2015

Incremental Sparse-PCA Feature Extraction For Data Streams

Jean-Pierre Nziga
Nova Southeastern University, jn461@nova.edu

NSUWorks Citation

Jean-Pierre Nziga. 2015. *Incremental Sparse-PCA Feature Extraction For Data Streams.* Doctoral dissertation. Nova Southeastern University. Retrieved from NSUWorks, College of Engineering and Computing. (365) http://nsuworks.nova.edu/gscis_etd/365.

Incremental Sparse-PCA Feature Extraction For Data Streams

By

Jean-Pierre Nziga

A dissertation submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in
Computer Information Systems

College of Engineering and Computing
Nova Southeastern University

2015

We hereby certify that this dissertation, submitted by Jean-Pierre Nziga, conforms to acceptable standards and is fully adequate in scope and quality to fulfill the dissertation requirements for the degree of Doctor of Philosophy.

James Cannady, Ph.D.　　　　Date
Chairperson of Dissertation Committee

Paul Cerkez, Ph.D.　　　　Date
Dissertation Committee Member

Rita M. Barrios, Ph.D.　　　　Date
Dissertation Committee Member

Approved:

Amon B. Seagull, Ph.D.　　　　Date

Interim Dean, College of Engineering and Computing

College of Engineering and Computing

Nova Southeastern University

2015

An Abstract of a Dissertation Submitted to Nova Southeastern University
in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

Incremental Sparse-PCA Feature Extraction For Data Streams

By
Jean-Pierre Nziga

October 2015

Intruders attempt to penetrate commercial systems daily and cause considerable financial losses for individuals and organizations. Intrusion detection systems monitor network events to detect computer security threats. An extensive amount of network data is devoted to detecting malicious activities.

Storing, processing, and analyzing the massive volume of data is costly and indicate the need to find efficient methods to perform network data reduction that does not require the data to be first captured and stored. A better approach allows the extraction of useful variables from data streams in real time and in a single pass. The removal of irrelevant attributes reduces the data to be fed to the intrusion detection system (IDS) and shortens the analysis time while improving the classification accuracy. This dissertation introduces an online, real time, data processing method for knowledge extraction.

This incremental feature extraction is based on two approaches. First, Chunk Incremental Principal Component Analysis (CIPCA) detects intrusion in data streams. Then, two novel incremental feature extraction methods, Incremental Structured Sparse PCA (ISSPCA) and Incremental Generalized Power Method Sparse PCA (IGSPCA), find malicious elements. Metrics helped compare the performance of all methods.

The IGSPCA was found to perform as well as or better than CIPCA overall in term of dimensionality reduction, classification accuracy, and learning time. ISSPCA yielded better results for higher chunk values and greater accumulation ratio thresholds. CIPCA and IGSPCA reduced the IDS dataset to 10 principal components as opposed to 14 eigenvectors for ISSPCA. ISSPCA is more expensive in terms of learning time in comparison to the other techniques.

This dissertation presents new methods that perform feature extraction from continuous data streams to find the small number of features necessary to express the most data variance. Data subsets derived from a few important variables render their interpretation easier.

Another goal of this dissertation was to propose incremental sparse PCA algorithms capable to process data with concept drift and concept shift. Experiments using WaveForm and WaveFormNoise datasets confirmed this ability. Similar to CIPCA, the ISSPCA and IGSPCA updated eigen-axes as a function of the accumulation ratio value, forming informative eigenspace with few eigenvectors.

Acknowledgments

This work would not have been possible without the support and help of many people. The author would like to express his gratitude to all those who contributed to the completion of this degree.

I am particularly thankful to my committee chair, Professor James Cannady for his guidance, assistance and patience throughout the dissertation process. I am also extending my appreciation to my committee members, Dr. Paul Cerkez for his advice and multiple revisions of this dissertation, Dr. Rita Barrios for her critique and valuable suggestions.

I am grateful to my entire family and friends for their encouragement and support over the years through the completion of this dissertation work.

I am very grateful to my wife, Dr. Yolande Nziga for her encouragement and assistance during this Ph.D. program.

I am also dedicating this dissertation to my lovely children, Lazlo and Darnell.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Operational information systems such as web traffic, face recognition programs, sensor measurements, and surveillance continuously generate large amounts of data to be mined for pattern discovery. Applications such as network intrusion detection systems (IDS) generate continuous streams of data to be analyzed in real time (Akhtar, 2011). Extracting useful knowledge from data streams is a difficult task. Existing approaches store the whole data off-line before analysis in a batch mode (Hebrail, 2008).

Batch processing of static datasets impacts the processing speed and requires large memory capacity, resulting in the necessity to develop methods to extract meaningful features from continuous data streams in real time (Chandrika & Kumar, 2011). An efficient algorithm should extract the optimal fraction of data elements sufficient to improve the analysis performance and obtain insights from systems under consideration. Existing data patterns need to be updated as new streams arrive.

Incremental principal component analysis (PCA) approaches have been proposed with the expectation to achieve the dimensionality reduction of data streams. PCA reduces dimensionality by projecting the dataset onto principal subsets to find components that display the maximum data variance (Nziga, 2011; Nziga & Cannady, 2012). Leading principal components are linear combinations of all the original variables, rendering their interpretation difficult. A successful dimensionality reduction technique that finds principal components with maximum variance of the dataset while combining few variables improves the interpretability and analysis of the data. In intrusion detection, it would be greatly beneficial to retrieve and analyze just the subset of variables to detect network unauthorized accesses. This result can be obtained with a method based on sparse PCA.

Sparse PCA produces modified principal components with sparse loadings (Zou, Hastie, & Tibshirani, 2006); each component is modeled as a linear combination of the subset's original attributes. However, the sparse PCA algorithms proposed so far process only static datasets. In order to reduce the dimensionality of continuous data streams, this dissertation describes incremental sparse PCA techniques. This dissertation presents the experimental results of using the sparse PCA methods on several datasets and comparing them to the output of the chunk incremental PCA algorithm.

Problem Statement and Goal

Mining of non-stopping data streams is computationally challenging. Existing feature selection and attributes extraction approaches perform poorly in locating relevant features from data streams that grow without a limit at a rate of several million or billion records per day (Domingos & Hulten, 2000). Traditional feature selection algorithms are designed and tuned for applications, requiring that the data be stored prior to processing off-line. Increasing numbers of applications, such as network intrusion detection systems, telecommunications, real-time surveillance, sensor networks, stock market tracking systems, road traffic analysis, and weather forecasting systems generate continuous streams of data to be processed online and in real time for knowledge extraction. The goal of this research was to implement improved methods that perform feature extraction from these continuous data streams.

Relevance and Significance

The tremendous increase in the amount of data produced in operational information systems such as web traffic, face recognition programs, sensor measurements, surveillance, and so on renders their mining for finding useful and unknown patterns difficult using the old paradigm of storing data before analysis (Hebrail, 2008). Furthermore, a plethora of feature extraction approaches proposed in the literature are inefficient solutions for continuous streams of data because of their reliance on static datasets or pre-available sets of data in a batch mode (Aboalsamh, Mathkour, Assassa, & Mursi, 2009; Dagher, 2010; Ohta & Ozawa, 2009; Ozawa, Pang, & Kasabov, 2008; Ozawa, Takeuchi, & Abe, 2010). Therefore, data streams cannot be mined for knowledge using algorithms developed for static datasets. Continuous data streams are captured in real time and fed to systems for online analysis. It is important to develop methods capable of extracting relevant features from continuous data streams in real time. Achieving this goal will address the limitations of existing feature selection approaches (offline processing of static dataset in a batch mode, poor processing speed, high memory requirement, and poor performance with large amount of data).

Contrary to previous studies of dimensionality reduction, which process static datasets in a batch and offline mode, this research focused on processing large and continuous numbers of data streams generated by applications such as network intrusion detection systems and consisting of millions or billions of records per day to extract the embedded knowledge and take appropriate action in a timely manner (Chandrika & Kumar, 2011). Dimensionality reduction is the process of reducing the number of random variables under consideration (Roweis & Saul, 2000). The process can be divided into feature selection and feature extraction. Feature selection techniques find a subset of the original variables or attributes. Feature extraction approaches, on the other hand, transform high-dimensional space data to a space of fewer dimensions. Challenges of mining data streams include minimizing resources requirements while providing acceptable results. The purpose of this research was to introduce a method that finds the minimum amount of relevant data from continuous data streams, resulting in a successful knowledge extraction and behavioral interpretations of applications under consideration. This algorithm will dynamically extract the optimal fraction of data elements sufficient to improve the analysis performance and obtain insights from the systems under analysis.

This research began by leveraging some incremental feature extraction approaches previously proposed for facial recognition applications (Aboalsamh et al., 2009; Ding, Tian, & Xu, 2009; Ozawa et al., 2008; Tokumoto & Ozawa, 2011; Yan & Liu, 2012). Facial recognition systems aim to automatically identify a person using facial features from large database of images. The approach in this research is capable of extracting features from continuous computer network data streams. Performance metrics of this approach, such as accuracy, speed, and memory use, were compared to those of well-known features extraction algorithms, using a real-world application such as network data for intrusion detection.

This study's feature extraction methods should be efficient and capable of detecting changing concepts in data distribution due to the highly dynamic nature of data streams (Chandrika & Kumar, 2011; Kholghi & Keyvanpour, 2011). Sliding window and forgetting factor approaches are considered to account for changes in data streams if necessary. Sliding window technique limits the amount of data streams being fed to the learner. It is a deterministic approach that prevents stale data from influencing the data analysis. Sliding window is popularly considered when handling evolving data (Bifet & Gavaldà, 2006, 2007; Datar, Gionis, Indyk, & Motwani, 2002; Guha & Koudas, 2002; Ikonomovska, Gorgevik, & Loskovska, 2007); forgetting factor moderates the balance between old and new observations (Levy & Lindenbaum, 2000). This is achieved by multiplying the previous singular values at each update by a scalar factor $f \in [0, 1]$.

Barriers and Issues

The majority of dimensionality reduction approaches proposed in the literature are designed to statically process a previously collected dataset (Chandrika & Kumar, 2011; Kholghi & Keyvanpour, 2011). However, an increasing number of applications such as network intrusion detection systems, stock market, sensor networks, telecommunication systems, and web applications generate continuous streams of data to be processed and analyzed in real time in order to react accordingly. Researchers, increasingly interested in online data extraction techniques and algorithms, have recently proposed incremental dimensionality reduction to improve facial recognition systems by analyzing the data stream instead of static datasets (Aboalsamh et al., 2009; Ding et al., 2009; Ozawa et al., 2008; Tokumoto & Ozawa, 2011; Yan & Liu, 2012). A facial recognition system is a computer application that aims to identify a person from a digital image automatically. Recognition is achieved by comparing facial features extracted from an image using algorithms to a facial database. The majority of facial recognition systems use one of the following algorithms for features extraction: PCA, LDA, hidden Markov model, or the multi-linear subspace learning. Similarly, intrusion detection systems extract relevant features from computer network data stream as a pre-processing step to pattern matching phase to keep intruders out.

Feature extraction algorithms designed for data streams that have yielded great results in other fields of research, such as chunk incremental principal component analysis, were implemented for the purpose of this research. Another issue considered in this research was the fact that one-pass incremental learning presents two important problems, as noted by Ozawa et al. (2008). First, the data stream is continuous; hence, it is impossible to keep part of training data to be utilized for learning. Moreover, the distribution of data is unknown, making it difficult to extract essential features only from initial training samples.

Definitions of Terms

Term	Definition
Dimensionality reduction	Process of reducing the number of random variables under consideration
Facial recognition system	Computer application that identifies a person from digital image, comparing facial features extracted from an image using algorithms to a facial database
Feature extraction	Transform high-dimensional space data to a space of fewer dimensions
Feature selection	Find a subset of the original variables or attributes
Forgetting factor	Moderate the balance between old and new observations
Sliding window technique	Limit the amount of data streams being fed to the learner

List of Acronyms

Acronym	Definition
ACO-SA	Ant Colony Optimization and Simulated Annealing
ACOMI	Ant-Colony Optimization and Mutual Information
BCD	Block Coordinate Descent
CCIPCA	Candid Covariance-free IPCA
CIPCA	Chunk Incremental Principal Component
DM	Diffusion Maps
DSPCA	Semi-definite programming Sparse Principal Component Analysis
FastMVU	Fast Maximum Variance Unfolding
GDA	Generalized Discriminant Analysis
GPSPCA	Generalized Power Method Sparse Principal Component Analysis
HLLE	Hessian Local Linear Embedding
IDS	Intrusion Detection Systems
IGSPCA	Incremental Generalized Power Method Sparse Principal Component Analysis
IKPCA	Incremental Kernel Principal Component Analysis
ILDA	Incremental Linear Discriminant Analysis
IPCA	Incremental Principal Component Analysis
IRFLD	Incremental Recursive Fisher Linear Discriminant
ISPCA	Incremental Sparse Principal Component Analysis
ISSPCA	Incremental Structured Sparse Principal Component Analysis
KPC	Kernel Principal Components
KPCA	Kernel Principal Component Analysis
LDA	Linear Discriminant Analysis
LE	Laplacian Eigenmaps
LLE	Local Linear Embedding
LTSA	Local Tangent Space Analysis
MCA	Minor Components Analysis
MDS	Multi-Dimensional Scaling
MI	Mutual Information
mRMR	minimum-Redundancy Maximum-Relevancy
PCA	Principal Component Analysis
PSO-MI	Particle Swarm Optimization method and Mutual Information
RFLD	Recursive Fisher Linear Discriminant
SEA	Streaming Ensemble Algorithm
SFA	Slow Feature Analysis
SNE	Stochastic Neighbor Embedding
SPCA	Sparse Principal Component Analysis
SPE	Stochastic Proximity Embedding

SSPCA	Structured Sparse Principal Component Analysis
SVD	Singular Value Decomposition
SVM-RFE	Support Vector Recursive Feature Elimination
T-IKPCA	Takeuchi Incremental Kernel Principal Component Analysis

Summary

Operational information systems generate continuous large amount of data to mine for knowledge discovery. Irrelevant and redundant attributes slow down the learning process and consume more computing resources. Dimensionality reduction contributes to reduce the number of random variables under consideration (Roweis & Saul, 2000).

PCA analyzes the interdependency between attributes to map the data to a lower dimensional space (because the size of attributes are lower than in the original dataset), such that the variance of the data in the low-dimensional representation is maximized. Unfortunately, PCA lacks sparseness of the principal vectors. Sparse principal component analysis (SPCA) addresses these limitations by modifying principal components with sparse loadings. SPCA methods adjust the PCA approaches by injecting sparseness into the loading vectors, similar to regularization methods, which inject sparseness to the parameter estimates in the regression setting.

This dissertation presents an incremental SPCA approach to extract features from data streams in real time. The goal was to find the minimum fraction of the original data that provides the maximum insight about the application under consideration. Metrics used in this dissertation have been representative and useful as benchmarks for comparison in well-known research studies.

Chapter 2

Review of the Literature

The high volume of data generated by today’s applications makes training and testing using classification methods difficult. Irrelevant and redundant attributes slow down the learning process, confuse learning algorithms, and consume more resources while increasing the classifier’s risk of over-fitting (Yu & Lui, 2003). Kohavi and John (1997) demonstrated that redundant and irrelevant features negatively impact the prediction accuracy of machine learning algorithms. The research community continues to develop data mining and machine learning algorithms for data pre-processing, classification, clustering, association rules, and virtualization. Feature selection techniques are extensively used for pattern recognition in data preprocessing, data mining, and machine learning to remove redundant and irrelevant features from high dimensional datasets in order to select a subset of relevant features to build robust learning models (Fukunaga, 1990). Feature extraction is another dimensionality reduction approach that transforms high-dimensional space data to a space of fewer dimensions. The purpose of the dimensionality reduction goal is to reduce the number of random variables under consideration (Roweis & Saul, 2000).

Feature Selection

According to Lui and Yu (2005), the large majority of feature selection algorithms can be classified under four categories: wrapper, filter, hybrid, and embedded.

Wrapper-based feature selection methods validate the goodness of a subset of features using a learning algorithm, as opposed to the filter-based feature selection algorithms that use metrics to assess the usefulness of any single feature (Guyon & Elisseeff, 2003). The process ends when an optimal set of algorithms is generated. Wrapper selection methods search for possible features through the dataset using search algorithms with the subset being constantly evaluated. By using a learning algorithm for features selection, wrapper methods are more accurate than filter methods. The main drawbacks of wrapper-based algorithms are their requirement for vast computational resources, in addition to their operation risk of over fitting the learning algorithm (Kohavi & John, 1997; Kohavi & Sommerfield, 1995).

Filter-based feature selection algorithms use metrics to classify each feature. Low ranking features are eliminated (Ahmad, Norwawi, Deris, & Othman, 2008). The intrinsic characteristic of the data is considered in evaluating the fitness of the feature subset. Filter-based feature selection techniques do not use a learning algorithm and require fewer computer resources. However, the resulting subset of features may not be good matches for classification algorithms (Zhu, Ong, & Dash, 2007).

Hybrid-feature selection methods assess the validity of subsets from the original dataset using an independent measure in conjunction with a learning algorithm (Das, 2001; Lui & Yu, 2005). There are many examples of this method in the literature. C.-K. Zhang and Hu (2005) proposed a hybrid-feature selection based on ant-colony optimization and mutual information (ACOMI) for forecasters at the Australian Bureau of Meteorology. Ant-colony optimization is used to find colonies between data points. The results were better than either of the individual ant-colony optimization or mutual information approaches.

Khushaba, Al-Ani, and Al-Jumaily (2007) proposed a feature-selection algorithm based on a mixture of particle swarm optimization method and mutual information (PSO-MI). PSO-MI showed an improved accuracy on a dataset of transient myoelectric signal, compared to particle swarm optimization or mutual information used separately. Y. Zhang, Ding, and Li (2007) presented a two-stage selection algorithm by combining ReliefF and minimum-Redundancy Maximum-Relevancy (mRMR) for gene expression data. The authors performed experiments comparing the mRMR-ReliefF selection algorithm with ReliefF, mRMR, and other feature selection methods using two classifiers: Support Vector Machine (SVM) and Naive Bayes. The authors used seven different datasets. According to the authors, experiments showed improved results using mRMR-ReliefF algorithm for gene selection compared to that of mRMR or ReliefF used separately.

Firouzi, Niknam, and Nayeripour (2008) proposed a hybrid evolutionary algorithm based on the combination of ant colony optimization and simulated annealing (ACO-SA). The researchers chose cluster center with the help of ACO and SA in order to achieve global optima. Ant colony optimization is used to find colonies between data points. Simulated annealing is a good local search algorithm for finding the best global position using the cumulative probability. Michelakos, Papageorgiou, and Vasilakopoulos (2010a) proposed a hybrid algorithm combining the cAnt-Miner2 and the mRMR feature selection algorithms. cAnt-Miner2 algorithm (Michelakos, Papageorgiou, & Vasilakopoulos, 2010b) is an extended approach of coping with continuous attributes introduced by the cAnt-Miner algorithm. cAnt-Miner algorithm (Otero, Freitas, & Jonhson, 2008) is an extension of Ant-Miner (Parpinelli, Lopes, & Freitas, 2002).

Ant-Miner copes with continuous attributes and therefore incorporates an entropy-based discretization method during the rule construction process. cAnt-Miner creates discrete intervals for continuous attributes on the fly and does not require a discretization method for preprocessing. Experimental results of the combination of cAnt-Miner2 and mRMR using public medical data sets yielded improved results compared to that of cAnt-Miner2 only. The proposed combination was better in terms of accuracy, simplicity, and computational cost compared to the original cAnt-Miner2 algorithm.

Mundra and Rajapakse (2010) proposed the support vector recursive feature elimination (SVM-RFE) for gene selection incorporating an mRMR filter. According to the authors, the approach improved the identification of cancer tissue from benign tissues on several benchmark datasets because it accounted for the redundancy among the genes compared to mRMR or SVM-RFE separately. Hossain, Pickering, and Jia (2011) proposed an approach for hyper-spectral data dimensionality reduction based on a measure of mutual information (MI) and principal components analysis (PCA) called MI-PCA, using a mutual information measure to find principal components, which are spatially most similar to all the target classes. The authors conducted experiments using hyper-spectral data with 191 bands covering the Washington, DC, area; results showed that two features selected from 191 using MI-PCA provided 98% and 93% classification accuracy for training and test data respectively, with a support vector machines classifier.

Feature Extraction

Other researchers considered feature extraction over feature selection for dimensionality reduction purposes (Agrafiotis, 2003; Donoho & Grimes, 2005; Duraiswami & Raykar, 2005; Hoffmann, 2007; Huber, Ramoser, Mayer, Penz, & Rubik, 2005; K. I. Kim, Jung, & Kim, 2002; Shawe-Taylor & Christianini, 2004; Zou et al., 2006). Feature extraction converts the data in the high-dimensional space to a lower dimensional space. Van der Maaten, Postma, and van den Herik (2009) compared several of these dimensionality reduction techniques in a technical report as described below.

Linear Dimensionality Reduction Techniques

Principal Component Analysis

Principal component analysis (PCA) is a linear method that reduces data dimensionality by performing a covariance analysis between factors as described by Hotelling (1933) and Pearson (1901) in their seminal works. PCA analyzes the interdependency between pairs of attributes to identify significant ones and performs a linear mapping of the data to a lower dimensional space (size of attributes lower than in the original dataset) such that the variance of the data in the low-dimensional representation is maximized. PCA constructs the data correlation matrix and computes eigenvectors matrixes. Eigenvectors that correspond to largest principal components are used to reconstruct a large fraction of the variance of the original data.

The goal of PCA is to find the matrix/vector Y such that:

$$Y = w\,X \tag{1}$$

Where: Y = $\boldsymbol{m}$-dimensional projected vector

X = the original $\boldsymbol{d}$-dimensional data vector

w = an *m*-by-*m* matrix where columns are the eigenvectors of $X^{\mathrm{T}} X$

The ***m*** projection vectors maximizing the variance of *Y* are derived from the eigenvectors $\boldsymbol{e_1}, \boldsymbol{e_2}, \boldsymbol{e_3} \ldots \boldsymbol{e_m}$ of the data set's covariance matrix ***E*** associated with the largest ***m*** eigenvalues.

The data covariance matrix is the following:

$$E = (n-1)^{-1} \sum_{i=1}^{n} (x_i - \mu)(x_i - \mu)^T \qquad (2)$$

The eigenvectors and eigenvalues are obtained by solving the following equations:

$$(E - \lambda_i I)e_i = 0, I = 1, \ldots, d \qquad (3)$$

PCA has shown great performance in various applications such as face recognition (Turk & Pentland, 1991), coin classification (Huber et al., 2005), and seismic series analysis (Posadas et al., 1993). PCA shows some limitations if the data has a very high dimension. For example, the computation of the eigenvectors might not be possible because the size of the covariance is proportional to the dimensionality of the data point. Therefore, performing PCA can be costly. An *N* dimensional matrix requires N^3 matrix inversion operations. Also, N^2 operations are required to store covariance matrix (Hotelling, 1933; Pearson, 1901).

PCA has shown great performance in various applications, including network intrusion detection (Nziga, 2011; Nziga & Cannady, 2012). However, the algorithms developed in these publications require that the data be stored and processed in a batch mode.

Linear Discriminant Analysis

Linear discriminant analysis (LDA) is a supervised technique that maximizes the linear reliability between data of different classes (Fisher, 1936). LDA finds linear mapping, maximizing the linear class separability in the reduced dimensionality of the data. Similar to PCA, LDA looks for linear combinations of variables that best represent the data. LDA models the difference between the classes of data. PCA does not take into account any difference in class. LDA's performance is optimal when dealing with continuous variables (variables with numeric values). LDA projections of continuous variables preserve complex structure in data for classification. LDA has shown improved classification results of large datasets in various applications such as speech recognition (Haeb-Umbach & Ney, 1992), document classification (Torkkola, 2001), and mammography (Chan et al., 1995).

Global Nonlinear Dimensionality Reduction Techniques

Global nonlinear dimensionality reduction techniques construct nonlinear transformation between a high dimensional dataset and its low dimensional representation while preserving global properties of the data.

Multidimensional Scaling

Multidimensional scaling (MDS) is a set of nonlinear techniques mapping the high dimensional dataset to a low dimensional representation and keeping the pairwise distances between data points whenever possible (Cox & Cox, 1994; Kruskal, 1964). MDS includes many different specific types that can be classified based on whether the similarities data are qualitative (called nonmetric MDS) or quantitative (metric MDS).

MDS is very popular for visualization of data (Tagaris et al., 1998) and in molecular modeling (Venkatarajan & Braun, 2004).

Stochastic Proximity Embedding

Stochastic proximity embedding (SPE) is a repetitive algorithm that employs an efficient rule, in comparison to MDS, to update the current estimate of the low dimensionality of the data (Agrafiotis, 2003). SPE minimizes the MDS raw stress function and is able to retain only distances in a neighborhood graph. SPE attempts to generate Euclidean coordinates for a set of data points to comply with a prescribed set of geometric constraints. The method begins with an initial configuration and iteratively refines it by repeatedly selecting pairs of objects at random and adjusting their coordinates so that their distances on the map match more closely their respective proximities. The adjustments are controlled by a learning rate parameter.

Isomap

Isomap attempts to resolve MDS limitations by incorporating the geodesic distances on a weighted graph. The generalization of the notion of a straight line to curved spaces is called geodesic. A geodesic is a locally length-minimizing curve. Isomap attempts to estimate the intrinsic geometry of a data manifold (dataset composed of many features and diverse elements) based on a rough estimate of each data point's neighbors (Tenenbaum, 1998). Isomap defines the geodesic distance to be the sum of edge weights along the shortest path between two nodes. The connectivity of each data point in the neighborhood graph is defined as its nearest *k* Euclidean neighbor in the high-dimensional space.

Fast Maximum Variance Unfolding

Fast maximum variance unfolding (FastMVU) defines a neighborhood graph on the data and retains pairwise distances in the resulting graph by minimizing the Euclidian distances between the data points. FastMVU begins with the construction of a neighborhood graph, connecting each data point to its given number of nearest neighbors. FastMVU attempts to maximize the sum of the squared Euclidian distances between datapoints (Weinberger, Sha, Zhu, & Saul, 2007).

Kernel PCA

Kernel PCA (KPCA) is an extension of PCA using kernel functions (Schoelkopf, Smola, & Mueller, 1998). KPCA computes the principal eigenvectors of the kernel matrix while the linear PCA computes those of the covariance matrix. KPCA constructs nonlinear mappings using the application of PCA in kernel space. The eigenvectors of the covariance matrix are scaled versions of the eigenvectors of the kernel matrix, and mappings performed by KPCA are closely related to the choice of the kernel function (Shawe-Taylor & Christianini, 2004). KPCA has shown encouraging results on face recognition (K. I. Kim et al., 2002), speech recognition (Lima et al., 2004), and novelty detection (Hoffmann, 2007).

Generalized Discriminant Analysis

Generalized discriminant analysis (GDA) or kernel LDA is the implementation of the LDA using kernel function (Baudat & Anouar, 2000). GDA maximizes the Fisher criterion using the kernel function in the high dimensional space. GDA deals with nonlinear discriminant analysis using kernel function operator to provide a mapping of the input vectors into high-dimensional feature space. Linear properties make it easy to extend and generalize the classical linear discriminant analysis (LDA) to nonlinear discriminant analysis. The formulation is expressed as an eigenvalue problem resolution.

Diffusion Maps

The diffusion maps (DM) framework originated from the field of dynamic systems (Lafon & Lee, 2006; Nadler, Lafon, Coifman, & Kevrekidis, 2006). DM is based on the definition of a Markov random walk on the graph of the data. The nonlinear method DM focuses on discovering the underlying manifold of the data, leveraging the relationship between heat diffusion and random walk Markov chain. DM gives a global description of the data set by integrating local similarities at different scales.

Stochastic Neighbor Embedding

Stochastic neighbor embedding (SNE) is a repetitive technique that aims to retain the pairwise distances between the data points in the low-dimensional representation of the data (Hinton & Roweis, 2002). In SNE, similarities of nearby points account more for the cost function, leading to a low-dimensional data representation that keeps mainly local properties of the manifold. According to the authors, SNE allows ambiguous objects, such as document count vector for the “word” bank, to have versions close to the images of both “river” and “finance” without forcing the images of outdoor concepts to be located close to those of corporate concepts.

