Feature Subset Selection and Feature Ranking for Multivariate Time Series

Hyunjin Yoon, Kiyoung Yang, and Cyrus Shahabi, *Member*, *IEEE*

Abstract—Feature subset selection (FSS) is a known technique to preprocess the data before performing any data mining tasks, e.g., classification and clustering. FSS provides both cost-effective predictors and a better understanding of the underlying process that generated the data. We propose a family of novel *unsupervised* methods for feature subset selection from Multivariate Time Series (MTS) based on Common Principal Component Analysis, termed $CLeVer$. Traditional FSS techniques, such as Recursive Feature Elimination (RFE) and Fisher Criterion (FC), have been applied to MTS data sets, e.g., Brain Computer Interface (BCI) data sets. However, these techniques may lose the correlation information among features, while our proposed techniques utilize the properties of the principal component analysis to retain that information. In order to evaluate the effectiveness of our selected subset of features, we employ classification as the target data mining task. Our exhaustive experiments show that $CLeVer$ outperforms RFE, FC, and random selection by up to a factor of two in terms of the classification accuracy, while taking up to 2 orders of magnitude less processing time than RFE and FC.

Index Terms—Data mining, feature evaluation and selection, feature extraction or construction, time series analysis, feature representation.

1 Introduction

Feature subset selection (FSS) is one of the techniques to preprocess the data before we perform any data mining tasks, e.g., classification and clustering. FSS is to identify a subset of original features/variables[1] from a given data set while removing irrelevant and/or redundant features [2]. The objectives of FSS are [1]:

- to improve the prediction performance of the predictors,
- to provide faster and more cost-effective predictors, and
- to provide a better understanding of the underlying process that generated the data.

FSS methods choose a subset of the original features to be used for the subsequent processes. Hence, only the data generated from those features need to be collected. The differences between feature *extraction* and FSS are:

- Feature subset selection maintains information on the original variables while this information is usually lost when feature extraction is used.
- After identifying the subset of the original features, only these features are measured and collected ignoring all the other features. However, feature extraction in general still requires the measuring of all the variables.

1. The features originally measured by the input devices, e.g., sensors, are called *variables* and the features constructed from these variables *features* as in [1]. However, the terms *variable* and *feature* are interchangeably used throughout this paper when there is no ambiguity.

- *The authors are with the Department of Computer Science, University of Southern California, Los Angeles, CA 90089.*
E-mail: {hjy, kiyoungy, shahabi}@usc.edu.

Manuscript received 20 Nov. 2004; revised 22 Mar. 2005; accepted 30 Mar. 2005; published online 19 July 2005.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDESI-0457-1104.

A time series is a series of observations,

$$x_i(t); [i = 1, \cdots, n; t = 1, \cdots, m],$$

made sequentially through time, where i indexes the measurements made at each time point t [3]. It is called a univariate time series when n is equal to 1 and a multivariate time series (MTS) when n is equal to or greater than 2.

MTS data sets are common in various fields, such as in multimedia, medicine, and finance. For example, in multimedia, Cybergloves used in the Human and Computer Interface applications have around 20 sensors, each of which generates $50 \sim 100$ values in a second [4], [5]. In [6], 22 markers are spread over the human body to measure the movements of human parts while people are walking. The data set collected is then used to recognize and identify the person by how he or she walks. In medicine, Electro Encephalogram (EEG) from 64 electrodes placed on the scalp are measured to examine the correlation of genetic predisposition to alcoholism [7]. In the Neuro-rehabilitation domain, kinematics data sets generated from sensors are collected and analyzed to evaluate the functional behavior (i.e., the movement of upper extremity) of poststroke patients [8].

The size of an MTS data set can become very large quickly. For example, the EEG data set utilizes tens of electrodes and the sampling rate is 256Hz. In order to process MTS data sets efficiently, it is inevitable to preprocess the data sets to obtain the relevant subset of features which will be subsequently employed for further processing. In the field of Brain Computer Interfaces (BCIs), the selection of relevant features is considered absolutely necessary for the EEG data set, since the neural correlates are not known in such detail [9]. Identifying optimal and valid features that differentiate the poststroke patients from the healthy subjects is also challenging in the Neuro-rehabilitation applications.

An MTS item is naturally represented in an $m \times n$ matrix, where m is the number of observations and n is the number of *variables*, e.g., sensors. However, the state of the art feature subset selection techniques, such as Recursive Feature elimination (RFE) [10] and Zero-Norm Optimization [11], require each item to be represented in one row. Consequently, to utilize these techniques on MTS data sets, each MTS item needs to be first transformed into one row or column vector. For example, in [9], where an EEG data set with 39 channels is used, an autoregressive (AR) model of order 3 [12] is utilized to represent each channel. Hence, each 39 channel EEG time series is transformed into a 117 dimensional vector. However, if each channel of EEG is considered separately, we will lose the correlation information among the variables.

In this paper, we propose a family of three novel *unsupervised* feature subset selection methods for multivariate time series (MTS) based on common principal component analysis (CPCA) named $CLeVer$.[2] It consists of $CLeVer$-*Rank*, $CLeVer$-*Cluster*, and $CLeVer$-*Hybrid*. In order to perform FSS on an MTS data set, $CLeVer$ first performs PCA on each MTS item to obtain the *loadings*, i.e., the direction cosines of principal components. Note that the loadings describe how much each variable contributes to each of the principal components (see Section 3.1 for a brief review of PCA). We then obtain the descriptive common principal components (DCPCs) which *agree most closely* with the principal components of all the MTS items [13]. The intuition to use the PCs and the DCPCs as a basis for *variable* subset selection is that they keep the most compact overview of the MTS items in a dramatically reduced space while retaining both the correspondence to the original variables and the correlation among the variables. The DCPC loadings show how much each variable contributes to each of the DCPCs. Based on the DCPC loadings, we devise three members of the $CLeVer$ family. $CLeVer$-*Rank* ranks each variable based on its contribution to the DCPCs. Subsequently, FSS can be done by choosing the top K ranked variables. Though the feature ranking technique is simple and scalable [1], the feature subsets selected by the feature ranking technique, however, may include *redundant* features. $CLeVer$-*Cluster* clusters the DCPC loadings to identify the variables that have similar contributions to each of the DCPCs. For each cluster, we obtain the *centroid* variable, eliminating all the *similar* variables within the cluster. These centroid variables form the selected subset of variables. $CLeVer$-*Hybrid* is similar to $CLeVer$-*Cluster*. The difference is that $CLeVer$-*Hybrid* ranks the variables within each cluster based on their contribution to the DCPCs. Subsequently, the top ranked variables from all the clusters will be part of the selected subset of variables. Our experiments show that, in most cases, the classification accuracy of the variable subsets obtained by the $CLeVer$ family is up to 100 percent better than the feature subsets selected by other methods, such as Recursive Feature Elimination (RFE) and Fisher Criterion (FC) [14]. Moreover, the $CLeVer$ approaches, which are considered *unsupervised feature subset selection methods* [15], take up to 2 orders of magnitude less time than RFE, which is a *wrapper* method [16], and FC.

2. $CLeVer$ is an abbreviation of descriptive **C**ommon principal component **L**oading-based **V**ariable subset selection method.

The remainder of this paper is organized as follows: In Section 2, the related work is discussed. Section 3 provides the background. Our proposed methods are described in Section 4, which is followed by the experiments and results in Section 5. Conclusions and future work are presented in Section 6.

2 Related Work

There are, in general, two approaches to feature subset selection (FSS), i.e., *wrapper* approach and *filter* approach. If any mining task, e.g., classification, is involved in the process of selecting the subset of features, it is called a *wrapper approach*; otherwise, it is a *filter approach* [17]. The former often performs better than the latter, but requires significantly more processing time [15]. For a detailed introduction to FSS, please refer to [1], [16].

When class labels are available, *supervised* FSS techniques can be utilized; otherwise, *unsupervised* FSS techniques [15] should be employed. The unsupervised FSS first computes the similarity between features and then removes the redundancy therein for the subsequent data mining tasks. Hence, in general, the unsupervised FSS incorporates partitioning or clustering of the original feature set. Consequently, each partition or cluster is represented by a single representative feature to form a subset of features.

