Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2020) 28: 182 – 195
© TÜBİTAK
doi:10.3906/elk-1904-7

Fast texture classification of denoised SAR image patches using GLCM on Spark

Caner ÖZCAN[1,*], **Okan ERSOY**[2], **İskender Ülgen OĞUL**[3]
[1]Department of Computer Engineering, Faculty of Engineering, Karabük University, Karabuk, Turkey
[2]School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, USA
[3]The Graduate School of Engineering and Sciences, İzmir Institute of Technology, İzmir, Turkey

Received: 01.04.2019 • **Accepted/Published Online:** 26.08.2019 • **Final Version:** 27.01.2020

Abstract: Classification of a synthetic aperture radar (SAR) image is an essential process for SAR image analysis and interpretation. Recent advances in imaging technologies have allowed data sizes to grow, and a large number of applications in many areas have been generated. However, analysis of high-resolution SAR images, such as classification, is a time-consuming process and high-speed algorithms are needed. In this study, classification of high-speed denoised SAR image patches by using Apache Spark clustering framework is presented. Spark is preferred due to its powerful open-source cluster-computing framework with fast, easy-to-use, and in-memory analytics. Classification of SAR images is realized on patch level by using the supervised learning algorithms embedded in the Spark machine learning library. The feature vectors used as the classifier input are obtained using gray-level cooccurrence matrix which is chosen to quantitatively evaluate textural parameters and representations. SAR image patches used to construct the feature vectors are first applied to the noise reduction algorithm to obtain a more accurate classification accuracy. Experimental studies were carried out using naive Bayes, decision tree, and random forest algorithms to provide comparative results, and significant accuracies were achieved. The results were also compared with a state-of-the-art deep learning method. TerraSAR-X images of high-resolution real-world SAR images were used as data.

Key words: Classification, machine learning, synthetic aperture radar, cluster computing, naive Bayes, decision tree, random forest

1. Introduction

Synthetic-aperture radar (SAR) is a radar system for high-resolution earth imaging and moving target detection that can be used on manned and unmanned aerial platforms [1]. High-resolution SAR images obtained from imaging systems such as TerraSAR-X and Sentinel provide content-based information. SAR systems have been used extensively for high resolution mapping and other remote sensing applications in the last two decades both in military and civilian areas due to their weather-independent and day-and-night features. Many studies with SAR images usually involve applications such as classification [2], target recognition [3], segmentation [4], and change detection [5]. Common applications include environmental and earth system monitoring, geoscience and climate change research, moving target and change detection, military surveillance and discovery, agriculture and forestry research, urbanization and illegal settlement follow-up, to security-related applications.

Due to high resolution, man-made objects as well as vegetation and ground classes can be detected in the images. SAR image classification involves each pixel to be assigned a class label according to some characteristics.

*Correspondence: canerozcan@karabuk.edu.tr

Various applications such as resource and environmental monitoring [6], agriculture and hydrology modeling [7], and urban planning [8] are carried out in the SAR image classification area.

The sizes of SAR data are rapidly increasing due to new satellite missions, high-quality sensors, and availability of public data archives. As the size of the data grows, resources become insufficient to store and process the big data, and new studies are being carried out to deal with it. In one study, a solution to solve the big data problem was presented in the context of satellite image processing [9]. A Bayesian algorithm based on information mining was developed by combining the unsupervised clustering results of different properties with a specific semantic concept at the pixel level for data fusion on SAR big data [10]. In another study, a classification method for big size SAR images was described using spatial covariance information based on texture to directly classify images in a supervised approach [11]. A semiautomated and fast semantic annotation on large set of TerraSAR-X archives which contains high-resolution SAR image patches was demonstrated. Analysis results provided that the annotation procedure ensures a robust system for real user requirements with interactive visualization tools [12].

These studies also show that problems still persist in the realization of processing required with big SAR images. Because of these problems, memory-efficient computing tools that can handle big-sized images can be quickly used. Among these computing tools, Apache Spark is a fast, in-memory data processing engine with an expanding usage to efficiently process big SAR images. For example, a distributed SAR image change detection method using Spark was proposed to provide less execution time within in-memory cluster computing framework [13]. The proposed framework can fully exploit the capabilities of a cluster system to handle big SAR images.

Relevant references are summarized next. Distributed parallel clustering algorithm which ensures change detection based on Spark framework was developed [14]. Using Map-Reduce framework, the degree of membership was first provided with the Map stage, then cluster centers are provided in parallel during the Reduce stage. An OpenCL-enabled Spark framework to accelerate Kernel Fuzzy C-Mean algorithm, which has computationally intensive operations, was presented for SAR change detection [15]. In addition to these, different studies were also implemented using Apache Spark framework to process SAR images fast and efficiently [16, 17].

An important issue in such studies is the necessity of noise filtering to increment quality of SAR images. SAR images have a typical, multiplicative, and randomly structured speckle noise. Speckle noise has negative effects on the fine detail properties of the images obtained, and it needs to be reduced especially in remote sensing applications. Many noise reduction methods with different features are proposed in the literature [18–20]. Sparsity-driven despeckling (SDD) method [20] was used in this study. This method depends on a total-variation approach employing different norms by providing successful smoothing in homogenous regions while maintaining properties such as edge and point scatterers. Speckle filtering is thus utilized as a step prior to creating the feature vectors.

