Article

Cloud Detection Using an Ensemble of Pixel-Based Machine Learning Models Incorporating Unsupervised Classification

Xiaohe Yu ^{1,*} and David J. Lary ² ¹ Geospatial Information Sciences, The University of Texas at Dallas, Richardson, TX 75080, USA² Hanson Center for Space Science, The University of Texas at Dallas, Richardson, TX 75080, USA; David.Lary@utdallas.edu

* Correspondence: xhyu66@gmail.com or xxy160430@utdallas.edu

Abstract: Remote sensing imagery, such as that provided by the United States Geological Survey (USGS) Landsat satellites, has been widely used to study environmental protection, hazard analysis, and urban planning for decades. Clouds are a constant challenge for such imagery and, if not handled correctly, can cause a variety of issues for a wide range of remote sensing analyses. Typically, cloud mask algorithms use the entire image; in this study we present an ensemble of different pixel-based approaches to cloud pixel modeling. Based on four training subsets with a selection of different input features, 12 machine learning models were created. We evaluated these models using the cropped LC8-Biome cloud validation dataset. As a comparison, Fmask was also applied to the cropped scene Biome dataset. One goal of this research is to explore a machine learning modeling approach that uses as small a training data sample as possible but still provides an accurate model. Overall, the model trained on the sample subset (1.3% of the total training samples) that includes unsupervised Self-Organizing Map classification results as an input feature has the best performance. The approach achieves 98.57% overall accuracy, 1.18% cloud omission error, and 0.93% cloud commission error on the 88 cropped test images. By comparison to Fmask 4.0, this model improves the accuracy by 10.12% and reduces the cloud omission error by 6.39%. Furthermore, using an additional eight independent validation images that were not sampled in model training, the model trained on the second largest subset with an additional five features has the highest overall accuracy at 86.35%, with 12.48% cloud omission error and 7.96% cloud commission error. This model's overall correctness increased by 3.26%, and the cloud omission error decreased by 1.28% compared to Fmask 4.0. The machine learning cloud classification models discussed in this paper could achieve very good performance utilizing only a small portion of the total training pixels available. We showed that a pixel-based cloud classification model, and that as each scene obviously has unique spectral characteristics, and having a small portion of example pixels from each of the sub-regions in a scene can improve the model accuracy significantly.

Keywords: landsat 8; machine learning; cloud detection; ensemble approaches; self organizing maps (SOM); NDSI; NDVI; whiteness; HOT

Citation: Yu, X.; Lary, D.J. Cloud Detection Using an Ensemble of Pixel-Based Machine Learning Models Incorporating Unsupervised Classification. *Remote Sens.* **2021**, *13*, 3289. <https://doi.org/10.3390/rs13163289>

Academic Editor: Filomena Romano

Received: 16 July 2021

Accepted: 17 August 2021

Published: 20 August 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Remote sensing imagery, such as that provided by the United States Geological Survey (USGS) Landsat satellites, has been widely used to study environmental protection, hazard analysis, and urban planning for decades. The usefulness and applications of remote sensing imagery continue to expand as more image-based models and algorithms have emerged so that we can derive more knowledge and information from satellite images. Due to the ubiquity of clouds, cloud pixels are a persistent presence in such imagery, especially in tropical areas. A study estimates that about 67% of the earth's surface is typically covered by cloud based on satellite data from July 2002 to April 2015 [1]. The presence of cloud has a serious impact on the use of remote sensing images. Cloud areas

appear as extremely bright pixels in the images. These pixels can cause issues in various remote sensing imagery analyses, including incorrect land surface classification, inaccurate atmosphere correction, low quality Aerosol Optical Depth (AOD) retrieval and false land surface change detection [2]. As a result, clouds are considered noise in most situations and are typically removed prior to further analysis, which makes cloud detection a crucial step for remote sensing image preprocessing.

Over the last few decades, a variety of methods have been developed for cloud detection. Let us briefly consider a few examples.

1.1. An Overview of Cloud Detection Approaches

Automated Cloud Cover Assessment (ACCA) [3], a scene specific cloud detection method was developed for Landsat 7. It employs two pass through ETM+ to establish the the reflective and thermal features of the cloud and non-cloud area in a scene, and then to identify the cloud in the rest area of the whole scene. This approach experiences difficulty in identifying cloud areas with snow and brightly illuminated desert areas.

Zhu et al. [4] proposed Function of Mask (Fmask) which is an objected-based cloud and cloud shadow approach for Landsat images, which has a 96.41% reported accuracy. The author of Fmask further improved Fmask in terms of increasing the performance for Landsat 4–7, making it suitable for Landsat 8 and Sentinel 2 imagery [5].

Foga et al. [6] compared the performance of ACCA, LEDAPS CCA, and CFmask (C version of Fmask) on three cloud validation dataset including IRISH, SPARCS, and Biome. Among these three algorithms, CFmask is reported with the best overall accuracy.

Hughes et al. [7] proposed a machine learning approach for automated cloud and cloud shadow detection by using neural network and spatial post-processing techniques, which achieves lower cloud shadow omission error and cloud commission error compared to Fmask.

