Mapping of large size hyperspectral imagery using fast machine learning algorithms

Junshi Xia, Naoto Yokoya, and Akira Iwasaki

Research Center for Advanced Science and Technology, The University of Tokyo, 4-6-1 Komaba, Meguro-ku, Tokyo 153-8904, Japan

Emails: xiajunshi@gmail.com; {yokoya, aiwasaki}@sal.rcast.u-tokyo.ac.jp.

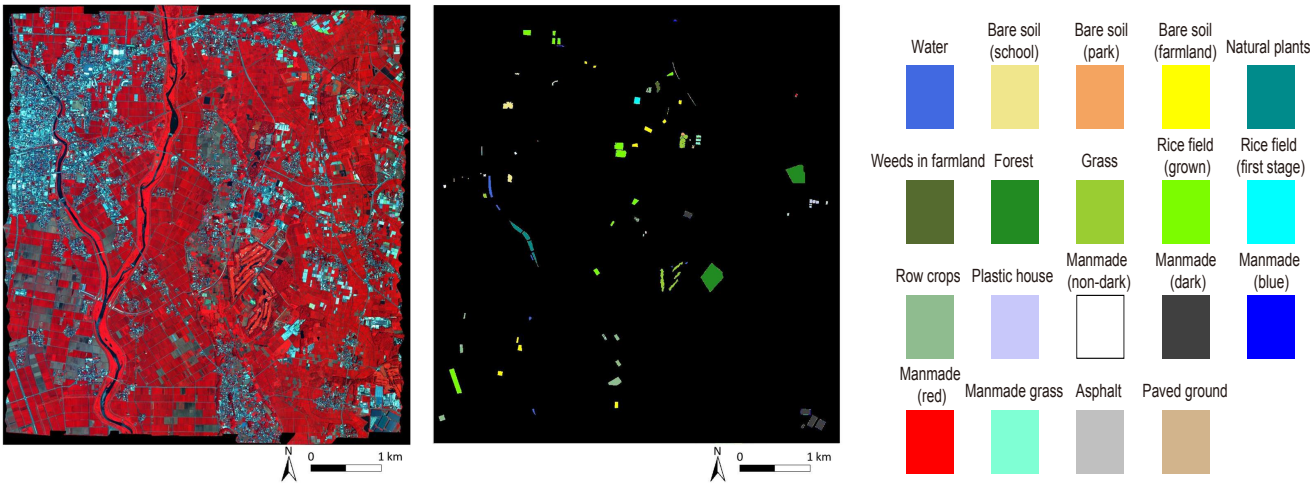
ABSTRACT

In this study, the authors propose to use two fast machine learning algorithms, i.e., Random Forest (RF) and Extreme Learning Machine (ELM), to classify large sized hyperspectral imagery. RF and ELM are based on different principles. The former one is the ensemble of decision trees (DT), which are trained with the bootstrap samples from the training set. The latter one is a feed forward neural network with a single layer of hidden nodes, where the weights connecting inputs to hidden nodes are randomly assigned and never updated. The two classifiers are applied to the hyperspectral image taken by Headwall's Hyperspec-VNIR-C imaging sensor over Chikusei, Ibaraki, Japan, on July 29, 2014, and compared with those from SVMs in terms of accuracy and computation time. Experimental results show that the global performances of the two proposed classifiers and SVMs are very similar, and the overall accuracies are over 90%. However, the computational cost of SVMs is much higher than those of RF and ELM. Besides, the performance of ELM can be further improved by increasing the number of hidden nodes.

Keywords: Land cover mapping, Hyperspectral image, Large size, Machine learning.

1 INTRODUCTION

Hyperspectral imaging (HSI) gains increasing interest in the remote sensing community since this technique allows us to obtain the images easily with detailed spectral and spatial information, which can provide a high discrimination capacity among different land cover classes (Chang [2003, 2007]). However, an increasing number of spectral bands results in decreasing the classification accuracy. This is well-known the Hughes phenomenon (also referred to as the curse of dimensionality) (Hughes [1968]). Support vector machines (SVMs) is a popular and successful algorithm for alleviating this phenomenon (Camps-Valls and Bruzzone [2005]; Melgani and Bruzzone [2004]; Vapnik [1995]). However, the large size of hyperspectral imagery often limits the possible use of SVMs, due to the difficulty of computation complexity and the selection of kernels and parameters. Thus, the alternative methods both considering computation complexity and performance, are required. In this presented work, the performance of two fast machine learning techniques, namely random forests (RF) (Breiman [2001]) and extreme learning machine (ELM) (Huang et al. [2004, 2006]), is investigated. More specifically, RF is a decision tree ensemble classifier, which is trained with the bootstrap samples from the training set and use a randomly selected subset of features to perform split in each node. ELM is a feed forward neural network with a single layer of hidden nodes by random selecting the weights in the connection between the input and hidden



(a) (b) (c)

Figure 1. (a) Three-band color composite of the Chikusei image. (b) Ground truth. (c) Legend of classes.

nodes. The results of RF and ELM are compared with those from SVMs in terms of accuracy and computation time.