Feature extraction is the process of generating feature vectors which allow better performance in image classification. One widely used method for feature extraction called gray-level cooccurrence matrix (GLCM) was proposed by Haralick et al. [21]. GLCM has recently been used in pattern recognition and texture analysis with remote sensing data. This method is applied to denoised images to obtain feature vectors. Many studies have been carried out using GLCM on SAR images. A preliminary research for mapping sea ice texture with SAR was presented using GLCM matrices to quantitatively evaluate textural properties [22]. Another method for water region extraction in SAR images uses GLCM-based features and the support vector machine classifier

[23]. The properties of water and nonwater areas were clearly illustrated by the GLCM-based features which were provided as inputs to the SVM classifier.

Pixel-based SAR image classification with current SAR data may be time-consuming [24]. For this reason, applying SAR image classification at patch level offers an alternative solution. Previously, we had a preliminary study [25] classifying noisy SAR images using only the naive Bayes method. In this study, fast classification of high-volume denoised SAR images is realized on patch level by using different machine learning algorithms. The SDD method is used on the input SAR images for successful noise reduction and then GLCM is proposed as a superior feature extraction technique that effectively represents the texture representing the spatial interaction of pixels in denoised SAR images. The classification process is achieved by using the naive Bayes (NB), decision tree (DT), and random forest (RF) trained with the image features extracted by the GLCM. The results were also compared with a state-of-the-art deep learning method. SAR image dataset consisting of different classes such as road, alongside, water, building, and vegetation areas was prepared by using public TerraSAR-X spotlight image archives.

2. Apache Spark clustering system

Apache Spark, as an advanced version of the Hadoop system, is an open-source, fast, and generic framework developed with Scala that enables work in parallel on large datasets [26]. Unlike Hadoop operating system, which reads and writes data on disk in each of the data processing steps, Spark is faster in data processing because it is optimized to work in-memory. Spark provides a speed increase of almost hundred times due to the transaction processing approach through the system memory. It operates on the Java Virtual Machine installed on the computer and has no platform dependency. Architecture diagram of Apache Spark is shown in Figure 1. Spark provides a general machine learning library that contains common machine learning functionalities for simplicity, scalability, and easy integration with other tools. Within the MLlib, there are many common machine learning and statistical approximation methods such as regression, classification, clustering analysis, collaborative filtering, and feature extraction as well as supporting functionality such as model evaluation and data import. Designed to run in parallel on clusters, MLlib is accessible from all Spark programming languages.

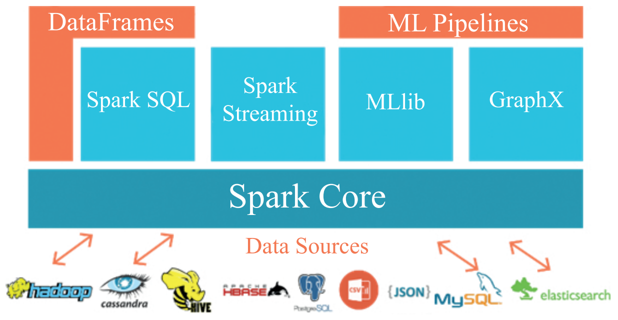

Figure 1. Architecture diagram of Apache Spark.

A Spark application runs as independent sets of processes using the service structure called the cluster manager on a group of machines. Spark performs tasks such as deployment, operation, and monitoring of the application using in-cluster manager named Standalone cluster manager. The main task of cluster manager is to provide resources to all applications. Spark Standalone system provides a simple method to run applications on a cluster. In this system, there is a single master and many workers, each with a configured amount of memory and CPU cores. When an application is run, one specifies how much memory the processors will use, as well as the total number of cores in all the processors [26]. A standalone cluster can be launched either manually, by starting a master and workers by hand, or by using custom prepared launch scripts. The Spark Standalone clustering system is shown in Figure 2. Spark Worker node executes the given code task assigned by the driver and Spark Master provides resources by tracking progress of workers. Spark Driver is a client or master node application which asks resources from a cluster manager.

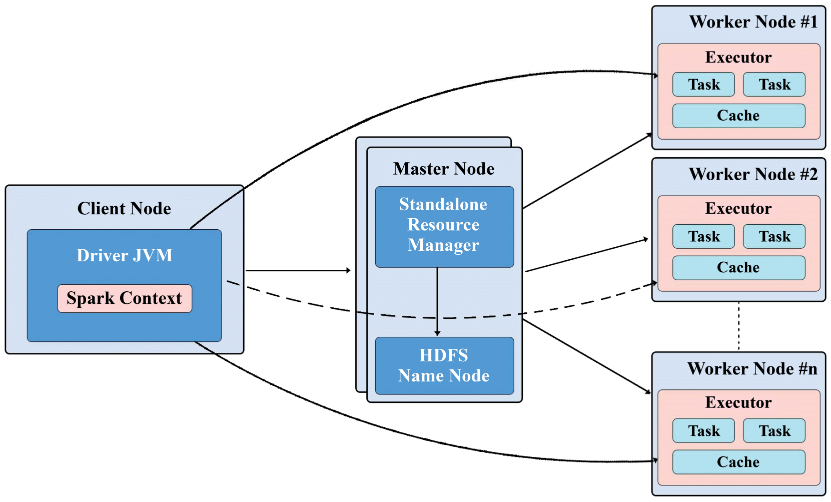

Figure 2. Spark Standalone clustering system.

