original data have been performed to establish the necessity of dimension reduction. Second, the discriminating power of the dimensionally reduced datasets has been investigated for downstream analyses such as clustering and classification. It has also been investigated how statistically significant the MDSR-NMF results are in comparison to other dimension reduction methods.

As previously mentioned, MDSR-NMF is a two-stage model with pretraining and stacking stages. The goal of MDSR-NMF is to transform an n dimensional feature space into a r dimensional altered feature space. Three shallow neural networks have been used in the pretraining stage. Therefore, in this case, $s + 1 = 3$. The first one reduces the original n dimensional feature space to r_1 dimensions, where r_1 is defined as $r_1 = n - \frac{n-r}{2}$. The second shallow network takes the r_1 dimensional feature vector as input and translates it to the r dimensional feature space. The third shallow network, i.e., $(s+1)^{th}$ network transforms the feature space from n dimensions to r dimensions. The deep neural network architecture at the stacking stage comprises the input layer, two hidden layers and the output layer. From input to output, the number of nodes in these layers is n, r_1, r and n.

Xavier normal initialization technique [34] has been shown to be effective for neural networks with sigmoid type activation functions. The elements of both the weight matrices $\mathbf{V}$ and $\mathbf{W}$ in the proposed MDSR-NMF model have been initialised using the Xavier initialization technique. The number of training epochs is decided dynamically. Training stops on reaching predefined stopping criteria based on the difference in the cost values of two consecutive epochs.

5.4.1 Quantifying the quality of low dimensional embedding

The quality of low dimensional embedding by MDSR-NMF has been investigated in two ways: the ability to preserve the local structure of data using the trustworthiness metric, and the effectiveness of dimension reduction by classification/cluster performance metrics when compared to the original data.

5.4.1.1 Local structure preservation

The trustworthiness score has been used to compute and assess the ability of MDSR-NMF to retain the local structure of data after dimension reduction in comparison to eight other dimension reduction methodologies. The spider/star plot depicts the outcome of the same (Figure 5.2).

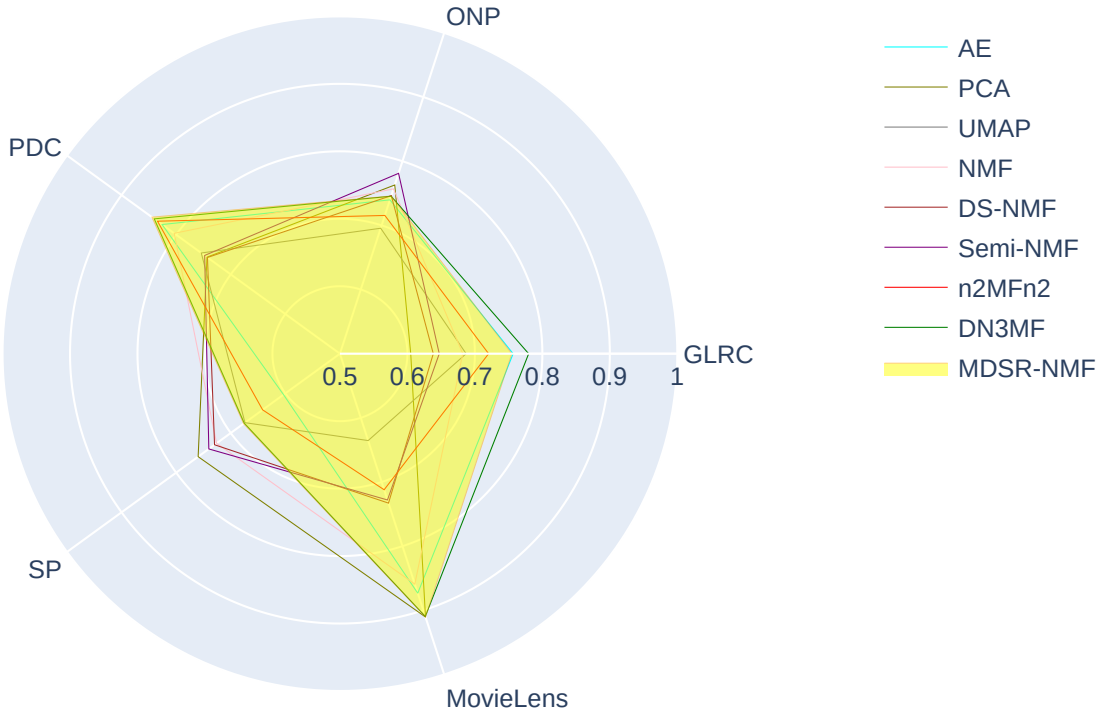

FIGURE 5.2: Trustworthiness scores of nine dimension reduction techniques including MDSR-NMF.

Five datasets are represented by five axes of the plot. A point on that axis represents the trustworthiness score of a dimension reduction technique for a given dataset. As a result, five points on five axes, representing five datasets, correspond to a dimension reduction approach. These points can be thought of as the vertices of a polygon. Consequently, there are nine polygons for nine dimension reduction strategies in Figure 5.2. The area of coverage of a polygon validates a dimension reduction method's effectiveness across all datasets combined. A higher area indicates improved algorithmic performance. From the plot, we can observe that MDSR-NMF has beaten other dimension reduction techniques for the PDC and MovieLens datasets and for GLRC and ONP datasets the trustworthiness score of MDSR-NMF is better than most of the others. The area bounded by the polygon of MDSR-NMF is shown in a shaded colour in Figure 5.2. We compute the area of the polygon by adding individual trustworthiness scores of the

TABLE 5.1: Sum of trustworthiness scores of nine dimension reduction techniques including MDSR-NMF on five datasets.

Dimension reduction techniques	Sum of trustworthiness scores
AE	4.55550328220533
PCA	4.38647684863791
UMAP	4.13297187263103
NMF	4.50973409888627
DS-NMF	4.22833659492512
Semi-NMF	4.29147059452664
n^2MFn^2	4.34404888837219
DN3MF	**4.72960521062986**
MDSR-NMF	4.68627962966105

dimension reduction techniques for all five datasets. It can be observed from Table 5.1 that the sum of trustworthiness scores of MDSR-NMF is the second highest among all. This value is just lower than that of DN3MF. However, the trustworthiness score of MDSR-NMF is able to beat the remaining seven dimension reduction methods. Thus, the quality of low dimensional embedding produced by MDSR-NMF is superior to that produced by the other dimension reduction methods and comparable to that of DN3MF.

5.4.1.2 Decision making: Comparison with the original data

By classifying and clustering both the original data and the low dimensional embedding produced by MDSR-NMF and then quantifying the results using various cluster validity and classification metrics, the effectiveness of dimension reduction by MDSR-NMF has been assessed. The low rank representation of the data promotes the usability of the same over the original one, which is why the dimension reduction is necessary.

Classification

Figures 5.3-5.7 presents the performance of MDSR-NMF and original data in terms of classification. For the PDC (Figure 5.5) dataset, MDSR-NMF generated low rank embedding has outperformed the original data for all four classifiers in terms of all four metrics. The same statistics for the ONP (Figure 5.4) dataset are three out of four. When the classifier evaluator is MCC, for the GLRC (Figure 5.3) dataset the same count

is three out of four and for the remaining metrics MDSR-NMF has outperformed the original in all four out of four cases. For MovieLens datasets, for ACC, FS and MCC performance metrics, MDSR-NMF has performed better than the original dataset for three out of four classification algorithms (Figure 5.7). In terms of CKS, the scoreline favouring MDSR-NMF is two out of four. In the case of the SP dataset, the performance metric of original data is better than the low rank embedding produced by MDSR-NMF on all occasions (Figure 5.6).

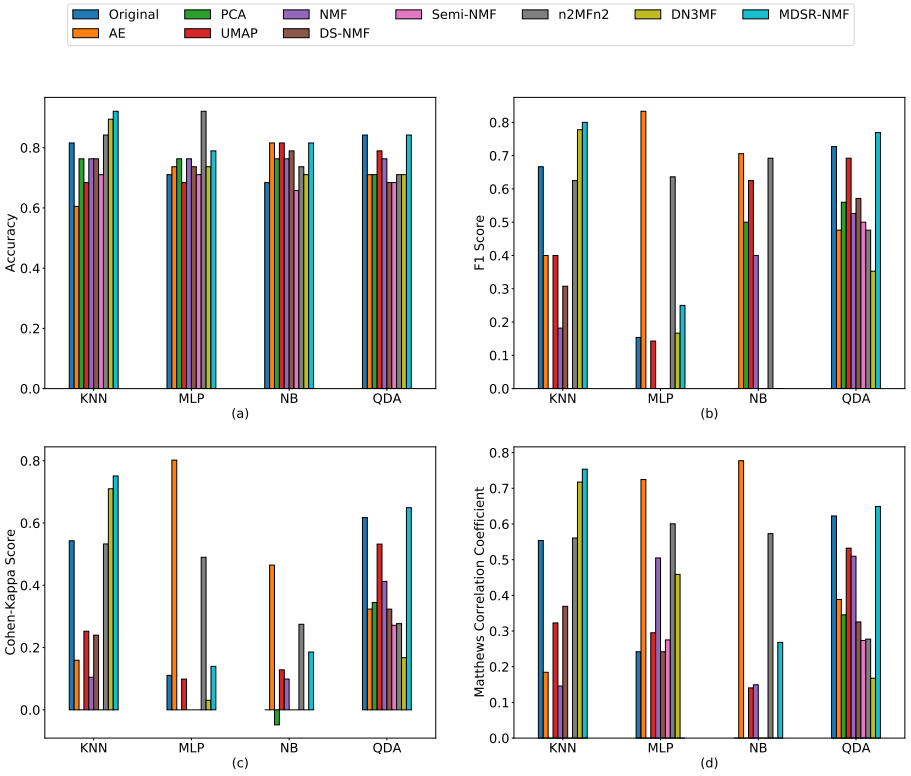

FIGURE 5.3: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the GLRC dataset by MDSR-NMF and eight other dimension reduction techniques along with the original data.

It is clear from these experiments that, for most of the cases, the MDSR-NMF projected data have outperformed the original data in terms of classification. This supports the requirement for both dimension reduction and the capacity to produce low rank embeddings that preserve the fundamental properties of the data.

Clustering

The performance comparison of clustering done on the low dimensional embedding

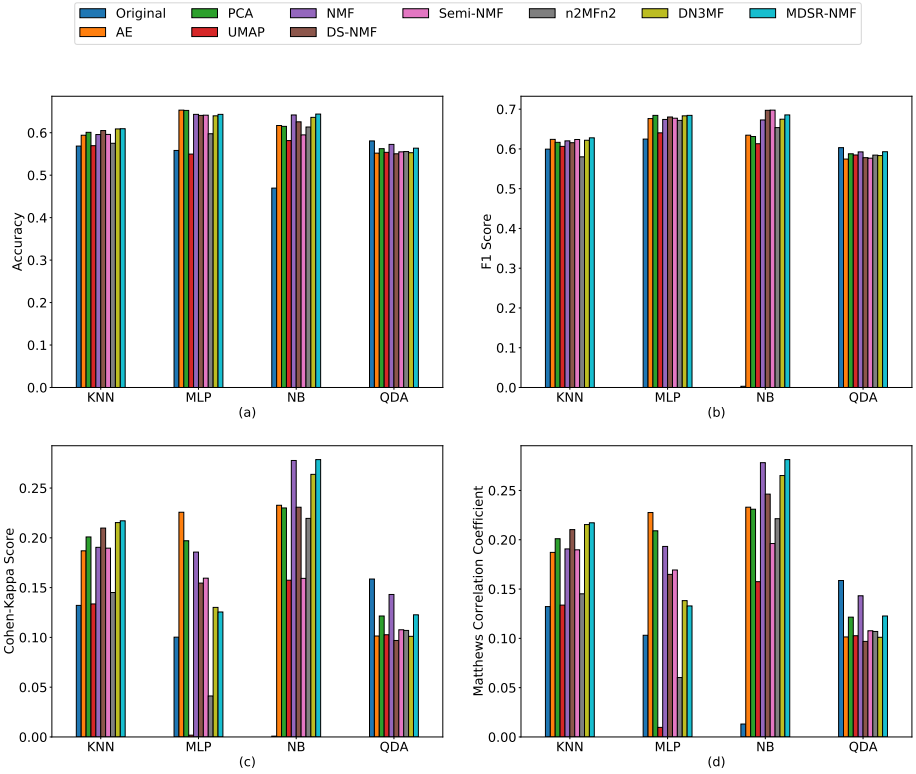

FIGURE 5.4: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the ONP dataset by MDSR-NMF and eight other dimension reduction techniques along with the original data.

produced by MDSR-NMF and the original data has been illustrated in Figures 5.8-5.12. For the ONP (Figure 5.9) dataset, for all four cluster validity indexes, MDSR-NMF has performed better than the original data with respect to all four clustering algorithms and for GLRC (Figure 5.8) dataset the same count is three out of four. For the PDC (Figure 5.10), SP (Figure 5.11) and MovieLens (Figure 5.12) datasets, for NMI cluster validity index, MDSR-NMF has performed better than the original data for four out of four clustering algorithms. In terms of AMI, the performance score favouring PDC and SP datasets is four out of four and for the MovieLens dataset, the same count is three. For the ARI metric, the performance score is three out of four in favour of MDSR-NMF for the PDC dataset and two out of four for both SP and MovieLens datasets. Similarly, in the case of JI, the scorelines favouring MDSR-NMF are two, three and four for PDC, SP and MovieLens datasets respectively.

As a result, it has been demonstrated that the low rank embedding produced by MDSR-NMF performs significantly better in terms of clustering in terms of preserving the essential characteristics of the original data. Dimension reduction is therefore

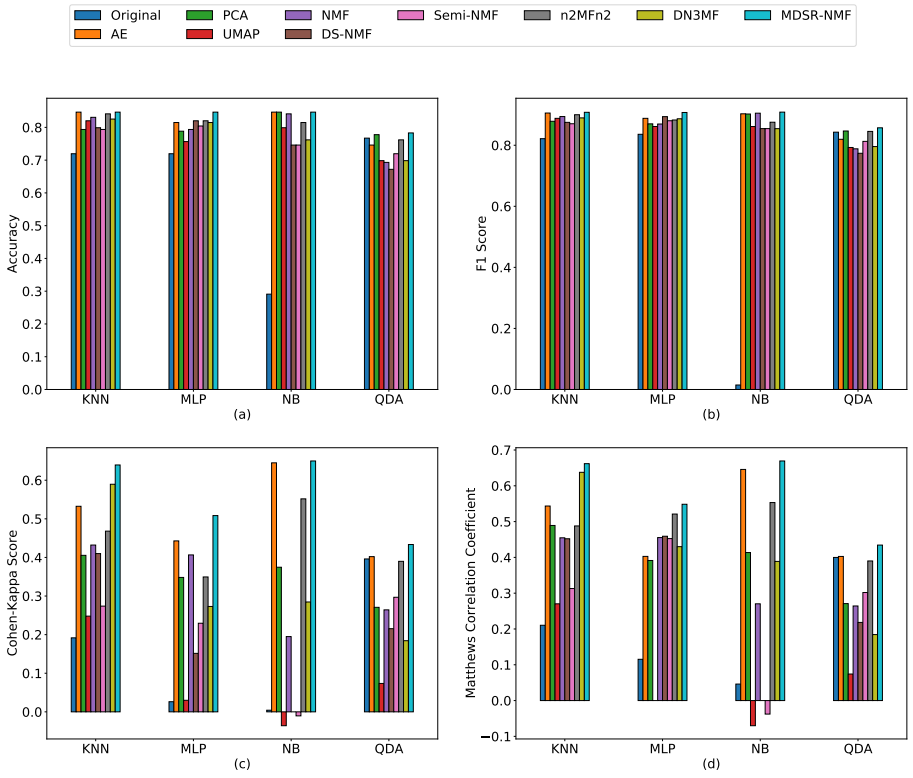

FIGURE 5.5: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the PDC dataset by MDSR-NMF and eight other dimension reduction techniques along with the original data.

required and justified.

5.4.2 Downstream analyses and statistical significance: Comparison with other models

The efficiency of dimension reduction has been assessed by performing classification and clustering on the low dimensional embedding produced by MDSR-NMF as well as those generated by the other eight dimension reduction methods. Several measures for monitoring classification and cluster performance have been employed to quantify the outcome. Pairwise p-values have been calculated as well, to demonstrate that MDSR-NMF outperforms other dimension reduction algorithms, in terms of generating output from an independent set of data. A p-value less than a specific threshold validates the statistical significance of the outcomes. Here, we have set the threshold value at 0.05. Thus, for a dataset, a classification/clustering algorithm and a classification/cluster validity measure, eight p-values have been calculated to compare the performance of

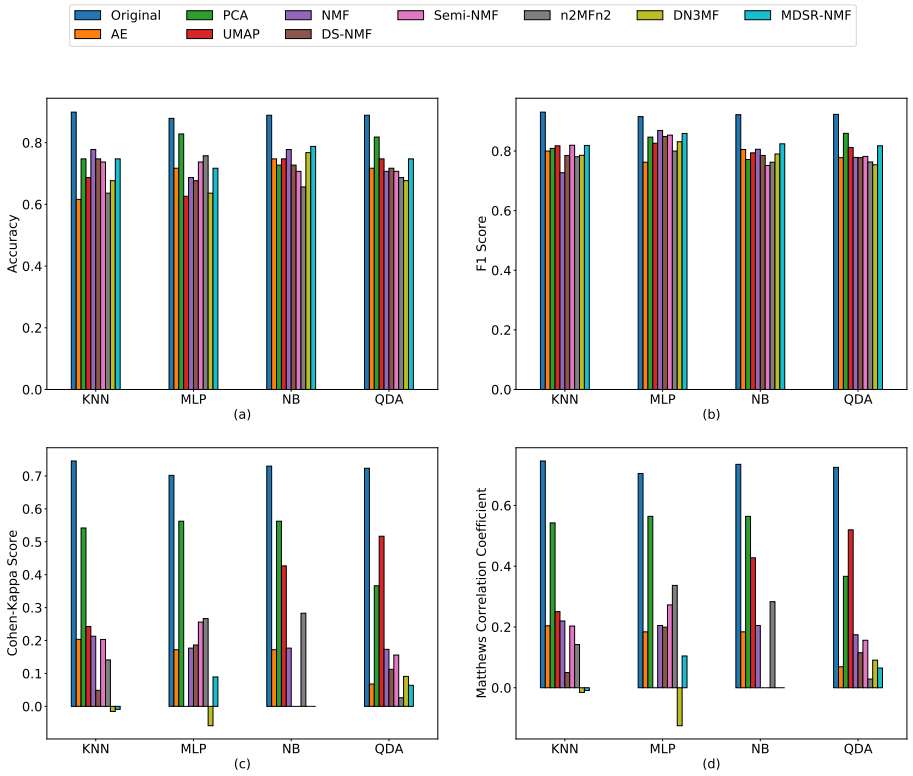

FIGURE 5.6: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the SP dataset by MDSR-NMF and eight other dimension reduction techniques along with the original data.

MDSR-NMF and eight other different dimension reduction strategies considered here. There are four classification/clustering techniques, resulting in a total of $8 \times 4 = 32$ p-values for each validity index against each dataset. This section of the experiment tries to determine the superiority of dimension reduction by MDSR-NMF using various types of classification and clustering techniques.

Classification

While working with the MDSR-NMF model for classification, the outcome has been depicted by Figures 5.3-5.7. The summary of the count of statistically significant p-values with respect to MDSR-NMF has also been presented in Table 5.2.

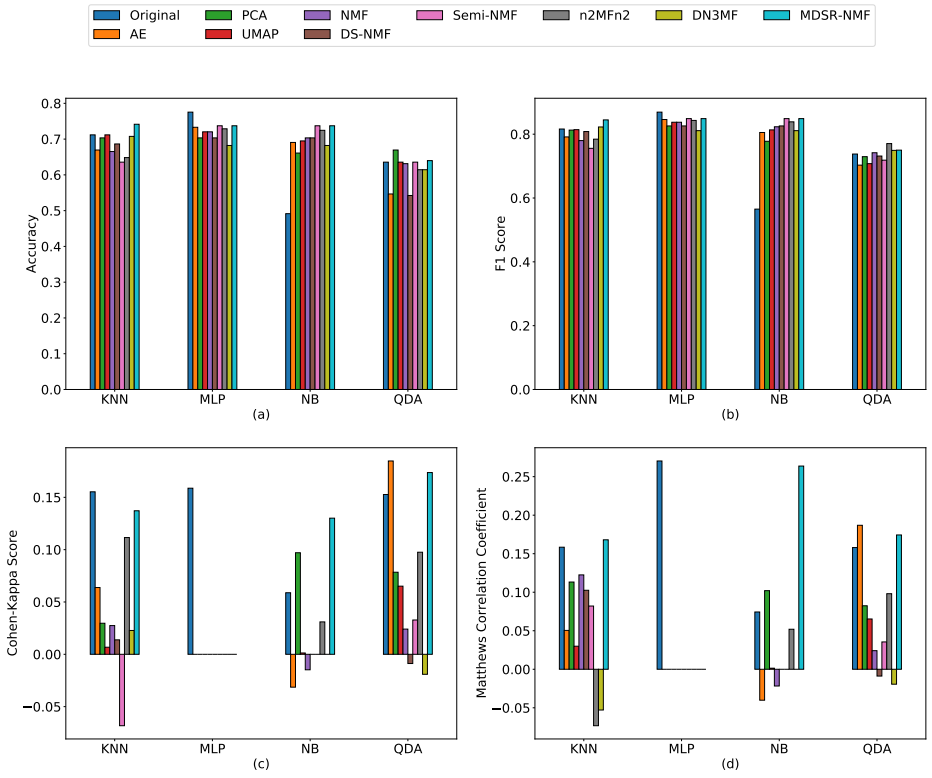

FIGURE 5.7: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the MovieLens dataset by MDSR-NMF and eight other dimension reduction techniques along with the original data.

For four classification techniques, MDSR-NMF has always achieved the highest accuracy score for the PDC (Figure 5.5(a)) dataset and three times for the GLRC (Figure 5.3(a)) and MovieLens (Figure 5.7(a)) datasets. The same counts are two and one respectively for the ONP (Figure 5.4(a)) and SP (Figure 5.6(a)) datasets. MDSR-NMF has surpassed the others in terms of F1 score all four times for PDC (Figure 5.5(b)), thrice for ONP (Figure 5.4(b)) and MovieLens (Figure 5.7(b)), twice for GLRC (Figure 5.3(b)) and once for SP (Figure 5.6(b)) datasets. When the Cohen-Kappa score is used as the classification performance indicator, the outcome favouring MDSR-NMF on the GLRC (Figure 5.3(c)) and ONP (Figure 5.4(c)) datasets is two out of four. For the PDC (Figure 5.5(c)) dataset, it is four out of four and for the MovieLens (Figure 5.7(c)) dataset, the same count is three out of four. In the case of the SP (Figure 5.6(c)) dataset, MDSR-NMF has failed to outperform the original dataset. The same statistics hold when the classification performance metric is the Matthews Correlation Coefficient (Figures 5.3(d), 5.4(d), 5.5(d), 5.6(d) and 5.7(d)).

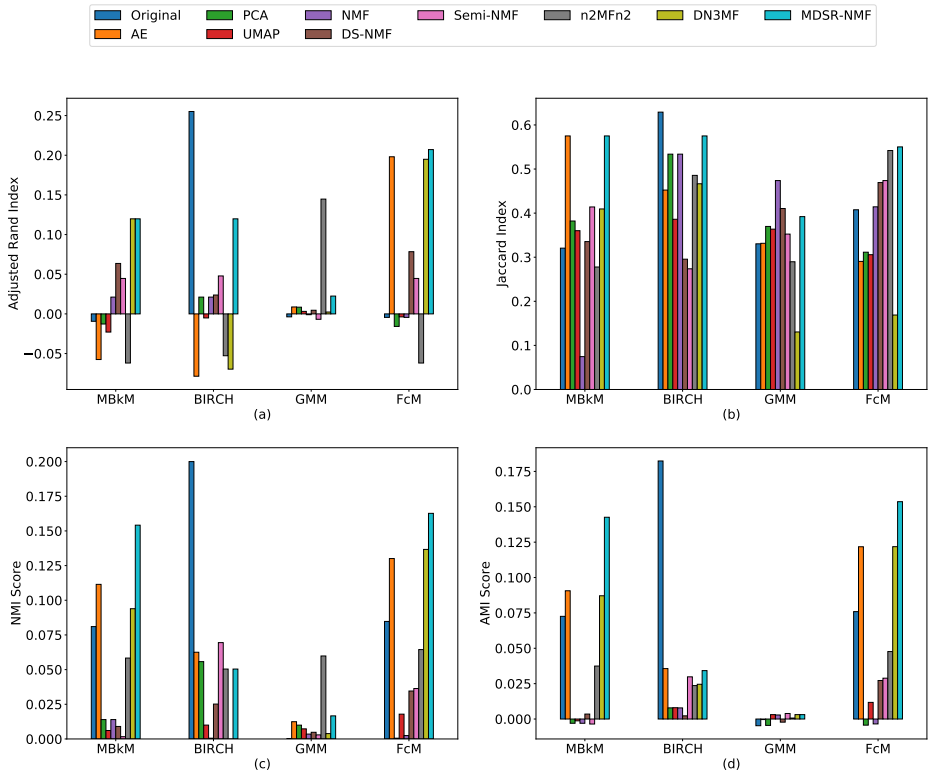

FIGURE 5.8: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the GLRC dataset by MDSR-NMF and eight other dimension reduction techniques along with the original data.

The description presented above makes it evident that the transformed dataset using MDSR-NMF has a higher accuracy score than the others in most cases, across GLRC, ONP, PDC and MovieLens datasets and four different classifier types. In terms of the SP dataset, MDSR-NMF has a competitive performance. A model's accuracy indicates how frequently it is accurate. We have calculated the F1 score, or the harmonic mean of precision and recall, in addition to accuracy. Figures 5.3-5.7 reveal that MDSR-NMF beat other models in terms of F1 score in the majority of cases except for the SP dataset, where the performance is comparable. Thus, the superiority of MDSR-NMF is validated by its Accuracy and F1 score. On the other hand, the Cohen-Kappa score is a statistical measure of inter-rater agreement. The figures demonstrate that MDSR-NMF produced higher positive Cohen-Kappa scores and outperformed the others in the majority of scenarios for four out of five datasets. As a result, it is feasible to assume that MDSR-NMF is able to keep and learn the fundamental characteristics of the input data, resulting in higher ratings. The quality of binary and multiclass classifications can be evaluated using the Matthews Correlation Coefficient. Better agreement is implied by

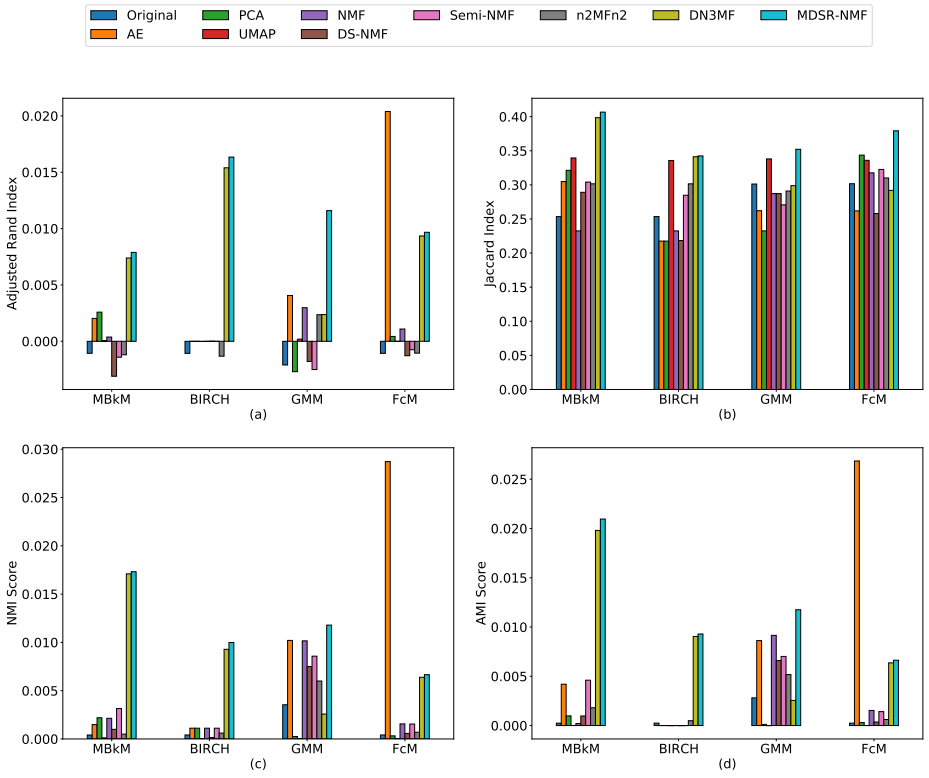

FIGURE 5.9: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the ONP dataset by MDSR-NMF and eight other dimension reduction techniques along with the original data.