2 Data set

The airborne hyperspectral dataset was taken by Headwall's Hyperspec-VNIR-C imaging sensor over Chikusei, Ibaraki, Japan, on July 29, 2014 between the times 9:56 to 10:53 UTC+9 (Yokoya and Iwasaki [2016]). The Hyperspec sensor recorded 512 bands in the spectral range from 363 to 1018 nm. Spectral binning was performed and the number of bands was reduced to 128 to increase the signal-to-noise ratio. The average height of the sensor above ground was approximately 900 m and a GSD was 2.5 m after geometric correction. The scene contains the large size of 2517 $\times$ 2335 pixels mainly including agricultural and urban areas in Chikusei. The entire dataset was composed of 13 flightlines, which were approximately parallel to the north-south direction and overlapped by approximately 35 % of each swath to reduce BRDF effects in the mosaic data. Atmospheric correction and BRDF correction were performed on each flightline image using the ATCOR-4 program, version 6.3. All the flightlines were mosaicked based on smoothly weighted averaging in overlapped areas so that edges of flightlines can be seamless. A Canon EOS 5D Mark II was also mounted on the same platform and high-resolution color images were sequentially acquired together with the hyperspectral data. The ground truth of 19 classes was collected via a field survey and visual inspection using the high-resolution color images. The 19 classes comprise water, three types of bare soil, seven types of vegetation, and eight types of man-made objects as shown in Table 2. Figure 1 (a) and (b) shows the three-band color composite image and the ground truth of the given hyperspectral data. The ground truthed dataset has been made available to the scientific community *.

*http://park.itc.u-tokyo.ac.jp/sal/hyperdata/.

Table 1. Class name and number of samples for training and test.

No.	Name	Train	Test
1	Water	20	2835
2	Bare soil (school)	20	2849
3	Bare soil (park)	20	276
4	Bare soil (farmland)	20	4842
5	Natural plants	20	4287
6	Weeds in farmland	20	1098
7	Forest	20	20506
8	Grass	20	6505
9	Rice field (grown)	20	13359
10	Rice field (first stage)	20	1258
11	Row crops	20	5951
12	Plastic house	20	2183
13	Manmade (non-dark)	20	1210
14	Manmade (dark)	20	7654
15	Manmade (blue)	20	421
16	Manmade (red)	20	212
17	Manmade grass	20	1030
18	Asphalt	20	791
19	Paved ground	20	135

3 RF and ELM

RF is an ensemble of T decision trees. During the testing stage, a class probability distribution $p_t(y_j|\mathbf{x}_j)$ for a given sample $\mathbf{x}_j$ is obtained by each decision tree and the final class label y_j^* is calculated by:

$$y_j^* = \arg\max_{y_j} \frac{1}{T} \sum_{t=1}^{T} p_t(y_j|\mathbf{x}_j) \tag{1}$$

During the training stage, all the trees are trained independently from each other with training set $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, where n is the number of training samples. For each decision tree, the quality of a given splitting function Φ is defined as

$$\mathbf{I}(\Phi) = \frac{|L|}{|L|+|R|} H(L) + \frac{|R|}{|L|+|R|} H(R) \tag{2}$$

where, L and R means the left and right node of a decision tree. $H(\cdot)$ measures the purity a set of training samples in terms of class labels and also calculated by using the Gini index (Breiman [2001]). $|\cdot|$ is the size of training samples.

The standard procedure of RF is to inject randomness at two points: random selection of training samples by tree and randomization of features by a node, and then to find a good splitting function, by evaluation Equation (2). After the splitting function is fixed, the tree is grown until some stopping criteria, such as a maximum tree depth, is reached (Breiman [2001]). Then, RF is generated by aggregating the predictions of the T trees.