3. Method design

In this section, we first describe the speckle noise filter applied to SAR images. Next, we describe the GLCM method for extracting feature vectors. Finally, we explain the details of the MLlib methods used for classifying image patches.

3.1. Speckle noise filtering

Speckle noise filtering to increment quality of SAR images is undertaken first. Speckle reduction is crucial and is used as a preprocessing step. This step facilitates the interpretation and analysis of images. In this study, we used the SDD method [20] as a despeckling filter. It is capable of using different norms controlled by a single parameter and provides better or similar denoising compared to other algorithms (PPB, SRAD, SAR-BM3D, and FPD) with shorter execution times [20]. According to this method, the SAR image denoising problem is defined as the following optimization problem [20]:

$$\hat{F} = arg \min_{F} J(F), \tag{1}$$

where the cost function $J(F)$ to be minimized is defined as

$$J(F) = \sum_{p} (F_p - G_p)^2 + \lambda \wedge (|(\partial F)_p|, f), \tag{2}$$

where G is the observed speckled image, F is the despeckled image, p is the pixel index number in the image, λ is the smoothing level, f is the norm value to be applied within TV regularization, and $\wedge(x, f)$ is the exponentiation operator defined as

$$\wedge(x,f) = \left\{ \begin{array}{cc} 0 & x=0, f=0 \\ x^f & otherwise \end{array} \right\} \tag{3}$$

for obtaining l_0-norm when f is 0. In the cost function, the first term is defined as a fidelity term which ensures F to be similar to G. The second term is a TV regularization term, which implies a penalty on the changes in image gradients. In this formulation, l_0-norm is obtained when f is 0, l_1-norm is obtained when f is 1, and the fractional norm is obtained when f is between 0 and 1. They all induce a sparse solution.

3.2. GLCM Feature Extraction

One of the simplest approaches for texture classification is the use of statistical quantities of histograms of an image or region. Although the use of the histogram gives information about the distribution of the densities, information about the relationship of the positions of the pixels in the texture is not provided. A statistical approach, GLCM, by Haralick et al. has been generally utilized to extract various texture features for various implementations and has proven to be a very successful texture identifier with satisfying results [21]. The GLCM effectively represents the texture pattern by computing how frequently a gray-level pixel i shows up in a particular spatial relationship with a gray-level pixel j in the image.

GLCM is a two-dimensional dependency matrix that considers the spatial relationship between adjacent or neighboring pixels. The GLCM method, which follows second-order statistics, determines the spatial relationship between pixels by computing the intensity differences between the center pixel and its neighbors. Due to the use of second-order probabilities, it provides better performance than other traditional methods in different applications. The number of rows and columns of the cooccurrence matrix depends only on the gray levels in the image regardless of the image size. GLCM is constructed by calculating how many times a pixel with value i is adjacent to a pixel horizontally with value j. The distance between the related pixel and its neighbor is called as offset. When calculating the GLCM, different offsets and angles around the respective pixel can be taken into account. In this study, the offset was chosen as $d=1$ and the four directions $0°$, $45°$, $90°$, and $135°$ were used as shown in Figure 3.

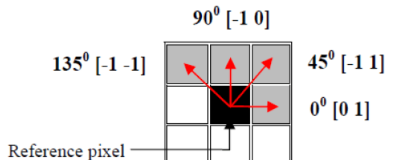

Figure 3. Cooccurrence matrix directions for extracting texture features.

The structure of the GLCM depends on two parameters. These parameters are the relative distance (d) and orientation (ϕ) between pixels in pairs. Here, d is expressed as an integer distance and ϕ is generally quantized in four directions (horiz.-0°, diag.-45°, vert.-90° and antidiag.-135°). The GLCM $C_{m,n}$ can be stated using distance d and direction ϕ as follows:

$$C_{i,j} = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} P\{I(X,Y)=i \ \& \ I(x \pm d\Phi_1, y \pm d\Phi_2) = j\}, \tag{4}$$

where P$\{\cdot\}$ is equal to 1 if the argument is true, and P$\{\cdot\}$ is 0 otherwise.

3.3. Spark MLlib algorithms

MLlib is a Spark machine learning library that makes machine learning scalable and easy. In this study, naive Bayes, decision tree, and random forest algorithms are used in the MLlib which allows fast and parallel processing on large datasets. The pseudocode of the proposed method is as follows:

Algorithm 1 Algorithm of the Proposed Model

```
Input_Training: Alongside, Road, Water, Vegetation, Building
Input_Analysis_Data: 1K to 10M Raw Datasets
Output: Classification Category Ratios
  Pre_Process:
    RDD: CVS to RDD
    Labeled_Data: LabeledPoint(key, string) for all training data
    Training_Data: union for all (Labeled_Data)
  Model_Training:
    NB_Model: Train (Training_Data, Lambda)
    NB_Accuracy: 70% - 30% Metrics && 10-Fold Cross Validation Metrics
    DTree_Model: Train (Training_Data, Tree_Depth)
    DTree_Accuracy: 70% - 30% Metrics && 10-Fold Cross Validation Metrics
    RFTree_Model: Train (Training_Data, Tree_Depth, Tree_Size)
    RFTree_Accuracy: 70% - 30% Metrics && 10-Fold Cross Validation Metrics
    Return: List (Models)
  Call Analysis Section: Parameters (Models && Input_Analysis_Data)
    foreach: Input_Analysis_Data
      prediction= model(pos_prob >neg_prob) ? pos : neg
      if Predict: true
        LongAccumulator++
      Print: Classification Ratios for Each Class & Model
  Visualization: All Results and Corresponding Datasets
    Visualize: Chart (Accuracy, Data_Size)
    Visualize: Chart (Time, Data_Size)
  End Analysis
```