Sparse Principal Component Analysis

PCA decomposition is a linear combination of the input coordinates where principal vectors form a low-dimensional subspace that corresponds to the direction of maximal variance in the data. PCA minimizes information loss and provides the closest linear subspace to the data (Zou et al., 2006). However, PCA lacks sparseness of the principal vectors, and linear combination may mix positive and negative weights, which might partly cancel each other. Sparse principal component analysis (SPCA) addresses these issues by modifying principal components with sparse loadings. SPCA methods adjust the PCA approaches by injecting sparseness to the loading vectors; this process is similar to regularization methods, which inject sparseness to the parameter estimates in the regression setting. Several approaches and algorithms performing SPCA have recently been proposed for batch processing of static dataset. Grbovic, Dance, and Vucetic (2012) proposed a methodology for adding two general types of feature grouping constraints into the original SPCA optimization procedure. D'Aspremont, El Ghaoui, Jordan, and Lanckriet (2007) proposed a direct formulation for SPCA using semidefinite programming (DSPCA). Jenatton, Obozinski, and Bach (2010) proposed a SPCA wherein the sparse patterns of all dictionary elements were structured and constrained to belong to a pre-specified set of shapes. Hein and Buhler (2010) proposed a nonlinear inverse power method for SPCA. Journee, Nesterov, Richtarik, and Sepulchre (2010) proposed a generalized power method for sparse principal component analysis.

Local Nonlinear Dimensionality Reduction Techniques

Local nonlinear dimensionality reduction techniques preserve properties of small neighborhoods around data points, therefore retaining the global layout of the data manifold for classification.

Local Linear Embedding

Local linear embedding (LLE) constructs a graph representation (Roweis & Saul, 2000). LLE attempts to preserve only local properties of the data by reducing sensitivity to short-circuiting in comparison with isomap and allowing for successful embedding of nonconvex manifolds. LLE writes the data points as a linear combination of their nearest neighbors and attempts to retain the reconstruction weights in the linear combinations. LLE has been applied with satisfaction in super-resolution, the problem of generating a high-resolution image from one or more low-resolution images (Chang, Yeung, & Xiong, 2004) and sound source localization (Duraiswami & Raykar, 2005).

Laplacian Eigenmaps

Laplacian eigenmaps (LE) use spectral techniques to perform dimensionality reduction, relying on the assumption that the data lies in a low dimensional manifold of a high dimensional space. Laplacian eigenmaps preserve local properties of the manifold in finding a reduced dimensionality of data representation (Belkin & Niyogi, 2002). Local properties are functions of the pairwise distances between near neighbors. Laplacian eigenmaps build a graph from the data set’s neighborhood information, with each data point serving as a node on the graph. The connectivity between nodes is governed by the proximity of neighboring points. Laplacian eigenmaps generate reduced dimensionality of a dataset by minimizing the distances between a data point and its *k* nearest neighbors in a weighted manner. The minimization of a cost function is based on the graph, ensuring that points close to each other on the manifold are mapped close to each other in the low dimensional space, thus preserving local distances. Applications of Laplacian eigenmaps on solid theoretical ground have shown some success, with graph Laplacian matrix converging to the Laplace–Beltrami operator as the number of points goes to infinity.

Hessian Local Linear Embedding

Hessian local linear embedding (HLLE) is a flavor of LLE, which minimizes the curviness of the large dataset into a low-dimensional space with the reduced dataset locally isometric (Donoho & Grimes, 2005). Based on sparse matrix techniques, HLLE yields results of a much higher quality than LLE. However, HLLE has a very costly computational complexity and therefore is not well suited for heavily-sampled manifolds.

Local Tangent Space Analysis

Local tangent space analysis (LTSA) describes local properties of the high-dimensional data using the local tangent space of each data point (Z. Zhang & Zha, 2002). LTSA assumes that there exists a linear mapping from a high-dimensional data point to its local tangent space if local linearity of the manifold is assumed. According to the authors, LTSA also assumes there exists a linear mapping from the corresponding low-dimensional data point to the same local tangent space. LTSA aligns linear mappings such that they construct the local tangent space of the manifold from the low-dimensional representation, simultaneously searching for the coordinates of the low-dimensional data representations and the linear mappings of the low-dimensional data points to the local tangent space of the high-dimensional data.

Incremental Dimensionality Reduction Approaches

The aforementioned dimensionality reduction techniques have displayed acceptable performance in extracting knowledge in various applications including data summarization and web data searching. These dimensionality reduction techniques were designed to perform on a stationary dataset and in off-line mode. Batch computation algorithms display limitations when dealing with large sets of data. These data mining algorithms were not designed for real-time data reductions. Increasing real world applications requires that the training set be dynamic, have an evolving nature, and be able to process continuous learning of new training data as they are added to the original set. A considerable number of applications generate a massive amount of continuous data streams, which must to be processed in real-time or stored online for interpretation, and then appropriate actions must be taken. Several incremental methods for the computation of reduced datasets have been proposed to address limitations of batch feature extraction approaches (Aboalsamh et al., 2009; Ding et al., 2009; Ozawa et al., 2008; Tokumoto & Ozawa, 2011; Yan & Liu, 2012). Incremental learning, also known as online learning, processes incoming streams sequentially while allowing the trained classifier to generate accuracy similar to that obtained with batch processing of the whole dataset. Incremental learning reads blocks of data at a time. Batch processing requires analyzing the complete dataset at once.

Incremental Principal Component Analysis

This method of analysis assumes that N training samples $\mathbf{a}^{(i)}$ are provided to a system initially: $\mathbf{a}^{(i)} \in \mathrm{R}^n$ $(i = 1, \ldots, N)$.

Applying PCA to the training samples produces the following eigenspace model:

$$\Omega = (\bar{\mathbf{a}}, \mathbf{U}_\mathrm{k}, \boldsymbol{\Lambda}_\mathrm{k}, N) \tag{4}$$

Where: $\bar{\mathbf{a}}$ is a mean vector of $\mathbf{a}^{(i)}$ $(i=1, \ldots, N)$,

$\mathbf{U}_\mathrm{k}$ is an n x k matrix with column vector corresponding to eigenvectors,

$\boldsymbol{\Lambda}_\mathrm{k} = \mathrm{diag}\{\lambda_1, \lambda_2, \ldots, \lambda_k\}$ is a k x k matrix with non-zero eigenvalues as diagonal elements. The value k determined as a function of certain criterion such as accumulation ratio, represents the number of eigen-axes spanning the eigenspace. The system computes Ω, keeps the information and throws away the entire training sample (Ozawa, Pang, & Kasabov, 2010).

Incremental Principal Component Analysis (IPCA) now assumes that the $(N + 1)^{th}$ training sample is provided as follows:

$$\mathbf{a}^{(N+1)} = \mathbf{y} \in \mathrm{R}^n \qquad (5)$$

This addition of the new sample creates changes in the mean vector and the covariance matrix, requiring the eigenspace model Ω to be updated. The new eigenspace model Ω' can be defined as follows:

$$\Omega' = (\bar{\mathbf{a}}', \mathbf{U}'_{k'}, \mathbf{\Lambda}'_{k'}, N+1) \qquad (6)$$

The eigenspace dimensions *k′* is k or *k+1* depending on whether or not **y** includes certain energy in the complementary eigenspace. The eigen-axes are rotated to adapt to the variation in the data distribution in three steps: mean vector update, eigenspace augmentation, and rotation of eigen-axes.

Mean vector update. This step is explained by the following equation*:*

$$\bar{\boldsymbol{a}}' = \frac{1}{N+1}(N\bar{\boldsymbol{a}} + \mathbf{y}) \in \mathrm{R}^n \qquad (7)$$

Eigenspace augmentation. Two criteria help decide whether the dimensional augmentation is needed or not. One of them is the norm of a residue vector defined below:

$$\mathbf{h} = (\mathbf{y} - \bar{\mathbf{a}}) - \mathbf{U}_k^T\,\mathbf{g} \qquad (8)$$

Where: $\mathrm{g} = \mathrm{U}_k^T (\mathrm{y} - \bar{\mathrm{a}})$

T represents the transposition of vectors and matrixes.

This criterion was adopted by the original IPCA proposed by Hall, Marshall, and Martin (1998). The other criterion is the accumulation ratio whose definition and incremental calculation are defined below,

$$A\,(\mathbf{U}_k) = \frac{\sum_{i=1}^{k} \lambda i}{\sum_{j=1}^{n} \lambda j} = \frac{(N+1)\sum_{i=1}^{k} \lambda i + \left\|\mathbf{U}_k^T\,(\mathbf{y} - \bar{a})\right\| \left\|\mathbf{U}_k^T\,(\mathbf{y} - \bar{a})\right\|}{(N+1)\ \sum_{j=1}^{n} \lambda j + \left\|\,(\mathbf{y} - \bar{a})\right\| \left\|(\mathbf{y} - \bar{a})\right\|} \quad (9)$$

where λ_i is the i^{th} largest eigenvalues, n represents the dimensionality of the input space and k the dimensionality of the current eigenspace. This criterion was used in the modified IPCA proposed by Ozawa, Pang, and Kasabov (2004).

The conditions on the eigenspace augmentation are represented respectively by:

$$[\text{Residue Vector Norm}] \quad \hat{\mathbf{h}} = \begin{cases} \frac{h}{||h||} & if \ \left||h|\right| > \eta \\ 0 & otherwise \end{cases} \quad (10)$$

$$[\text{Accumulation Ratio}] \quad \hat{\mathbf{h}} = \begin{cases} \frac{h}{||h||} & if \ A\,(\mathbf{U}_k) < \theta \\ 0 & otherwise \end{cases} \quad (11)$$

Eigenspace rotation. If either *Residue Vector Norm* or *Accumulation Ratio* above is satisfied, the dimension of the eigenspace increases from k to $k+1$. A new eigen-axis $\hat{\mathbf{h}}$ is added to the eigenvector matrix $\mathbf{U}_k$. If neither of those conditions is met, the dimensionality does not change. The eigen-axes are then rotated to adapt to the new data distribution. If the rotation is given by a rotation matrix $\mathbf{R}$, the eigenspace update is represented by the following:

- If there is a new eigen-axis to be added, $\mathbf{U}'_{k+1} = [\mathbf{U}'_k, \hat{\mathbf{h}}]\mathbf{R}$.
- Otherwise, $\mathbf{U}'_k = \mathbf{U}_k\mathbf{R}$.

Weng, Zhang, and Hwang (2003) proposed an incremental principal component analysis (IPCA) algorithm called candid covariance-free IPCA (CCIPCA) which computes the principal components of a sequence of samples incrementally without estimating the covariance matrix (or covariance-free). The method keeps the scale of observations and computes the mean of observations incrementally. However, the highest possible efficiency is not guaranteed in case of unknown sample distribution. While the method is designed for real-time applications, it does not allow iterations.

Zhao, Yuen, and Kwok (2006) pointed to the lack of guarantee on the approximation error as a major limitation of existing IPCA methods. They then proposed a new IPCA method based on the idea of singular value decomposition (SVD) updating algorithm called SVDU-IPCA for face-recognition. SVDU-IPCA approximation error is bounded. SVDU-IPCA algorithm can be easily extended to a kernel version. The authors claimed that experimental results show that the difference of the average recognition accuracy between the proposed incremental method and the batch-mode method is less than 1%.

Ross, Lim, Lin, and Yang (2008) attempted to address the limitations of existing algorithms that tracked objects well in controlled environments but failed in the presence of significant variation of the object's appearance or surrounding illumination. The authors proposed a tracking method that incrementally learns a low-dimensional subspace representation and adapts online to changes in the appearance of the target. The model is based on incremental algorithms for principal component analysis and includes two features: a method for correctly updating the sample mean and a forgetting factor to ensure less modeling power is expended on fitting older observations. These two features improved the overall tracking performance. According to the authors, experiments demonstrated the effectiveness of the tracking algorithm in indoor and outdoor environments where the target objects underwent large changes in pose, scale, and illumination.

Ozawa et al. (2008) presented a pattern classification system in which feature extraction and classifier learning were simultaneously carried both online and in one pass where training samples were presented only once. The authors extended incremental principal component analysis in combination with classifier models. Training samples must be learned one by one due to the limitation of IPCA. To overcome this problem, chunk IPCA was proposed, in which a chunk of training samples is processed at a time. The authors conducted experiments using several large-scale data sets to demonstrate the scalability of chunk IPCA under one-pass incremental learning environments. Results suggested that chunk IPCA can reduce the training time more effectively than IPCA, unless the number of input attributes is too large.

Aboalsamh et al. (2009) proposed various incremental PCA training and relearning strategies applicable to the candid covariance-free incremental principal component algorithm. The authors studied the effect of the number of increments and sizes of the eigen-vectors on the correct rate of face recognition. The results suggested that batch PCA is inferior to methods IPCA1 through 4 (Aboalsamh et al., 2009) and that all IPCAs are practically equivalent with IPCA3, yielding slightly better results than the other IPCAs.

Ding et al. (2009) proposed an adaptive approach for online extraction of the kernel principal components (KPC). First, a kernel covariance matrix is correctly updated to adapt to the changing characteristics of data. Second, KPC are recursively formulated to overcome the batch nature of standard KPCA, deriving the formulation from the recursive eigen-decomposition of kernel covariance matrix and indicating the KPC variation caused by the new data. The method alleviates the sub-optimality of the KPCA method for non-stationary data, in addition to maintaining a constant update speed and memory usage as the data size increases. According to the authors, experiments showed improvements in both computational speed and approximation accuracy.

Dagher (2010) introduced a recursive algorithm of calculating the discriminant features of the PCA-LDA procedure. The algorithm computed the principal components of a sequence of vectors incrementally without estimating the covariance matrix (meaning covariance-free) while computing the linear discriminant directions along which the classes are well separated. The procedure merges two algorithms based on principal component analysis (PCA) and linear discriminant analysis (LDA) running sequentially. Experiments were applied to face recognition problems, and results showed a high average success rate of the proposed algorithm compared to PCA, LDA, and PCA-LDA algorithms in batch mode.

Tokumoto and Ozawa (2011) proposed an incremental learning algorithm of kernel principal component analysis (IKPCA) for online feature extraction in pattern recognition problems by extending the incremental KPCA or T-IKPCA proposed by Takeuchi, Ozawa, and Abe (2007). T-IKPCA is able to learn new data incrementally without keeping past training data. T-IKPCA learns data chunk individually in order to update the eigen-feature space. T-IKPCA performs the eigenvalue decomposition for every data in the chunk. The authors extended T-IKPCA such that an eigen-feature space learning is conducted by performing the eigenvalue decomposition only once for a chunk of given data. For each new chunk of training data, IKPCA first selects linearly independent data based on the cumulative proportion. Then, the eigenspace augmentation is conducted by calculating the coefficients for the selected linearly independent data, and the eigen-feature space is rotated based on the rotation matrix that can be obtained by solving a kernel eigenvalue problem. Experiments showed that IKPCA can learn an eigen-feature space very fast without sacrificing the recognition accuracy.

Kompella, Luciw, and Schmidhuber (2011) proposed an incremental version of slow feature analysis (SFA) called IncSFA by combining incremental principal components analysis and minor components analysis (MCA). According to the authors, IncSFA, along with non-stationary environments, is amenable to episodic training, is not corrupted by outliers, and is covariance-free, unlike standard batch-based SFA. These properties make IncSFA a useful unsupervised preprocessor for autonomous learning agents and robots. In IncSFA, the CCIPCA and MCA updates take the form of Hebbian and anti-Hebbian updating, extending the biological plausibility of SFA. In both single node and deep network versions, IncSFA learns to encode its input streams (such as high-dimensional video) by informative slow features representing meaningful abstract environmental properties. It can handle cases where batch SFA fails. Hierarchical IncSFA derives the driving forces from a complex and continuous input video stream in a completely online and unsupervised manner.

Yan and Liu (2012) proposed an approach to retrieve an image in a data stream using principle component analysis by subdividing image into several blocks and extracting image principle features. According to the authors, the proposed approach efficiently retrieved an image and met the needs of wide bandwidth network traffic.

Incremental Linear Discriminant Analysis

According to Fukunaga (1990), there exist equivalent variants of Fisher's criterion to generate the projection matrix U and maximize class separability of the dataset:

$$\arg\max_{U} \frac{|U^T S_B U|}{|U^T S_W U|} = \arg\max_{U} \frac{|U^T S_T U|}{|U^T S_W U|} = \arg\max_{U} \frac{|U^T S_B U|}{|U^T S_T U|} \tag{12}$$

where

$$S_B = \sum_{i=1}^{C} n_i\,(m_i - \mu)\,(m_i - \mu)^T \tag{13}$$

is the between-class scatter matrix,

$$S_W = \sum_{i=1}^{C} \sum_{x \in Ci} (x - m_i)\,(x - m_i)^T \tag{14}$$

is the within-class scatter matrix, and

$$S_T = \sum_{all\ x} (x - \mu)\,(x - \mu)^T = S_B + S_W \tag{15}$$

is the total scatter matrix, C is the total number of classes, n_i the sample number of class i, m_i the mean of class i, and μ the global mean. The S_T matrix is used to better keep discriminatory data during the update (T.-K. Kim, Stenger, Kittler, & Cipolla, 2011).

Pang, Ozawa, and Kasabov (2005) presented a method for deriving an updated discriminant eigenspace for classification when a burst of data containing new classes is being added to an initial discriminant eigenspace in the form of random chunks. The authors proposed an incremental linear discriminant analysis (ILDA) in two forms: a sequential ILDA and a chunk ILDA. Experiments compared the proposed ILDA against the traditional batch LDA in terms of discriminability, execution time, and memory use with additional consideration of increasing volume of data. According to the authors, the results showed that the proposed ILDA can effectively evolve a discriminant eigenspace over a fast and large data stream and extract features with superior discriminability in classification when compared with other methods.

Ghassabeh and Moghaddam (2007) introduced new adaptive learning algorithms to extract linear discriminant analysis features from multidimensional data in order to reduce the data dimension space. New adaptive algorithms for the computation of the square root of the inverse covariance matrix $\Sigma^{-1/2}$ were introduced. These algorithms preceded an adaptive principal component analysis algorithm in order to construct an adaptive multivariate multi-class LDA algorithm. The adaptive nature of the new optimal feature extraction method makes it appropriate for online pattern recognition applications. According to the authors, experimental results using synthetic, real multi-class, and multi-dimensional sequence of data demonstrated the effectiveness of the new adaptive feature extraction algorithm.

Ohta and Ozawa (2009) proposed an online feature extraction method called incremental recursive Fisher linear discriminant (IRFLD) based on recursive Fisher linear discriminant (RFLD), a batch learning algorithm proposed by Xiang, Fan, and Lee (2006). The number of discriminant vectors is limited to the number of classes minus one, due to the rank of the between-class scatter matrix in the conventional linear discriminant analysis (LDA). RFLD and the proposed IRFLD eliminate this limitation. In the proposed IRFLD, effective discriminant vectors are recursively searched for the complementary space of a conventional ILDA subspace. The authors also proposed a convergence criterion for the recursive computations defined by using the class separability of discriminant features projected on the complementary subspace. According to the authors, experiments results showed that the recognition accuracies of IRFLD outperform ILDA in terms of recognition accuracy. However, the advantage of IRFLD against ILDA depends on datasets.

T. K. Kim et al. (2011) proposed an incremental learning solution for LDA and its applications to object recognition problems, applying the sufficient spanning set approximation in three steps: updates for the total scatter matrix, the between-class scatter matrix, and the projected data matrix. The proposed online solution closely agreed with the batch solution in term of accuracy while significantly reducing the computational complexity, even when the number of classes as well as the set size is large. Moreover, the incremental LDA method is useful for semi-supervised online learning. Label propagation is done by integrating the incremental LDA into an expectation maximization framework.

Ozawa et al. (2010) stated that PCA and LDA transform inputs into linear features that are not always effective for classification purposes. The authors suggested for future research the extension of the incremental learning approach based on kernel PCA (Takeuchi et al., 2007) and kernel LDA (Cai, He, & Han, 2007). Another PCA limitation is the fact that it finds a small number of important factors while involving all original variables (Jenatton et al., 2010).

Chapter 3

Methodology

As Figure 1 shows, intrusion detection systems monitor network events for analysis to find computer security threats such as malware, spyware, and access violations.

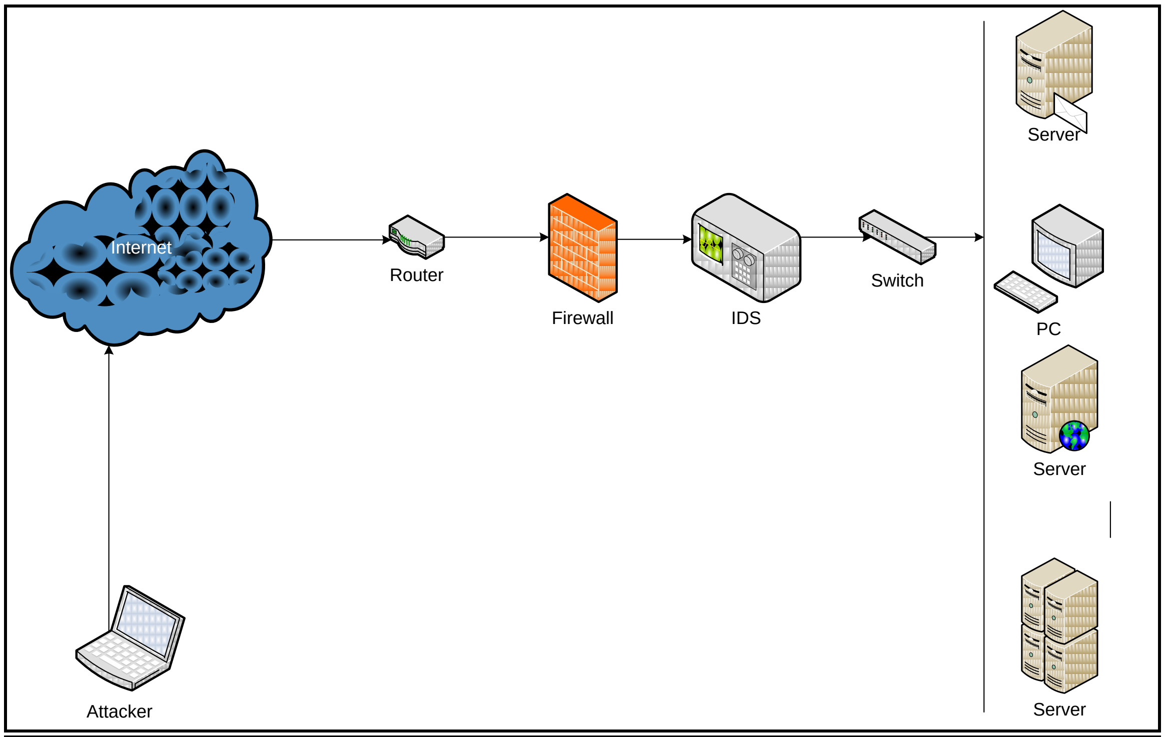

Figure 1. A network system with an IDS.

Overview of Research Methodology

Increasing numbers of applications generate large streams of data to be processed online and in real time for knowledge extraction. Processing such a large volume of data produced by a plurality of operational information systems renders their monitoring difficult using the old paradigm of storing data before analysis (Hebrail, 2008). Mining of continuous data streams of information to gather relevant attributes is computationally challenging (Domingos & Hulten, 2000). Noise and irrelevant attributes worsen the prediction accuracy.

This research aimed to find a method that dynamically extracts an optimal subset of data elements sufficient to obtain insights from massive data streams and take appropriate actions. This study proposed approaches, which when applied to the continuous network data streams efficiently, reduce the data dimensionality without negatively impacting its classification accuracy.

Principal component analysis is a popular dimensionality reduction technique used in a large variety of research domains. Nziga (2011) implemented PCA to considerably reduce the network dataset for intrusion detection. Nziga and Cannady (2012) combined PCA and mutual information to extract important features from the network dataset. This dissertation extends the original research presented by Nziga and Cannady, introducing a variation in the method of processing the data. Instead of storing the data off-line to be processed in a batch mode, this research aimed to process the data online and in real time for knowledge extraction.

This research was subdivided into five phases. First, a previously proposed incremental features extraction algorithm for facial recognition applications was implemented to process intrusion detection data streams. Using dimensionality reduction techniques on intrusion detection systems improves their performance. Then, two sparse principal component analysis techniques were presented. The next two phases introduced new incremental features extraction algorithms. Finally, we evaluated the concept drift impact using the newly proposed techniques.

Phase 1. Chunk Incremental Principal Component: Application to Network Data Streams

Incremental PCA-based methods have recently been proposed that allow adding new data and updating of PCA representation for face recognition (Pang et al., 2008; Yan & Liu, 2012).

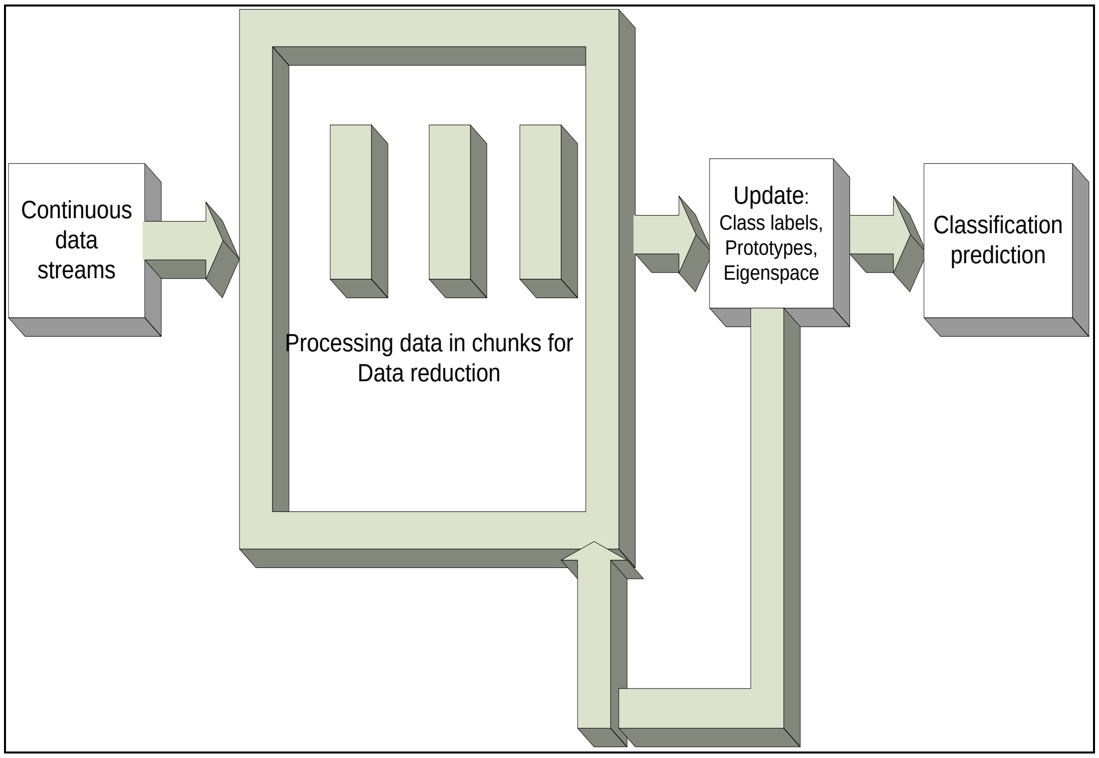
Continuous data streams
Processing data in chunks for Data reduction
Update: Class labels, Prototypes, Eigenspace
Classification prediction

Figure 2. Chunk incremental PCA approach.

Figure 2 shows Chunk IPCA that was implemented to evaluate network data streams. Chunk IPCA can overcome the problem with IPCA, processing a chunk of training samples at a time.

- A set of initial training samples D_0 is provided prior to the start of the incremental learning.
- An initial eigenspace model is obtained by applying PCA to $D_{0.}$
- The smallest dimensionality *k* of the eigenspace is determined, with an accumulation ratio larger than *θ*.
- The eigenspace model and L training samples become input to begin Chunk IPCA.
    - Eigen-axis selection to obtain augmented eigen-axes
    - Eigenspace rotation to obtain to obtain eigen-problem
    - Obtain the updated eigenvector matrix
    - Update the mean vector
    - Update the eigenspace model

The following metrics were gathered for evaluation:

- The impact of the initial data size on the classification accuracy, starting with 5% of the training dataset. Then increasing by 5% until no behavioral change was noticeable.
- The impact of the chunk size on the classification accuracy, staring with 100 samples of the training set. The chunk size increased by 100 at a time, until no output change was noticeable.