TABLE 5.2: The summary of the count (out of 32) of statistically significant p-values for each classification performance metric against each dataset with respect to MDSR-NMF.

Dataset	ACC	FS	CKS	MCC
GLRC	17	22	25	25
ONP	19	10	21	21
PDC	29	30	30	31
SP	11	08	21	21
MovieLens	27	28	08	11

a higher MCC score, indicating that the model is also able to maintain the class characteristics of the original dataset in the transformed dataset as well. The MCC score of MDSR-NMF is higher than that of the other models except for the SP dataset, where the performance of MSDSR-NMF is comparable to others (Figures 5.3-5.7). The above explanation illustrates the superiority of MDSR-NMF over other dimension reduction methods with respect to intrinsic property preservation criteria as well as statistical measures.

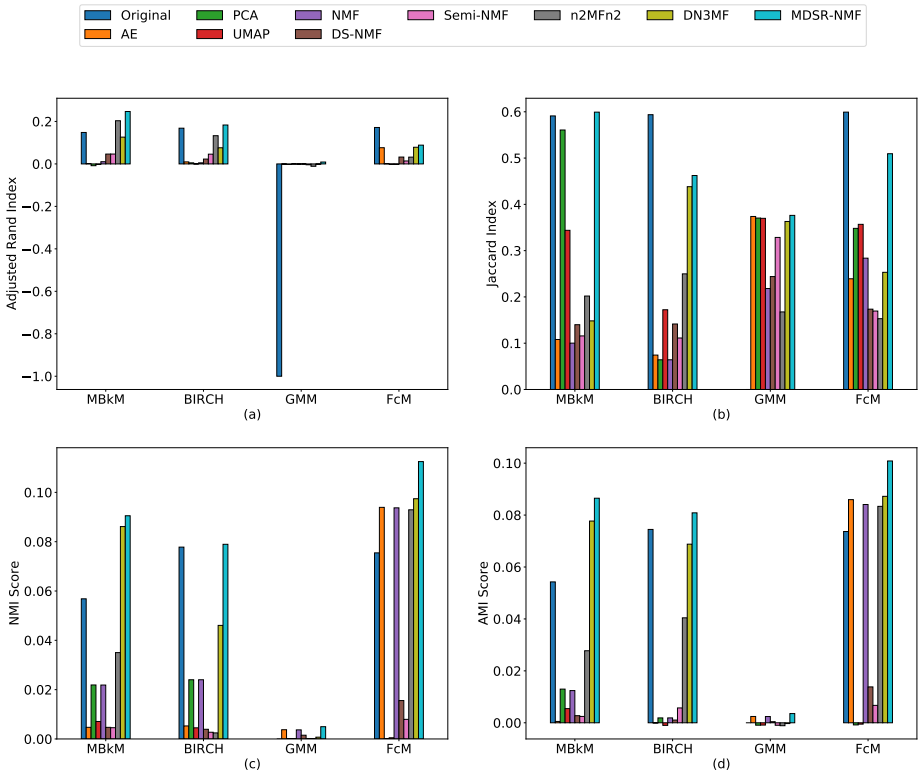

FIGURE 5.10: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the PDC dataset by MDSR-NMF and eight other dimension reduction techniques along with the original data.

For each classification performance index against each dataset, out of a total of 32 p-values, the count of p-values less than the determined threshold (0.05) is presented in Table 5.2. The above statistics indubitably quantify the quality of low rank embedding produced by MDSR-NMF over others.

Clustering

For clustering purposes with the MDSR-NMF model, Figures 5.8-5.12 present the outcome. Table 5.3 provides an overview of the count of statistically significant p-values for MDSR-NMF for clustering.

MDSR-NMF has achieved the highest performance score for the Adjusted Rand index for the PDC (Figure 5.10(a)) dataset for all four clustering approaches considered here. This count is three out of four for the GLRC (Figure 5.8(a)) and ONP (Figure 5.9(a)) datasets and one for the SP (Figure 5.11(a)) and MovieLens (Figure 5.12(a)) datasets.

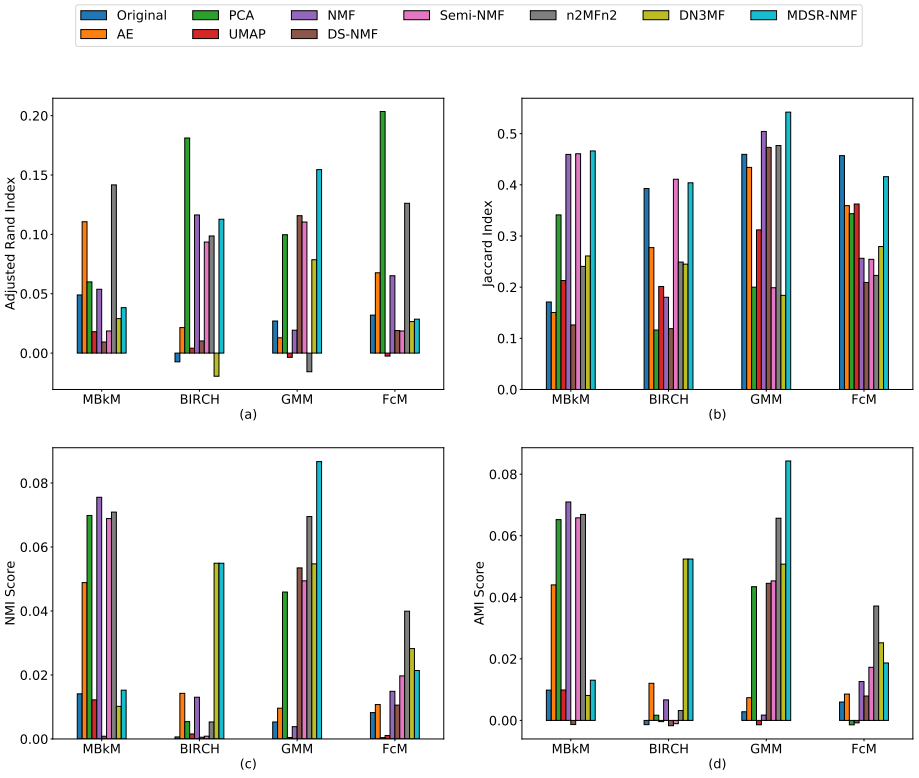

FIGURE 5.11: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the SP dataset by MDSR-NMF and eight other dimension reduction techniques along with the original data.

When using the Jaccard Index as the cluster validity estimator, MDSR-NMF has outperformed the others in four out of four clustering algorithms on the PDC (Figure 5.10(b)) and ONP (Figure 5.9(b)) datasets. This value ranks three out of four for the GLRC (Figure 5.8(b)) and SP (Figure 5.11(b)) datasets and two for the MovieLens (Figure 5.12(b)) dataset. MDSR-NMF projected transformed space has achieved the highest NMI score among the other dimension reduction techniques four out of four times for the PDC (Figure 5.10(c)) dataset, thrice for ONP (Figure 5.9(c)) and MovieLens (Figure 5.12(c)) datasets and twice for GLRC (Figure 5.8(c)) and SP (Figure 5.11(c)) datasets. The same statistics hold when the cluster performance evaluator is AMI except for the MovieLens dataset where the count favouring MDSR-NMF is four out of four (Figures 5.8(d), 5.9(d), 5.10(d), 5.11(d) and 5.12(d)).

The Adjusted Rand Index (ARI) measures the similarity of two data clusters. Figures 5.8-5.12 demonstrate that MDSR-NMF outperformed other dimension reduction approaches across five datasets and four clustering algorithms in terms of the ARI

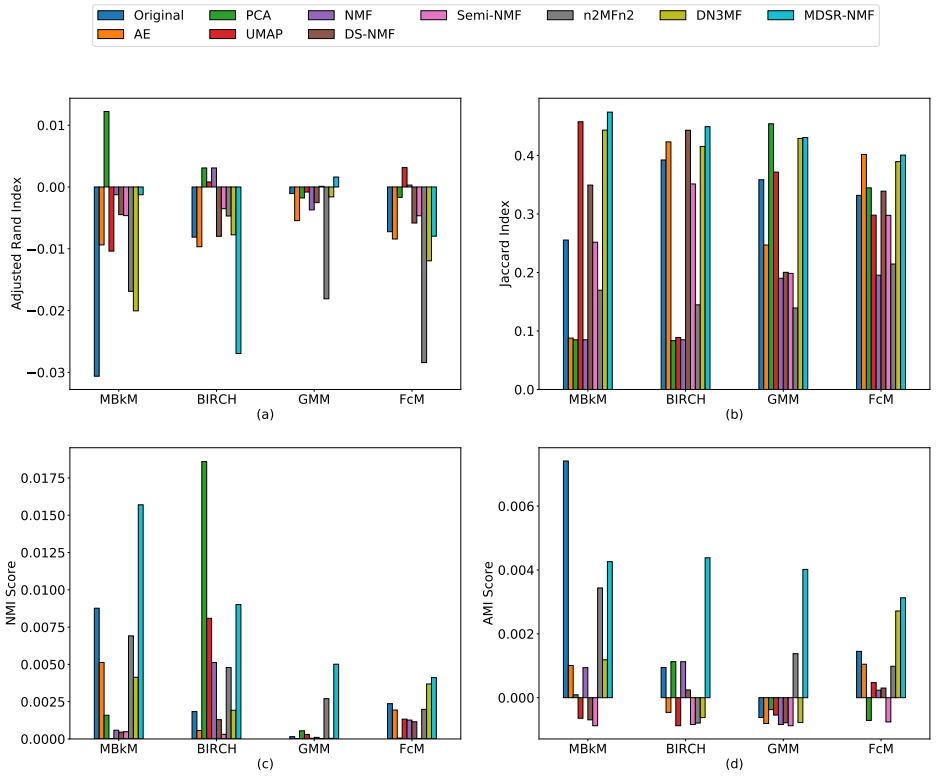

FIGURE 5.12: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the MovieLens dataset by MDSR-NMF and eight other dimension reduction techniques along with the original data.

score. The Jaccard Index is used to calculate the similarity between two sets. MDSR-NMF surpassed the others in terms of the Jaccard Index. Thus, it might be stated that MDSR-NMF learned the basic properties of the input and accurately mapped them to a low rank representation. NMI is defined as the normalization of the Mutual Information score to scale the outcomes in the interval $[0, 1]$. This measure is not corrected for chance. In contrast, the AMI score remains constant regardless of how the class or cluster label is arranged. The results of the NMI and AMI scores demonstrate that MDSR-NMF has performed better than other dimension reduction techniques (Figures 5.8-5.12). The enhanced performance of MDSR-NMF indicates that, in comparison to the other methods examined here, the low rank representation of the datasets employing MDSR-NMF has been able to preserve the intrinsic properties of the original data more successfully.

Out of a total of 32 p-values for each cluster validity index against each dataset, Table 5.3 displays the count of p-values that fall below the decided threshold, i.e., 0.05.

TABLE 5.3: The summary of the count (out of 32) of statistically significant p-values for each cluster performance metric against each dataset with respect to MDSR-NMF.

Dataset	ARI	JI	NMI	AMI
GLRC	20	21	18	17
ONP	25	29	24	24
PDC	23	28	29	29
SP	06	15	06	09
MovieLens	19	21	28	29

The aforementioned tally unequivocally demonstrates how good low rank embedding produced by MDSR-NMF is compared to others.

5.4.3 Discussion

In terms of trustworthiness, it is evident that MDSR-NMF has demonstrated its effectiveness in dimension reduction by maintaining the granular relationships of data by scoring the highest trustworthiness score for two datasets, having better than average scores for another two and average scores for the remaining one. With respect to the overall area, MDSR-NMF has just performed below DN3MF outperforming the remaining models.

Each dimension reduction algorithm has a total of $5 \times 4 \times 4 = 80$ performance scores for five datasets, four classification algorithms and four classification performance measures. It is evident that on 54 out of 80 instances, the MDSR-NMF predicted datasets have outperformed the original data. On the other hand, while comparing the performance with other dimension reduction methods, MDSR-NMF has obtained the highest rating, 48 times out of 80. In contrast to DN3MF, MDSR-NMF has bettered its performance for ONP dataset when compared to other dimension reduction techniques, otherwise, the rest of the performances are similar to that of DN3MF. Thus, it is undeniable that MDSR-NMF is superior to the others.

By comparing the p-values that the low rank embeddings generated by MDSR-NMF for various datasets produce, it is demonstrated that these embeddings are statistically significant and can outperform both the original and other dimensionally reduced datasets generated by different dimension reduction methods including n^2MFn^2 and DN3MF. For the GLRC dataset, 89 out of 128 $(4 \times 4 \times 8)$ cases represent statistically

significant findings pertaining to MDSR-NMF for all classifiers and classification metrics. the figures 71, 120, 61 and 74, respectively, are the same counts for the PDC, ONP, SP and MovieLens datasets. In order to produce statistically meaningful low dimensional embeddings, MDSR-NMF is, therefore, proven to be more effective than other dimension reduction methods.

Analogous to classification, MDSR-NMF has been validated for clustering using four clustering methods and four cluster validity metrics across five datasets. In terms of clustering, MDSR-NMF has registered a higher performance of 67 times out of 80 potential scenarios when comparing the performance against the original data. Compared to other dimension reduction methods, MDSR-NMF has a notable superiority count of 57 out of 80. Considering the previous discussion, it is evident that MDSR-NMF has shown to be more effective than the other dimension reduction techniques considered here in most of the cases.

In addition to outperforming the original and other dimensionally reduced datasets produced by different dimension reduction methods, MDSR-NMF's low rank embeddings for various datasets have been shown to be statistically significant based on the comparative p-values they produce. The GLRC dataset has resulted in 76 MDSR-NMF related statistically significant performance count out of 128 $(4 \times 4 \times 8)$. The similar counts for the PDC, ONP, SP and MovieLens datasets are 102, 109, 36 and 97, respectively. As a result, MDSR-NMF outperforms other dimension reduction methods in producing statistically meaningful low-dimensional embeddings.

5.5 Convergence Analysis

Based on the experimental results, we aim to establish the convergence of the proposed MDSR-NMF model. The convergence plots for four datasets, GLRC, ONP, PDC, SP and MovieLens are shown in Figure 5.13. Figure 5.13 has five subplots in each row. Each subplot illustrates the variation of the cost function Φ with respect to iteration. The convergence plots of the first shallow neural network architecture are depicted in Figures 5.13(a), (e), (i), (m) and (q). The first shallow neural network lowers the input feature space dimension n to r_1 (Section 5.4). Figures 5.13(b), (f), (j), (n) and (r) depict the convergence plots of the second shallow neural network architecture. The second

shallow network lowers r_1 dimensional feature space to r dimension. Figures 5.13(c), (g), (k), (o) and (s) show how $(s+1)^{th}$ shallow network converges. $(s+1)^{th}$ shallow network reduces n dimensional feature space directly to r dimensional feature space. Finally, Figures 5.13(d), (h), (l), (p) and (t) depict the convergence plots of the deep model or stacking stage.

Figures 5.13(a) through 5.13(d) depict the cost versus iteration plots for the GLRC dataset with $r_1 = 398$ and $r = 97$. Figures 5.13(e) - 5.13(h) show the convergence plots for ONP dataset with $r_1 = 40$ and $r = 22$. Figures 5.13(i) - 5.13(l) display the convergence plots for PDC dataset with $r_1 = 414$ and $r = 75$ and Figures 5.13(m) - 5.13(p) depict the convergence plots for SP dataset with $r_1 = 24$ and $r = 17$. The convergence plots for the MovieLens dataset for $r_1 = 1063$ and $r = 445$ have been depicted in Figures 5.13(q) - 5.13(t).

An initial spike in the error curve has been noted in the cost versus iteration plots. This might be because the initialization of the weights has been so close to ideal that the application of a fixed momentum factor has driven the error up somewhat for a short number of iterations before a consistent drop in the error value. Finally, for shallow networks, the error curve has nearly flattened out after a certain number of iterations. In some stacked network scenarios, the tiny fluctuation in the error value during training may also be the result of the fixed momentum component. Overall, the decreasing nature of the cost over time demonstrates that both the shallow and deep networks converge.

5.6 Analysis of Computational Complexity

The computational complexity of MDSR-NMF is measured in terms of the number of operations performed. The upper bound of the complexity is given in terms of $\mathcal{O}$ notation. MDSR-NMF has two stages: pretraining and stacking.

The computational complexity of the pertaining stage of MDSR-NMF is the same as that of n^2MFn^2 and is given as $\mathcal{O}(t_p(mnr + n^2r))$ where, t_p denotes the number of epochs, (m, n) being the order of the input matrix $\mathbf{X}$ and r is the number of nodes in the slender layer. In the pretraining stage of MDSR-NMF, there are s consecutive shallow

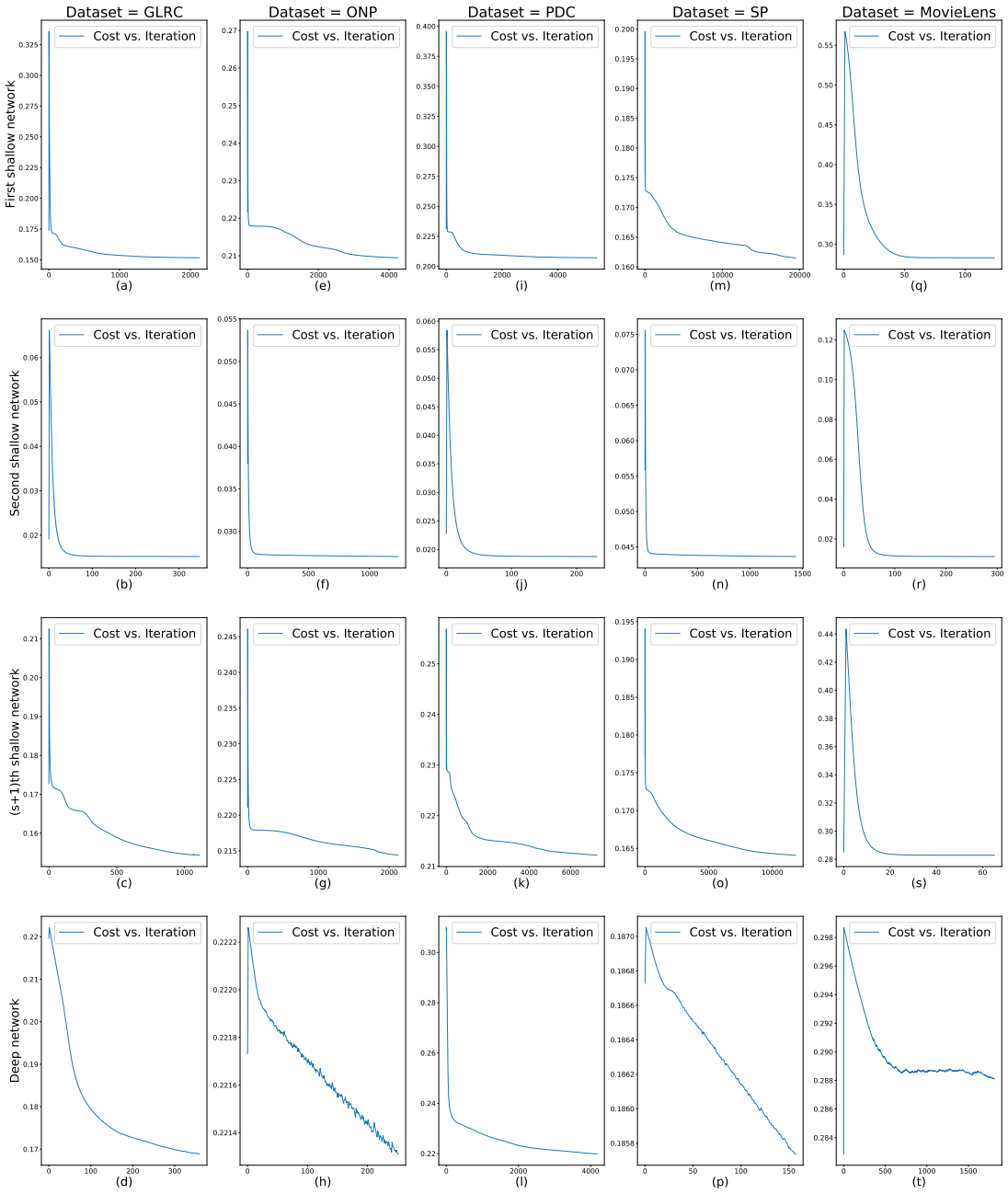

FIGURE 5.13: Cost vs. iteration plots of MDSR-NMF for (a)-(d) GLRC, (e)-(h) ONP, (i)-(l) PDC, (m)-(p) SP and (q)-(t) MovieLens dataset for both pre-training and stacking stages of MDSR-NMF.

models. As a result, the computational complexity of the pretraining stage of MDSR-NMF is $\mathcal{O}(st_p(mnr_1 + n^2r_1))$, where r_1 is the size of the slenderest layer of the first shallow model and $r_1 > r_2 > ... > r_s$.

The pretraining stage of MDSR-NMF has another $(s+1)^{th}$ shallow model which directly reduces the input dimension n to r. Following a similar argument as above, we can say that in this case, the computational complexity will be $\mathcal{O}(t_{p_1}(mnr + n^2r))$,

where t_{p_1} is the number of epochs. As $r_s = r$ and $r_s < r_1$, so we can discard $\mathcal{O}(t_{p_1}(mnr + n^2r))$. The complexity of the forward pass in the stacking stage may be calculated in a similar way as $\mathcal{O}(mr_0r_1)$, where $r_0 = n$ and $r_0 > r_1 > ... > r_s$. The computational complexity of calculating $\mathbf{A}$ (equation 5.3.5) is $\mathcal{O}(r_0r_1r_0)$, i.e., $\mathcal{O}(n^2r_1)$. Thus, $\mathcal{O}(mn + n^2r_1)$ operations are involved in the computation of Φ (equation 5.3.4). $\mathcal{O}(mnr_1 + n^2r_1)$ gives the upper bound on the number of operations in the backward pass. As a result, the overall computational cost of an epoch of the MDSR-NMF stacking stage is $\mathcal{O}(mnr_1 + n^2r_1)$. The complexity for t_s epochs is $\mathcal{O}(t_s(mnr_1 + n^2r_1))$. Hence, the computational complexity of MDSR-NMF is $\mathcal{O}(st_p(mnr_1 + n^2r_1) + t_s(mnr_1 + n^2r_1))$, i.e., $\mathcal{O}((st_p + t_s)(m + n)nr_1)$.

5.7 Conclusions

A large dataset can be dimensionally reduced using a variety of ways. We have combined the benefits of NMF, a classic matrix factorization approach and deep learning for dimension reduction in this architecture. MDSR-NMF, a novel deep learning model, has been designed for NMF towards dimensionality reduction. Pretraining and stacking are the two stages of MDSR-NMF. These pretraining and stacking steps contain two phases: deconstruction and reconstruction. The novel architecture has been designed in such a way that it resembles the factorization behaviour of the traditional NMF procedure with respect to producing a set of unique factor matrices.

The technique for weight initialization, transfer function, objective function and learning algorithm has been designed in such a way that it contributes to optimal MDSR-NMF learning. The regularizer has been purposely designed in such a manner that it ensures the best possible approximation of the input matrix as the product of two factor matrices. The superiority of MDSR-NMF over eight other dimension reduction strategies has been established through extensive experimentation on five prominent datasets using both classification and clustering. Three NMF-based approaches, three additional classic dimension reduction algorithms and previously designed two models are among the eight other dimension reduction strategies. In terms of local structure preservation criteria, MDSR-NMF has performed little below that of the previously developed model DN3MF but has performed better than the remaining models. During

experimentation, a total of 4 classification algorithms, 4 classification performance measures, 4 clustering methods and 4 cluster validity indexes have been utilised. More or less the performance of MDSR-NMF is similar to that of DN3MF both in comparison to the original dataset and in comparison with other dimension reduction techniques. Experimentation has been performed to demonstrate the convergence of MDSR-NMF. The computational complexity of the model has also been presented.

The design of our next model architecture has been motivated by the way a human being learns a new concept by constantly referring to the original text in order to maintain the correct path of learning and increase the efficacy of knowledge acquisition.

Chapter 6

Input Guided Multiple Deconstruction Single Reconstruction neural network for Non-negative Matrix Factorization (IG-MDSR-NMF)

6.1 Introduction

The aspiration of simulating the factorization behaviour of the traditional NMF technique ensuring the outcome of a unique pair of factor matrices of the reconstructed input matrix has motivated us for the progressive development of the previous three models (Chapters 3-5). In this chapter, we have fused the advantages of conventional iterative learning with those of deep learning in a way that resembles the trait of human learning. While learning, humans always attempt to disintegrate the concepts into smaller fragments and try to learn hierarchically referring back to the original details frequently ensuring the correctness of the learning. Thus, we can claim that human learning is always input-guided.

Thus, here we have designed a model, called Input Guided Multiple Deconstruction Single Reconstruction neural network for Non-negative Matrix Factorization (IG-MDSR-NMF) towards dimension reduction [28]. Deconstruction and reconstruction are the two phases of the model. The layers in the deconstruction phase receive the hierarchically processed output of the preceding layer along with a copy of the original data as input. Thus the model is called “Input Guided". There is only one layer in the reconstruction phase, which ensures a true realization of the NMF technique generating a unique pair of non-negative factor matrices. The main objective of the study is to find a low rank approximation of the input data to get rid of the curse of dimensionality problem.

The quality of low dimensional embedding produced by IG-MDSR-NMF has been verified based on the extent of input shape preservation. The need for dimension reduction over the original data has also been judged and justified. The superiority of the low rank approximation by IG-MDSR-NMF has also been verified over nine well-known dimension reduction techniques, experimenting on five datasets for both classification and clustering. The efficacy of IG-MDSR-NMF has been justified for downstream analyses (classification and clustering) on different types of datasets. The statistical significance of the results has also been established.

The rest of the chapter is organised as follows. Section 6.2 describes the motivation behind the architecture and learning of IG-MDSR-NMF. The detailed design and derivation of respective learning rules have been presented in Section 6.3. Subsequently, Section 6.4 depicts the results following the experimentation procedure described in Chapter 2, with an adequate analysis. The convergence analysis of IG-MDSR-NMF and the analysis of computational complexity have been presented in Sections 6.5 and 6.6. Finally, Section 6.7 concludes the chapter.

6.2 Motivation behind Architecture and Learning

The philosophy behind the novel architecture of the Input Guided Multiple Deconstruction Single Reconstruction neural network for Non-negative Matrix Factorization model is described here.

Learning of human beings is an iterative process where complex concepts are gradually broken down into simpler ones. Along with the fragmented concepts, humans always refer to the original material whenever needed, ensuring the correctness of learning. A Deep neural network learns hierarchically, i.e., the representation of data is learned in a layerwise approach. That is, the layer l_1 connecting the input layer l_0 learns directly from the input and the next layer l_2 learns from the learned representation of l_1 and so on. Therefore, the deeper the network grows and the upper-level layers receive a more abstract form of representation of the original input. If an intermediate layer, for some reason, is unable to learn appropriately from its previous layer, this improper learning will percolate to the succeeding layers and finally will affect decision making to a great extent. Thus, there is a possibility of deviation in learning the actual information content of the raw data. To overcome this phenomenon, a novel architecture has been designed in this article, where the input layer l_0 is additionally connected with the layers $l_2, l_3, ...$, till the second last layer of the architecture. The design enforces the model to be guided by the original representation of the input in the process of hierarchically learning representation of the same, mimicking the human way of learning. As each layer of the model is guided by the input, the model has been named as Input Guided Multiple Deconstruction Single Reconstruction neural network for Non-negative Matrix Factorization (IG-MDSR-NMF).

IG-MDSR-NMF has been developed for the task of NMF. The model is divided into two phases, viz., deconstruction and reconstruction. The input to the neural network is transformed into latent space during the deconstruction phase and the network attempts to reconstruct the input from its low rank representation during the reconstruction phase. There are multiple deconstruction layers leading towards the slenderest layer of the network from the input but there is only one reconstruction layer connecting the slenderest and the output layers of the network. Hence, the model is called Multiple Deconstruction Single Reconstruction neural network model. The novel architecture tries to learn the low rank representation of the input data in a stepwise manner in the deconstruction phase of the model. Whereas, in the reconstruction phase, it directly tries to reconstruct the input from the latent space. The architecture has been designed in this manner to simulate the factorization behaviour of the traditional NMF technique. By taking only one layer in the reconstruction step, the model can synthesize a unique pair of constituent non-negative factor matrices. The detailed philosophy

behind such architecture has been described in Chapter 5 Section 5.2.

