IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 22, 2017, accepted June 23, 2017, date of publication July 17, 2017, date of current version August 8, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2726543

Big Feature Data Analytics: Split and Combine Linear Discriminant Analysis (SC-LDA) for Integration Towards Decision Making Analytics

JASMINE KAH PHOOI SENG, (Member, IEEE), AND KENNETH LI-MINN ANG, (Senior Member, IEEE)
School of Computing and Mathematics, Charles Sturt University, Sydney, NSW 2678, Australia
Corresponding author: Jasmine Kah Phooi Seng (jasmine.seng@gmail.com)

ABSTRACT This paper introduces a novel big feature data analytics scheme for integration toward data analytics with decision making. In this scheme, a split and combine approach for a linear discriminant analysis (LDA) algorithm termed SC-LDA is proposed. The SC-LDA replaces the full eigenvector decomposition of LDA with much cheaper eigenvector decompositions on smaller sub-matrices, and then recombines the intermediate results to obtain the exact reconstruction as for the original algorithm. The splitting or decomposition can be further applied recursively to obtain a multi-stage SC-LDA algorithm. The smaller sub-matrices can then be computed in parallel to reduce the time complexity for big data applications. The approach is discussed for an LDA algorithm variation (LDA/QR), which is suitable for the analytics of Big Feature data sets. The projected data vectors into the LDA subspace can then be integrated toward the decision-making process involving classification. Experiments are conducted on real-world data sets to confirm that our approach allows the LDA problem to be divided into the size-reduced sub-problems and can be solved in parallel while giving an exact reconstruction as for the original LDA/QR.

INDEX TERMS Big data, feature extraction, linear discriminant analysis, classification, computational complexity.

I. INTRODUCTION

The rising importance of Big Data computing stems from advances in many technologies such as sensors, computer networks, data storage, cluster computer systems, cloud computing facilities, and data analytics. The data is considered as ''Big'' when it meets the requirements of the ''four *V*'s'', such as *Volume*, *Variety*, *Velocity*, and *Valu*e. The first three *V*'s are three defining properties or dimensions of Big Data. Volume refers to the amount of data, variety refers to the number of types of data, and velocity refers to the speed of data processing. Big Data analytics is concerned with the automatic algorithmic extraction and logical analysis of information found in huge data volumes to help in enrichment of business value chains or to bring significant science and engineering advances.

Big data is a disruptive force presenting opportunities as well as challenges. With the sheer volume and dimensionality of data, researchers in different fields face formidable challenges in dealing with large-scale data sets using traditional analytics methods. Traditional information processing and statistical learning tools such as principal component analysis (PCA), subspace clustering, etc., need to be re-examined in today's high-dimensional data regimes. Before re-examining the right tools, it is important to understand the nature of the Big Data sets. Big Data sets in fact can be divided into two categories: (1) Big Sample Data Sets; and (2) Big Feature Data Sets. A data set normally has n observations/samples and m variables/ features (dimensions). How the datasets are classified depends on samples denoted as n and the features denoted as m of the original data before the growth in the data volume. In the first category ($n >> m$), the data dimension is not large. As the volume grows due to more samples, the dimension of n increases, but the dimension of m may remain the same. In the second category ($m >> n$), m is large due to a big number of variables/features and can be further increased as the number

of features increases. Meanwhile n can also be increased due to the growth in sample volume.

Many real world applications or datasets fall into the second category. One specific research field which involves high dimensions and big volume data is gene classification in computational biology. For years, genes have been studied and mapped. A typical human genome contains more than 20,000 genes. Each gene data has a long sequence and is made up of over 3 million base pairs. Simply mapping a genome requires a hundred gigabytes of data, and sequencing multiple genomes and tracking gene interactions multiplies that number many times — hundreds of petabytes in some cases. Thus, genes data sets are classified under the second category of Big Feature Data Sets. Another example are the data sets used in remote sensing applications. Each data sample after reshaping the image into a vector covers large areas of the earth's surface before the size of data sets further grows as a long-term image sequence. In social networking applications, the web analytics and multimedia analytics involve multimodal data such as text, image, audio and video. Each data in the data sets consists of large numbers of different modalities of features before the volume is growing up.

There are two classical dimensionality reduction methods used widely together with machine learning in many applications; principal component analysis (PCA) and linear discriminant analysis (LDA). PCA is an unsupervised linear dimensionality reduction method, which seeks a subspace of the data that have the maximum variance and subsequently projects the input data onto it. PCA may not give good classification performance because it does not take into account any difference in the data class. On the other hand, LDA is a powerful traditional statistical technique for supervised data dimensionality reduction [1], [2] and has been applied successfully towards many applications to deal with high-dimensional data. It is different from PCA in that it reduces dimensionality while preserving as much of the class discriminatory information as possible. It explicitly attempts to model the difference between the classes of data. This make it more suitable if the data analytics involves decision making.

PCA and LDA are commonly cascaded when used for performing classification tasks [3]. In this case PCA is often used as a preprocessing to reduce the dimensionality of the data before performing the LDA. However, for Big Feature Data Sets, PCA may not be suitably performed before LDA because it would incur loss of some useful discriminatory information prior to the LDA stage [4]. LDA has been studied extensively and various extensions have been developed. However, LDA for Big Data in general, or specifically for Big Feature Data Sets has not been fully explored or exploited. In addition, Big data decision analytics is still at the infancy stage. A Big data feature extraction technique which is suitable and can be integrated to the Big data decision analytics is highly desired.

In this paper, a novel approach of Big Feature data analytics for integration towards Big data decision analytics is introduced. In this approach, a technique termed the Split and Combine Linear Discriminant Analysis (SC-LDA) is proposed for Big Feature Data Analytics. In contrast to most of the existing LDA and extension methods aiming to improve the speed and efficiency, or focusing on incremental learning, our approach not only targets to reduce the computation, it also splits the original LDA problem into two size-reduced sub-problems, solves the sub-problems (with much cheaper computational cost) with a base algorithm separately, and then combines the results from the sub-problems to obtain the final solution. We term this as a single stage SC-LDA. Like PCA, the classical LDA requires an eigenvector decomposition on the full data matrix which is computationally very expensive. The SC-LDA replaces the full eigenvector decomposition with much cheaper eigenvector decompositions on smaller sub-matrices which can be computed in parallel, and then recombines the intermediate results to obtain the exact reconstruction as for the original algorithm.