ELM is generalized single hidden layer feedforward neural networks (SLFNs). The output function is expressed as:

$$f(\mathbf{x}_i) = \sum_{j=1}^{\delta} \beta_j h_j(\mathbf{x}_i) = \mathbf{h}(\mathbf{x}_i)\boldsymbol{\beta} \tag{3}$$

where, $\boldsymbol{\beta} = [\beta_1, \beta_2, ..., \beta_\delta]^\top$ is the vectors of weights between the hidden layer of δ nodes and the output node and $\mathbf{h}(\mathbf{x}_i) = [h_1(\mathbf{x}_i), h_2(\mathbf{x}_i), ..., h_\delta(\mathbf{x}_i)]$ is the vector of hidden layer of $\mathbf{x}_i$. Specifically, $\mathbf{h}(\cdot)$ is the feature mapping from the D-dimensional input space to the δ-dimensional hidden-layer feature space.

The standard SLFNs can approximate these n samples with zero error means that $\sum_i \|f(\mathbf{x}_i) - y_i\| = 0$. Thus, the n equations can be written compactly as:

$$\mathbf{H}\boldsymbol{\beta} = \mathbf{Y} \tag{4}$$

where, $\mathbf{Y}$ is the target matrix and $\mathbf{H}$ is the hidden-layer output matrix:

$$\mathbf{H} = \begin{bmatrix} \mathbf{h}(\mathbf{x}_1) \\ \vdots \\ \mathbf{h}(\mathbf{x}_n) \end{bmatrix} = \begin{bmatrix} h_1(\mathbf{x}_1) & \cdots & h_\delta(\mathbf{x}_1) \\ \vdots & \vdots & \vdots \\ h_1(\mathbf{x}_n) & \cdots & h_\delta(\mathbf{x}_n) \end{bmatrix} \tag{5}$$

The output weights in equation (4) are given by the following smallest norm least-squares solution (Huang et al. [2006]):

$$\boldsymbol{\beta} = \mathbf{H}^+\mathbf{Y} \tag{6}$$

where, $\mathbf{H}^+$ is the Moor-Penrose generalized inverse of the hidden layer output matrix $\mathbf{H}$.

In ELM, a feature mapping $\mathbf{H}$ from input space to a higher dimensional space is needed. Previous studies of (Huang and Chen [2007, 2008]) approved that almost all nonlinear piece-wise continuous functions can be used as output functions of the hidden-nodes. The Sigmoid function is usually adopted as the nonlinear piece-wise continuous function:

$$g(\omega, b, \mathbf{x}_i) = \frac{1}{1 + \exp\left(-(\omega \cdot \mathbf{x}_i + b)\right)} \tag{7}$$

where, $\{\omega_j, b_j\}_{i=1}^{\delta}$ are randomly generated values that can define a continuous probability distribution (i.e., $\int g = 1$). Thus, $\mathbf{h}(\mathbf{x}_i)$ is defined based on the nonlinear piece-wise continuous function $g(\omega_i, b_i)$:

$$\mathbf{h}(\mathbf{x}_i) = [g(\omega_1, b_1, \mathbf{x}_i), ..., g(\omega_\delta, b_\delta, \mathbf{x}_i)] \tag{8}$$

The training and prediction steps of ELM are listed in Algorithm 1.

4 Experimental results

In this work, a small set of labeled samples (20 samples per class) is randomly selected from the reference data as the training set. The rest of the pixels form the testing set. For RF, two parameters, i.e., the number of trees and number of selected features split in each node, are empirically fixed. The values are set to 50 and 12, respectively. For ELM, the number of nodes in a hidden layer is set to 128. A free library for SVM, i.e., LibSVM with MATLAB implementation, is chosen for training the SVM classifier, which is used to compare RF

Algorithm 1 Extreme learning machine

Training phase

Input: $\{\mathbf{X}, \mathbf{Y}\} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$: training samples, δ: the number of nodes in a hidden layer. g: the Sigmoid function .

Output: The output weight $\boldsymbol{\beta}$.

1: Randomly select the $\{\omega_1, ..., \omega_\delta\}$ and $\{b_1, ..., b_\delta\}$

2: For each training sample $\mathbf{x}_i$, calculate the output layer matrix: $\mathbf{h}(\mathbf{x}_i) = [g(\omega_1, b_1, \mathbf{x}_i), ..., g(\omega_\delta, b_\delta, \mathbf{x}_i)]$

3: Calculate the output weight: $\boldsymbol{\beta} = \mathbf{H}^{+}\mathbf{Y}$

Prediction phase

Input: A new sample $\mathbf{x}^*$. The output weight $\boldsymbol{\beta}$. The sigmoid function g. $\{\omega_1, ..., \omega_\delta\}$ and $\{b_1, ..., b_\delta\}$