Recall that our objective is to perform the FSS for the multivariate time series (MTS) data set to obtain compact form of its representation while preserving its characteristics. The most related fields would be the ones that deal with Electroencephalogram (EEG) data sets, where each EEG item is represented as a matrix. In particular, extensive research has been conducted on Electroencephalogram (EEG) data sets in the field of Brain Computer Interfaces (BCIs). The EEG data set is collected using multiple electrodes placed on the scalp. The sampling rate is hundreds of Hertz. The selection of relevant features is considered absolutely necessary for the EEG data set, since the neural correlates are not known in such detail [9].

Hidden Markov Model (HMM) and variants of Coupled HMM (CHMM) have been applied to the EEG data set from UCI KDD Archive [18] for classification in [19]. The best accuracies obtained are 90.0 ± 0.0 percent using Distance Coupled HMM (DCHMM) and 90.5 ± 5.6 percent using Multivariate HMM for the EEG-1 data set. The accuracy of 78.5 ± 8.0 percent is obtained using Multivariate HMM for the EEG-2 data set. However, the EEG-1 data set contains only 20 measurements for two subjects from two arbitrary electrodes (F4 and P8). EEG-2 data set contains 20 measurements from the same two electrodes for each subject. Moreover, it is not clear how the two subjects out of 122 subjects and the two electrodes out of 64 are chosen.

Guyon et al. proposed Recursive Feature Elimination (RFE) using Support Vector Machine in [10], whose procedure can be briefly described as follows: 1) train the classifier, 2) compute the ranking criterion for all features, and 3) remove the feature with lowest ranking criterion. This procedure is then repeated until the required number of features remain. Though RFE has been shown to perform well, RFE cannot be applied directly to MTS data sets; each MTS item should be represented as one row vector.

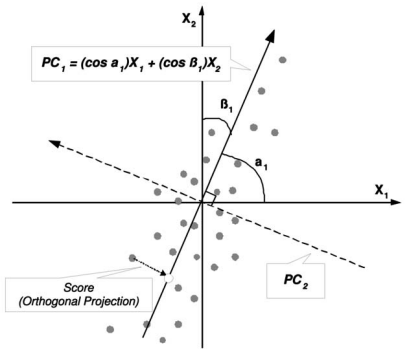


Fig. 1. Two principal components obtained for one multivariate data with two variables x_1 and x_2 measured on 30 observations.

Extending RFE for MTS data sets, FSS is performed on the EEG data set with 39 channels in [9]. In order to apply RFE to MTS data sets, each EEG item is first broken into 39 separate channels and, for each channel, autoregressive (AR) fit of order 3 [12] is computed. Subsequently, each channel is thus represented by three autoregressive coefficients. RFE is then performed on this transformed data set to select significant channels. As shown in Section 5.5, this wrapper method is very time consuming. Moreover, by considering the channels separately, they lose the correlation information among channels, which would, in turn, result in poor generalization.

The most related work to our paper is the PCA-based FSS method, called Principal Feature Analysis (PFA) [20]. The data sets used is the set of features extracted from image databases, which is represented in a feature vector. PCA is then performed on a matrix that contains all the feature vectors, after which the features with similar principal component loadings are grouped together. Again, PFA cannot reduce the amount of data to be collected and requires all the raw data to perform the feature extraction.

3 Background

In this section, we briefly review principal component analysis and common principal component analysis. For more details, please refer to [21], [13].

3.1 Principal Component Analysis

Principal Component Analysis (PCA) has been widely used for multivariate data analysis and dimension reduction [21]. Intuitively, PCA is a process to identify the directions, i.e., principal components (PCs), where the variances of scores (orthogonal projections of data points onto the directions) are maximized and the residual errors are minimized assuming the least square distance. These directions, in nonincreasing order, explain the variations underlying original data points; the first principal component describes the maximum variation, the subsequent direction explains the next maximum variance and so on.

Fig. 1 illustrates principal components obtained on a very simple (though unrealistic) multivariate data with only two variables (x_1, x_2) measured on 30 observations. Geometrically, the principal component is a linear transformation of original variables and the coefficients defining this transformation are called *loadings*. For example, the first principal component (PC1) in Fig. 1 can be described as a linear combination of original variables x_1 and x_2, and the two coefficients (loadings) defining PC1 are the cosines of the angles between PC1 and variables x_1 and x_2, respectively. The loadings are thus interpreted as the contributions or weights on determining the directions.

In practice, PCA is performed by applying Singular Value Decomposition (SVD) to either a covariance matrix or a correlation matrix of an MTS item depending on the data set. That is, when a covariance matrix A is decomposed by SVD, i.e., $A = U\Lambda U^T$, a matrix U contains the variables' loadings for the principal components, and a matrix Λ has the corresponding variances along the diagonal [21].

In the data mining literature, PCA has been used mainly for dimension reduction by projecting multidimensional data set onto the first few principal components. However, PCA is a multivariate statistical analysis for one multivariate data item where all the input data considered for the analysis should be arranged in a single matrix. In our application domain dealing with a set of MTS items, this one-sample method should be generalized to a multisample method [22]. One approach to tackling this generalization problem is to obtain a set of PCs that are common across all MTS items [13], [22], which will be briefly described in the next section.

3.2 Common Principal Component

Common Principal Component Analysis (CPCA) is a generalization of PCA for $N(\geq 2)$ multivariate data items, where the ith data item, $(i = 1, \ldots, N)$, is represented in an $m_i \times n$ matrix. That is, all the data items have the same number of variables, i.e., n, while each data item may have different number of observations. CPCA is based on the assumption that there exists a common subspace across all multivariate data items and this subspace should be spanned by the orthogonal components. Various efforts have been made to find the common components defining this common subspace [13], [22], [23], [24].

One approach, proposed in [13], obtained the Common Principal Components (CPC) by bisecting the angles between their principal components after each multivariate data item undergoes PCA. That is, each of multivariate data items is first described by its first p principal components. Then, the CPCs are obtained by successively bisecting the angles between their ith $(i = 1, \ldots, p)$ principal components. These CPCs define the common subspace that *agrees most closely* with every subspace of the multivariate data items. Fig. 2 gives a plot of two multivariate data items A and B. Let A and B be denoted as a swarm of white points and black points, respectively, and have the same number of variables, i.e., x_1 and x_2, measured on 20 and 30 observations, respectively. The first principal component of each data set is obtained using PCA and the common component is obtained by bisecting the angle between those two principal components. We will refer to this CPC model as *Descriptive Common Principal Component* (DCPC) throughout the paper.

As compared to the CPC model proposed in [22] that produces *unordered* common principal components, these DCPCs are orthogonal to each other and nonincreasingly

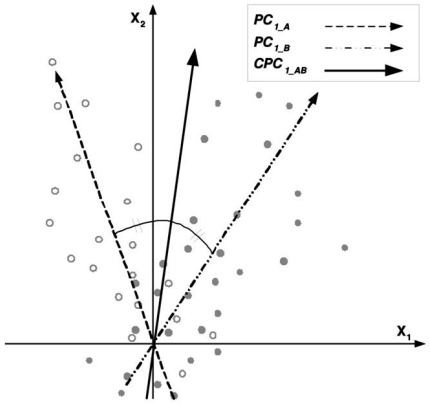


Fig. 2. A common principal component of two multivariate data items

ordered. That is, the first DCPC represents the common direction along which every multivariate data item has its maximum variance. In addition, the (i,j)th loading value of the DCPC matrix can be interpreted as the contribution of the jth original variable with respect to the ith common principal component, which is analogous to the way the principal component loadings are interpreted. Finally, the CPC model described in [22] is only really applicable to the covariance-based PCA and not to the more frequently encountered correlation-based analysis [21].

4 $CLeVer$ Algorithms

We propose a family of novel *unsupervised* variable subset selection methods for multivariate time series (MTS) based on common principal component analysis (CPCA), named $CLeVer$. $CLeVer$ involves three phases: 1) principal components (PCs) computation per MTS item, 2) descriptive common principal components (DCPCs) computation across all principal components, and 3) variable subset selection using DCPC loadings of variables. We devise three variations of $CLeVer$ during the third phase by using different techniques. The three variations are $CLeVer$-$Rank$, $CLeVer$-$Cluster$, and $CLeVer$-$Hybrid$, based on variable ranking, K-means clustering, and both, respectively. Fig. 3 illustrates the entire process of $CLeVer$.