Second, the splitting or decomposition can be further applied recursively towards the single stage SC-LDA to obtain a two stage SC-LDA, and the process continued on to obtain a multi-stage SC-LDA decomposition. Third, to further improve the scalability, we propose to solve each sub-problem in parallel. It is important to note our approach of SC-LDA is particularly suitable for Big Feature data analytics. Experiments are conducted to confirm our approach allows the LDA problem to be divided into the size-reduced sub-problems and can be solved in parallel while giving an exact reconstruction as for a state-of-the-art LDA algorithm variation (LDA/QR [5]) suitable for Big Feature Data Sets. Our approach is to split a large problem into sub-problems to be solved separately before recombination. To the best of our knowledge, no similar work has been reported for classification tasks applications, and specifically for the LDA.

The focus of our paper is to demonstrate that the LDA problem can be computed in parallel using a split and combine approach to increase the computational efficiency for big data applications while giving an exact reconstruction using the LDA/QR as an example application. The performance of the LDA/QR for classification tasks has been demonstrated by the authors in [5] to be competitive with other state-of-the-art algorithms for discriminant analysis. With appropriate modifications, the split and combine approach can also be used for other algorithms for dimensionality reduction and discriminant analysis. The remainder of the paper is organized as follows: Section II reviews related works for some existing LDA algorithms and variations used in the paper. Section III presents the proposed SC-LDA algorithm for a classification task, and discusses extensions for multi-stage decompositions. The computational complexity of the SC-LDA with other LDA algorithms is discussed in Section IV. The SC-LDA decompositions give an exact reconstruction as for the original LDA/QR algorithm on condition that the data samples maintain the linear independence after the split. This condition is implicit in the definition of the QR where $A \in R^{m \times n}$ can be decomposed as $A = QR$

if it has linearly independent columns. Some experimental results on real-world data sets to validate the approach and the linear independence condition is given in Section V. Finally, discussions and conclusions are given in Sections VI and VII.

II. RELATED WORKS

Because Big Feature Data Analytics has not been fully explored and Big data decision analytics is at its infancy stage, this section only briefly reviews current research for the LDA and some of its related variations from two directions as useful background information for our research in the SC-LDA for Big Feature data analytics: (i) incremental learning LDA algorithms; and (ii) LDA for undersampled data sets. We will also briefly comment on how our work is related in regards to applications for (i) and (ii).

A. INCREMENTAL LEARNING LDA ALGORITHMS

Many existing LDA algorithms are batch algorithms, which require that the data must be available in advance and be given once altogether. Incremental learning LDA algorithms allow the advantages to deal with data streams when new data are presented, and learning all the data from the beginning is not required. This will reduce the requirement for large memory and high computational complexity because the system would not need to maintain a huge memory to store the data either previously learned or newly presented.

There have been two main approaches to updating LDA features: indirect and direct. In the indirect approach, the incremental algorithms are used to update the matrices which are involved in computing the LDA features and then the LDA features are computed through solving an eigen decomposition problem. For example, Pang *et al.* [6] presented incremental algorithms to update the within-class and between-class scatter matrices and used them to update the LDA features. Ye *et al.* [7] used an incremental dimension reduction (IDR) algorithm with QR decomposition for adaptive computation of the reduced forms of within-class and between-class scatter matrices. The proposed algorithm by Uray *et al.* [8] involves performing PCA on an augmented matrix and then updating the LDA features. Kim *et al.* [9], [10] used sufficient spanning approximations for updating the mixture scatter matrix, the between-class scatter matrix, and the projected data matrix. None of these algorithms deals with the LDA features directly, and updating the LDA features is instead done by solving an eigenvalue decomposition problem.

In contrast to the techniques above, there are incremental algorithms that update LDA features directly. Chatterjee and Roychowdhury [11] proposed an incremental self-organized LDA algorithm for updating the LDA features. Their approach involved the incremental computation of $Q^{-1/2}$, where Q is the correlation matrix of the input data. In other works, Demir and Ozmehmet [12] proposed online local learning algorithms for updating LDA features incrementally using error-correcting and the Hebbian learning rules. Moghaddam *et al.* [13], [14] and Ghassabeh and Moghaddam [15] derived new incremental algorithms to accelerate the convergence rate of the proposed algorithm in [11]. The proposed algorithms are derived based on the steepest descent, conjugate direction, Newton–Raphson, and quasi-Newton methods. Moghaddam *et al.* [13], [14] and Ghassabeh and Moghaddam [15] used an implicit cost function to find the optimal step size in order to accelerate the convergence rate. However, their works did not have access to the explicit cost function, and could not guarantee the convergence of the proposed algorithms. We note that the incremental learning algorithms (both direct and indirect) in [6]–[15] are focused mainly on reducing the computational requirements when new data are presented to avoid computing and learning all the data from the beginning. However, the incremental schemes would still require an initial LDA transformation to be performed on the data set before the application of the incremental learning. In contrast to incremental learning LDA schemes, the SC-LDA proposed in this paper is focused on reducing the computational complexity for the initial LDA transformation. With suitable application, the SC-LDA techniques can also serve as an incremental learning LDA approach. Some discussion of this is given towards the end of the paper.

B. LDA FOR UNDERSAMPLED DATA SETS

A well-known problem in classical LDA is termed as the singularity or undersampled problem where the number of features (m) in the data matrix is large compared with the number of samples (n). In our definition, this would be classified as a Big Feature Data Set for a very large data matrix to be analyzed. A common solution to address this singularity problem is to apply PCA for feature dimensionality reduction, prior to using the LDA. However, the disadvantage of this approach is that it would incur loss of some useful discriminatory information prior to the LDA stage. Other approaches which have been proposed to address the singularity problem is to use different variations of the classical LDA objective function. In this paper, we make use of a LDA algorithm variant termed LDA/QR [5] which has been designed to address the singularity issue. The rest of this section gives some discussions on the LDA algorithm as well as other LDA variations which will be used for comparisons in this paper. Given a data matrix of full column rank with class label

$$A = [a_1 \dots a_n] = [\mathcal{A}_1 \dots \mathcal{A}_k] \in \mathbf{R}^{m \times n}$$

where each $a_i (1 \le i \le n)$ is a data point in an m-dimensional space and each block matrix $\mathcal{A}_i \in \mathbf{R}^{\mathrm{m} \times \mathrm{n_i}} (1 \le i \le k)$ is a collection of data items in the ith class, $n_i (1 \le i \le k)$ is the size of the class i, and the total number of data items in data set A is $n = \sum_{-i=1}^{k} n_i$. Let N_i denote the set of column indices that belong to the class i. The global centroid c of A and the local centroid c_i of each class A_i are given by

$$c = \frac{1}{n} Ae, C_i = \frac{1}{n_i} \mathcal{A}_{\mathrm{i}} e_i, i = 1, \dots, k \qquad (1)$$

respectively, where

$$e = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \in \mathbf{R}^{\mathrm{n}}, e_i = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \in \mathbf{R}^{\mathrm{n_i}}, i = 1, \ldots, k. \quad (2)$$