A multi-feature combined (MFC) approach is proposed for the Chinese GaoFen1 cloud detection by using spectral features in combination with geometric and texture features, which has a 96.8% accuracy [8].

All these aforementioned approaches are single temporal approaches, which only require one scene for implementation. In contrast to the single temporal approach, a multi-temporal approach (Tmask) is proposed for cloud, shadow, and snow by using multiple images at the same location [9]. It generates a time series model which is used to predict the TOA reflectance surface. These surfaces are then compared with Landsat images to differentiate clouds, shadows, and snow. However, this approach requires at least 15 clear observations in each pixel to generate a robust time series model, which makes it less applicable in places that have long been covered by snow or cloud.

Candra et al. [10] proposed an automated cloud and cloud shadow detection method by using multi temporal Landsat8 images, which is named MCM. This approach makes use of the reflectance differences between two images at the same location to identify cloud and cloud shadows, which is especially effective in tropical areas.

The use of deep learning techniques including CNN, RNN and GCN have recently garnered much attention for remote sensing image classification tasks because they are capable of extracting high-level features from images. Zhu et al. [11] reviewed the major advances of deep learning in remote sensing. Xie et al. [12] proposed a multilevel cloud detection method based on simple linear iterative clustering (SLIC) and a deep convolutions neural network (CNN). This method achieves a better result compared with a scene learning-based approach proposed by Zhenyu [13] and progressive refinement scheme approach proposed by Zhang et al. [14]. Authors of [15] proposed a cloud detection method (MSCN) based on Fully Convolutional Networks (FCN) [16] by fusing multi-scale convolutional features for cloud detection, which is effective in snow and areas covered by non-cloud bright objects.

Another CNN-based cloud detection method is trained on high resolution WV-2 satellite images, which not rely on SWIR or IR bands and can be applied to Sentinel

imagery [17]. Zi et al. [18] proposed a novel cloud detection method for Landsat 8 images by using Simple Linear Iterative Cluster (SLIC), PCA Network (PCANet), Support Vector Machine (SVM), and Conditional Random Field (CRF). This approach combines statistical models, classical machine learning methods, and a deep learning network to generate a robust model for cloud detection and achieves an accurate result. Yang et al. [19] proposed a CNN-Based cloud detection method by using thumbnails of remote sensing images instead of the original remote sensing images. To handle the coarse resolution of thumbnail images, a cloud detection neural network feature pyramid module, and boundary refinement block techniques are employed to generate accurate cloud prediction results. This work has been further extended by Guo [20] for cloud and snow coexistence scenarios by proposing a new model DSnetV2.

In addition to cloud detection, deep learning-based techniques have been extensively applied to remote sensing image classification problems, especially for hyper-spectral images. Graph convolutional networks (GCNs) are a new emerging network architecture that can handle and model long-range spatial relations. Shahraki and Prasad [21] proposed a cascade framework of 1-D CNNs and GCNs for a hyper-spectral classification problem. Qin et al. [22] extended the GCNs by considering spatial and spectral neighbors. Pu et al. [23] proposed a localized graph convolutional filtering-based GCNs method for hyper-spectral image classification. Traditional GCNs are computationally expensive because the spatial matrices are constructed. Hong et al. [24] showed that miniGCNs can be trained in minibatch fashion for classification problems. The miniGCNs are more robust, and are capable of handling out-of-samples with lower computation cost compared to traditional GCNs.

Based on the data required, these cloud detection methods can be divided into single temporal and multi-temporal approaches. The single temporal approach seeks to identify cloud pixels based on imagery at a single time, while a multi-temporal approach makes use of imagery from multiple comparable timeframes for a same area to identify cloud pixels by comparing the pixel differences between cloud free images and cloudy ones [25]. Depending on the algorithm used, cloud detection can be categorized into a classical algorithm-based approach and machine learning approach [26]. The classical algorithm refers to methods which have specific steps to be followed for input imagery and to generate the output mask like the FMask [4]. On the other hand, machine learning approaches take the advantage of existing data and learn from a training set without human interference to generate an output [18].

Cloud detection is still challenging in these aspects. First, cloud pixels are hard to identify from “bright” areas such as snow by traditional rule-based approaches. The multi-temporal approach requires more than one image at the same location, which is less applicable for low temporal resolution satellites, such as Landsat. While deep learning-based approaches generally enable better cloud detection accuracy, they require a high performance GPU which may not be available. Some other approaches require additional information in combination with the spectral features to improve the performance, which requires extra labor efforts.

1.2. A Pixel-Based Approach Using an Ensemble of Learners

This paper has four distinct goals. (1) Propose a cloud detection modeling approach for cloud detection by only using the 10 wavelength bands available at a *single pixel* as an information source without the need of any other ancillary data. (2) Engineer important predictor features that could increase the model accuracy. (3) Investigate the influence of the training sample size on the model accuracy, thus finding the smallest possible training sample size that could balance the training time and machine learning model accuracy. (4) Explore the importance rank of predictors and the optimal hyper-parameter settings for cloud mask prediction.

