Send Orders for Reprints to reprints@benthamscience.ae

The Open Automation and Control Systems Journal, 2014, *6*, 1969-1981

MapReduce-based Parallel Learning for Large-scale Remote Sensing Images

Fenghua Huang*

Sunshine College, Fuzhou University, Fuzhou, Fujian, 350015, China

Abstract: Machine learning applied to large-scale remote sensing images shows inadequacies in computational capability and storage space. To solve this problem, we propose a cloud computing-based scheme for learning remote sensing images in a parallel manner: (1) a hull vector-based hybrid parallel support vector machine model (HHB-PSVM) is proposed. It can substantially improve the efficiency of training and prediction for the large-scale samples while guaranteeing classification accuracy. (2) The MapReduce model is used to achieve parallel extraction of the classification features for the remote sensing images, and the MapReduce-based HHB-PSVM model (MapReduce-HHB-PSVM) is used to implement the training and prediction for large-scale samples. (3) MapReduce-HHB-PSVM is applied to land use classification, enabling various types of land use to be classified more efficiently by using fused hyperspectral images. Experimental results show that MapReduce-HHB-PSVM can substantially improve classification efficiency of large-scale remote sensing images while guaranteeing classification accuracy, and it can promote the machine interpretation of ground objects information extracted from the large-scale remote sensing images to be conducted intelligently.

Keywords: Parallel support vector machine, large-scale remote sensing images, cloud computing, Mapreduce model

1. INTRODUCTION

Processing of large-scale remote sensing data is very resource-intensive and remote sensing images can be easily segmented according to scale. Therefore, remote sensing data can be processed using parallel computing techniques. Machine learning of large-scale remote sensing images is a very complex project which demands a systematic approach. The existing machine learning systems, however, are based on the single machine and the traditional distributed models, and have the limitation of the capabilities in computation and storage. Cloud computing provides parallel processing and distributed storage for huge amounts of information, and it has been widely used for large-scale processing of remote sensing images [1-5]. In cloud-based processing of remote sensing data, the high-performance, highly available and scalable cloud computing techniques are combined with the distributed storage and parallel computing models to quickly process huge amounts of remote sensing data and generate remote sensing information products in batches [5]. The key to implementing cloud-based processing of remote sensing data is developing schemes that can process remote sensing images in a parallel manner by using cloud computing-based parallel processing architecture (e.g. MapReduce), as well as an efficient distributed storage platform for remote sensing data [5]. To approach the computation resource inadequacies in the machine learning of large-scale remote sensing image, we propose a MapReduce-based scheme. It can extract the classification features from the large-scale remote sensing images, train and predict large-scale samples in a parallel manner. In addition to guaranteeing classification accuracy, the proposed scheme can improve the classification efficiency of large-scale remote sensing images, and promote the machine interpretation of ground objects information extracted from the large-scale remote sensing images to be conducted intelligently.

*Address correspondence to this author at the Sunshine College, Fuzhou University, Fuzhou, Fujian, 350015, China;
E-mail: Fenghuait@sina.com

2. HADOOP AND MAPREDUCE

Hadoop is an open-source cloud computing platform and an imitation of the Google computing techniques [6]. The core of Hadoop includes HDFS and MapReduce. The largest advantage of Hadoop when it comes to implementing distributed computing is that it enables efficient local data processing by combining the Hadoop distributed file system (HDFS) with the MapReduce programming model [7,8]. MapReduce is a reduced cloud computing model common among cloud computing platforms. It can execute parallel applications in large-scale distributed clusters and is highly robust and widely available [9]. MapReduce consists primarily of Map and Reduce, which can be used to disassemble tasks and aggregate the intermediate results, respectively. This computing model enables users to compute a large amount of data by easily writing distributed parallel programs.

During the operation of MapReduce, the user program first divides the input file into several file blocks (with a 64 M default block size), and hands these blocks over to the master control program on the Master nodes and the

1874-4443/14

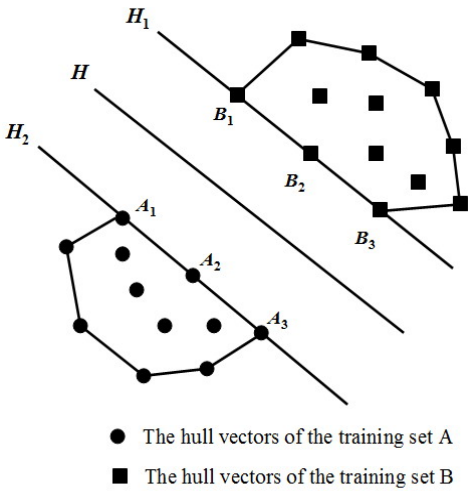


Fig. (1). Convex hulls of the sample set.

execution processing program on other nodes in the cluster [10]. Master will allocate the Map or Reduce tasks according to the loads of nodes in the cluster and the idle nodes will be given priority. The Map tasks on the nodes will process the input data blocks using related functions (these functions can be self-defined) and then output the intermediate results (i.e. the <key, value> pair), and store them into the buffer [11]. The intermediate results in the buffer will be periodically written into the R queues on the local hard disk. The Reduce nodes will then acquire the position information of the <key, value> pairs from the local hard disk and the intermediate results via Master (by calling the remote process, Master can read the intermediate results in the buffer from the hard disk of the local node that is performing the Map task) [11]. The Reduce node will sort all acquired intermediate data according to the key values, and put the data with the same key values into the same group. Finally, the Reduce node will call the user-defined Reduce function to process the sorted intermediate data and temporarily store the output destination file into HDFS. After all Map and Reduce tasks are accomplished, Master node will return the control back to the start of MapReduce program [12].

3. HULL VECTOR-BASED HYBRID PARALLEL SUPPORT VECTOR MACHINE (HV-HYBRID-PSVM)

3.1. Hull Vector

The hull vector (HV) is the minimum convex set (a polyhedron in the feature space) which contains a class of training samples, also known as a convex hull [13]. The idea of the hull vector comes from the geometric significance of the SVM optimal classification hyperplane. That is, to ensure that the minimum distance between two classes of samples is maximized, the support vector can only appear at the very edge of the sample set [14-16]. Let $\boldsymbol{A}$ and $\boldsymbol{B}$ represent the training sample sets that correspond to different classes, respectively. To maximize the minimum distance between them, the support vector point must be within the convex hull. Instead, it can only occur at the convex hull of the sample set labeled in Fig. (**1**).

The hull vector usually contains samples that are at the very edge of the training sample set, i.e. the samples at the convex hull of the training sample set (convex vertexes). It has been proven in [15] that SVM incremental learning with the hull vector rather than the original training sample set produces consistent results [15]. Because the hull vector is usually a part of the entire original training sample set, SVM incremental learning with the hull vector rather than the original training sample set will greatly reduce the number of samples that need to be included in SVM training and increase the training speed [15, 17]. Meanwhile, because the support vector is part of the hull vector, the support vector will not be abandoned during incremental learning, and the hull vector can better represent the classification information in the original training sample set. Therefore, compared with the common SVM incremental learning strategies that adopt the support vector set rather than the original training sample set, the hull vector-based strategy can achieve better classification accuracy without compromising training speed [17]. The hull vector computation method used in this paper is the same with that in [17].