The algorithms developed for $CLeVer$, including the discussions on the practical issues, are presented in the following sections. Table 1 lists the notations used in the remainder of this paper, if not specified otherwise.

4.1 PC and DCPC Computations

The first and second phases of $CLeVer$ are incorporated into Algorithm 1. It first obtains PCs per MTS item and then DCPCs across them consecutively. The required input to Algorithm 1 is a whole set of MTS items. The principal components of an MTS item is obtained by applying Singular Value Decomposition (SVD) to its correlation matrix in Line 4. Even though there are n PCs for each item, only the first $p(< n)$ PCs, which are adequate for the purpose of representing each MTS item, are taken into consideration. In general, p is determined based on the percent ratio of the sum of the variances explained by the first p PCs to the total variance underlying the original MTS item, which ranges between 70 percent and 90 percent [21]. Algorithm 1 takes the sum of variation, i.e., the threshold to determine p, as an input. That is, for each input MTS item, p is determined to be the minimum value such that the total variation explained by its first p PCs exceeds the provided threshold δ for the first time (Line 8). Since the MTS items can have different values for p, p is finally determined as their maximum value in Line 10.

Each MTS item is now represented as a $p \times n$ matrix whose rows are its first p PCs and columns represent variables. Note that each column of this matrix has a one-to-one correspondence to the original variable at the same position and describes the contributions of the corresponding variable to each of the p PCs.

Let a $p \times n$ PC matrix of each MTS item be denoted as L_i $(i = 1, \ldots, N)$. Then, the DCPCs that *agree most closely* with all N sets of p PCs are successively defined by the eigenvectors of the matrix $H = \sum_{i=1}^{N} L_i^T L_i$:

$$SVD(H) = SVD\left(\sum_{i=1}^{N} L_i^T L_i\right) = V \Lambda V^T, \tag{1}$$

where rows of V are eigenvectors of H and the first p of them define p $DCPC$s for N MTS items. Λ is a diagonal matrix whose diagonal elements are the eigenvalues of H and describe the total discrepancy between DCPCs and PCs. For example, the first eigenvalue implies the overall closeness of the first DCPC to the first PCs of every MTS item (for more details, please refer to [13]). This computation of DCPCs is captured by Lines 15 $\sim$ 16.

As is similar to the PC matrix of an MTS item, the rows of DCPC matrix are p DCPCs, the columns represent n variables, and its (i,j)th element is a DCPC loading of the jth variable to the ith DCPC. The correspondence to the original variables is still retained in the columns of DCPC matrix and each original variable is thus featured by its DCPC loadings. This DCPC matrix is the common base structure on which the three subsequent variable subset selection methods are performed.

Algorithm 1. $ComputeDCPC$: PC and DCPC Computations.
Require: an MTS data set with N items and δ {a predefined threshold}
1. $DCPC \leftarrow \emptyset$;
2. **for** i=1 to N **do**
3. $X \leftarrow$ the ith MTS item;
4. $[U, S, U^T] \leftarrow$ SVD(correlation matrix of X);
5. $loading(i) \leftarrow U$;
6. $variance \leftarrow diag(S)$;
7. $percentVar \leftarrow 100 \times (variance / \sum_{j=1}^{n} variance_j)$;
8. $p_i \leftarrow$ number of the first p $percentVar$ elements whose cumulative sum $\geq \delta$;
9. **end for**
10. $p \leftarrow max(p_1, p_2, \ldots, p_N)$;
11. **for** i=1 to N **do**
12. $L(i) \leftarrow$ the first p rows of $loading(i)$;
13. $H(i) \leftarrow H(i-1) + (L(i)^T \times L(i))$;
14. **end for**
15. $[V, S, V^T] \leftarrow$ SVD(H);
16. $DCPC \leftarrow$ first p rows of V;

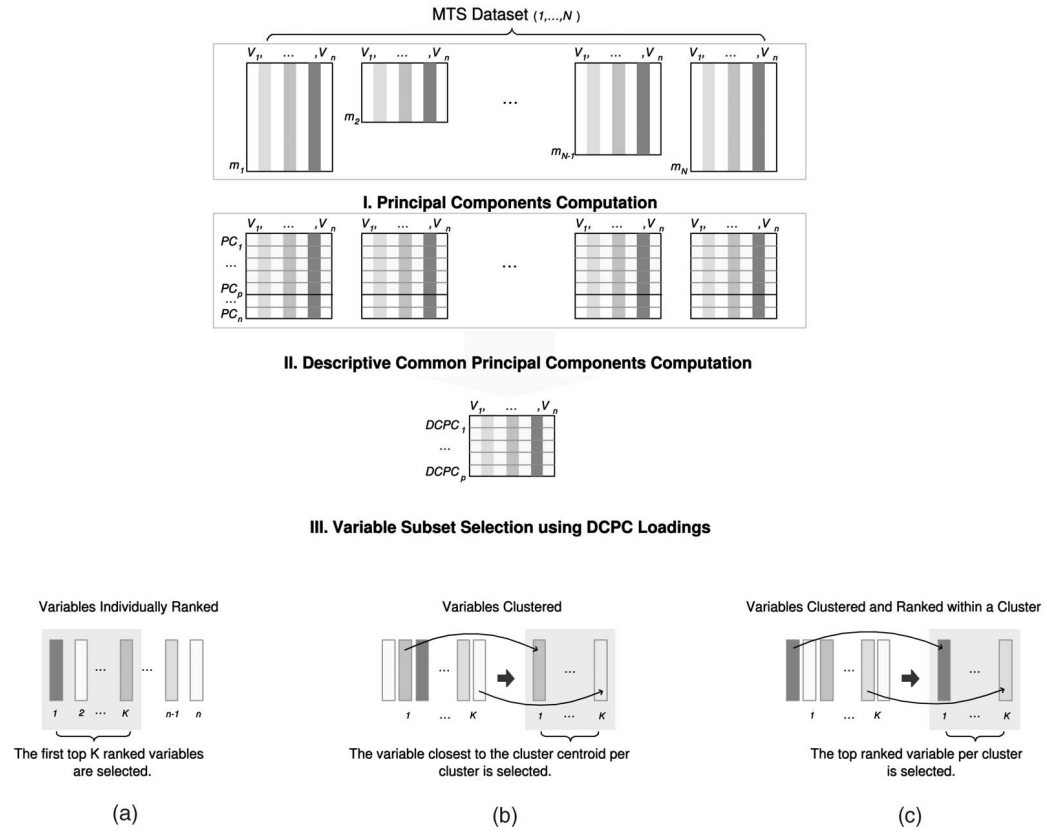


Fig. 3. The processes of $CLeVer$ family: (a) $CLeVer$-$Rank$ yields the ranks of variables based on each variable's DCPC loadings, (b) $CLeVer$-$Cluster$ selects the centroids of K clusters on DCPC loadings, and (c) $CLeVer$-$Hybrid$ chooses the top ranked variables within each of K clusters on DCPC loadings.

4.2 $CLeVer$-$Rank$

$CLeVer$-$Rank$, described in Algorithm 2, ranks each variable based on its contribution to the DCPCs. For a variable v_1, the score is given by the ℓ^2-norm ($|v_1|$) of its DCPC loading vector. Let one of the variables be denoted as a vector v_1 consisting of p DCPC loadings, that is, $v_1 = (l_1, l_2, \ldots, l_p)$. Then, the score of v_1 is defined by

$$|v_1| = \sqrt{l_1^2 + l_2^2 + \ldots + l_p^2}. \quad (2)$$

The intuition of using ℓ^2-norm to measure the contribution of each of variables is based on the observation that the stronger impact a variable has on the common principal components, the larger absolute DCPC loading value it has, as well as the further away it lies from the origin. Therefore, the norm describing the length of a vector in general can be used as a valid score in this context. In Lines $3 \sim 5$, the score of each variable, i.e., the ℓ^2-norm of DCPC loadings of each variable is computed after obtaining the descriptive common principal components in Line 2.