6.3 IG-MDSR-NMF

In this section, we have described the detailed architecture of IG-MDSR-NMF followed by its learning.

6.3.1 Architecture

IG-MDSR-NMF is a deep neural network architecture made up of an input layer, s hidden layers and an output layer. Consider a given data matrix, $\mathbf{U} = [u_{pi}]_{m \times n'}$. We process $\mathbf{U}$ following the methodology described in Chapter 2 Section 2.3, to generate a matrix $\mathbf{X} = [x_{pi}]_{m \times n}$ with each element being non-negative. Each row of $\mathbf{X}$ is now used as input to the model.

The uppermost/last hidden layer of IG-MDSR-NMF has been envisioned to be the model's slender layer, with r nodes that extract $r < n'$ features. As mentioned in Section 3.3.1 of Chapter 3 there is no restriction of r with respect to the number of samples m. The responsibility of the output layer is to reconstruct the original data using the extracted features. The model may be separated into two phases - deconstruction and reconstruction. The input and s hidden layers comprise the deconstruction phase. The reconstruction phase consists of the slenderest and output layers. The latent representation of the input is obtained at the end of the deconstruction step. During the reconstruction process, the model tries to regenerate the input from this latent representation. As there is more than one deconstruction layer, but only one reconstruction layer, the design is referred to as Multiple Deconstruction Single Reconstruction (MDSR) deep learning architecture. Figure 6.1 depicts the architecture of IG-MDSR-NMF.

In IG-MDSR-NMF, we have employed three different types of activation functions. After preprocessing, data is put into the network. To follow the typical neural network architecture, an identity function has been used as the activation function for the input nodes. Thus, the output of any input node is the same as its input. The sigmoid function has been employed for the hidden layers, whereas the ReLU activation function

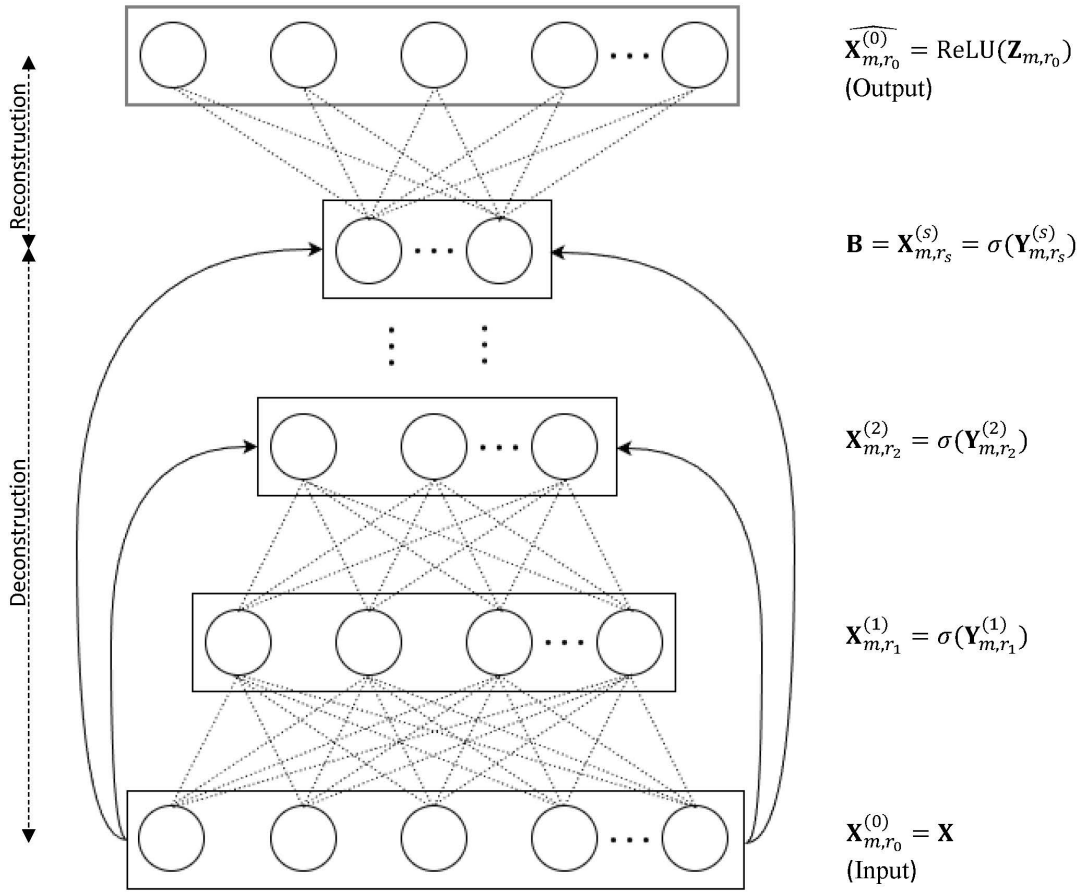

FIGURE 6.1: The architecture of IG-MDSR-NMF.

has been used for the output layer nodes. Sigmoid function maps the input of any interval $(-\infty, +\infty)$ to $(0, 1)$ as output. In contrast, all negative values are discarded with zero by the ReLU activation function. Since ReLU eliminates all negative components in order to meet non-negativity, the ReLU activation function experiences data loss, while the sigmoid function does not. Therefore, the model's non-negativity condition has been met by using the sigmoid and ReLU activation functions and the data loss issue has been avoided by using the sigmoid function in the hidden layers.

We consider the input layer as the 0^{th} layer of the model having $r_0 = n$ number of nodes. The input to this layer is denoted as $\mathbf{X}^{(0)} = [x^{(0)}_{pi_0}]_{m\times r_0}$, where $\mathbf{X}^{(0)} = \mathbf{X}$. The input layer connects the first hidden layer of the model having $r_1 < r_0$ number of nodes. The output of the first hidden layer is now expressed in matrix form for all samples as $\mathbf{X}^{(1)} = [x^{(1)}_{pi_1}]_{m\times r_1}$ and is defined as

$$\mathbf{X}^{(1)} = \sigma(\mathbf{Y}^{(1)}) \tag{6.3.1}$$

where, $\sigma(\mathbf{Y}^{(1)})$ is an $m \times r_1$ matrix and each element of the matrix is computed by applying the sigmoid activation function (σ) to the corresponding element of $\mathbf{Y}^{(1)}$, where $\mathbf{Y}^{(1)} = [y^{(1)}_{pi_1}]_{m \times r_1}$ is defined as

$$\mathbf{Y}^{(1)} = \mathbf{X}^{(0)}\mathbf{V}^{(1)} \tag{6.3.2}$$

Here $\mathbf{V}^{(1)} = [v^{(1)}_{i_0 i_1}]_{r_0 \times r_1}$ is the weight matrix between the input and the first hidden layers. The output of the first hidden layer connects to the second hidden layer containing r_2 nodes, where $r_2 < r_1$. Additionally, the input layer is also connected to this second hidden layer. The weight matrix between the first and second hidden layers is denoted by $\mathbf{V}^{(2)} = [v^{(2)}_{i_1 i_2}]_{r_1 \times r_2}$ and the weight matrix between the input and second hidden layers is $\tilde{\mathbf{V}}^{(2)} = [\tilde{v}^{(2)}_{i_0 i_2}]_{r_0 \times r_2}$. The output of the second hidden layer is expressed as $\mathbf{X}^{(2)} = [x^{(2)}_{pi_2}]_{m \times r_2}$ and is defined as

$$\mathbf{X}^{(2)} = \sigma(\mathbf{Y}^{(2)}) \tag{6.3.3}$$

where, $\mathbf{Y}^{(2)} = [y^{(2)}_{pi_2}]_{m \times r_2}$ is computed as

$$\mathbf{Y}^{(2)} = \mathbf{X}^{(1)}\mathbf{V}^{(2)} + \mathbf{X}^{(0)}\tilde{\mathbf{V}}^{(2)} \tag{6.3.4}$$

Similarly, the third hidden layer is connected to the output of the second hidden layer and the input layer. Eventually, the slenderest layer, i.e., the s^{th} hidden layer of the model is connected to the output of the $(s-1)^{th}$ hidden layer and the input layer. The number of nodes in this slenderest layer is $r_s = r$. It is to be noted that $r = r_s < r_{s-1} < ... < r_2 < r_1 < r_0 = n$. The weight matrix between the s^{th} and $(s-1)^{th}$ hidden layers is denoted by $\mathbf{V}^{(s)} = [v^{(s)}_{i_{s-1} i_s}]_{r_{s-1} \times r_s}$ and the weight matrix between the input and s^{th} hidden layers is $\tilde{\mathbf{V}}^{(s)} = [\tilde{v}^{(s)}_{i_0 i_s}]_{r_0 \times r_s}$. The output of this slenderest layer is denoted by $\mathbf{B} = \mathbf{X}^{(s)}$, where $\mathbf{X}^{(s)} = [x^{(s)}_{pi_s}]_{m \times r_s}$ is defined as

$$\mathbf{X}^{(s)} = \sigma(\mathbf{Y}^{(s)}) \tag{6.3.5}$$

Here, $\mathbf{Y}^{(s)} = [y^{(s)}_{pi_s}]_{m \times r_s}$ is determined as

$$\mathbf{Y}^{(s)} = \mathbf{X}^{(s-1)}\mathbf{V}^{(s)} + \mathbf{X}^{(0)}\tilde{\mathbf{V}}^{(s)} \tag{6.3.6}$$

The slenderest layer concludes the deconstruction phase of the model and marks the beginning of its reconstruction phase. The output of the slenderest layer, $\mathbf{B}$, represents

the low rank representation of the input $\mathbf{X}$. The reconstruction phase comprises a single reconstruction layer, producing the output of the model, i.e., the regenerated input $\widehat{\mathbf{X}} = \widehat{\mathbf{X}^{(0)}}$. Here, $\widehat{\mathbf{X}^{(0)}} = [\widehat{x_{pi_0}^{(0)}}]_{m \times r_0}$ is computed as

$$\widehat{\mathbf{X}^{(0)}} = ReLU(\mathbf{Z}) \tag{6.3.7}$$

where we get $\mathbf{Z} = [z_{pi_0}]_{m \times r_0}$ as

$$\mathbf{Z} = \mathbf{X}^{(s)}\mathbf{W} \tag{6.3.8}$$

Here, $\mathbf{W} = [w_{i_s i_0}]_{r_s \times r_0}$ represents the weight matrix between the slenderest layer and the output layer of the model.

The elements of the weight matrices $\mathbf{V}^{(1)}$, $\mathbf{V}^{(2)}$, ..., $\mathbf{V}^{(s)}$ and $\tilde{\mathbf{V}^{(2)}}$, $\tilde{\mathbf{V}^{(3)}}$, ..., $\tilde{\mathbf{V}^{(s)}}$ are unrestricted, while the elements of the weight matrix $\mathbf{W}$ must be non-negative to meet the non-negativity requirement of the NMF algorithm. The two non-negative components of the regenerated input matrix $\widehat{\mathbf{X}}$ are the slender layer output $\mathbf{B}$ and the weight matrix $\mathbf{W}$.

6.3.2 Learning

The objective of IG-MDSR-NMF is to find the best possible reconstruction ($\hat{\mathbf{X}}$) of the input matrix ($\mathbf{X}$) while factorizing $\mathbf{X}$ into $\mathbf{B}$ and $\mathbf{W}$. Thus, the objective function is defined as the mean square loss of the original and the regenerated input, i.e., we want to minimize $\|\mathbf{X} - \widehat{\mathbf{X}}\|_F$. Thus, the cost function Φ is defined as

$$\Phi = \frac{1}{2mn} \sum_{p=1}^{m} \sum_{j=1}^{n} (x_{pj} - \widehat{x}_{pj})^2 \tag{6.3.9}$$

IG-MDSR-NMF have been trained using a stochastic gradient descent method based on adaptive estimation of first-order and second-order moments employing the Adam optimisation technique [57]. The use of sigmoid activation function in the hidden layers of the network ensures the non-negativity requirement of the latent space representation ($\mathbf{B}$) of the input data matrix. Similarly, the ReLU activation function in the output layer guarantees the non-negativity of the regenerated input matrix $\hat{\mathbf{X}}$. In IG-MDSR-NMF, the non-negativity of the other factor matrix, i.e., the weight matrix $\mathbf{W}$ has been

assured by replacing the negative elements arising in the course of the updation of weights during backpropagation with zeros.

6.4 Experimental Results, Analysis and Discussion

There are two components to the presentation and justification of the performance of IG-MDSR-NMF in terms of the quality of low rank embedding generated by the model. First, by comparing its capability to maintain the local structure of data, the quality of dimension reduction using IG-MDSR-NMF has been assessed. The effectiveness of the low rank embedding in comparison to the original data has also been examined and verified, indicating the necessity for dimension reduction. Second, the discriminating competency of the dimensionally reduced dataset has been investigated for downstream analyses such as clustering and classification. Additionally, the statistical significance of the results produced by IG-MDSR-NMF in comparison to other dimension reduction methods has been examined.

The Xavier normal initialization approach [34] is an effective weight initialization technique for neural networks with sigmoid activation functions. The elements of all weight matrices in the proposed IG-MDSR-NMF model have been initialised using the same. IG-MDSR-NMF model has the input layer, s hidden layers and the output layer. In our implementation, we have considered $s = 3$. The number of training epochs is determined dynamically; training ends when the difference in the cost values in two consecutive epochs reaches a predetermined threshold.

6.4.1 Quantifying the quality of low dimensional embedding

Two approaches have been used to study the quality of low dimensional embedding by IG-MDSR-NMF. First, employing trustworthiness metrics to study the ability to preserve the local structure of the data, and second, comparing the effectiveness of dimension reduction by classification/cluster performance metrics with the original data.

6.4.1.1 Local structure preservation

Using the trustworthiness score, it has been calculated and compared how well IG-MDSR-NMF can retain the local structure of the data after dimension reduction compared to nine different dimension reduction techniques. The spider/star plot illustrates the result of the same (Figure 6.2).

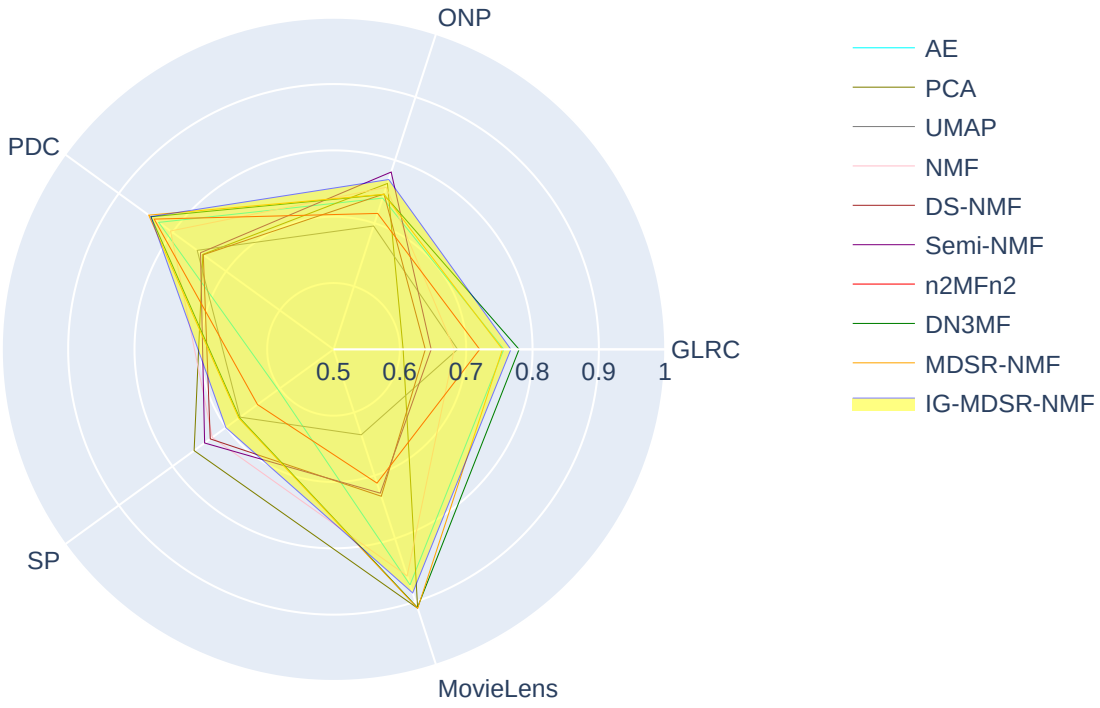

FIGURE 6.2: Trustworthiness scores of ten dimension reduction techniques including IG-MDSR-NMF.

The five axes of the plot relate to five datasets. The trustworthiness score of a dimension reduction method for a given dataset is represented by a point on that axis. Thus, with a dimension reduction strategy, there are five points on five axes, each representing one of five datasets. These points serve as polygon vertices. Figure 6.2 depicts ten polygons representing ten dimension reduction strategies. The area covered by a polygon demonstrates the effectiveness of a dimension reduction approach over all datasets combined. The algorithm's performance improves as the enclosed region grows larger. From the plot, we can note that IG-MDSR-NMF has beaten other dimension reduction techniques for the PDC dataset and for GLRC, ONP, SP and MovieLens datasets, the trustworthiness score of IG-MDSR-NMF is better than most of the others. The shaded

TABLE 6.1: Sum of trustworthiness scores of ten dimension reduction techniques including IG-MDSR-NMF on five datasets.

Dimension reduction techniques	Sum of trustworthiness scores
AE	4.55550328220533
PCA	4.38647684863791
UMAP	4.13297187263103
NMF	4.50973409888627
DS-NMF	4.22833659492512
Semi-NMF	4.29147059452664
n^2MFn^2	4.34404888837219
DN3MF	4.72960521062986
MDSR-NMF	4.68627962966105
IG-MDSR-NMF	**4.73123216172758**

polygon in Figure 6.2 represents the overall performance of IG-MDSR-NMF. To compute the area of the polygon, we add individual trustworthiness scores of the dimension reduction techniques for all five datasets. It can be observed from Table 6.1 that the sum of trustworthiness scores of IG-MDSR-NMF is the highest among all. Thus, the quality of low dimensional embedding produced by IG-MDSR-NMF is superior to that produced by the other dimension reduction methods.

6.4.1.2 Decision making: Comparison with the original data

The efficacy of dimension reduction using IG-MDSR-NMF has been evaluated by performing classification and clustering on the low dimensional embeddings produced by the model as well as the original data and then quantifying the results using various classification and cluster validity indexes. This study explains why dimension reduction is required, underlining the fact that low rank data representation improves its usability over the original.

Classification

Figures 6.3-6.7 presents the performance of IG-MDSR-NMF and original data in terms of classification. For the GLRC (Figure 6.3) and PDC (Figure 6.5) datasets, IG-MDSR-NMF generated low rank embedding has outperformed the original data for all four classifiers in terms of all four metrics. For ONP and MovieLens datasets, for ACC, CKS and MCC performance metrics, IG-MDSR-NMF has performed better than the original

dataset for three out of four classification algorithms (Figures 6.4, 6.7). In terms of FS, for the ONP dataset, the scoreline favouring IG-MDSR-NMF is four out of four and for the MovieLens dataset, the same count is three out of four. In the case of the SP dataset, the performance metric of original data is better than the low rank embedding produced by IG-MDSR-NMF on all occasions (Figure 6.6).

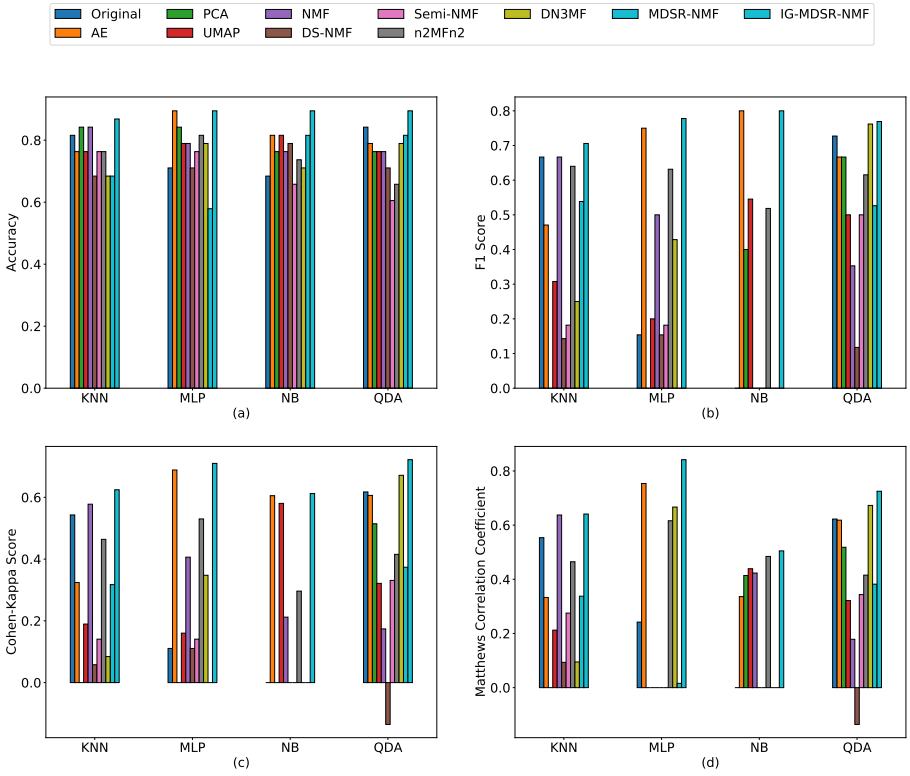

FIGURE 6.3: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the GLRC dataset by IG-MDSR-NMF and nine other dimension reduction techniques along with the original data.

Hence, it follows that the majority of the time, the projected data by IG-MDSR-NMF have outperformed the original data in terms of classification. This explains why dimension reduction and the capability to create low rank embeddings that preserve the fundamental properties of the data are necessary.

Clustering

The performance comparison of clustering done on the low dimensional embedding produced by IG-MDSR-NMF and the original data has been illustrated in Figures 6.8-6.12. For the ONP (Figure 6.9), SP (Figure 6.11) and MovieLens (Figure 6.12) datasets,

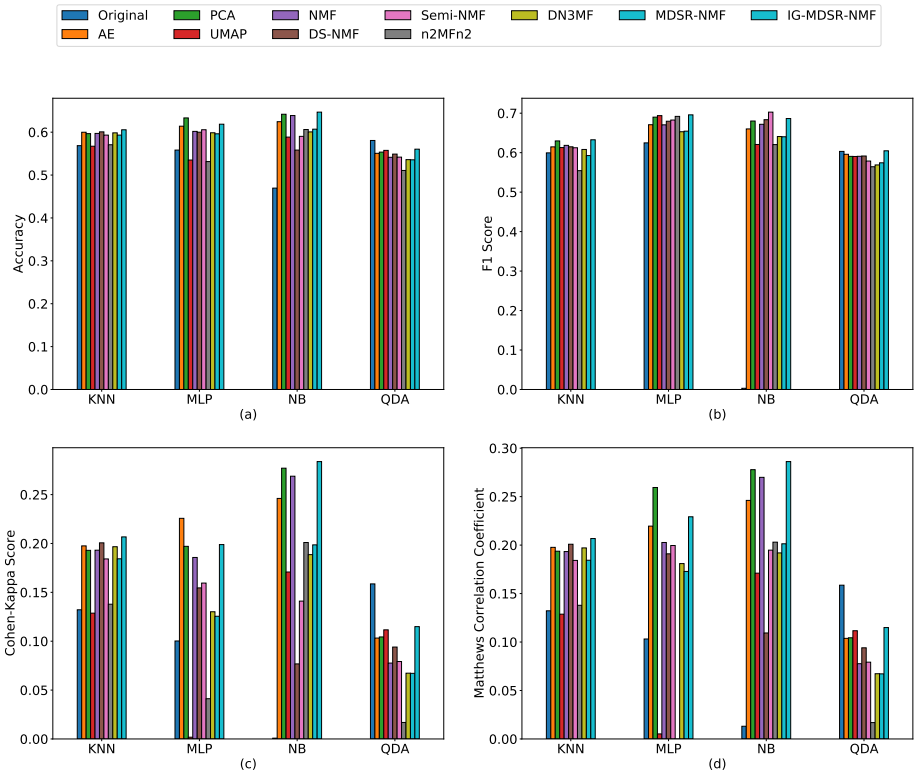

FIGURE 6.4: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the ONP dataset by IG-MDSR-NMF and nine other dimension reduction techniques along with the original data.

for all four cluster validity indexes, IG-MDSR-NMF has performed better than the original data with respect to all four clustering algorithms. For the GLRC (Figure 6.8) and PDC (Figure 6.10) datasets, for AMI cluster validity index, IG-MDSR-NMF has performed better than the original data for four out of four clustering algorithms. For the ARI metric, the performance score is three out of four in favour of IG-MDSR-NMF for both GLRC and PDC datasets. For the GLRC dataset, the performance score against four clustering methods in favour of IG-MDSR-NMF in terms of JI and NMI metrics is three and for the PDC dataset, the same count is two and four respectively.

Thus, it is proven that the low rank embedding produced by IG-MDSR-NMF performs significantly better in terms of clustering over the original, preserving the essential characteristics of the same. Thus, dimension reduction is required and justified.

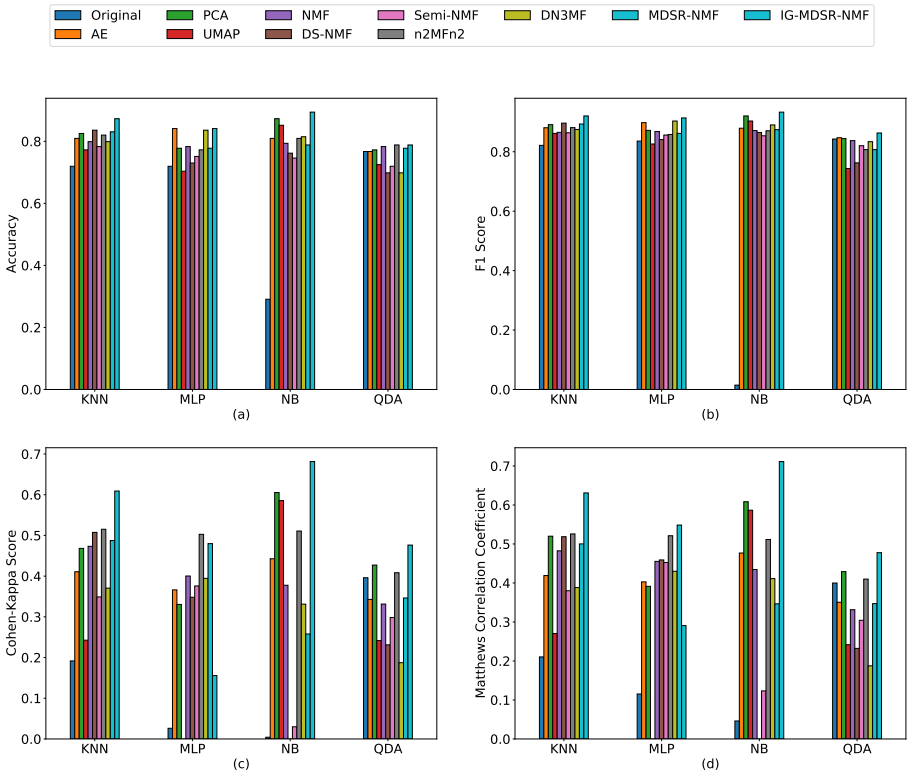

FIGURE 6.5: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the PDC dataset by IG-MDSR-NMF and nine other dimension reduction techniques along with the original data.