3.2. Hybrid Parallel Support Vector Machine (Hybrid-PSVM)

There are many approaches when it comes to utilizing parallel support vector machines (PSVM) and these methods revolve mostly around the same idea, i.e. partitioning the large-scale training samples according to some criteria, generating local support vector sets by giving each partition SVM training separately, generating a new local support vector set by giving the local support vector sets a combination training or a feedback training again, and then repeating these procedures until the classification accuracy meets the pre-determined threshold. The currently popular PSVM design modes include the layered mode, feedback mode, grouped mode and hybrid mode [18]. To construct a PSVM model suitable for distributed parallel processing, the hybrid PSVM method achieves satisfying training accuracy, speed and parallelism by combing the advantages of other PSVM models [18]. In 2012, Zhang Yiwu and Lin Liang proposed a hybrid distributed parallel SVM model (DHybrid-PSVM) in [19, 20]. This model combines the advantages of the grouped PSVM and the layered PSVM models, and is a representative hybrid parallel SVM model. This model divides the original training sample set into $\boldsymbol{n}$ subsets first, then gives these subsets SVM training to generate their respective support vectors, combines these support vectors and divides them into $\boldsymbol{k}$ subsets, gives these subsets the SVM training to generate their respective support vectors, and finally combines these support vectors to acquire the global support vectors and the final SVM classification model. The final training accuracy of this model is close to the accuracy achieved when giving the original sample set a serial training , but the time consumption is at only $1/r$ of the latter method, where $r=1/(1/n^2+1/k^2)$. Therefore, this model substantially improves the training speed (only second to grouped PSVM). Its training accuracy is higher than that of the grouped PSVM and layered PSVM, but is inferior to what can be achieved by feedback PSVM or serial training [19, 20].

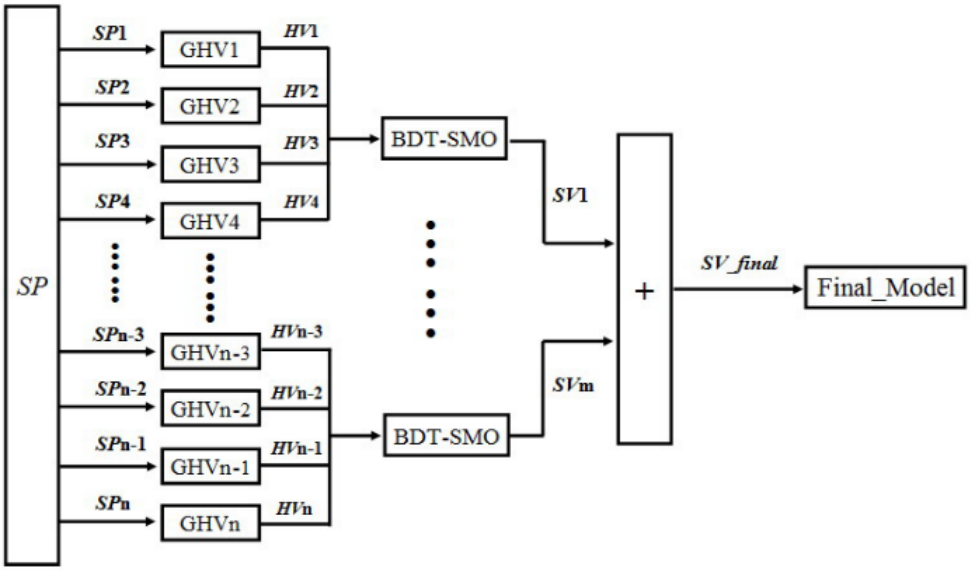


Fig. (2). Parallel training procedure of HHB-PSVM.

3.3. Hull Vector-based Hybrid Parallel SVM (HHB-PSVM)

To further improve the training accuracy of Hybrid-PSVM, we propose a hull vector-based hybrid parallel SVM (HHB-PSVM) in this subsection, as an improvement to the DHybrid-PSVM model introduced in Section 3.2. Fig. (**2**) provides the parallel training procedures of HHB-PSVM, where GHV denotes the hull vector computation method described in [17], BDT-SMO is a multi-category classification method in [21] used for efficiently classifying the fused hyperspectral images.

HHB-PSVM begins by dividing the original training sample set into ***n*** subsets, computes the hull vectors of these subsets via GHV, combines these hull vectors and divides them into ***m*** subsets, gives parallel training to these subsets using BDT-SMO and outputs their respective support vector sets, and finally combines these support vector sets to acquire the global support vector sets and the final SVM classification model. As for classification accuracy, the ordinary support vector is replaced with the hull vector in the first layer of HHB-PSVM, so the loss of classification information contained in the training sample set that is input to BDT-SMO is greatly alleviated, and the final training accuracy of HHB-PSVM is much higher than that of DHybrid-PSVM. As for the training time, although computing the hull vector takes more time than computing the ordinary support vector, the introduction of BDT-SMO can reduce the time spent on computing the support vector set at the second layer of HHB-PSVM. Therefore, the training time consumption of HHB-PSVM is approximately the same as that of DHybrid-PSVM. As for parallelism, both HHB-PSVM and DHybrid-PSVM share the same parallel structure and thus have satisfactory parallelism. To sum up, HHB-PSVM is more suitable than DHybrid-PSVM for parallel training of large-scale samples.

4. MAPREDUCE-BASED PARALLEL LEARNING FOR LARGE-SCALE REMOTE SENSING IMAGES

4.1. Boundary Overlapping-based Method for Segmenting Remote Sensing Images

The boundary overlapping-based segmenting method will be used in this paper to extract features from large-scale remote sensing images in a parallel manner. The optimal granularity will be decided upon via experiments to maximize the efficiency in parallel feature extraction from remote sensing images. To minimize the segmentation complexity, we propose applying a row-based method to the process of geometrically segmenting remote sensing images, one which takes into account the optimal size of the sliding window for textural feature extraction and the final number of partitions. The segmentation procedures are shown in Fig. (**3**).

To avoid the impact of image segmentation on extraction of textural features at the partition boundary, the boundary overlapping method is used to segment the remote sensing images at a fixed size. Finally, the sub-images are recombined by combing and clearing the overlapping parts of the processed sub-images. In Fig. (**4**), A denotes the panoramic remote sensing image, A_1, A_2….A_n denote the sub-images acquired after segmentation, L and h denote the number of rows and columns of A, respectively, a denotes the number of rows of each partition, and k denotes the width of the sliding window. Without taking the boundary effect into consideration, ***A*** can be divided into ***n*** parts

($n=\lceil L/a \rceil$), so the size of each sub-image as follows:

$$S_1 = (a + \left\lceil \frac{k}{2} \right\rceil) \times h \tag{1}$$

$$S_i = (a+k) \times h \ \ (i=2, 3 \ldots n\text{-}1) \tag{2}$$

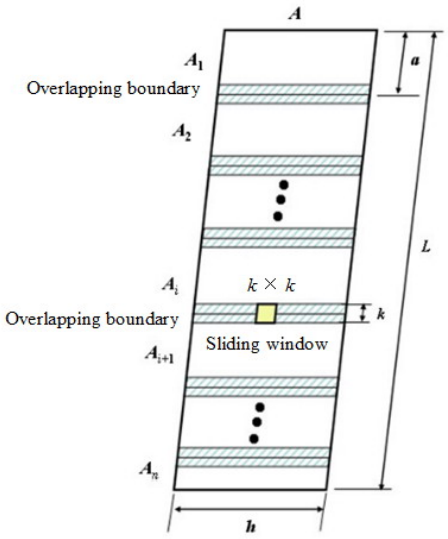


Fig. (3). Boundary overlapping-based method for segmenting remote sensing images.