Subsequently, the scores are sorted in a nonincreasing order and both the scores and the variable IDs are retained for the subsequent processing in Line 6. Finally, the first K variables, i.e., the top variables in the ranked list, are simply selected to form the variable subset of size K in Line 7. This way, only *significant* variables with the highest scores in terms of their contribution to the common principal component model remain in the selection.

Algorithm 2. $CLeVer$-$Rank$.

Require: an MTS data set, K {the number of variables to select}, and δ {a predefined threshold}

1. $selected \leftarrow \emptyset$;
2. $DCPC \leftarrow computeDCPC$(MTS, δ);
3. **for** i=1 to n **do**
4. $score(i) \leftarrow$ norm($DCPC$ loadings of the ith variable);
5. **end for**
6. $[IDX, score] \leftarrow$ sort $score$ in a nonincreasing order;
7. $selected \leftarrow IDX(1..K)$;

TABLE 1
Notations Used in This Paper

Symbol	Definition
N	number of MTS items in an MTS data set
n	number of variables in an MTS item
K	number of clusters for K-means clustering or number of variables to select
p	number of PCs for each MTS item to be used for computing DCPCs

Many variable selection algorithms include variable ranking as a base method due to its simplicity, scalability, and good empirical success [1]. However, the ranking-based variable subset selection techniques suffer from such a common shortcoming that it may lead to the selection of *redundant* variables. That is, we may possibly achieve the same performance on the subsequent mining task even with a smaller number of variables if the selected subset of variables includes redundant variables. Furthermore, a variable that is not important by itself may prove essential for the subsequent mining task (e.g., classification) when taken with other variables.

4.3 *CLeVer-Cluster*

CLeVer-Cluster is proposed to tackle the above-mentioned limitations of ranking based variable selection. The principle of this approach is to select the complementary variables rather than individually scored variables. *CLeVer-Cluster* utilizes a clustering method to group the *similar* variables together and select the least redundant variables (see Fig. 3b *CLeVer-Cluster*). The intuition behind using the clustering technique for the variable selection is based on the observation that variables with similar pattern of loading values will be highly correlated and have high mutual information [20].

Algorithm 3. *CLeVer-Cluster*.
Require: an MTS data set, K {the number of clusters}, and δ {a predefined threshold}
1. $selected \leftarrow \emptyset$;
2. $DCPC \leftarrow computeDCPC$(MTS, δ);
3. **for** i=1 to 20 **do**
4. $[index(i), centroid(i)] \leftarrow K$ means($DCPC$ loadings, K);
5. **end for**
6. $[IDX, C] \leftarrow$ choose the best clustering result with the lowest within-cluster sums of point-to-centroid Euclidean distances;
7. **for** k=1 to K **do**
8. $selected(k) \leftarrow$ the closet variable to $C(k)$ within the kth cluster;
9. **end for**

Algorithm 3 takes an MTS data set and the number of clusters as its inputs and performs K-means clustering on the DCPC loadings obtained in Line 2. Since the K-means clustering method can reach the local minima, we iterate the K-means clustering 20 times (Lines 3 $\sim$ 5). Among the 20 clustering trials, the one with the minimum sum of Euclidean Distances between each cluster centroid and the column vectors within the cluster is chosen as the best clustering result in Line 6. Finally, the least redundant variable subset for the given K is obtained by extracting K column vectors closest to the centroid of each cluster and identifying their corresponding original variables (Lines 7 $\sim$ 9).

4.4 *CLeVer-Hybrid*

CLeVer-Hybrid utilizes both the ranking part of *CLeVer-Rank* and the clustering part of *CLeVer-Cluster* for variable subset selection. After performing the same clustering as *CLeVer-Cluster*, instead of finding the median variable for each cluster, we rank the variables within each cluster employing the same method used for *CLeVer-Rank* and then select the variable with the highest score to be the representative variable for that cluster (see Fig. 3c *CLeVer-Hybrid*). *CLeVer-Cluster* and *CLeVer-Hybrid* effectively perform variable subset selection by eliminating variables that contribute to the common principal components *similarly*. As compared to *CLeVer-Rank*, one of the drawbacks of these clustering-based approaches would be that the selected variable subset is not *cumulative*. That is, when we change the number of clusters from, e.g., three to four, the variables selected with three clusters are not necessarily the subset of the variables selected with four clusters. Moreover, if we change the number of clusters, i.e., the number of selected variables, we have to repeat (the clustering part of) them. In this context, *CLeVer-Rank* can decide the variable subsets to be used in the subsequent processing, i.e., classification, with more flexibility than *CLeVer-Cluster* and *CLeVer-Hybrid*.

5 Performance Evaluation

Since the focus of our paper is to select a subset of original variables (not the extracted features that in general lose the correspondence to the variables), one of the best ways to evaluate the selected variables would be to have them validated by the domain experts, e.g., to see whether the selected subset corresponds to the experts' observations or not. However, the domain knowledge is not always available for every MTS data set and, also, the experts are not always in agreement.

Hence, we first show the effectiveness of *CLeVer* in terms of classification performance via Support Vector Machine (SVM). The reason SVM is chosen for the predictor is that it determines a classifier which minimizes not only the empirical risk (i.e., the training error) but the confidence interval (i.e., the generalization or test error). For the detailed description of SVM, please refer to [25]. We conducted several experiments on three real-world data sets. After obtaining a subset of variables using *CLeVer-Rank*, *CLeVer-Cluster*, and *CLeVer-Hybrid*, we performed classification using only that selected subset of variables. Support Vector Machine (SVM) with linear kernel is employed as a classifier employing leave-one-out cross validation for evaluation. For the multiclass data set, the one-against-one method is adopted for the multiclass SVM. All the algorithms of *CLeVer* for the experiments are implemented in $Matlab^{TM}$. SVM classification is completed with *LIBSVM* [26] on a machine with Pentium IV 3.2GHz CPU and 3 GB of RAM.

5.1 Data Sets

The **HumanGait data set** [6] has been used for identifying a person by recognizing his/her gait at a distance. In order to capture the gait data, a 12-camera VICON system was utilized with 22 reflective markers attached to each subject. For each reflective marker, 3D position, i.e., x, y, and z, are acquired at 120Hz, generating 66 values at each timestamp. 15 subjects, which are the labels assigned to the data set, participated in the experiments and were required to walk at four different speeds, nine times for each speed. The total

TABLE 2
Summary of Data Sets Used in the Experiments

	HumanGait	BCAR	BCI MPI
# of variables	66	11	39
average length	133	454	1280
# of labels	15	2	2
# of items per label	36	44/34	1000
total # of items	540	78	2000

number of data items is 540 ($15 \times 4 \times 9$) and the average length is 133.

The Motor Behavior and Rehabilitation Laboratory, University of Southern California, collected the **Brain and Behavior Correlates of Arm Rehabilitation (BCAR) kinematics data set** [8] to study the effect of Constraint-Induced (CI) physical therapy on the poststroke patients' control of their upper extremities. The functional specific task performed by subjects was a continuous three phase reach-grasp-place action; he/she was asked to reach to a target object, either a cylinder or a card, grasp it by posing the pronated forearm posture, place it into a designated hole, release it, and finally bring his/her hand back to the starting switch. Four control (i.e., healthy) subjects and three poststroke subjects experiencing a different level of impairment participated in the experiments. Five subjects of them repeated the task five times and the remaining two performed it seven times under two different conditions, i.e., for two different objects (cylinder and card). The performance of each trial was traced by six 6D electromagnetic Mini-Bird sensors attached on the index nail, thumb nail, dorsal hand, distal dorsal forearm, lateral mid upper arm, and shoulder, respectively. Each of them produced its 3D position (i.e., x, y, and z) and orientation (i.e., pitch, yaw, and roll) which were filtered using a 0-lag Butterworth low-pass filter with a 20Hz cut-off frequency at the rate of 120Hz. Unlike other data sets, the BCAR data set kept the record of 11 dependent features rather than 36 *raw* variables at each timestamp. They were defined by experts in advance and calculated from the raw variables by the device software provided with the sensors; some of them were just raw variables (e.g., wrist sensor X, Y, and Z coordination) while others were synthesized from raw variables (e.g., aperture was computed as tangential displacement of two sensors on thumb and index nail). Note that these 11 variables were considered as original variables throughout the experiments. The total number of data items per condition is 39 ($5 \times 5 + 2 \times 7$), i.e., 78 in total for the two conditions, and their average length is about 454 (i.e., about 3.78 seconds).