Let

$$S_b = \sum_{i=1}^{k} n_i \left(c_i - c\right)\left(c_i - c\right)^T \quad (3)$$

$$S_w = \sum_{i=1}^{k} \sum_{j \in N_i} \left(a_j - c\right)\left(a_j - c\right)^T \quad (4)$$

$$S_t = \sum_{j=1}^{n} \left(a_j - c\right)\left(a_j - c\right)^T \quad (5)$$

where S_b, S_w, and S_t are called the between-class scatter matrix, the within-class scatter matrix, and the total scatter matrix, respectively. It is well-known that the total scatter matrix is related to the between-class scatter matrix and the within-class scatter matrix by,

$$S_t = S_b + S_w \quad (6)$$

The classical LDA objective function is then given by Equation (7) as,

$$G = arg \max_{G \in \mathbf{R}^{\mathrm{m} \times 1}} trace \left(\left(G^T S_t G \right)^{-1} \left(G^T S_b G \right) \right) \quad (7)$$

where $G \in \mathbf{R}^{m \times l}$ projects data in a m-dimensional space to the l-dimensional subspace.

To address the undersampled (singularity) LDA problem, Chu *et al*. [5] proposed an efficient algorithm called LDA/QR which uses a modification/variation of the classical LDA objective function as shown in Equation (8),

$$G = arg \max_{G \in \mathbf{R}^{\mathrm{m} \times 1}} trace \left(\left(G^T S_t G \right)^{(+)} \left(G^T S_b G \right) \right) \quad (8)$$

where $(\cdot)^{(+)}$ denotes the pseudoinverse of $(\cdot)$.

Another variation of the LDA is the uncorrelated LDA (ULDA) [16]. ULDA was originally proposed by Chu *et al.* [16] for extracting feature vectors with uncorrelated attributes. Later on, the ULDA was generalized by researchers in [17] and [18] to address undersampled problems based on simultaneous diagonalization of the scatter matrices. The new ULDA objective function is shown in Equation (9).

$$G = arg \max_{G \in \mathbf{R}^{\mathrm{m} \times 1}, G^T S_t G = 1} trace \left(\left(G^T S_t G \right)^{(+)} \left(G^T S_b G \right) \right) \quad (9)$$

III. SPLIT AND COMBINE LINEAR DISCRIMINANT ANALYSIS (SC-LDA) FOR BIG FEATURE DATA

For Big Feature data analytics with large and high-dimensional data sets, a huge computational complexity and lack of available storage may be critical issues. For ease of discussion, this section first discusses a single stage SC-LDA for a classification task using two-fold cross validation (extensions to v-fold cross-validation will be briefly discussed later) which is designed towards binary or two-class decision making. The approach will then be extended to discuss the multi stage SC-LDA towards multiclass decision making.

A. SINGLE STAGE SC-LDA

For classification tasks, the data set is first divided into two sets for training and testing. This is termed as the training set and the testing set respectively. We will illustrate the SC-LDA for a two-fold cross validation algorithm using a 1:1 ratio for the training and testing sets. At the start, the input data is stored in an $m \times n$ matrix, A, where m denotes the number of features and n denotes the number of samples. The class data is stored in a $1 \times n$ vector, C. The output of the algorithm would be three matrices denoted as Y_1, Y_2, Y_{Test}, and three row vectors C_1, C_2, and C_{Test} where Y_1 and Y_2 are two training sub-matrices of size $m \times n/4$, and Y_{Test} is a matrix of size $m \times n/2$.

The row vectors C_1, C_2, and C_{Test} holds the class labels for the respective samples. All the data samples in the original data matrix A will be used to form the three matrices. The data partitioning scheme presented here needs class vectors to train a classifier prior to performing the classification task. The SC-LDA algorithm for a classification task can be described using six steps. Steps 1 to 5 are used for the training process, whereas Step 6 is used for the testing process.

Step 1 Splitting into Data Sub-Matrices Y_1, Y_2, Y_{Test} and Class Row Vectors C_1, C_2, C_{Test}

Input: A ($m \times n$ matrix), C ($1 \times n$ vector).
For $i = 1$ to 2 do
  For $j = 1$ to $n/4$ do
    1.1 Randomly select a column x from A (without replacement) and mark the selected column.
    1.2 Select the corresponding column entry y from C.
    1.3 Set x as the j-th column in the sub-matrix Y_i.
    1.4 Set c as the j-th entry in the row vector C_i.
  End For
End For
    1.5 Set Y_{Test} to the remaining unmarked columns in C.
Output: Y_1 and Y_2 ($m \times n/4$), Y_{Test} ($m \times n/2$), C_1 and C_2 ($1 \times n/4$), C_{Test} ($1 \times n/2$).

Step 2 Solving Sub-Problems in Parallel

Input: Y_1 and Y_2 ($m \times n/4$), C_1 and C_2 ($1 \times n/4$).
Do in Parallel
    2.1 Compute the economic QR factorization of Y_i as $Y_i = Q_i R_i$ where Q_i is an orthogonal matrix of size ($m \times n/4$) and R_i is an upper triangular matrix of size ($n/4 \times n/4$).
End Do
Output: Q_1 and Q_2 ($m \times n/4$), R_1 and R_2 ($n/4 \times n/4$).

Step 3 Combining Intermediate Solved Data Matrices

This step combines the intermediate results from Step (2). We make use of the following notation in our description. Let the economic QR factorization for the full data matrix Y be $Y = QR$ where Y is the column concatenation of Y_i (i = 1,2) and Q is an orthogonal matrix of size ($m\times$ $n/2$) and R is an upper triangular matrix of size ($n/2 \times n/2$). We will partition the Y, Q, and R matrices as shown in Fig. 1.

Input: Q_1 and Q_2 ($m \times n/4$), R_1 and R_2 ($n/4 \times n/4$).