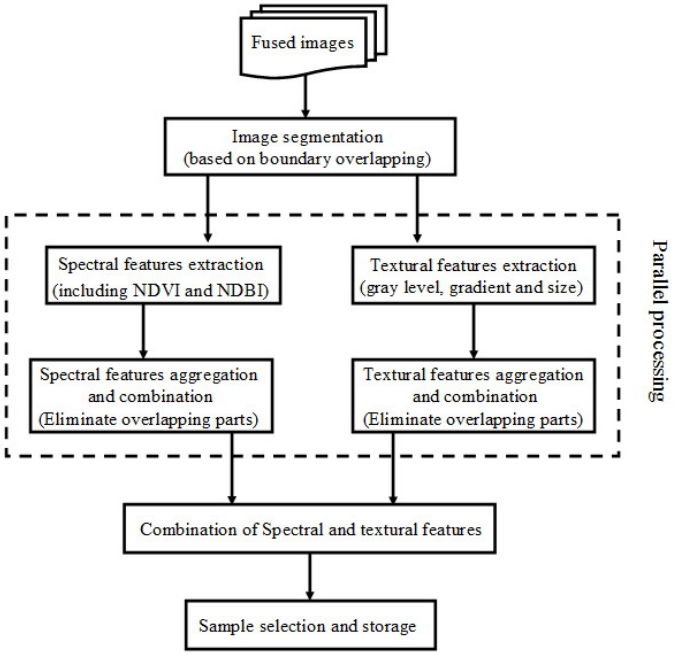


Fig. (4). Scheme for the parallel extraction of classification features.

$$S_n = [\mathrm{mod}(\frac{L}{a}) + \left\lceil \frac{k}{2} \right\rceil] \times h \tag{3}$$

In equation (3), mod( ) represents the remainder computation function. Consider that in Experimental Zone 1, the remote sensing image size is 768×9558, the optimal sliding window size is 15×15. So when k=15, a=150, and n=64, the size of A_i is about 165×768 and the data size is about 500M. A small segmentation granularity (i.e. the value of a is small) means a large value of n and more sub-images. Accordingly, a single node in HDFS will be processed more quickly, but the communication overhead between nodes will increase, and the system's parallel efficiency will deteriorate. This makes the choice of segmentation granularity very important.

4.2. MapReduce-based Scheme for Parallel Extraction of Classification Features From Remote Sensing Images

Extracting pixel-wise features from many images is very complex and burdensome, because the data size and computation complexity is prohibitive, especially when the textural feature extraction requires the window convolution method to be used to compute the level of gray, the gradient and the scale of the central pixels. This is a classical data-intensive problem and imposes high demands on the system's data storage and parallel computation capabilities. Our procedures for parallel extraction of spectral and textural features are shown in Fig. (**4**).

Fig. (**4**) shows that the boundary overlapping-based segmentation has to be performed on the input remote sensing images at a certain granularity in order to extract the classifi-

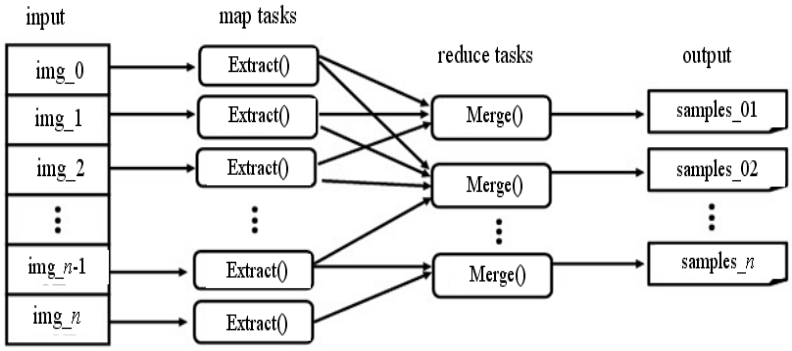


Fig. (5). MapReduce-based procedures for parallel extraction of features from remote sensing images.

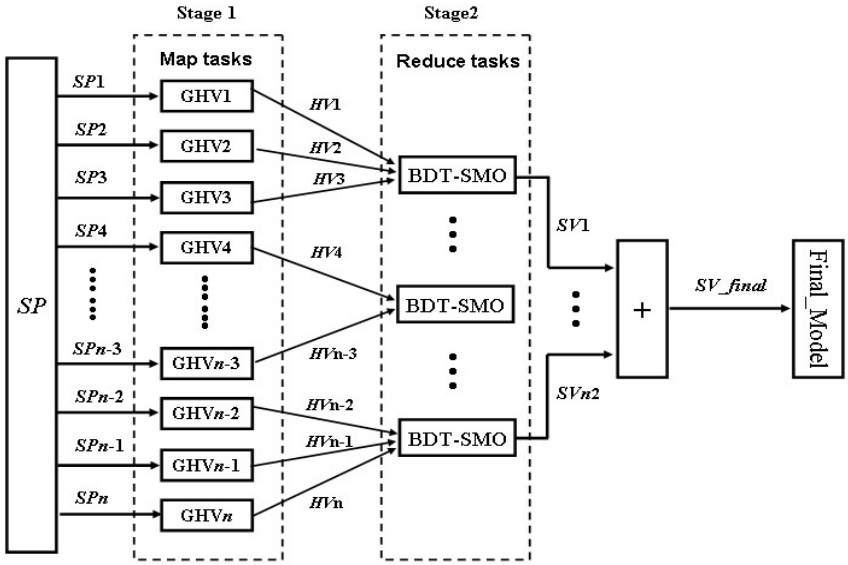


Fig. (6). MapReduce-based HHB-PSVM parallel training strategy.

cation features in a parallel manner. Then, the spectral and textural features of the partitions should be extracted in a parallel manner (textural features are limited to specific wave bands). What is left is to aggregate and combine these extracted spectral and textural features, eliminate the overlapping parts of the partitions, combine the two aggregated types of features, and store the final sample sets into HDFS.

The MapReduce parallel programming model is well suited for parallel extraction of classification features from large-scale images. The MapReduce-based procedures for parallel extraction of classification features from large-scale remote sensing images are shown in Fig. (**5**). The features of each image partition are extracted in a parallel manner using a self-defined Map function of Extract (), which can extract spectral and textural features simultaneously. When an image partition is processed via the Map task, a new unprocessed image will be selected from a set of unprocessed images to extract its features. After all image partitions are processed, the self-defined Reduce function of Merge() will be used to eliminate the overlapping boundaries in the extracted characteristic indexes, combine the two types of features, and output a number of sample set files into HDFS. The sample sets acquired in this way can be further filtered to provide the training sample set, the test sample set and the to-be-predicted sample set.

4.3. MapReduce-based HHB-PSVM Parallel Training Strategy (MapReduce-HHB-PSVM)

PSVM parallel training is conducted in order to improve its training speed and extendibility while guaranteeing classification accuracy. Section 3.2 indicates that when compared with other PSVM models, HHB-PSVM has a satisfying level of classification accuracy, training speed and parallelism. In the MapReduce-based parallel mode, the priority processing for the local data, the distributed file system is very efficient, the mode is highly tolerant and extendable, and there is no frequent and data-intensive communication among nodes in the cluster. Therefore, it is well suited for parallel computation of data from the big data sets. The MapReduce-based HHB-PSVM parallel training strategy is given in Fig. (**6**).