6.4.2 Downstream analyses and statistical significance: Comparison with other models

The efficiency of dimension reduction has been evaluated by performing classification and clustering on the low dimensional embedding resulting from IG-MDSR-NMF as well as that generated by the other nine dimension reduction methodologies. To quantify the same, a variety of measures assessing classification and cluster performances have been employed. In order to demonstrate the superiority of IG-MDSR-NMF over other dimension reduction algorithms in terms of generating output from an independent set of data, pairwise p-values have also been computed. The statistical significance of the results is justified by a p-value below a certain threshold. In this case, 0.05 has been chosen as the threshold. Thus, nine p-values have been calculated for a dataset, a classification/clustering algorithm and a classification/cluster validity index, comparing the performance of IG-MDSR-NMF to that of nine different dimension reduction methods. Four classification/cluster techniques produce a total of $9 \times 4 = 36$ p-values

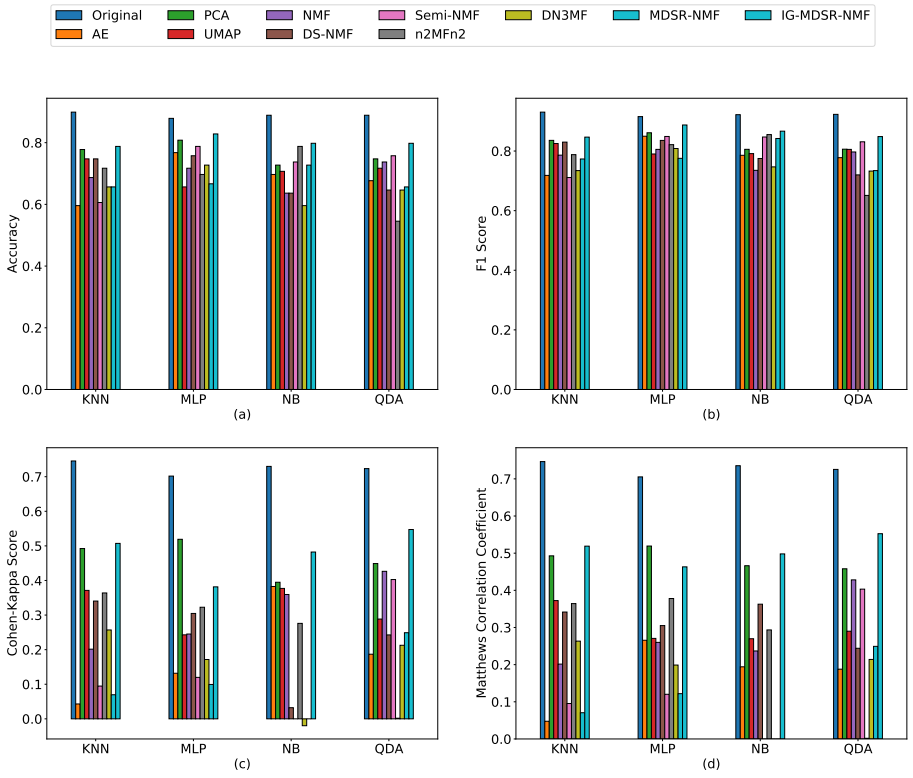

FIGURE 6.6: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the SP dataset by IG-MDSR-NMF and nine other dimension reduction techniques along with the original data.

for each validity index against each dataset. This section of the experiment tries to establish the superiority of IG-MDSR-NMF produced low dimensional embedding over other dimension reduction techniques in terms of a number of classification and clustering techniques.

Classification

While working with the IG-MDSR-NMF model for classification, the outcome has been depicted by Figures 6.3-6.7. The summary of the count of statistically significant p-values with respect to IG-MDSR-NMF has also been presented in Table 6.2.

For four classification techniques, IG-MDSR-NMF has always achieved the highest accuracy score for GLRC (Figure 6.3(a)), PDC (Figure 6.5(a)), SP (Figure 6.6(a)) and MovieLens (Figure 6.7(a)) dataset and three times for the ONP (Figure 6.4(a)) dataset.

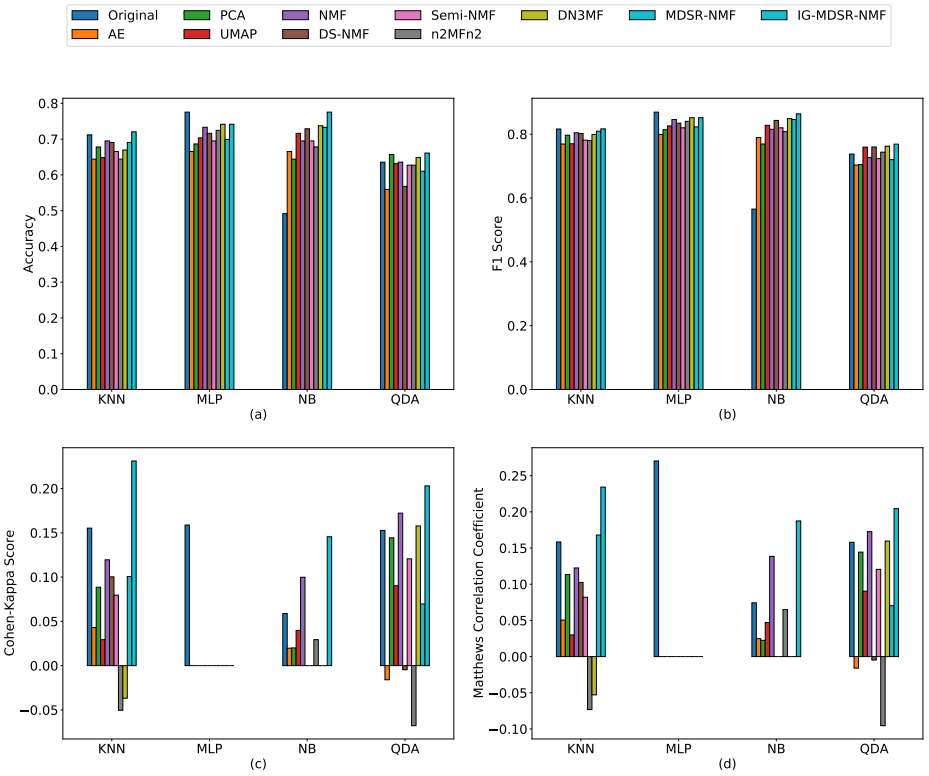

FIGURE 6.7: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the MovieLens dataset by IG-MDSR-NMF and nine other dimension reduction techniques along with the original data.

The same statistics hold when the classification performance metric is the F1 score (Figures 6.3(b), 6.4(b), 6.5(b), 6.6(b) and 6.7(b)). IG-MDSR-NMF has surpassed the others in terms of Cohen-Kappa score on the GLRC (Figure 6.3(c)) and MovieLens (Figure 6.7(c)) datasets using four out of four classification techniques, thrice for ONP (Figure 6.4(c)), PDC (Figure 6.5(c)) and SP (Figure 6.6(c)) datasets. When the Matthews Correlation Coefficient is used as the classification performance indicator, the outcome favouring IG-MDSR-NMF is the same as that of the Cohen-Kappa score (Figures 6.3(d), 6.4(d), 6.5(d), 6.6(d) and 6.7(d)).

The above explanation makes it quite evident that the accuracy score of the dimensionally reduced dataset using IG-MDSR-NMF has outperformed the others in most cases, across all five datasets and four types of classifiers. The frequency of the correctness of the model has been quantified in terms of accuracy. The F1 score, which is the harmonic mean of precision and recall, has been calculated along with accuracy. Figures 6.3-6.7 demonstrate that IG-MDSR-NMF has, in the majority of cases,

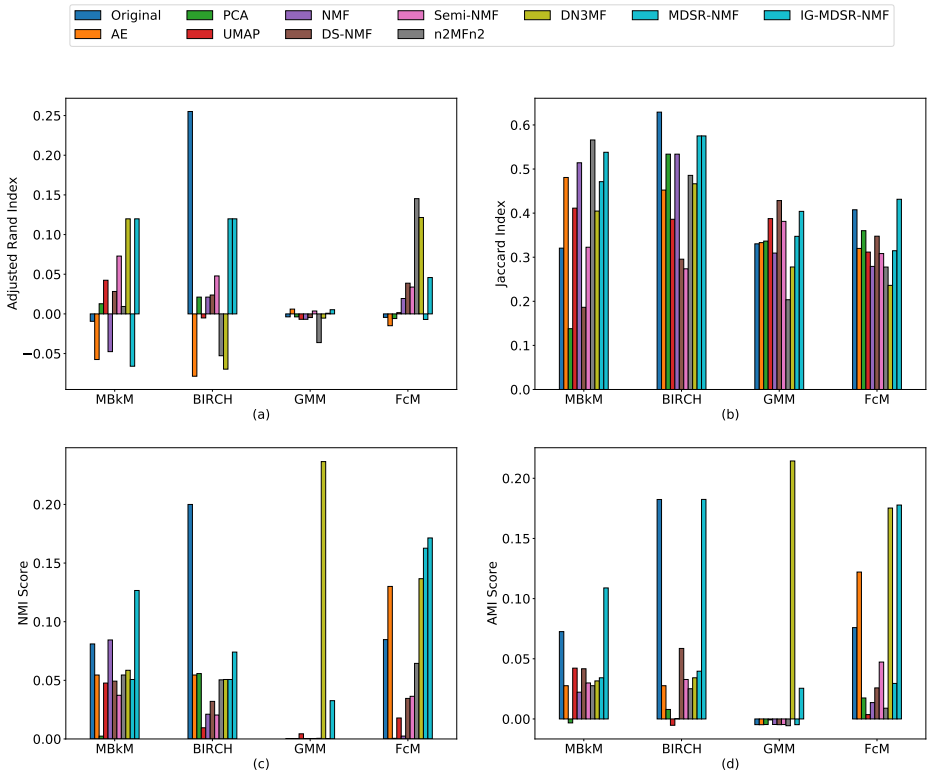

FIGURE 6.8: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the GLRC dataset by IG-MDSR-NMF and nine other dimension reduction techniques along with the original data.

outperformed other models in terms of F1 score. Therefore, IG-MDSR-NMF's superiority is supported by its F1 score and accuracy. In contrast, a statistical measure of inter-rater agreement is the Cohen-Kappa score. The pictorial representations demonstrate that IG-MDSR-NMF has, in most cases, outperformed the others and produced higher positive Cohen-Kappa ratings. Consequently, better scores may be obtained by inferring that IG-MDSR-NMF is able to preserve and acquire the intrinsic properties of the input. The quality of binary and multiclass classifications is evaluated using the Matthews Correlation Coefficient. The model's ability to retain the original dataset's class attributes in the modified dataset is represented by a higher MCC score, which also suggests better agreement. In terms of the MCC score, Figures 6.3-6.7 demonstrate that IG-MDSR-NMF has performed better than the other models. Thus, IG-MDSR-NMF outperforms the other dimension reduction methods in terms of intrinsic property preservation measures as well as statistical ones.

For each classification performance index against each dataset, out of a total of 36 p-values, the count of p-values less than the determined threshold (0.05) is presented in

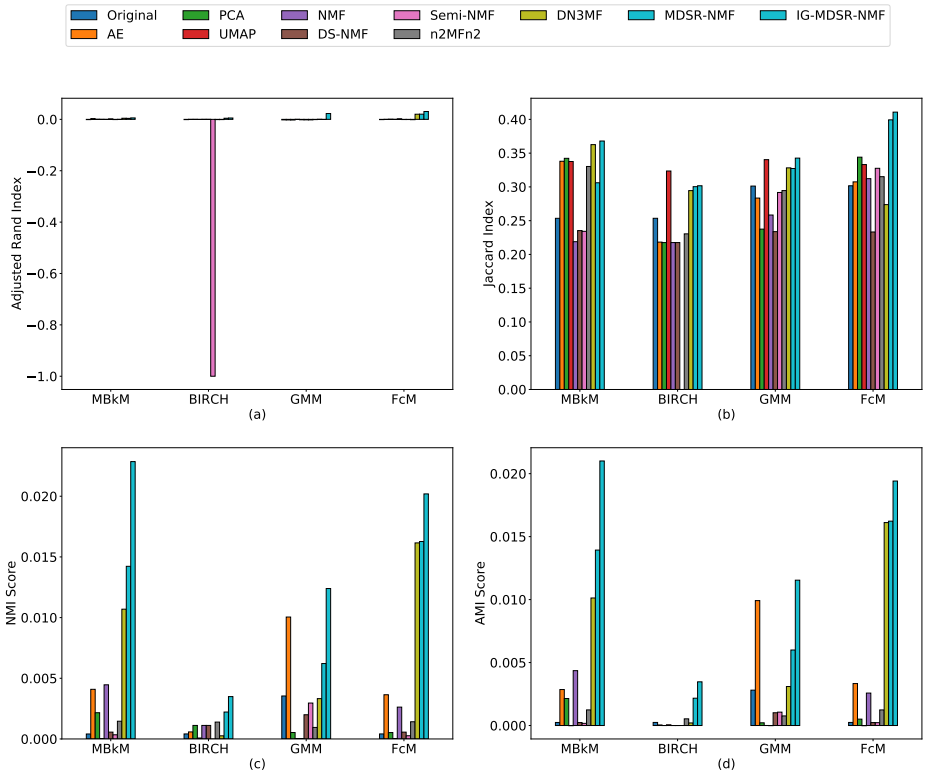

FIGURE 6.9: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the ONP dataset by IG-MDSR-NMF and nine other dimension reduction techniques along with the original data.

TABLE 6.2: The summary of the count (out of 36) of statistically significant p-values for each classification performance metric against each dataset with respect to IG-MDSR-NMF.

Dataset	ACC	FS	CKS	MCC
GLRC	34	31	34	34
ONP	25	32	24	24
PDC	27	27	34	35
SP	24	22	19	20
MovieLens	35	31	27	27

Table 6.2. The above statistics indubitably quantify the quality of low rank embedding produced by IG-MDSR-NMF over others.

Clustering

For clustering purposes with the IG-MDSR-NMF model, the Figures 6.8-6.12 present the outcome. Table 6.3 provides an overview of the count of statistically significant

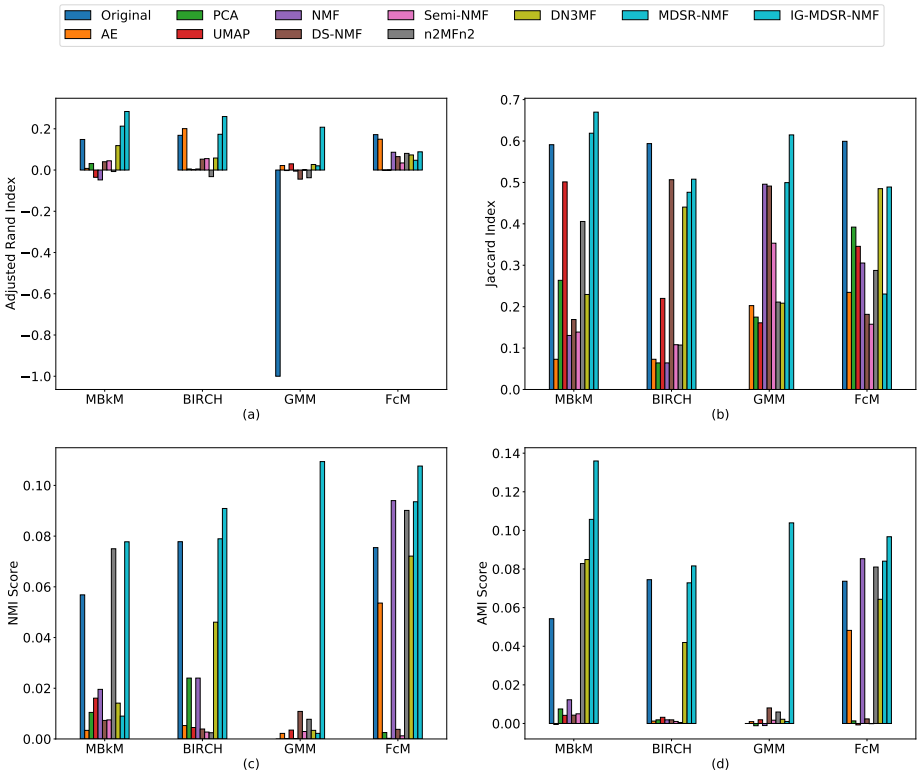

FIGURE 6.10: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the PDC dataset by IG-MDSR-NMF and nine other dimension reduction techniques along with the original data.

p-values for IG-MDSR-NMF for clustering.

IG-MDSR-NMF has achieved the highest performance score for the Adjusted Rand index for the ONP (Figure 6.9(a)) and MovieLens (Figure 6.12(a)) datasets for all four clustering approaches considered here. This count is two out of four for the GLRC (Figure 6.8(a)) dataset and three out of four for the PDC (Figure 6.10(a)) and SP (Figure 6.11(a)) datasets. When using the Jaccard Index as the cluster validity estimator, IG-MDSR-NMF has outperformed the others in four out of four clustering algorithms on the PDC (Figure 6.10(b)) and SP (Figure 6.11(b)) datasets. This value ranks three out of four for the ONP (Figure 6.9(b)) and MovieLens (Figure 6.12(b)) datasets and for the GLRC (Figure 6.8(b)) dataset this count is two out of four. When the cluster validity index is Normalized Mutual Information score, IG-MDSR-NMF has outperformed others four out of four times for the PDC (Figure 6.10(c)) dataset, thrice for GLRC (Figure 6.8(c)), ONP (Figure 6.9(c)), SP (Figure 6.11(c)) and MovieLens (Figure 6.12(c)) datasets. IG-MDSR-NMF projected transformed space has achieved the highest Adjusted Mutual Information score among the other dimension reduction techniques four

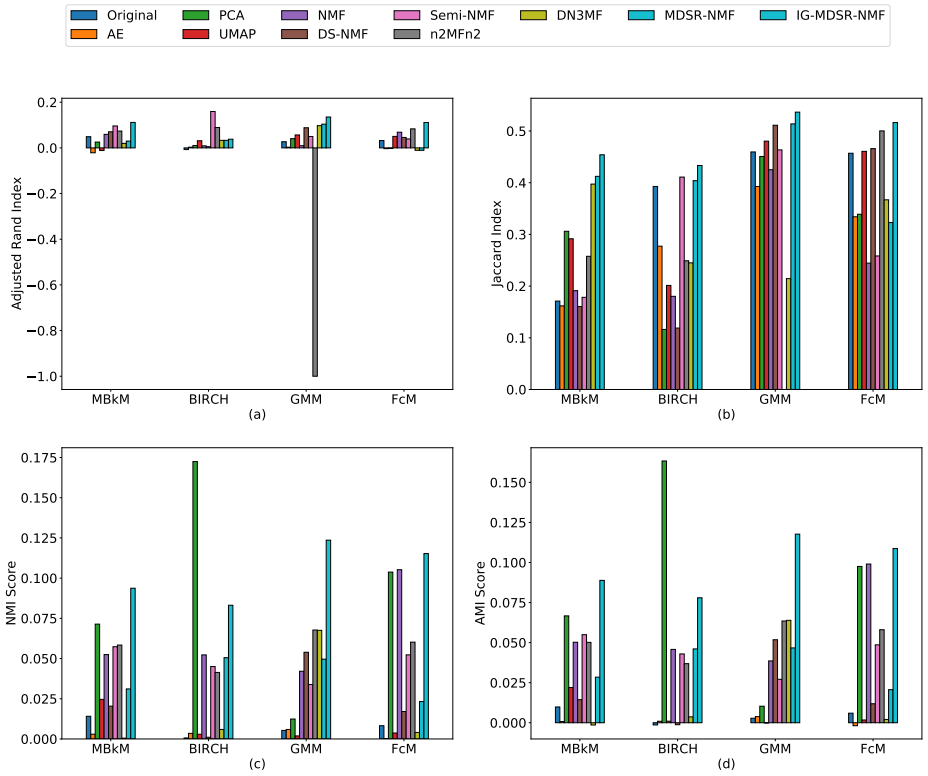

FIGURE 6.11: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the SP dataset by IG-MDSR-NMF and nine other dimension reduction techniques along with the original data.

times on the PDC (Figure 6.10(d)) and MovieLens (Figure 6.12(d)) datasets for all the four clustering algorithms. The count is three for the GLRC (Figure 6.8(d)), ONP (Figure 6.9(d)) and SP (Figure 6.11(d)) datasets.

The similarity of the two data clusters can be determined using the Adjusted Rand Index (ARI). Figures 6.8-6.12 demonstrate that IG-MDSR-NMF outperformed other dimension reduction approaches in terms of the ARI score across five datasets and four clustering algorithms. The Jaccard Index is used to compare the similarities of two sets. IG-MDSR-NMF has outperformed the others in terms of the Jaccard Index as well. Thus, it might be stated that IG-MDSR-NMF learned the elemental properties of the input and correctly mapped them to a low rank representation. NMI is defined as the scaling of outcomes in $[0,1]$ by normalizing the Mutual Information score. This measure is not adjusted for chance. In contrast, the AMI score remains constant regardless of the class or cluster label permutation. Figures 6.8-6.12 demonstrate that IG-MDSR-NMF outperforms other dimension reduction techniques in terms of NMI and AMI scores. The better performance of IG-MDSR-NMF justifies that the low rank

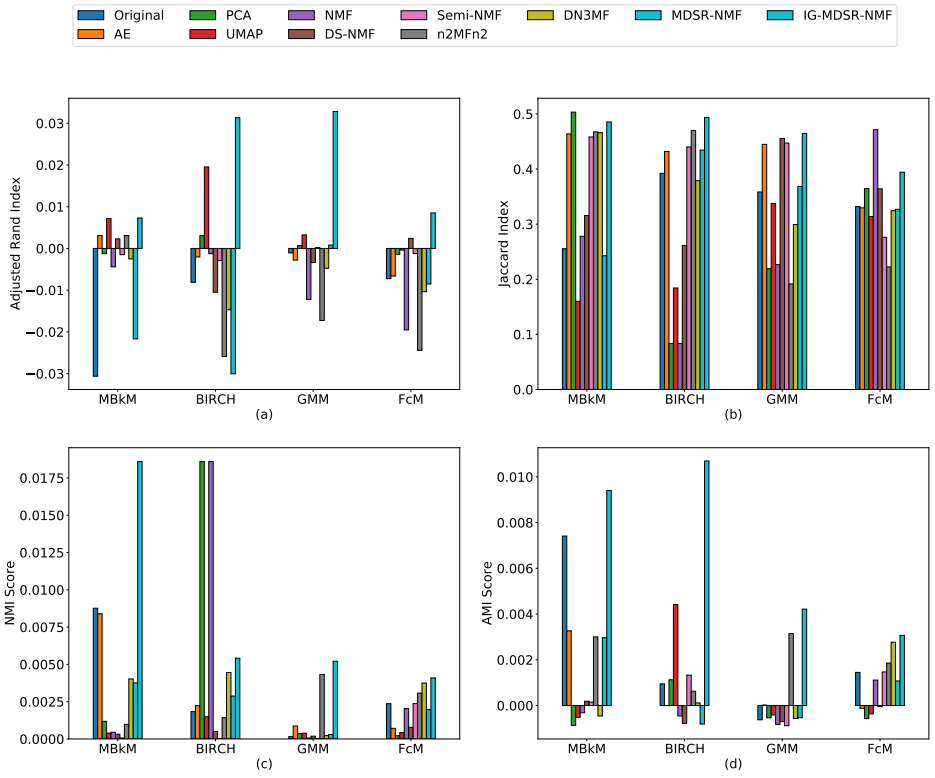

FIGURE 6.12: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the MovieLens dataset by IG-MDSR-NMF and nine other dimension reduction techniques along with the original data.

TABLE 6.3: The summary of the count (out of 36) of statistically significant p-values for each cluster performance metric against each dataset with respect to IG-MDSR-NMF.

Dataset	ARI	JI	NMI	AMI
GLRC	14	13	29	25
ONP	26	33	35	34
PDC	33	33	36	36
SP	14	20	23	22
MovieLens	33	34	29	32

representation of datasets using IG-MDSR-NMF has been able to better preserve the underlying features of the original data than the other techniques evaluated here.

Out of a total of 36 p-values for each cluster validity index against each dataset, Table 6.3 displays the count of p-values that fall below the decided threshold, i.e., 0.05. The aforementioned tally unequivocally demonstrates how good low rank embedding produced by IG-MDSR-NMF is compared to others.

6.4.3 Discussion

In terms of the trustworthiness score (Figure 6.2 and Table 6.1), IG-MDSR-NMF has clearly outperformed not only other dimension reduction algorithms considered here also all of the three previously developed models (n^2MFn^2, DN3MF and MDSR-NMF).

There is a total of $5 \times 4 \times 4 = 80$ performance scores for each dimension reduction approach for five datasets, four classification algorithms and four classification performance measures. On 57 out of 80 instances, it can be observed that the IG-MDSR-NMF projected datasets have outperformed the original data as far as classification is concerned. However, in a performance comparison with the other dimension reduction methods, IG-MDSR-NMF has achieved the highest rating of 72 times out of 80. When compared to other dimension reduction methods, IG-MDSR-NMF has been able to recover the moderate performances of the previously developed models for both ONP and SP datasets. For the remaining cases, the performance of IG-MDSR-NMF is similar to that of MDSR-NMF. Thus the efficacy of IG-MDSR-NMF has improved over the previously built models.

In addition to outperforming the original and other dimensionally reduced datasets produced by various dimension reduction methods, the low rank embeddings generated by IG-MDSR-NMF for various datasets are also demonstrated to be statistically significant in terms of the comparative p-values they have resulted in. For all classifiers and classification metrics, the total number of statistically significant results for IG-MDSR-NMF is 133 out of 144 $(4 \times 4 \times 9)$ for the GLRC dataset. For the PDC, ONP, SP and MovieLens datasets, the corresponding counts are 105, 123, 85 and 120. Therefore, it is proven that IG-MDSR-NMF is more effective than other dimension reduction algorithms in generating low dimensional embeddings that are statistically significant.

As with classification, the competency of IG-MDSR-NMF has been demonstrated through the application of four clustering methods and four cluster validity metrics across five datasets for clustering. When comparing the clustering performance to the original data, IG-MDSR-NMF has demonstrated superior performance 74 times out of 80 potential scenarios. Among the other dimension reduction methods, IG-MDSR-NMF has a notable 65 out of 80 superiority count over the others. Given the above discussion, it is evident that IG-MDSR-NMF has shown to be more effective than the

other dimension reduction techniques that are being considered here in most of the situations.

Not only the low rank embeddings generated by IG-MDSR-NMF for different datasets outperform the original and other dimensionally reduced datasets produced by different dimension reduction methods, but their comparative p-values also demonstrate that they are statistically significant. The overall number of statistically significant performance counts connected to IG-MDSR-NMF for all clustering techniques and cluster validity indexes combined is 81 out of 144 $(4 \times 4 \times 9)$ for the GLRC dataset. The counts of the PDC, ONP, SP and MovieLens datasets are 128, 138, 79 and 128, respectively. Therefore, it has been demonstrated that IG-MDSR-NMF produces low dimensional embeddings that are statistically significant and more effective than other dimension reduction techniques.

The datasets that we have experimented on are, as mentioned in Section 3.4.3 Chapter 3, split into two sets based on the relationship between the number of samples and characteristics. In terms of dimension reduction and invariance to the relation between the number of samples and characteristics, IG-MDSR-NMF has shown superiority for both kinds of datasets. Moreover, neither the number of samples nor the characteristics may restrict the value of the reduced dimension. IG-MDSR-NMF is distinguished from a number of other popular dimension reduction techniques by these characteristics. It is, therefore, demonstrated that IG-MDSR-NMF is not restricted by input dimension and has broad applicability. Consequently, IG-MDSR-NMF has achieved better results for different classification and clustering methodologies on two different types of datasets than the other nine state-of-the-art dimension reduction methods. Thus, the unique input-guided architecture places itself apart from others.

6.5 Convergence analysis

Based on the experimental results, we aim to establish the convergence of the proposed IG-MDSR-NMF model. The convergence plot for the IG-MDSR-NMF model is shown in Figure 6.13. The plot illustrates the variation of the cost function Φ against iteration for all five datasets. Overall, the decreasing nature of the cost over time validates that the model converges. It can also be observed from the plot that the initial cost value for

all the datasets starts from a high position and after a few initial epochs, the value of the cost function has almost reached a straight line parallel to the horizontal axis. That is, there are very nominal changes in the cost value. Thus, we can conclude that the model has converged. Figure 6.13 depict the cost versus iteration plot for the GLRC, ONP, ONP, SP and MovieLens datasets with $r = 115$, $r = 33$, $r = 154$, $r = 13$ and $r = 319$ respectively.

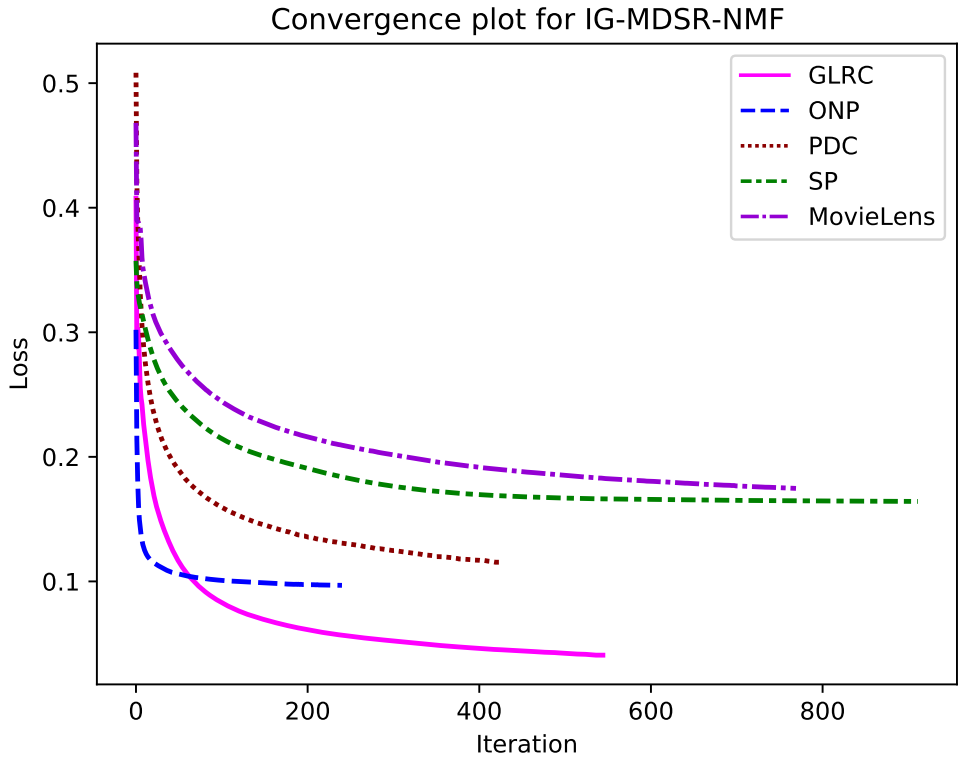

FIGURE 6.13: Loss vs. iteration plots of IG-MDSR-NMF for GLRC, ONP, PDC, SP and MovieLens dataset.