3.3.1. Naive Bayes in Spark MLlib

Naive Bayes algorithm is a multiclass ML algorithm based on probability principles with strong independence assumptions for each feature pair. Creating the NB model is easy because it does not require complex iterative parameter estimation; thus, it offers a suitable substructure especially for very large datasets. In the NB classification, data that has already been assigned to a class label are passed to the system at a certain rate. The conditional probability distribution for each feature is computed with the probabilistic operations performed on the training data. Then Bayes' theorem is applied to compute the probability distribution of label given an observation. The test data is evaluated by using previously obtained probability values and the category of the test data is predicted. It can provide reasonable performance in many cases when appropriate feature engineering is used. NB classifier is given in the following steps [27]:

The aim of the algorithm is to maximize the probability of the target class given the x features. C is the possible outcome of k classes, $C_1, C_2, ..., C_k$. Each sample is represented by an n-dimensional vector, $X = \{x_1, x_2, ..., x_n\}$. Given an unknown data sample X, without class label, the classifier will predict X belongs to the class having the highest posterior probability (P), conditioned on X. It is only possible that X belongs to class C:

$$P(C_i|X) > P(C_j|X) \qquad for\, 1 \leq j \leq m,\, j \neq i \tag{5}$$

The class C_i which maximized $P(C_i|X)$ is named as the maximum posteriori hypothesis. According to Bayes' theorem,

$$P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)} \tag{6}$$

If $P(X)$ is the same in all classes, it is aimed to maximize the dividend only. If $P(C_i)$ probabilities are unknown, classes are assumed to be equal, and then we just maximize $P(X|C_i)$. When many sets of data are given, computing $P(X|C_i)$ will be computationally expensive. Reduction of the computational complexity in the evaluation of $P(X|C_i)P(C_i)$ is only done with the naive assumption of class conditional independence using

$$P(X|C_i) \approx \prod_{k=1}^{n} P(x_k|C_i) \tag{7}$$

We can easily estimate the probabilities $P(x_1|C_i)$, $P(x_2|C_i)$, ..., $P(x_n|C_i)$ from the training set. By evaluating $P(X|C_i)P(C_i)$ for each class C_i, the X class label is estimated. The predicted class label is the class C_i for which $P(X|C_i)P(C_i)$ is the maximum.

NB classifiers generally do not perform as well as more complex models because they use rigid assumptions about the data. However, they are quite fast for both training and prediction. They have very few tunable parameters. These advantages constitute valid reasons for choosing NB in our work.

3.3.2. Decision tree in spark MLlib

Decision tree is a popular method for machine learning tasks in classification. DT is widely used since it is easily interpreted, handles categorical features, provides multiclass classification, does not require feature scaling, and is able to capture nonlinearities [28]. The DT is a greedy algorithm that constructs the trees in a top-down recursive way of divide and conquer manner by applying a series of decision rules. A tree structure is created, and the class labels are expressed at the leaves of the tree. A final tree predicts the same label for all instances that reach the leaf node. Each section is selected by choosing the best split from a set of possible splits to maximize the gain of information on a tree node. The split chosen in each tree node is obtained by calculating the set $arg\max_s IG(D,s)$ which is the maximization of information gain when a split s is applied to a dataset D. Two different impurity measures (Gini impurity and entropy) is offered to classify dataset. The Gini impurity is given by [29]:

$$\sum_{i=1}^{C} f_i(1-f_i), \tag{8}$$

where C is the number of unique labels, and f_i is the frequency of label i at a node. The impurity measure for entropy is defined by:

$$\sum_{i=1}^{C} -f_i log(f_i). \tag{9}$$

The information gain is based on subtracting the parent node impurity from the weighted sum of the two child node impurities. The information gain is defined by:

$$IG(D,s) = Impurity(D) - \frac{N_{left}}{N} Impurity(D_{left}) - \frac{N_{right}}{N} Impurity(D_{right}), \tag{10}$$

where the dataset D with size N is obtained by splitting s partitions, and D_{left} and D_{right} of sizes N_{left} and N_{right}, respectively. MLlib supports DT for binary and multiclass classification using categorical features. The method splits data into rows and provides distributed training with millions of instances [29].

3.3.3. Random forest in Spark MLlib

Random forest is one of the most successful machine learning models. RF is an ensemble learning algorithm of decision trees for solving supervised learning tasks such as classification. It combines multiple decision trees to produce even more powerful models to get a more accurate and stable prediction by reducing the risk of overfitting. The algorithm builds a model consisting of multiple decision trees, based on different subsets of data using a random sample of data at the training stage. This randomness constitutes an advantageous feature of the RF method, enabling the model to be more robust than a single decision tree and overcoming the overfitting problem of the training data. Like DT, RF can also handle categorical features to perform a multiclass classification process and does not require feature scaling by capturing ability for nonlinearities and feature interactions [30].