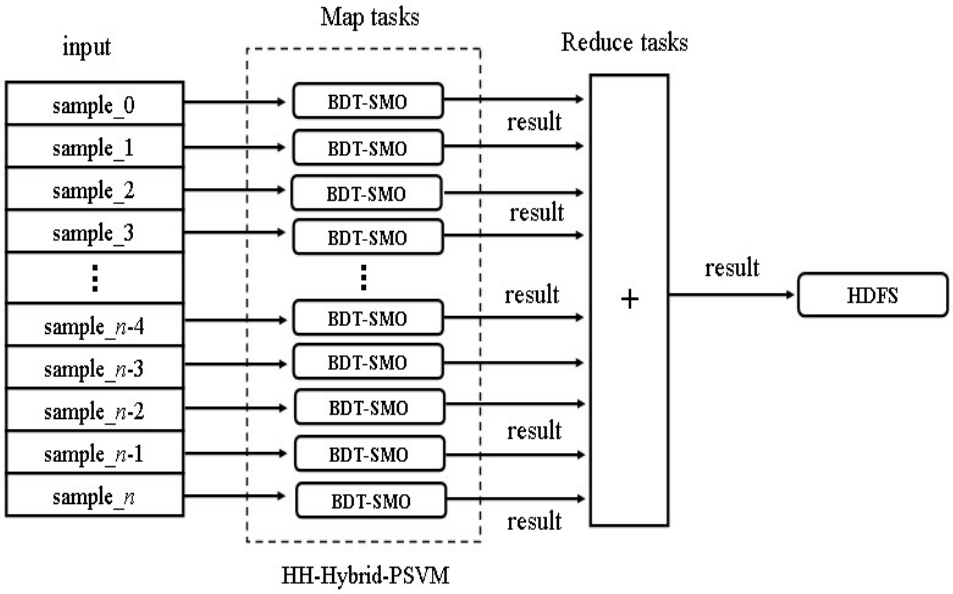
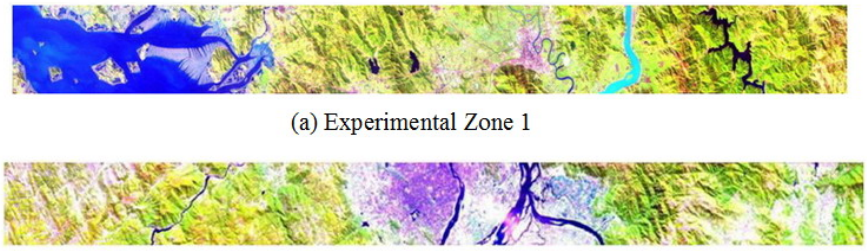


Fig. (7). MapReduce-based strategies for parallel prediction of remote sensing images.



(a) Experimental Zone 1

(b) Experimental Zone 2

Fig. (8). Hyperion hyperspectral images of experimental zones (RGB: 164/48/31).

The entire training process consists of two stages. In the first stage, the original training sample set is divided at a granularity into n sub-sample sets (SP_0~SP_n). The map() function (i.e. GHV function) is used to process the sub-sample sets in a parallel manner and n hull vector sets, HV_1~HV_n. The number of computing nodes used in the first stage is $n_1=n$. The second stage is responsible for combining the hull vector sets from the first stage into n_2 subsets ($n_2=\lfloor \log_2(n_1) \rfloor = \lfloor \log_2(n) \rfloor$), training these subsets separately using the Reduce() functions (BDT-SMO classification functions) to generate the corresponding support vector sets SV_1~SV_{n2}, directly combining SV_1~SV_{n2} into the final support vector set ***SV_final*** and the classification model, ***Final_model.***

4.4. MapReduce-based Strategy for Remote Sensing Images Parallel Prediction

Prediction of large-scale to-be-predicted samples is also a data-intensive problem. The usual approach to this problem involves a decision made for each to-be-predicted samples or pixel and it has great potential to adopt the parallel techniques. Our work involves prediction of large-scale remote sensing images, so finding the means to improve the prediction efficiency is of great importance. The MapReduce-based strategy for parallel prediction of remote sensing images provides a good solution to this problem and its procedures are shown in Fig. (**7**).

First, divide the to-be-measured sample set into n subsets (sample_0~sample_n), and determine the optimal value of n experimentally (the optimal granularity for the to-be-measured sample set). Then we process the n subsets by assigning them to n independent Map() functions (BDT-SMO classification function), and provide the predictions of each subset. Finally, combine these predictions and store the combination into HDFS.

5. EXPERIMENTAL RESULTS AND ANALYSIS

5.1. Experimental Preperaton

(1) Overview of Essential Data and Experimental Zone

Experiments were carried out on two zones to improve the reliability of our land utilization classification results. The Hyperion hyperspectral images and the ALI high-resolution images with the same time phase were collected from the same space-borne platform (EO-1). The Hyperion hyperspectral images of both zones are shown in Fig. (**8**).

Table 1. Samples selected for Experimental Zone 1.

Classes of Land Uses	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}
No. of training samples	6058	4874	8154	5497	7745	6332	6521	6335	5648	9258	7864	5426
No. of test samples	2606	2416	3448	2584	2488	3154	2602	2588	2512	5785	4888	4846

Table 2. Samples selected for Experimental Zone 2.

Classes of Land Uses	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}
No. of training samples	6244	7020	8056	7064	7008	7008	8214	6822	8544	9888	7848	6944
No. of test samples	3654	3321	2948	2865	2660	2530	2956	2485	3452	3968	3210	2568

The Hyperion hyperspectral images of Experimental Zone 1 and the corresponding ALI panchromatic images were collected on March 10, 2003, the size of the images being 258×3176 and 774×9528, respectively. Experimental Zone 1 spans TaiZhou City and the eastern part of Wenzhou City, passing through the Sanmen County, Linhai County, Huangyan County, Wenling County and Yuhuan County of Taizhou City, as well as Leqing County of Wenzhou City from south to north, covering an area of 76.61km^2. The Hyperion hyperspectral images of Experimental Zone 2 and the corresponding ALI panchromatic images were collected on March 26, 2003, the size of the images being 254×3186 and 762×9558, respectively. Experimental Zone 2 spans the middle and eastern part of Fuzhou City, passing through the Luoyuan County, Lianjiang County, Jin'an County, Gulou County, Taijiang County, Changshan County, Minhou County, and Fuqing County from north to south, covering an area of 73.41km^2.

(2) Image pre-processing and fusion

The collected Hyperion hyperspectral image and the ALI panchromatic image of the above zones need to be pre-processed, and the pre-processing procedures involve the unqualified wave bands elimination (the original image has 242 wave bands), radiation value conversion, incorrect line removal, streak recovery, Smile effect elimination, atmospheric correction, and geometric accuracy correction. The aim of pre-processing is to generate the Hyperion image including 134 wave bands based on ground reflectance and the ALI image including 1 Panchromatic-band. The Gram_ Schmidt method is then employed to fuse the two types of images. Because the Hyperion images and the ALI images of the two Zones are obtained by the EO-1 satellite and have the same time phase, the image fusion result is excellent.