In contrast to the whole image-based approaches just described, and to mitigate the challenges just enumerated, in this study we have taken a pixel-based approach that only

requires a single image. We have used an ensemble machine learning approach that has been tested with Landsat 8 imagery.

Compared with the multi-temporal approaches [25], this single-temporal approach is more feasible because of the mono-temporal feature of Landsat data in nature. Unlike other classical approaches that need auxiliary data [4], this approach only requires the information on the 10 wavelength bands for a single pixel of Landsat 8 imagery. In comparison to machine learning approaches under a deep learning frame work, the computational facility is not as demanding as deep learning, so a GPU is not needed in this research. This proposed machine learning model uses an ensemble approach which simultaneously employs multiple decision tree learners for cloud detection. The input parameters are tuned in two ways. On the one hand we optionally include unsupervised classification results from the application of a self-organizing map (SOM) as one of the input features for the ensemble model training. On the other hand, 5 indices that are calculated from the 10 wavelength bands signal are included as input features for model training.

Four distinct training subsets were generated from the Biome cloud validation dataset for model training [6]. Models are generated based on different sets of input parameters and different training samples. Then, their performance is compared against Fmask 4.0.

2. Materials and Methods

2.1. Data Sources

This study uses the L8 cloud cover assessment validation dataset as the source for model training and validation [6,27]. The L8 Biome dataset includes 96 Landsat 8 images sampled across the world, and with a manually generated cloud mask for each image. The name Biome is given because the target scenes are sampled based on biome types. The path/row of the scenes covering the land are sampled across the world on the basis of biome types which include urban, forest, shrubland, grass/cropland, snow/ice, wetlands, and water. In order to create a heterogeneous dataset, path/rows with obviously clustering patterns are discarded and will be re-sampled from their biome types. Then scenes within the final set of path/rows will be labeled as clear, mid-cloudy, and cloudy manually based on the cloud coverage percentage within each scene. A total of 96 scenes are prepared covering 8 biome types in 3 cloud coverage levels as shown in Figure 1.

Figure 1. Global distribution of the 96 unique Landsat 8 Cloud Cover Assessment scenes, adopted from [6].

2.2. Data Pre-Processing: The Intra-Group and the Ultra-Group

Validation masks and all spectrum band layers from the Biome dataset are cropped to 4000×4000 pixels for model training and validation (Figure 2). These 96 cropped Biome images are then divided into two groups, the intra-group and the ultra-group. The intra-group includes 88 images from which we sub-sample to provide our training pixels. The name “intra-group” rather than “training-group” was given, because only a tiny portion (up to 1.3%) of the intra-group are used as the model training source to show the effectiveness of our pixel based approach. Therefore, more than 98% of the data in the intra-group are “new” to the models, which also makes the intra-group qualified for model performance evaluation. On the other hand, the ultra-group contain 8 images that our models have never seen, which means the ultra-group are 100% “new” to the models. Images in both groups will be used for model performance evaluation.

Figure 2. Landsat8 scenes are centre-cropped into 4000×4000 pixel images.

For each biome type, eleven cropped images are selected as the intra-group, and we hold the one remaining image as the ultra-group (See Table 1). Then the 10 wavelength bands of each scene are stacked as a TIFF file for model training and validation. However, band 8 has a 15 m resolution which is higher than other bands. We downgrade the resolution for band 8 to make the imagery resolution the same as the other bands.

Table 1. Number of Scenes included in the Intra-Group and the Ultra-Group.

Intra-Group	Ultra-Group	Land Types
11	1	Barren
11	1	Forest
11	1	Grass/Crops
11	1	Shrubland
11	1	Snow/Ice
11	1	Water
11	1	Urban
11	1	Wetlands
88	8	Total

2.3. Ensemble Machine Learning Classification Approach

Ensemble learning is a machine learning paradigm where multiple weak learners are combined to improve the training performance. Various types of learner aggregation can be employed, including Bootstrap Aggregation (Bagging), Boosting and random space. In this paper, all three of these tree ensemble approaches are used, and the particular approach that will be employed in the tree-based ensemble model will be determined at the Bayesian Optimization stage.

Bagging aggregates trained weak learners by creating many bootstrap replicas and training each weak learner on one replica. Typically, the number of bootstrap replicas

can vary from just a few up to several hundred. Each replica is generated by drawing N samples out of N observations with replacement (see Figure 3a). Drawing samples with replacement omits an average of 37% of the total observations which decreases the correlation between each weak learner to avoid over-fitting. Each weak learner is trained on a single replica by randomly extracting features for each split node which is called the random forest technique. Then the ensemble model makes a prediction on new data by taking the most voted predictions from individual learners for use in the classification.

Figure 3. Schematics illustrating the various ensemble model approaches. (a) Bagging, (b) Boosting.