- The impact of the accumulation ratio factor θ, a positive value between [0, 1] on the classification accuracy,
- The CPU usage
- The processing time

The full chunk IPCA algorithm as proposed by Ozawa et al. (2008) follows.

```
**Input**:
    • Chunk IPCA algorithm
    • Initial training set $D_0 = \{(\mathbf{x}^{(i)}, z^{(i)}) \mid i = 1, \dots N\}$.
    • The number $P$ of prototypes,
    • The number $M$ of search points for threshold and search range $[\theta_1, \theta_M]$
Initialization:
    1) Call Training of initial Eigenspace to obtain the threshold $\theta$ and the initial
       eigenspace model $\Omega = (\bar{\mathbf{a}}, \mathbf{U}_k, \mathbf{\Lambda}_k, N)$ of $D_0$.
    2) $P' \leftarrow \min(P, N)$
    3) Select $P'$ training sample randomly from $D_0$ as reference vectors and put them
       into a set $\gamma$
loop // Prediction and Learning
Input: A new chunk of training samples
$D = \{(\mathbf{y}^{(i)}, z^{(i)}) \mid i = 1, \dots L\}$.
if $P' < P$ then
Select $\min(P - P', L)$ training samples randomly from $D$
put them into $\gamma$
end if
Call Update of Classifier to update the prototype $\gamma'$
Call Classification to predict the class labels $z(\mathbf{y}^{(i)})$ of queries $\mathbf{y}^{(i)}$ $(i = 1, \dots, L)$ in $D$
Apply chunk IPCA to $Y = \{\mathbf{y}^{(1)}, \dots, \mathbf{y}^{(L)}\}$
    1) Call Selection of Eigenaxes to obtain a matrix $\mathbf{H}_l$ of the $l$ augmented eigenaxes
    2) Solve an intermediate eigenproblem to obtain a rotation matrix $\mathbf{R}$ and an
       eigenvalue matrix $\mathbf{\Lambda}'_{k+l}$
    3) Update the mean vector $\bar{\mathbf{a}}'$ and the eigenvector matrix $\mathbf{U}'_{k+l}$
    Update the eigenspace model as follows:
        $\Omega = (\bar{\mathbf{a}}, \mathbf{U}_k, \mathbf{\Lambda}_k, N) \leftarrow \Omega' = (\bar{\mathbf{a}}', \mathbf{U}'_{k+l}, \mathbf{\Lambda}'_{k+l}, N+L)$
        Output: Prediction $z(\mathbf{y}^{(i)})$ $(i = 1, \dots, L)$
end loop
```

Algorithm 1. Chunk IPCA: Learning and Classification

Chunk IPCA (CIPCA) reduces the dimensionality of the input data stream. The data in the next chunk is used to construct the feature space. The above one-pass incremental algorithm can be better explained by the following learning algorithm steps (Ozawa et al., 2004):

Step 0:

(1) A small percentage of training samples $D_0 = \{(\mathbf{x}^{(i)}, z^{(i)})|i = 1, \dots N\}$.are used to construct the initial eigenspace $\Omega = (\bar{\mathbf{a}}, \mathbf{U}_k, \mathbf{\Lambda}_k, N)$.

(2) From the covariance matrix of the initial training samples, compute the eigenvector matrix **U** and the eigenvalue matrix **Λ**.

(3) The feature vectors $\bar{\mathbf{a}}$ is obtained by projecting all the initial training samples into the eigenspace.

(4) A classification algorithm is applied to feature vectors $\bar{\mathbf{a}}$ to generate the prototypes γ. In the CIPCA, the authors used *k*-Nearest Neighbors algorithm.

Step 1: CIPCA is applied to new chuck of *L* training samples, then update the current eigenspace $\Omega = (\bar{\mathbf{a}}, \mathbf{U}_k, \mathbf{\Lambda}_k, N)$, $D = \{(\mathbf{y}^{(i)}, z^{(i)})|i = 1, \dots L\}$

(1) Call Update of Classifier to update the prototype γ'.

(2) Call Classification to predict the class labels $z(\mathbf{y}^{(i)})$ of queries $\mathbf{y}^{(i)}$ ($i = 1, \dots ,L$) in D.

(3) Call Selection of Eigenaxes to obtain a matrix $\mathbf{H}_l$ of the *l* augmented eigenaxes. The accumulation ratio is updated and should be less than the given threshold value θ. The accumulation ratio specifies the amount of signal energy that should be retained to construct the feature spaces efficiently.

(4) Solve an intermediate eigenproblem to obtain a rotation matrix $\mathbf{R}$ and an eigenvalue matrix $\mathbf{\Lambda}$.

Step 2: Update the mean vector $\bar{\mathbf{a}}'$ and the eigenvector matrix $\mathbf{U}'_{k+l}$

Step 3: Update the eigenspace model as follows:
$\Omega = (\bar{\mathbf{a}}, \mathbf{U}_{\mathrm{k}}, \mathbf{\Lambda}_{\mathrm{k}}, N) \leftarrow \Omega' = (\bar{\mathbf{a}}', \mathbf{U}'_{k+l}, \mathbf{\Lambda}'_{k+l}, N+L)$

Step 4: **Output**: Prediction $z(\boldsymbol{y}^{(i)})$ $(i = 1, \dots, L)$

Step 5: Go back to Step 1.

PCA generally produces dense directions that are too complex to explain the dataset (He, Monteiro, & Park, 2011). PCA performs linear combinations of attributes to find the subset that increases variance in the dataset. However, the reduced data subset resulting from PCA is based on all original variables. The linear combination renders the resulting subset difficult to interpret and, for example, in IDS, impossible to identify a set of specific relevant attributes that need to be fed to intrusion detection systems. The interpretation of principal components is possible when they are composed from only a fraction of the original variables. Sparse principal component analysis (SPCA) achieves a reasonable trade-off between the conflicting goals of explaining all variables and using near orthogonal vectors constructed from as few features as possible (Grbovic et al., 2012). SPCA improves the relevance and interpretability of the component. SPCA also reveals the underlying structure of the dataset better than PCA (Grbovic et al., 2012). SPCA can be effectively stored in addition to simplifying the interpretation of the inherent structure and information associated with the dataset (He et al., 2011). Moreover, SPCA can be computed faster than PCA under certain conditions (Y. Zhang & El Ghaoui, 2011).

The novel methods in this research consisted of developing improved incremental feature extraction approaches called ISSPCA (incremental structure sparse principal component analysis) and IGSPCA (incremental global power for sparse principal component analysis), leveraging the structured sparse principal component analysis technique (Jenatton et al., 2010) and the generalized power method for sparse principal component analysis approach (Journee et al., 2010) respectively.

Phase 2. Sparse Principal Component Analysis

2-a. Structured Sparse Principal Component Analysis

Jenatton et al. (2010) proposed structured sparse PCA (SSPCA) to demonstrate the variance of the data by factors that are sparse and meet some constraints useful to model the data under consideration. Sparsity patterns of dictionary elements are constrained to a pre-specified set of shapes, encoding higher order by factors that are sparse while taking into account some data structural model constraints. Applied to face recognition database, SSPCA selects sparse convex areas corresponding to a more natural segment of faces (e.g., mouth, eyes). According to the authors, their approach led to a more interpretable decomposition of the data. The full SSPCA procedure is outlined below (Jenatton et al., 2010).

```
Input: Dictionary size r, data matrix $X$.
Initialization: Initialization of $U, V$ (possibly random).
    While (stopping criteria not reached)
        Update $(\prod^{G})_{G \in g}$ Closed-form solution.
        Update $U$ by BCD:
        for $t = 1$ to $T_u$, for $k = 1$ to $r$:
            $U^k \leftarrow \Pi_{\Omega u}(U^k + \|V^k\|_2^{-2} (XV^k - UVT^T V^k))$.
        Update $V$ by BCD:
        for $t = 1$ to $T_v$, for $k = 1$ to $r$:
```

$$V^k \leftarrow \mathrm{Diag}\,(\varsigma^k)\ \mathrm{Diag}\,(\|\,\mathrm{U}^{\mathrm{k}}\,\|_2^2\,\varsigma^{\mathrm{k}} + np\lambda 1)^{-1}(X^T U^k - VU^T U^k + \|\mathrm{U}^k\|_2^2 V^k).$$

Output: Decomposition *U*, *V*

Algorithm 2. Structured Sparse Principal Component Analysis

Structured sparse principal component analysis will be applied on network datasets in a batch mode to gather some benchmark metrics.

2-b. Generalized Power Method for Sparse Principal Component Analysis

Journee et al. (2010) proposed generalized power method for sparse PCA (GPSPCA) to extract sparse dominant principal components of dataset. The authors optimized their proposal by maximizing the convex function. According to the authors, their approach, which they tested on a set of random and gene expression, outperformed other algorithms, extracting richest and interpretable components. The authors proposed four formulations of the sparse PCA problem, namely: Single-unit sparse PCA via *l*1-Penalty, Single-unit sparse PCA via Cardinality Penalty, Block sparse PCA via *l*1-Penalty, and Block sparse PCA via Cardinality Penalty.

The formulation that worked best for the experiments in this research was the Single-unit sparse PCA via *l*1-Penalty. The full GPSPCA procedure based on this formulation follows (Journee et al., 2010).

```
Input: Data matrix $\mathbf{A} \in \mathbf{R}^{pxn}$
       Sparsity-controlling parameter $\gamma \geq 0$
       Initial iterate $x \in \mathbf{S}^{p}$
Output: A locally optimal sparsity pattern $P$
begin
       repeat
              $x \leftarrow \sum_{i=1}^{n} [|\mathbf{a}_i^T x| - \gamma] + \text{sign}(\mathbf{a}_i^T x)\, a_i$
              $x \leftarrow \frac{x}{||x||}$
       until a stopping criterion is satisfied

       Construct vector $P \in \{0,1\}^n$ such that $\begin{cases} p_i = 1 \ if \ |\mathbf{a}_i^T x| > \gamma \\ p_i = 0 \ otherwise \end{cases}$
end
```

Algorithm 3. Generalized power method for Sparse Principal Component Analysis

Phase 3. Incremental Structured Sparse Principal Component Analysis (ISSPCA)

As stated earlier, PCA reduces the dimensionality of a dataset using a vector space transformation. PCA performs linear combinations of variables to find the lower number of principal components that maximize variance in the data. While PCA successfully finds maximum variance components that are composed of all initial variables, it renders their interpretation difficult.

Sparse PCA, on the other hand, finds sparse vectors that are used as weights during linear combinations. Sparse PCA finds principal components as linear combinations of sparse vectors. The advantage of sparse PCA is interpretability as it extracts components with few non-zero coefficients. For example, in facial recognition, sparse PCA aims to extract local components (i.e., parts of the face).

SSPCA, as proposed by Jenatton et al. (2010), uses batch processing. Because the focus of this research was to extract features from data streams, an incremental approach of SSPCA, called ISSPCA, was implemented. An extensive evaluation was performed using network data streams as well as other datasets.

Incremental Sparse Principal Component Analysis

Similar to the CIPCA proposed by Ozawa et al. (2008), the incremental sparse PCA assumes that N training samples $\mathbf{a}^{(i)}$ are provided to a system initially; $\mathbf{a}^{(i)} \in \mathrm{R}^n$ $(i=1, \ldots, N)$.

A Sparse PCA method was then applied to the training samples to generate an eigenspace model:

$$\Omega = (\bar{\mathbf{a}}, \mathbf{U}_\mathrm{k}, \mathbf{\Lambda}_\mathrm{k}, N) \qquad (4)$$

Where: $\bar{\mathbf{a}}$ is a mean vector of $\mathbf{a}^{(i)}$ $(i=1, \ldots, N)$,

$\mathbf{U}_\mathrm{k}$ is an n x k matrix with column vector corresponding to sparse eigenvectors,

$\mathbf{\Lambda}_\mathrm{k} = \mathrm{diag}\{\lambda_1, \lambda_2, \ldots., \lambda_k\}$ is a k x k matrix with non-zero eigenvalues as diagonal elements. The value k, the number of eigen-axes spanning the eigenspace was determined as a function of accumulation ratio criterion. The system computed Ω, kept the information and threw away the entire training sample (Ozawa et al., 2010).

Then the $(N + 1)$th training sample was provided to the Incremental Sparse Principal Component Analysis (ISPCA):

$$\mathbf{a}^{(N+1)} = \mathbf{y} \in \mathrm{R}^n \qquad (5)$$

The new eigenspace model Ω' was defined as follows:

$$\Omega' = (\bar{\mathbf{a}}', \mathbf{U}'_{\mathrm{k}'}, \mathbf{\Lambda}'_{\mathrm{k}'}, N+1) \qquad (6)$$

The eigenspace dimensions k' is k or $k+1$ depending on whether or not **y** includes certain energy in the complementary eigenspace. The eigen-axes were rotated to adapt to the variation in the data distribution in three steps: mean vector update, eigenspace augmentation, and rotation of eigen-axes. (Equations 7, 8, 9, 10 and 11)

The full chunk ISSPCA algorithm based on the chunk IPCA proposed by Ozawa et al. (2008) follows:

```
Input:
    • Chunk ISSPCA algorithm
    • Initial training set $D_0 = \{(\mathbf{x}^{(i)}, z^{(i)}) | i = 1, \dots N\}$.
    • The number $P$ of prototypes,
    • The number $M$ of search points for threshold and search range $[\theta_1, \theta_M]$
Initialization:
    1) Call Training of initial Eigenspace to obtain the threshold $\theta$ and the initial
       eigenspace model $\Omega = (\bar{\mathbf{a}}, \mathbf{U}_k, \mathbf{\Lambda}_k, N)$ of $D_0$, using SSPCA
    2) $P' \leftarrow \min(P, N)$
    3) Select $P'$ training sample randomly from $D_0$ as reference vectors and put them
       into a set $\gamma$
loop // Prediction and Learning
Input: A new chunk of training samples
$D = \{(\mathbf{y}^{(i)}, z^{(i)}) | i = 1, \dots L\}$.
if $P' < P$ then
Select min ($P - P', L$) training samples randomly from $D$
put them into $\gamma$
end if
Call Update of Classifier to update the prototype $\gamma'$
Call Classification to predict the class labels $z(\mathbf{y}^{(i)})$ of queries $\mathbf{y}^{(i)}$ ($i = 1, \dots, L$) in $D$
Apply chunk ISSPCA to Y = { $\mathbf{y}^{(1)}, \dots, \mathbf{y}^{(L)}$ }
    4) Call Selection of Eigenaxes to obtain a matrix $\mathbf{H}_l$ of the $l$ augmented eigenaxes
    5) Solve an intermediate eigenproblem to obtain a rotation matrix $\mathbf{R}$ and an
       eigenvalue matrix $\mathbf{\Lambda}'_{k+l}$
    6) Update the mean vector $\bar{\mathbf{a}}'$ and the eigenvector matrix $\mathbf{U}'_{k+l}$
    Update the eigenspace model as follows:
        $\Omega = (\bar{\mathbf{a}}, \mathbf{U}_k, \mathbf{\Lambda}_k, N) \leftarrow \Omega' = (\bar{\mathbf{a}}', \mathbf{U}'_{k+l}, \mathbf{\Lambda}'_{k+l}, N+L)$
        Output: Prediction $z(\mathbf{y}^{(i)})$ ($i = 1, \dots, L$)
end loop
```

Algorithm 4. Chunk ISSPCA: Learning and Classification

Two flavors of Incremental Structured Sparse Principal Component Analysis were evaluated in this research: ISSPCA-L1 and ISSPCA-Lq. ISSPCA-L1 is a convex model based on the standard L1 norm regularization. This is also called exact regularization. ISSPCA-Lq is a non-convex model with Lq quasi-norm regularization q in (0,1). In this experiment, q = 0.5; the following metrics were gathered for ISSPCA-L1 and ISSPCA-Lq evaluation:

- The impact of the initial data size on the classification accuracy. Starting with 5% of the training set, increased the initial data size by 5% until no change was noticeable.
- The impact of the chunk size on the classification accuracy. Starting with 100 samples, increased the chunk size by 100 until no change was perceivable in the output.
- The impact of the accumulation ratio factor θ, a positive value between [0, 1] on the classification accuracy.
- Minimum subset of original attributes. PCA finds principal components that cannot be interpreted (Nziga & Cannady, 2012). Sparse PCA finds principal components as linear combinations of sparse vectors, extracting variables with few non-zero coefficients. This study identified the first few attributes sufficient to represent the dataset.
- The CPU usage.
- The processing time.

Phase 4. Incremental Generalized Power Method Sparse Principal Component Analysis (IGSPCA)

The generalized power method sparse PCA proposed by Journee et al. (2010) employs batch processing. According to the authors, the algorithm displays great convergence properties and outperforms existing batch techniques in quality of the reduced dataset and the computation time. This research implemented an incremental approach of GPSPCA, called IGSPCA with the goal to extract features from data streams. Network data streams performed an extensive evaluation.

Similar to the CIPCA proposed by Ozawa et al. (2008), the incremental sparse PCA assumed that N training samples $\mathbf{a}^{(i)}$ are provided to a system initially: $\mathbf{a}^{(i)} \in \mathrm{R}^{n}$ $(i=1, \ldots, N)$.

A Sparse PCA method was then applied to the training samples to generate an eigenspace model:

$$\Omega = (\bar{\mathbf{a}}, \mathbf{U}_{\mathrm{k}}, \mathbf{\Lambda}_{\mathrm{k}}, N) \qquad (4)$$

Where: $\bar{\mathbf{a}}$ is a mean vector of $\mathbf{a}^{(i)}$ $(i=1, \ldots, N)$,

$\mathbf{U}_{\mathrm{k}}$ is an n x k matrix with column vector corresponding to sparse eigenvectors,

$\mathbf{\Lambda}_{\mathrm{k}} = \mathrm{diag}\{\lambda_{1}, \lambda_{2}, \ldots., \lambda_{k}\}$ is a k x k matrix with non-zero eigenvalues as diagonal elements. The value k, the number of eigen-axes spanning the eigenspace was determined as a function of accumulation ratio criterion. The system computed Ω, kept the information, and threw away the entire training sample (Ozawa et al., 2010).

Then the $(N + 1)$th training sample was provided to the Incremental Sparse Principal Component Analysis (ISPCA):

$$\mathbf{a}^{(N+1)} = \mathbf{y} \in \mathrm{R}^n \qquad (5)$$

The new eigenspace model Ω' was defined as follows:

$$\Omega' = (\bar{\mathbf{a}}', \mathbf{U}'_{k'}, \mathbf{\Lambda}'_{k'}, N+1) \qquad (6)$$

The eigenspace dimensions k' was k or $k+1$ depending on whether or not **y** included certain energy in the complementary eigenspace. The eigen-axes were rotated to adapt to the variation in the data distribution in three steps: mean vector update, eigenspace augmentation, and rotation of eigen-axes (Equations 7, 8, 9, 10 and 11).

The full chunk IGSPCA algorithm based on the chunk IPCA proposed by Ozawa et al. (2008) follows:

```
Input:
  • Chunk IGSPCA algorithm
  • Initial training set $D_0 = \{(\mathbf{x}^{(i)}, z^{(i)}) | i = 1, \dots N\}$.
  • The number $P$ of prototypes,
  • The number $M$ of search points for threshold and search range $[\theta_1, \theta_M]$
Initialization:
  1) Call Training of initial Eigenspace to obtain the threshold $\theta$ and the initial
     eigenspace model $\Omega = (\bar{\mathbf{a}}, \mathbf{U}_k, \mathbf{\Lambda}_k, N)$ of $D_0$, using GPSPCA
  2) $P' \leftarrow \min(P, N)$
  3) Select $P'$ training sample randomly from $D_0$ as reference vectors and put them
     into a set $\gamma$
loop // Prediction and Learning
Input: A new chunk of training samples
$D = \{(\mathbf{y}^{(i)}, z^{(i)}) | i = 1, \dots L\}$.
if $P' < P$ then
Select min $(P - P', L)$ training samples randomly from $D$
put them into $\gamma$
end if
Call Update of Classifier to update the prototype $\gamma'$
Call Classification to predict the class labels $z(\mathbf{y}^{(i)})$ of queries $\mathbf{y}^{(i)}$ $(i = 1, \dots, L)$ in $D$
Apply chunk IGSPCA to $Y = \{ \mathbf{y}^{(1)}, \dots, \mathbf{y}^{(L)} \}$
```

```
        4) Call *Selection of Eigenaxes* to obtain a matrix $\mathbf{H}_l$ of the $l$ augmented eigenaxes
        5) Solve an intermediate eigenproblem to obtain a rotation matrix $\mathbf{R}$ and an
           eigenvalue matrix $\mathbf{\Lambda}'_{k+l}$
        6) Update the mean vector $\bar{\mathbf{a}}'$ and the eigenvector matrix $\mathbf{U}'_{k+l}$
    Update the eigenspace model as follows:
        $\Omega = (\bar{\mathbf{a}}, \mathbf{U}_k, \mathbf{\Lambda}_k, N) \leftarrow \Omega' = (\bar{\mathbf{a}}', \mathbf{U}'_{k+l}, \mathbf{\Lambda}'_{k+l}, N+L)$
        Output: Prediction $z(\mathbf{y}^{(i)})$ $(i = 1, \ldots, L)$
end loop
```

Algorithm 5. Chunk IGSPCA: Learning and Classification

The following metrics were gathered for IGSPCA evaluation:

- The impact of the initial data size on the classification accuracy. Starting with 5% of the training set, increased the initial data size by 5% until no change was noticeable.
- The impact of the chunk size on the classification accuracy. Starting with 100 samples, increased the chunk size by 100 until no change was perceivable in the output.
- The impact of the accumulation ratio factor θ, a positive value between [0, 1] on the classification accuracy.
- Minimum subset of original attributes. PCA found principal components that cannot be interpreted (Nziga & Cannady, 2012). Sparse PCA found principal components as linear combinations of sparse vectors, extracting variables with few non-zero coefficients. This study identified the first few attributes sufficient to represent the dataset.
- The CPU usage.
- The processing time.

Similar to Chunk IPCA, the Incremental Sparse PCA algorithms presented in this dissertation reduced the dimensionality of the input data stream, using the data in the next chunk to construct the feature space. The implemented one-pass incremental sparse algorithms are better explained by the following learning steps:

Step 0:

(1) A small percentage of training samples $D_0 = \{(\mathbf{x}^{(i)}, z^{(i)}) | i = 1, \ldots N\}$.Were used to construct the initial eigenspace $\Omega = (\bar{\mathbf{a}}, \mathbf{U}_k, \mathbf{\Lambda}_k, N)$.

(2) From the covariance matrix of the initial training samples, computed the eigenvector matrix $\mathbf{U}$ and the eigenvalue matrix $\mathbf{\Lambda}$. This computation used one of the Sparse PCA algorithms presented above. This framework can be expanded to use other sparse PCA algorithms in the future for comparative studies and improvements.

(3) The feature vectors $\bar{\mathbf{a}}$ was obtained by projecting all the initial training samples into the eigenspace.

(4) A classification algorithm was applied to feature vectors $\bar{\mathbf{a}}$ to generate the prototypes γ. While in the CIPCA, the authors (Ozawa et al., 2010) used *k*-Nearest Neighbors algorithm; in this dissertation, the researcher used the decision tree classification. This framework can be expanded to use other classifiers (SVM, BayesNet, NaiveBayes, etc …) for comparative studies and improvements in the future.

Step 1: Incremental Sparse PCA is applied to new chuck of L training samples, then update the current eigenspace $\Omega = (\bar{\mathbf{a}}, \mathbf{U}_k, \mathbf{\Lambda}_k, N)$, $D = \{(\mathbf{y}^{(i)}, z^{(i)}) | i = 1, \ldots L\}$.

1. Call Update of Classifier to update the prototype *γ'*.
2. Call Classification to predict the class labels $z(\mathbf{y}^{(i)})$ of queries $\mathbf{y}^{(i)}$ $(i = 1, \ldots, L)$ in D.
3. Call Selection of Eigenaxes to obtain a matrix $\mathbf{H}_l$ of the l augmented eigenaxes. The accumulation ratio is updated and should be less than the given threshold value θ. The accumulation ratio specifies the amount of signal energy that should be retained to construct the feature spaces efficiently.
4. Solve an intermediate eigenproblem to obtain a rotation matrix **R** and an eigenvalue matrix **Λ.**

Step 2: Update the mean vector $\bar{\mathbf{a}}'$and the eigenvector matrix $\mathbf{U}'_{k+l}$

Step 3: Update the eigenspace model as follows:
$$\Omega = (\bar{\mathbf{a}}, \mathbf{U}_k, \mathbf{\Lambda}_k, N) \leftarrow \Omega' = (\bar{\mathbf{a}}', \mathbf{U}'_{k+l}, \mathbf{\Lambda}'_{k+l}, N+L)$$

Step 4: **Output**: Prediction $z(\mathbf{y}^{(i)})$ $(i = 1, \ldots, L)$

Step 5: Go back to Step 1.

Another major difference between CIPCA and the new IGSPCA is in the computation of the eigenvalue matrix. In CIPCA, the eigenvalue matrix was computed using the Matlab function `eig(A)` that returns a vector of the eigenvalues of matrix A. In IGSPCA, the eigenvalue matrix was computed using the Matlab function `eigs(A,k)` and that returns the `k` largest magnitude eigenvalues.

Phase 5. Incremental Sparse Principal Component Analysis: Impact of Concept Drift and Concept Shift

Concept drift and concept shift are respectively gradual and quick changes in continuous streams of data. Both concept drift and concept shift cause variables in data streams to change with the potential to degrade the predictability accuracy in the long run. It is important to validate the effectiveness of ISSPCA and IGSPCA, the newly implemented incremental feature extraction approaches, to handle concept drift and concept shift.

The following two datasets were generated with gradual concept drift: WaveForm and WaveFormNoise. The WaveForm dataset is constituted of three classes of waves, 21 attributes, and 5,000 instances. The WaveFormNoise dataset is constituted of three classes of waves 40 attributes, and 5,000 instances.

Concept shift can be simulated on a dataset by randomly selecting some attributes and changing their values in a consistent manner (Morshedlou & Barforoush, 2009). The researcher generated a new dataset with concept shift by shuffling the values of the first and the last attributes of the WafeFormNoise dataset, for all instances while keeping the class label intact. The researcher then randomly split the new dataset into two samples; 1,000 records formed the testing sample, and 4,000 records formed the training subset.

The following metrics were gathered for ISSPCA and IGSPCA evaluation:

- The impact of the initial data size on the classification accuracy. Starting with 5% of the training set, increased the initial data size by 5% until no change was noticeable.

- The impact of the chunk size on the classification accuracy. Starting with 100 samples, increased the chunk size by 100 until no change was perceivable in the output.
- The impact of the accumulation ratio factor θ, a positive value between [0, 1] on the classification accuracy.
- The CPU usage.
- The processing time.

The performance of ISSPCA and IGSPCA was compared to that of chunk IPCA (Ozawa et al., 2008). Experimental results demonstrated improvements using the new approaches with respect to dimensionality reduction, processing speed, resource requirements, and classification rate. Test results such as classification accuracy, processing time, memory requirements were thoroughly analyzed to show the best performing streaming-based feature extraction technique that helps keep intruders out of the network.

Data Sets

Experiments described in the previous section were conducted on a continuous data streams generated from the network data set kddcup.data_10_percent_corrected (DARPA KDD Cup '99 data set), a widely used data set for network intrusion detection systems. It is of size ~75 MB and composed of 494,021 connection records or vectors, 41 features or attributes, and one class label. It represents a 10% subset of the full KDD dataset. The KDD Cup '99 data set was built based on the data recorded in the DARPA’99 Intrusion Detection System evaluation program (Lippmann, Haines, Fried, Korba, & Das, 2000). The full data is almost one gigabyte of uncompressed data and contains about 5 million connection vectors of 41 attributes each. Each record is labeled as either normal or as an attack with a specific attack type. There are in total four main attack categories: DoS (Denial of Service), R2L (unauthorized access from a remote machine), U2R (unauthorized access to local super-user with root privileges), and probing (surveillance, port scanning).

In addition to the KDD Cup '99 data set, the Poker Hand dataset was considered for a complete evaluation of the proposed incremental feature extraction approach. The Poker-Hand dataset has 1,000,000 records of 11 attributes each. Ten of the attributes are predictive and one class attribute describes the PokerHand. Each instance corresponds to five playing cards drawn from a desk of 52 cards.