(3) Extraction and Combination of Classification Features

Because the distribution of ground objects (e.g. roads, buildings and rivers) in the Experimental Zones shows obvious textural characteristics, the classification features in this paper are the spectral characteristics and the textural characteristics. The "overlearning" problem is not acute for SVM high-dimensional feature learning. With this in mind, as many characteristic indexes as possible are used in this paper to improve classification accuracy. A total of 178 characteristic indexes are used in our work, including 136 spectral characteristic indexes and 42 textural characteristic indexes. The spectral characteristic indexes consist of 134 ground reflectance characteristics, 1 normalized difference vegetation index (NDVI) characteristic, and 1 normalized difference build-up index (NDBI) characteristic. The textural characteristic indexes consist of 6 gray distribution characteristics (second moment, contrast grade, degree of correlation, variance, inverse different moment and information entropy extracted using the gray-level co-occurrence matrix method), 15 gray level gradient characteristics (15 gradient characteristics, including the small gradient advantage, large gradient advantage, and the non-uniformity of the gray distribution, which are extracted using the gray level-gradient co-occurrence matrix method proposed by Hong Jiguang in 1984 [22]), and 21 textural scale characteristics (semi-variogram, energy and averages, all extracted from the 7 high-frequency components after two levels of Wavelet decompositions). It has been proven in [23] that the accuracy and efficiency of classifying fused hyperspectral images can be improved by using the above-mentioned 178 indexes.

(4) Sample Selection

Based on the actual utilization of land, the Experimental Zones are classified into 12 types, including farmland (C_1), woodland (C_2), garden (C_3), grassland (C_4), nudation (C_5), warehouse land (C_6), residential land (C_7), public administration land/ commercial service land/land concerning foreign affairs (C_8), special land (C_9), transportation land (C_{10}), water space (C_{11}) and others (C_{12}). According to field data and a comprehensive analysis of the remote sensing image, the representative areas were selected for ground object sample extraction. The training samples are usually required to be no fewer in number than the testing samples, and the training samples for each class of ground object are also required to be no fewer in number than the characteristic dimensionality of the classes. With the high-resolution images of the Experimental Zones and the 1:10000 land utilization maps, samples that can suitably represent each class of land utilization are selected, as shown in Tables **1** and **2**.

After sample extraction, the training samples and the test samples need to be normalized, respectively, in order to

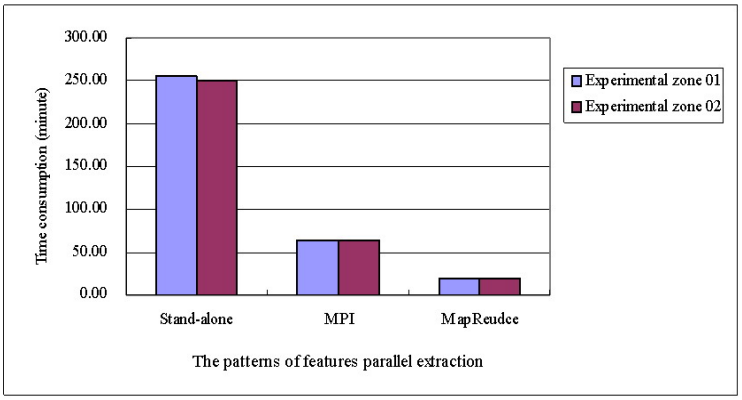


Fig. (9). Comparison of time consumption for extracting classification features from remote sensing images at different parallel modes.

eliminate the differences in various characteristic indexes concerning dimensions, order of magnitude and data dispersion.

5.2. Configuration of Hadoop Cloud Computing Experimental Platform

The Hadoop cloud computing experimental platform adopts the bus topology in this paper, including 1 major Hadoop node (Namenode), 1 backup Hadoop node (Secondary NameNode), 10 Hadoop data nodes (Datanode) and 1 virtual machine server. The CPUs type for all nodes in the platform is 2.50GHZ Intel Dual Core E5200 with 3.2G memory and a 500G hard disk. The operating system on both the NameNode and the Secondary NameNode is Ubuntu Server 12.10. Half of the DataNodes use Red Hat Linux as their operating systems and the other half use Windows XP sp3. The vvirtual machine server operating system is Windows Server 2003 and the virtual machine software is VMware workstation v10.0.

5.3. Experiments on MapReduce-based Parallel Extraction of Classification Features

Distributed parallel computing is usually evaluated by the metrics of speedup ratio and parallel efficiency. The speedup ratio is defined as the ratio of serial computing time to parallel computing time [24] and can be used to measure the acceleration achieved via parallel computing. The parallel efficiency is defined as the ratio of speedup ratio to the total number of processes and can be used to indicate the average speedup ratio of all processes. A high parallel efficiency means that the parallel system is efficient [25]. The parallel efficiency is usually smaller than 1 and is 1 only in the case of linear speedup ratio [26].

(1) Impact of different parallel modes on classification feature extraction time

The theoretical time complexity of classification feature extraction from remote sensing images is O(n), i.e. in theory the time consumption scales linearly to the data size. But in a practical distributed parallel environment, the actual time consumption is subject to network environment and communication overheads, and thus varies depending on the classification methods being applied. Features of the remote sensing images from Experimental Zones 1 and 2 were classified in stand-alone mode, traditional MPI mode and MapReduce mode, respectively. The time consumption is shown in Fig. (**9**).

Fig. (**9**) shows that both the traditional MPI and the MapReduce schemes can accelerate classification feature extraction from remote sensing images. The speedup in feature extraction they provide over the stand-alone mode is 3.98/3.99 and 17.25/12.96, respectively. Obviously, MapReduce enables the classification features of remote sensing images to be extracted at a greater speed, reducing the time consumption from 4 hours to about 20 minutes.

(2) Impacts of the number of partitions on classification feature extraction time

Before performing parallel extraction of classification features from the remote sensing images of the Experimental Zones, the images must be segmented using the boundary overlapping-based method.

Because the segmentation granularity has a big impact on the overall parallel extraction efficiency, the optimal segmentation granularity is determined experimentally in this paper. First, we segment the remote sensing images of Experimental Zones 1 and 2 at different granularities, and then process the images in a parallel manner at the stand-alone, traditional MPI and MapReduce modes, respectively. The methods are evaluated using the metrics of time consumption, speedup ratio and parallel efficiency in order to determine the optimal segmentation granularity for different schemes. Experimental results are shown in Figs. (**9**-**12**) (only the results related to Experimental Zone 2 are presented here due to the similarity between the results of two Zones).

These three figures show that MapReduce provides the least time consumption and the minimum is achieved when there are 20 partitions (about 1/50 of the stand-alone mode). The MapReduce method's speedup ratio is also visibly higher than that of the MPI method, and increases almost linearly

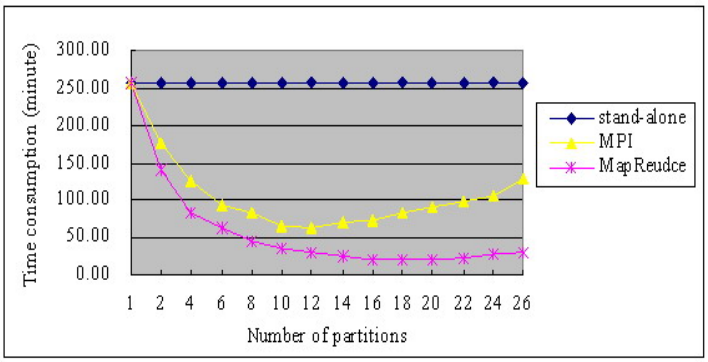


Fig. (10). Comparison of time consumption for extracting classification features from remote sensing images when the images are segmented into a varied number of partitions.