Boosting trains learners sequentially (see Figure 3b). It maintains a weight distribution for all training observations and each observation is assigned a weight indicating its importance. By decreasing the weight for correctly classified observations and increasing the weight for misclassified observations at each round, trained learners are able to focus more on hard observations that have been misclassified by previous learners. Distinct from bagging which generates replicas for individual learner training, boosting trains all models with the same dataset but with different weight. Instead of taking the most votes of the predictions from individual learners, boosting combines the predictions from individual learners with weights. As a result, boosting is able to make individual learners focus more on hard observation points round by round, thus to obtain a premium result from weak learners. In this paper, AdaBoost, RUSboost, and LSBoost are candidate boosting methods employed for our ensemble models.

2.4. Representative Sampling

There are a total of $88\,4000 \times 4000$ pixel images in the intra-group, a total of 1.408 billion pixels. A small portion of the samples will be drawn from the intra-group and used for model training. Generally, the more training data we use, the better machine learning model performance we achieve. However, for the task of cloud detection, the cloud label is very labor intensive to generate and is not widely available. One goal of this research is to explore a machine learning modeling approach that uses as small a fraction of the intra-group dataset as possible but can still fit the data well. In order to compare the performance of models trained on different sizes of training subsets, a representative random sampling approach is proposed to sample four subsets from the 88 cropped images. Each of the 88 images has 10 wavelength bands. After stacking the 10 wavelength bands together, each stacked image becomes a $4000 \times 4000 \times 10$ matrix. Then a validation label is attached to the stacked matrix thus forms a $4000 \times 4000 \times 11$ matrix for each image. These 3-dimensional images are converted to 2-dimension image tables, which are then concatenated along the

horizontal direction and form an image pool table (Figure 4a). The next step is to draw representative samples from this image pool table (Figure 4b). One column of the image pool table is sorted and divided into n bins. These bins have an uniform width covering the range of pixel values of the column, which could potentially reveal the distribution of values in the column. Then m samples are randomly drawn from each bin of the column and this sampling process is repeated for all columns in the image pool table. Duplicated rows could be drawn when repeating the sampling process across each column, which could lead to an unbalanced sampling subset. Any duplicated samples are removed after the sampling process complete, and a sample subset that covers the value range of each column is obtained without any duplication. The bin number n and the sample number m are adjusted to obtain different training samples from the whole dataset. In this research, 4 training samples subsets are generated for model training (see Table 2).

Figure 4. Flow chart illustrating the image stacking and representative sampling procedures. In the stacking and merging procedure (a), the 10 wavelength bands as well as the label mask for each of the 88 images are stacked as an image table. Then, the 88 image tables are merged into one image pool table. The B1–B10 and Label in the headers represent the spectral band number and the pixels class labels. DN in tables represent the digital value. In the representative sampling procedure (b), the image pool table is sorted on a certain column (in green), then n bins are selected from the sorted column and m pixels are drawn from each bin. A final training sample is generated after repeating this process over all columns.

Table 2. Representative Training Samples. *n* represents the number of bins, *m* represents the number of pixels sampled in each bin, and the Total represents the total pixels sampled for each subset.

Subset Name	Bins (<i>n</i>)	Observations in Bin (<i>m</i>)	Total	Percentage
Sample 1	256	100	185,841	0.01%
Sample 2	512	200	714,972	0.05%
Sample 3	1024	300	2,063,945	0.15%
Sample 4	5120	600	18,722,275	1.30%

2.5. Feature Selection

2.5.1. Indices for Cloud Differentiating

A set of 5 indices is derived from the 10 wavelength bands and used as input features. These include *NDSI*, *NDVI*, *Whiteness*, *HOT*, and *B5/B6 ratio* (See Equations (1)–(4)). These indices are sensitive to cloud pixels and often have a threshold that are able to differentiate cloud pixels from others [4]. Instead of testing these indices with a threshold value, they are integrated as part of input parameters into the proposed machine learning model.

$$\begin{aligned} NDSI &= (Band\ 3 - Band\ 6) / (Band\ 3 + Band\ 6) \\ NDVI &= (Band\ 5 - Band\ 4) / (Band\ 5 + Band\ 4) \end{aligned} \quad (1)$$

NDSI and *NDVI* represent the normalized difference snow index and the normalized difference vegetation index, respectively. Both indices usually have a value near zero for clouds. Sometimes they could be larger for thin cloud over highly vegetated areas but should be less than 0.8.

$$\begin{aligned} MeanVis &= (Band\ 2 + Band\ 3 + Band\ 4) / 3 \\ Whiteness &= \sum_{i=2}^4 \left| \frac{(Band\ i - MeanVis)}{MeanVis} \right| \end{aligned} \quad (2)$$

The Whiteness index was first proposed by Gomez-Chova et al. [28] for Environmental SATellite (ENVISAT) and was modified as in Equation (2) later [4] for Landsat satellite images. The modified whiteness is effective in excluding non-cloud pixels due to their “non-flat” reflectance feature compared to the “flat” reflectance of cloud pixels.

$$HOT = Band\ 2 - 0.5 \times Band\ 4 - 0.08 \quad (3)$$

The Haze Optimized Transformation (*HOT*) index was first proposed by Zhang et al. [29] and was modified by Zhu et al. [4], which is useful to separate haze and thin cloud from clear-sky pixels.

$$B5/B6\ Ratio = \frac{B5}{B6} \quad (4)$$

The *B5/B6* ratio is able to separate “bright” rocks from cloud pixels as “bright” rocks usually have a higher *Band 5* value than *Band 4* while cloud has a higher *Band 4* value than *Band 5*.