6.6 Analysis of computational complexity

The computational complexity of IG-MDSR-NMF has been measured in terms of the number of operations performed. IG-MDSR-NMF has $s + 1$ deconstruction layers including the input layer and one reconstruction layer. The input, i.e., each row of $\mathbf{X}$, travels through the identity function (activation function) at the input layer, hence the computational complexity is $\mathcal{O}(mr_0)$. The following step is defined in equation (6.3.2), where the complexity is $\mathcal{O}(mr_0r_1)$. The value of $\mathbf{Y}^{(1)}$ is now sent via the activation

function σ (equation 6.3.1), with the complexity of this operation being $\mathcal{O}(mr_1)$. The following step using equation (6.3.4) includes $\mathcal{O}(mr_1r_2 + mr_0r_2)$ operations. Thereafter, similar to the previous layer the value of $\mathbf{Y}^{(2)}$ passes through the activation function σ (equation 6.3.3) contributing $\mathcal{O}(mr_2)$ to the overall complexity. Proceeding this way the model finally computes $\widehat{\mathbf{X}^{(0)}}$ (equations 6.3.7 and 6.3.8) and for this the computational complexity is $\mathcal{O}(mr_sr_0 + mr_0)$. Thus, the forward pass comprises $\mathcal{O}(mr_0 + mr_0r_1 + mr_1 + mr1r_2 + mr_0r_2 + ... + mr_sr_0 + mr_0)$ operations. As mentioned before, $n = r_0 > r_1 > r_2 > ... > r_s = r$. Removing the lower order terms, the computational cost of the forward pass is $\mathcal{O}(mr_0r_1)$. The computational complexity of Φ (equation 6.3.9) is $\mathcal{O}(mr_0)$. The major task of backward propagation is to update the weights. With similar arguments, we can conclude that the backward pass entails $\mathcal{O}(mr_0r_1)$ operations. Thus, the computational cost of an epoch is $\mathcal{O}(mr_0r_1)$. The complexity for t such epochs is $\mathcal{O}(tmr_0r_1)$. Hence, the overall computational complexity of IG-MDSR-NMF is $\mathcal{O}(tmnr_1)$.

6.7 Conclusions

There are numerous techniques to dimensionally reduce a huge dataset with a large number of attributes. In this chapter, we have combined the benefits of NMF, a conventional matrix factorization technique and deep learning for dimension reduction to develop a novel model (IG-MDSR-NMF) of neural networks. The way how a human being learns a new concept by frequently referring to the original text to maintain the proper direction of learning and enhancing the effectiveness of knowledge gain has inspired the design of IG-MDSR-NMF. At each step of hierarchical learning, IG-MDSR-NMF is assisted by the input and hence the models are called “Input Guided". IG-MDSR-NMF has been constructed in such a manner that it mimics the factorization behaviour of the classic NMF technique.

Extensive analysis of the quality of dimension reduction of five popular datasets using IG-MDSR-NMF has been performed to compare with that of nine other dimension reduction algorithms. These nine dimension reduction techniques include six NMF-based approaches and three conventional dimension reduction algorithms. Preserving the local shape of the original data in the altered space is considered a benchmark of the

dimension reduction algorithms. IG-MDSR-NMF has outclassed other dimension reduction methodologies in terms of local shape preservation. The discriminating ability of low rank embedding by IG-MDSR-NMF has been tested and validated by comparing their performances using both classification and clustering over that of the original dataset, which in turn justifies the need for dimension reduction. During experimentation, a total of four classification algorithms, four classification performance measures, four clustering methods and four cluster validity indexes have been used. The findings also support IG-MDSR-NMF's superiority over other dimension reduction methodologies evaluated here.

For greater comprehension, the substantial result set has been well supported by its statistical significance. The outcomes clearly show that, in terms of intrinsic property preservation principles as well as statistical performance, IG-MDSR-NMF is better than other dimension reduction methods. Experiments have also been carried out to demonstrate the convergence of IG-MDSR-NMF. The computational complexity of IG-MDSR-NMF has also been studied in terms of the number of elementary operations performed.

Throughout this thesis, we have used the NMF technique encapsulated with deep learning architecture to factorize the input matrix into two non-negative matrices. Although our only objective is to find a low dimensional non-negative representation of the input matrix, we have been confronting the models to produce a pair of non-negative factors as output. The second factor can only be used together with the low rank embedding to regenerate the original matrix, which is not at all our objective. In other words, the objective of the present thesis is to obtain low rank embedding of the original data hybridizing the notion of NMF and neural networks. For achieving better low rank embedding, let us relax the second factor matrix from being non-negative, to introduce Relaxed Non-negative Matrix Factorization (RNMF), a novel kind of matrix factorization technique.

Chapter 7

Input Guided Multiple Deconstruction Single Reconstruction neural network for Relaxed Non-negative Matrix Factorization (IG-MDSR-RNMF)

7.1 Introduction

In the previous chapter, IG-MDSR-NMF has been designed following the human trait of learning by referring back to the original data in due course of hierarchical fragmented learning. The unique pair of output factors of IG-MDSR-NMF follow the non-negativity criteria resembling the true notion of NMF. The main objective of this thesis is to find a low rank approximation of the input data to get rid of the curse of dimensionality problem. Constraints such as the non-negativity of both the factor matrices limit the learning of the model to some extent. On the other hand, relaxing the non-negativity criteria of the coefficient matrix does not hamper the overall aim of the model, rather may improve the quality of low dimensional embedding. The input as well as its low rank approximation, i.e., the basis matrix adheres to the non-negativity constraint, only the non-negativity restriction of the coefficient matrix is relaxed. This

novel idea has been called “Relaxed Non-negative Matrix Factorization (RNMF)". The model realizing the same has been named Input Guided Multiple Deconstruction Single Reconstruction neural network for Relaxed Non-negative Matrix Factorization (IG-MDSR-RNMF) [28].

The low dimensional embedding quality of IG-MDSR-RNMF has been justified by measuring the extent of preservation of the input shape. It has also been determined and established that dimension reduction over the original data is necessary. By testing on five datasets for both classification and clustering, the superiority of the low rank approximation by IG-MDSR-RNMF has also been confirmed over ten popular dimension reduction strategies. The effectiveness of IG-MDSR-RNMF has been demonstrated on several dataset types for downstream analyses in terms of clustering and classification. Additionally, the statistical significance of the results has been determined.

The remainder of the chapter is structured as follows. The motivation behind the IG-MDSR-RNMF architecture and learning is explained in Section 7.2. In Section 7.3, the design and derivation of the corresponding learning rules have been provided. Following the experimentation process outlined in Chapter 2, the findings are depicted in Section 7.4 along with a sufficient analysis. In Sections 7.5 and 7.6, the IG-MDSR-RNMF convergence analysis and the computational complexity study are described respectively. Section 7.7 brings the chapter to a conclusion.

7.2 Motivation behind Architecture and Learning

IG-MDSR-RNMF has been build on top of IG-MDSR-NMF. IG-MDSR-NMF is a true realization of NMF, where both the resulting factors, i.e., the basis matrix and the coefficient matrix follow the non-negativity criteria. A relaxed version of the model, called Input Guided Multiple Deconstruction Single Reconstruction neural network for Relaxed Non-negative Matrix Factorization (IG-MDSR-RNMF) has been designed, where the non-negativity constraint of the coefficient matrix has been relaxed. That is, in IG-MDSR-RNMF, the basis matrix adheres to the non-negativity constraint, whereas the coefficient matrix is unconstrained.

Our main objective is to find a low dimensional non-negative representation of the input matrix $\mathbf{X}$. The low rank representation $\mathbf{B}$ should embed relevant information

in a computationally efficient manner. The second factor **W** is only used to regenerate the input **X**. In IG-MDSR-RNMF we have relaxed the non-negativity constraint of the coefficient matrix **W**, whereas the non-negativity constraint of the basis matrix **B** remains intact. Finding a set of non-negative factor matrices is like confronting the estimation of the low dimensional embedding by enforcing the non-negativity criteria of the coefficient matrix. The relaxation does not compromise with the ultimate objective of finding the non-negative low dimensional representation of the input. Whereas, relaxation helps manoeuvre the learning more efficiently to find the best possible low rank representation of the input. Relaxing the non-negativity criteria of one of the factor matrices has been presented as a new class of matrix factorization technique called, Relaxed Non-negative Matrix Factorization (RNMF).

7.3 IG-MDSR-RNMF

In this section, we have described the architecture of IG-MDSR-RNMF followed by its learning.

7.3.1 Architecture

The architecture of IG-MDSR-RNMF is the same as that of IG-MDSR-NMF (Section 6.3.1 of Chapter 6); only the non-negativity requirement of the weight matrix **W** has been relaxed. That is the slender layer output **B** will follow the non-negativity requirement but, the weight matrix **W** connecting the slender layer and the output layer will be unconstrained.

7.3.2 Learning

The objective of IG-MDSR-RNMF is to find the best possible reconstruction ($\hat{\mathbf{X}}$) of the input matrix (**X**) while factorizing **X** into **B** and **W**. Thus, the objective function is defined as the mean square loss of the original and the regenerated input, i.e., we want to minimize $\|\mathbf{X} - \widehat{\mathbf{X}}\|_F$. Thus, the cost function Φ is defined as

$$\Phi = \frac{1}{2mn} \sum_{p=1}^{m} \sum_{j=1}^{n} (x_{pj} - \widehat{x}_{pj})^2 \tag{7.3.1}$$

IG-MDSR-RNMF have been trained using a stochastic gradient descent method based on adaptive estimation of first-order and second-order moments employing the Adam optimisation technique [57]. The use of sigmoid activation function in the hidden layers of the network ensures the non-negativity requirement of the latent space representation ($\mathbf{B}$) of the input data matrix. Similarly, the ReLU activation function in the output layer guarantees the non-negativity of the regenerated input matrix $\hat{\mathbf{X}}$. No restriction has been applied on the weight matrix $\mathbf{W}$.

7.4 Experimental Results, Analysis and Discussion

The performance of IG-MDSR-RNMF has been demonstrated and validated in two parts. To begin, the quality of dimension reduction using IG-MDSR-RNMF has been assessed by comparing its capacity to retain the local structure of data. The requirement for dimension reduction, or the efficacy of low rank embedding compared to the original data, has also been investigated and verified. Second, the discriminative capacity of the dimensionally reduced dataset is investigated for downstream analyses like classification and clustering. The statistical significance of IG-MDSR-RNMF results compared to other dimension reduction strategies has also been investigated.

The Xavier normal initialization approach [34] is an effective weight initialization technique for neural networks with sigmoid activation functions. The elements of all weight matrices in the proposed IG-MDSR-RNMF model have been initialised using the same. IG-MDSR-RNMF model has the input layer, s hidden layers and the output layer. In our implementation, we have considered $s = 3$. The number of training epochs is determined dynamically; training ends when the difference in the cost values in two consecutive epochs reaches a predetermined threshold.

7.4.1 Quantifying the quality of low dimensional embedding

A pair of approaches have been used to examine the quality of low dimensional embedding by IG-MDSR-RNMF. First, employing trustworthiness metrics to study the model's ability to retain its local structure in low dimensional embedding, and second, comparing the classification/cluster performance metrics of the low dimensional

embedding by IG-MDSR-RNMF to the original data to determine how effective the dimension reduction was.

7.4.1.1 Local structure preservation

Using the trustworthiness score, the superiority of IG-MDSR-RNMF over ten other dimension reduction techniques in terms of maintaining the local structure of the data after dimension reduction has been calculated and compared. The spider/star plot illustrates the same (Figure 7.1).

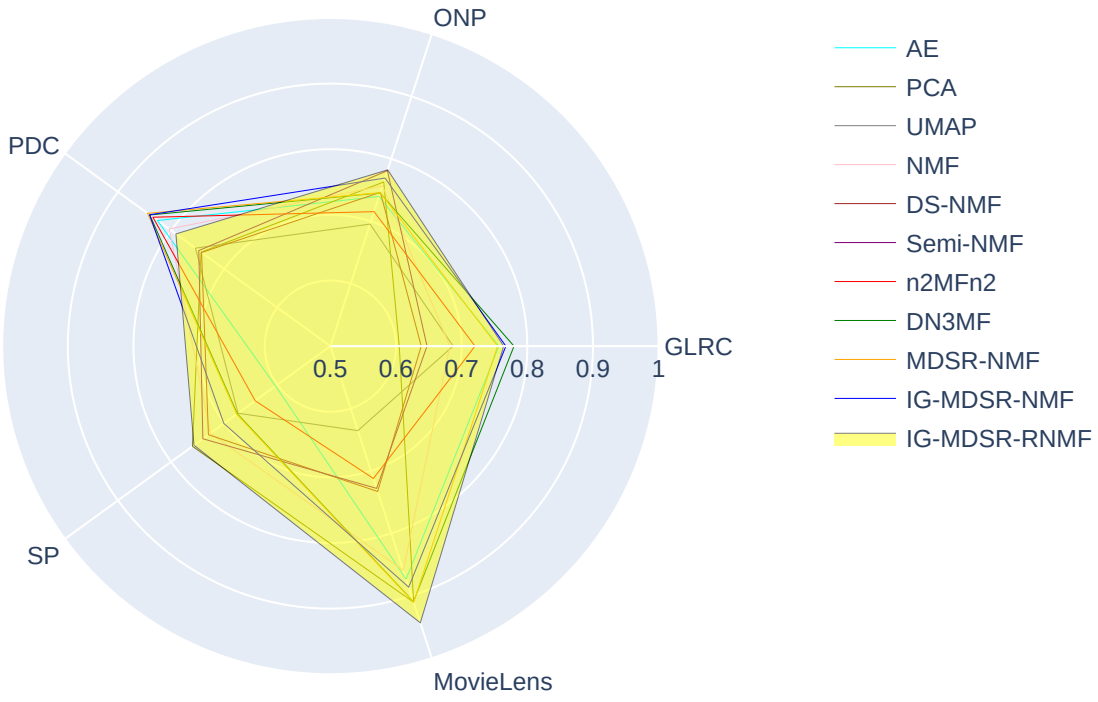

FIGURE 7.1: Trustworthiness scores of eleven dimension reduction techniques including IG-MDSR-RNMF.

There are five axes in the plot, each representing a dataset. The trustworthiness score of a dimension reduction strategy for a certain dataset is represented as a point on that axis. Thus, for a dimension reduction strategy, there are five points on five axes, that correspond to five datasets. These points can be considered as polygon vertices. Figure 7.1 shows eleven polygons representing eleven dimension reduction strategies. The area covered by a polygon demonstrates the success of a dimension reduction technique across all datasets. The algorithm performance is justified as the covered area increases. From the depiction, we can note that IG-MDSR-RNMF has beaten other dimension reduction techniques for the ONP and MovieLens datasets. For the SP dataset,

TABLE 7.1: Sum of trustworthiness scores of eleven dimension reduction techniques including IG-MDSR-RNMF on five datasets.

Dimension reduction techniques	Sum of trustworthiness scores
AE	4.55550328220533
PCA	4.38647684863791
UMAP	4.13297187263103
NMF	4.50973409888627
DS-NMF	4.22833659492512
Semi-NMF	4.29147059452664
n^2MFn^2	4.34404888837219
DN3MF	4.72960521062986
MDSR-NMF	4.68627962966105
IG-MDSR-NMF	4.73123216172758
IG-MDSR-RNMF	**4.80171249545558**

the performance of IG-MDSR-RNMF is just a notch below the highest one. The performance is comparable with GLRC and average with the PDC dataset. The shaded polygon in Figure 7.1 represents the overall performance of IG-MDSR-RNMF. To compute the area of the polygon, we add individual trustworthiness scores of the dimension reduction techniques for all five datasets. It can be observed from Table 7.1 that the sum of trustworthiness scores of IG-MDSR-RNMF is the highest among all. Thus, the quality of low dimensional embedding produced by IG-MDSR-RNMF is superior to that produced by the other dimension reduction methods.

7.4.1.2 Decision making: Comparison with the original data

Using a variety of classification and cluster validity measures, the effectiveness of dimension reduction by IG-MDSR-RNMF has been assessed by performing classification and clustering on both the original data and the low dimensional embedding generated by IG-MDSR-RNMF. This part of the experiment highlights the need for dimension reduction by showing how the low rank representation of the data improves its usability over the original one.

Classification

Figures 7.2-7.6 presents the performance of IG-MDSR-RNMF and original data in terms of classification. For the GLRC (Figure 7.2) dataset, IG-MDSR-RNMF generated low

rank embedding has outperformed the original data for all four classifiers in terms of all four metrics. For the PDC (Figure 7.4) datasets, IG-MDSR-RNMF generated low rank embedding has outperformed the original data for all four classifiers in terms of ACC, FS and MCC metrics. For the CKS metric, the same count is three out of four. For ONP and MovieLens datasets, for all four performance metrics, IG-MDSR-RNMF has performed better than the original dataset for three out of four classification algorithms (Figures 7.3, 7.6). In terms of ACC, FS and CSK, for the SP dataset, the scoreline favouring IG-MDSR-RNMF is two out of four (Figure 7.5) and in terms of MCC score, IG-MDSR-RNMF has performed better than the original only once.

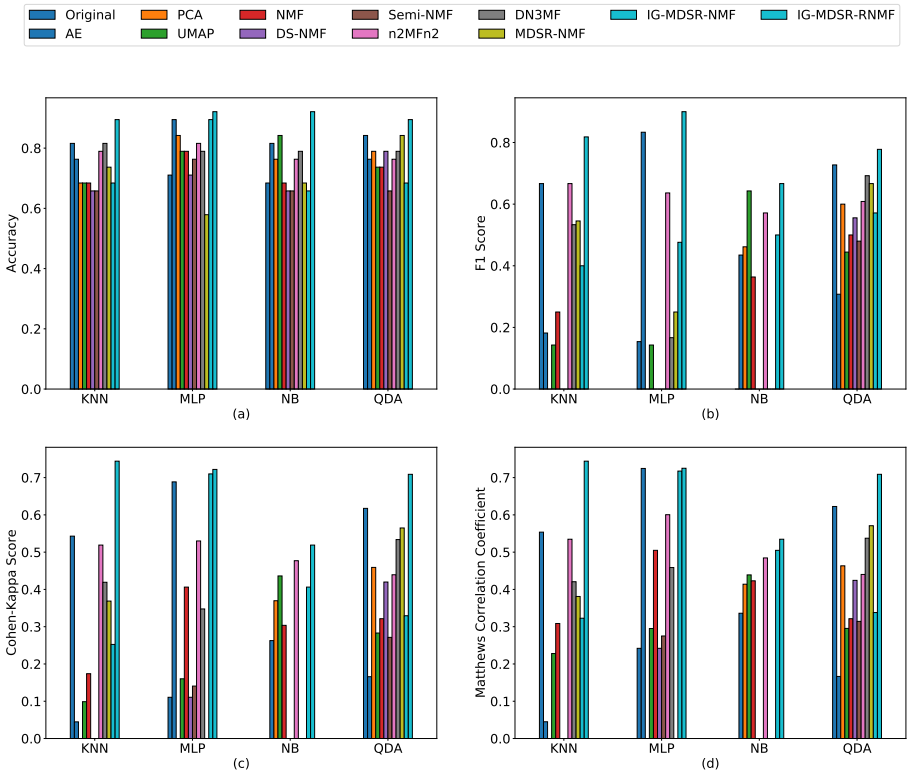

FIGURE 7.2: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the GLRC dataset by IG-MDSR-RNMF and ten other dimension reduction techniques along with the original data.

Thus, it is clear that in the majority of situations, IG-MDSR-RNMF projected data outperformed the original data in terms of classification. This justifies the requirement for dimension reduction as well as the capacity to generate low rank embeddings that preserve data's elemental features.

Clustering

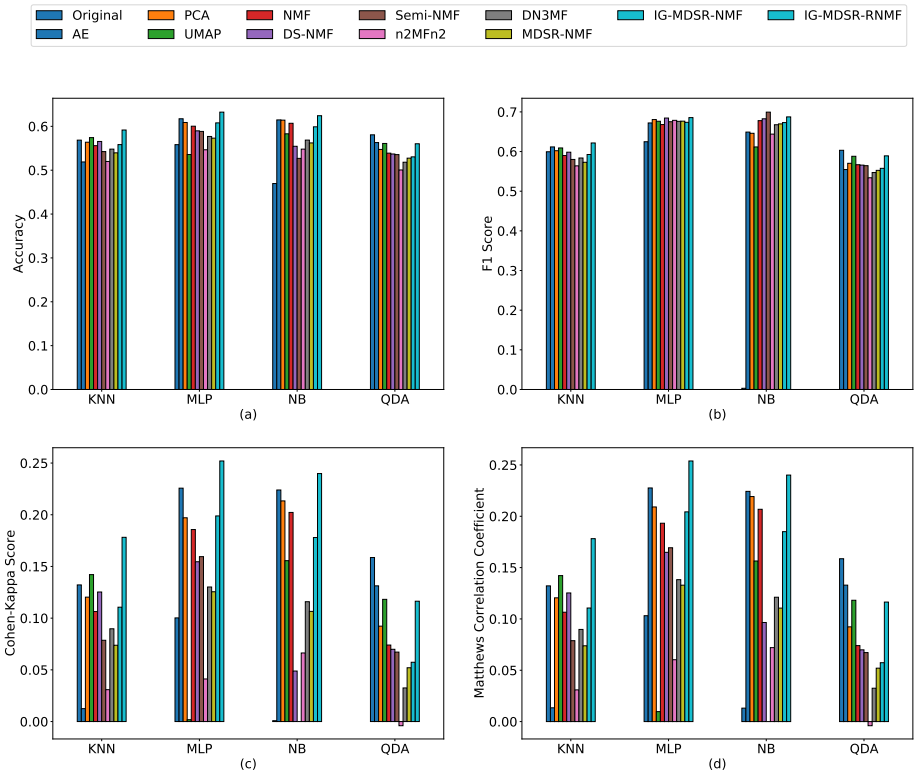

FIGURE 7.3: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the ONP dataset by IG-MDSR-RNMF and ten other dimension reduction techniques along with the original data.

The performance comparison of clustering done on the low dimensional embedding produced by IG-MDSR-RNMF and the original data has been illustrated in Figures 7.7-7.11. For the GLRC (Figure 7.7) and SP (Figure 7.10) datasets, for all four cluster validity indexes, IG-MDSR-RNMF has performed better than the original data with respect to all four clustering algorithms. For the MovieLens (Figure 7.11) dataset IG-MDSR-RNMF has performed better for all four clustering algorithms in terms of all cluster validity indexes considered here except NMI, where the same statistics favouring IG-MDSR-RNMF is three out of four. For NMI and AMI cluster validity indexes, IG-MDSR-RNMF has performed better than the original data for four out of four clustering algorithms on the PDC (Figure 7.9) dataset and for the ARI and JI metric these numbers are two out of four. For the ONP dataset (Figure 7.8) the performance score is four out of four in favour of IG-MDSR-RNMF for ARI and JI metrics and for the remaining two metrics i.e., NMI and AMI, the same count is three out of four.

As a result, it has been demonstrated in terms of clustering performance that the low

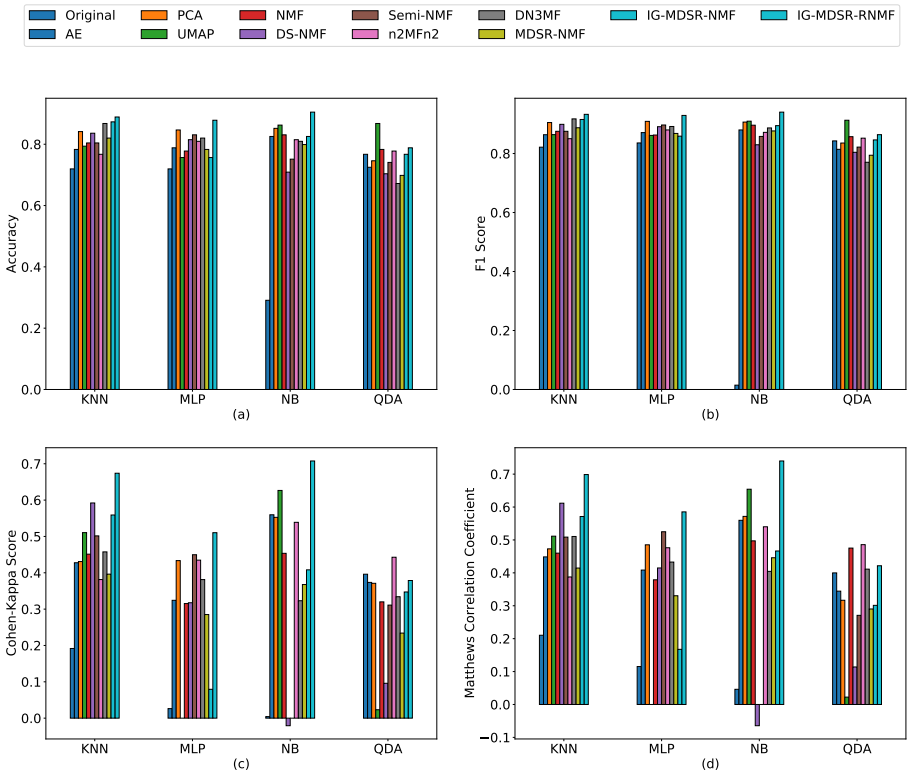

FIGURE 7.4: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the PDC dataset by IG-MDSR-RNMF and ten other dimension reduction techniques along with the original data.

rank embedding produced by IG-MDSR-RNMF is significantly better at retaining the essential features of the original data. Therefore, dimension reduction is also essential and warranted.

7.4.2 Downstream analyses and statistical significance: Comparison with other models

The efficiency of dimension reduction has been examined by performing classification and clustering on the low dimensional embedding produced by IG-MDSR-RNMF as well as the other nine dimension reduction approaches. Several measures assessing classification and cluster performance have been used to quantify the same. Pairwise p-values have also been determined to demonstrate the superiority of IG-MDSR-RNMF over other dimension reduction algorithms in terms of generating output from an independent set of data. A p-value less than a specific threshold indicates that the results are statistically significant. The threshold value used in this case is 0.05. In order to

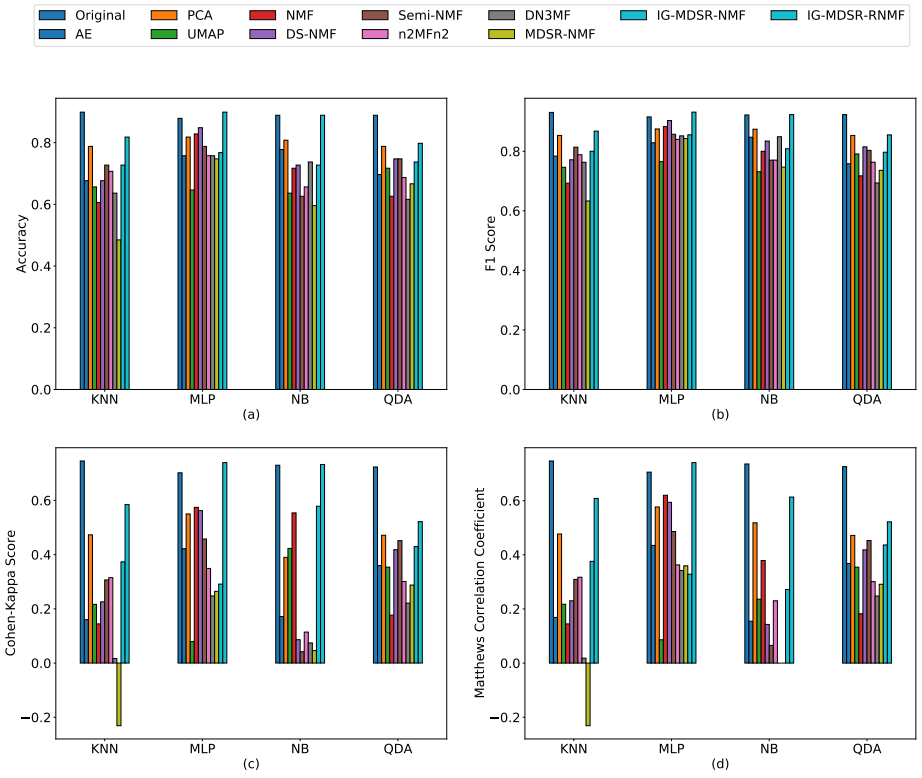

FIGURE 7.5: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the SP dataset by IG-MDSR-RNMF and ten other dimension reduction techniques along with the original data.

compare the performance of IG-MDSR-RNMF with that of ten different dimension reduction techniques, ten p-values have been calculated for a dataset, a classification/-clustering methodology and a classification/cluster validity index. There will be a total of $10 \times 4 = 40$ p-values for each classification/cluster validity index against each dataset as there are four classification/cluster methods. This part of the experiment aims to ascertain whether IG-MDSR-RNMF is a better dimension reduction technique than others in terms of different classification and clustering algorithms.