3.1 Calculate Z as the matrix expression of ($Q_2R_2 - Q_1Q_1^{\mathrm{T}}(Q_2R_2)$).

3.2 Compute the economic QR factorization of Z as $Z = Q_zR_z$, where Q_z is an orthogonal matrix of size ($m\times n/4$) and R_z is an upper triangular matrix of size ($n/4 \times n/4$).

3.3 Reconstruct the matrices Q and R from its various sub-matrices as $Q_A = Q_1$, $Q_B = Q_z$, $R_A = R_1$, $R_B = Q1^{\mathrm{T}}(Q2R2)$ andR_C=R_Z.

Output: $Q(m \times n/2)$, R ($n/2 \times n/2$).

Step 4 Computing the LDA Objective Function

This step computes the LDA objective function. We make use of a LDA algorithm variant termed LDA/QR. The detailed derivations can be found in [5].

Input: $Q(m \times n/2)$, R ($n/2 \times n/2$), C_1 and C_2 ($1 \times n/4$).

4.1 Construct a binary class label matrix C_z for the column concatenation of C_i (i = 1,2) where C_z is a matrix of size ($n/2 \times k$) and k is the number of classes in the data set. Each row in the C_z matrix contains only a single "1" element, with all other elements in the row set to "0". The k^{th} element for the n^{th} row is set to "1" if the corresponding column in the data matrix Y belongs to the k^{th} class.

4.2 Compute the LDA objective function as $G = Q(R^{-\mathrm{T}}C_z)$.

Output: $G(m \times k)$.

Step 5 Projecting Training Vectors into the LDA Subspace for Classification Training

Input: Y ($m \times n/2$), $G(m\times k)$, concatenated matrix $\{[C_1,C_2]\}$ ($1 \times n/2$).

5.1 Compute the LDA training vector projections $Y_{\mathrm{LDA_Train}}$ as $Y_{\mathrm{LDA_Train}} = G^{\mathrm{T}}Y$.

5.2 The training vectors together with its corresponding class labels can then be applied towards training a supervised classifier.

Output: Trained classifier.

Step 6 Evaluating Performance of Trained Classifier With Cross-Validation

Input: Y_{Test} ($m \times n/2$), $G(m \times k)$, C_{Test} ($1 \times n/2$), trained classifier.

6.1 Compute the LDA testing vector projections $Y_{\mathrm{LDA_Test}}$as $Y_{\mathrm{LDA_Test}} = G^{\mathrm{T}}Y_{\mathrm{Test}}$.

6.2 The testing vectors together with its corresponding class labels can then be applied towards evaluating the performance of the trained classifier. For performing two-fold cross-validation, the data matrices Y and Y_{Test} are swapped and the Steps (2) to (6) are repeated.

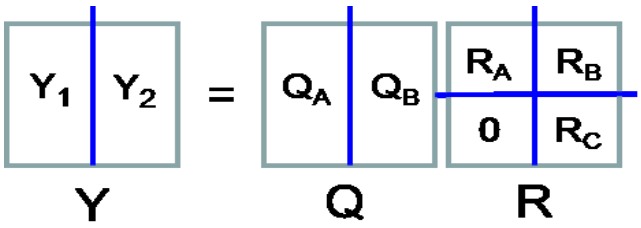

FIGURE 1. Partitioning of Y, Q and R matrices.

The algorithm can be adapted to perform v-fold cross validation by splitting the data into v separate sub-matrices. In this case, Step (2) can still be performed for all the data sub-matrices in parallel, but the combination in Step (3) will require a multi-stage SC-LDA decomposition. For example, Fig. 2 shows a partitioning of the Y matrix to perform a 4-fold cross validation when the data is split into four separate sub-matrices, and the split and combine parallel processes for its subsequent recombination. The approach can be extended for 8-fold, 16-fold cross validation and so on. A 8-fold cross validation will split the data into eight sub-matrices, a 16-fold cross validation will split the data into 16 sub-matrices, and so on. Further splits can be performed on the sub-matrices on condition that the sub-matrices retain linearly independent columns after the split. The following section discusses the multi-stage SC-LDA and gives a graphical representation.

B. MULTI STAGE SC-LDA

The Single Stage SC-LDA discussed in the preceding section can be further split recursively into multiple stages for a Multi Stage SC-LDA decomposition. A graphical example of a two-stage splitting is shown in Fig. 3. In this case, the figure shows the sub-matrix Y_2 is split into a further two sub-matrices Y_{21} and Y_{22} (i.e. $Y_2 = \{[Y_{21}, Y_{22}]\}$) and the QR decomposition is performed separately on Y_{21} and Y_{22} before being passed into its Combine Data Stage. Although not shown in Fig. 3, the data sub-matrix Y_1 can be similarly split into two sub-matrices Y_{11} and Y_{12} to give a full two stage SC-LDA decomposition. This recursive decomposition can be further applied towards splitting the sub-matrices Y_{11}, Y_{12}, Y_{21}, and Y_{22} for a full three stage

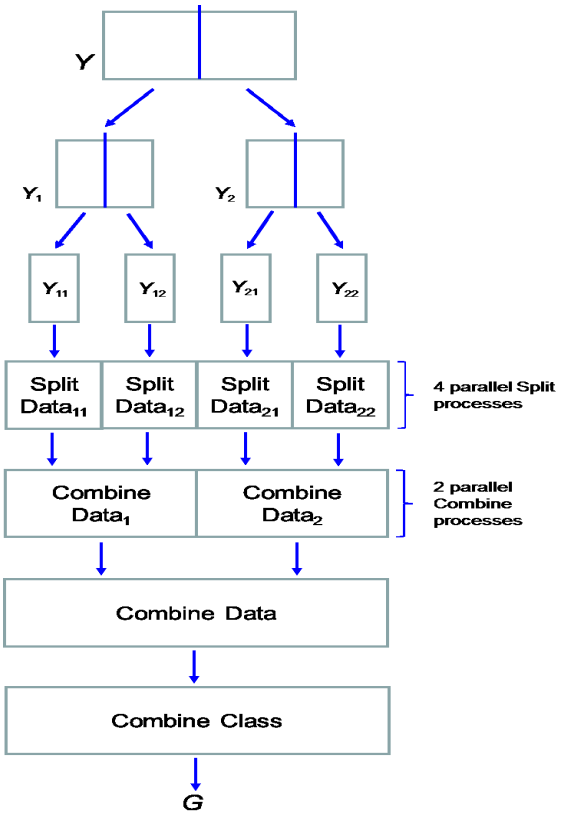

FIGURE 2. Partitioning of Y matrix for 4-fold cross validation and the split and combine processes.