The following two datasets, which are publicly available, were also used to assess how the proposed approaches handled concept drift.

1. Waveform. The WaveForm dataset is constituted of three classes of waves and 21 attributes. Attributes are continuous values between 0 and 6. There are 5,000 instances. The generator is used to acquire instances with gradual concept drift (Breiman, Friedman, Olshen, & Stone, 1984). The artificial Waveform dataset was used by Gama, Rocha, and Medas (2003).
2. WaveFormNoise. The WaveFormNoise dataset is constituted of 3 classes of waves and 40 attributes (Breiman et al., 1984). Attributes are continuous values between 0 and 6. The later 19 attributes are all noise attributes with mean = 0 and variance = 1. There are 5,000 instances.

A new dataset was simulated by shuffling the values of the first and the last attributes of the WafeFormNoise dataset to assess how the proposed approaches handled concept shift.

Experimental Setup

The performance of ISSPCA and IGSPCA was compared to that of chunk IPCA (Ozawa et al., 2008). The chunk IPCA method updated the eigenspace model with a chunk of training samples in a single operation. For each network dataset, the chunk size started at 100 (L = 100). This value increased by 100 to evaluate the chunk size impact on the feature extraction algorithm and the classification accuracy. Datasets were randomly divided into two subsets: training samples and test samples.

An initial eigenspace was constructed by applying a percentage p (i.e. p = 5%) of the training samples to the conventional PCA. The remaining 100 – p (i.e., 95%) of training samples were sequentially provided to the incremental approach under consideration. The impact of the initial eigenspace was evaluated by increasing the rate of the initial dataset by five until classification accuracy showed no change. For each experiment, learning time was recorded using a Matlab function `cputime`.

Measures

The following measures were collected for each incremental feature extraction approach under consideration in this research study: computational cost/time, the size of extracted features, and the impact of the data chunk on classification accuracy results with continuous network intrusion data streams. The impact of the data streams speed was evaluated as well (data moving at constant versus variable speeds). The goal of the sparse PCA feature extraction techniques was to improve benchmark measurements obtained using the chunk IPCA (Ozawa et al., 2008). In addition, the impact of concept drift and concept shift was evaluated for both algorithms. For each set of results gathered in tables, graphs were generated as well.

Resources

The resources necessary to conduct this research included the following:

1. Dimensionality reduction algorithms (Incremental PCA, Sparse PCA).
2. Various datasets (See data sets section above for details).
3. Matlab: a high-level language and interactive environment for numerical computation, programming, and visualization.
4. Java Software Development Environment.
5. Weka Libraries (classification libraries).
6. HP Laptop running Windows 7, Intel Duo CPU 2 GHz, T6400, 4 GB RAM.

Chapter 4

Results, Data Analysis, and Summary

Introduction

This dissertation introduces two new incremental sparse PCA methods to extract features from large data streams. PCA reduces the dimensionality of large datasets by finding linear combinations of all variables corresponding to maximal data variance. Sparse PCA finds sets of sparse vectors that are used as weights in performing linear combinations with maximal data variance. Sparse PCA retrieves the small number of features capable of capturing the most of the variance. For example, Sparse PCA in gene expression helps find principal components consisting only of very important genes, rending their interpretation easier.

First, Chunk Incremental PCA algorithm (Pang et al., 2008; Yan & Liu, 2012) was implemented and evaluated as a baseline for the approaches proposed in this dissertation. Two new incremental methods based on recent Sparse PCA techniques were then proposed in this research. The new features extraction techniques are called Incremental Structured Sparse PCA (ISSPCA) and Incremental Global Power for Sparse PCA (IGSPCA), based on the structured PCA (Jenatton et al., 2010) and generalized power method for sparse PCA (Journee et al., 2010) respectively. Two flavors of the ISSPCA were evaluated, namely ISSPCA-L1 and ISSPCA-Lq. ISSPCA-L1 is a convex model based on the standard L1 normalization, whereas ISSPCA-Lq is non-convex with Lq quasi-norm regularization q=0.5.

Four datasets (Table 1) were used to evaluate the performance of the proposed incremental feature extraction algorithms as well as CIPCA.

Table 1. Datasets Used to Evaluate Algorithms

Datasets	Training data size	Testing data size	Classes	Attributes
DARPA KDD Cup	395,218	98,804	23	41
Poker-Hand	25,000	20,000	10	10
WaveForm	4,000	1,000	3	21
WaveFormNoise	4,000	1,000	3	40

For each algorithm and each dataset, the following metrics were collected:

- The impact of the chunk size on the quality of the extracted data subset and its classification accuracy.
- The CPU time: This is the amount of time (in seconds) actually used for executing program instructions. CPU time includes neither the Input and Output time nor other idle durations.
- The dimensionality of the reduced dataset.
- The smallest amount of features capable of capturing the most variance.

The following sections provided a comparison of the results obtained in this research study followed by detailed interpretations. All four datasets used for this research study were divided into training and testing samples. Each was split randomly as follows: 20% for training and 80% for testing.

Intrusion Detection Dataset

DARPA KDD Cup '99 is a widely used data set for network intrusion detection systems. It has 494,021 connection records or vectors, 41 features or attributes, and one class label. The KDD Cup '99 data set was built by recording data in the DARPA’99 Intrusion Detection System evaluation program (Lippmann et al., 2000).

Impact of Data Chunk Size

Chunk IPCA or CIPCA: Similar to Ozawa et al. (2008), an initial eigenspace was generated based on the value 0.1% of initial data subset from the training samples. Then, the remaining training sample was fed to the algorithm in sequence for learning. To evaluate the influence of chunk size on the feature extraction and the classification accuracy, the experiment started with 100 samples (L = 100), then increased the chunk size by 100 until the number reached 1,400. The experiment is considered a one-pass because training samples were evaluated once and there was no overlap between chunks of training data. Five trials were executed and the average performance data points were recorded.

Table 2 and Figure 3 show the dimensionality of the reduced datasets at the completion of the learning for various values of training sample chunk sizes. We can observe that for both CIPCA and IGSPCA, the dimension of the extracted dataset is not affected by the chunk size. In addition, we realized that the dimensionality of the reduced dataset is the same for both CIPCA and IGSPCA. Conversely, the dimensionality of the reduced dataset using ISSPCA-L1 and ISSPCA-Lq bested that of the other two algorithms when chunk size was greater than 1,100.

Table 2. Intrusion Detection Dataset: Impact of Chunk Size on Dimensionality, $\theta = 0.9$

Chunk Size	Algorithm (Reduced dimension)			
	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
100	17.8	17.6	23	23
200	17.6	17.6	26	22.5
300	17.6	17.8	21	19.5
400	17.6	17.6	28.5	21.5
500	17.6	17.8	19.6	19
600	17.6	17.6	19.6	17.6
700	17.6	17.6	23	17.25
800	17.6	17.6	19	18.75
900	17.6	17.8	19.75	18.4
1,000	17.6	18	17	18.4
1,100	17.6	17.6	16.4	15.8
1,200	17.6	17.8	15.2	16.2
1,300	17.6	17.6	16.2	16.4
1,400	17.6	17.6	15.6	15.4

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

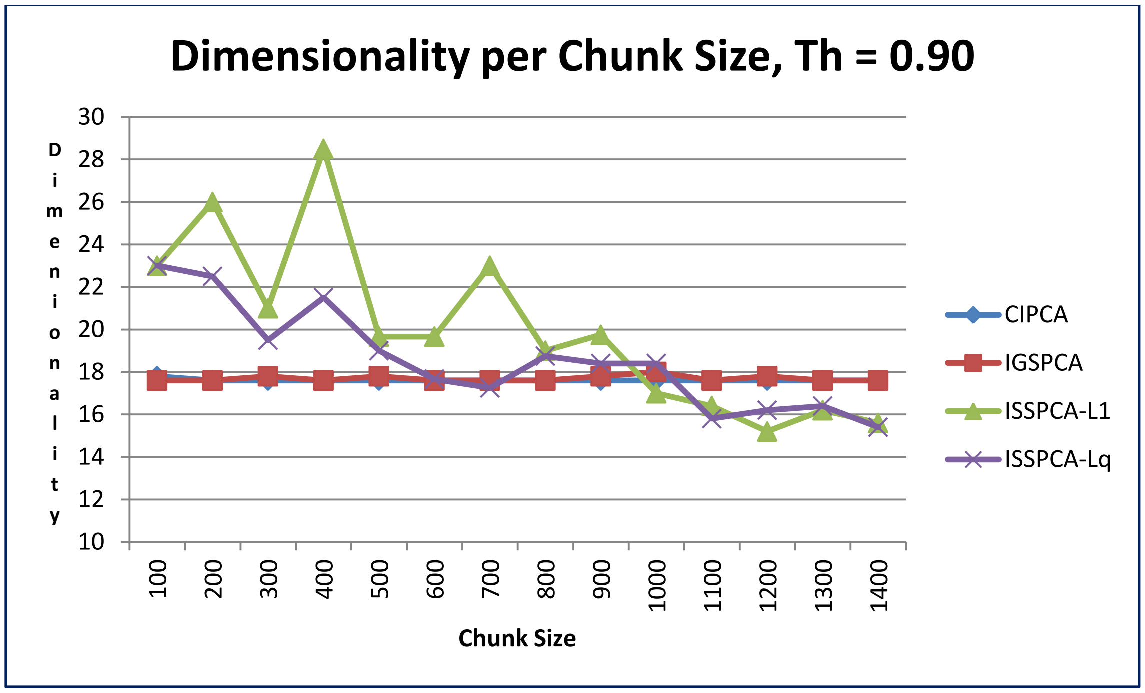

Figure 3. Intrusion detection dataset: Impact of chunk size on dimensionality, $\theta = 0.9$.

Table 3 and Figure 4 show the classification accuracy of the reduced datasets at the completion of the learning for various values of training sample chunk sizes.

Table 3. Intrusion Detection Dataset: Impact of Chunk Size on Classification Accuracy, θ = 0.9

Chunk Size	Algorithm (%)			
	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
100	98.67	96.10	99.2	99.46
200	98.68	93.98	99.29	99.17
300	98.69	99.42	99.20	99.12
400	98.69	99.39	99.17	99.13
500	98.62	99.49	99.16	99.28
600	98.71	99.46	99.16	99.24
700	98.69	98.63	99.31	99.24
800	98.63	99.38	99.34	99.14
900	98.65	99.46	99.37	99.21
1,000	98.79	99.51	99.17	98.95
1,100	98.72	99.36	99.27	99.28
1,200	98.76	90.85	99.20	99.31
1,300	98.66	98.49	99.08	99.25
1,400	98.69	99.14	99.26	99.35

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

For θ = 0.90, the classification accuracy of ISSPCA-L1 and ISSPCA-Lq is greater than that of CIPCA. The classification accuracy of IGSPCA is greater than that of CIPCA for chunk sizes of [300… 600] and [800… 1100]. For L = 200 and L = 1 and L=200, CIPCA yielded a better classification accuracy than IGSPCA.

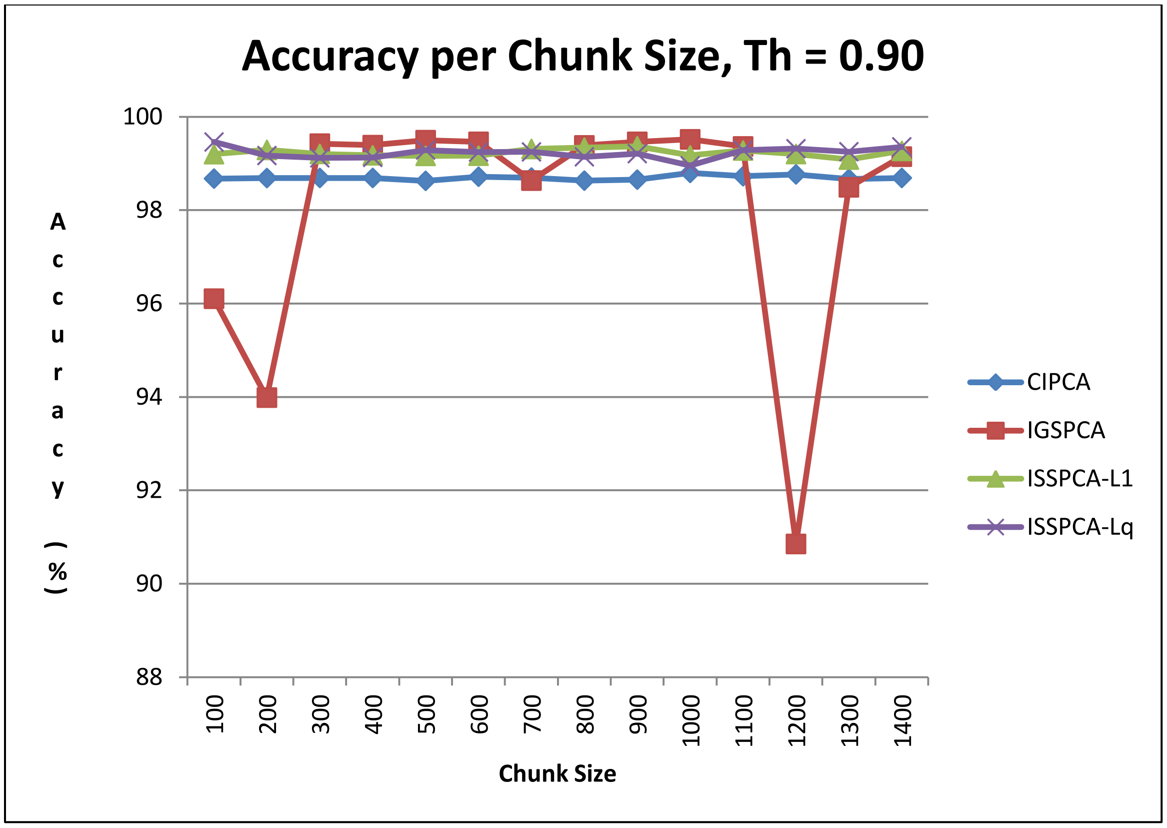

Figure 4. Intrusion detection dataset: Impact of chunk size on classification accuracy, θ = 0.9.

Table 4 and Figure 5 respectively show the learning time to generate the reduced datasets at the completion of the learning for various values of training sample chunk sizes.

Table 4. Intrusion Detection Dataset: Impact of Chunk Size on Learning Time, θ = 0.9

Chunk Size	Algorithm (Seconds)			
	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
100	22.59	40.54	1,293.0	1,208.9
200	27.07	38.44	729.88	621.69
300	22.9	27.7	422.0	358.0
400	21.72	25.93	444.6	278.1
500	21.09	23.83	267.7	169.9

Table 4. Intrusion Detection Dataset: Impact of Chunk Size on Learning Time, $\theta = 0.9$ (continued)

Chunk Size	Algorithm (Seconds)			
	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
600	22.90	23.86	160.1	188.0
700	24.19	29.12	200.8	141.6
800	26.28	22.34	119.8	115.1
900	28.67	28.18	124.8	115.3
1,000	32.5	35.1	92.65	99.94
1,100	34.2	37.1	82.3	75.1
1,200	36.14	38.65	65.62	79.69
1,300	44.2	37.1	67.1	71.3
1,400	44.0	41.0	63.7	58.1

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

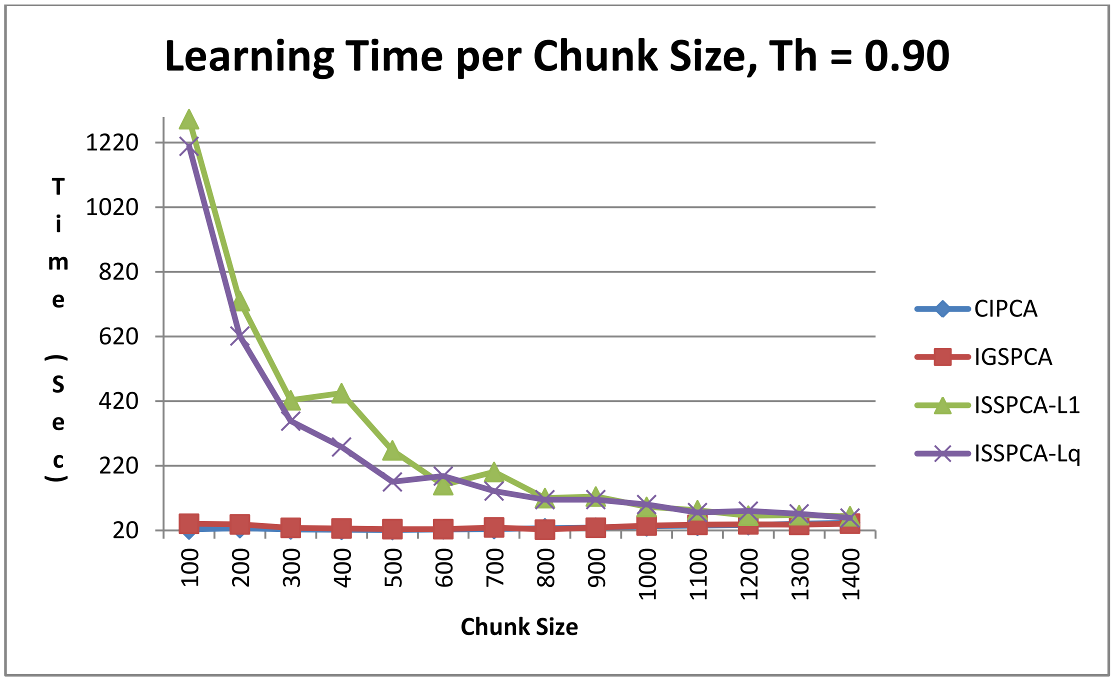

Figure 5. Intrusion detection dataset: Impact of chunk size on learning time, $\theta = 0.9$.

The learning time required for either ISSPCA-L1 or ISSPCA-Lq is extremely high, especially for lower chunk size values. The condition improves for higher chunk size values. On the other hand, the learning time required by CIPCA and IGSPCA are mostly identical all along. PCA reduces matrices dimensionality by finding principal eigenvectors with the largest eigenvalues.

In CIPCA, as the new data chunk arrives, the dimensionality of the newly reduced data matrix or eigenspace is a function of the accumulation ratio threshold. Selecting a least optimum threshold θ would cause the eigenaxes to augment frequently. In order to evaluate the impact of the impact of the accumulation ratio factor θ on the reduced dataset as well as its classification accuracy, experiments used various positive values between [0, 1] with each of the following algorithms: CIPCA, IGSPCA, ISSPCA-L1 and ISSPCA-Lq.

Table 5. Intrusion Detection Dataset: Impact of Accumulation Ratio on Dimensionality, L = 500

Accumulation Ration Threshold	Algorithm (Reduced dimension)			
	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
0.1	17.6	36.2	4.3	3.3
0.2	17.6	41.0	6.6	2.67
0.3	17.6	24.8	6.3	6.0
0.4	4.6	14.2	8.7	8.3
0.5	5.6	7.4	9.3	9.3
0.6	7.4	9.4	14.3	9.3
0.7	10.4	9.4	13.67	11.0
0.8	13.6	12.4	19.0	16.0
0.9	17.6	17.8	19.67	19.0

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

For data streams with a chunk size of L = 500, the dimensionality of the reduced dataset is minimal for ISSPCA-L1 and ISSPCA-Lq when θ is in [0, 0.5].
On the other hand, CIPCA and IGSPCA obtained their smaller dimensionality for θ in [0.4, 0.5] and θ in [0.5, 0.7], respectively. Note that the original network intrusion detection dataset has 41 attributes. The results from this experiment are available in Table 5 and Figure 6.

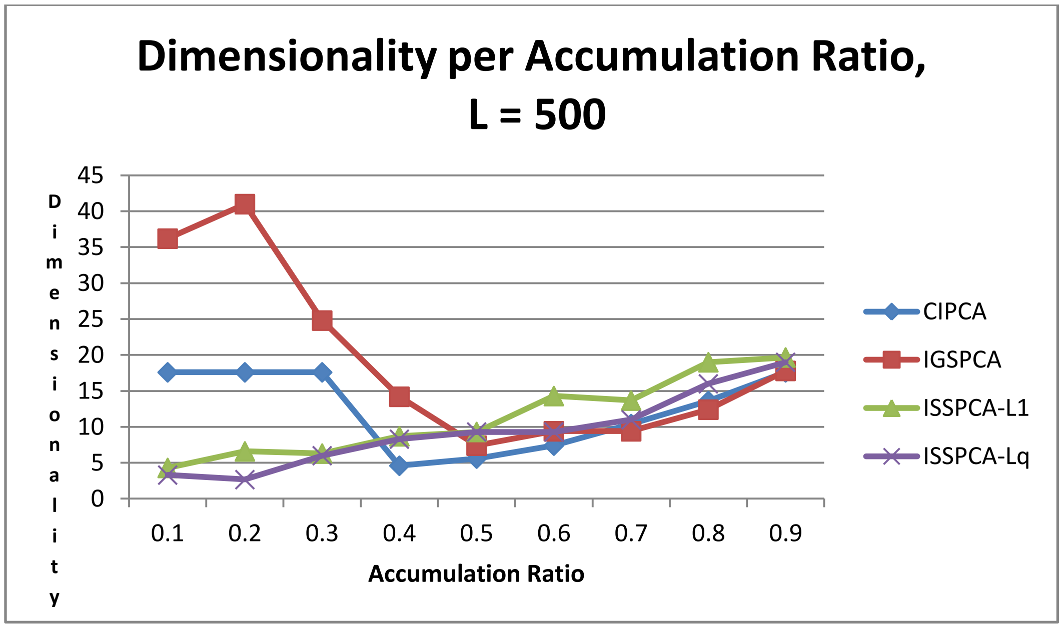

Figure 6. Intrusion detection dataset: Impact of accumulation ratio on dimensionality, L = 500.

Next, the experiments featured evaluation of classification accuracy impact on the reduced dataset using various positive values of accumulation ratio factor θ between [0, 1] for each of the following algorithms: CIPCA, IGSPCA, ISSPCA-L1, and ISSPCA-Lq.

Table 6. Intrusion Detection Dataset: Impact of Accumulation Ratio on Classification Accuracy, L = 500

Accumulation Ration Threshold	Algorithm (%)			
	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
0.1	98.69	97.85	96.16	98.31
0.2	98.668	97.1	81.23	98.95
0.3	98.766	99.82	98.92	79.31
0.4	98.81	99.15	98.99	99.2
0.5	98.67	99.06	99.32	99.25
0.6	99.03	98.85	99.13	99.06
0.7	99.19	99.11	99.13	99.19
0.8	99.25	99.47	99.37	99.3
0.9	98.63	99.5	99.16	99.29

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

Table 6 and Figure 7 show little variation among CIPCA, IGSPCA, ISSPCA-L1, and ISSPCA-Lq for θ between [0.4, 0.9]. IGSPCA achieved the maximum classification accuracy rate of 99.82% for $\theta = 0.3$.

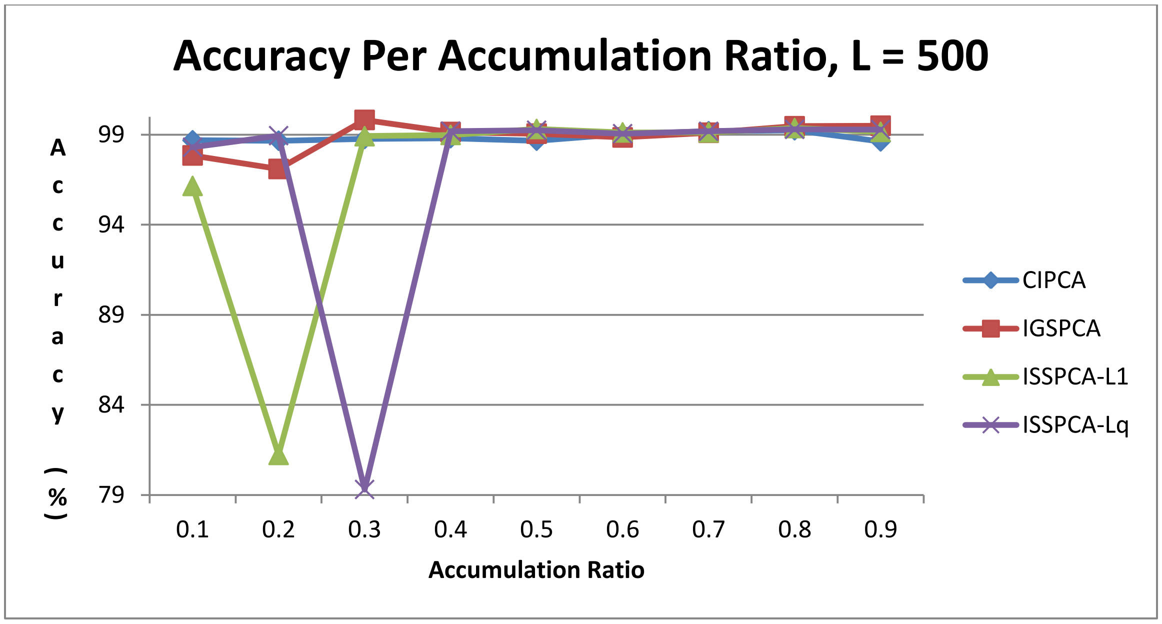

Figure 7. Intrusion detection dataset: Impact of accumulation ratio on classification accuracy, L = 500.

Table 7. Intrusion Detection Dataset: Impact of Accumulation Ratio on Learning Time, L = 500

Accumulation Ration Threshold	Algorithm (Seconds)			
	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
0.1	44.012	33.35	80.2	64.23
0.2	40.248	36.09	77.29	67.47
0.3	36.142	27.95	77.45	82.74
0.4	16.79	25.24	88.21	86.46
0.5	17.13	22.14	92.75	97.53
0.6	17.62	24.93	150.47	96.53
0.7	19.59	26.55	132.15	107.23
0.8	20.13	25.14	235.24	170.08
0.9	23.87	30.46	244.15	195.58

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

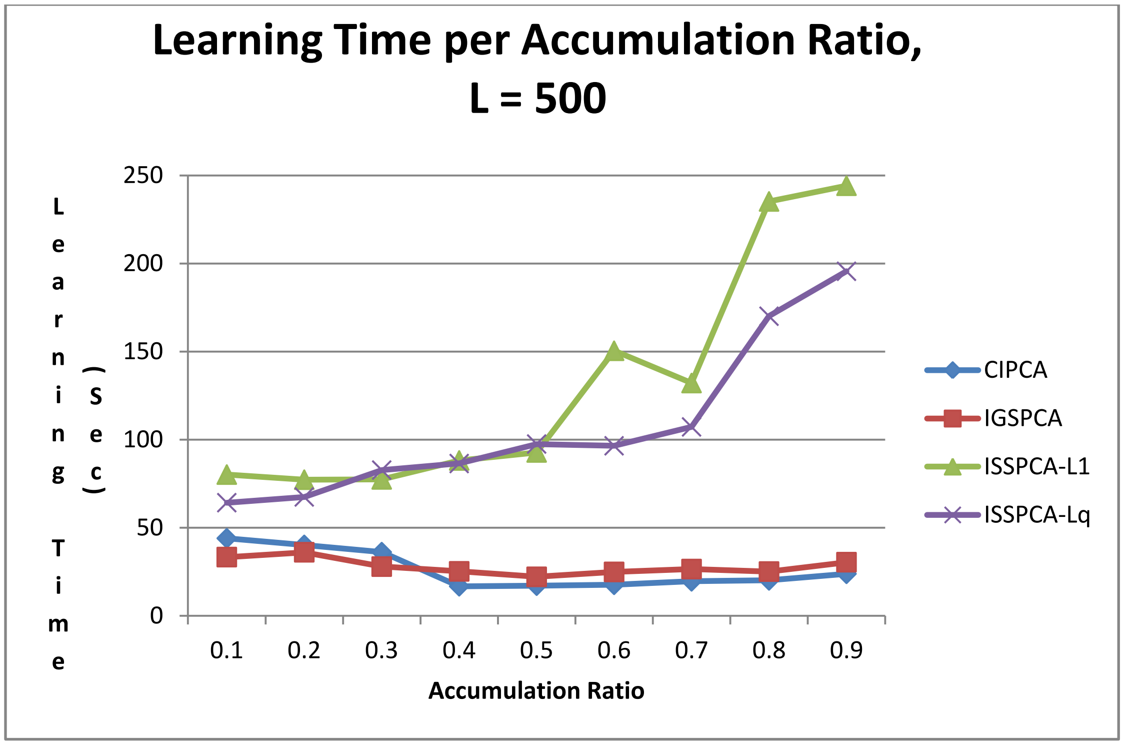

Figure 8. Intrusion detection dataset: Impact of accumulation ratio on learning time, L = 500.