The **Brain Computer interface (BCI) data set** at the **Max Planck Institute (MPI)** [9] was gathered to examine the relationship between the brain activity and the motor imagery, i.e., the imagination of limb movements. Eight right handed male subjects participated in the experiments, out of which three subjects were filtered out after preanalysis. 39 electrodes were placed on the scalp to record the EEG signals at the rate of 256Hz. The total number of items is 2,000, i.e., 400 items per subject.

Table 2 shows the summary of the data sets used in the experiments.

5.2 Threshold Value δ

We first performed a set of experiments to study the sensitivity of our proposed techniques, the $CLeVer$ family, to the threshold value δ in Algorithm 1, which determines the number of PCs, p, to be obtained. In general, the minimum number of PCs whose corresponding eigenvalues account for 70 percent $\sim$ 90 percent of the total variance would be chosen [21]. Hence, we performed classification on the HumanGait data set to see how the classification accuracies of $CLeVer$ would change based on δ, varying δ from 70 percent to 90 percent increasing it by 5 percent each time. As recommended in [27], we employed stratified 10 fold cross validation on the HumanGait data set using LIBSVM [26] with linear kernel. For a detailed description on how to perform the classification using LIBSVM for the MTS data sets, please refer to Section 5.3.

The experiments have been conducted using $CLeVer$-$Cluster$, $CLeVer$-$Rank$, and $CLeVer$-$Hybrid$. Table 3 summarizes the classification accuracies using $CLeVer$-$Rank$. The other two methods yield similar results. It shows that $CLeVer$ is only sensitive to the threshold value δ when

TABLE 3
Classification Accuracy (%) Summary for Different Threshold Values δ on HumanGait Using $CLeVer$-$Rank$

	# of Selected Variables						
δ	10	20	30	40	50	60	66
70%	91.4815	99.0741	99.8148	99.8148	100.0000	99.2593	99.4444
75%	91.4815	99.0741	99.8148	100.0000	99.4444	99.4444	99.4444
80%	**91.2963**	**99.2593**	**100.0000**	**99.4444**	**99.8148**	**99.8148**	**99.6296**
85%	86.6667	98.3333	99.6296	99.8148	99.4444	99.8148	99.4444
90%	86.8519	97.5926	99.8148	99.6296	99.4444	99.4444	99.4444

the number of selected features is less than 20. However, when the number of selected features is greater than 20, the performance differences of $CLeVer$ across the specified range of δ values are almost indistinguishable.

In general, if we choose too low of a threshold value, the PCs do not reflect *sufficient* variances of the data set, while, if we choose too high of a threshold value, the PCs may include noise information of the data set as well. For the rest of the experiments, we choose the δ value to be 80 percent, which is the mean value of 70 percent and 90 percent.

5.3 Classification Performance

A Support Vector Machine (SVM) with linear kernel was adopted for the classifier to evaluate the classification performance of $CLeVer$ approaches. We performed leave-one-out cross validation (CV) for all the three data sets. For leave-one-out CV, one of the items in the data set is chosen to be the test data, while all the other items are employed as the training data. This is repeated N times and the average classification accuracy is reported.

Note that, for the MTS data set to be fed into SVM, each of the MTS items should be represented as a vector with the same dimension; we call this process *vectorization*. Some transformations are considered for the vectorization: 1) using either upper or lower triangle of its covariance or correlation matrix, 2) using Dynamic Time Warping (DTW) to make every MTS items equi-length,[3] and 3) using linear interpolation. We utilized the correlation matrix transformation for our experiments. Based on our previous work [30], where correlation matrix is used to compute the similarity between two MTS items, using correlation matrix either was comparable or outperformed DTW and linear interpolation in terms of precision/recall.

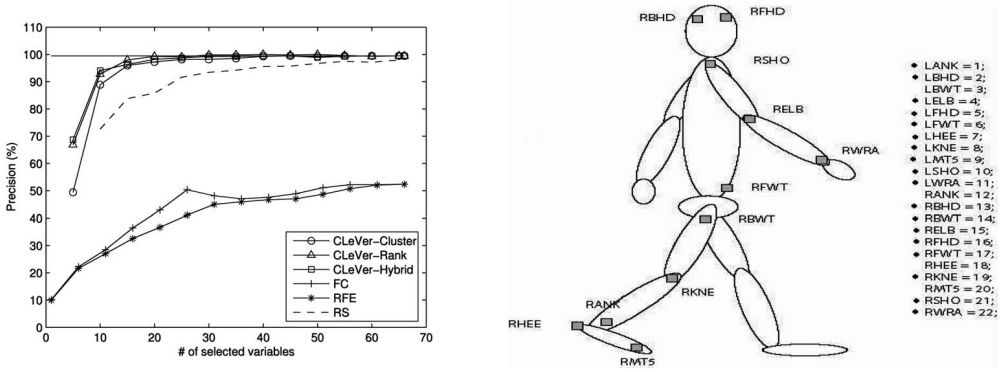
We subsequently compared the classification performance of $CLeVer$ approaches to those of exhaustive search selection, random selection, Recursive Feature Elimination [9], Fisher Criterion [14], and all variables selected.

- Exhaustive Search Selection (ESS): All possible combinations of variable subsets were tested to find the combinations which yield the best and the worst classification performance only when it was tractable. In addition, the corresponding classification precisions were averaged over all possible combinations given the number of selected variables. The ESS method was performed only on the BCAR data set due to the intractability of ESS for the large data sets. Note that ESS is not a practical approach in general, but it was done to generate ground-truth for evaluation when possible.
- Random Selection (RS): We randomly selected the target number of variables and then performed leave-one-out cross validation using SVM with linear kernel. Each random selection was repeated 10 times and the average performance is reported.
- Recursive Feature Elimination (RFE): As in [9], we utilized the forward-backward linear prediction [12] to obtain the autoregressive (AR) fit of each variable for order 3. That is, in order to *vectorize* an MTS item for SVM, each variable was represented with three coefficients and each MTS item was consequently represented as a row vector with $3 \times n$ values, where n was the number of variables. RFE was then employed to rank the n variables using the average of the three primal values $w's$ per variable. Linear kernel SVM was applied to train RFE for all the data sets, and the one-versus-one multiclass SVM is utilized for the HumanGait data set. For small data sets, i.e., BCAR and HumanGait, RFE within *The Spider* [31] was employed, while, for large data sets, i.e., BCI_MPI, one of the LIBSVM tools [26] was modified and utilized. Once the rank of variables was determined by RFE, the variable subset of size K was composed with the first K variables with the high rankings, after which leave-one-out cross validation was performed using the $3 \times K$ AR fit coefficients corresponding to these K variables selected.
- Fisher Criterion (FC): As is done for RFE, every MTS item was *vectorized* with three AR fit coefficients per variable. The Fisher Criterion implemented within *The Spider* [31] was then employed to rank the variables. As in [9], the implementation was slightly modified so that the average of three Fisher scores per variable was used to finally rank the original n variables. The subsequent variable subset selection and classification experiments were performed exactly the same way as RFE.

Fig. 4a presents the generalization performances of the selected subsets of variables on the HumanGait data set. The X axis is the number of selected subset of variables, i.e., the number of clusters K for the $CLeVer$ family, and the Y axis is the classification accuracy. The figure shows that a subset of 22 variables selected by $CLeVer\text{-}Rank$ out of 66 performs the same as the one using all the variables, which is more than 99.4 percent of accuracy. When 35 or more variables are selected, the performances of $CLeVer\text{-}Cluster$, $CLeVer\text{-}Rank$, and $CLeVer\text{-}Hybrid$ are more or less the same. The 22 variables selected by $CLeVer\text{-}Rank$ are from only 18 markers (marked with a filled circle in Fig. 4b) out of 22, which would mean that the values generated by the remaining four markers do not contribute much to the identification of the person. From this information, we may be able to better understand the characteristics of the human walking.