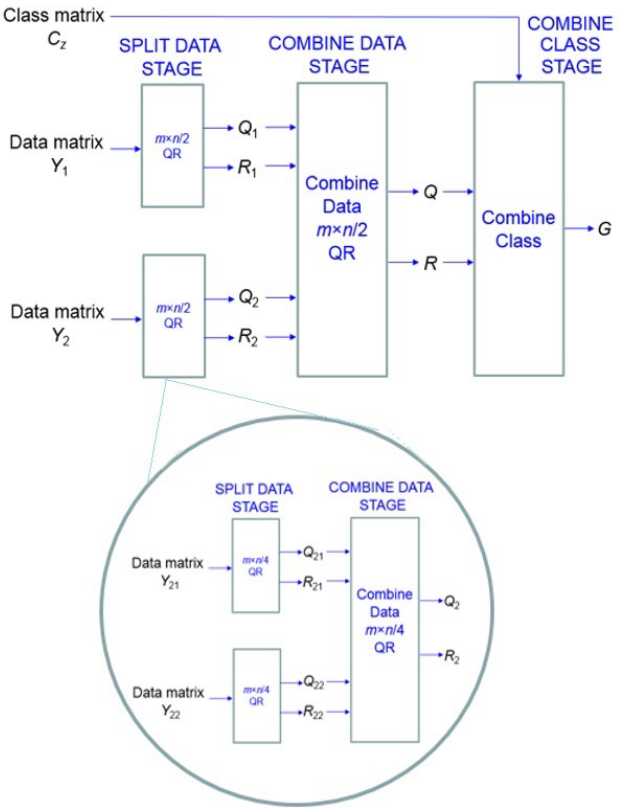

FIGURE 3. Graphical representation of Two-Stage SC-LDA.

SC-LDA decomposition, and the process can be continued for further stages of SC-LDA decomposition. The QR decomposition for each of the sub-matrices can then be computed in parallel to reduce the time complexity for the eigenvector decompositions.

IV. COMPUTATIONAL COMPLEXITY OF SC-LDA

The computational complexity of the SC-LDA (Steps 2 and 3) consists of QR decompositions and some matrix operations. The original LDA/QR to perform a QR decomposition on a data matrix of size $m \times n$ would have a complexity of $O(mn^2)$. We can consider the original LDA/QR algorithm as a SC-LDA decomposition at level/stage 0 (i.e. without any splitting). The single stage SC-LDA reduces the complexity to three QR computations of $O(m(n/2)^2)$. Two QR computations are required for the Split Data Stage, and another QR computation is required for the Combine Data Stage. The two stage SC-LDA further reduces the complexity to QR computations of $O(m(n/4)^2)$. Table I shows the computational complexity of QR decompositions for various configurations of the SCA-LDA. It is important to note that the multi stage SC-LDA configurations allow the QR computations to be performed in parallel and significantly reduce the time complexity. For Big Feature Data Sets, many decomposition levels can be utilized to split the original data matrix into multiple smaller sub-matrices to be computed in parallel by separate core processing units. In the literature, several

TABLE 1. Main computational costs for various SC-LDA decomposition configurations.

Number of decomposition stages	QR computations required			
	$(m\times n)$	$(m\times n/2)$	$(m\times n/4)$	$(m\times n/8)$
0	1	-	-	-
1	-	3	-	-
2	-	1	6	-
3	-	1	2	12

methods are available for QR computations. The experiments used the householder method [5] where a QR computation of an $m \times n$ matrix has a main computational cost of $4mn^2 - (4/3)n^3$ flops. The main computational cost of the single stage SC-LDA is given by $3 \times (4m(n/2)^2 - 4/3(n/2)^3)$, whereas the main computational costs of the two stage and three stage SC-LDA decompositions are given by $6 \times (4m(n/4)^2 - 4/3(n/4)^3) + (4m(n/2)^2 - 4/3(n/2)^3)$ and $12 \times (4m(n/8)^2 - 4/3(n/8)^3) + 2 \times (4m(n/4)^2 - 4/3(n/4)^3) + (4m(n/2)^2 - 4/3(n/2)^3)$ respectively.

Fig. 4 shows a comparison of the computational cost for the single stage SC-LDA with two other LDA algorithms (the LDA/QR and the ULDA) for various ratios of m/n from 0.1 to 50. The SC-LDA computational costs include the QR decompositions and the associated matrix operations for Steps 1 to 4. The computational costs for Steps 5 and 6 will be the same for all LDA algorithms. A computational

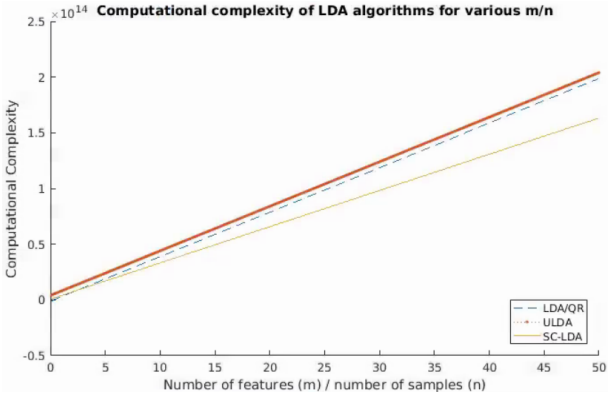

FIGURE 4. Computational complexity of Single-Stage SC-LDA vs other LDA variations.

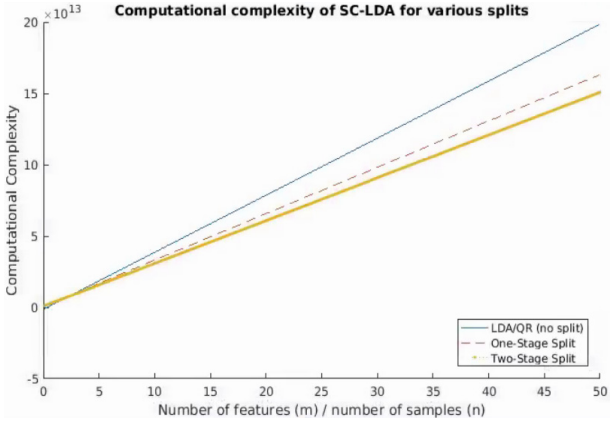

FIGURE 5. Computational complexity of SC-LDA for various splitting configurations.