Table 7 and Figure 8 show a close similarity in learning time for both CIPCA and IGSPCA. ISSPCA-L1 and ISSPCA-Lq required greater CPU time to achieve the same learning exercise. Besides, learning time for ISSPCA-L1 and ISSPCA-Lq increased with the value of the accumulation ratio factor.

The experiments performed for Figures 3, 4, and 5 were conducted using an accumulation ratio factor of 0.9. The impact evaluation of the accumulation ratio showed good dimensionality reduction size and classification accuracy rates for $\theta = 0.7$ (Figures 6 and 7).

For an accumulation ratio factor $\theta = 0.7$, the researcher then re-evaluated the influence of chunk size on the feature extraction and the classification accuracy. The experiment started with 100 samples, and then increased the chunk size by 100 until the size reached 1,400. Similar to all experiments conducted in this dissertation, five trials were executed and the average performance data points were recorded.

Table 8. Intrusion Detection Dataset: Impact of Chunk Size on Dimensionality, $\theta = 0.7$

Chunk Size	Algorithm (Reduced dimension)			
	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
100	10.4	9.8	15	13
200	10.4	10.2	15	14
300	10.4	10.6	15	13.5
400	10.4	9.8	14.5	11
500	10.4	9.4	13.33	11
600	10.4	9.4	15	11.67
700	10.4	9.8	15	12
800	10.6	9.6	15.5	13.25
900	10.4	10.2	13.8	14.8
1,000	10.4	9.4	16.4	11.2
1,100	10.6	10	15	10.2
1,200	10.4	10.4	13.6	12.4
1,300	10.6	9.6	13.6	12.4
1,400	10.6	9.8	13.6	13

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

Table 2 and Figure 3 show the dimensionality of the reduced datasets at the completion of the learning for various values of training sample chunk sizes with an accumulation ratio criteria $\theta = 0.9$. Table 8 and Figure 9 also provide the dimensionality of the reduced datasets at the completion of the learning for various values of training sample chunk sizes, but with a different value of accumulation ratio threshold $\theta = 0.7$.

A comparison of Figure 3 and Figure 10 shows a lower dimensionality of the resulting dataset for all four algorithms (CIPCA, IGSPCA, ISSPCA-L1, and ISSPCA-Lq) for $\theta = 0.7$. Also, the dimension of the extracted dataset is not affected by the chunk size as it was for $\theta = 0.9$. Therefore, for both CIPCA and IGSPCA, the dimension of the extracted dataset is around 10. This is a much better value in comparison with reduced dataset of about 18 dimensions with $\theta = 0.9$. Further, the dimensionality of the reduced dataset using ISSPCA-L1 fluctuates between 14 and 15 for $\theta = 0.7$, a much better result compared to up to 28 obtained for $\theta = 0.9$. Also, the dimensionality of the reduced dataset using ISSPCA-Lq varies between 11 and 15 for $\theta = 0.7$. This figure is lower than the score of up to 23 while using an accumulation ratio $\theta = 0.9$ (Figure 3).

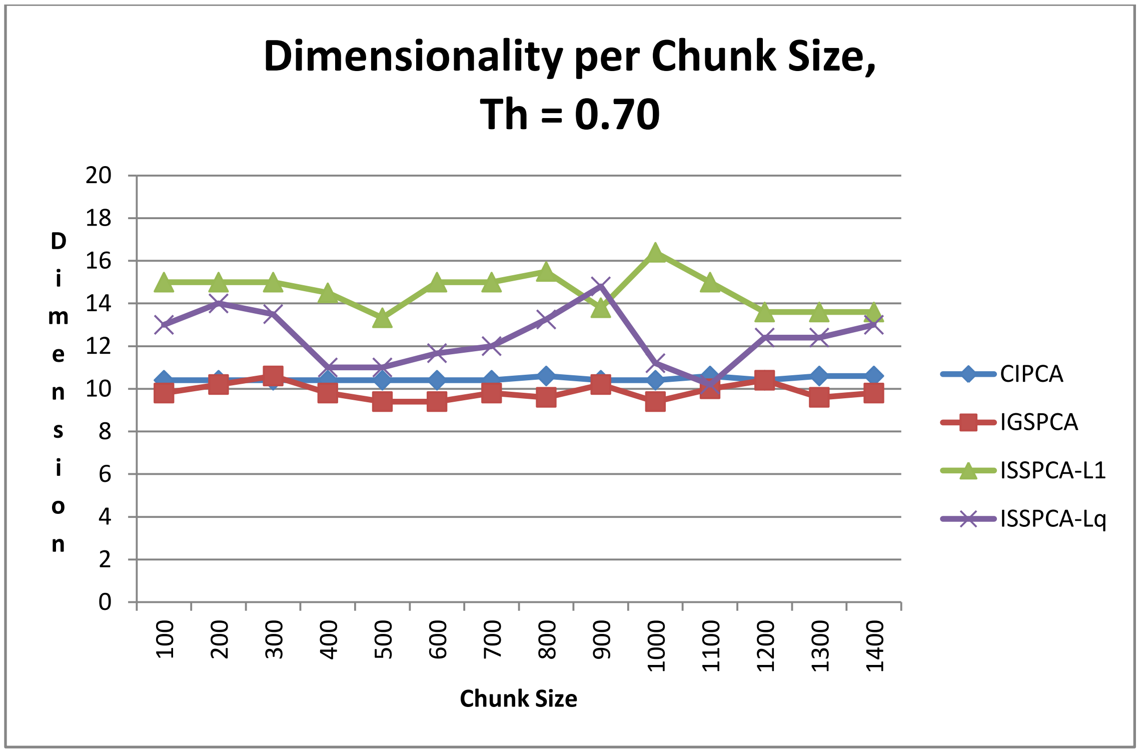

Figure 9. Intrusion detection dataset: Impact of chunk size on dimensionality, $\theta = 0.7$

Table 3 and Figure 4 show the classification accuracy of the reduced datasets at the completion of the learning for various values of training sample chunk sizes, using an accumulation ratio factor θ = 0.9. Table 9 and Figure 10 show the classification accuracy of the reduced datasets at the completion of the learning for various values of training sample chunk sizes, using an accumulation ratio factor θ = 0.7. For θ = 0.7, the classification accuracy rates for all four algorithms (CIPCA, IGSPCA, ISSPCA-L1, and ISSPCA-Lq) are greater than those obtained for θ = 0.9. In fact, they are all in the 99% range, with the exception of ISSPCA-Lq for L=800. ISSPCA-Lq has the greatest classification accuracy rate of 99.39% for L=1100.

Table 9. Intrusion Detection Dataset: Impact of Chunk Size on Classification Accuracy, θ = 0.7

Chunk Size	Algorithm (%)			
	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
100	99.09	99.27	99.37	99.25
200	99.09	99.08	99.12	99.28
300	99.08	99.17	99.17	99.24
400	99.2	99.11	99.24	99.19
500	99.19	99.11	99.13	99.19
600	99.09	99.18	99.18	99.22
700	99.13	99.05	99.18	99.12
800	99.27	99.17	99.26	98.89
900	99.08	99.24	99.16	99.15
1,000	99.11	99.1	99.1	99.15
1,100	99.11	99.06	99.16	99.39
1,200	99.08	99.16	99.08	99.16
1,300	99.19	99.19	99.13	99.21
1,400	99.11	99.22	99.19	99.14

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

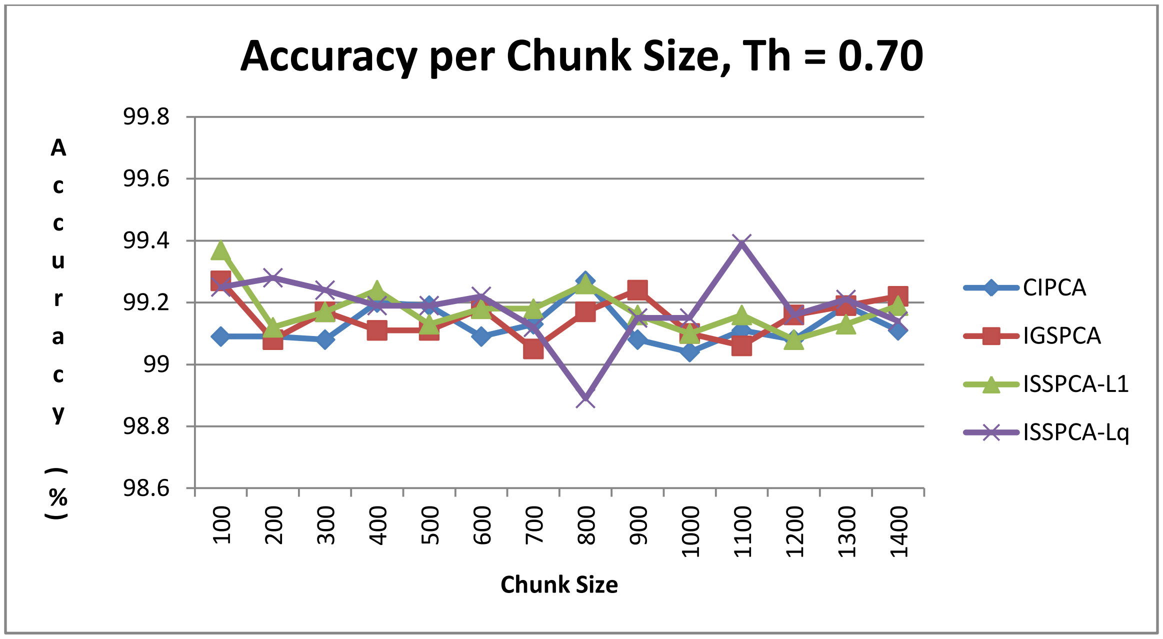

Figure 10. Intrusion detection dataset: Impact of chunk size on classification accuracy, θ = 0.7

Table 4 and Figure 5 show the learning time for various values of training sample chunk sizes, using an accumulation ratio factor θ = 0.9. Table 9 and Figure 10 show the learning time for various values of training sample chunk sizes, using an accumulation ratio factor θ = 0.7. For θ = 0.7, the CPU time required to complete the learning using each of the four algorithms (CIPCA, IGSPCA, ISSPCA-L1, and ISSPCA-Lq) are lower than that obtained for θ = 0.9. Therefore, both CIPCA and IGSPCA are faster algorithms than ISSPCA-L1 and ISSPCA-Lq.

Table 10. Intrusion Detection Dataset: Impact of Chunk Size on Learning Time, $\theta = 0.7$

Chunk Size	Algorithm (Seconds)			
	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
100	9.17	35.31	419.29	296.53
200	23.3	50.18	369.49	384.59
300	19.79	41.91	262.92	226.6
400	18.55	37.43	148.21	154.74
500	17.46	22.16	114.29	105.02
600	18.92	24.83	114.18	98.78
700	19.46	26.58	104.15	89.29
800	22.04	29.71	117.76	83.11
900	23.41	30.58	113.67	101.43
1,000	26.73	27.11	122.63	81.15
1,100	31.12	29.74	103.48	75.09
1,200	32.89	34.1	76.12	92.0
1,300	37.27	32.59	77.49	66.09
1,400	40.97	37.49	80.53	74.15

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

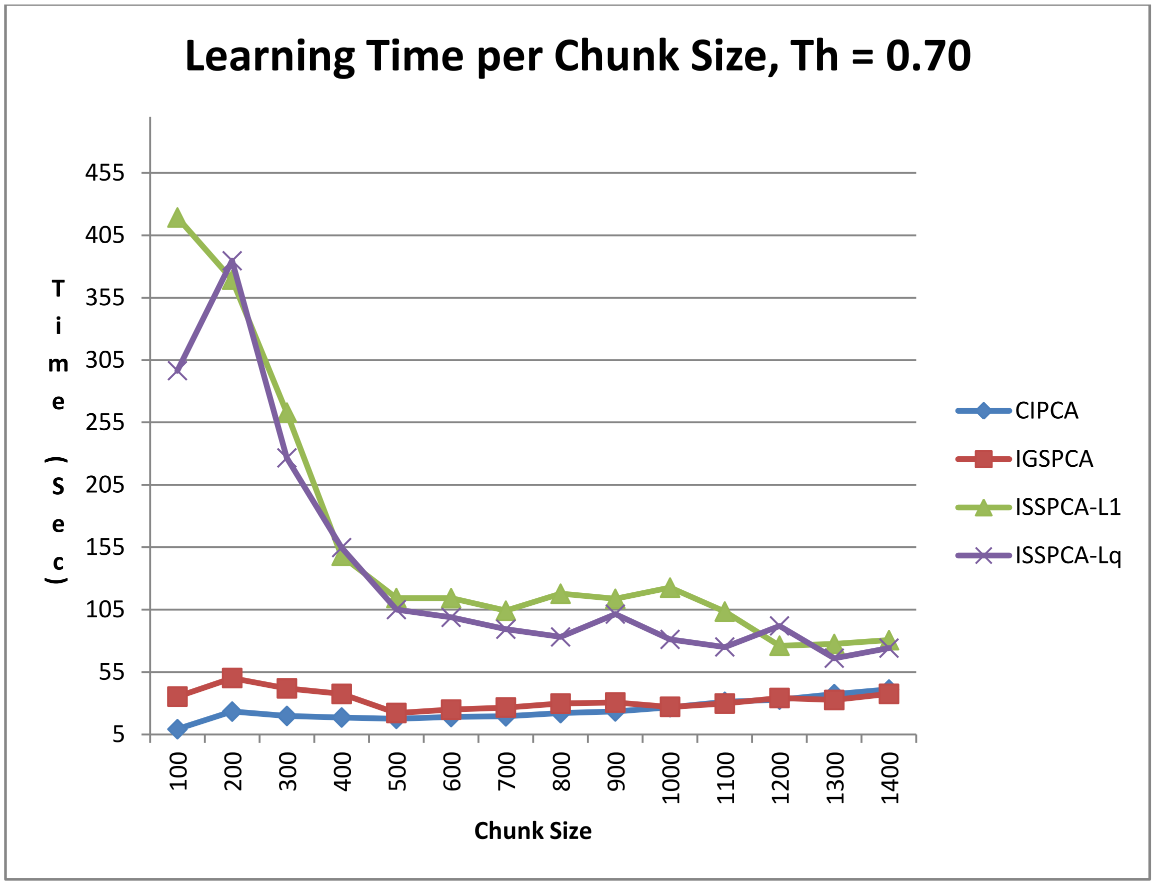

Figure 11. Intrusion detection dataset: Impact of chunk size on learning time, $\theta = 0.7$.

All experiments described so far in this dissertation used an initial data rate of 0.1 (or 10%) of the training data. Table 11 and Figure 12 show the dimensionality of the reduced datasets at the completion of the learning for various values of initial data rate with an accumulation ratio factor $\theta = 0.7$ and a constant chunk size L = 500. Figure 12 illustrates that both CIPCA and IGSPCA generated reduced subsets of lower dimensionality in comparison with ISSPCA-L1 and ISSPCA-Lq. Moreover, CIPCA and IGSPCA results were not influenced by the initial data rate. For increasing initial data rate, ISSPCA-L1 and ISSPCA-Lq generated subsets of higher dimensions. This effect was more pronounced for ISSPCA-Lq.

Table 11. Intrusion Detection Dataset: Impact of Initial Data on Dimensionality, $\theta = 0.7$ and L = 500

Initial Data	Algorithm (Reduced dimension)			
Rate	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
0.05	10.8	11.2	13.33	11
0.1	10.4	9.4	13.33	11
0.15	10.2	9.8	14.67	12
0.2	9.4	10.4	14.33	12.67
0.3	10	10	14.33	13.33
0.4	10.4	9.8	19	16
0.5	9.6	10.2	15.6	16

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

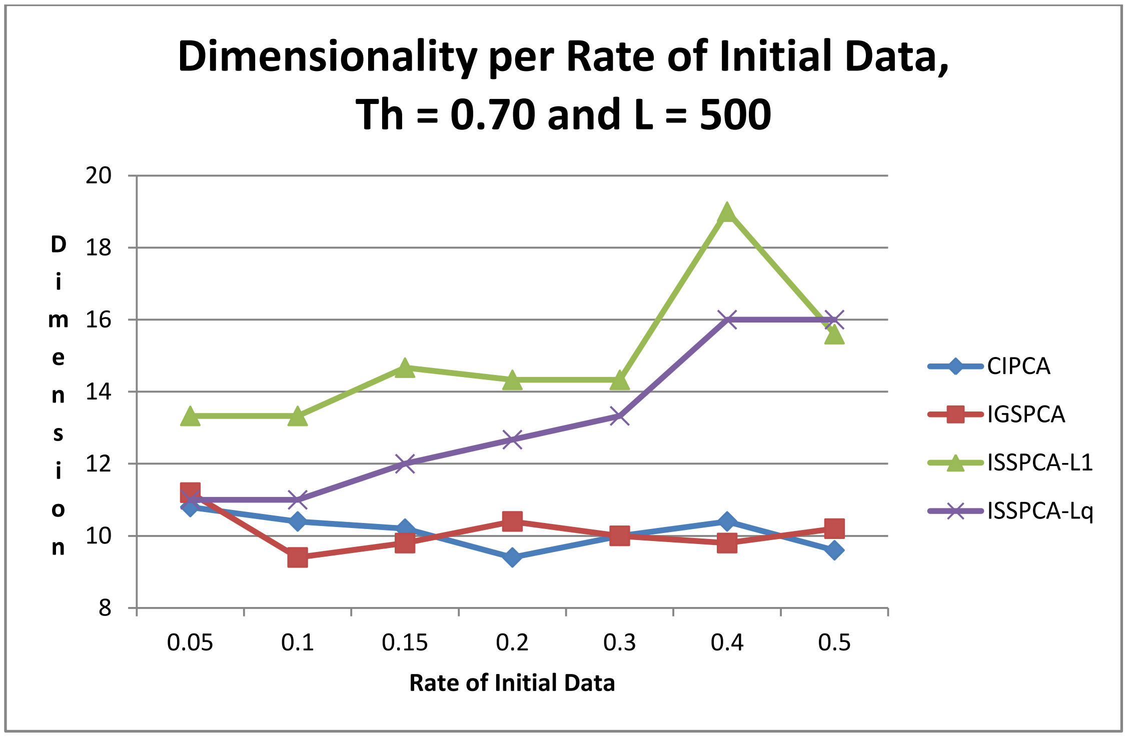

Figure 12. Intrusion detection dataset: Impact of initial data on dimensionality, $\theta = 0.7$ and L = 500.

Table 12 and Figure 13 show the classification accuracy of the reduced datasets at the completion of the learning for various values of initial data rate with an accumulation ratio factor $\theta = 0.7$ and a constant chunk size L = 500. While all four algorithms (CIPCA, IGSPCA, ISSPCA-L1, and ISSPCA-Lq) displayed an accuracy rate for an initial data rate of 0.1, the history is totally different for larger initial data size. IGSPCA, ISSPCA-L1 and ISSPCA-Lq were not influenced by the initial data rate. IGSPCA started to show a lower accuracy rate for an initial data rate of half the size of the whole training sample. CIPCA, on the other hand, displayed a very poor performance once the initial data rate was 15% or more of the training sample (Figure 13).

Table 12. Intrusion Detection Dataset: Impact of Initial Data on Classification Accuracy, $\theta = 0.7$ and L = 500

Initial Data	Algorithm (%)			
Rate	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
0.05	98.3	99.28	99.16	99.3
0.1	99.19	99.11	99.13	99.19
0.15	87.75	99.29	99.2	99.31
0.2	89.32	99.31	99.02	99.09
0.3	93.93	99.19	99.16	99.17
0.4	82.83	99.14	99.22	99.15
0.5	85.91	88.64	99.12	99.16

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

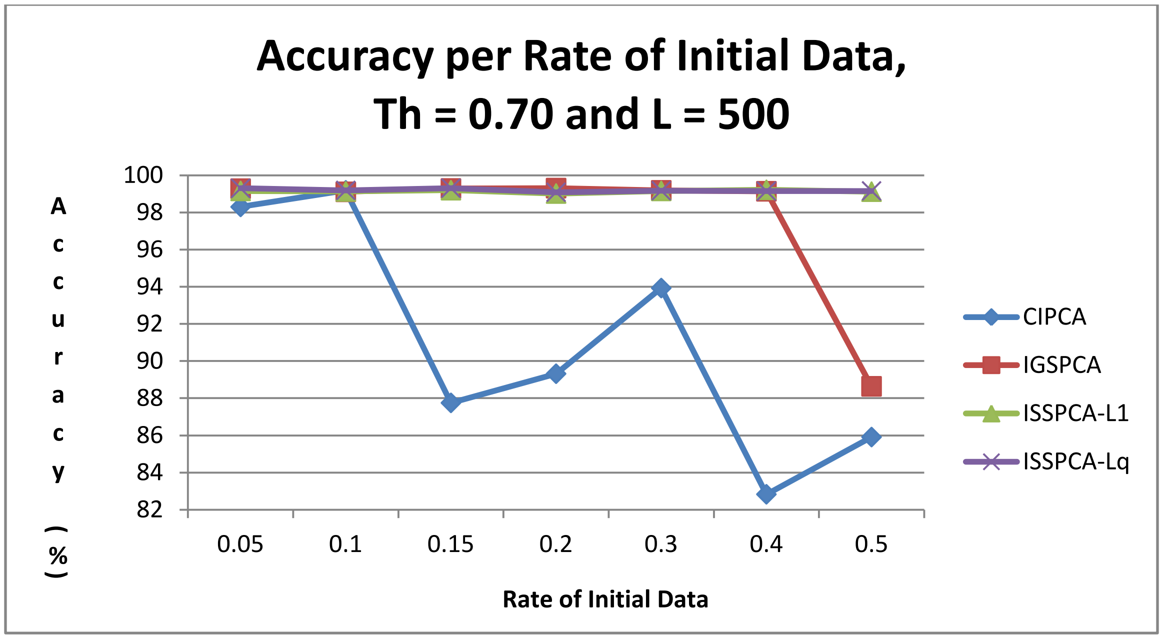

Figure 13. Intrusion detection dataset: Impact of initial data on classification accuracy, θ = 0.7 and L = 500.

Table 13 and Figure 14 show the CPU time at the completion of the learning for various values of initial data rate with an accumulation ratio factor $\theta = 0.7$ and a constant chunk size L = 500. As expected, the learning time diminished with the increase of initial data rate for all four algorithms (CIPCA, IGSPCA, ISSPCA-L1, and ISSPCA-Lq). CIPCA and IGSPCA showed similar learning times. Learning times for CIPCA and IGSPCA were the lower than those of ISSPCA-L1 and ISSPCA-Lq (Figure 14).

Table 13. Intrusion detection dataset – Impact of Initial data on learning time - $\theta = 0.7$ – L = 500

Initial Data	Algorithm (Seconds)			
Rate	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
0.05	24.02	23.15	130.61	102.32
0.1	20.6	21.36	107.49	90.17
0.15	18.48	24.89	119.99	95.26
0.2	17.21	19.66	110.96	94.9
0.3	15.34	16.48	101.66	82.11
0.4	15.09	14.41	125.06	95.41
0.5	12.06	12.34	88.95	69.56

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

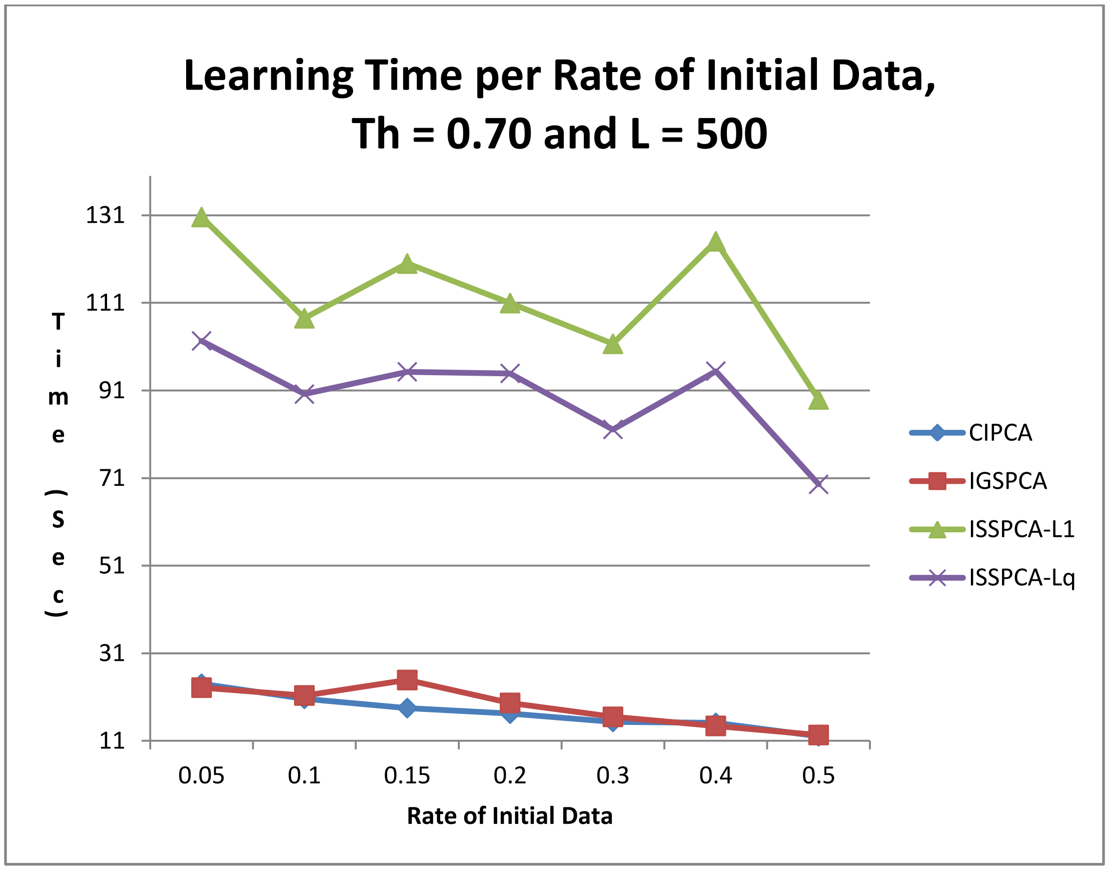

Figure 14. Intrusion detection dataset: Impact of initial data on learning time, $\theta = 0.7$ and L = 500.

Table 14 and Figure 15 show the number of original attributes contributing to the dimensionality reduction, at various dimensions for all four algorithms (CIPCA, IGSPCA, ISSPCA-L1, and ISSPCA-Lq). This test was conducted using the equivalent batch mode of each algorithm, which corresponded to a chunk size equal to the size of the whole training set. CIPCA used the regular PCA by performing linear combinations of all input variables. ISSPCA-L1 and ISSPCA-Lq used the majority of all input variables as well (Figure 15). IGSPCA, on the other hand, achieved linear combinations using few input attributes. This function facilitated the interpretability of the reduced subset by allowing a focus on specific variables for analysis.