The performance using the subsets of variables obtained by RFE and Fisher Criterion is much worse than the ones using the $CLeVer$ approaches. It is even worse than random selection of variable subsets. Even when using all the variables, the classification accuracy is around 52 percent, which is worse than that of 10 variables selected by the $CLeVer$ approaches. Considering the fact that RFE on three AR coefficients performed well in [9], this may indicate that, for the HumanGait data set, the correlation information among variables is more important than for the BCI MPI data set. Hence, each variable should not be taken

3. Even though the main purpose of DTW is not to make MTS items equi-length, DTW has indeed been used to generate the sequences of the same length [28], [29]. In this paper, DTW is considered as a way of *vectorizing* an MTS item into one row vector, after which the MTS items will be fed into SVM for classification.



(a) (b)

Fig. 4. (a) HumanGait data set, Classification Evaluation, (b) 22 Markers are utilized to capture the human walking motion for the HumanGait data set [32]. The markers with a filled circle represent 18 markers from which the 22 variables are selected by $CLeVer$-$Rank$, which yields the same performance accuracy as the one using all the 66 variables.

out separately to compute the autoregressive coefficients, by which the correlation information would be lost. Note that, in [9], the order 3 for the autoregressive fit is identified after proper model selection experiments, which would mean that for the HumanGait data set, the order of the autoregressive fit should be determined, again, after comparing models of different orders. This shows that transforming an MTS item into a vector is not a trivial task.

Fig. 5 shows the classification performance of the selected variables on the BCAR data set under two different conditions: cylinder and card. In the cylinder (or card) condition, a cylinder (or a card) was used as the target object, while the pronated forearm posture was taken by a subject to perform the continuous reach-grasp-place task. BCAR is the simplest data set with 11 original variables and the number of MTS items for each condition is just 39 (17 out of 39 are performed by poststroke subjects). Hence, we applied the Exhaustive Search Selection (ESS) method to find all the possible variable combinations, for each of which we performed leave-one-out cross validation with the SVM linear classifier. It took about 27 minutes to complete the whole ESS experiments. The result of ESS shows that 100 percent classification accuracy can be achieved by no less than four variables out of 11 in both conditions. The dotted lines in Fig. 5 represent the best, the average, and the worst performance obtained by ESS, respectively, given the number of selected variables.

Figs. 5a and 5c illustrate the comparative result of $CLeVer$-$Rank$, $CLeVer$-$Cluster$, and $CLeVer$-$Hybrid$ for the cylinder and the card condition, respectively. All three members of the $CLeVer$ family show comparable performance when more than four variables are selected and the best performance of $CLeVer$ is far beyond the average accuracy of ESS in most cases. Besides, $CLeVer$ can achieve the same performance as the one using all 11 variables by using less number of variables, i.e., six and seven for the card and cylinder conditions, respectively. In particular, $CLeVer$-$Rank$ produces 100 percent accuracy with seven and eight variables for the cylinder and the card condition, respectively, which is even better than the one using all variables.

In Figs. 5b and 5d, the best performance produced by $CLeVer$ is compared to RFE and FC. As illustrated, both RFE and FC never beat $CLeVer$-$Rank$ and $CLeVer$-$Hybrid$ except when two variables are selected. These observations imply that $CLeVer$ never eliminates useful information in its variable selection on the BCAR data set, where synthesized features capturing the kinematics information are used for the original variables and the number of variables is relatively small.

For the BCI MPI data set, Fig. 6 shows the performance comparison. Note that this is the same data set used in [9], where they performed the feature subset selection per subject. However, our goal is to obtain the variable subsets across all the subjects, which would make the subset of variables selected by $CLeVer$ more applicable for subsequent data mining tasks. All the items from the five subjects were therefore considered for variable subset selection in our experiments. Moreover, while the regularization parameter $\mathcal{C}_s$ was estimated via 10 fold cross validation from the training data sets in [9], we used the default value, which is 1.

Fig. 6 depicts that when the number of selected variables is less than six, RFE performs better than the $CLeVer$ family and the FC technique. When the number of selected variables is greater than six, however, $CLeVer$-$Cluster$ performs far better than RFE, and when more than 21 variables are selected, $CLeVer$ family by far outperforms RFE and FC. The classification performance using all the variables is shown in solid horizontal line. Note again that, even using all the variables, the performance of RFE is worse than that of 11 variables selected by $CLeVer$-$Cluster$.

5.4 Redundancy Reduction

In order to evaluate the effectiveness of the selected feature subsets, other than the classification accuracy, the representation entropy [33] is employed to measure the amount of redundancy present in the selected variable subsets. The variable subset selection algorithms based on the variable ranking such as RFE, FC, and $CLeVer$-$Rank$ suffer from a common shortcoming that redundant variables may be selected. Besides, a variable that is not significant by itself but possibly essential in the subsequent mining tasks when taken with others may never be selected because variables

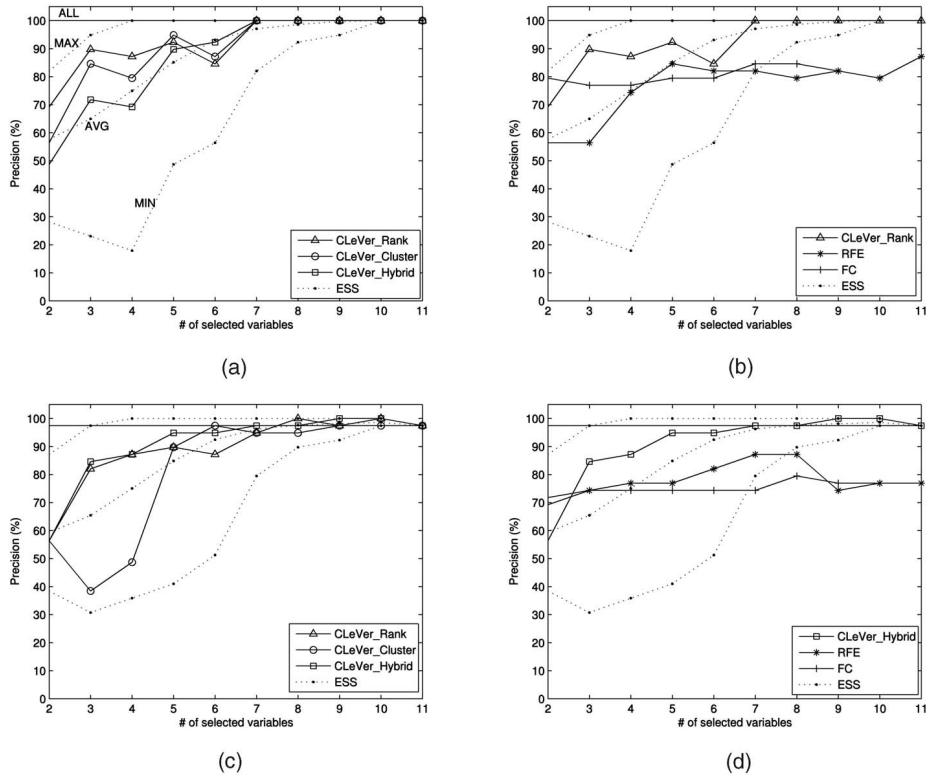


Fig. 5. BCAR data set, Classification Evaluation. (a) Cylinder: $CLeVer$. (b) Cylinder: $CLeVer$ versus RFE/FC. (c) Card: $CLeVer$. (d) Card: $CLeVer$ versus RFE/FC.

are individually scored in the ranking based methods. Therefore, the representation entropy would be a valid index to determine the effectiveness of the proposed clustering-based variable subset selection methods, i.e., $\boldsymbol{CLeVer}$-$Cluster$ and $\boldsymbol{CLeVer}$-$Hybrid$, in terms of redundancy reduction.