2.5.2. Self Organizing Map

A self-organizing map (SOM) is an unsupervised clustering method that uses a neural network to represent training data in a low-dimensional space [30]. The 10-dimensional input vectors are grouped into 100 clusters with 100 neurons. Each neuron has a weight vector which will be updated based on competitive learning approach at iterations. At each iteration, all neurons will be compared with a random selected training vector, and the one that is the closest in distance to the training point will be the winner and rewarded for moving closer to the training point by updating the weights. A neighborhood function is used to update the weights of the neighbors of a winning neuron in order to preserve the topological properties of the input space. Each cluster links to those image pixels sharing similar features. Although the SOM clustering results may not be able to directly

differentiate cloud pixels from others pixels, it can provide useful information for the pixel type that can be utilized as an input feature for the training of the ensemble models. The SOM learning process can be represented by the neuron weight update process that is defined by (5).

$$\begin{aligned}\Delta W_{j,i} &= \eta(t) \cdot T_{j,I(x)}(t) \cdot (x_i - w_{i,j}) \\ \eta(t) &= \eta_0 \exp\left(-\frac{t}{\tau_\eta}\right) \\ T_{j,I(x)}(t) &= \exp\left(-\frac{S_{j,I(x)}^2}{2\sigma(t)^2}\right) \\ S_{j,i} &= \|w_j - w_i\| \\ \sigma(t) &= \sigma_0 \exp\left(-\frac{t}{\tau_0}\right)\end{aligned}\quad (5)$$

where $\Delta W_{j,i}$ denotes the updated weight value, t is the epoch, i and j are neuron indices, and $I(x)$ is the winning neuron. $\eta(t)$ refers to the learning rate which controls the neuron learning speed. $T_{j,I(x)}(t)$ denotes the neighborhood function which defines the extent of neighbor neurons of a winning neuron. $S_{j,i}$ is the distance between neurons, and the $S_{j,i}$ is the neighborhood size.

2.6. Model Training and Hyper-Parameter Optimization

2.6.1. Model Training

A total of twelve tree-based ensemble models were trained on the four subsets sampled from the 88 cropped Landsat 8 images. Depending on the selection of input features, these machine learning models could be divided into three groups. Each model group consists of four models with the same specifications but trained on four different sizes of training samples.

The features used to train the first model group were the 10 wavelength bands from the 88 cropped images. For the second model group, in addition to the 10 wavelength bands, the 5 derived indices discussed in Equations (1)–(4) were also included as input features. The training process of the third model group consists of two steps. The first step is to build an SOM model on the training sample and use the SOM model to classify all the pixels from the training sample into clusters. The second step is to then include the SOM cluster label as an extra input feature in addition to the 10 wavelength bands. Then an ensemble learning model will be trained by employing the 10 wavelength bands and the SOM labels as the input features. The target variables for each subset are the pixel class which consists of cloud, clear, thin cloud, and cloud shadow. Models are summarized in Table 3 and the sample sizes are summarized in Table 2.

Table 3. Model names and features.

Model Name	Input Features	Training Subset
88Mdl_Sample 1	10 bands	Sample 1
88Mdl_Sample 2	10 bands	Sample 2
88Mdl_Sample 3	10 bands	Sample 3
88Mdl_Sample 4	10 bands	Sample 4
88Mdl_SOM_Sample 1	10 bands; SOM Class	Sample 1
88Mdl_SOM_Sample 2	10 bands; SOM Class	Sample 2
88Mdl_SOM_Sample 3	10 bands; SOM Class	Sample 3
88Mdl_SOM_Sample 4	10 bands; SOM Class	Sample 4
88Mdl_5Index_Sample 1	10 bands; 5 Derived Indices	Sample 1
88Mdl_5Index_Sample 2	10 bands; 5 Derived Indices	Sample 2
88Mdl_5Index_Sample 3	10 bands; 5 Derived Indices	Sample 3
88Mdl_5Index_Sample 4	10 bands; 5 Derived Indices	Sample 4

2.6.2. Hyper-Parameters Optimization

Hyper-parameters are the parameters that control the model training process, and they are parameters that can be optimized. Different combinations of hyper-parameters can lead to different model performance. Hyper-parameter optimization refers to the process used to find a set of hyper-parameters that yields an optimal model which minimizes the loss function on the given validation dataset. Grid search, random search, and Bayesian optimization are three of the most common approaches to perform hyper-parameter optimization. Grid search exhaustively searches all the combinations from a manually defined hyper-parameter space of a model and is the most expensive approach especially when feature space is large. Random search is a slight modification of grid search where hyper-parameter combinations are randomly selected from a manually defined hyper-parameter space instead of exhaustively enumerating of all combinations. This search method is faster than grid search, but both of them are limited to prior knowledge of hyper-parameter distribution specifications. Bayesian optimization, on the other hand, attempts to find the global optimum in minimum steps without the need to manually define each hyper-parameter sample points. Bayesian optimization builds a probabilistic model on the hyper-parameter values and the objective function, the surrogate model. A Gaussian process (GP)-based surrogate model is a popular one for Bayesian optimization because it is cheap to evaluate. The function to propose hyper-parameters combinations is referred to as the acquisition function, which is an important part of a surrogate model. The Bayesian optimization algorithm can be defined by Equation (6).