Table 14. Intrusion Detection Dataset: Input Variables Contributing in Data Reduction

Dimension	Algorithm (Number of contributing attributes)			
Reduced Subset	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
10	41	18	39	39
11	41	20	39	39
12	41	19	39	39
13	41	21	39	39
14	41	24	39	39
15	41	25	39	39
16	41	28	39	39
18	41	30	39	39

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

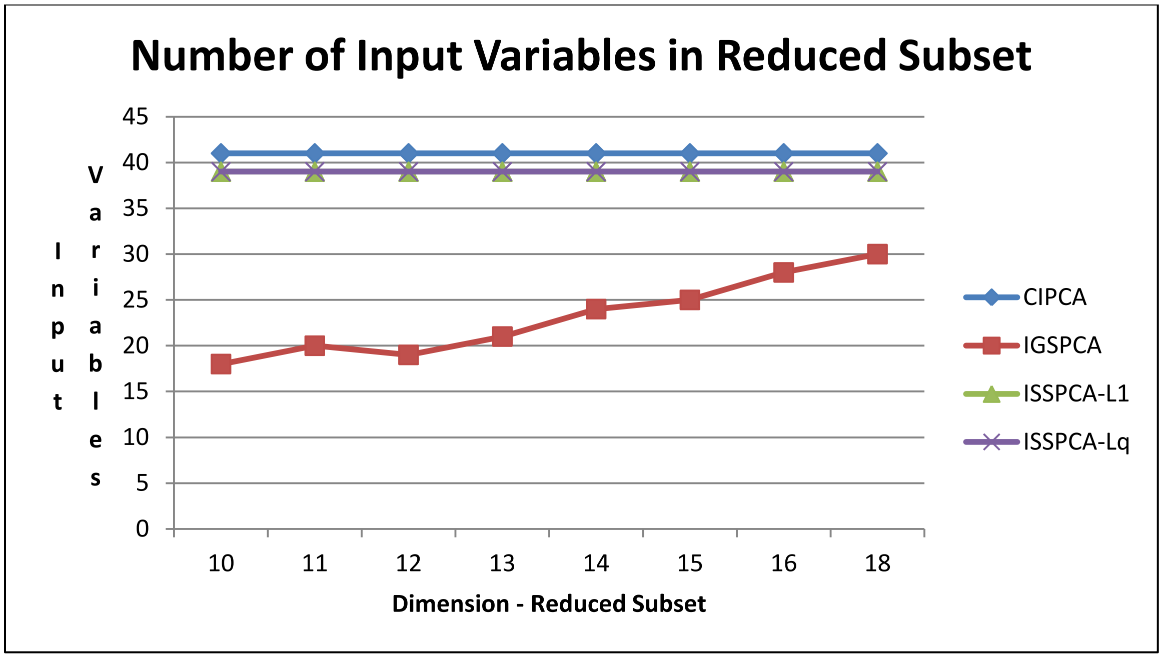

Figure 15. Intrusion detection dataset: Input variables contributing in data reduction.

Summary of Results

Several experiments were performed on the network intrusion detection dataset using four incremental algorithms: CIPCA and the following techniques proposed in this dissertation, namely IGSPCA, ISSPCA-L1, and ISSPCA-Lq.

For an accumulation ratio factor $\theta = 0.90$, IGSPCA performed as well as CIPCA in terms of dimensionality reduction. Both IGSPCA and CIPCA were not influenced by the data chunk size L. ISSPCA-L1 and ISSPCA-Lq, on the other hand, were impacted by the data chunk size. The fluctuation was accentuated for ISSPCA-L1. ISSPCA-L1 and ISSPCA-Lq achieved a better dimensionality reduction for L = 1,000 or greater (Figure 3). With respect to classification accuracy, the accuracy rates of all four algorithms were within the same range (Figure 4). However, both ISSPCA-L1 and ISSPCA-Lq required large CPU time to learn the training dataset (Figure 5).

For lower accumulation ratio criteria $\theta < 0.40$, ISSPCA-L1 and ISSPCA-Lq bested CIPCA and IGSPCA in terms of dimensionality reduction (Figure 6). However, learning times required by ISSPCA-L1 and ISSPCA-Lq were higher for accumulation ratio criteria $\theta > 0.40$ (Figure 8). The discrepancy in dimensionality reduction shrank for $\theta = 0.70$ (Figures 9 and 10). The variation in the rate of initial data had no impact on the dimensionality reduction using CIPCA and IGSPCA. It did, however, influence the performance of ISSPCA-L1 and ISSPCA-Lq (Figure 12). The accuracy rate of CIPCA was greatly influenced by the rate of initial data (Figure 13).

While CIPCA and IGSPCA achieved a similar performance overall, IGSPCA found principal components composed from a small number of the original variables (Figure 15). CIPCA found principal components as linear combinations of all original variables. Principal component are not easily interpretable if composed from all original variables. Figure 4 shows two accuracy rate outliers for IGSPCA using the intrusion detection dataset for L=200 and L=1200. The repeatability of the results leads us to believe that the lower accuracy in these specific cases are likely due to the order of giving training samples using those chunk sizes on this particular dataset, resulting in the eigenspace not having all energy.

Poker-Hand Dataset

The Poker-Hand dataset has 1,000,000 records of 10 predictive attributes and one class attribute each. Each instance corresponds to five playing cards drawn from a desk of 52 cards. The PokerHand dataset is not divided into training and testing samples. In order to create those, the researcher randomly split the original dataset; 20,000 records formed the testing sample, and 25,000 records formed the training subset.

Impact of Data Chunk Size

For chunk IPCA or CIPCA, an initial eigenspace was generated based on the value of 0.1% of initial data subset from the training samples. Then, the remaining training sample was fed to the algorithm in sequence for learning. To evaluate the influence of chunk size on the feature extraction and the classification accuracy, the experiment started with 10 samples, and then increased the chunk size by 20 until the number reached 200, with an accumulation ratio factor $\theta = 0.3$. The experiment was a one-pass because training samples were evaluated once and there was no overlap between chunks of training data. Five trials were executed, and the average performance data points were recorded.

Table 15 and Figure 16 show the dimensionality of the reduced datasets at the completion of the learning for various values of training sample chunk sizes. Table 15 and Figure 16 illustrate that for all four algorithms (CIPCA, IGSPCA, ISSPCA-L1, and ISSPCA-Lq), the dimensionality of the extracted dataset was not affected by the chunk size. In addition, the dimensionality of the reduced dataset was the same for each technique.

Table 15. Poker Dataset: Impact of Chunk Size on Dimensionality, $\theta = 0.3$

Chunk Size	Algorithm (Reduced dimension)			
	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
10	3	3	3	3
30	3	3	3	3
50	3	3	3	3
70	3	3	3	3
90	3	3	3	3
110	3	3	3	3
130	3	3	3	3
150	3	3	3	3
170	3	3	3	3
190	3	3	3	3
200	3	3	3	3

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

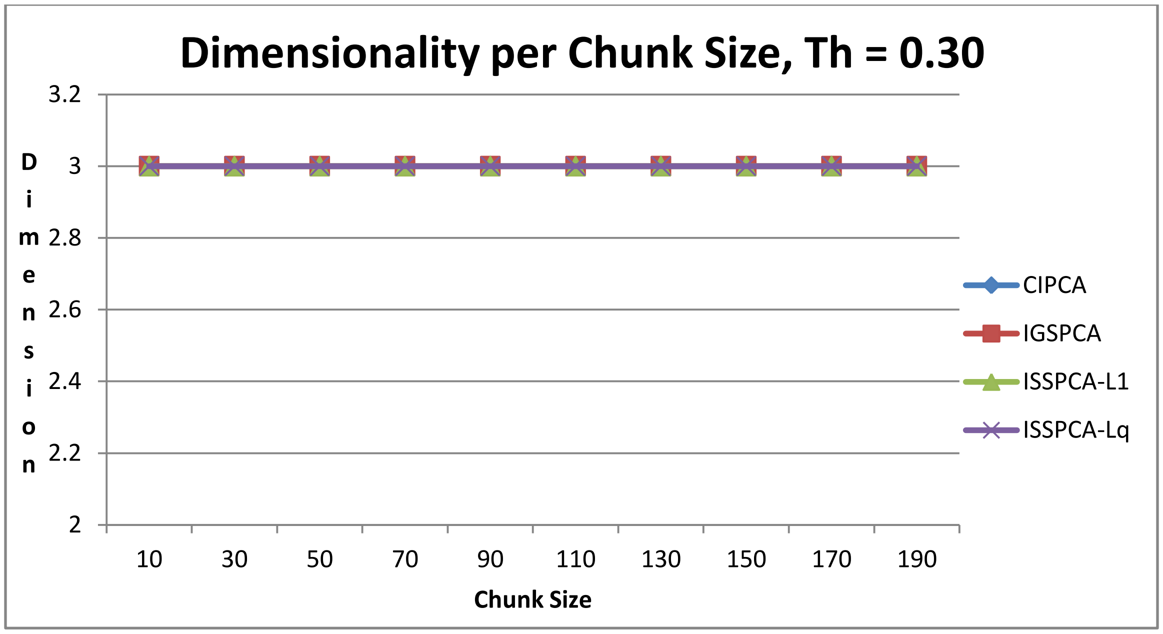

Figure 16. PokerHand dataset: Impact of chunk size on dimensionality, $\theta = 0.3$.

Table 16 and Figure 17 show the classification accuracy of the reduced datasets for at the completion of the learning for various values of training sample chunk sizes. The classification accuracy of CIPCA was the lowest for each chunk size of the training sample. The classification accuracy of ISSPCA-L1 and ISSPCA-Lq was greater than that of CIPCA. The classification accuracy of IGSPCA was the greatest for chunk sizes of the training sample of 50 or more. There is a tie between ISSPCA-Lq and IGSPCA for chunk size = 110. ISSPCA-Lq shows higher classification accuracy for chunk sizes of 10, 30 and 110.

Table 16. Poker-Hand Dataset: Impact of Chunk Size on Classification Accuracy, $\theta = 0.3$

Chunk Size	Algorithm (%)			
	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
10	49.26	49.95	49.39	50.81
30	49.26	49.6	49.8	50.21
50	49.26	50.4	49.1	49.752
70	49.26	50.1	50.26	50.14
90	49.26	50.24	49.89	49.96
110	49.25	50.1	49.8	50.12
130	49.26	50.35	50.34	49.18
150	49.25	50.49	49.63	49.86
170	49.26	50.15	49.86	49.82
190	49.26	50.24	50.17	49.19
200	49.26	50.53	50.17	49.94

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

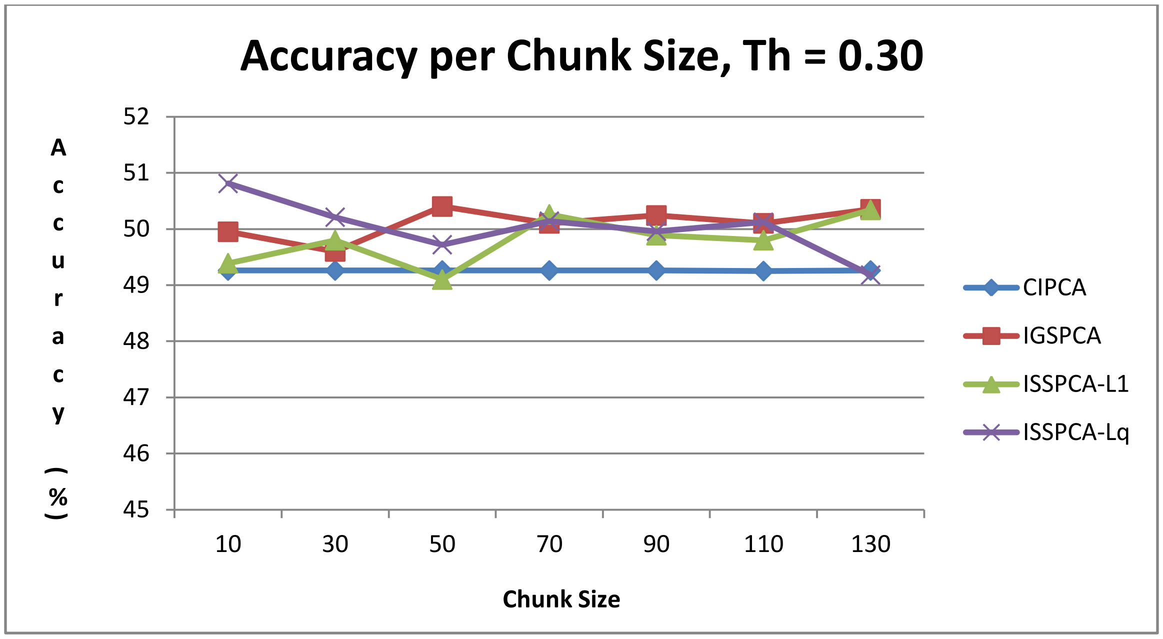

Figure 17. PokerHand dataset: Impact of chunk size on classification accuracy, $\theta = 0.3$.

Table 17 and Figure 18 show the learning time to generate the reduced datasets for various values of training sample chunk sizes. The learning times required for either ISSPCA-L1 or ISSPCA-Lq were the highest, especially for lower chunk size values. The learning times required by CIPCA and IGSPCA were mostly identical and were not significantly impacted by the training sample chunk size.

Table 17. Poker-Hand Dataset – Impact of Chunk Size on Learning Time, $\theta = 0.3$

Chunk Size	Algorithm (Seconds)			
	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
10	1.12	6.32	61.33	100.57
30	0.63	2.48	20.84	33.09
50	0.63	1.78	12.67	20.11
70	0.57	1.4	9.27	14.21
90	0.57	1.33	7.32	11.09
110	0.53	2.2	10.7	14.74
130	0.85	1.79	9.25	13.36
150	0.87	1.79	8.14	11.67
170	0.92	1.68	7.22	10.37
190	0.92	1.69	6.75	9.35
200	0.83	1.62	6.21	8.96

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

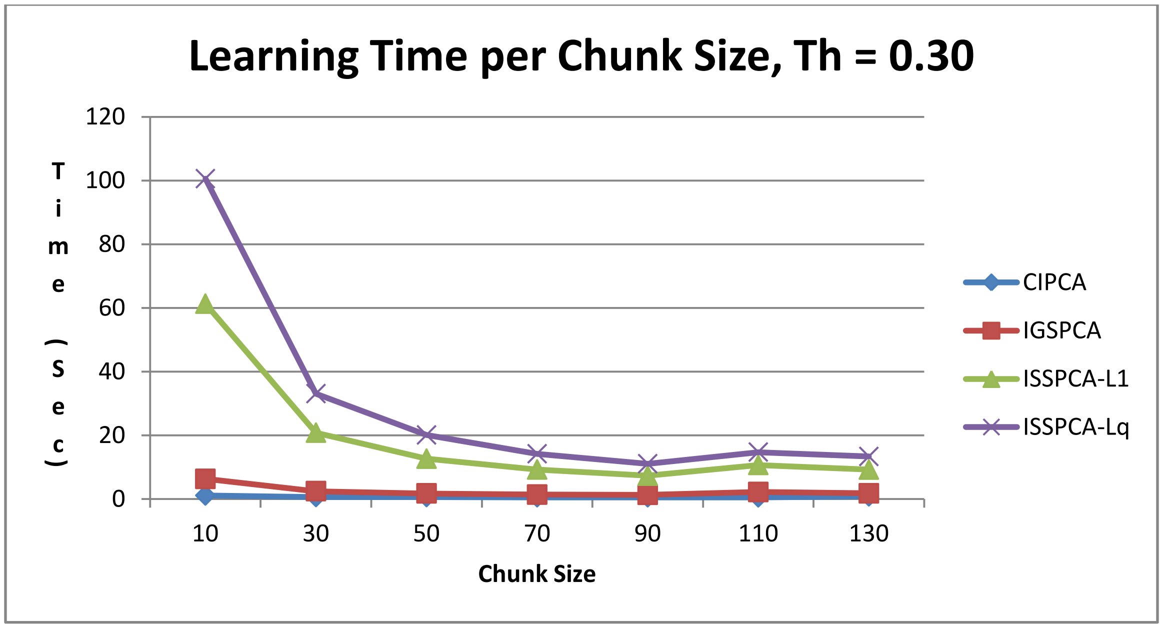

Figure 18. PokerHand dataset: Impact of chunk size on learning time, $\theta = 0.3$.

Summary of Results

Multiple experiments were performed on the PokerHand dataset using all four incremental algorithms (CIPCA, IGSPCA, ISSPCA-L1, and ISSPCA-Lq). For an accumulation ratio factor $\theta = 0.30$, all four incremental algorithms performed identically in terms of dimensionality reduction and were not at all influenced by the data chunk size L (Figure 16). With respect to classification accuracy, the accuracy rates of all three algorithms proposed in this dissertation, namely IGSPCA, ISSPCA-L1 and ISSPCA-Lq, were higher than that obtained using CIPCA (Figure 17). Both ISSPCA-L1 and ISSPCA-Lq required large CPU time to learn the training dataset (Figure 18).

WaveForm Dataset

The WaveForm dataset has 5,000 instances of 21 predictive attributes and one class attribute each. Attributes are continuous values between 0 and 6. The generator is used to acquire instances with gradual concept drift (Breiman et al., 1984). The WaveForm dataset is not divided into training and testing samples. In order to create those, the researcher randomly split the original dataset into two samples; 1,000 records formed the testing sample, and 4,000 records formed the training subset.

Impact of Data Chunk Size

For chunk IPCA (CIPCA), an initial eigenspace was generated based on the value of 0.1% of initial data subset from the training samples. Then, the remaining training sample was fed to the algorithm in sequence for learning. To evaluate the influence of chunk size on the feature extraction and the classification accuracy, the experiment started with 10 samples, then increased the chunk size by 20 until the size reached 100 with an accumulation ratio factor $\theta = 0.8$. The experiment was a one-pass because training samples were evaluated once, and no overlap existed between chunks of training data. Five trials were executed and the average performance data points were recorded.

Table 18 and Figure 19 show the dimensionality of the reduced datasets at the completion of the learning for various values of training sample chunk sizes. Table 18 and Figure 19 illustrate that IGSPCA achieved a better dimensionality reduction than CIPCA, ISSPCA-L1, and ISSPCA-Lq. The dimension of the extracted dataset was not much affected by the chunk size. The dimensionality of the reduced dataset for ISSPCA-L1increased unexpectedly for L = 100.

Table 18. WaveForm Dataset: Impact of Chunk Size on Dimensionality, $\theta = 0.8$

Chunk Size	Algorithm (Reduced dimension)			
	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
10	10.8	9.4	11.4	11.6
20	10.4	9.4	11.2	11.2
30	10.4	9.6	12.2	11.8
40	10.2	9.4	12.4	12.4
50	10.2	9.6	12.4	12.6
60	10.2	9.6	12.4	12.6
70	10.4	9.6	12.2	12.6
80	10.2	9.6	12.2	12.8
90	10.4	9.6	12.2	12.8
100	10.2	9.6	18.4	12.6

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

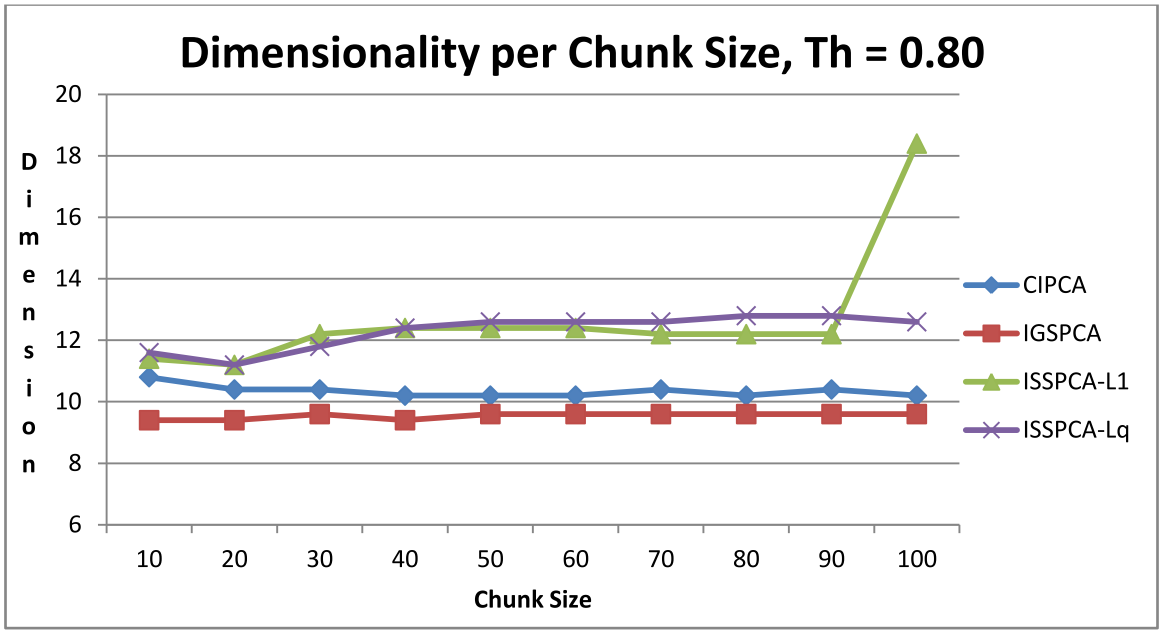

Figure 19. WaveForm dataset: Impact of chunk size on dimensionality, $\theta = 0.8$.

Table 19 and Figure 20 show the classification accuracy of the reduced datasets at the completion of the learning for various values of training sample chunk sizes. The classification accuracy for both CIPCA and IGSPCA are 81.46 ± 0.44 and 80.79 ± 0.41 respectively; they are not significantly impacted by the chunk size of the training sample. Figure 20 shows that both ISSPCA-L1 and ISSPCA-Lq had lower classification accuracy rates.

Table 19. WaveForm Dataset: Impact of Chunk Size on Classification Accuracy, $\theta = 0.8$

Chunk Size	Algorithm (%)			
	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
10	81.3	81.18	75.88	72.64
20	81.62	80.82	75.62	72.48
30	81.18	80.76	74.76	73.58
40	81.2	80.78	73.94	76.4
50	81.4	80.64	76.08	75
60	81.9	80.48	73.76	76.96
70	81.78	81.2	73.5	75.08
80	81.56	80.92	71.28	75.9
90	81.02	80.5	72.84	75.98
100	81.6	80.38	77.96	76.76

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

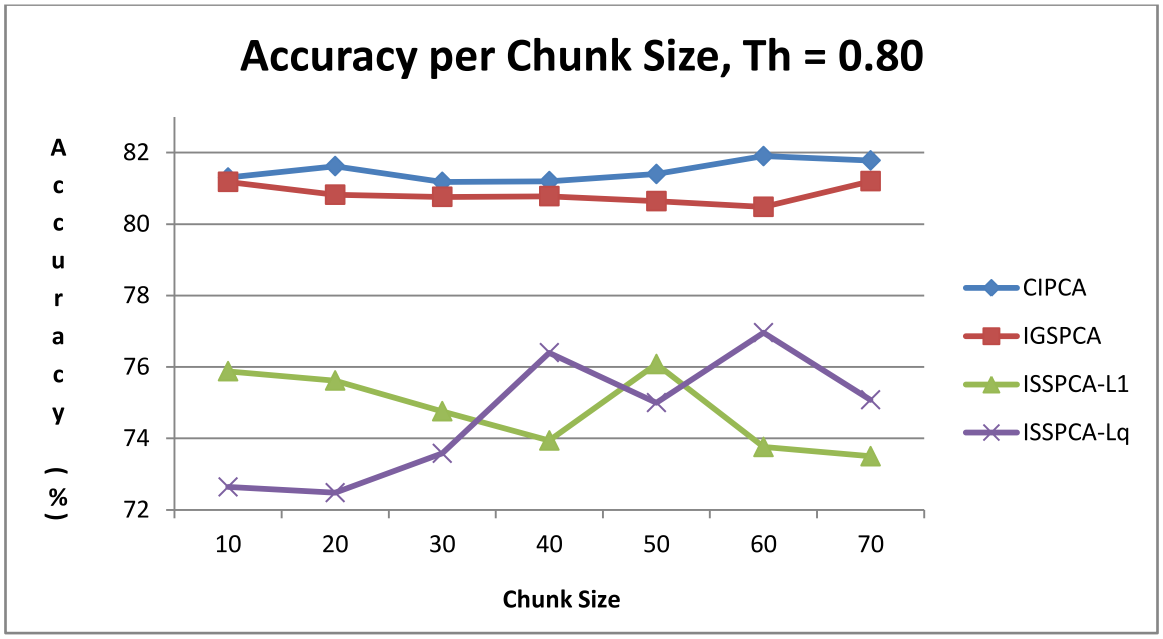

Figure 20. WaveForm dataset: Impact of chunk size on classification accuracy, $\theta = 0.8$.

Table 20 and Figure 21 show the learning time to generate the reduced subsets at the completion of the learning for various values of training sample chunk sizes. The learning time required for either ISSPCA-L1 or ISSPCA-Lq was the highest, especially for lower chunk size values. The learning time required by IGSPCA was a little greater than that required by CIPCA. CIPCA learning time was 0.17 ± 0.02 second for any training sample chunk size greater or equal 20. IGSPCA was more expensive for L = 10 and L = 20.

Table 20. WaveForm Dataset: Impact of Chunk Size on Learning Time, $\theta = 0.8$

Chunk Size	Algorithm (Seconds)			
	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
10	0.24	2.52	25.76	25.05
20	0.17	1.55	13.33	13.46
30	0.16	1.16	9.3	8.96
40	0.16	1.0	7.02	6.88
50	0.15	0.88	5.41	5.67
60	0.17	0.85	4.68	5
70	0.16	0.81	3.81	4.34
80	0.17	0.78	3.42	3.75
90	0.18	0.73	3.08	8.17
100	0.19	0.72	8.17	2.88

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

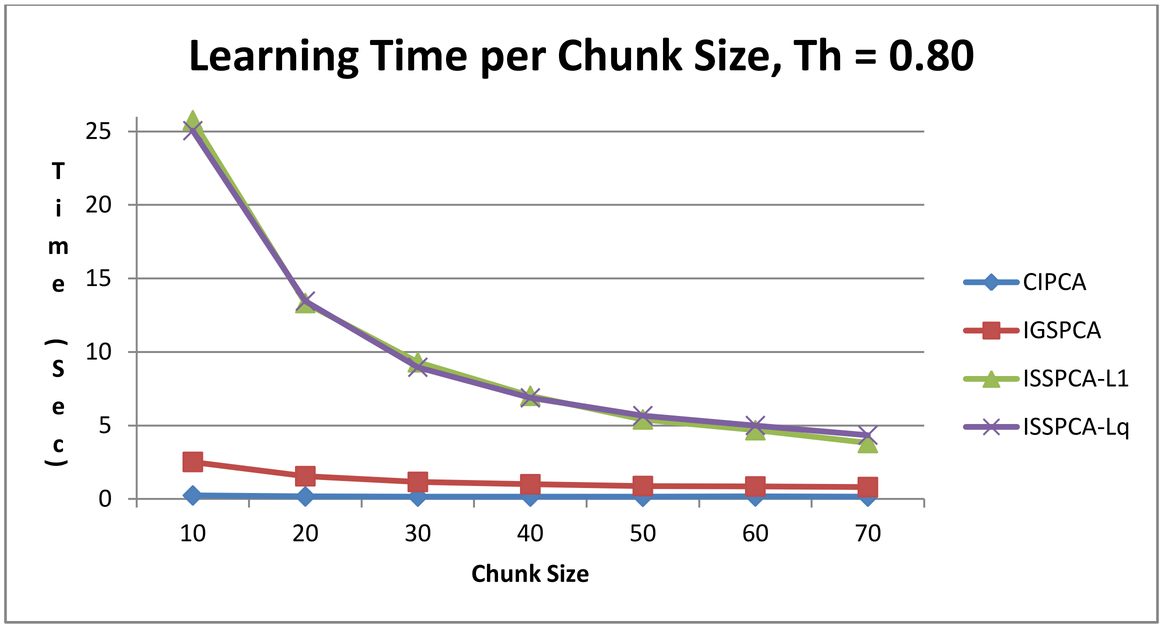

Figure 21. WaveForm dataset: Impact of chunk size on learning time, $\theta = 0.8$.

Summary of Results

Experiments were conducted on the WaveForm dataset using all four incremental algorithms (CIPCA, IGSPCA, ISSPCA-L1, and ISSPCA-Lq). For an accumulation ratio factor $\theta = 0.80$, IGSPCA performed better in term of dimensionality reduction, followed by CIPCA, ISSPCA-L1, and ISSPCA-Lq. While ISSPCA-L1 and ISSPCA-Lq yielded the same results for almost every chunk size L, ISSPCA-L1's dimensionality increased by 50% for L =100, from 12 to 18 principal components (Figure 19). CIPCA and IGSPCA were not influenced by the data chunk size L (Table 18). With respect to classification accuracy, the accuracy rates of CIPCA and IGSPCA were higher than those obtained using ISSPCA-L1 and ISSPCA-Lq (Figure 20). Both ISSPCA-L1 and ISSPCA-Lq required large CPU time to learn the training dataset (Figure 21), especially for lower values of chunk size, L < 40 (Table 20).

WaveFormNoise Dataset

The WaveFormNoise dataset has 5,000 instances of 40 predictive attributes and one class attribute each. Attributes are continuous values between 0 and 6. The later 19 attributes are all noise attributes with mean 0 and variance 1. The generator was used to acquire instances with gradual concept drift (Breiman et al., 1984). Similar to the WaveForm dataset, the WaveFormNoise dataset is not divided into training and testing samples. The researcher randomly split the original dataset into two samples; 1,000 records formed the testing sample, and 4,000 records formed the training subset.