The node splitting is the basic step of the RF algorithm and only a random subset of properties is considered by the algorithm for the splitting of a node. It is important that finding root node and division of nodes run randomly in the RF. The nodes are branched, and tree structures are formed according to the determined splitting rules such as maximum information gain, maximum information gain rate, and minimum Gini index by using the created training data. The information gain and the Gini index obtained by using attributes a to divide the sample set D is showed with given node splitting formula below [31].

$$Gain(D,a) = Ent(D) - \sum_{v=1}^{V} - \frac{|D^v|}{|D|} Ent(D^v), \tag{11}$$

$$Gini(D,a) = \sum_{v=1}^{V} - \frac{|D^v|}{|D|} Gini(D^v), \tag{12}$$

where D^v represents that the v branch node contains all the samples in D that have value a^v on the attribute a.

4. Experimental results

In this study, TerraSAR-X single polarized (VV channel) and spotlight mode images which cover an entire region of Visakhapatnam, India is utilized. Image dataset is obtained from public radar image database of Airbus Defense and Space. Some features of this dataset that are important for this study are given in Table 1.

We have created a dataset which contains five different categories of SAR patch images by combining Google Earth ground truth to determine which patch belongs to which category. The dataset consists of 1700 SAR image patches each with a size of 100 x 100, including different classes such as water, building, vegetation,

Table 1. Some features of the utilized dataset.

Features	Values
type of product	single polarized (VV channel)
imaging mode	spotlight
size of image (row x col)	19000x22000
image resolution (m)	1

road, and alongside. The road class consists of highways, intersections, railways, and interchanges while the vegetation class consists of grass, forest, and farm areas. The building area consists of house, work place, and warehouse. Specific examples of these identified categories are given in Figure 4.

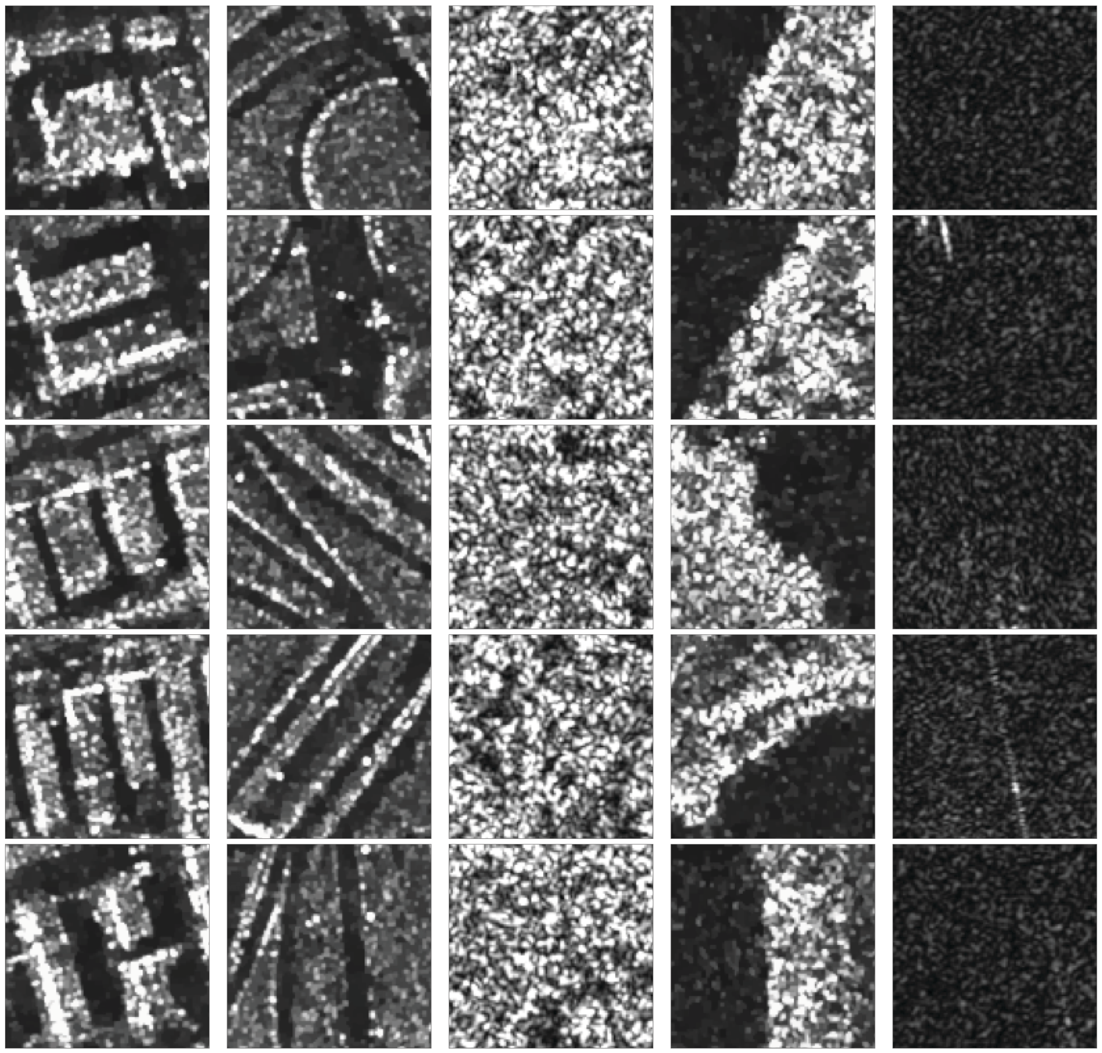

Figure 4. Image patches of each category.