$$\begin{aligned} X_t &= \operatorname{argmax}_X u(X|D_{1:t-1}) \\ D_{1:t-1} &= (X_1, y_1), \dots, (X_{t-1}, y_{t-1}) \end{aligned} \quad (6)$$

where u is the acquisition function, X_t is the hyper-parameter sampling point at iteration t , $D_{1:t-1}$ is the objective function values and hyper-parameters samples from previous $t - 1$ iterations. Bayesian optimization tries to find the hyper-parameter combinations at step t that is able to maximize the acquisition function u . Then the evaluation results from the objective function at step t which will be added to previous results to update the GP. Hyper-parameters for the tree-based ensemble models are optimized by Bayesian optimization. Tuned parameters includes the number of learners, the ensemble approach, the learning rate, the minimum leaf number, split criteria, the number of variables used in each node, and evaluation times.

2.7. Cloud Mask Prediction and Accuracy Evaluation

There are 4 training subsets sampled from the 88 images in the intra-group. Based on Table 2, each training subset accounts for only a very small portion of the total pixels present in the 88 images. The portion is not directly set but determined by the bin number n and the sample number m in the representative sampling stage. The portions range from 0.01% to 1.30% as shown in Table 2. Once the training processes are complete, these models are applied to the 88 images in the intra-group and the 8 images in the ultra-group to generate the predicted cloud masks. Fmask 4.0 is also applied to the 88 images in the intra-group and the 8 images in the ultra-group. The masks generated using the Fmask 4.0 algorithm include five classes which are clear land, clear water, cloud shadow, snow, and cloud. These classes are grouped into cloud, clear, and shadow to match the output of the machine learning models.

Each of the LC8 Biome images has an associated manually generated mask created by an analyst for validation purposes [6]. Previous cloud masking studies reveal that the differences due to the analyst's interpretation is about 7% [31]. To avoid this difference, all these cloud masks are digitized by a single analyst. Then, image-based confusion matrices are generated for each model and Fmask. For each image, there will be 13 confusion matrices generated, 12 of which are for the 12 model predictions, and the thirteenth

for Fmask. These confusion matrices furnish the foundation of the accuracy measure calculations discussed in the next section.

3. Results

3.1. Accuracy Measurements Establish

Confusion matrices of the entire intra-group and ultra-group are generated for each of the machine learning models and the Fmask algorithm. These confusion matrices are then used to calculate the performance metrics including the correctness, the cloud commission error, and the cloud omission error at the image level and the group level. These measurements are defined in Equations (7)–(9):

$$\text{Correctness} = \frac{\text{Cloud as Cloud} + \text{Clear as Clear} + \text{Shadow as Shadow}}{\text{Total pixel}} \quad (7)$$

$$\text{Cloud Commission Error} = \frac{\text{Clear as Cloud} + \text{Shadow as Cloud}}{\text{Total Clear} + \text{Total Shadow}} \quad (8)$$

$$\text{Cloud Omission Error} = \frac{\text{Total Cloud} - \text{Cloud as Cloud}}{\text{Total Cloud}} \quad (9)$$

where correctness measures the overall model correctness for all three classes, cloud commission error measures the portion of pixels that are estimated as cloud but are actually not, cloud omission error measures the portion of cloud pixels that failed to be detected. Because those non-cloud pixels that are labeled as cloud will be removed in most remote sensing image analysis, cloud commission error could cause information loss. Compared to information lost, a failure of detecting cloud pixels will have a more severe negative impact on the image analysis. Therefore, controlling cloud omission error is the first priority of this research. Among all of the 88 intra-group images, only 32 images have a shadow class labeled. Limited by the training and validation size, shadow detection is not the goal of this research. Therefore, the commission error and omission error were built for cloud pixel identification.

3.2. Visual Comparison and Accuracy Assessment

For each of the 88 images we produced 14 predicted cloud masks, 12 of which are from our 12 machine learning models, one from Fmask, and one from the ground truth mask. Figure 5 illustrates four sample images with their associated ground truth image, ground truth mask, model prediction mask from model 88Mdl_5index_sample4, and Fmask. For these four sample images, both from the machine learning models and Fmask, we see that they perform well on cloud detection. However, higher commission and omission errors can be observed compared to the machine learning model prediction mask. For the forest image in the first row in Figure 5, both the prediction mask and Fmask show some degree of omission error, but Fmask has more missed cloud pixels. For the shrub image in the second row, Fmask mis-classified a large water zone as cloud. For the grass/crops image in the third row, Fmask missed some cloud pixels again in the thick cloud mixture zone. For the image in the fourth row, both masks provide a good fit based on a visual comparison.