Impact of Data Chunk Size

For chunk IPCA (CIPCA), an initial eigenspace was generated based on the value of 0.1% of initial data subset from the training samples. Then, the remaining training sample was fed to the algorithm in sequence for learning. To evaluate the influence of chunk size on the feature extraction and the classification accuracy, the experiment started with 10 samples, then increased the chunk size by 10 until it reached 50 with an accumulation ratio criteria $\theta = 0.4$. The experiment was a one-pass because training samples were evaluated once, and no overlap existed between chunks of training data. Five trials were executed and the average performance data points were recorded.

Table 21 and Figure 22 show the dimensionality of the reduced datasets at the completion of the learning for various values of training sample chunk sizes. Table 21 and Figure 22 also show a similar performance between IGSPCA and CIPCA in terms of dimensionality reduction in comparison with ISSPCA-L1 and ISSPCA-Lq. In addition, the dimension of the extracted dataset obtained by IGSPCA and CIPCA was not impacted by the training chunk size, whereas ISSPCA-L1 and ISSPCA-Lq performed better for larger values of L.

Table 21. WaveFormNoise Dataset: Impact of Chunk Size on Dimensionality, $\theta = 0.4$

Chunk Size	Algorithm (Reduced dimension)			
	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
10	8	8	11.4	11.6
20	8.2	8	11.6	10.6
30	8	8	10.2	9.8
40	8	8.2	10.8	10.2
50	8	7.8	10	10.2

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

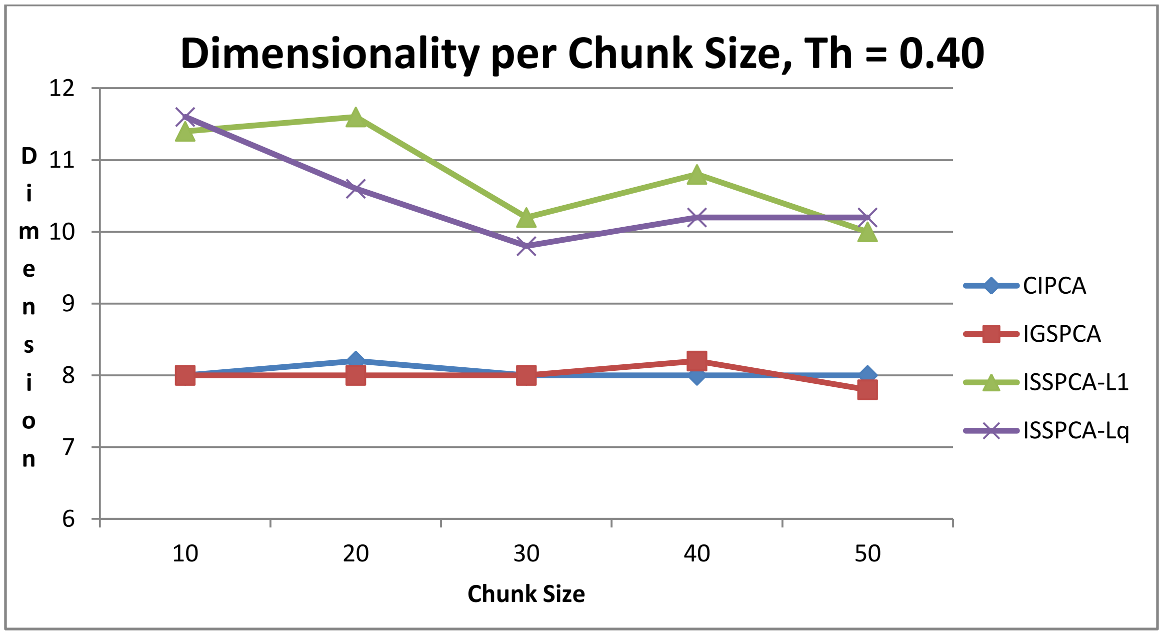

Figure 22. WaveFormNoise dataset: Impact of chunk size on dimensionality, $\theta = 0.4$.

Table 22 and Figure 23 show the classification accuracy of the reduced datasets at the completion of the learning for various values of training sample chunk sizes. The classification accuracy for both CIPCA and IGSPCA 80.35 ± 0.83 and 80.2 ± 0.32

respectively, and did not display any significant impact with respect to the chunk size of the training sample. Figure 23 shows that both ISSPCA-L1 and ISSPCA-Lq had lower classification accuracy rates.

Table 22. WaveFormNoise Dataset: Impact of Chunk Size on Classification Accuracy, $\theta = 0.4$

Chunk Size	Algorithm (%)			
	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
10	81.18	80.2	59.68	64.48
20	79.8	80.18	62.84	65.34
30	80.42	79.88	59.6	62.2
40	79.84	80.02	59.28	64.88
50	79.52	80.52	63.62	63.8

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

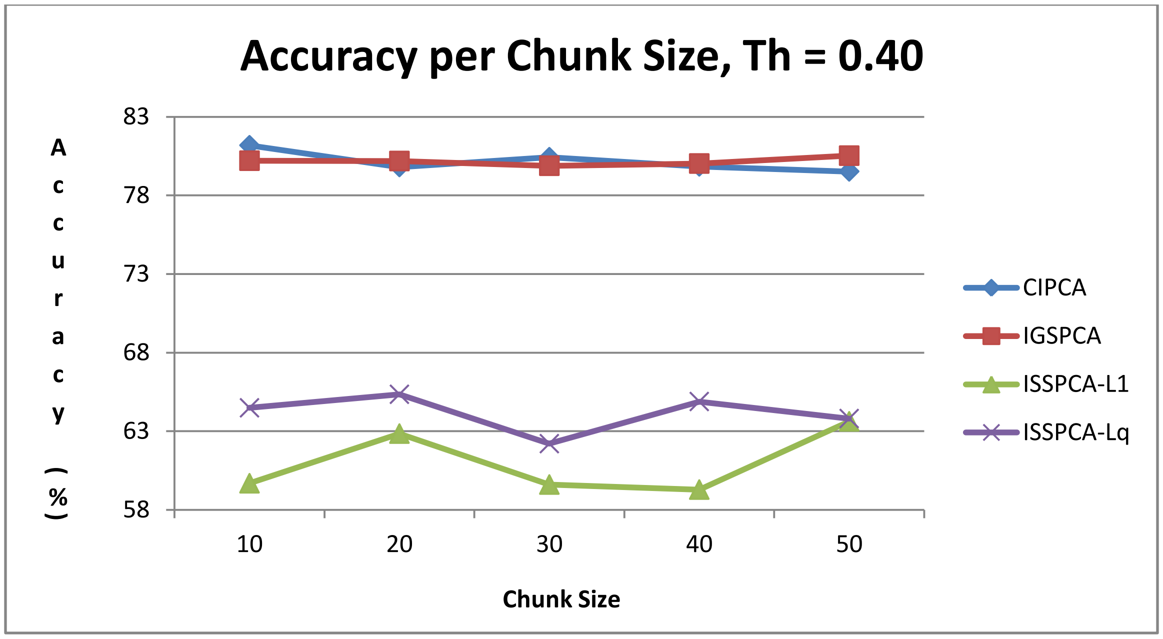

Figure 23. WaveFormNoise dataset: Impact of chunk size on classification accuracy, $\theta = 0.4$.

Table 23 and Figure 24 show the learning time to generate the reduced subsets at the completion of the learning for various values of training sample chunk sizes. Similar to WaveForm dataset, the learning time required for either ISSPCA-L1 or ISSPCA-Lq was the highest, particularly for lower chunk size values. CIPCA learning time was 0.215 ± 0.055 Sec. IGSPCA learning time was 1.62 ± 0.71 Sec. IGSPCA was more expensive for L = 10 and L = 20.

Table 23. WaveFormNoise Dataset: Impact of Chunk Size on Learning Time, $\theta = 0.4$

Chunk Size	Algorithm (Seconds)			
	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
10	0.27	2.33	22.8	24.97
20	0.19	1.38	12.16	11.84
30	0.16	1.11	6.86	7.46
40	0.17	0.99	5.86	4.47
50	0.16	0.91	4.47	4.77

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

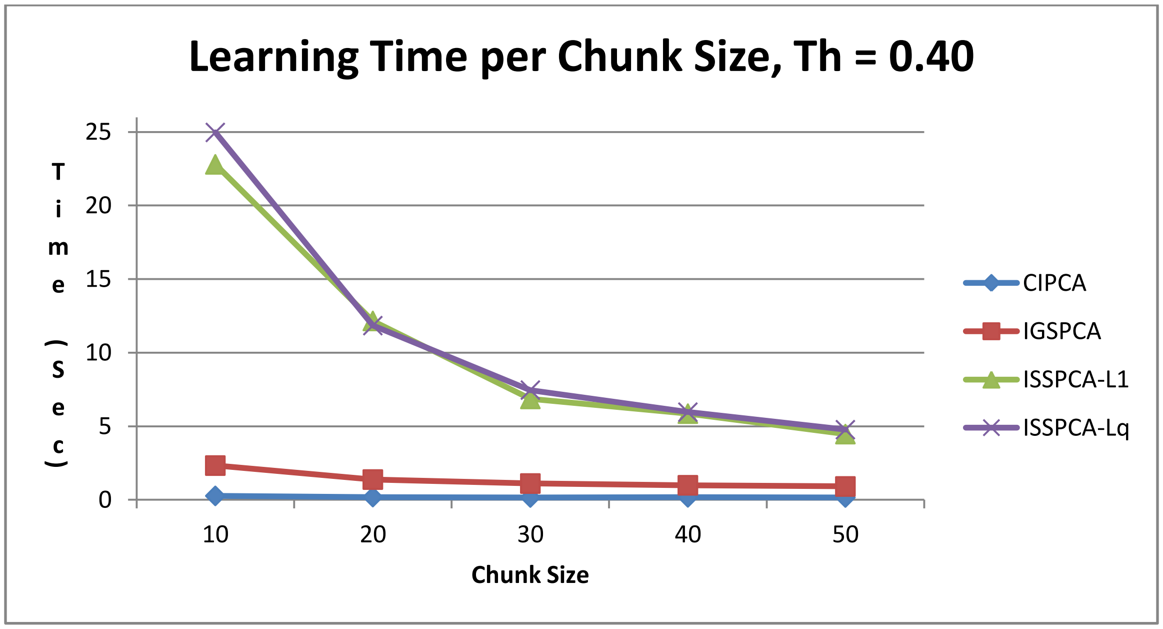

Figure 24. WaveFormNoise dataset: Impact of chunk size on learning time, $\theta = 0.4$.

Table 24 and Figure 25 show the number of WaveFormNoise original attributes contributing to the dimensionality reduction, at various dimensions for all four algorithms (CIPCA, IGSPCA, ISSPCA-L1, and ISSPCA-Lq). This experiment used the equivalent batch mode of each of the four algorithms. This approach corresponded to a chunk size equal to the size of the whole training set. CIPCA used the regular PCA by performing linear combinations of all input variables. Surprisingly, ISSPCA-L1 and ISSPCA-Lq used all WaveFormNoise input variables as well to form the reduce data subset (Figure 25 and Table 24). IGSPCA, on the other hand, achieved linear combinations using few input attributes. This ability facilitated the interpretability of the reduced subset by allowing the algorithm to focus on specific variables for analysis.

Table 24. WaveForm Noise Dataset: Input Variables Contributing in Data Reduction

Dimension	Algorithm (Number of contributing attributes)			
Reduced Subset	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
7	40	23	40	40
8	40	24	40	40
10	40	26	40	40
11	40	27	40	40
12	40	28	40	40

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

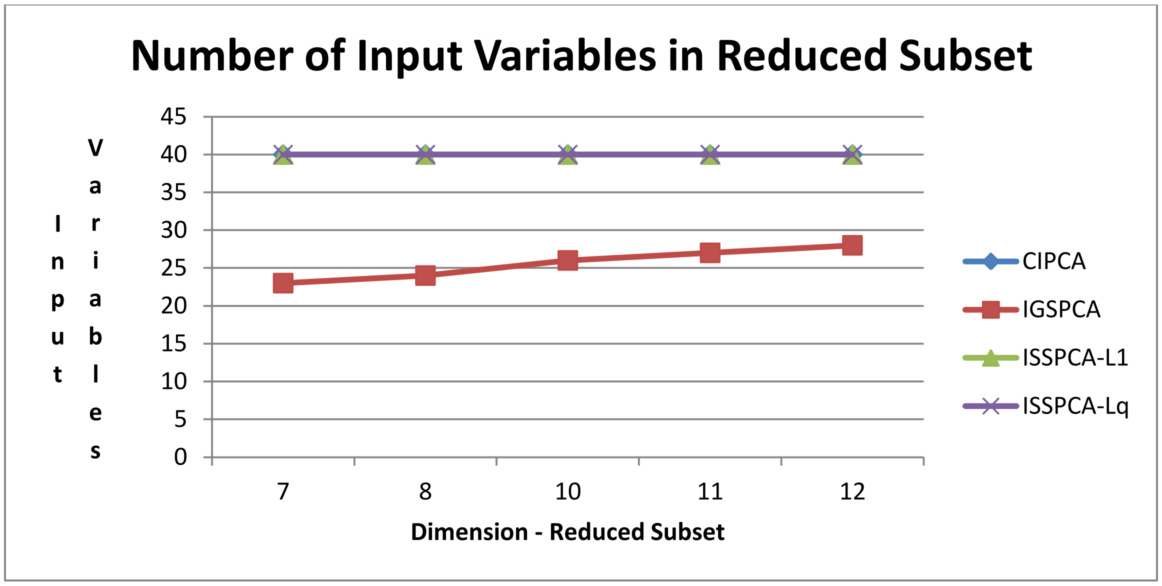

Figure 25. WaveFormNoise dataset: Input variables contributing in data reduction.

Summary of Results

Multiple experiments were conducted on the WaveFormNoise dataset using all four incremental algorithms (CIPCA, IGSPCA, ISSPCA-L1, and ISSPCA-Lq). For an accumulation ratio factor $\theta = 0.40$, IGSPCA and CIPCA performed better in term of dimensionality reduction, followed by ISSPCA-Lq and ISSPCA-L1 (Figure 22). CIPCA and IGSPCA were not influenced by the data chunk size L (Table 21). With respect to classification accuracy, the match rates of CIPCA and IGSPCA were higher than those obtained using ISSPCA-L1 and ISSPCA-Lq (Figure 23). Both ISSPCA-L1 and ISSPCA-Lq required large CPU time to learn the training dataset (Figure 24), especially for lower values of chunk size L < 40 (Table 23). While CIPCA and IGSPCA achieved similar performance overall, IGSPCA found principal components composed from a small number of the original variables (Figure 25). CIPCA found the reduced subset as linear combinations of all original variables. Principal component are not easily interpretable if composed from all original variables.

Incremental Sparse Principal Component Analysis: Impact of Concept Drift and Concept Shift

Concept drift and concept shift cause variables in data streams to change with the potential to degrade the predictability accuracy in the long run. Experiments in this research aimed to validate the effectiveness of the proposed algorithms IGSPCA, ISSPCA-L1, and ISSPCA-Lq in comparison to CIPCA to handle concept drift and concept shift. All four algorithms used the accumulation ratio when updating the eigenspace to check if its dimensionality should be augmented. The dimensionality was increased if the new training chunk sample included new energy or important information in the complementary eigenspace.

The following two datasets were generated with gradual concept drift: WaveForm and WaveFormNoise. The WaveForm dataset consists of three classes of waves, 21 attributes, and 5,000 instances. The WaveFormNoise dataset is constituted of three classes of waves, 40 attributes and 5,000 instances.

Table 21 and Figure 22 show a similar performance between IGSPCA and CIPCA in term of dimensionality reduction. In addition, the dimension of the extracted dataset obtained by IGSPCA and CIPCA was not impacted by the training chunk size. Regarding the classification accuracy, Table 22 and Figure 23 show that accuracy rates from both CIPCA and IGSPCA were within the same range and did not display any impact with respect to the chunk size of the training sample.

The WaveFormNoise dataset has 5,000 instances with gradual concept drift (Breiman et al., 1984). The researcher generated a new dataset with concept shift by shuffling the values of the first and the last attributes of the WafeFormNoise dataset, for all instances and maintaining the class label intact (Morshedlou & Barforoush, 2009).

The researcher then randomly split the new dataset into two samples; 1,000 records formed the testing sample, and 4,000 records formed the training subset.

Impact of Data Chunk Size

For chunk IPCA (CIPCA), an initial eigenspace was generated based on the value of 0.1% of initial data subset from the training samples. Then, the remaining training sample was fed to the algorithm in sequence for learning. To evaluate the influence of chunk size on the feature extraction and the classification accuracy, the experiment started with 10 samples, then increased the chunk size by 10 until it reached 50 with an accumulation ratio criteria $\theta = 0.4$. The experiment was a one-pass because training samples were evaluated once, and no overlap existed between chunks of training data. Five trials were executed and the average performance data points were recorded.

Table 25 and Figure 26 show the dimensionality of the reduced datasets at the completion of the learning for various values of training sample chunk sizes. Table 25 and Figure 26 also show a similar performance between IGSPCA and CIPCA in terms of dimensionality reduction in comparison with ISSPCA-L1 and ISSPCA-Lq. It is remarkable to notice no difference between IGSPCA and CIPCA in term of dimensionality reduction using either the original or the modified WafeFormNoise dataset (table 21 and table 25). For L in [10 … 50], using the original and the modified WafeFormNoise datasets, ISSPCA-L1 achieved a dimensionality reduction of 10.8 ± 0.8 and 10.2 ± 0.9 respectively. ISSPCA-Lq achieved a dimensionality reduction of 10.7 ± 0.9 for the WafeFormNoise and 10.3 ± 0.9 for the modified dataset.

Table 25. WaveFormNoise_ConceptShift: Impact of Chunk Size on Dimensionality, θ = 0.4

Chunk Size	Algorithm (Reduced dimension)			
	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
10	8	8	10.4	11
20	8.2	8	11.6	11
30	8	8	10.4	9.8
40	8	8.2	9.8	11.2
50	8	7.8	10.2	9.4

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

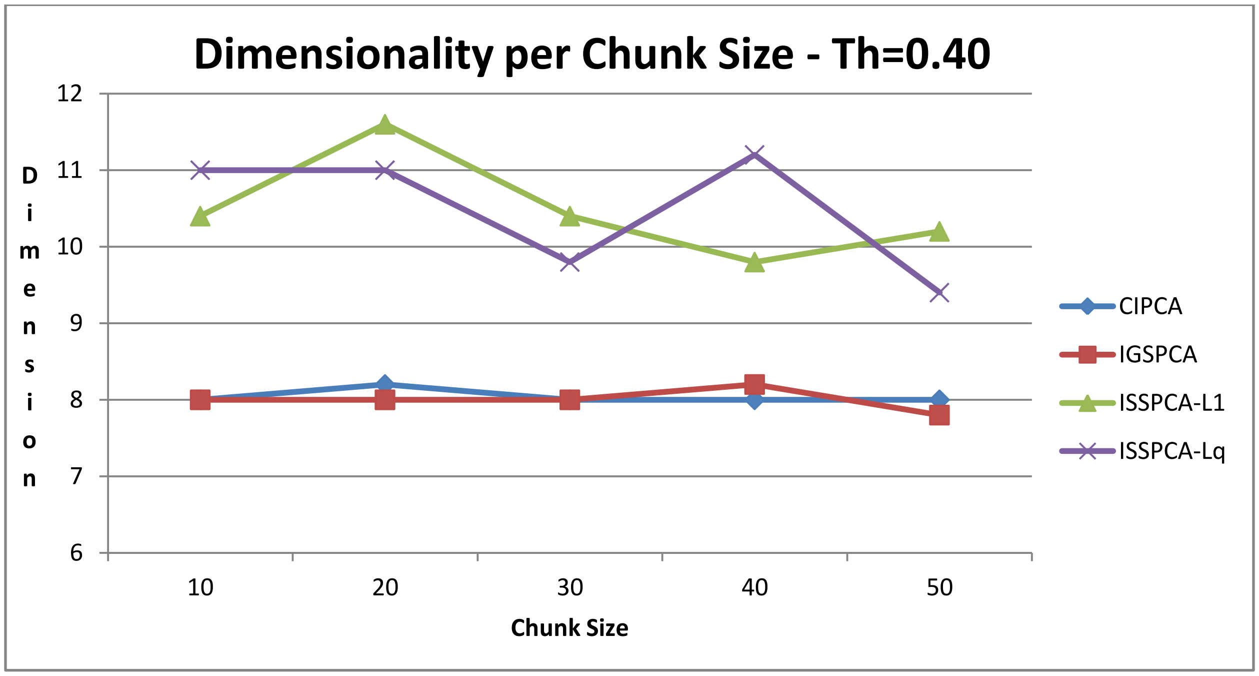

Figure 26. WaveFormNoise_ConceptShift: Impact of chunk size on dimensionality, θ = 0.4.

Table 26 and Figure 27 show the classification accuracy of the reduced datasets at the completion of the learning for various values of training sample chunk sizes. The classification accuracy for CIPCA and IGSPCA using the WaveFormNoise_ConceptShift dataset are 80.29 ± 0.79 and 80.2 ± 0.32 respectively. The classification accuracy for

CIPCA and IGSPCA using the WaveFormNoise dataset are 80.35 ± 0.83 and 80.2 ± 0.32 respectively. There is no significant impact with introducing a concept shift to the WaveFormNoise dataset .

Table 26. WaveFormNoise_ConceptShift: Impact of Chunk Size on Classification Accuracy,
$\theta = 0.4$

Chunk Size	Algorithm (%)			
	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
10	81.08	80.08	61.92	62.1
20	79.94	80.18	62.26	60.88
30	80.44	79.88	61.98	61.88
40	80.06	80.14	62.46	62.74
50	79.50	80.52	62.74	66.24

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

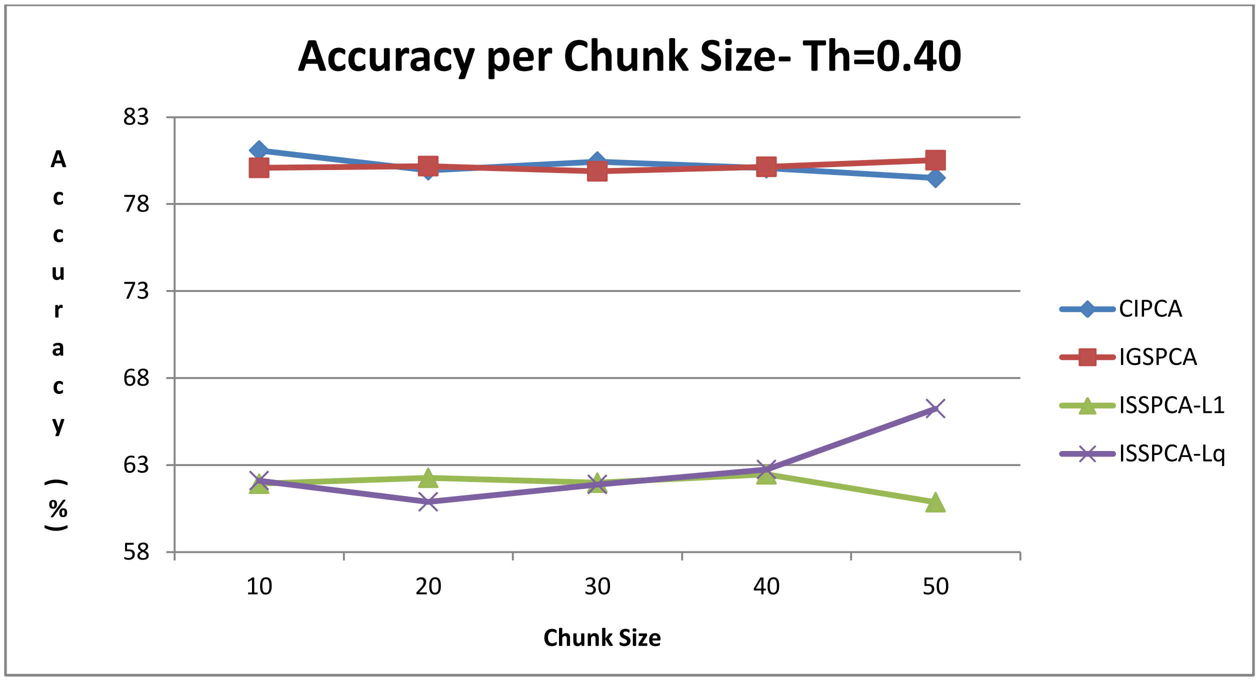

Figure 27. WaveFormNoise_ConceptShift: Impact of chunk size on classification accuracy, $\theta = 0.4$.

Table 27 and Figure 28 show the learning time to generate the reduced subsets at the completion of the learning for various values of training sample chunk sizes. The learning time required by IGSPCA was slightly greater than that required by CIPCA. CIPCA learning time was 0.24 ± 0.08 sec. IGSPCA learning time was 1.64 ± 0.75 sec. The learning time using the original WaveFormNoise dataset was 0.215 ± 0.055 sec and 1.62 ± 0.71 sec for CIPCA and IGSPCA respectively.

Table 27. WaveFormNoise_ConceptShift: Impact of Chunk Size on Learning Time, θ = 0.4

Chunk Size	Algorithm (Seconds)			
	CIPCA	IGSPCA	ISSPCA-L1	ISSPCA-Lq
10	0.32	2.39	22.46	26.22
20	0.21	1.35	13.06	14.5
30	0.16	1.12	8.27	8.14
40	0.18	1.09	6.29	6.36
50	0.19	0.89	5.22	4.92

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

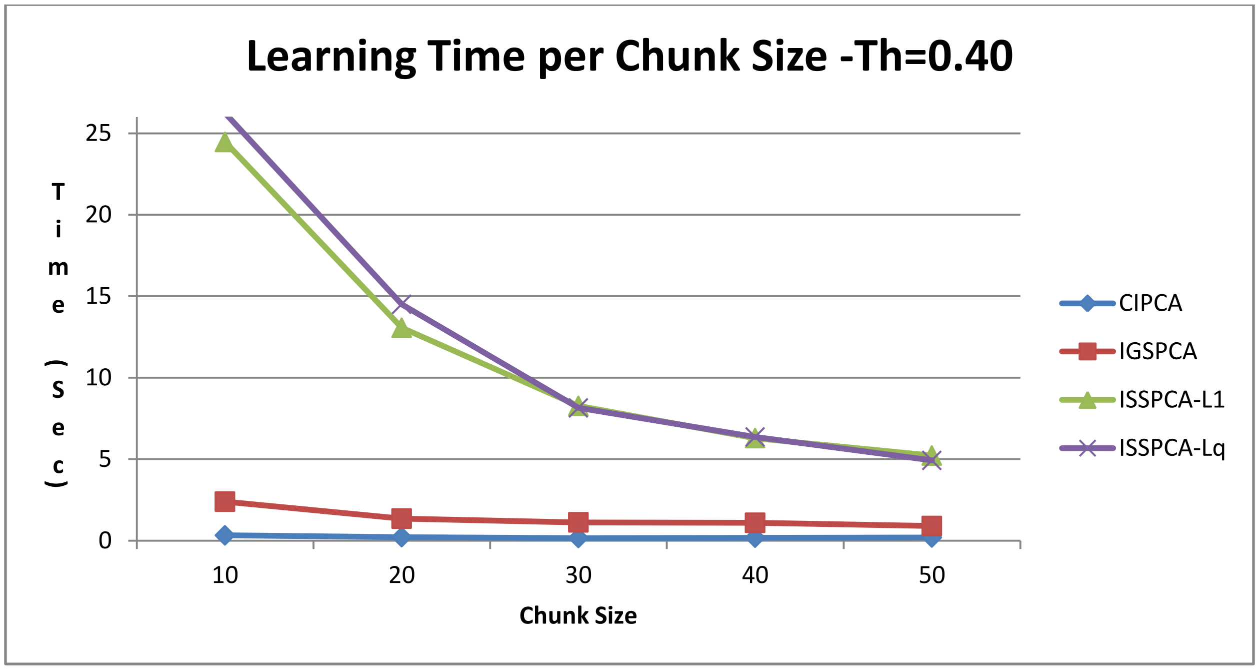

Figure 28. WaveFormNoise_ConceptShift: Impact of chunk size on learning time, θ = 0.4.