First, we describe the representation entropy measure proposed in [33]. Let the eigenvalues of the $K \times K$ covariance matrix of a selected feature set of size K be $\lambda_j, j = 1, \ldots, K$. Let $\tilde{\lambda}$ be defined as

$$\tilde{\lambda}_j = \frac{\lambda_j}{\sum_{j=1}^{K} \lambda_j}. \tag{3}$$

$\tilde{\lambda}_j$ has similar properties like probability, that is, $0 \leq \tilde{\lambda}_j \leq 1$ and $\sum_{j=1}^{K} \tilde{\lambda}_j = 1$. Then, the representation entropy (E_R) is defined as

$$E_R = -\sum_{j=1}^{K} \tilde{\lambda}_j log \tilde{\lambda}_j. \tag{4}$$

The representation entropy attains a minimum value when all the information is present along a single feature. That is, the other features selected are redundant. In contrast, E_R is maximum when information is equally distributed among all the features. That is, the least redundant subset of variables is selected. Hence, in our case, the higher the E_R, the better. We first compute the DCPCs as in Algorithm 1. For the λ in (3), the eigenvalues corresponding to the common principal components describing the common space across all MTS items only with the K selected variables are exploited. Then, the E_R value in (3), i.e., the representation entropy for the selected variable subset of size K, is obtained.

Table 4 summarizes the representation entropy for the three data sets used in the previous experiments. The representation entropy is computed when the number of selected variables is approximately half the number of all variables, i.e., when $K \approx n/2$.

As expected, in most cases, the clustering-based feature subset selection methods produce higher representation entropy than the ranking based feature subset selection methods. More specifically, $\boldsymbol{CLeVer}$-$Cluster$ and $\boldsymbol{CLeVer}$-$Hybrid$ produce consistently higher representation entropy than $\boldsymbol{CLeVer}$-$Rank$ across all three data sets. This indicates redundant variables are really eliminated during the process of K-means clustering and selecting a representative

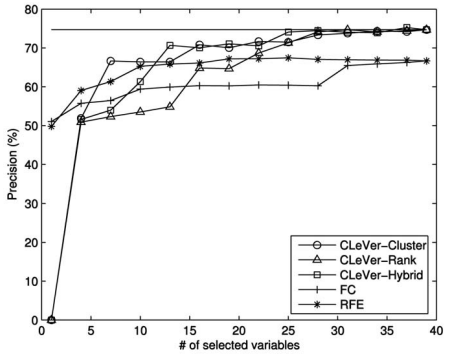


Fig. 6. BCI MPI data set, Classification Evaluation.

TABLE 4
Representation Entropy on Three Different Data Sets

Method	Dataset			
	HumanGait	BCAR (Cylinder)	BCAR (Card)	BCI MPI
	$n = 66, K = 33$	$n = 11, K = 6$		$n = 39, K = 19$
CLeVer-Rank	0.9479	1.4716	1.5827	2.6621
CLeVer-Cluster	1.0578	1.6756	1.6703	2.4311
CLeVer-Hybrid	1.0109	1.6474	1.6468	2.4399
RFE	2.0514	1.5537	1.4735	2.1342
FC	1.8142	1.7364	1.5472	2.0518

variable of each cluster. As compared to RFE and FC, $CLeVer$-*Cluster* and $CLeVer$-*Hybrid* attain lower presentation entropy than RFE and FC in two cases. Hence, further investigation is required to understand these unexpected results. Nonetheless, for the data sets where *unsupervised* variable selection is the only option due to the absence of class labels, this representation entropy would be one of the evaluation indexes to validate the selected variables.

5.5 Processing Time

Table 5 summarizes the processing time of the five feature subset selection methods employed for the experiments. The processing time for the $CLeVer$ family includes the time to perform Algorithm 1 and the time to perform each of the three variations while varying K from 2 to $n-1$ for each data set. For example, K changes from 2 to 65 for the HumanGait data set. The processing time for RFE and FC includes the time to transform the data to three Autoregressive fit coefficients and the time to perform the feature subset selection, i.e., to obtain the ranks of all the variables.

For $CLeVer$-*Rank*, as soon as Algorithm 1 is done, the ranks of all the variables are computed instantly. For example, it took only about 4 seconds to obtain the ranks of all the 66 variables of the HumanGait data set. However, for $CLeVer$-*Cluster* and $CLeVer$-*Hybrid*, the majority of time is spent on performing the K-means clustering. For example, it took more than 85 seconds to perform the K-means clustering, while it took less than four seconds to execute Algorithm 1 for the HumanGait data set. This is due to the fact that K-means clustering is repeated 20 times and the one with the minimum sum of Euclidean Distances between each cluster centroid and the column vectors within the cluster is chosen as the best clustering result in Line 7 of Algorithm 3.

For RFE and FC, the time to transform the data takes more than 95 percent of the total time. That is, it took 113 seconds to transform the HumanGait data set, while only a couple of seconds were required for the rest of FC. For the BCI MPI data set, whose average length is 1,280, it took around 7,600 seconds to transform the data set. Recall that the order of the autoregressive (AR) model, which is 3 in [9], has been decided after comparing models of different orders employing 10 fold cross validation, which would increase the total processing time even more, if the time to decide the AR order is considered.

Even after the transformation of the data set, a considerable amount of time is still required for RFE. This is because RFE is a *wrapper* feature selection method. That is, RFE utilizes the classifier within the feature selection procedure to select the *best* features which produce the *best* classification precision. $CLeVer$, on the other hand, is an *unsupervised* feature selection method [15]. That is, $CLeVer$ does not utilize the class label, nor, thusly, the classification accuracy. Intuitively, $CLeVer$ computes the similarities between variables and the importance of each variable based on how much each variable contributes to the common principal components, without using the class labels.

Since $CLeVer$ does not include the classification procedure, it takes less time to yield the feature subset selection than the *wrapper* methods. For example, for the BCI MPI data set, $CLeVer$ took less than 12 seconds to compute the DCPCs and around 35 seconds to perform the K-means clustering on the loadings of the DCPCs, while RFE took more than 7,800 seconds. Across the data sets, the $CLeVer$ approaches take up to 2 orders of magnitude less time than RFE, while performing better than RFE in classification accuracy by up to a factor of two.

TABLE 5
Comparison of Processing Time in Seconds for Different FSS Methods on Three Different Data Sets

Processing time (Seconds)	HumanGait	BCAR	BCI MPI
CLeVer-Rank	3.672	0.078	11.813
CLeVer-Cluster	88.594	0.610	46.281
CLeVer-Hybrid	91.547	0.617	46.328
RFE	862.174	9.039	7886.844
FC	113.907	6.469	7594.941

6 Conclusions and Future Work

In this paper, we proposed a family of three novel *unsupervised* feature subset selection methods for multivariate time series (MTS), based on common principal component analysis (CPCA), termed *descriptive Common principal component Loading based Variable subset selection method* ($CLeVer$). $CLeVer$ utilizes the properties of the principal components and the descriptive common principal components (DCPCs) to retain the correlation information among the variables. $CLeVer$-$Rank$ ranks the variables based on their contribution to the common principal components. Subsequently, the top K ranked variables are chosen. $CLeVer$-$Cluster$ and $CLeVer$-$Hybrid$ perform clustering on the loadings of the DCPCs to select a subset of variables. Our exhaustive experiments on three real-world data sets indicate that the family of $CLeVer$ techniques outperforms other feature selection methods, such as Recursive Feature Elimination (RFE), Fisher Criterion (FC), and random selection by up to a factor of two in terms of the classification accuracy. Moreover, $CLeVer$ takes up to 2 orders of magnitude less processing time than RFE and FC.

We intend to extend this research in two directions. First, we plan to extend this work to be able to estimate the optimal number of variables, i.e., the optimal number of clusters K using, e.g., the Gap statistic in [34]. We also plan to generalize this research and use k-way PCA [35] to perform PCA on a k-way array. Hence, all the MTS items are to be represented in a 3-way array, and this would require no transformation such as vectorization that may still result in information loss.

Acknowledgments

This research has been funded in part by US National Science Foundation (NSF) grants EEC-9529152 (IMSC ERC), IIS-0238560 (PECASE), and IIS-0307908, and unrestricted cash gifts from Microsoft. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the US National Science Foundation. The authors would like to thank Dr. Carolee Winstein and Jarugool Tretiluxana for providing the BCAR data set and valuable feedback and Thomas Navin Lal for providing the BCI MPI data set. The authors would also like to thank the anonymous reviewers for their valuable comments.

References

[1] I. Guyon and A. Elisseeff, "An Introduction to Variable and Feature Selection," *J. Machine Learning Research,* vol. 3, pp. 1157-1182, Mar. 2003.

[2] H. Liu, L. Yu, M. Dash, and H. Motoda, "Active Feature Selection Using Classes," *Proc. Pacific-Asia Conf. Knowledge Discovery and Data Mining,* 2003.