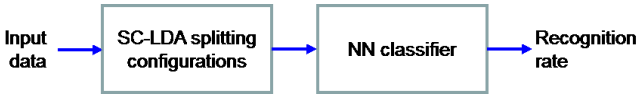

FIGURE 6. Experimental approach for SC-LDA splitting configurations.

simplification can be derived by noting that the calculations for SC-LDA for the LDA/QR only require the product of Q_2R_2 which is equal to Y_2. A remark here is that the split and combine approach can be applied towards other LDA formulations which may require separate application of the Q_2 and R_2 matrices. Thus, we have retained the general structure for the split and combine approach in Fig. 2 and Fig. 3, and modifications for computational processing can be performed as required. As mentioned previously, the original LDA/QR can be seen as a level/stage 0 SC-LDA implementation. The number of samples n used was 10,000. Thus, for a ratio of m/n of 1, the data matrix will have a size of $10,000 \times 10,000$. For a ratio of m/n of 50, the data matrix will have a size of $500,000 \times 10,000$. These are large matrices for eigenvector decomposition. A ratio of m/n of 1 would indicate a square data matrix A with an equal number of data features and samples. A ratio smaller than 1 would indicate a Big Data matrix with fewer number of features than samples (i.e. an example of a Big Sample Data Set), whereas a ratio greater than 1 would indicate a Big Data matrix with larger number of features than samples (i.e. an example of a Big Feature Data Set).

The computational cost of the single stage SC-LDA was lower than ULDA for all ratios of m/n. On the other hand, the results showed that there is a tradeoff for SC-LDA and LDA/QR. For ratios of m/n smaller than 1, the computational cost for LDA/QR was lower than for SC-LDA. However for Big Feature Data Sets (m >> n), the SC-LDA gave a 20% and 18% decrease in computational complexity compared to ULDA and LDA/QR. The observations showed that the SC-LDA is useful to reduce computational cost for Big Feature Data Sets but may incur more computational cost for Big Sample Data Sets. As discussed in the Introduction section, many real world applications or datasets from computational biology, remote sensing, social networking, etc. fall into the category of Big Feature Data Sets. Fig. 5 shows a further comparison for the computational complexity of SC-LDA for various splitting decompositions. The Two Stage split gave a 26% and 24% reduction in computational complexity over ULDA and LDA/QR respectively. Further stages of decomposition for the SC-LDA would result in a further reduction in computational complexity over the other LDA algorithms.

V. EXPERIMENTAL RESULTS

The split and combine approach discussed in Section III has demonstrated that the theoretical formulation allows the SC-LDA to be split recursively. However, there is a condition in the definition of the QR decomposition that the data samples maintain the linear independence after the split. This section presents experimental results to validate the approach using two datasets for face recognition (ORL [23] and Yale [24]) containing real-world variations. The ORL dataset contains 400 images of 40 people. There are 10 samples for each person with variations such as facial expressions and appearance. The Yale dataset contains variations in facial expression and lighting conditions. We used 640 images from the Yale dataset for 10 people. These datasets are examples of Big Feature Data Sets, where the number of features (m) >> the number of samples (n). For example, after reshaping the facial image into column vectors, we obtained an input data matrix of $10,304 \times 400$ and $2,500 \times 640$ for the ORL and Yale datasets respectively. Fig. 6 shows an overview of the experimental approach.

We used the nearest neighbor (k-NN) classifier ($k = 3$) with the Euclidean distance metric to perform the classification task. The experiments used half of the input data for training the classifier, and the remaining half was used for evaluation to give the recognition rate. We used the following splitting configurations (0-split (SC-LDA 0), 1-split (SC-LDA 1), 2-split (SC-LDA 2) and 3-split (SC-LDA 3)). The experiments were similar except for the change in the construction of the LDA objective function (G) due to the

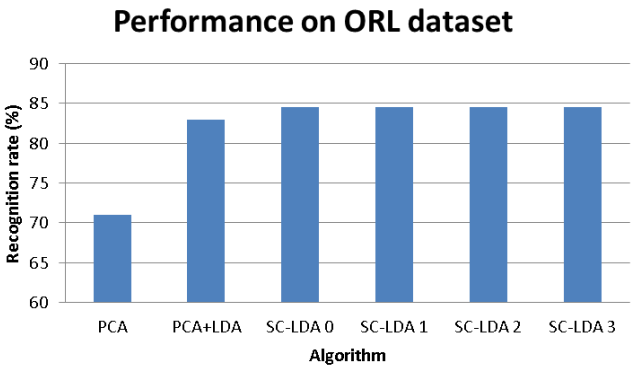

FIGURE 7. Recognition accuracies for various splitting configurations for ORL dataset.

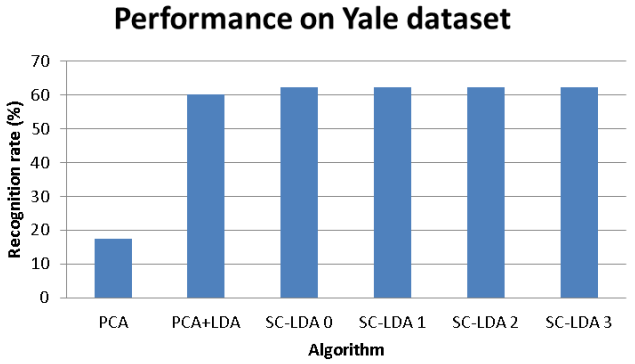

FIGURE 8. Recognition accuracies for various splitting configurations for Yale dataset.

splitting. The 0-split configuration used the original LDA/QR algorithm. For comparisons, the PCA and PCA+LDA (Fisherface) [25] methods were also included.

Fig. 7 and Fig. 8 show the recognition accuracy which were obtained for the various splitting configurations for ORL and Yale respectively. We obtained similar recognition rates for the various splitting configurations, showing that the linear independence condition was met for these real-world data sets. We did not detect any degradation in the performance even for 3-split (SC-LDA 3) thus confirming that the SC-LDA gave an exact reconstruction. For the 3-split, the training data was split into eight subsets and then recombined. For example, each data subset for ORL had a matrix size of $10,304 \times 25$ with a large imbalance between m and n. As the number of samples increases for larger data sets, we expect the linear independence condition to be met even for many stages of splitting.