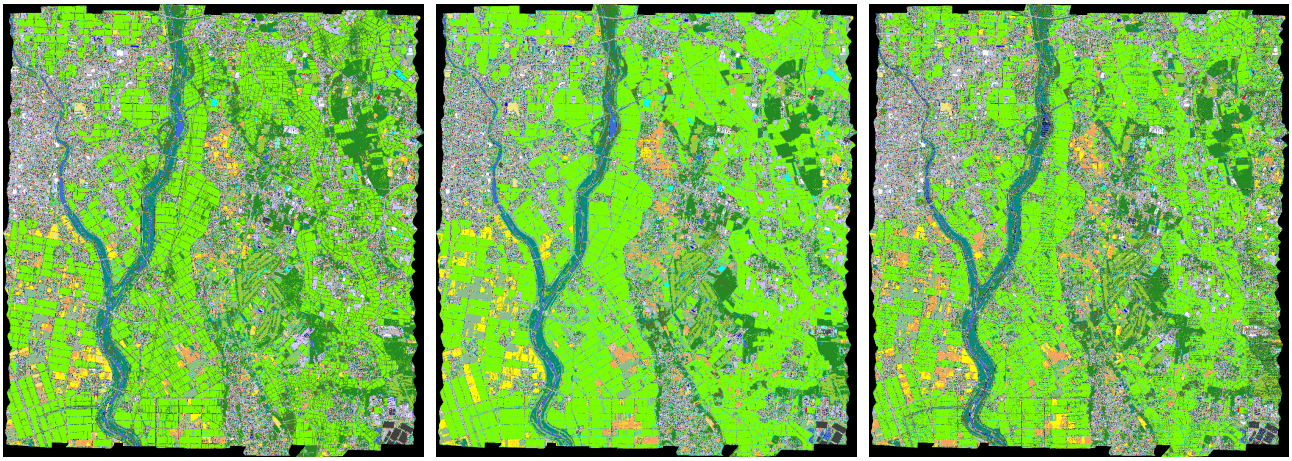
Output: Class label of $\mathbf{x}^*$.

1: Calculate the output layer matrix: $\mathbf{h}(\mathbf{x}^*) = [g(\omega_1, b_1, \mathbf{x}^*), ..., g(\omega_\delta, b_\delta, \mathbf{x}^*)]$.

2: $y^* = \mathbf{h}(\mathbf{x}^*)\boldsymbol{\beta}$. Assign the number of column which gets the greatest value among the columns to the class label of $\mathbf{x}^*$.

Table 2. Overall, average and class-specific accuracies obtained for RF, ELM and SVM classifiers

Classifier	RF	ELM	SVM
OA	90.07	91.02	**91.11**
AA	92.46	91.96	**92.67**
κ	88.60	89.68	**89.80**
Class 1	**97.70**	96.57	93.04
Class 2	93.98	**96.16**	91.66
Class 3	87.97	**100.00**	**100.00**
Class 4	**78.70**	52.48	51.22
Class 5	**97.62**	**97.62**	98.22
Class 6	**96.42**	76.84	94.78
Class 7	86.68	**97.44**	95.04
Class 8	93.84	**98.61**	95.92
Class 9	94.17	**98.55**	98.34
Class 10	97.76	**100.00**	99.04
Class 11	81.35	**89.97**	66.78
Class 12	95.63	**97.65**	89.07
Class 13	88.92	94.08	**94.35**
Class 14	88.79	68.26	**98.21**
Class 15	99.27	**100.00**	**100.00**
Class 16	94.55	**100.00**	99.51
Class 17	94.80	98.73	**99.71**
Class 18	97.44	85.79	**99.75**
Class 19	91.20	**98.40**	96.12
Time (s)	41.78	42.14	118.32



(a) (b) (c)

Figure 2. Classification maps of (a) RF, (b) ELM, and (c) SVMs.