For cross-validation, we divided our dataset into training and validation sets with 1530 and 170 samples corresponding to 90%-10% division, respectively. The distribution of training and validation patches for both groups was performed randomly. Numbers of patches for each class in the training and validation sets are listed in Table 2 for 10-fold cross validation.

Table 2. Training and validation sets.

Category	Training set	Validation set
Building	378	42
Road	360	40
Vegetation	288	33
Alongside	207	23
Water	297	32
TOTAL	1530	170

First, speckle noise filtering was implemented on SAR images by applying the SDD algorithm described in Section 3. Then, feature matrices of the despeckled image patches were constructed using the GLCM feature extraction. These matrices represented as vectors were input to the ML algorithm for training. The resulting classifier was tested with the testing set. Figure 5 shows the system flow diagram.

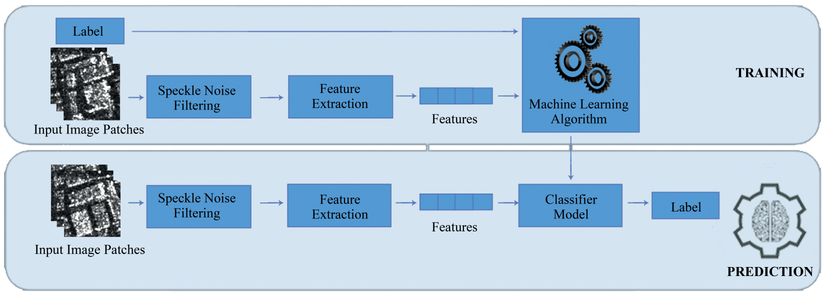

Figure 5. Flow diagram of image classification system using ML classifier.

The experiments were conducted on 5 cluster nodes with one master and 4 worker nodes. The internal memory was set at 6 GB for worker nodes. We simply placed a compiled version of Spark to generate Spark Standalone mode on each node of the cluster. We have achieved very satisfactory classification results using all possible configuration states. Classification accuracies of different ML algorithms is given in Figure 6. The training and test sets were obtained by dividing the data into 60%–40%, 70%–30% and 80%–20% subsets. With these partitions, accuracies of 90%, 91%, and 90% were obtained in the NB classification, respectively. With 90%-10% division of data, 10-fold cross validation was also used, and accuracy of 86% was obtained on average. When DT was used, the accuracies were 91%, 94%, and 94%, respectively. In addition, the accuracy obtained by 10-fold cross validation was 85% on average. When RF was used, the accuracies were 95%, 95%, and 96%, respectively. In addition, the accuracy obtained by 10-fold cross validation was 88% on average. The classification accuracy of RF was superior compared with the NB and DT methods.

The results were also compared with LeNET [32], which is a state-of-the-art deep learning method. All of the parameters used in network training are the parameters of the standard LeNET architecture. The CNN architecture has 2 convolutional and 1 fully connected layers. The convolution layer uses 20 filters with a size of 5×5. A rectified linear unit (ReLU) activation function is applied to the output feature maps and local response normalization (LRN) layer implemented behind the ReLU layer. The second layer is the pooling layer and this operation is performed by using a 3 × 3 dimension with stride is equal to 2. The third layer is also a convolution layer comprised of 50 filters using a 5 × 5 kernel with stride 1 and zero padding. ReLU activation function and LRN layers follow this convolution layer again. A pooling layer is a new layer added after the convolutional layer. The last layer consists of a fully connected layer. Sigmoid function, which is one of the most widely used activation functions, is used in the CNN model. The training and test sets were obtained by dividing the data into 60%–40%, 70%–30%, and 80%–20% subsets. With these partitions, accuracies of 92%, 93%, and 93% were obtained in this method, respectively. According to the results, although the deep learning method gives better results than the NB method, the success of DT and RF methods is still higher.

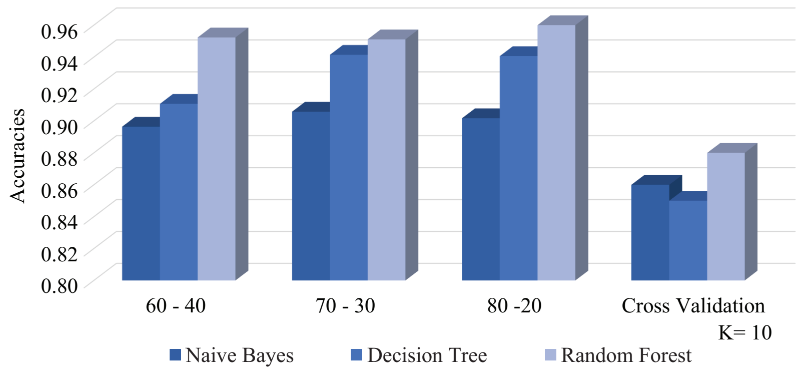

Figure 6. Classification accuracies of different ML algorithms.

The classifiers were then used with big size SAR images, and the execution time measurements obtained given in Table 3. As shown in the table, the results obtained with the CPU Knime analytics platform were also measured to show the superiority of the Spark design over the systems without clustering. The advantage of using cluster is evident for each method. As shown in Table 3, cluster system with 1 Master and 4 Worker implementation for NB method is 2.8 times faster compared with 1 Master implementation on the average due to its efficient memory-based distributed computing engine. In the same speed-up comparison, the average for the DT is 1.9 times, and 2.6 times for the RF. In addition, NB runs 1.3 times faster than the DT and 2.9 times faster than the RF. These measurement results show that as the data size increases, computational system performance improvement also increases.