The recognition rates for the SC-LDA were also higher than PCA and slightly higher than PCA+LDA for both datasets. This showed the usefulness of the LDA/QR objective function which performed better than the well-known Fisherface method (PCA+LDA). As a final remark, note that the objectives of the experiments were to validate the split and combine approach and investigate the linear independence condition. In this paper, we make no attempt to claim very high performance for the classification task. The reader can refer to the paper in [26] which uses more advanced feature extraction and kernel classification techniques to achieve higher classification rates.

VI. FURTHER APPLICATIONS

This section briefly discusses the application of the SC-LDA approach towards three other application scenarios: (i) incremental LDA algorithms; (ii) pipelined and parallel SC-LDA architectures; and (iii) distributed computations in sensor networks. As mentioned in the Introduction, the SC-LDA can also serve as an incremental approach for LDA algorithms with some modifications. In this case, the Split Data Stage would consist of two sub-matrices to be merged in the Combined Data Stage. The first sub-matrix would contain the QR decomposition of the original batch data, and the second sub-matrix would contain the QR decomposition of the newly presented data. The remaining two stages (Combine Data Stage and Combine Class Stage) would remain the same.

A second scenario where the SC-LDA approach will be useful is for pipelined or parallel processing LDA architectures. It is to be noted that the computational comparisons in Section IV has not exploited the full potential of SC-LDA when more processor resources are available in the form of pipelining and multi-processing for distributed computing implementations. For a pipelined implementation, the SC-LDA architecture can be supplemented with pipeline registers between each of its different stages. For example, two pipeline registers can be inserted between the Split Data Stage/Combine Data Stage and the Combine Data Stage/Combine Class Stage to decrease latency, and improve the throughput. For a multiprocessor implementation, the different computations in the Split Data Stage can be distributed from a server farm to many computational service units to be performed in parallel, and then the intermediate results recombined in the Combine Data Stage. The final point to note is that the SC-LDA gives an exact reconstruction as for the original LDA/QR. The work by Chu *et al.* [5] has shown that the LDA/QR performs competitively with other LDA algorithms for classification tasks.

A third scenario where the SC-LDA approach will be useful is for distributed computations in sensor networks to reduce the energy consumption in battery-powered sensor nodes. The energy consumption for large-scale big data sensor networks containing thousands of sensor nodes, and the need for decentralized algorithms has been recently identified to be an important research challenge [19]–[21]. The work by Liang *et al.* [21] demonstrates a decentralized dimensionality reduction (DDR) algorithm for performing distributed computations in sensor networks. Their decentralized approach is achieved by trading-off some recognition performance when compared with other centralized approaches (although the authors showed that the performance degradation is very small and for practical purposes is comparable). It is important to note that our proposed SC-LDA gives the exact reconstructions as the original (centralized) LDA/QR. Thus, there

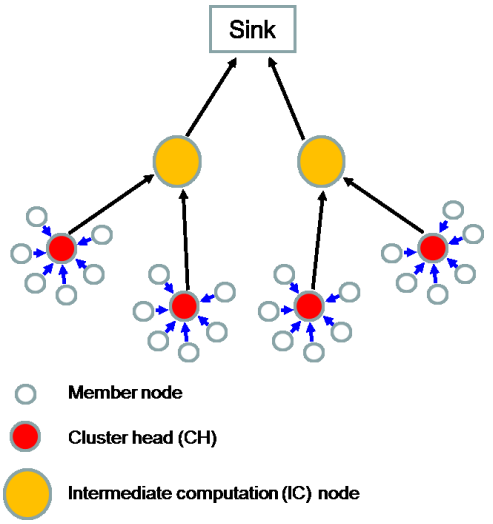

FIGURE 9. Application of SC-LDA using the LEACH protocol for distributed computations in large-scale sensor networks.

will be no trade-offs between performance and computational complexity.

Fig. 9 shows an application of the SC-LDA for decentralized computations in sensor networks using the Low Energy Adaptive Clustering Hierarchical (LEACH) protocol [22]. The LEACH protocol is a well-known data gathering protocol which utilizes a hierarchical clustering structure consisting of member nodes and cluster heads to optimize the life time of a sensor network. In a traditional approach, the data will first be forwarded from member nodes to a central location (sink) before the application computations (e.g. LDA) can commence. By using the SC-LDA, each cluster of member nodes gathers the data within the cluster and forwards to its cluster head (CH). Each CH can then perform local computations (i.e. the Split processes) without requiring the data from other clusters. The intermediate results from CHs are forwarded to intermediate computation (IC) nodes to perform the first set of Combine Data processes. The final Combine Data and Combine Class processes are performed by the sink node to obtain the LDA/QR result. The network can contain further levels of IC nodes for larger networks and hierarchies. The decentralized and hierarchical nature allows the LDA computations to be effectively distributed among the different types of sensor nodes (member nodes, CHs, ICs) for large-scale big data sensor networks. The ICs are larger devices which would have more computational power and memory storage than the CHs, which in turn would have more computational capabilities than the individual member nodes.

VII. CONCLUSION

This paper has presented a Big Feature data analytics approach embracing the SC-LDA technique which is a split and combine approach for the LDA/QR algorithm variation. The LDA/QR is a specialized LDA variation to deal with the undersampled problem faced by Big Feature Data Sets. Our approach has allowed the LDA problem to be divided into the size-reduced sub-problems and can be solved in parallel, while giving an exact reconstruction as for the original LDA/QR. The SC-LDA with this specific property is essential for Big Feature data analytics. Furthermore, the approach can be further developed towards the Big Feature data analytics with decision making. The single stage SC-LDA has first been presented followed by a multi-stage SC-LDA. The SC-LDA is particularly suitable for the analytics of Big Feature Data Sets, while giving an exact reconstruction as for the original algorithm. Experimental results on two real-world datasets have been used to validate the approach. Three application scenarios of SC-LDA have also been presented. For future work, we will investigate the split and combine approach for other dimensionality reduction algorithms and its applications for distributed computations in large-scale sensor networks.

REFERENCES

[1] S. Dudoit, J. Fridlyand, and T. P. Speed, "Comparison of discrimination methods for the classification of tumors using gene expression data," *J. Amer. Stat. Assoc.*, vol. 97, no. 457, pp. 77–87, 2002.

[2] W.-H. Yang, D.-Q. Dai, and H. Yan, "Feature extraction and uncorrelated discriminant analysis for high-dimensional data," *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 5, pp. 601–614, May 2008.

[3] P. N. Belhumeur, J. P. Hespanha, and D. Kriegman, "Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 7, pp. 711–720, Jul. 1997.