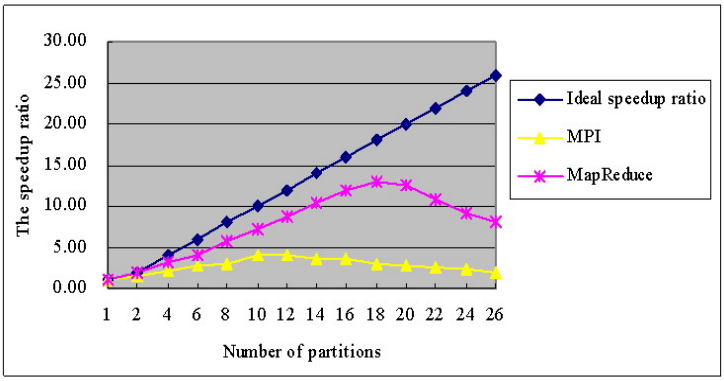


Fig. (11). Comparison of speedup ratio for extracting classification features from remote sensing images when the images are segmented into a varied number of partitions.

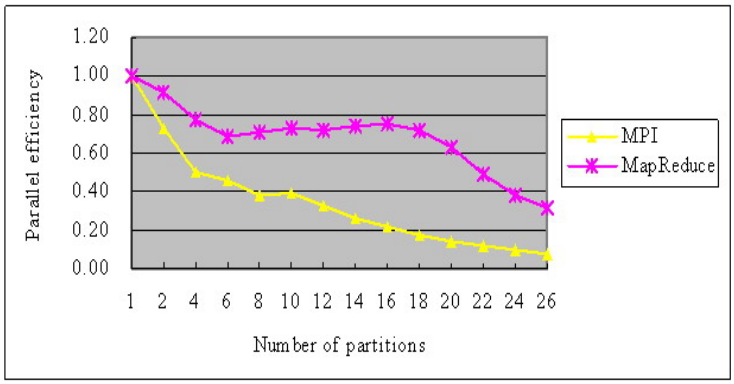


Fig. (12). Comparison of parallel efficiency for extracting classification features from remote sensing images when the images are segmented into different numbers of partitions.

when the number of partition is smaller than 18. The MapReduce method's speedup ratio decreases when there are more than 18 partitions. The reason for this is that parallel efficiency deteriorates due to the growth of communication overhead between nodes with the increase in the number of nodes. The MPI method's speedup ratio decreases when there are more than 10 partitions, because the parallel efficiency deteriorates seriously due to frequent communication and data transfer between nodes with the increase in the number of communicating nodes in the MPI cluster.

(3) Impact of Different Data Sizes on Remote Sensing Image Classification Feature Extraction Time

In the experiments (1) and (2), parallel processing is done after the same remote sensing image is segmented into sub-images at different granularities. In this subsection, parallel extraction is done at the stand-alone, traditional MPI and MapReduce modes, respectively, after the remote sensing images of different sizes are segmented at the same granularity (the default partition size being 64M). The experimental results are shown in Fig. (**13**).

Fig. (**13**) shows that when the data size of remote sensing images is small (less than 10 data blocks), the advantages of MapReduce and MPI are not obvious (their time consumption may be higher than that of stand-alone mode when the

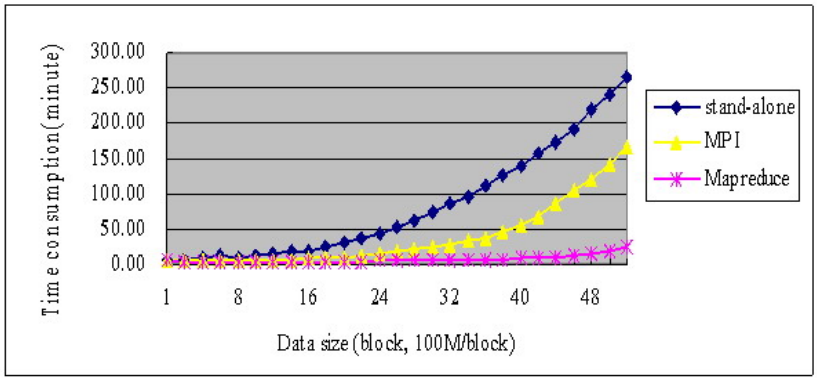


Fig. (13). Comparison of time consumption for classification feature extraction from remote sensing images with different data sizes.

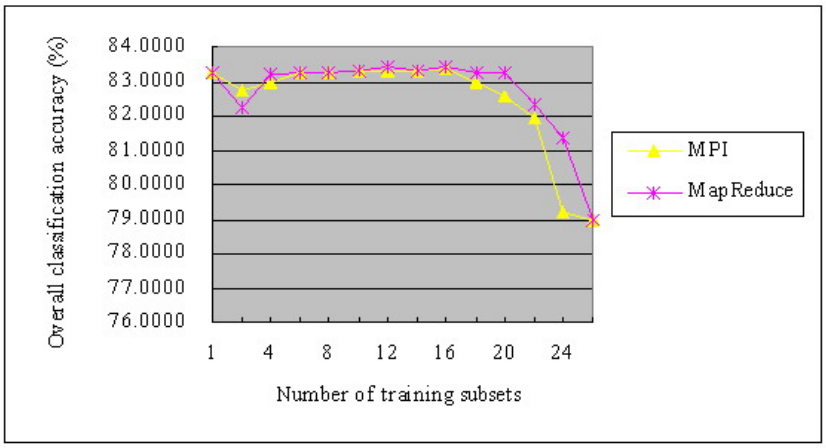


Fig. (14). Comparison of HHB-PSVM classification accuracy when the number of training sub-sets is different.

data size is very small). But in case of huge data size (more than 10 data blocks), the time consumption of the stand-alone mode increases rapidly and exponentially, and is much larger than the MapReduce and MPI modes. In cases where there are over 30 data blocks, MPI mode's time consumption increases rapidly as well, and is much larger than that of the MapReduce mode. The time consumption of the MapReduce mode, however, does not increase considerably.

To sum up, compared with other schemes, the MapReduce method is better suited for parallel processing of huge amounts of data. When used to extract classification features from large-scale remote sensing images, the MapReduce method can provide the highest extraction speed, great speedup ratio and parallel efficiency, and the optimal segmentation granularity is about 1/20 of the area of the Experimental Zone 2's image (data size being approximately 200M).

5.4. Experiments on MapReduce-HHB-PSVM Parallel Training

(1) Analysis of MapReduce-HHB-PSVM classification accuracy

Section 4.3 demonstrates that compared with other PSVM training methods, the classifiers based on HHB-PSVM parallel training provide the highest accuracy, a satisfactory speed and parallelism. Both MPI and MapReduce models are suitable for HHB-PSVM parallel training. To facilitate comparison, the original training sample set is first divided into different numbers of sub-sample sets. The HHB-PSVM parallel training will then be performed at the MPI and MapReduce mode, respectively. Finally, the impacts of the two methods on the overall HHB-PSVM classification accuracy is analyzed. Due to the similarity between the results from Experimental Zones 1 and 2, only the results related to Experimental Zone 2 are shown here in Fig. (**14**).