Classification

While working with the IG-MDSR-RNMF model for classification, the outcome has been depicted by Figures 7.2-7.6. The summary of the count of statistically significant p-values with respect to IG-MDSR-RNMF has also been presented in Table 7.2.

For four classification techniques, IG-MDSR-RNMF has always achieved the highest

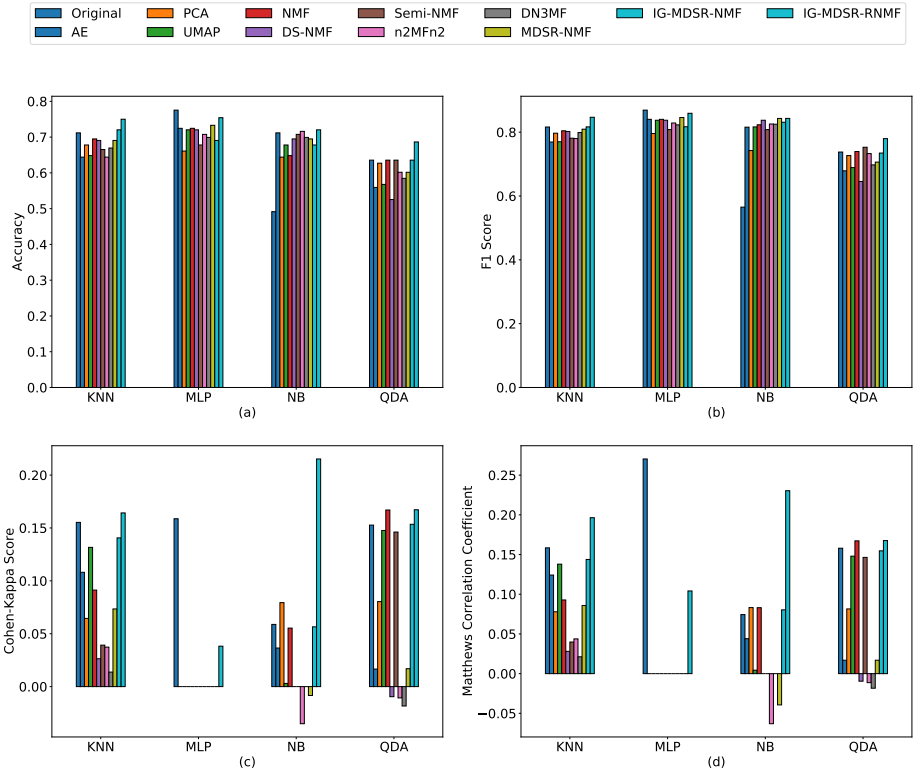

FIGURE 7.6: Mean performance scores of the classification algorithms on the dimensionally reduced dataset instances of the MovieLens dataset by IG-MDSR-RNMF and ten other dimension reduction techniques along with the original data.

performance score for all four metrics for GLRC (Figure 7.2), SP (Figure 7.5) and MovieLens (Figure 7.6) datasets and three times for the ONP (Figure 7.3) and PDC (Figure 7.4) datasets.

The previous explanation demonstrates that in the majority of cases, for all five datasets and four types of classifiers, the accuracy score of the dimensionally reduced dataset using IG-MDSR-RNMF outperformed the others. Accuracy measures how often the model is right. Along with accuracy, we calculated the F1 score, which is the harmonic mean of precision and recall. The majority of the time, IG-MDSR-RNMF has done better in terms of F1 score than other models (Figures 7.2-7.6). Therefore, the accuracy and F1 score of IG-MDSR-RNMF justify its superiority. Conversely, the Cohen-Kappa score serves as a statistical measure of inter-rater agreement. The graphical representations demonstrate that IG-MDSR-RNMF has outperformed the others in most cases and produced higher positive Cohen-Kappa scores. Consequently, it can be said that IG-MDSR-RNMF achieves superior results by being able to preserve and pick up on the intrinsic characteristics of the input. The Matthews Correlation Coefficient

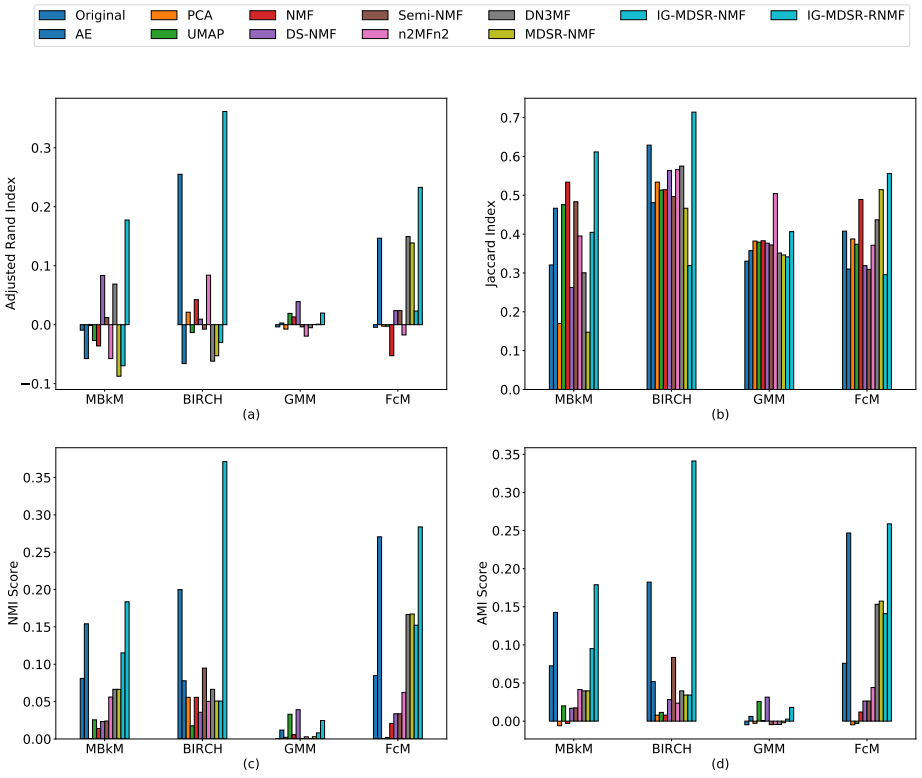

FIGURE 7.7: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the GLRC dataset by IG-MDSR-RNMF and ten other dimension reduction techniques along with the original data.

TABLE 7.2: The summary of the count (out of 40) of statistically significant p-values for each classification performance metric against each dataset with respect to IG-MDSR-RNMF.

Dataset	ACC	FS	CKS	MCC
GLRC	30	30	31	31
ONP	27	32	27	27
PDC	31	36	31	33
SP	35	35	34	34
MovieLens	15	16	33	34

evaluates the quality of binary and multiclass classifications. A higher MCC score indicates better agreement, implying that the model can also keep the class features of the original dataset in the transformed dataset. The Figures 7.2-7.6 reveal that IG-MDSR-RNMF beat the other models in terms of MCC score. The accompanying discussion illustrates that IG-MDSR-RNMF outperforms other dimension reduction techniques in terms of both statistical and intrinsic property preservation criteria.

For each classification performance index against each dataset, out of a total of 40

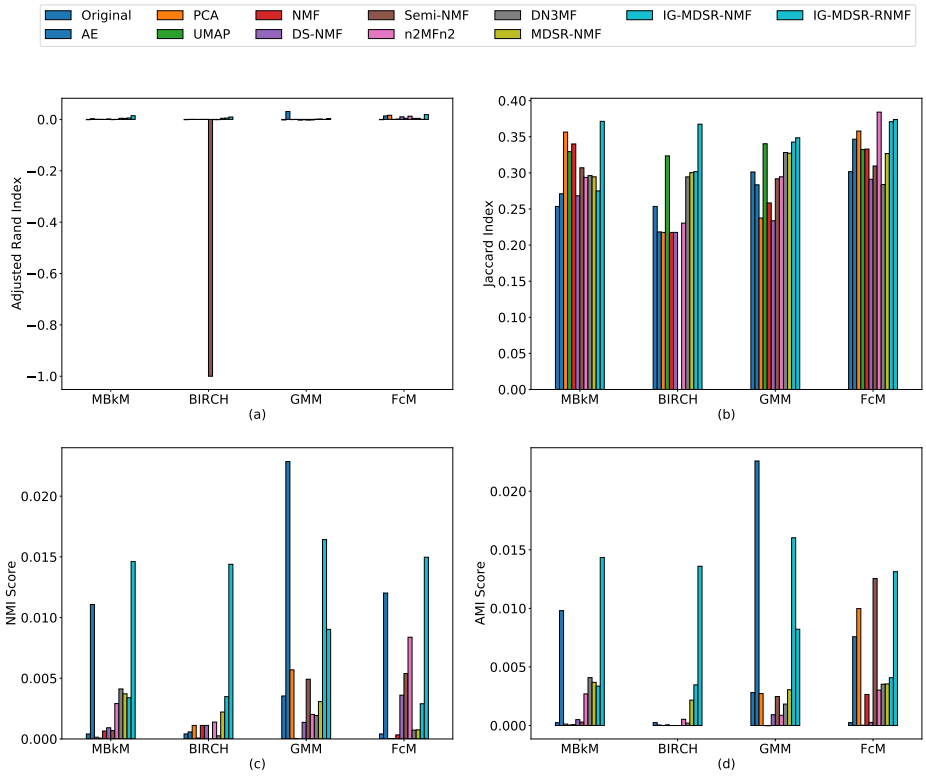

FIGURE 7.8: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the ONP dataset by IG-MDSR-RNMF and ten other dimension reduction techniques along with the original data.

p-values, the count of p-values less than the determined threshold (0.05) is presented in Table 7.2. The above statistics indubitably quantify the quality of low rank embedding produced by IG-MDSR-RNMF over others.

Clustering

For clustering purposes with the IG-MDSR-RNMF model, the Figures 7.7-7.11 present the outcome. Table 7.3 provides an overview of the count of statistically significant p-values for IG-MDSR-RNMF for clustering.

IG-MDSR-RNMF has achieved the highest performance score for the Adjusted Rand index for the SP (Figure 7.10(a)) and MovieLens (Figure 7.11(a)) datasets for all four clustering approaches considered here. This count is three out of four for the GLRC (Figure 7.7(a)) dataset and two out of four for the PDC (Figure 7.9(a)) and ONP (Figure 7.8(a)) datasets. When using the Jaccard Index as the cluster validity estimator,

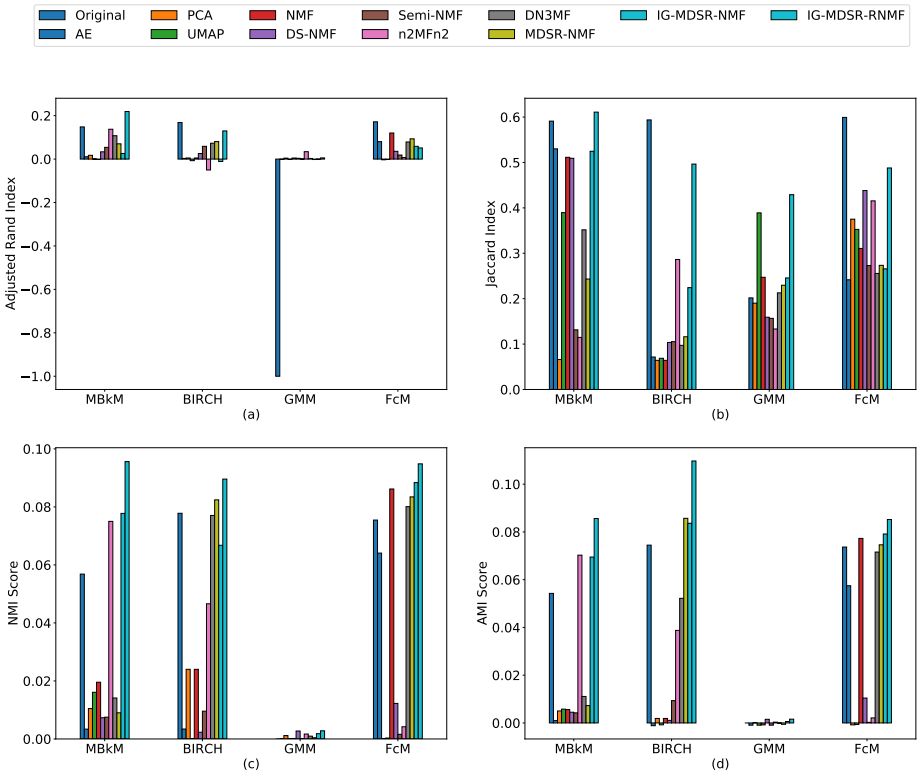

FIGURE 7.9: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the PDC dataset by IG-MDSR-RNMF and ten other dimension reduction techniques along with the original data.

IG-MDSR-RNMF has outperformed the others in four out of four clustering algorithms on the PDC (Figure 7.9(b)) dataset. This value ranks three out of four for the GLRC (Figure 7.7(b)), ONP (Figure 7.8(b)) and SP (Figure 7.10(b)) datasets. For the MovieLens (Figure 7.11(b)) dataset, this count is one out of four. When the cluster validity index is Normalized Mutual Information score, IG-MDSR-RNMF has outperformed others four out of four times for the PDC (Figure 7.9(c)) and SP (Figure 7.10(c)), thrice for the GLRC (Figure 7.7(c)) and MovieLens (Figure 7.11(c)) and once for the ONP (Figure 7.8(c)) dataset. IG-MDSR-RNMF projected transformed space has achieved the highest Adjusted Mutual Information score among the other dimension reduction techniques four times on the PDC (Figure 7.9(d)), SP (Figure 7.10(d)) and MovieLens (Figure 7.11(d)) datasets for all the four clustering algorithms. The count is three for the GLRC (Figure 7.7(d)) and ONP (Figure 7.8(d)) datasets.

The measure of the similarity between two data clusterings can be done using Adjusted Rand Index (ARI). In terms of ARI score, IG-MDSR-RNMF has outperformed other dimension reduction strategies across five datasets and four clustering algorithms,

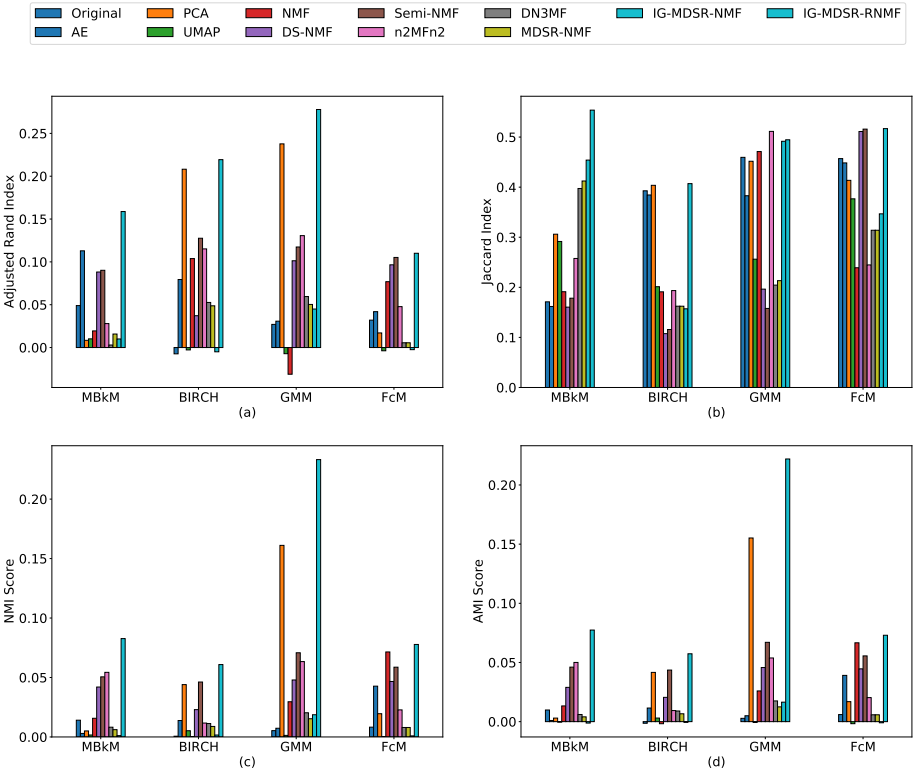

FIGURE 7.10: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the SP dataset by IG-MDSR-RNMF and ten other dimension reduction techniques along with the original data.

as shown in Figures 7.7-7.11. The Jaccard Index is used to compare two sets in terms of similarity. In relation to the Jaccard Index, IG-MDSR-RNMF has done better than the others. Therefore, one might claim that IG-MDSR-RNMF has successfully learned the essential properties of the input and mapped them to a low rank representation. The Normalization of Mutual Information score to scale the results in $[0, 1]$ is known as NMI. This measure is uncorrected for chance. In contrast, the AMI score remains constant regardless of the permutation of the class or cluster label. Figures 7.7-7.11 demonstrate that IG-MDSR-RNMF outperforms other considered dimension reduction techniques in terms of both NMI and AMI scores. The better performance of IG-MDSR-RNMF reveals that the low rank representation of datasets using IG-MDSR-RNMF has been able to preserve the underlying features of the original data better than the other techniques evaluated here.

Out of a total of 40 p-values for each cluster validity index against each dataset, Table 7.3 displays the count of p-values that fall below the decided threshold, i.e., 0.05.

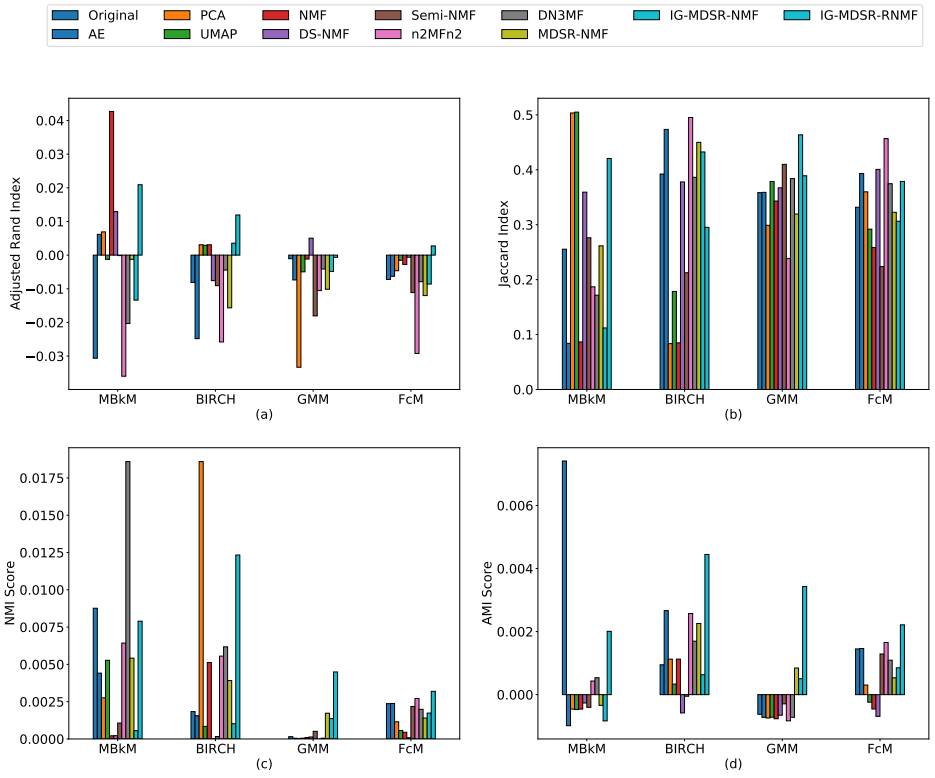

FIGURE 7.11: Mean performance scores of the clustering algorithms on the dimensionally reduced dataset instances of the MovieLens dataset by IG-MDSR-RNMF and ten other dimension reduction techniques along with the original data.

TABLE 7.3: The summary of the count (out of 40) of statistically significant p-values for each cluster performance metric against each dataset with respect to IG-MDSR-RNMF.

Dataset	ARI	JI	NMI	AMI
GLRC	33	34	32	32
ONP	27	30	17	19
PDC	27	37	35	33
SP	22	11	27	27
MovieLens	27	14	30	38

The aforementioned tally unequivocally demonstrates how good low rank embedding produced by IG-MDSR-RNMF is compared to others.

7.4.3 Discussion

Just like the previously designed models, the capacity of IG-MDSR-RNMF to retain the local structure of data over the others has also been investigated using the trustworthiness score metric. IG-MDSR-RNMF's performance has been compared separately to

ten different dimension reduction techniques, including n^2MFn^2, DN3MF, MDSR-NMF and IG-MDSR-NMF, across all five datasets. It has been observed that IG-MDSR-RNMF has demonstrated its goodness in dimension reduction by preserving the granular relationship of data and outperforming the other dimension reduction techniques.

For five datasets, four classification methods, and four classification performance measures, each dimension reduction algorithm has a total of $5 \times 4 \times 4 = 80$ performance scores. It can be observed that IG-MDSR-RNMF projected datasets outperformed the real data on 62 out of 80 instances as far as the downstream analyses are concerned. When compared to other dimension reduction methods, IG-MDSR-RNMF had the highest performance rating of 72 out of 80. It can be observed that the performance of IG-MDSR-RNMF over the original SP dataset has been improved over the previous models. Thus, the superiority of IG-MDSR-RNMF over the others is unquestionable.

The low rank embeddings produced by IG-MDSR-RNMF for various datasets not only outperform the original and other dimensionally reduced datasets produced by different dimension reduction methods, but they have also been shown to be statistically significant in terms of the comparative p-values. The aggregate count of statistically significant findings connected to IG-MDSR-RNMF for all classifiers and classification measures is 122 out of 160 $(4 \times 4 \times 10)$ for the GLRC dataset. These counts for the PDC, ONP, SP and MovieLens datasets are 113, 131, 138 and 98, respectively. Thus, IG-MDSR-RNMF has outperformed other dimension reduction methods in terms of producing statistically meaningful low-dimensional embeddings.

Similar to classification, four clustering techniques and four cluster validity measures have been applied across five datasets to demonstrate IG-MDSR-RNMF's competence. When comparing clustering performance to the original data, IG-MDSR-RNMF has outperformed the original data by 73 times out of 80. IG-MDSR-RNMF has outperformed other dimension reduction techniques by 62 out of 80. Given the preceding discussion, it is evident that in the vast majority of cases, IG-MDSR-RNMF has outperformed the other dimension reduction methodologies reviewed here.

The low rank embeddings generated by IG-MDSR-RNMF for different datasets have been shown to be statistically significant based on the comparative p-values they produce, in addition to outperforming the original and other dimensionally reduced datasets

produced by different dimension reduction methods. Out of 160 $(4 \times 4 \times 10)$, for the GLRC dataset, 131 cases, out of all clustering techniques and cluster validity indices, are statistically significant performant for IG-MDSR-RNMF. The counts of 93, 132, 87 and 109 have been found for the PDC, ONP, SP and MovieLens datasets respectively. Thus, it has been demonstrated that IG-MDSR-RNMF produces low dimensional embeddings that are statistically significant and are more efficient than other dimension reduction techniques.

7.5 Convergence Analysis

Based on the experimental results, we aim to establish the convergence of the proposed IG-MDSR-RNMF model. The convergence plot for the IG-MDSR-RNMF model is shown in Figure 7.12. The plot illustrates the variation of the cost function Φ against iteration for all five datasets. Overall, the decreasing nature of the cost over time validates that the model converges. It can also be observed from the plot that the initial cost value for all the datasets starts from a high position and after a few initial epochs, the value of the cost function has almost reached a straight line parallel to the horizontal axis. That is, there are very nominal changes in the cost value. Thus, we can conclude that the model has converged. Figure 7.12 depict the cost versus iteration plot for the GLRC, ONP, ONP, SP and MovieLens datasets with $r = 118$, $r = 24$, $r = 327$, $r = 9$ and $r = 470$ respectively.

7.6 Analysis of Computational Complexity

The neural network architecture of both IG-MDSR-NMF and IG-MDSR-RNMF are the same. The learning procedure of both models is also the same. The only difference is in the output of the models. In IG-MDSR-NMF both factor matrices adhere to the non-negativity criteria but in IG-MDSR-RNMF only one matrix follows the non-negativity criteria and the other does not. Thus the computational complexity of both models is also the same. Based on the discussion illustrated in the previous Chapter 6 Section 6.6, the overall computational complexity of IG-MDSR-RNMF is $\mathcal{O}(tmnr_1)$, where t is the number of epochs, (m, n) is the dimension of the input matrix $\mathbf{X}$ and r_1 is the maximum

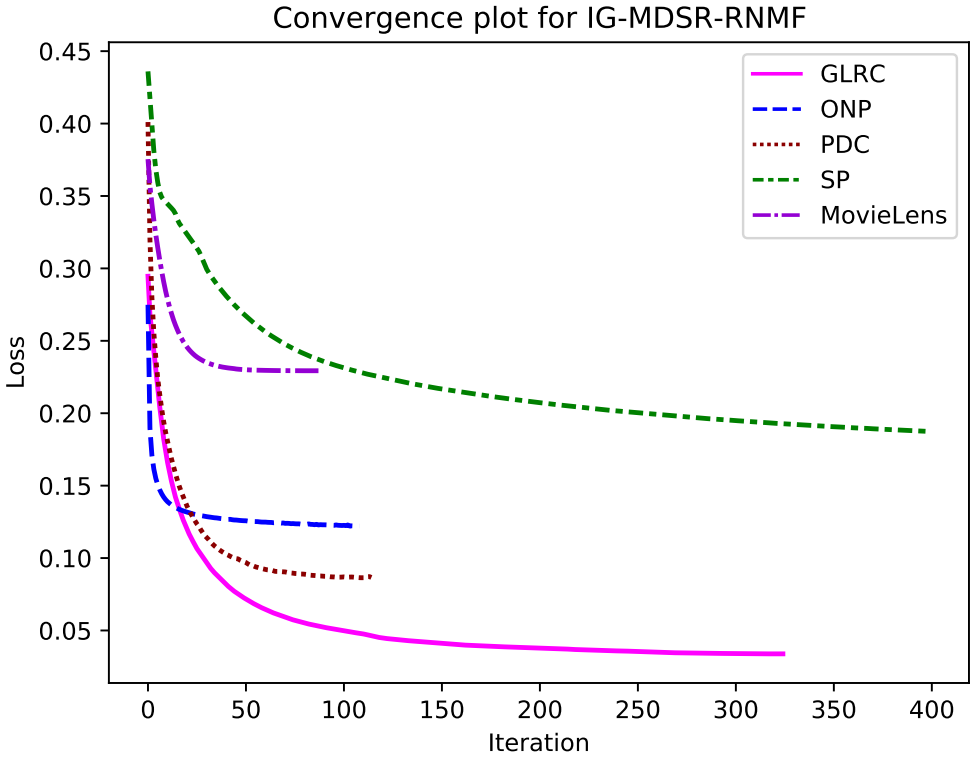

FIGURE 7.12: Loss vs. iteration plots of IG-MDSR-RNMF for GLRC, ONP, PDC, SP and MovieLens dataset.

number of nodes among the hidden layers of IG-MDSR-RNMF. According to the architecture, the sequence of the number of nodes per layer is $n = r_0 > r_1 > r_2 > ... > r_s = r$. Hence, the first hidden layer has the maximum number of nodes among all hidden layers and thus has been used to estimate the overall computational complexity of IG-MDSR-RNMF.