7

Table 3. Execution times of test studies for different dataset sizes on different platforms.

Image size (GB)	# of Vectors	Naive Bayes (ms)			Decision tree(ms)		Random forest (ms)	
		CPU Knime	1 Master	1 Ms.-4 Wr.	1 Master	1 Ms.-4 Wr.	1 Master	1 Ms.-4 Wr.
0.014	$1x10^3$	825	25	20	10	11	27	28
0.14	$1x10^4$	2126	36	10	13	16	68	65
1.3	$1x10^5$	16,450	45	50	27	32	400	382
13	$1x10^6$	165,246	250	124	122	71	2388	1025
130	$1x10^7$	1,621,332	13,581	7255	12,353	6683	31,060	12,402
1300	$1x10^8$	14,616,206	145,965	52,233	126,731	68,370	398,863	155,094

As shown in Figure 7, the classification of the 1.3 TB-sized image was achieved with an acceleration of about 2.7 times by the 1 master and 4 worker system (the abbreviation M stands for Million). From the experiments, it is clear that machine learning classifiers (NB, DT, RF) perform well for classification at patch level on SAR images. We believe that more performance increase would especially occur when the dataset is more abundant and diverse.

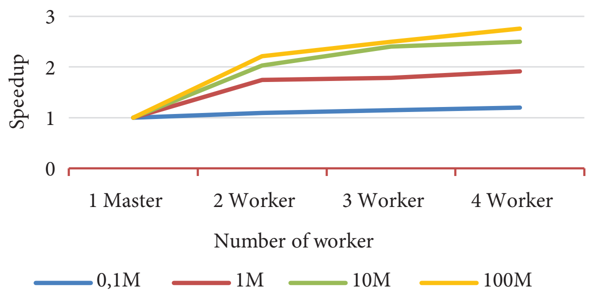

Figure 7. Spark testing phase speedups as a function of number of nodes.

5. Conclusions

Fast classification of SAR images has become more and more important due to increased data sizes. In this study, SAR image classification at patch level was achieved efficiently, especially by using the ML methods on Apache Spark. The testing accuracy of 91% was achieved with five classes. The obtained classification accuracy indicates that the despeckling process increases classification accuracy, and the GLCM method provides successful feature extraction. Experimental studies were carried out using naive Bayes, decision tree, and random forest to provide comparative results. The results were also compared with a state-of-the-art deep learning method. TerraSAR-X images of high-resolution real-world SAR images were used as data. Spark clustering design reduces execution times compared with traditional methods, and improvement increases as the data size grows. In future work, the feature extraction method can be improved by utilizing new features, and new classes can be defined by building larger and more diverse SAR datasets. If the size and the amount of data is increased, different deep learning algorithms can also be utilized.

Acknowledgment

This work was supported by the Scientific and Technological Research Council of Turkey (TÜBİTAK) under 2219-International Postdoctoral Research Scholarship Programme. The authors thank TÜBİTAK for the financial and scientific support.

References

[1] Moreira A, Prats-Iraola P, Younis M, Krieger G, Hajnsek I et al. A tutorial on synthetic aperture radar. IEEE Geoscience and Remote Sensing Magazine 2013; 1 (1): 6-43. doi: 10.1109/MGRS.2013.22483011

[2] Sakarya U, Demirpolat C. SAR image time-series analysis framework using morphological operators and global and local information-based linear discriminant analysis. Turkish Journal of Electrical Engineering & Computer Sciences 2018; 26: 2958-2966. doi:10.3906/elk-1712-339

[3] Chang YL, Chiang CY, Chen KS. SAR image simulation with application to target recognition. Progress in Electromagnetics Research 2011; 119: 35-57. doi: 10.2528/PIER11061507

[4] Horritt MS. A statistical active contour model for SAR image segmentation. Image and Vision Computing 1999; 17 (3-4): 213-224. doi: 10.1016/S0262-8856(98)00101-2

[5] Dekker RJ. Speckle filtering in satellite SAR change detection imagery. International Journal of Remote Sensing 1998; 19 (6): 1133-1146. doi: 10.1080/014311698215649

[6] Ampe EM, Vanhamel I, Salvadore E, Dams J, Bashir I et al. Impact of urban land-cover classification on groundwater recharge uncertainty. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing 2012; 5 (6): 1859-1867. doi: 0.1109/JSTARS.2012.2206573

[7] Attema EPW, Duchossois G, Kohlhammer G. Ers-1/2 SAR land applications: overview and main results. In: IEEE 1998 International Geoscience and Remote Sensing Symposium Proceedings; Seattle, WA, USA; 1998. pp. 1796-1798.

[8] Gamba P, Aldrighi M. SAR data classification of urban areas by means of segmentation techniques and ancillary optical data. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing 2012; 5 (4): 1140-1148. doi: 10.1109/JSTARS.2012.2195774

[9] Nafornita C, Isar A, Matanie N, Caba C. Solution of a big data problem. Simulator for denoising of single look complex SAR images. In: International Symposium on Signals Circuits and Systems; Iasi, Romania; 2017. pp. 1-4.