Fig. (**14**) shows that when the number of training sub-sets is n=1 (indicating stand-alone serial training), the overall classification accuracy is 84.89%. When $2<n\leqq 20$, with the increase in n, the overall HHB-PSVM classification accuracy based on MAP and MapReduce is close to that of serial training and stable. In the case of n=16, the MapReduce-HHB-PSVM classification accuracy reaches the maximum at about 84.90%, slightly greater than the accuracy of serial training at 84.89%. In the case of n>16, the classification accuracy of MPI-based HHB-PSVM (MPI-HHB-PSVM) deteriorates quickly, becoming much smaller than that of MapReduce-HHB-PSVM. When n>20, the classification accuracy of MapReduce-HHB-PSVM deteriorates quickly. The reason for this is that when the number of partitions is too high, the classification accuracy decreases due to an inadequate number of samples in the single training sub-set and the imbalanced sample structure. The classification accuracy of HHB-PSVM is in large part subject to the types of SVM kernels and the PSVM parallel training schemes and is almost immune to the way to implement parallelism.

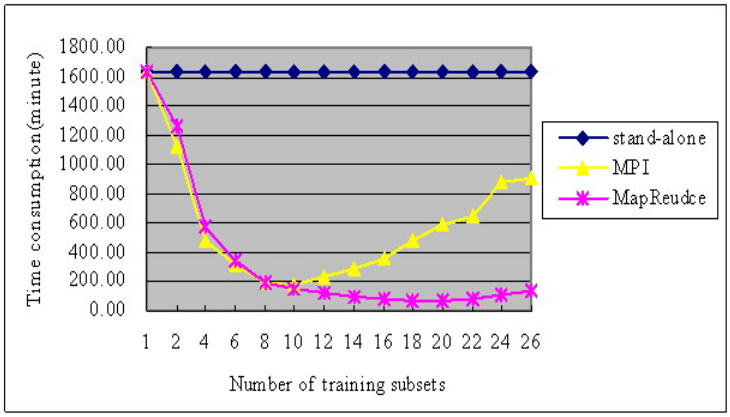


Fig. (15). Comparison of HHB-PSVM training time when the number of training subsets is different.

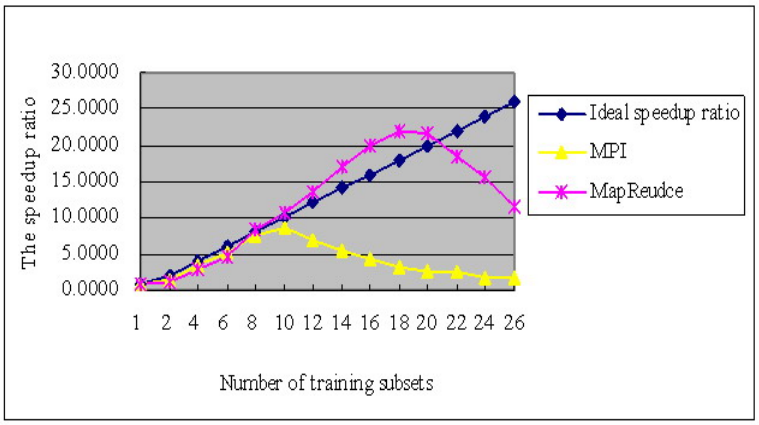


Fig. (16). Comparison of HHB-PSVM speedup ratios when the number of training subsets is different.

(2) Analysis of MapReduce-HHB-PSVM Parallel Training Efficiency

The classification accuracy of the MPI- and MapReduce-based HHB-PSVM methods is approximately close to that of serial training, but the two methods greatly improve the training efficiency. The original training sample set is segmented with different granularities. The respective impacts of the number of subsets acquired after segmentation on the training time, speedup ratio and parallel efficiency of HHB-PSVM are shown in Figs. (**15, 16** and **17**). Due to the similarity between the results of Experimental Zones 1 and 2, only the results related to Experimental Zone 2 are shown below.

Fig. (**15**) shows that compared to the stand-alone mode, the MPI and MapReduce methods can both effectively improve the training speed of HHB-PSVM. When the number of training sub-sets is n>10, the time consumption of the MapReduce-based parallel training is much lower than that of the MPI and the stand-alone methods. When n=18, MapReduce-HHB-PSVM consumes the least amount of training time.

Fig. (**16** and **17**) show that when the number of training subsets n<7, the speedup ratio of the MPI and MapReduce methods is less than the ideal ratio. The speedup ratio and parallel efficiency of MPI-HHB-PSVM is higher than that of MapReduce-HHB-PSVM. But when n>7, the speedup ratio and parallel efficiency of MapReduce-HHB-PSVM increase rapidly, and are much higher than that of MPI-HHB-PSVM. In the case of n>10, the speedup ratio and parallel efficiency of MPI-HHB-PSVM deteriorate enormously, due to frequent communication and data transfers between nodes when there are many communication nodes in the MPI cluster. In the case of n>18, the speedup ratio and parallel efficiency of MapReduce-HHB-PSVM deteriorate enormously. The reason for this is that task scheduling, copy duplicating and data communication occurs more frequently with the increase in the number of training subsets, resulting in more communication overhead.

When there are 18 training subsets, MapReduce-HHB-PSVM provides the highest training speed, satisfying speedup ratio and parallel efficiency, the segmentation granularity of parallel training being about 5,000 training samples per subset (i.e. the total number of training samples from the experimental zone divided by the number of training subsets). That is, each class (up to a total of 12) has 420 training samples in average, about 2.36 times the number of characteristic indexes (178), meaning that the number of training samples meets the basic requirements for SVM training.

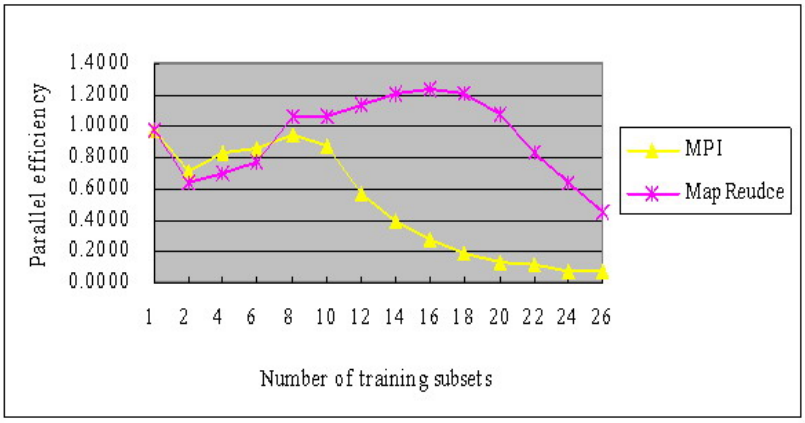


Fig. (17). Comparison of HHB-PSVM parallel efficiency when the number of training subsets is different.