7.7 Conclusions

A large dataset with numerous attributes can be dimensionally reduced using different methods. In this chapter, we have developed a novel neural network model (IG-MDSR-RNMF) by combining the advantages of deep learning for dimension reduction with the benefits of NMF. The design of IG-MDSR-RNMF has been inspired by the way humans learn new concepts by constantly consulting the original text to keep learning on track and increase the efficacy of knowledge acquisition. As the input assists IG-MDSR-RNMF at every stage/level of hierarchical learning, the model has been named

"input guided". The way that IG-MDSR-RNMF has been designed allows it to imitate the factorization behaviour of the traditional NMF method. Through a modified matrix factorization technique known as Relaxed-NMF, where only the basis matrix satisfies the non-negativity requirement, IG-MDSR-RNMF presents an enhanced form of learning.

A thorough experimentation of the quality of dimension reduction using IG-MDSR-RNMF on five well-known datasets has been performed in order to compare the outcome with ten other dimension reduction methods. Three traditional dimension reduction algorithms and seven NMF-based techniques make up these ten dimension reduction methodologies. Maintaining the local shape of the original data in the transformed space is seen as a standard for dimension reduction methods. IG-MDSR-RNMF has demonstrated superiority over other dimension reduction techniques for the preservation of local shape. By contrasting their results using both classification and clustering over the original dataset, the quality of low rank embedding by IG-MDSR-RNMF has been examined and verified, which in turn supports the necessity of dimension reduction. Four clustering techniques, four cluster validity indices, four classification algorithms and four classification performance measurements have all been employed in the course of the experimentation. The results corroborate the advantages of IG-MDSR-RNMF over the other dimension reduction techniques examined in this study.

The statistical significance of the large result set has been provided for better understanding. The results unequivocally demonstrate that IG-MDSR-RNMF outperforms other dimension reduction techniques taken into consideration here in terms of both statistical performance and intrinsic property preservation principles. Additionally, experiments have been conducted to show that IG-MDSR-RNMF converges over time. The amount of elementary operations executed has also been explored in relation to the computational complexity of IG-MDSR-RNMF.

Throughout this thesis, we have developed and demonstrated five different neural network models to find the low dimensional embedding of the given input to overcome the problem of the curse of dimensionality. The next chapter concludes the thesis along with some highlights of different directions for future work.

Chapter 8

Conclusions and Scope of Further Research

The key aspects of each of the contributing chapters of the thesis have been compiled in this chapter. The chapter also sheds light on the potential areas for future research into deep learning based NMF techniques.

8.1 Conclusions

In this thesis, we intend to combine the advantages of the classical iterative NMF approach with those of the neural network design. The concluding remarks of the contributory chapters have been summarized here under architecture and results.

8.1.1 Architecture

The standard NMF approach updates the factors of the input matrix via a block coordinate descent scheme. In Chapter 3, we have designed a neural network model, called Non-negative Matrix Factorization Neural Network (n^2MFn^2) for NMF, where this limitation has been removed by updating both factors concurrently [24, 25].

n^2MFn^2 has a single hidden layer that serves as the system's slender layer. The neural network model is divided into two phases: deconstruction and reconstruction.

Hence, the architecture might be referred to as the Single Deconstruction Single Reconstruction (SDSR) framework. The model's non-negative factors are the slender layer output and the weight matrix that connects the slender and output layers. Because the weight matrix must adhere to the non-negativity requirement, a variation of the popular He weight initialization technique, known as the Modified He initialization technique (mHe), has been developed to ensure model consistency in terms of non-negativity. The ReLU activation function has also been modified.

The objective function has been developed to reduce overfitting by employing L1 regularization/Lasso regularisation. The innovative regularizer guarantees the best feasible approximation of the input data matrix. Furthermore, the regularising parameter has been set so that it has a controlled influence on the regularizer when n^2MFn^2 attempts to regenerate the input. The architecture's learning rules have been derived while remaining within the model's constraints. The update rules adjust each element of the weight matrix independently to meet the requirement. In n^2MFn^2, the rule for determining the value of the adaptive learning rate is fixed, but it has been formulated in a manner that it directs each weight value separately to fulfil the non-negative requirements. Choosing a fixed learning rate *a priori* that meets the non-negativity criterion is challenging; hence, an adaptive learning rate for neural networks has been proposed.

n^2MFn^2 realizes NMF in a shallow neural network architecture. However, deep neural network architecture may be employed for NMF in order to take advantage of hierarchical learning during the dimension reduction of large datasets. In Chapter 4, we have developed a deep learning model, known as Deep Neural Network for Non-negative Matrix Factorization (DN3MF), for the task of NMF, aiming for low rank approximation of the data matrix [27]. There are two stages to the model: pretraining and stacking. The pretraining stage uses a shallow neural network architecture, whereas the stacking stage uses a deep neural network design.

There are several deconstruction layers and an equal number of reconstruction levels. Thus, the model adheres to the Multiple Deconstruction Multiple Reconstruction (MDMR) paradigm. To meet the non-negativity condition, n^2MFn^2 has employed the ReLU activation function. However, due to its tendency to eliminate negative components, the ReLU activation function is prone to data loss. In contrast to ReLU, using

the sigmoid activation function helps prevent data loss. To meet the model's non-negativity criteria, the sigmoid activation function has been carefully changed. The Xavier initialization approach solves the exploding or vanishing gradient problem. Similar to n^2MFn^2, DN3MF has used regularisation in the model's objective function to achieve the best possible approximation of the input matrix. The formulation of a one-of-a-kind adaptive learning mechanism has aided the model's success.

DN3MF has a severe flaw: it fails to generate two distinct factor matrices. In Chapter 5, we have solved the same by designing an innovative deep learning framework. We have developed a deep learning model, called Multiple Deconstruction Single Reconstruction Deep Neural Network Model for Non-negative Matrix Factorization (MDSR-NMF) to approximate the data matrix at a low rank using NMF [26]. The methodology consists of two stages: pretraining and stacking.

The model's pretraining stage is achieved using a shallow neural network architecture. A one-of-a-kind deep learning architecture has been designed for stacking. The design of this stacking stage sets it apart from other contemporary deep learning models. In the MDSR-NMF stacking stage design, the number of layers between the input layer and the slenderest layer of the framework is different from the number of layers between the slenderest layer and the output layer of the framework. There is no limit to the number of layers between the input and the slenderest layer of the MDSR-NMF architecture. Here, the slenderest layer connects straight to the output layer. This novel approach guarantees a unique pair of factor matrices for the reconstructed input matrix. Thus, MDSR-NMF mimics the factorization behaviour of conventional NMF approaches.

The goal of emulating the factorization behaviour of the standard NMF approach while assuring the output of a unique pair of factor matrices of the reconstructed input matrix has inspired us to develop the previous three models. In Chapter 6, we have combined the advantages of traditional iterative learning with those of deep learning in a way that mimics the nature of human learning. While learning, humans usually try to break down concepts into smaller chunks and learn hierarchically, returning back to the source regularly to ensure the accuracy of the learning. Thus, we might conclude that human learning is always input-guided.

In Chapter 6, we have developed a model called Input Guided Multiple Deconstruction Single Reconstruction Neural Network for Non-negative Matrix Factorization (IG-MDSR-NMF) for dimension reduction [28]. The model consists of two phases: deconstruction and reconstruction. The deconstruction phase layers receive the preceding layer's hierarchically processed output as input, as well as a copy of the original data. Thus, the model is named "Input Guided". The reconstruction phase consists of only one layer, ensuring that the NMF model generates a unique pair of non-negative factor matrices.

Although our main objective is to create a low-dimensional non-negative representation of the input matrix, we have been experimenting with previously developed models to produce a pair of non-negative components as output. The second component can only be used in conjunction with the low rank embedding to reproduce the original matrix, which is not at all our goal but to obtain better low rank embedding of the original space. In Chapter 7, we relaxed the second factor matrix from being non-negative and introduced Relaxed Non-negative Matrix Factorization (RNMF), a novel matrix factorization approach, with an aim to even better low rank embedding of the original data [28].

8.1.2 Results

Throughout this thesis, we have developed and exhibited five distinct neural network models for determining the low dimensional embedding of a given input in order to avoid the curse of dimensionality. The performance of the models in dimension reduction has been demonstrated in two parts. First, the models' capacity to preserve the local structure of data has been compared, and the requirement for dimension reduction has been justified over the original data. Second, the efficacy of the dimensionally reduced dataset is investigated for downstream analyses such as classification and clustering. Furthermore, the appropriate p-values have been calculated to determine the statistical significance of the outcomes.

The performance and effectiveness of n^2MFn^2 (in Chapter 3) in preserving the local structure of data have been extensively evaluated using the trustworthiness score across five datasets comparing individually against six other prominent dimension reduction techniques. n^2MFn^2 has demonstrated superior or competitive performance,

highlighting its efficacy in dimension reduction. In terms of validation through classification techniques, n^2MFn^2 has shown robust performance in low rank approximation with algorithms like KNN, NB, MLP and QDA. Besides, for different clustering methods, n^2MFn^2 has established its superiority over the original and other dimension reduction algorithms considered here. Statistical analysis reveals that the low dimensional embeddings produced by n^2MFn^2 are not only effective but also statistically significant across datasets, consistently outperforming the other dimension reduction techniques. Overall, n^2MFn^2 emerges as a superior dimensionality reduction technique, excelling in maintaining data structure integrity and generating meaningful low dimensional representations for diverse analytical tasks.

In Chapter 4, DN3MF has been thoroughly evaluated using the trustworthiness score metric across five datasets, and compared against seven other dimension reduction techniques including n^2MFn^2. It has consistently demonstrated superior performance in preserving the local structure of data, outperforming other methods across most of the datasets. For classification tasks, low rank approximations produced by DN3MF have been validated using various algorithms, such as NB, MLP, QDA and KNN, alongside centroid-based (MBkM, FcM, GMM) and hierarchical (BIRCH) clustering techniques. Here, it has consistently outperformed the original data and achieved the highest ratings compared to other techniques in most of the instances. In clustering evaluations too, DN3MF has demonstrated superior performance in both cases. Statistically, DN3MF's low-dimensional embeddings have consistently proved to be significant across datasets, surpassing both original data and outcomes from other dimension reduction methods. It has consistently achieved top performance ratings compared to the other techniques, underscoring its effectiveness and reliability. Thus, DN3MF has proved its efficacy in dimension reduction, capable of maintaining data integrity, and producing valuable low dimensional representations for complex data analysis. From the experimental analyses, it can be clearly observed that, in contrast to n^2MFn^2, the performance of DN3MF has bettered in most of the circumstances.

MDSR-NMF (Chapter 5) has consistently demonstrated its efficacy in preserving local data structures, achieving top trustworthiness scores for two datasets and above-average scores for the others. Overall, MDSR-NMF has performed slightly below the previous model (DN3MF) but outperformed the other methods. In terms of classification and clustering, statistically, MDSR-NMF has outperformed the original data and

achieved the highest ratings compared to the other methods in most of the cases. Low-rank embeddings generated by MDSR-NMF have been statistically significant across datasets, as indicated by comparative p-values. Thus, MDSR-NMF has justified its efficacy over others.

IG-MDSR-NMF, developed in Chapter 6, stands out as a superior dimension reduction technique based on extensive evaluations across multiple datasets and comparisons with nine other dimension reduction methods including the above models developed in this thesis. It has excelled in preserving local data structures, validated by trustworthiness scores, and has consistently outperformed competitors in both classification and clustering tasks. Statistically significant low-rank embeddings produced by IG-MDSR-NMF have underscored its effectiveness across diverse dataset characteristics, showcasing robust performance and broad applicability. Its innovative input-guided design places it apart, reaffirming its status as a leading choice for dimensionality reduction in complex data analysis scenarios.

IG-MDSR-RNMF (Chapter 7), a dimension reduction technique with a novel Relaxed-NMF technique, has proved its ability to preserve local data structure effectively. Compared to ten other state-of-the-art methods (including the previous four models developed in the thesis) across five datasets, IG-MDSR-RNMF has consistently outperformed in both classification and clustering tasks. It produces statistically significant low-rank embeddings, demonstrating robustness across varying dataset characteristics. Its input-guided principles and Relaxed-NMF technique, positions it apart as a superior choice for dimensionality reduction, validated through extensive experimental evaluations.

The above discussion on the performance of five models developed one after the other starting from n^2MFn^2 up to IG-MDSR-RNMF, clearly shows that with each development the overall performance has either mostly improved than the previous one or has at least registered at par performance in some situations. Thus, in terms of experimental outcomes, the overall agenda of the progressive development of models has been able to justify their efficacy over the previous ones, and definitely has performed better than the other six dimension reduction techniques considered here for comparison. The convergence analysis presented in each chapter has experimentally established the convergence of all five models. The computational complexities have

also been calculated for all five models to present the upper bound ($\mathcal{O}$) of computation cost in terms of the number of elementary operations performed.

8.2 Future scope of research

Some possible enhancements of the current thesis and some scope for further research have been outlined here.

8.2.1 Scope of enhancements

The theoretical connections between NMF and k-means, providing strong support and theoretical foundations for NMF-based clustering, are well established [19, 20, 21, 74]. It has been proved that NMF is equivalent to a relaxed k-means clustering yielding a soft partitioning [29]. Moreover, Orthogonal NMF has been shown to boil down to k-means clustering [19, 74]. On the other hand, in our models, we are not exploiting the natural clustering property of NMF. Our one and only intention is dimension reduction through NMF. In future, the models developed in the thesis can be explored in the direction of NMF-based clustering.

We have performed comparative analyses of the performance of the developed models in terms of their ability to preserve the local structure of data in the low-dimensional space with that of the traditional autoencoder model along with other relevant techniques, using the trustworthiness metric. Several other similar variations of the trustworthiness metric have been developed to date. The trustworthiness framework developed by Lee et al. [67, 68, 69] is one such variation. In future, such comparisons can also be performed with others such metrics.

8.2.2 Scope for further research

The models described in the thesis have been designed to be applied to numerical data matrices. In real life, there are a number of data types like image, sequence and spatial data to name a few. NMF based neural network models developed in this thesis are not

compatible with applications on these diverse kinds of data. In future, we aim to build deep learning based models for NMF to be applied to image data.

In the real world, the majority of datasets naturally comprise several views or representations. It becomes natural to combine these several representations in order to achieve greater performance rather than depending just on one since it is often the case that they offer complimentary and compatible information. In order to perform better than merely concatenating views, the secret to learning from multiple viewpoints (multi-view) is to make use of each view's distinctive understanding. Multi-view clustering is a difficult aspect of unsupervised learning from many views of unlabeled data since unlabeled data are abundant in real-world scenarios and growing amounts of them come in various views from different sources. Organising items into clusters based on several representations of the object is the aim of multi-view clustering [42, 77, 117]. We aim to develop deep neural network based models on NMF to overcome these problems.

Web-scale dyadic data and other large datasets have been subjected to the application of NMF [76, 123]. In such instances, using a cluster of devices to accelerate the factorization process is preferable. Nevertheless, implementing NMF effectively in a distributed setting is challenging. Through the use of novel update functions, local aggregation and comprehensive parallelism exploration can be made possible. We might have to split the original matrix into blocks and partition the factor matrices into equivalent blocks in order to accommodate the new form. When updating a factor matrix, the update methods should enable modifying different blocks both separately and concurrently. Additionally, it should be able to make a distributed implementation easier such that simultaneous updates of multiple factor matrix blocks are possible. The current research work can be further enhanced to encompass these kinds of problems.

An intelligent system's capability is generally measured in terms of its generalisation ability; the higher the generalisation ability, the more accurate is the prediction of the system for new events. Intelligent systems are an approximation of the true model of a problem, which is known as a hypothesis. The risk is described as a mismatch between the right answer and the hypothesis, or an accumulation of mistakes. However, if the actual model is unknown, the risk cannot be determined. Meanwhile, empirical

risk is used to approximate the true answer. However, the disparity between the projected outcomes from hypothesis overfitting on the sample data and the true solution of the samples might lead to mistakes. Nowadays, intelligent systems have increasingly used empirical risk reduction to determine the optimal solution. However, it has been observed that this technique might cause overfitting of the classification function, resulting in poor generalisation performance. The reason for this is that the empirical risk, which is used to estimate the genuine risk, is not a trustworthy predictor of the algorithm's performance due to the small number of training samples compared to the real-world data that must be classified. To address this issue, statisticians have introduced the idea of generalisation error bounds, which give a measure of the bias and convergence rate between the actual and predicted risks of a learning algorithm. As a result, the primary goal is to increase algorithm efficiency and performance, particularly for real-world issues [118]. There is no way to compute the confidence risk correctly; so, only an estimated interval may be provided, which also allows for the calculation of just an upper bound on the overall error with the accurate value [103]. These notions of generalisation error bounds can be studied on the neural network based NMF models.

In real life, a very small percentage of data are labelled, whereas, most of the data are unlabelled. To tackle this, semi-supervised NMF can be formulated to work on both labelled and unlabeled data. This kind of NMF technique aims to leverage minimal labelled data and plentiful unlabeled data to generate effective part-based representations and more accurate low-dimensional representations. In future, research can focus on incorporating various types of supervised information into the NMF framework, designing deep semi-supervised NMF architecture. It may also aid in feature selection along with feature extraction [89].

Appendix A

Tables of p-values for n^2MFn^2 vs. others depicting classification performances

In the thesis for each of the five developed models, one summary table of p-values with respect to the classification performances and another summary table of p-values with respect to the clustering performances have been provided in the Chapters 3-7. For illustration, detailed p-values of one such Table 3.2 have been provided. In order to restrict the size of the thesis, we have not included similar tables for clustering in Chapter 3, and for the other models developed in Chapters 4-7.

The following Tables A.1-A.20 depict the actual p-values obtained for n^2MFn^2 over six other dimension reduction techniques for each of the five datasets, each of the four classifiers and each of the four classification performance evaluators. From these tables, the summary Table 3.2 has been obtained.

TABLE A.1: *p*-values for the classification performances of n²MFn² and other dimension reduction algorithms on the GLRC dataset in terms of accuracy.

Techniques	KNN	MLP	NB	QDA
n²MFn² vs. AE	0.0084	0.0262	0.0293	0.0251
n²MFn² vs. PCA	0.0034	0.0001	0.0016	0.0045
n²MFn² vs. UMAP	0.0001	0.0000	0.0001	0.0000
n²MFn² vs. NMF	0.0046	0.0013	0.0084	0.0015
n²MFn² vs. DS-NMF	0.0067	0.0000	0.0001	0.0000
n²MFn² vs. Semi-NMF	0.0002	0.0000	0.0000	0.0000

The count of *p*-values in Table A.1 less than the assumed threshold value of 0.05, is 24 out of 24.

TABLE A.2: *p*-values for the classification performances of n²MFn² and other dimension reduction algorithms on the GLRC dataset in terms of F1 score.

Techniques	KNN	MLP	NB	QDA
n²MFn² vs. AE	0.0045	0.1668	0.0421	0.0041
n²MFn² vs. PCA	0.0000	0.0000	0.0000	0.0002
n²MFn² vs. UMAP	0.0000	0.0000	0.0001	0.0000
n²MFn² vs. NMF	0.0180	0.0000	0.0019	0.0002
n²MFn² vs. DS-NMF	0.0000	0.0000	0.0000	0.0000
n²MFn² vs. Semi-NMF	0.0000	0.0000	0.0000	0.0003

The count of *p*-values in Table A.2 less than the assumed threshold value of 0.05, is 23 out of 24.

TABLE A.3: *p*-values for the classification performances of n²MFn² and other dimension reduction algorithms on the GLRC dataset in terms of Cohen-Kappa score.

Techniques	KNN	MLP	NB	QDA
n²MFn² vs. AE	0.0007	0.1390	0.0220	0.0076
n²MFn² vs. PCA	0.0000	0.0000	0.0000	0.0002
n²MFn² vs. UMAP	0.0000	0.0000	0.0001	0.0000
n²MFn² vs. NMF	0.0064	0.0000	0.0003	0.0002
n²MFn² vs. DS-NMF	0.0000	0.0000	0.0000	0.0000
n²MFn² vs. Semi-NMF	0.0000	0.0000	0.0000	0.0001

The count of *p*-values in Table A.3 less than the assumed threshold value of 0.05, is 23 out of 24.

TABLE A.4: *p*-values for the classification performances of n^2MFn^2 and other dimension reduction algorithms on the GLRC dataset in terms of Matthew's Correlation Coefficient score.

Techniques	KNN	MLP	NB	QDA
n^2MFn^2 vs. AE	0.0088	0.0972	0.1070	0.0071
n^2MFn^2 vs. PCA	0.0000	0.0000	0.0001	0.0001
n^2MFn^2 vs. UMAP	0.0016	0.0000	0.0003	0.0000
n^2MFn^2 vs. NMF	0.0020	0.0000	0.0015	0.0003
n^2MFn^2 vs. DS-NMF	0.0000	0.0000	0.0000	0.0000
n^2MFn^2 vs. Semi-NMF	0.0000	0.0000	0.0000	0.0001

The count of *p*-values in Table A.4 less than the assumed threshold value of 0.05, is 22 out of 24.

TABLE A.5: *p*-values for the classification performances of n^2MFn^2 and other dimension reduction algorithms on the ONP dataset in terms of accuracy.

Techniques	KNN	MLP	NB	QDA
n^2MFn^2 vs. AE	0.0006	0.0000	0.1501	0.0000
n^2MFn^2 vs. PCA	0.0000	0.0000	0.0992	0.0000
n^2MFn^2 vs. UMAP	0.0000	0.0630	0.5776	0.0000
n^2MFn^2 vs. NMF	0.0000	0.0001	0.0229	0.0000
n^2MFn^2 vs. DS-NMF	0.0000	0.0002	0.7660	0.0000
n^2MFn^2 vs. Semi-NMF	0.0001	0.0001	0.3470	0.0002

The count of *p*-values in Table A.5 less than the assumed threshold value of 0.05, is 18 out of 24.

TABLE A.6: *p*-values for the classification performances of n^2MFn^2 and other dimension reduction algorithms on the ONP dataset in terms of F1 score.

Techniques	KNN	MLP	NB	QDA
n^2MFn^2 vs. AE	0.0076	0.0001	0.0154	0.0005
n^2MFn^2 vs. PCA	0.0011	0.1091	0.9377	0.0000
n^2MFn^2 vs. UMAP	0.0094	0.0363	0.0001	0.0000
n^2MFn^2 vs. NMF	0.0081	0.0000	0.8657	0.0000
n^2MFn^2 vs. DS-NMF	0.0037	0.0000	0.0709	0.0000
n^2MFn^2 vs. Semi-NMF	0.0161	0.0000	0.0239	0.0002

The count of *p*-values in Table A.6 less than the assumed threshold value of 0.05, is 20 out of 24.

TABLE A.7: *p*-values for the classification performances of n^2MFn^2 and other dimension reduction algorithms on the ONP dataset in terms of Cohen-Kappa score.

Techniques	KNN	MLP	NB	QDA
n^2MFn^2 vs. AE	0.0010	0.0000	0.1172	0.0000
n^2MFn^2 vs. PCA	0.0000	0.0000	0.0887	0.0000
n^2MFn^2 vs. UMAP	0.0000	0.0302	0.9324	0.0000
n^2MFn^2 vs. NMF	0.0000	0.0000	0.0351	0.0000
n^2MFn^2 vs. DS-NMF	0.0000	0.0002	0.5428	0.0000
n^2MFn^2 vs. Semi-NMF	0.0001	0.0001	0.2203	0.0002

The count of *p*-values in Table A.7 less than the assumed threshold value of 0.05, is 19 out of 24.

TABLE A.8: *p*-values for the classification performances of n^2MFn^2 and other dimension reduction algorithms on the ONP dataset in terms of Matthew's Correlation Coefficient score.

Techniques	KNN	MLP	NB	QDA
n^2MFn^2 vs. AE	0.0010	0.0001	0.0825	0.0000
n^2MFn^2 vs. PCA	0.0000	0.0000	0.0489	0.0000
n^2MFn^2 vs. UMAP	0.0000	0.0303	0.9444	0.0000
n^2MFn^2 vs. NMF	0.0000	0.0001	0.0289	0.0000
n^2MFn^2 vs. DS-NMF	0.0000	0.0004	0.4803	0.0000
n^2MFn^2 vs. Semi-NMF	0.0001	0.0002	0.2754	0.0002

The count of *p*-values in Table A.8 less than the assumed threshold value of 0.05, is 20 out of 24.

TABLE A.9: *p*-values for the classification performances of n^2MFn^2 and other dimension reduction algorithms on the PDC dataset in terms of accuracy.

Techniques	KNN	MLP	NB	QDA
n^2MFn^2 vs. AE	0.0248	0.0136	0.7492	0.0004
n^2MFn^2 vs. PCA	0.0000	0.0006	0.0010	0.0018
n^2MFn^2 vs. UMAP	0.0000	0.0000	0.0757	0.0000
n^2MFn^2 vs. NMF	0.0027	0.0016	0.0001	0.0280
n^2MFn^2 vs. DS-NMF	0.0001	0.0001	0.0000	0.0000
n^2MFn^2 vs. Semi-NMF	0.0000	0.0002	0.0000	0.0000

The count of *p*-values in Table A.9 less than the assumed threshold value of 0.05, is 22 out of 24.

TABLE A.10: *p*-values for the classification performances of n^2MFn^2 and other dimension reduction algorithms on the PDC dataset in terms of F1 score.

Techniques	KNN	MLP	NB	QDA
n^2MFn^2 vs. AE	0.0268	0.0072	0.9238	0.0000
n^2MFn^2 vs. PCA	0.0001	0.0011	0.0003	0.0007
n^2MFn^2 vs. UMAP	0.0000	0.0001	0.1006	0.0000
n^2MFn^2 vs. NMF	0.0102	0.0013	0.0012	0.0216
n^2MFn^2 vs. DS-NMF	0.0001	0.0001	0.0000	0.0000
n^2MFn^2 vs. Semi-NMF	0.0000	0.0002	0.0000	0.0000

The count of *p*-values in Table A.10 less than the assumed threshold value of 0.05, is 22 out of 24.

TABLE A.11: *p*-values for the classification performances of n^2MFn^2 and other dimension reduction algorithms on the PDC dataset in terms of Cohen-Kappa score.

Techniques	KNN	MLP	NB	QDA
n^2MFn^2 vs. AE	0.0333	0.0013	0.1906	0.0001
n^2MFn^2 vs. PCA	0.0000	0.0000	0.0007	0.0001
n^2MFn^2 vs. UMAP	0.0001	0.0001	0.0343	0.0000
n^2MFn^2 vs. NMF	0.0065	0.0053	0.0007	0.0174
n^2MFn^2 vs. DS-NMF	0.0006	0.0000	0.0000	0.0000
n^2MFn^2 vs. Semi-NMF	0.0000	0.0000	0.0000	0.0000

The count of *p*-values in Table A.11 less than the assumed threshold value of 0.05, is 23 out of 24.

TABLE A.12: *p*-values for the classification performances of n^2MFn^2 and other dimension reduction algorithms on the PDC dataset in terms of Matthew's Correlation Coefficient score.

Techniques	KNN	MLP	NB	QDA
n^2MFn^2 vs. AE	0.0493	0.0032	0.1308	0.0001
n^2MFn^2 vs. PCA	0.0001	0.0000	0.0028	0.0002
n^2MFn^2 vs. UMAP	0.0000	0.0003	0.0326	0.0000
n^2MFn^2 vs. NMF	0.0033	0.0090	0.0038	0.0195
n^2MFn^2 vs. DS-NMF	0.0001	0.0000	0.0000	0.0000
n^2MFn^2 vs. Semi-NMF	0.0003	0.0000	0.0000	0.0000

The count of *p*-values in Table A.12 less than the assumed threshold value of 0.05, is 23 out of 24.

TABLE A.13: *p*-values for the classification performances of n²MFn² and other dimension reduction algorithms on the SP dataset in terms of accuracy.

Techniques	KNN	MLP	NB	QDA
n²MFn² vs. AE	0.1657	0.9163	0.6236	0.0526
n²MFn² vs. PCA	0.0000	0.0000	0.0002	0.0000
n²MFn² vs. UMAP	0.2199	0.2025	0.9246	0.0002
n²MFn² vs. NMF	0.2790	0.4443	0.4601	0.0062
n²MFn² vs. DS-NMF	0.0338	0.0696	1.0000	0.0027
n²MFn² vs. Semi-NMF	0.0265	0.0631	1.0000	0.0054

The count of *p*-values in Table A.13 less than the assumed threshold value of 0.05, is 10 out of 24.

TABLE A.14: *p*-values for the classification performances of n²MFn² and other dimension reduction algorithms on the SP dataset in terms of F1 score.

Techniques	KNN	MLP	NB	QDA
n²MFn² vs. AE	0.1618	0.5713	0.6060	0.0334
n²MFn² vs. PCA	0.0000	0.0003	0.0018	0.0000
n²MFn² vs. UMAP	0.2605	0.2820	0.4985	0.0002
n²MFn² vs. NMF	0.4410	0.6412	0.7656	0.0048
n²MFn² vs. DS-NMF	0.0321	0.5772	0.2885	0.0059
n²MFn² vs. Semi-NMF	0.0273	0.5167	0.1622	0.0063

The count of *p*-values in Table A.14 less than the assumed threshold value of 0.05, is 11 out of 24.