[3] A. Tucker, S. Swift, and X. Liu, "Variable Grouping in Multivariate Time Series Via Correlation," *IEEE Trans. Systems, Man, and Cybernetics B,* vol. 31, no. 2, 2001.

[4] M.W. Kadous, "Temporal Classification: Extending the Classification Paradigm to Multivariate Time Series," PhD dissertation, Univ. of New South Wales, 2002.

[5] C. Shahabi, "AIMS: An Immersidata Management System," *Proc. Very Large Data Bases Biennial Conf. Innovative Data Systems Research,* 2003.

[6] R. Tanawongsuwan and A.F. Bobick, "Performance Analysis of Time-Distance Gait Parameters under Different Speeds," *Proc. Fourth Int'l Conf. Audio and Video-Based Biometric Person Authentication,* June 2003.

[7] X.L. Zhang, H. Begleiter, B. Porjesz, W. Wang, and A. Litke, "Event Related Potentials during Object Recognition Tasks," *Brain Research Bull.,* vol. 38, no. 6, 1995.

[8] C. Winstein and J. Tretriluxana, "Motor Skill Learning after Rehabilitative Therapy: Kinematics of a Reach-Grasp Task," *Soc. Neuroscience,* Oct. 2004.

[9] T.N. Lal, M. Schröder, T. Hinterberger, J. Weston, M. Bogdan, N. Birbaumer, and B. Schölkopf, "Support Vector Channel Selection in BCI," *IEEE Trans. Biomedical Eng.,* vol. 51, no. 6, June 2004.

[10] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene Selection for Cancer Classification Using Support Vector Machines," *Machine Learning,* vol. 46, nos. 1-3, pp. 389-422, Jan. 2002.

[11] J. Weston, A. Elisseeff, B. Schölkopf, and M.E. Tipping, "Use of the Zero-Norm with Linear Models and Kernel Methods," *J. Machine Learning Research,* vol. 3, pp. 1439-1461, Nov. 2003.

[12] T.K. Moon and W.C. Stirling, *Mathematical Methods and Algorithms for Signal Processing.* Prentice Hall, 2000.

[13] W. Krzanowski, "Between-Groups Comparison of Principal Components," *J. Am. Statistical Assoc.,* vol. 74, no. 367, 1979.

[14] C.M. Bishop, *Neural Networks for Pattern Recognition.* Oxford Univ. Press, 1996.

[15] S.K. Pal and P. Mitra, *Pattern Recognition Algorithms for Data Mining: Scalability, Knowledge Discovery, and Soft Granular Computing.* Boca Raton, Fla.: Chapman Hall/CRC Press, May 2004.

[16] R. Kohavi and G.H. John, "Wrappers for Feature Subset Selection," *Artificial Intelligence,* vol. 97, nos. 1-2, pp. 273-324, 1997.

[17] J. Han and M. Kamber, *Data Mining: Concepts and Techniques,* chapter 3, p. 121, Morgan Kaufmann, 2000.

[18] S. Hettich and S.D. Bay, "The UCI KDD Archive," http://kdd.ics.uci.edu, 1999.

[19] S. Zhong and J. Ghosh, "HMMs and Coupled HMMs for Multi-Channel EEG Classification," *Proc. Int'l Joint Conf. Neural Networks,* 2002.

[20] I. Cohen, Q. Tian, X.S. Zhou, and T.S. Huang, "Feature Selection Using Principal Feature Analysis," Univ. of Illinois at Urbana-Champaign, 2002.

[21] I.T. Jolliffe, *Principal Component Analysis.* Springer, 2002.

[22] B.N. Flury, "Common Principal Components in *k* Groups," *J. Am. Statistical Assoc.,* vol. 79, no. 388, pp. 892-898, 1984.

[23] W. Krzanowski, "Orthogonal Components for Grouped Data: Review and Applications," *Statistics in Transition,* vol. 5, no. 5, pp. 759-777, Oct. 2002.

[24] J.R. Schott, "Some Tests for Common Principal Components in Several Groups," *Biometrika,* vol. 78, pp. 771-777, 1991.

[25] V. Vapnik, *Statistical Learning Theory.* New York: Wiley, 1998.

[26] C.-C. Chang and C.-J. Lin, "LIBSVM—A Library for Support Vector Machines," http://www. csie. ntu.edu.tw/~cjlin/libsvm/, 2004.

[27] R. Kohavi, "A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection," *Proc. Int'l Joint Conf. Artificial Intelligence,* 1995.

[28] M. Shah and R. Jain, *Motion-Based Recognition,* chapter 15, Kluwer Academic Publishers, 1997.

[29] H. Sakoe and S. Chiba, "Dynamic Programming Algorithm Optimization for Spoken Word Recognition," *IEEE Trans. Acoustics, Speech, and Signal Processing,* vol. 26, no. 1, 1978.

[30] K. Yang and C. Shahabi, "A PCA-Based Similarity Measure for Multivariate Time Series," *Proc. Second ACM Int'l Workshop Multimedia Databases,* 2004.

[31] J. Weston, A. Elisseeff, G. BakIr, and F. Sinz, "Spider: Object-Orientated Machine Learning Library," http://www.kyb.tuebingen.mpg.de/bs/people/spider/, 2005.

[32] "Marker Configurations for the Humangait Dataset," ftp://ftp.cc.gatech.edu/pub/gvu/cpl/walkers/speed_control_data/doc, 2003.

[33] P. Mitra, C. Murthy, and S.K. Pal, "Unsupervised Feature Selection Using Feature Similarity," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, no. 3, pp. 301-312, Mar. 2002.

[34] R. Tibshirani, G. Walther, and T. Hastie, "Estimating the Number of Clusters in a Data Set Via the Gap Statistic," *J. Royal Statistical Soc.: Series B (Statistical Methodology),* vol. 63, no. 2, pp. 411-423, 2001.

[35] D. Leibovici and R. Sabatier, "A Singular Value Decomposition of a *k*-Way Array for a Principal Component Analysis of Multiway Data, PTA-*k*," *Linear Algebra and Its Applications,* 1998.

Hyunjin Yoon received the BS and MS degrees in computer science and engineering from Ewha Womans University, Seoul, Korea, in 1996 and 1998, respectively. In 2003, she received another MS degree in computer science from the University of Southern California (USC). She is currently working toward the PhD degree in computer science (information laboratory) at USC. She worked as a researcher at the Center for Advanced Information System, KAIST, Korea, from 1998 to 1999. Her research interests include multidimensional data analysis, data mining, and machine learning, especially with the applications in human-computer interactions.

Kiyoung Yang is currently a PhD student in the Computer Science Department, the University of Southern California. He received the bachelor's degree in computer science from the Korea Advanced Institute of Science and Technology (KAIST) in 1996 and the master's degree from the University of Southern California in 2002. From 1996 to 2000, he worked for a couple of Information Technology (IT) companies, including Oracle Korea, as a senior technical consultant. He has served on many conference programs as an external reviewer, such as IEEE ICDE 2004 and ACM SIGMOD 2004. His research interests include multivariate time series analysis based on the machine learning and statistical pattern recognition techniques.

Cyrus Shahabi received the PhD degree in computer science from the University of Southern California in August 1996. He is currently an associate professor and the director of the Information Laboratory (InfoLAB) at the Computer Science Department and also a research area director at the US National Science Foundation's Integrated Media Systems Center (IMSC) at the University of Southern California. He has two books and more than 100 articles, book chapters, and conference papers in the areas of databases and multimedia. His current research interests include peer-to-peer systems, streaming architectures, geospatial data integration, and multidimensional data analysis. He is currently on the editorial board of *IEEE Transactions on Parallel and Distributed Systems* and is the program committee chair of ICDE NetDB 2005 and ACM GIS 2005. He is also serving on many conference program committees, such as IEEE ICDE 2006, SSTD 2005, ACM CIKM 2005, and ACM SIGMOD 2004. He was the recipient of the 2002 US National Science Foundation CAREER Award and the 2003 Presidential Early Career Awards for Scientists and Engineers (PECASE). In 2001, he also received an award from the Okawa Foundations. He is a member of the IEEE.