5.5. Experiments on MapReduce-based Parallel Prediction of Remote Sensing Images

In predicting the to-be-measured sample set, fixed related classifiers need to be used to decide on each sample or pixel of the to-be-measured sample set. Its theoretical time complexity is O(n), meaning that its time consumption is linear to the size of the to-be-measured data. Similar to the classification feature extraction in Section 5.3, the MapReduce method is also well suited for large-scale to-be-measured sample parallel prediction. Parallel prediction experiments were performed on the to-be-measured sample sets from Experimental Zones 1 and 2 using the BDT-SMO classifiers acquired via parallel training in Section 5.4. The impacts of different parallel modes (single machine, MPI and MapReduce), different numbers of subsets obtained after segmentation and different data sizes on the time consumption, speedup ratio and parallel efficiency of parallel prediction for to-be-measured sample sets were analyzed. Details are omitted here due to the similarity of the experimental results with the results of parallel extraction of classification features from remote sensing images in Section 5.3. Only the final results related to Experimental Zone 2 are given here: compared with MPI, the MapReduce method is the more suitable one for large-scale to-be-measured sample parallel prediction, because the MapReduce method provides the greatest prediction speed (about 15 times that of the single machine scheme) and stisfactory speedup ratio and parallel efficiency. It is appropriate to acquire 36 subsets after segmentation for parallel prediction, i.e. the segmentation granularity is about 110M per partition.

CONCLUSION

In order to improve the machine learning efficiency for large-scale remote sensing images while guaranteeing classification accuracy, a hull vector-based hybrid parallel support vector machine model(HHB-PSVM) is proposed, which uses the cloud computing and parallel SVM techniques. We also propose a MapReduce-based HHB-PSVM distributed parallel model(MapReduce-HHB-PSVM), which can extract the classification features from large-scale remote sensing images and perform learning and prediction on the large-scale sample sets in a parallel manner. Experimental results show that compared with the stand-alone (serial) mode and the traditional MPI mode, the proposed MapReduce-HHB-PSVM parallel learning scheme can greatly improve the learning efficiency for the large-scale remote sensing images, while at the same time guaranteeing classification accuracy, and promote the machine interpretation of ground objects information extracted from the large-scale remote sensing images to be conducted intelligently.

CONFLICT OF INTEREST

The authors confirm that this article content has no conflicts of interest.

ACKNOWLEDGMENTS

This work is supported by the Science and Technology Project of Fujian Province Education Department of China (NO. JA13364).

REFERENCES

[1] H.C. Bao, L. Fang, and R.Y. Liu, ,"Research and application of the storage way of land use change records", *Journal of Zhejiang University (Science Edition)*, vol.38, no.2, pp. 218-222, 2011.

[2] J.B. Lai, X.L. Luo, T. Yu and P.Y. Jia, "Remote sensing data organization model based on cloud computing", *Computer Science*, vol. 40 ,no.7, pp. 80-84, 2013.

[3] H.P. Yang, Z.F. Shen, J.C. Luo and W. Wu, "Recent developments in high performance geocomputation for massive remote sensing data", *Journal of Geo-Information Science*, vol.15, no.1, pp.128-136, 2013.

[4] Y. Liu, W. Guo, W.S. Jiang and J.Y. Gong, "Research of remote sensing service based on cloud computing mode", *Application Research of Computers*, vol.26, no.9, pp. 3428-3431, 2009.

[5] F.H. Ren and J.N. Wang, "Turning remote sensing to cloud services: technical research and experiment", *Journal of Remote Sensing*, vol.16, no. 6, pp. 1339-1346, 2012.

[6] B. Wan and L. Yang, "Data center: GIS function warehouse", *Earth Science-Journal of China University of Geosciences*, vol.35, no.3, pp.357-361, 2010.

[7] L.J. Zhang, F. Huan and Y.D. Wang, "Parallel image processing implementation under hadoop cloud platform", *China Information Security*, no.10, pp.59-62, 2012.

[8] D. Jeffrey and G. Sanjay, "MapReduce: simplified data processing on large clusters", *Communication of the ACM*, no.9, pp.107-113, 2008.

[9] S.K. Hao, "Brief analysis of the architecture of hadoop HDFS and mapreduce", *Designing Techniques of Posts and Telecommunications*, no.7, pp. 37-42, 2012.

[10] Y.S. Fan, "Comparison of two essential databases in cloud computing", *Journal of Mianyang Normal University*, vol. 31, no. 8, pp. 68-71, 2012.

[11] Y. Liu, L. Chen, N. Jing and W. Xiong, "Parallel batch-buildiing remote sensing images tile pyramid with mapreduce", *Journal of Wuhan University (Information Science Edition)*, vol. 38, no. 3, pp. 278-282, 2013.

[12] G.Z. Sun, F. Xiao and X. Xiong, "Study on scheduling and fault tolerance strategy of mapreduce", *Microelectronics & Computer*, vol. 9 , no. 24, pp.1-2, 2007.

[13] Y.H. Liu and L.Q. Huang, "Parallel support vector machines based on cloud computing", *Journal of Nanyakn Institute of Technology*, vol.3, no. 2, pp. 26-29, 33, 2011.

[14] D.H. Li, S.X. Du and T.J. Wu, "Fast incremental learning algorithm of linear support vector machine based on hull vectors", *Journal of Zhejiang University (Engineering Science)*, vol. 40, no.2, pp. 202-206, 2006.

[15] W.H. Ceng and J. Ma, "A new incremental learning algorithm for support vector machine", *Journal of Xiamen University (Natural Science)*, vol. 41, no. 6, pp. 687-691, 2002.

[16] J. Tan and F.Xu, "Gradual incremental learning algorithm of support vector machine based on hull vector", *Journal of South-Central University for Nationalities*, vol.30, no.3, pp. 94-97, 2012.

[17] B. Wen, G.L. Shan and X.S. Duan, "Research of incremental learning algorithm based on KKT conditions and hull vectors", *Computer Science*, vol.40, no.3, pp.255-258, 2013.

[18] X.M. Jin and Y.M. Wen, "On parallel learning based on support vector machines", *Journal of Hunan University (Natural Science)*, vol. 35, no.7, pp. 74-79, 2008.

[19] R.Y. Li and H.P. Li, "Research and application of improved parallel SMO", *Computer Engineering and Design*, vol. 30, no. 22, pp. 5162-5165, 2009.

[20] Y.W. Zhang, "*SVM algorithm optimization and application based on hadoop distributed platform*", M.S. thesis, Zhongshan University, Guangzhou, China, 2012.

[21] F. Huang, "Research on classification of hyperspectral remote sensing imagery based on BDT-SMO and combined features", *Journal of Multimedia*, vol.9, no.3, pp.456-462, 2014.

[22] J.G. Hong, "Gray level-gradient co-occurrence matrix method for textural analysis", *Acta Automatica Sinica*, vol. 10, no. 1, pp. 22-25, 1984.

[23] F. Huang and L. Yan, "Combined kernel-based BDT-SMO classification of hyperspectral fused images", *The Scientific World Journal*, vol. 2014, Article ID 738250 (13 pages), 2014.

[24] H. Shen, X.C. Shi and H.H. Qiu, "Study on parallel processing technology of massive remote sensing image based on MPI", *Gnss World of China*, vol.37, no.6, pp.73-76, 2012.

[25] L. Liu, "Design of Real-time Processing Platform for Satellite Remote Sensing Data Based on MPI", *Spacecraft Engineering*, vol. 22, no. 3, pp. 130-134, 2013.

[26] M. Yang, Q.S. Guo and H.T. He, "MPI-based Parallel Processing of Common Oceanic Numerical Model", *Experimental Technology and Management*, no. 5, pp. 257-259, 2011.

Received: September 22, 2014 Revised: November 03, 2014 Accepted: November 06, 2014