TABLE A.15: *p*-values for the classification performances of n²MFn² and other dimension reduction algorithms on the SP dataset in terms of Cohen-Kappa score.

Techniques	KNN	MLP	NB	QDA
n²MFn² vs. AE	0.4244	0.3395	0.5520	0.1531
n²MFn² vs. PCA	0.0011	0.0000	0.0001	0.0000
n²MFn² vs. UMAP	0.5087	0.2578	0.4393	0.0006
n²MFn² vs. NMF	0.2559	0.0237	0.1978	0.0127
n²MFn² vs. DS-NMF	0.0851	0.0019	0.0434	0.0018
n²MFn² vs. Semi-NMF	0.1207	0.0036	0.0014	0.0071

The count of *p*-values in Table A.15 less than the assumed threshold value of 0.05, is 13 out of 24.

TABLE A.16: *p*-values for the classification performances of n^2MFn^2 and other dimension reduction algorithms on the SP dataset in terms of Matthew's Correlation Coefficient score.

Techniques	KNN	MLP	NB	QDA
n^2MFn^2 vs. AE	0.3922	0.3833	0.4120	0.1512
n^2MFn^2 vs. PCA	0.0010	0.0000	0.0001	0.0000
n^2MFn^2 vs. UMAP	0.4990	0.2498	0.5536	0.0006
n^2MFn^2 vs. NMF	0.2360	0.0859	0.2202	0.0124
n^2MFn^2 vs. DS-NMF	0.0808	0.0145	0.4007	0.0017
n^2MFn^2 vs. Semi-NMF	0.1056	0.0256	0.0033	0.0067

The count of *p*-values in Table A.16 less than the assumed threshold value of 0.05, is 11 out of 24.

TABLE A.17: *p*-values for the classification performances of n^2MFn^2 and other dimension reduction algorithms on the MovieLens dataset in terms of accuracy.

Techniques	KNN	MLP	NB	QDA
n^2MFn^2 vs. AE	0.0362	0.0062	0.0001	0.0003
n^2MFn^2 vs. PCA	0.0033	0.0002	0.0003	0.0153
n^2MFn^2 vs. UMAP	0.0000	0.0051	0.0025	0.0021
n^2MFn^2 vs. NMF	0.3590	0.0017	0.0043	0.0008
n^2MFn^2 vs. DS-NMF	0.0181	0.0113	0.0020	0.0001
n^2MFn^2 vs. Semi-NMF	0.0006	0.0004	0.0025	0.0002

The count of *p*-values in Table A.17 less than the assumed threshold value of 0.05, is 23 out of 24.

TABLE A.18: *p*-values for the classification performances of n^2MFn^2 and other dimension reduction algorithms on the MovieLens dataset in terms of F1 score.

Techniques	KNN	MLP	NB	QDA
n^2MFn^2 vs. AE	0.0031	0.0067	0.0004	0.0000
n^2MFn^2 vs. PCA	0.0044	0.0002	0.0000	0.0569
n^2MFn^2 vs. UMAP	0.0000	0.0053	0.0023	0.0016
n^2MFn^2 vs. NMF	0.1140	0.0017	0.0039	0.0004
n^2MFn^2 vs. DS-NMF	0.0304	0.0125	0.0053	0.0000
n^2MFn^2 vs. Semi-NMF	0.0003	0.0004	0.0118	0.0002

The count of *p*-values in Table A.18 less than the assumed threshold value of 0.05, is 22 out of 24.

TABLE A.19: *p*-values for the classification performances of n^2MFn^2 and other dimension reduction algorithms on the MovieLens dataset in terms of Cohen-Kappa score.

Techniques	KNN	MLP	NB	QDA
n^2MFn^2 vs. AE	0.0019	1.0000	0.2635	0.0000
n^2MFn^2 vs. PCA	0.3937	0.0782	0.8833	0.1393
n^2MFn^2 vs. UMAP	0.0033	1.0000	0.0324	0.1191
n^2MFn^2 vs. NMF	0.1687	1.0000	0.0032	0.0018
n^2MFn^2 vs. DS-NMF	0.0869	1.0000	0.0000	0.0001
n^2MFn^2 vs. Semi-NMF	0.0027	1.0000	0.0001	0.0225

The count of *p*-values in Table A.19 less than the assumed threshold value of 0.05, is 11 out of 24.

TABLE A.20: *p*-values for the classification performances of n^2MFn^2 and other dimension reduction algorithms on the MovieLens dataset in terms of Matthew's Correlation Coefficient score.

Techniques	KNN	MLP	NB	QDA
n^2MFn^2 vs. AE	0.1708	1.0000	0.0121	0.0000
n^2MFn^2 vs. PCA	0.8240	0.0601	0.0930	0.1242
n^2MFn^2 vs. UMAP	0.0884	1.0000	0.0019	0.1109
n^2MFn^2 vs. NMF	0.3209	1.0000	0.0295	0.0016
n^2MFn^2 vs. DS-NMF	0.3673	1.0000	0.0000	0.0001
n^2MFn^2 vs. Semi-NMF	0.0593	1.0000	0.0000	0.0199

The count of *p*-values in Table A.20 less than the assumed threshold value of 0.05, is 9 out of 24.

Bibliography

[1] Matej Artac, Matjaz Jogan, and Ales Leonardis. Incremental PCA for On-line Visual Learning and Recognition. In *Object Recognition Supported by User Interaction for Service Robots*, volume 3, pages 781–784. IEEE, 2002.

[2] Ron Artstein and Massimo Poesio. Inter-coder agreement for computational linguistics. *Computational linguistics*, 34(4):555–596, 2008.

[3] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

[4] Arie Ben-David. About the relationship between ROC curves and Cohen's kappa. *Engineering Applications of Artificial Intelligence*, 21(6):874–882, 2008.

[5] Christopher M Bishop, Markus Svensén, and Christopher KI Williams. GTM: The generative topographic mapping. *Neural Computation*, 10(1):215–234, 1998.

[6] Ingwer Borg and Patrick JF Groenen. *Modern Multidimensional Scaling: Theory and Applications*. Springer Science & Business Media, 2005.

[7] Hervé Bourlard and Yves Kamp. Auto-association by multilayer perceptrons and singular value decomposition. *Biological Cybernetics, Springer*, 59(4-5):291–294, 1988.

[8] Leo Breiman. Bagging predictors. *Machine learning*, 24:123–140, 1996.

[9] Ioan Buciu, Nikos Nikolaidis, and Ioannis Pitas. Nonnegative matrix factorization in polynomial feature space. *IEEE Transactions on Neural Networks*, 19(6):1090–1100, 2008.

[10] Deng Cai, Xiaofei He, Jiawei Han, and Thomas S Huang. Graph regularized nonnegative matrix factorization for data representation. *IEEE transactions on pattern analysis and machine intelligence*, 33(8):1548–1560, 2010.

[11] Deng Cai, Xiaofei He, Xiaoyun Wu, and Jiawei Han. Non-negative matrix factorization on manifold. In *2008 eighth IEEE international conference on data mining*, pages 63–72. IEEE, 2008.

[12] José E Chacón and Ana I Rastrojo. Minimum adjusted Rand index for two clusterings of a given size. *Advances in Data Analysis and Classification*, 17(1):125–133, 2023.

[13] Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46, 1960.

[14] Pierre Comon. Independent component analysis, A new concept? *Signal Processing*, 36(3):287–314, 1994.

[15] Paulo Cortez and Alice Maria Gonçalves Silva. Using data mining to predict secondary school student performance. In *the Proceedings of 5th Annual Future Business Technology Conference, Porto, Portugal*, pages 5–12. The European Multidisciplinary Society for Modelling and Simulation Technology, The European Technology Institute Bvba (EUROSIS-ETI), 2008.

[16] Michael AA Cox and Trevor F Cox. Multidimensional scaling. In *Handbook of data visualization*, pages 315–347. Springer, 2008.

[17] Tukur Dahiru. P-value, a true test of statistical significance? A cautionary note. *Annals of Ibadan postgraduate medicine*, 6(1):21–26, 2008.

[18] Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

[19] Chris Ding, Xiaofeng He, and Horst D Simon. On the equivalence of nonnegative matrix factorization and spectral clustering. In *Proceedings of the 2005 SIAM international conference on data mining*, pages 606–610. SIAM, 2005.

[20] Chris Ding, Tao Li, Wei Peng, and Haesun Park. Orthogonal nonnegative matrix t-factorizations for clustering. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 126–135, 2006.

[21] Chris HQ Ding, Tao Li, and Michael I Jordan. Convex and semi-nonnegative matrix factorizations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(1):45–55, 2008.

[22] David L Donoho and Carrie Grimes. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. In *Proceedings of the National Academy of Sciences*, volume 100, pages 5591–5596. National Acad Sciences, 2003.

[23] Dheeru Dua and Casey Graff. UCI machine learning repository, 2019.

[24] Prasun Dutta and Rajat K De. A Neural Network Model for Matrix Factorization: Dimensionality Reduction. In *2022 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)*, pages 1–6. IEEE, 2022.

[25] Prasun Dutta and Rajat K De. n^2MFn^2: Non-negative Matrix Factorization in A Single Deconstruction Single Reconstruction Neural Network Framework for Dimensionality Reduction. In *2022 International Conference on High Performance Big Data and Intelligent Systems (HDIS)*, pages 79–84. IEEE, 2022.

[26] Prasun Dutta and Rajat K. De. MDSR-NMF: Multiple deconstruction single reconstruction deep neural network model for non-negative matrix factorization. *Network: Computation in Neural Systems*, 34(4):306–342, 2023. PMID: 37818635.

[27] Prasun Dutta and Rajat K De. DN3MF: Deep Neural Network for Non-negative Matrix Factorization towards Low Rank Approximation. *Pattern Analysis and Applications*, 27(4):112, 2024.

[28] Prasun Dutta and Rajat K De. Input Guided Multiple Deconstruction Single Reconstruction neural network models for Matrix Factorization. *arXiv preprint arXiv:2405.13449*, 2024.

[29] Mickael Febrissy, Aghiles Salah, Melissa Ailem, and Mohamed Nadif. Improving NMF clustering by leveraging contextual relationships among words. *Neurocomputing*, 495:105–117, 2022.

[30] Kelwin Fernandes, Pedro Vinagre, and Paulo Cortez. A proactive intelligent decision support system for predicting the popularity of online news. In *Proceedings of the Portuguese Conference on Artificial Intelligence*, pages 535–546. Springer, 2015.

[31] Ronald A Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2):179–188, 1936.

[32] Mikel Galar, Alberto Fernández, Edurne Barrenechea, Humberto Bustince, and Francisco Herrera. An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes. *Pattern Recognition*, 44(8):1761–1776, 2011.

[33] Mark Girolami and Colin Fyfe. Stochastic ICA contrast maximisation using Oja's nonlinear PCA algorithm. *International Journal of Neural Systems*, 8(05n06):661–678, 1997.

[34] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.

[35] GH Golub and C Reinsch. Singular value decomposition and least squares solutions. *Numerische Mathematik*, 14:403–420, 1970.

[36] Zhenxing Guo and Shihua Zhang. Sparse deep nonnegative matrix factorization. *Big Data Mining and Analytics*, 3(1):13–28, 2019.

[37] Harry H Harman. *Modern Factor Analysis*. University of Chicago Press, Chicago, Illinois, United States, 1976.

[38] Hotelling Harold. Relations between two sets of variates. *Biometrika*, 28(3/4):321–377, 1936.

[39] F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *Acm Transactions on Interactive Intelligent Systems (TIIS)*, 5(4):1–19, 2015.

[40] Tamir Hazan, Simon Polak, and Amnon Shashua. Sparse image coding using a 3D non-negative tensor factorization. In *Tenth IEEE International Conference on Computer Vision (ICCV'05)*, volume 1, pages 50–57. IEEE, 2005.

[41] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1026–1034, 2015.

[42] Xiangnan He, Min-Yen Kan, Peichu Xie, and Xiao Chen. Comment-based multi-view clustering of web 2.0 items. In *Proceedings of the 23rd international conference on World wide web*, pages 771–782, 2014.

[43] Xiaofei He, Deng Cai, Shuicheng Yan, and Hong-Jiang Zhang. Neighborhood preserving embedding. In *Proceeding of the Tenth IEEE International Conference on Computer Vision*, volume 2, pages 1208–1213. IEEE, 2005.

[44] Xiaofei He and Partha Niyogi. Locality preserving projections. In *Proceeding of the Advances in Neural Information Processing Systems*, volume 16, pages 153–160, 2004.

[45] Geoffrey E Hinton and Richard S Zemel. Autoencoders, minimum description length and Helmholtz free energy. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 3–10, 1994.

[46] Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(6):417, 1933.

[47] Patrik O Hoyer. Non-negative sparse coding. In *12th IEEE workshop on neural networks for signal processing*, pages 557–565. IEEE, 2002.

[48] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of classification*, 2:193–218, 1985.

[49] Aapo Hyvarinen. Fast and robust fixed-point algorithms for independent component analysis. *IEEE Transactions on Neural Networks*, 10(3):626–634, 1999.

[50] Aapo Hyvärinen and Erkki Oja. Independent component analysis: algorithms and applications. *Neural Networks*, 13(4-5):411–430, 2000.

[51] Yuan Wang Yunde Jia and Changbo Hu Matthew Turk. Fisher non-negative matrix factorization for learning local features. In *Asian conference on computer vision*, pages 27–30. Citeseer, 2004.

[52] Binyan Jiang, Xiangyu Wang, and Chenlei Leng. A direct approach for sparse quadratic discriminant analysis. *The Journal of Machine Learning Research*, 19(1):1098–1134, 2018.

[53] William B Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. *Contemporary Mathematics*, 26(189-206):1, 1984.

[54] Giuseppe Jurman, Samantha Riccadonna, and Cesare Furlanello. A comparison of MCC and CEN error measures in multi-class prediction. *PLoS One*, 7(8), 2012.

[55] Philip M Kim and Bruce Tidor. Subsystem Identification Through Dimensionality Reduction of Large-Scale Gene Expression Data. *Genome Research*, 13(7):1706–1718, 2003.

[56] Yong-Deok Kim and Seungjin Choi. Weighted nonnegative matrix factorization. In *2009 IEEE international conference on acoustics, speech and signal processing*, pages 1541–1544. IEEE, 2009.

[57] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[58] Teuvo Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, 1990.

[59] Teuvo Kohonen. Things you haven't heard about the Self-Organizing Map. In *Proceedings of the IEEE International Conference on Neural Networks*, pages 1147–1156. IEEE, 1993.

[60] Irene Kotsia, Stefanos Zafeiriou, and Ioannis Pitas. A novel discriminant non-negative matrix factorization algorithm with applications to facial image characterization problems. *IEEE Transactions on Information Forensics and Security*, 2(3):588–595, 2007.

[61] Joseph B Kruskal and Myron Wish. Multidimensional Scaling. *Sage University Paper Series on Quantitative Applications in the Social Sciences, Sage Publications*, pages 07–011, 1978.

[62] Thibault Laugel, Marie-Jeanne Lesot, Christophe Marsala, Xavier Renard, and Marcin Detyniecki. Inverse classification for comparison-based interpretability in machine learning. *arXiv preprint arXiv:1712.08443*, 2017.

[63] Thibault Laugel, Marie-Jeanne Lesot, Christophe Marsala, Xavier Renard, and Marcin Detyniecki. Comparison-based inverse classification for interpretability in machine learning. In *Proceedings of the International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pages 100–111. Springer, 2018.

[64] Yann LeCun. Modeles connexionnistes de l'apprentissage (connectionist learning models). *Ph.D. thesis, Universite de Paris VI*, 1987.

[65] Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

[66] Daniel D Lee and H Sebastian Seung. Algorithms for Non-negative Matrix Factorization. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 556–562, 2001.

[67] John Lee and Michel Verleysen. Quality assessment of nonlinear dimensionality reduction based on *K*-ary neighborhoods. *New Challenges for Feature Selection in Data Mining and Knowledge Discovery*, pages 21–35, 2008.

[68] John A Lee and Michel Verleysen. Quality assessment of dimensionality reduction: Rank-based criteria. *Neurocomputing*, 72(7-9):1431–1443, 2009.

[69] John A Lee and Michel Verleysen. Scale-independent quality criteria for dimensionality reduction. *Pattern Recognition Letters*, 31(14):2248–2257, 2010.

[70] Seokjin Lee and Hee-Suk Pang. Feature extraction based on the non-negative matrix factorization of convolutional neural networks for monitoring domestic activity with acoustic signals. *IEEE Access*, 8:122384–122395, 2020.

[71] L Li and YJ Zhang. Bilinear Form-Based Non-Negative Matrix Set Factorization. *Chinese J. Computers*, 32(8):1536–1549, 2009.

[72] Le Li and Yu-Jin Zhang. Non-negative matrix-set factorization. In *Fourth International Conference on Image and Graphics (ICIG 2007)*, pages 564–569. IEEE, 2007.

[73] Stan Z Li, Xin Wen Hou, Hong Jiang Zhang, and Qian Sheng Cheng. Learning spatially localized, parts-based representation. In *2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, volume 1, pages I–I. IEEE, 2001.

[74] Tao Li and Chris Ding. The relationships among various nonnegative matrix factorization methods for clustering. In *Sixth International Conference on Data Mining (ICDM'06)*, pages 362–371. IEEE, 2006.

[75] Zhizheng Liang, Youfu Li, and Tuo Zhao. Projected gradient method for kernel discriminant nonnegative matrix factorization and the applications. *Signal Processing*, 90(7):2150–2163, 2010.

[76] Chao Liu, Hung-chih Yang, Jinliang Fan, Li-Wei He, and Yi-Min Wang. Distributed nonnegative matrix factorization for web-scale dyadic data analysis on mapreduce. In *Proceedings of the 19th international conference on World wide web*, pages 681–690, 2010.

[77] Jialu Liu, Chi Wang, Jing Gao, and Jiawei Han. Multi-view clustering via joint nonnegative matrix factorization. In *Proceedings of the 2013 SIAM international conference on data mining*, pages 252–260. SIAM, 2013.

[78] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[79] Yun Mao and Lawrence K Saul. Modeling distances in large-scale networks by matrix factorization. In *Proceedings of the 4th ACM SIGCOMM conference on Internet Measurement*, pages 278–287, 2004.

[80] Marina Meilă. Comparing clusterings—an information based distance. *Journal of multivariate analysis*, 98(5):873–895, 2007.

[81] Pablo Mesejo, Daniel Pizarro, Armand Abergel, Olivier Rouquette, Sylvain Beorchia, Laurent Poincloux, and Adrien Bartoli. Computer-aided classification of gastrointestinal lesions in regular colonoscopy. *IEEE Transactions on Medical Imaging*, 35(9):2051–2063, 2016.

[82] Nasser Mohammadiha and Arne Leijon. Nonnegative matrix factorization using projected gradient algorithms with sparseness constraints. In *2009 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, pages 418–423. IEEE, 2009.

[83] Bruno A Olshausen and David J Field. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision Research, Elsevier*, 37(23):3311–3325, 1997.

[84] Karl Pearson. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

[85] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in Python. *The Journal of Machine Learning Research*, 12:2825–2830, 2011.

[86] C Radhakrishna Rao. The utilization of multiple measurements in problems of biological classification. *Journal of the Royal Statistical Society. Series B (Methodological)*, 10(2):159–203, 1948.

[87] Russell Reed and Robert J MarksII. Neural smithing: supervised learning in feedforward artificial neural networks. *Mit Press*, 1999.

[88] Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[89] Farid Saberi-Movahed, Kamal Berahman, Razieh Sheikhpour, Yuefeng Li, and Shirui Pan. Nonnegative Matrix Factorization in Dimensionality Reduction: A Survey. *arXiv preprint arXiv:2405.03615*, 2024.

[90] C Okan Sakar, Gorkem Serbes, Aysegul Gunduz, Hunkar C Tunc, Hatice Nizam, Betul Erdogdu Sakar, Melih Tutuncu, Tarkan Aydin, M Erdem Isenkul, and Hulya Apaydin. A comparative analysis of speech signal processing algorithms for Parkinson's disease classification and the use of the tunable Q-factor wavelet transform. *Applied Soft Computing, Elsevier*, 74:255–263, 2019.

[91] Susan S Schiffman, M Lance Reynolds, and Forrest W Young. *Introduction to Multidimensional Scaling*. Academic Press, New York, 1981.

[92] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Kernel principal component analysis. In *Proceedings of the International Conference on Artificial Neural Networks*, pages 583–588. Springer, 1997.

[93] Amnon Shashua and Tamir Hazan. Non-negative tensor factorization with applications to statistics and computer vision. In *Proceedings of the 22nd international conference on Machine learning*, pages 792–799, 2005.

[94] Zhen-qiu Shu, Xiao-jun Wu, Cong Hu, Cong-zhe You, and Hong-hui Fan. Deep semi-nonnegative matrix factorization with elastic preserving for data representation. *Multimedia Tools and Applications*, 80(2):1707–1724, 2021.

[95] Zhenqiu Shu, Qinghan Long, Luping Zhang, Zhengtao Yu, and Xiao-Jun Wu. Robust Graph Regularized NMF with Dissimilarity and Similarity Constraints for ScRNA-seq Data Clustering. *Journal of Chemical Information and Modeling*, 62(23):6271–6286, 2022.

[96] Zhenqiu Shu, Yanwu Sun, Jiali Tang, and Congzhe You. Adaptive graph regularized deep semi-nonnegative matrix factorization for data representation. *Neural Processing Letters*, 54(6):5721–5739, 2022.

[97] Paris Smaragdis. Non-negative matrix factor deconvolution; extraction of multiple sound sources from monophonic inputs. In *Independent Component Analysis and Blind Signal Separation: Fifth International Conference, ICA 2004, Granada, Spain.*, pages 494–499. Springer, 2004.

[98] Paris Smaragdis. Convolutive speech bases and their application to supervised speech separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(1):1–12, 2006.

[99] Marina Sokolova and Guy Lapalme. A systematic analysis of performance measures for classification tasks. *Information processing & management*, 45(4):427–437, 2009.

[100] Hyun Ah Song, Bo-Kyeong Kim, Thanh Luong Xuan, and Soo-Young Lee. Hierarchical feature extraction by multi-layer non-negative matrix factorization network for classification task. *Neurocomputing*, 165:63–74, 2015.

[101] C Spearmen. General intelligence objectively determined and measured. *American Journal of Psychology*, 15:107–197, 1904.

[102] Douglas Steinley. Properties of the hubert-arable adjusted rand index. *Psychological methods*, 9(3):386, 2004.

[103] Haichao Sun and Jie Yang. The generalization of non-negative matrix factorization based on algorithmic stability. *Electronics*, 12(5):1147, 2023.

[104] Antonio J Tallón-Ballesteros and José C Riquelme. Data mining methods applied to a digital forensics task for supervised machine learning. *Computational intelligence in digital forensics: forensic investigation and applications*, pages 413–428, 2014.

[105] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[106] Alaa Tharwat. Classification assessment methods. *Applied computing and informatics*, 17(1):168–192, 2020.

[107] Ming Tong, Yiran Chen, Mengao Zhao, Haili Bu, and Shengnan Xi. A deep discriminative and robust nonnegative matrix factorization network method with soft label constraint. *Neural Computing and Applications*, 31(11):7447–7475, 2019.

[108] Warren S Torgerson. Multidimensional scaling: I. Theory and method. *Psychometrika*, 17(4):401–419, 1952.

[109] George Trigeorgis, Konstantinos Bousmalis, Stefanos Zafeiriou, and Bjoern Schuller. A deep semi-nmf model for learning hidden representations. In *International conference on machine learning*, pages 1692–1700. PMLR, 2014.

[110] George Trigeorgis, Konstantinos Bousmalis, Stefanos Zafeiriou, and Björn W Schuller. A deep matrix factorization method for learning attribute representations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(3):417–429, 2016.

[111] Laurens van der Maaten. Learning a Parametric Embedding by Preserving Local Structure. In David van Dyk and Max Welling, editors, *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*, volume 5 of *Proceedings of Machine Learning Research*, pages 384–391, Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA, 16–18 Apr 2009. PMLR.

[112] Jarkko Venna and Samuel Kaski. Neighborhood Preservation in Nonlinear Projection Methods: An Experimental Study. In Georg Dorffner, Horst Bischof, and Kurt" Hornik, editors, *Artificial Neural Networks — ICANN 2001*, pages 485–491, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg.

[113] Paolo Vineis and Alberto Rainoldi. Neural networks and logistic regression: analysis of a case-control study on myocardial infarction. *Journal of clinical epidemiology*, 50(11):1309–1310, 1997.

[114] Nguyen Xuan Vinh and Julien Epps. Information theoretic measures for clusterings comparison: variants, properties, normalization and correction for chance. *Journal of Machine Learning Research*, 11:2837–2854, 2010.

[115] Nguyen Xuan Vinh, Julien Epps, and James Bailey. Information theoretic measures for clusterings comparison: is a correction for chance necessary? In *26th annual international conference on machine learning (ICML'09)*, pages 1073–1080, 2009.

[116] Michael E Wall, Andreas Rechtsteiner, and Luis M Rocha. Singular value decomposition and principal component analysis. In *A Practical Approach to Microarray Data Analysis*, pages 91–109. Springer, 2003.

[117] Dexian Wang, Tianrui Li, Ping Deng, Jia Liu, Wei Huang, and Fan Zhang. A generalized deep learning algorithm based on NMF for multi-view clustering. *IEEE Transactions on Big Data*, 9(1):328–340, 2022.

[118] Juan Wang, Siyu Lai, and Mingdong Li. Improved image fusion method based on NSCT and accelerated NMF. *Sensors*, 12(5):5872–5887, 2012.

[119] Yu-Xiong Wang and Yu-Jin Zhang. Nonnegative matrix factorization: A comprehensive review. *IEEE Transactions on Knowledge and Data Engineering*, 25(6):1336–1353, 2012.

[120] Max Welling and Markus Weber. Positive tensor factorization. *Pattern Recognition Letters*, 22(12):1255–1261, 2001.

[121] Mingming Yang and Songhua Xu. Orthogonal Nonnegative Matrix Factorization using a novel deep Autoencoder Network. *Knowledge-Based Systems*, 227:107236, 2021.

[122] Fanghua Ye, Chuan Chen, and Zibin Zheng. Deep autoencoder-like nonnegative matrix factorization for community detection. In *Proceedings of the 27th ACM international conference on information and knowledge management*, pages 1393–1402, 2018.

[123] Jiangtao Yin, Lixin Gao, and Zhongfei Zhang. Scalable nonnegative matrix factorization with block-wise updates. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2014, Nancy, France, September 15-19, 2014. Proceedings, Part III 14*, pages 337–352. Springer, 2014.

[124] Jiho Yoo and Seungjin Choi. Orthogonal nonnegative matrix tri-factorization for co-clustering: Multiplicative updates on stiefel manifolds. *Information processing & management*, 46(5):559–570, 2010.

[125] Jinshi Yu, Guoxu Zhou, Andrzej Cichocki, and Shengli Xie. Learning the hierarchical parts of objects by deep non-smooth nonnegative matrix factorization. *IEEE Access*, 6:58096–58105, 2018.

[126] Stefanos Zafeiriou, Anastasios Tefas, Ioan Buciu, and Ioannis Pitas. Exploiting discriminant information in nonnegative matrix factorization with application to frontal face verification. *IEEE Transactions on Neural Networks*, 17(3):683–695, 2006.

[127] Zihao Zhan, Wen-Sheng Chen, Binbin Pan, and Bo Chen. Deep Grouped Non-Negative Matrix Factorization Method for Image Data Representation. In *2021 International Conference on Machine Learning and Cybernetics (ICMLC)*, pages 1–6. IEEE, 2021.

[128] Daoqiang Zhang, Zhi-Hua Zhou, and Songcan Chen. Non-negative matrix factorization on kernels. In *PRICAI 2006: Trends in Artificial Intelligence: 9th Pacific Rim International Conference on Artificial Intelligence Guilin, China.*, pages 404–412. Springer, 2006.

[129] Hui Zhang, Huaping Liu, Rui Song, and Fuchun Sun. Nonlinear Non-negative Matrix Factorization using Deep Learning. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 477–482. IEEE, 2016.

[130] Sheng Zhang, Weihong Wang, James Ford, and Fillia Makedon. Learning from incomplete ratings using non-negative matrix factorization. In *2006 SIAM international conference on data mining*, pages 549–553. SIAM, 2006.

[131] Yang Zhao, Huiyang Wang, and Jihong Pei. Deep non-negative matrix factorization architecture based on underlying basis images learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(6):1897–1913, 2019.

[132] Hui Zou, Trevor Hastie, and Robert Tibshirani. Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, 15(2):265–286, 2006.