Summary of Results

Multiple experiments were conducted on the WaveFormNoise_ConceptShift dataset using all four incremental algorithms (CIPCA, IGSPCA, ISSPCA-L1, and ISSPCA-Lq). No significant differences were found using either the WaveFormNoise_ConceptShift dataset or the WaveFormNoise dataset. Results are compiled in the following table.

Table 28. WaveFormNoise_ConceptShift: Performance Comparison

	WaveFormNoise dataset			**WaveFormNoise_ConceptShift dataset**		
	Dimensionality Reduction	Classification Accuracy (%)	Learning Time (Sec)	Dimensionality Reduction	Classification Accuracy (%)	Learning Time (Sec)
CIPCA	8.1 ± 0.1	80.35 ± 0.83	0.215 ± 0.055	8.1 ± 0.1	80.29 ± 0.79	0.24 ± 0.08
IGSPCA	8 ± 0.2	80.2 ± 0.32	1.62 ± 0.71	8 ± 0.2	80.2 ± 0.32	1.64 ± 0.75
ISSPCA-L1	10.8 ± 0.8	61.45 ± 2.17	13.63 ± 9.16	10.7 ± 0.9	61.66 ± 0.8	14.84 ± 9.62
ISSPCA-Lq	10.7 ± 0.9	63.77 ± 1.57	14.87 ± 10.1	10.3 ± 0.9	63.56 ± 2.68	15.57 ± 10.65

Note. CIPCA = Chunk Incremental Principal Component Analysis, IGSPCA = Incremental Global Sparse Principal Component Analysis, ISSPCA = Incremental Structured Sparse Principal Component Analysis.

Chapter 5

Conclusion

Operational information systems such as web traffic, face recognition, sensor measurements, surveillance, and network intrusion detection systems continuously generate large amounts of data to be analyzed in real time for pattern discovery (Akhtar, 2011). Existing approaches store the whole data off-line before analysis in a batch mode (Hebrail, 2008).

Incremental principal component analysis (PCA) approaches have been proposed with the expectation to achieve the dimensionality reduction of data streams. However, PCA finds principal components as combinations of all the original variables, rendering their interpretation difficult. Computing principal components with maximum variance of the dataset while combining few variables improves the interpretability and analysis of the data. Sparse PCA produces principal components with sparse loadings, modeled as a linear combination of the subset’s original attributes (Zou et al., 2006).

This research was subdivided into two major stages. First, a previously proposed incremental features extraction algorithm CIPCA (Ozawa et al., 2008) was implemented to process intrusion detection data streams. This stage was followed by the proposition of novel incremental feature extraction methods based on two sparse principal component analysis techniques (Jenatton et al., 2010; Journee et al., 2010). The proposed incremental sparse PCA approaches extracted features from data streams in real time and found the minimum fraction of the original data sufficient to provide the maximum insight about the application under consideration. The proposed approaches were tested using four datasets: DARPA KDD Cup, Poker-Hand, WaveForm, and WaveFormNoise.

An important contribution of this dissertation is the development of methods able to dynamically extract optimal subset of data elements sufficient to gain insights from massive data streams and take appropriate actions. IGSPCA and ISSPCA were applied to the network data streams and efficiently reduced the data dimensionality to 8 and 10 respectively ($\theta = 0.5$ and $L = 500$), without negatively impacting the classification accuracies. CIPCA's performance was in the same range. The advantage of the proposed IGSPCA and ISSPCA was finding principal components with maximum variance of the dataset by combining few variables and improving the interpretability and analysis of the data. CIPCA found principal components as linear combinations of all the original variables. Another contribution of this dissertation is the capability of IGSPCA and ISSPCA to incrementally process data streams with concept drift and concept shift. Experiments using WaveForm and WaveFormNoise datasets confirmed these properties. Future research should include repeating the experiments in this dissertation using exclusively identified sparse variables of each dataset and analyzing the results. The framework proposed in this dissertation can be expanded to incorporate future sparse PCA algorithms for comparative studies and improvements evaluation. Moreover, this dissertation used the decision tree classifier for one-pass incremental learning. More efficient classifiers (SVM, BayesNet, NaiveBayes, etc.) can be added to the framework for future comparative studies and experiments.

Summary

Information systems continuously generate large amounts of continuous streams of data to be analyzed in real time. Existing data extraction approaches store the whole data off-line for analysis in a batch mode (Hebrail, 2008). Batch processing of static datasets impacts the processing speed and requires large memory capacity (Chandrika & Kumar, 2011). New algorithms are needed to extract optimal fraction of data elements and update data patterns need as new streams arrive. Incremental principal component analysis (PCA) approaches have been proposed for dimensionality reduction of data streams. PCA are linear combinations of all the original variables. Sparse PCA produces principal components with sparse loadings modeled as a linear combination of few original attributes. This research aimed to find a method that dynamically extracts subset of data elements to obtain insights from massive data streams.

Building on the one-pass CIPCA algorithm (Ozawa et al., 2004), this dissertation presents two incremental sparse PCA approaches to reduce the dimensionality of the input data stream: ISSPCA (incremental structure sparse principal component analysis) and IGSPCA (incremental global power for sparse principal component analysis), leveraging the structured sparse principal component analysis technique (Jenatton et al., 2010) and the generalized power method for sparse principal component analysis approach (Journee et al., 2010) respectively. CIPCA reduces the dimensionality of the input data stream in chunks using PCA. The data in the next chunk is used to construct the feature space. Sparse PCA reduces dataset via a linear combination of few original variables.

ISSPCA incrementally reduces the input data stream in chunks using structured sparse PCA (Jenatton et al., 2010). IGSPCA uses generalized power method for sparse PCA (Journee et al., 2010) and incrementally extracts sparse dominant principal components.

Comparison of results obtained in this dissertation using CIPCA, IGSPCA and ISSPCA presented in this research showed that IGSPCA are mostly at par or outperformed CIPCA. ISSPCA performance improved with larger chunk sizes. It was also observed that IGSPCA reduced data using the least number of original variables. In addition, the approaches presented in this dissertation were capable of handling concept drift and concept shift.

References

Aboalsamh, H. A., Mathkour, H. I., Assassa, G. M., & Mursi, M. F. M. (2009). Face recognition using incremental principal components analysis. *Proceedings of the International Conference on Computing, Engineering and Information*, 39-43. doi:10.1109/ICC.2009.56

Agrafiotis, D. K. (2003). Stochastic proximity embedding. *Proceedings of the Journal of Computational Chemistry*, *24*(10), 1,215-1,221. doi:10.1002/jcc.10234

Ahmad, F. K., Norwawi, N. M., Deris, S., & Othman, N. H. (2008). A review of feature selection techniques via gene expression profiles. *Proceedings of the International Symposium on Information Technology*, *2*, 1-7. doi:10.1109/ITSIM.2008.4631678

Akhtar, N. (2011). Statistical data analysis of continuous streams using streams DSMS. *Proceedings of the International Journal of Database Management Systems*, *3*(2), 89-99. doi:10.5121/ijdms.2011.3206

Baudat, G., & Anouar, F. (2000). Generalized discriminant analysis using a kernel approach. *Journal on Neural Computation*, *12*(10), 2,385-2,404. doi:10.1162/089976600300014980

Belkin, M., & Niyogi, P. (2002). Laplacian eigenmaps and spectral techniques for embedding and clustering. *Journal on Advances in Neural Information Processing Systems*, *14*, 585-591. doi:10.1162/089976603321780317

Bifet, A., & Gavaldà, R. (2006). Kalman filters and adaptive windows for learning in data streams. *Proceedings of the International Conference on Discovery Science*, *9*, 29-40. doi:10.1007/11893318_7

Bifet, A., & Gavaldà, R. (2007, April 26-28). Learning from time-changing data with adaptive windowing. *Proceedings of the Seventh SIAM International Conference on Data Mining*. doi:10.1137/1.9781611972771.42

Breiman, L., Friedman, J.-H., Olshen, R.-A., & Stone, C.-J. (1984). *Classification and regression trees*. Belmont, CA: Wadsworth International Group.

Cai, D., He, J., & Han, J. (2007). Efficient kernel discriminant analysis via spectral regression. *Proceedings of the IEEE International Conference on Data Mining*, *7*, 427-432. doi;10.1109/icdm.2007.88

Chan, H. P., Wei, D., Helvie, M. A., Sahiner, B., Adler, D. D., Goodsitt, M. M., & Petrick, N. (1995). Computer-aided classification of mammographic masses and normal tissue: Linear discriminant analysis in texture feature space. *Journal of Physics in Medicine and Biology*, *40*(5), 857-876.

Chandrika, J., & Kumar, K. R. A. (2011). A novel conceptual framework for mining high speed data streams. *Proceedings of International Conference on Business, Engineering and Industrial Applications (ICBEIA)*, 211-215. doi:10.1109/ICBEIA.2011.5994246

Chang, H., Yeung, D.-Y., & Xiong, Y. (2004). Super-resolution through neighbor embedding. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *1*, 275-282. doi:10.1109/CVPR.2004.1315043

Cox, T., & Cox, M. (1994). *Multidimensional scaling*. London, England: Chapman & Hall.

Dagher, I. (2010). Incremental PCA-LDA algorithm. *Proceedings of the IEEE International Conference on Computational Intelligence for Measurement Systems and Applications*, 97-101. doi:10.1109/CIMSA.2010.5611752

Das, S. (2001). Filters, wrappers and boosting-based hybrid for feature selection. *Proceedings of the International Conference on Machine Learning*, *18*, 74-81.doi:10.1.1.124.5264

D'Aspremont, A., El Ghaoui, L., Jordan, M. I., & Lanckriet, G. R. G. (2007). A direct formulation of sparse PCA using semidefinite programming. *Proceedings of the SIAM Review*, *49*(3), 434-448. doi:10.1137/050645506

Datar, M., Gionis, A., Indyk, P., & Motwani, R. (2002). Maintaining stream statistics over sliding windows. *Proceedings of the SIAM Journal on Computing*, *31*(6), 1,794-1,813. doi:10.1137/s0097539701398363

Ding, M., Tian, Z., & Xu, H. (2009). Adaptive kernel principal analysis for online feature extraction. *Proceedings of the World Academy of Science, Engineering and Technology*, *35*, 288-293. doi:10.1.1.192.9495

Domingos, P., & Hulten, G. (2000). Mining high speed data streams. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, *6*, 71-80. doi:10.1145/347090.347107

Donoho, D. L., & Grimes, C. (2005). Hessian eigenmaps: New locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences*, *102*(21), 7,426-7,431. doi:10.1.1.3.673

Duraiswami, R., & Raykar, V. C. (2005). The manifolds of spatial hearing. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, *3*, 285-288. doi:10.1.1.75.3784

Fan, W. (2004). Systematic data selection to mine concept-drifting data streams. *Proceedings of the ACM SIGKDD International Conference on Knowledge discovering and data mining, 10*, 128-137.

Firouzi, B. B., Niknam, T., & Nayeripour, M. (2008). A new evolutionary algorithm for cluster analysis. *Proceedings of World Academy of Science, Engineering and Technology, 36*, 584-588. doi:10.1.1.193.3820

Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics, 7*, 179-188. doi: 10.1111/j.1469-1809.1936.tb02137.x

Fukunaga, K. (1990). *Introduction to statistical pattern recognition*. San Diego, CA: Academic Press.

Gama, J., Rocha, R., & Medas, P. (2003). Accurate decision trees for mining high-speed data streams. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 9*, 532-528. doi:10.1145/956750.956813

Ghassabeh, Y. A., & Moghaddam, H. A. (2007). A new incremental optimal feature extraction method for on-line applications. *Lecture Notes in Computer Science, 4633*, 399-410. doi: 10.1007/978-3-540-74260-9_36

Grbovic, M., Dance, C. R., & Vucetic, S. (2012). Sparse principal component analysis with constraints. *Proceedings of the AAAI Conference on Artificial Intelligence, 26*, 935-941. Retrieved from https://www.aaai.org/ocs/index.php/AAAI/AAAI12/paper/viewFile/5109/5503

Guha, S., & Koudas, N. (2002). Approximating a data stream for querying and estimation: Algorithms and performance evaluation. *Proceedings of the International Conference on Data Engineering*, 567-576. doi:10.1109/icde.2002.994775

Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research, 3*, 1,157-1,182. doi:10.1.1.3.8934

Haeb-Umbach, R., & Ney, H. (1992). Linear discriminant analysis for improved large vocabulary continuous speech recognition. *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, 1*, 13-16. doi:10.1109/icassp.1992.225984

Hall, P., Marshall, D., & Martin, R. R. (1998). Incremental eigenanalysis for classification. *Proceedings of the British Conference on Machine Vision, 1*, 286-295. doi:10.1.1.40.4801

He, Y., Monteiro, R. D. C., & Park, H. (2011). An efficient algorithm for sparse PCA. *Proceedings of the SIAM International Conference on Data Mining*. Retrieved from http://fodava.gatech.edu/files/reports/FODAVA-10-26.pdf

Hebrail, G. (2008). Data stream management and mining. In F. Fogelman-Soulie, D. Perrotta, J. Piskorski, & R. Steinberger (Eds.), *Mining massive data sets for security* (pp. 89-102). Amsterdam, Netherlands: IOS Press.

Hein, M., & Buhler, T. (2010). An inverse power method for nonlinear eigenproblems with applications in 1-spectral clustering and sparse PCA. *Proceedings of Advances in Neural Information Processing Systems*, *23*. Retrieved from http://arxiv.org/abs/1012.0774

Hinton, G. E., & Roweis, S. T. (2002). Stochastic neighbor embedding. *Journal in Advances in Neural Information Processing Systems*, *15*, 833-840. Retrieved from https://www.cs.nyu.edu/~roweis/papers/sne_final.pdf

Hoffmann, H. (2007). Kernel PCA for novelty detection. *Journal of Pattern Recognition*, *40*(3), 863-874. doi:10.1016/j.patcog.2006.07.009

Hossain, M. A., Pickering, M., & Jia, X. (2011). Unsupervised feature reduction based on a mutual information measure for hyperspectral image classification. *Proceedings of the Australian Space Science Conference*, *11*, 1720-1723. doi:10.1109 /IGARSS.2011.6049567

Hotelling, H. (1933). Analysis of a complex of statistically variables into principal components. *Journal of Educational Psychology*, *24*(6), 417-441. doi:10.1037/h0071325

Huber, R., Ramoser, K., Mayer, K., Penz, H., & Rubik, M. (2005). Classification of coins using an eigenspace approach. *Pattern Recognition Letters*, *26*(1), 61-75. doi:10.1016/j.patrec.2004.09.006

Ikonomovska, E., Gorgevik, D., & Loskovska S. (2007). A survey of stream data mining. *Proceedings of National Conference with International Participation*, *8*, 16-22. Jenatton, R., Obozinski, G., & Bach, F. (2010). Structured sparse principal component analysis. *Proceedings of the International Conference on Artificial Intelligence and Statistics*, *13*(9). Retrieved from http://arxiv.org/pdf/0909.1440.pdf

Journee, M., Nesterov, Y., Richtarik, P., & Sepulchre, R. (2010). Generalized power method for sparse principal component analysis. *Journal of Machine Learning Research*, *11*, 517-553. Retrieved from http://www.jmlr.org/papers/volume11/journee10a/journee10a.pdf

Kholghi, M., & Keyvanpour, M. (2011). An analytical framework for data stream mining techniques based on challenges and requirements. *International Journal of Engineering Science and Technology*, *3*(3), 2,507-2,513. Retrieved from http://arxiv.org/ftp/arxiv/papers/1105/1105.1950.pdf

Khushaba, R. N., Al-Ani, A., & Al-Jumaily, A. (2007). Swarm intelligence based dimensionality reduction for myoelectric control. *Proceedings of the International Conference on Intelligent Sensors, Sensor Networks and Information, 3*, 577-582. doi:10.1109/ISSNIP.2007.4496907

Kim, K. I., Jung, K., & Kim, H. J. (2002). Face recognition using kernel principal component analysis. *Proceedings of the IEEE Conference on Signal Processing Letters*, *9*(2), 40-42. doi:10.1109/97.991133

Kim, T.–K., Stenger, B., Kittler, J., & Cipolla, R. (2011). Incremental linear discriminant analysis using sufficient spanning sets and its applications. *Proceedings of the International Journal of Computer Vision*, *91*(2), 216-232. doi: 10.1007/s11263-010-0381-3

Kohavi, R., & John, G. H. (1997). Wrapper for feature subset selection. *Proceedings of the Conference on Artificial Intelligence, 97*(1), 273-324. doi:10.1016/ S0004-3702(97)00043-X

Kohavi, R., & Sommerfield, D. (1995). Feature subset selection using the wrapper model: Over fitting and dynamic search space topology. *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, *1*, 192-197. doi:10.1.1.40.4640

Kompella, V. R., Luciw, M., & Schmidhuber, J. (2011). Incremental slow feature analysis: Adaptive and episodic learning from high-dimensional input streams. *Neural Computation*, *24*(11), 2,994-3,024. Retrieved from http://arxiv.org/abs/1112.2113

Kruskal, J. B. (1964). Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Journal of Psychometrika*, *29*(1), 1-27. doi:10.1007/BF02289565

Lafon, S., & Lee, A. B. (2006). Diffusion maps and coarse-graining: A unified framework for dimensionality reduction, graph partitioning, and data set parameterization. *Proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence*, *28*(9), 1,393-1,403. doi:10.1109/TPAMI.2006.184

Levy, A., & Lindenbaum, M. (2000). Sequential Karhunen-Loeve basis extraction and its application to images. *Proceedings of the IEEE Transactions on Image Processing*, *9*(8), 1,371-1,374. doi:10.1109/83.855432

Lima, A., Zen, H., Nankaku, Y., Miyajima, C., Tokuda, K., & Kitamura, T. (2004). On the use of kernel PCA for feature extraction in speech recognition. *Proceedings of the IEICE Transactions on Information Systems*, *E87-D(12*), 2,802-2,811. Retrieved from http://www.academia.edu/1823923/On_the_use_of_kernel_ PCA_for_feature_extraction_in_speech_recognition

Lippmann, J., Haines, J. W., Fried, D. J., Korba, J., & Das, K. (2000). The 1999 DARPA off-line intrusion detection evaluation. *Proceedings of the International Journal of Computer and Telecommunications Networking*, *34*(4), 579-595. doi:10.1016/S1389-1286(00)00139-0

Lui, H., & Yu, L. (2005). Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on Knowledge and Data Engineering*, *17*(4), 491-502. doi:10.1109/TKDE.2005.66

Michelakos, I., Papageorgiou, E., & Vasilakopoulos, M. (2010a). Hybrid classification algorithm evaluated on medical data. *Proceedings of the IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises*, 98-103. doi:10.1109/WETICE.2010.22

Michelakos, I., Papageorgiou, E., & Vasilakopoulos, M. (2010b). A Study of cAnt-Miner2 parameters using medical data sets. *Proceedings of the IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises*, *19*, 119-121. doi:10.1109/wetice.2010.52

Morshedlou, H., & Barforoush, A.A. (2009). A new history based method to handle the recurring concept shifts in data streams. Proceedings of the International Scholarly and Scientific Research and Innovation, *3*(10), 891-896.

Mundra, P. A., & Rajapaske, J. C. (2010). SVM-RFE with MRMR filter for gene selection. *Proceedings of the IEEE Transactions on NanoBioscience*, *9*(1), 31-37. doi:10.1109/TNB.2009.2035284

Nadler, B., Lafon, S., Coifman, R. R., & Kevrekidis, I. G. (2006). Diffusion maps, spectral clustering and the reaction coordinate of dynamical systems. *Journal on Applied and Computational Harmonic Analysis: Special Issue on Diffusion Maps and Wavelets*, *21*, 113-127. doi:10.1016/j.acha.2005.07.004

Nziga, J.-P. (2011). Minimal dataset for network intrusion detection systems via dimensionality reduction. *Proceedings of the IEEE International Conference on Digital Information Management*, *6*, 168-173. doi:10.1109/ICDIM.2011.6093368

Nziga, J.-P., & Cannady, J. (2012). Minimal dataset for network intrusion detection systems via MID-PCA: A hybrid approach. *Proceedings of the IEEE International Conference on Intelligent Systems*, *6*(1), 453-460.

Ohta, R., & Ozawa, S. (2009). An incremental learning algorithm of recursive Fisher linear discriminant. *Proceedings of the International Joint Conference on Neural Networks*, 2,310-2,315. doi:10.1109/IJCNN.2009.5178963

Otero, F. E., Freitas, A. A., & Johnson, C. G. (2008). cAnt-Miner: An ant colony classification algorithm to cope with continuous attributes. *Proceedings of the International Conference on Ant Colony Optimization and Swarm Intelligence*, *6*, 48-59. doi:10.1007/978-3-540-87527-7_5

Ozawa, S., Pang, S., & Kasabov, N. (2004). A modified incremental principal component analysis for on-line learning of feature space and classifier. *Proceedings of the Pacific Rim International Conferences on Artificial Intelligence: Trends in Artificial Intelligence,* 231-240. doi:10.1.1.109.7398

Ozawa, S., Pang, S., & Kasabov, N. (2008). Incremental learning of chunk data for online pattern classification systems. *Proceedings of the IEEE Transactions on Neural Network*, *19*(6), 1,061-1,074. doi:10.1109/tnn.2007.2000059

Ozawa, S., Pang, S., & Kasabov, N. (2010). Online feature extraction for evolving intelligent systems. In A. Plamen, P. D. Filev, & N. Kasabov (Eds.), *Evolving intelligent systems: Methodology and applications* (pp. 151-172). doi: 10.1002/9780470569962.ch7

Ozawa, S., Takeuchi, Y., & Abe, S. (2010). A fast incremental kernel principal component analysis for online feature extraction. *Lecture Notes in Computer Science, 6230,* 487-497. doi:10.1007/978-3-642-15246-7_45

Pang, S., Ozawa, S., & Kasabov, N. (2005). Incremental linear discriminant analysis for classification of data streams. *Proceedings of the IEEE Transactions on Systems, Man, and Cybernetics, Part B*, *35*(5), 905-914. doi:10.1109/TSMCB.2005.847744

Parpinelli, R. S., Lopes, H. S., & Freitas, A. A. (2002). Ant-Miner is an ant colony optimization algorithm for classification task. *Proceedings of the IEEE Transactions on Evolutionary Computation*, *6*(4), 321-332. doi:10.1007/978-3-540-87527-7_5

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, *2*, 559-572. Retrieved from http://pbil.univ-lyon1.fr/R/pearson1901.pdf

Posadas, A. M., Vidal, F., de Miguel, F., Alguacil, G., Peña, J., Ibañez, J. M., & Morales, J. (1993). Spatial-temporal analysis of a seismic series using the principal components method: The Antequera Series, Spain, 1989. *Journal of Geophysical Research*, *98*(B2), 1,923–1,932. doi:10.1029/92JB02297

Ross, D. A., Lim, J., Lin, R.–S., & Yang, M.–H. (2008). Incremental learning for robust visual tracking. *Proceedings of the International Journal of Computer Vision*, *77*(1-3), 125-141. doi:10.1007/s11263-007-0075-7

Roweis, S. T., & Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Journal of Science*, *290*(5000), 2,323-2,326. doi:10.1.1.111.3313

Schoelkopf, B., Smola, A. J., & Mueller, K. R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Journal on Neural Computation*, *10*(5), 1,299-1,319. doi:10.1162/089976698300017467

Shawe-Taylor, J., & Christianini, N. (2004). *Kernel methods for pattern analysis*. Cambridge, England: Cambridge University Press.

Tagaris, G. A., Richter, W., Kim, S. G., Pellizzer, G., Andersen, P., Ugurbil, K., & Georgopoulos, A. P. (1998). Functional magnetic resonance imaging of mental rotation and memory scanning: A multidimensional scaling analysis of brain activation patterns. *Journal of Brain Research*, *26*(2-3), 106-112. doi:10.1016/s0165-0173(97)00060-x

Takeuchi, Y., Ozawa, S., & Abe, S. (2007). An efficient incremental kernel principal component analysis for online feature selection. *Proceedings of the International Joint Conference on Neural Networks,* 2,346-2,351. doi:10.1109/IJCNN. 2007.4371325

Tenenbaum, J. B. (1998). Mapping a manifold of perceptual observations. *Proceedings of the International Conference in Advances in Neural Information Processing Systems*, *10*, 682-688. doi:10.1.1.137.7601

Tokumoto, T., & Ozawa, S. (2011). A fast incremental kernel principal component analysis for learning stream of data chunks. *Proceedings of the International Joint Conference on Neural Networks*, 2,881-2,888. doi:10.1109/IJCNN.2011.6033599

Torkkola, K. (2001). Linear discriminant analysis in document classification. *Proceedings of the IEEE ICDM Workshop on Text Mining,* 800-806. doi: 10.1.1.106.6309

Turk, M. A., & Pentland, A. P. (1991). Face recognition using eigen-faces. *Proceedings of Computer Vision and Pattern Recognition*, 586-591. doi: 10.1109/CVPR.1991. 139758

van der Maaten, L. J. P, Postma, E. O. & van den Herik, H. J. (2009). *Dimensionality reduction: A comparative review* [Tilburg University Technical Report]. doi:10.1.1.112.5472

Venkatarajan, M. S., & Braun, W. (2004). New quantitative descriptors of amino acids based on multidimensional scaling of a large number of physical-chemical properties. *Journal of Molecular Modeling*, *7*(12), 445-453. doi:10.1007/s00894-001-0058-5

Wang, H., Fan, W., Yu, P.-S., & Han, J. (2003). Mining concept-drifting data streams using ensemble classifiers. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 9*, 226-235.

Weinberger, K. Q., Sha, F., Zhu, Q., & Saul, L. K. (2007). *Graph Laplacian regularization for large-scale semidefinite programming***.** Cambridge, MA: MIT Press.

Weng, J., Zhang, Y., & Hwang, W. S. (2003). Candid covariance-free incremental principal component analysis. *Proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence*, *25*(8), 1034-1040. doi:10.1109/ TPAMI.2003.1217609

Xiang, C., Fan, X. A., & Lee, T. H. (2006). Face recognition using recursive Fisher linear discriminant. *Proceedings of the IEEE Transactions on Image Processing, 15*(8), 2,097-2,105. doi:10.1109/TIP.2006.875225

Yan, H.–B., & Liu, Y.–S. (2012). Image retrieval in data stream using principle component analysis. *Proceedings of the International Conference on Consumer Electronics, Communications and Networks, 2,* 2,634-2,637. doi: 10.1109/CECNet.2012.6201899

Yu, L., & Lui, H. (2003). Efficiently handling feature redundancy in high-dimensional data. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 9*, 685-690. doi: 10.1.1.129.5544

Zhao, H., Yuen, P. C., & Kwok, J. T. (2006). Incremental principal component analysis and its application for face recognition. *Proceedings of the IEEE Transactions on Systems, Man and Cybernetics* (Part B), *36*(4), 873-886. doi: 10.1109/ TSMCB.2006.870645

Zhang, C.–K., & Hu, H. (2005). Feature selection using the hybrid of ant colony optimization and mutual information for the forecaster. *Proceedings of the International Conference on Machine Learning and Cybernetics, 3*, 1728-1,732. doi:10.1109/ICMLC.2005.1527223

Zhang, Y., Ding, C. & Li, T. (2007). A two-stage gene selection algorithm by combining reliefF and mRMR. *Proceedings of the IEEE International Conference on Bioinformatics and Bioengineering*, 7, 164-171. doi: 10.1186/1471-2164-9-S2-S27

Zhang, Y., & El Ghaoui, L. E. (2011). Large-scale sparse principal component analysis with application to test data. *Proceedings of the Advances in Neural Information Processing Systems*. Retrieved from http://www.eecs.berkeley.edu/~elghaoui /Pubs/nips2011.zhang.elghaoui.pdf

Zhang, Z., & Zha, H. (2002). Principal manifolds and nonlinear dimensionality reduction via local tangent space alignment. *SIAM Journal of Scientific Computing, 26*(1), 313-338. doi: 10.1.1.4.3693

Zhu, Z., Ong, Y., & Dash, M. (2007). Wrapper-filter feature selection algorithm using a memetic framework. *IEEE Transactions on Systems, Man, and Cybernetics Society, 37*(1), 70-76. doi:10.1109/TSMCB.2006.883267

Zou, H., Hastie, T., & Tibshirani, R. (2006). Sparse principal component analysis. *Journal of Computational & Graphical Statistics, 15*(2), 265-286. doi:10.1198/ 106186006X113430