[4] D. Q. Dai and P. C. Yuen, "Face recognition by regularized discriminant analysis," *IEEE Trans. Syst., Man, Cybern., B (Cybern.)*, vol. 37, no. 4, pp. 1080–1085, Aug. 2007.

[5] D. Chu, L.-Z. Liao, M. K. P. Ng, and X. Wang, "Incremental linear discriminant analysis: A fast algorithm and comparisons," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 11, pp. 2716–2735, Nov. 2015.

[6] S. Pang, S. Ozawa, and N. Kasabov, "Incremental linear discriminant analysis for classification of datastream," *IEEE Trans. Syst., Man. Cybern., B, (Cybern.)*, vol. 35, no. 5, pp. 905–914, May 2005.

[7] J. Ye, Q. Li, H. Xiong, H. Park, R. Janardan, and V. Kumar, "IDR/QR: An incremental dimension reduction algorithm via QR decomposition," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 9, pp. 1208–1222, Sep. 2005.

[8] M. Uray, D. Skocaj, P. M. Roth, and A. L. H. Bischof, "Incremental LDA learning by combining reconstructive and discriminative approaches," in *Proc. Brit. Mach. Vis. Conf. BMVC*, 2007, pp. 272–281.

[9] T. Kim, S. Wong, B. Stenger, J. Kittler, and R. Cipolla, "Incremental linear discriminant analysis using sufficient spanning set approximation," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2007, pp. 1–8.

[10] T. Kim, S. Wong, B. Stenger, J. Kittler, and R. Cipolla, "Incremental linear discriminant analysis using sufficient spanning sets and its applications," *Int. J. Comput. Vis.*, vol. 9, no. 2, pp. 216–232, 2011.

[11] C. Chatterjee and V. P. Roychowdhury, "On self organizing algorithms and networks for class-separability features," *IEEE Trans. Neural Netw.*, vol. 8, no. 3, pp. 663–678, Mar. 1997.

[12] G. K. Demir and K. Ozmehmet, "Online local learning algorithms for linear discriminant analysis," *Pattern Recognit. Lett.*, vol. 26, no. 4, pp. 421–431, 2005.

[13] H. A. Moghaddam, M. Matinfar, S. M. S. Sadough, and K. A. Zadeh, "Algorithms and networks for accelerated convergence of adaptive LDA," *Pattern Recognit.*, vol. 38, no. 4, pp. 473–483, 2005.

[14] H. A. Moghaddam and M. Matinfar, "Fast adaptive LDA using Quasi–Newton algorithm," *Pattern Recognit. Lett.*, vol. 28, no. 4, pp. 613–621, 2007.

[15] Y. A. Ghassabeh and H. A. Moghaddam, "Adaptive linear discriminant analysis for online feature extraction," *Mach. Vis. Appl.*, vol. 24, no. 4, pp. 777–794, 2013.
[16] D. Chu, S. T. Goh, and Y. S. Hung, "Characterization of all solutions for undersampled uncorrelated linear discriminant analysis problems," *SIAM J. Matrix Anal. Appl.*, vol. 32, no. 3, pp. 820–844, 2011.
[17] J. Ye, "Characterization of a family of algorithms for generalized discriminant analysis on undersampled problems," *J. Mach. Learn. Res.*, vol. 6, pp. 483–502, Dec. 2005.
[18] J. Ye, R. Janardan, Q. Li, and H. Park, "Feature reduction via generalized uncorrelated linear discriminant analysis," *IEEE Trans. Knowl. Data Eng.*, vol. 18, no. 10, pp. 1312–1322, Oct. 2006.
[19] L. M. Ang, K. P. Seng, and A. M. Zungeru, "Optimizing energy consumption for big data collection in large-scale wireless sensor networks with mobile collectors," *IEEE Syst. J.*, to be published. [Online]. Available: http://ieeexplore.ieee.org/document/7914656/
[20] C. Lenzen and R. Wattenhofer, "Distributed algorithms for sensor networks," *Philos. Trans. Math. Phys. Eng. Sci.*, vol. 370, no. 1958, pp. 11–26, 2012.
[21] J. Liang, G. Yu, B. Chen, and M. Zhao, "Decentralized dimensionality reduction for distributed tensor data across sensor networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 11, pp. 2174–2186, Nov. 2016.
[22] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Trans. Wireless Commun.*, vol. 1, no. 4, pp. 660–670, Oct. 2002.
[23] F. S. Samaria and A. C. Harter, "Parameterisation of a stochastic model for human face identification," in *Proc. 2nd IEEE Workshop Appl. Comput. Vis.*, Dec. 1994, pp. 138–142.
[24] *Yale Face Database*. Accessed on Mar. 1, 2017. [Online]. Available: http://vision.ucsd.edu/content/yale-face-database
[25] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, "Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection," in *Proc. 4th Eur. Conf. Comput. Vis.*, 1996, pp. 45–58.
[26] K. Seng, L.-M. Ang, and C. Ooi, "A combined rule-based and machine learning audio-visual emotion recognition approach," *IEEE Trans. Affect. Comput.*, to be published. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7506248

JASMINE KAH PHOOI SENG (M'17) received the B.Eng. and Ph.D. degrees from the University of Tasmania, Australia. Before returning to Australia, she was a Professor and the Head of the Computer Science and Networked System Department with Sunway University. Before joining Sunway University, she was an Associate Professor with the School of Electrical and Electronic Engineering, Nottingham University. She is currently an Adjunct Professor with the School of Computing and Mathematics, Charles Sturt University. She has published over 230 papers in journals and international refereed conferences. Her research interests are in intelligent visual processing, multimodal signal processing, artificial intelligence, multimedia wireless sensor network, affective computing, the development of intelligent system, and multimodal big data analytics.

KENNETH LI-MINN ANG (SM'17) received the B.Eng. and Ph.D. degrees from Edith Cowan University, Australia. He was an Associate Professor with University of Nottingham at Malaysia. He is currently a Senior Academic Staff with the School of Computing and Mathematics, Charles Sturt University, Australia, and the Leader of the Intelligent Analytics and Sensing Research Group with CSU. He has published over a 100 papers in journals and international refereed conferences. He is the author of the book *Wireless Multimedia Sensor Networks on Reconfigurable Hardware*. His research interests are in visual information processing, embedded systems and wireless sensor networks, reconfigurable computing, the development of real-world computer systems, large-scale data gathering in wireless multimedia sensor systems, big data analytics for sensor networks, and multimedia Internet of Things. He is a Fellow of the Higher Education Academy, U.K.