To better explore the performance of machine learning model and Fmask, the accuracy assessment Table 4 is summarized for the four images in Figure 5. Both machine learning models and Fmask have good overall correctness (above 85%); however, Fmask has significant omission error in the first and third image, and has high commission error in the second image. This quantitative result agrees with the visual comparison result. The machine learning model has good consistent performance on the four sample images.

To provide a more comprehensive accuracy assessment, accuracy assessment tables are summarized for both the entire intra-group (Table 5) and the ultra-group (Table 6) for each of the machine learning models and for the Fmask algorithm. Tables 5 and 6 are sorted on cloud omission error in ascending order. From Table 5, the 88Mdl_SOM_sample4 performs the best on all measurements on the 88 intra-group. As can be seen in Table 6, the

88Mdl_5index_sample3 has the lowest cloud omission error and the model 88Mdl_sample3 has the best correctness performance in the ultra-group. Fmask 4.0 has the second lowest cloud omission error.

Figure 5. Sample image comparison between the validation mask, 88Mdl_5index_sample4 model predicted mask and Fmask. Each row displays an image with a certain type. The first column represents the ground truth image and the second column represents the ground truth mask, which is manually generated by expert. The third column represents the model prediction results and the fourth column displays the Fmask application result.

Table 4. Model predicted mask and Fmask accuracy comparison at image level for Figure 5. OE refers to omission error and CE refers to commission error.

Source Scene ID	Mask Type	Correct	Cloud OE	Cloud CE	Land Type
LC81750622013304LGN00	Predicted	90.30%	24.50%	0.12%	Forest
	Fmask	86.52%	33.94%	0.62%	
LC80010732013109LGN00	Predicted	97.47%	3.78%	0.62%	Shrub
	Fmask	85.39%	0.26%	8.73%	
LC80290372013257LGN00	Predicted	95.82%	3.86%	0.77%	Grass/Crops
	Fmask	86.65%	27.78%	0.66%	
LC80640452014041LGN00	Predicted	99.57%	5.18%	0.11%	Urban
	Fmask	98.22%	0.55%	0.80%	

Table 5. Intra-group accuracy assessment for machine learning models and the Fmask.

Model Names	Correct	Cloud OE	Cloud CE
88Mdl_SOM_sample4	98.57%	1.18%	0.93%
88Mdl_sample4	98.35%	1.19%	1.08%
88Mdl_5index_sample4	98.45%	1.33%	0.98%
88Mdl_5index_sample3	97.30%	1.37%	2.12%
88Mdl_sample3	98.00%	1.68%	1.38%
88Mdl_sample2	97.59%	2.15%	1.61%
88Mdl_5index_sample2	97.62%	2.16%	1.57%
88Mdl_5index_sample1	97.13%	2.50%	1.43%
88Mdl_sample1	96.76%	2.60%	1.65%
88Mdl_SOM_sample3	96.10%	3.76%	2.26%
88Mdl_SOM_sample2	95.06%	5.03%	2.87%
88Mdl_SOM_sample1	94.46%	5.22%	3.31%
Fmask 4.0	88.45%	7.57%	10.82%

Table 6. Ultra-group accuracy assessment for machine learning models and the Fmask.

Model Names	Correct	Cloud OE	Cloud CE
88Mdl_5index_sample3	86.35%	12.48%	7.96%
Fmask 4.0	83.09%	13.76%	10.69%
88Mdl_sample3	87.00%	14.78%	5.80%
88Mdl_sample2	86.81%	15.23%	5.23%
88Mdl_sample4	85.53%	15.60%	7.00%
88Mdl_SOM_sample1	86.15%	15.61%	5.84%
88Mdl_SOM_sample3	86.31%	15.67%	6.19%
88Mdl_5index_sample2	86.85%	15.72%	5.04%
88Mdl_SOM_sample4	86.00%	15.95%	6.38%
88Mdl_SOM_sample2	86.25%	15.99%	5.74%
88Mdl_5index_sample4	86.05%	16.68%	5.30%
88Mdl_sample1	84.77%	17.28%	5.06%
88Mdl_5index_sample1	85.61%	17.60%	3.99%

As can be seen in Table 6, Fmask has the lowest performance for all metrics when compared to the machine learning models. To have a in-depth exploration on the factors influencing the Fmask performance, the eight images with the lowest performance are summarized (Table 7). Four of the eight images are plotted in Figure 6. The accuracy measurements are summarized at image level for the the Fmask and the five indices machine learning model trained on sample4 for comparison. The machine learning model performed well on all images, while the Fmask has high variability across the images. The overall correctness could decrease to as low as 7.47% for the Ice/Snow images. Four out of the eight images are for the ice/snow land type. These images have the common features

that the background reflectance is “bright”. This feature causes trouble for Fmask in two ways. In some images, Fmask mis-classifies ice/snow as cloud, while in some images Fmask mis-classifies cloud as clear pixels. This quantitative finding conforms with what one can see by visual inspection of the results in Figure 6, where we see that Fmask has difficulty in detecting cloud in images in the first and third row. In the first row, Fmask mis-classifies the cloud pixels as snow/ice-covered clear pixel. In the third row, Fmask mis-classifies snow/ice-covered clear pixels as cloud. As can be seen in the fourth row of Figure 6, water is another case with “bright” background, where Fmask mis-classifies large water regions as cloud pixels.