[10] Alonso K, Datcu M. Image information mining: an accelerated bayesian algorithm for data fusion of SAR big data. In: Proceedings of 10th European Conference on Synthetic Aperture Radar; Berlin, Germany; 2014. pp. 604-607.

[11] Tonye E, Fotsing J, Bernard EZ, Tankam NT, Kanaa T et al. Contribution of variogram and feature vector of texture for the classification of big size SAR images. In: Seventh International Conference on Signal Image Technology & Internet-Based Systems; Dijon, France; 2011. pp. 382-389.

[12] Dumitru CO, Schwarz G, Cui S, Espinoza-Molina D, Datcu M. Semi-automated semantic annotation of big archives of high resolution SAR images. In: Proceedings of EUSAR 11th European Conference on Synthetic Aperture Radar; Hamburg, Germany; 2016; pp. 687-690.

[13] Zhu H, Guo Y, Niu M, Yang G, Jiao L. Distributed SAR image change detection based on spark. In: IEEE 2015 International Geoscience and Remote Sensing Symposium; Milan, Italy; 2015. pp. 4149-4152.

[14] Zhu H, Guo Y, Niu M, Qiu L, Jiao L et al. SAR image change detection based on Spark-FLICM algorithm. In: IEEE 2016 International Geoscience and Remote Sensing Symposium; Beijing, China; 2016. pp. 3354-3357.

[15] Zhu H, Kou J, Qiu L, Guo Y, Niu M et al. Distributed SAR image change detection with opencl-enabled spark. In: Proceedings of the first Workshop on Emerging Technologies for software-defined and reconfigurable hardware-accelerated Cloud Datacenters; New York, NY, USA; 2017. pp. 1-6.

[16] Sharma T, Shokeen V, Mathur S. Multiple K Means++ clustering of satellite image using Hadoop MapReduce and Spark. International Journal of Advanced Studies in Computer Science and Engineering 2016; 5 (4): 23-31.

[17] Huang W, Meng L, Zhang D, Zhang W. In-memory parallel processing of massive remotely sensed data using an Apache Spark on Hadoop YARN model. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing 2017; 10 (1): 3-19. doi: 10.1109/JSTARS.2016.2547020

[18] Yu Y, Acton ST. Speckle reducing anisotropic diffusion. IEEE Transactions on Image Processing 2002; 11 (11): 1260-1270. doi: 10.1109/TIP.2002.804276

[19] Argenti F, Bianchi T, Lapini A, Alparone L. Fast MAP despeckling based on laplacian-gaussian modeling of wavelet coefficients. IEEE Geoscience and Remote Sensing Letters 2012; 9 (1): 13-17. doi: 10.1109/LGRS.2011.2158798

[20] Ozcan C, Sen B, Nar F. Sparsity-driven despeckling for SAR images. IEEE Geoscience Remote Sensing Letters 2016; 3 (1): 115-119. doi: 10.1109/LGRS.2015.2499445

[21] Haralick RM, Shanmugam K, Dinstein IH. Textural features for image classification. IEEE Transactions on Systems, Man, and Cybernetics 1973; SMC-3 (6): 610-621. doi: 10.1109/TSMC.1973.4309314

[22] Soh L, Tsatsoulis C. Texture analysis of SAR sea ice imagery using gray level co-occurances matrices. IEEE Transactions on Geoscience and Remote Sensing 1999; 37 (2): 780-795. doi: 10.1109/36.752194

[23] Lv W, Yu Q, Yu W. Water extraction in SAR images using GLCM and Support Vector Machine. In IEEE 10th International Conference on Signal Processing Proceedings; Beijing, China; 2010. pp. 740-743.

[24] Zhao J, Guo W, Cui S, Zhang Z, Yu W. Convolutional Neural Network for SAR image classification at patch level. In IEEE International Geoscience and Remote Sensing Symposium; Beijing, China; 2016; pp. 945-948.

[25] Ozcan C, Ersoy O, Ogul IU. Classification of SAR image patches with Apache Spark using GLCM texture features. In: International Conference on Advanced Technologies 3rd World Conference on Big Data; Izmir, Turkey; 2018. pp. 1-7.

[26] Karau H, Konwinski A, Wendell P, Zaharia M. Learning Spark: Lightning-Fast Big Data Analytics. 1st ed. Sebastopol, CA, USA: O'Reilly Media, Inc., 2015.

[27] Han J, Kamber M, Pei J. Data Mining: Concepts and Techniques. Waltham, MA, USA: Elsevier, Second Edition, 2006.

[28] Rojas I, Joya G, Catala A. Advances in Computational Intelligence. Cadiz, Spain: Springer, 2017.

[29] Karau H, Warren R. High Performance Spark: Best Practices for Scaling and Optimizing Apache Spark. Sebastopol, CA, USA: O'Reilly Media, Inc., First Edition, 2017.

[30] Karim R, Alla S. Scala and Spark for Big Data Analytics: Explore The Concepts of Functional Programming, Data Streaming, and Machine Learning. Birmingham, UK: Packt Publishing; First Edition, 2017.

[31] Man W, Ji Y, Zhang Z. Image classification based on improved random forest algorithm. In: IEEE 3rd International Conference on Cloud Computing and Big Data Analysis; Chengdu, China; 2018. pp. 346-350.

[32] LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. Proceedings of the IEEE 1998; 86 (11): 2278-2324. doi: 10.1109/5.726791