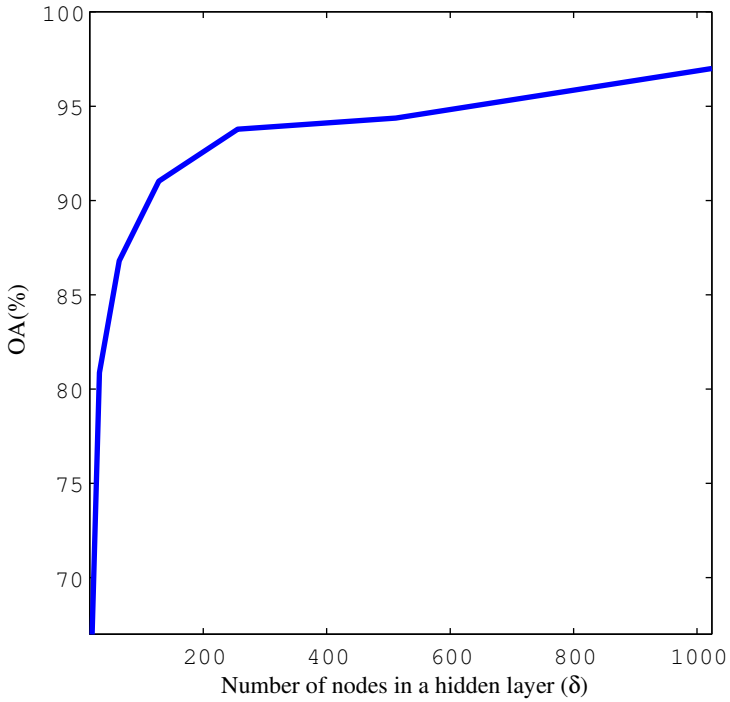


Figure 3. Sensitivity to the change of the number of nodes in a hidden layer (δ).

and ELM (Chang and Lin [2011]). A Gaussian RBF kernel is used, and the required parameters are selected by a grid search using a three-fold cross validation.

Table 2 summarizes the overall, average and class-specific accuracies obtained for RF, ELM, and SVM classifiers. From the table, we can see that the overall accuracies obtained for three classifiers are quite similar. SVMs shows a slightly better performance. However, RF and ELM gain the best results for four and twelve classes, respectively. The computational cost is compared in Table 2. Experimental results are all carried out on a computer with Intel Xeon 2 CPUs, 3.2 GHz, and 16 GB of memory. It can be found that RF and ELM are much more efficient than SVMs. For illustrative purposes, Figure 2 shows the obtained classification maps.

In Xia et al. [2016], we found that RF is robust to the number of features split in each node, and larger ensemble size has non-significant influence while increasing the computation time. Effect of the number of nodes in a hidden layer (δ) in ELM is depicted in Figure 3. When δ becomes larger, ELM tends to have better performances. However, it leads to higher computation time.

5 Conclusion

In this work, we propose to use two fast learning methods, RF and ELM, for the classification of large size hyperspectral data. Experimental results confirmed the effectiveness of ELM and RF when compared with SVMs. Thus, both RF and ELM methods can be considered attractive for the classification of large size hyperspectral data. In the future, we would like to investigate the potential use of parallel computing towards greater efficiency on large size datasets.

Acknowledgment

The authors would like to that KAKENHI Grant Number 24360347 and JSPS KAKENHI Grant Number 16F16053 for supporting this work.

References

Breiman, L. (2001). Random forest. *Machine Learn.*, 45(1):5–32.

Camps-Valls, G. and Bruzzone, L. (2005). Kernel-based methods for hyperspectral image classification. *IEEE Trans. Geosci. Remote Sens.*, 43(6):1351–1362.

Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Trans. Intell. Systems Tech.*, 2:27:1–27:27. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Chang, C. I. (2003). *Hyperspectral Imaging: Techniques for Spectral Detection and Classification.* Plenum Publishing Co.

Chang, C. I. (2007). *Hyperspectral Data Exploitation: Theory and Applications.* Wiley-Interscience, Hoboken, NJ.

Huang, G. B. and Chen, L. (2007). Convex incremental extreme learning machine. *Neurocomput.*, 70(16-18):3056–3062.

Huang, G. B. and Chen, L. (2008). Enhanced random search based incremental extreme learning machine. *Neurocomput.*, 71(16-18):3460–3468.

Huang, G. B., Zhu, Q. Y., and Siew, C. K. (2004). Extreme learning machine: a new learning scheme of feedforward neural networks. In *2004. Proceedings. 2004 IEEE International Joint Conference on Neural Networks*, volume 2, pages 985–990, Budapest, Hungary.

Huang, G. B., Zhu, Q. Y., and Siew, C. K. (2006). Extreme learning machine: theory and applications. *Neurocomputing*, 70(1-3):489–501.

Hughes, G. (1968). On the mean accuracy of statistical pattern recognizers. *IEEE Trans. Infor. Theory*, 14(1):55–63.

Melgani, F. and Bruzzone, L. (2004). Classification of hyperspectral remote sensing images with support vector machines. *IEEE Trans. Geosci. Remote Sens.*, 42(8):1778–1790.

Vapnik, V. N. (1995). *The Nature of Statistical Learning Theory*. Springer.

Xia, J., Yokoya, N., and Iwasaki, A. (2016). Hyperspectral image classification with canonical correlation forests. *IEEE Trans. Geosci. Remote Sens.*, in press.

Yokoya, N. and Iwasaki, A. (2016). Airborne hyperspectral data over chikusei. Technical Report SAL-2016-05-27, Space Application Laboratory, the University of Tokyo, Japan.