The other three images types listed in Table 7 are grass/crops, forest, and shrubland. The grass/crops, forest and shrub images are similar in that they absorb a large fraction of the visible bands and make the images “dark”. This effect could influence the performance of Fmask. As can be seen in the Table 7, Fmask has high cloud omission error over these three land types. The machine learning model, on the other hand, has a stable performance across the different land types. The cloud commission errors are NaN in the table because this image is 100% covered by thin cloud; thus, by definition, no cloud commission error can be calculated. For the shrubland image, Fmask mis-classifies large thin cloud zones as clear pixels (second row of Figure 6).

Table 7. Model predicted mask and Fmask accuracy comparison for low Fmask accuracy images.

Source Scene ID	Mask Type	Correct	Cloud OE	Cloud CE	Land Type
LC80060102014147LGN00	Predicted Fmask	98.35% 69.67%	0.00% 57.33%	0.00% 6.54%	Ice/Snow
LC80211222013361LGN00	Predicted Fmask	97.65% 55.15%	2.70% 73.21%	1.90% 5.99%	Ice/Snow
LC80290372013257LGN00	Predicted Fmask	95.82% 86.65%	3.86% 27.78%	0.77% 0.66%	Grass/Crops
LC81720192013331LGN00	Predicted Fmask	100.00% 64.92%	0.00% 35.08%	nan nan	Forest
LC82271192014287LGN00	Predicted Fmask	99.90% 7.47%	0.92% 26.71%	0.04% 97.04%	Ice/Snow
LC81490122013218LGN00	Predicted Fmask	99.27% 18.49%	0.35% 87.72%	5.45% 0.74%	Shrubland
LC80200462014005LGN00	Predicted Fmask	96.88% 90.03%	8.71% 15.42%	0.09% 0.81%	Ice/Snow
LC80210072014236LGN00	Predicted Fmask	98.21% 79.96%	8.35% 15.79%	0.83% 16.80%	Water

Figure 6. Fmask miss-classified images. Each row displays an image with a certain type. The first column represents the ground truth image and the second column represents the ground truth mask. The third column represents the model prediction results and the fourth column displays the Fmask application result.

3.3. Feature Importance and Topology Analysis

Machine learning has been traditionally treated as a “black box”, because most of the machine learning algorithms focus on solving problems based on training data but do not readily give interpretations on how the input variables influence the prediction outcome. The unsupervised SOM classification method and the supervised tree-based ensemble classification method provides ways to explore these influences of the input variables on

the prediction outcomes. As in Figure 7, a 10-by-10 network is constructed, which includes 100 neurons. There is a weight vector associate with each neuron and the weight vector has the same dimension as input vectors. As the training process goes, these weight vectors move to the center of input vector clusters. Once the training process completes, the SOM model is applied to the training dataset to explore the variable characteristics. Each neuron represents a cluster, and each input vector will be assigned to one of the 100 neurons. A useful feature of SOMs is topology preservation, which means those close vectors in the input space remain close together after being projected to 2D planes [32]. Therefore, a high dimensional vectors can be visualized in a 2D plane.

Figure 7. The four SOM plots. A total of 100 neurons are plotted on a 10 by 10 gridded plane. Axis x and axis y are the neuron location indices. (a) The connection map displays the neuron topology, (b) the hits map represents the number of vectors associated with each neuron, (c) the SOM neighbor weight distance map displays the distance between each neuron, and (d) displays the weight of each input component (spectral bands) in a 2D plane.

Figure 7a illustrates the SOM neuron structure in a 2D gridded plane. As can be seen in Figure 7b, the SOM hits map displays how many input vectors fall within each SOM class. There are several large clusters of vectors annotated with red circles which have hit numbers of greater than 300,000. The Figure 7c indicates the feature space distance between each node. Black and dark red represent longer inter-SOM class separation distances while yellow and orange represent shorter inter-SOM class separation distances. Several red and black connections marked with blue ovals segment the whole input vectors into different zones. Each zone shares some degree of similarity with the adjacent classes. The black regions indicate the longest inter-class separations. Figure 7d displays the weights of each input spectral bands on the neuron plane. Similar patterns in weight planes indicate high correlation between the different bands. As can be seen, band 1, band 2, band 3, band 4, band 5 and band 8 indicate a strong correlation pattern in the 2D topology. Band 6 and band 7 share some degree of similarity. On the other hand, band 9 and band 10 give unique information to the SOM model. These plots give information of how the input variables influence the unsupervised classification and the similarity between classes. The SOM model is able to integrate these many input variables and provide new insights into the data, which can be used as an additional input feature for the tree-based models to enhance their performance.

One advantage of tree-based models is that they are easy to interpret. The decision tree algorithms internally select the variable which can most reduce the entropy at each decision tree splitting node. The variable that can most reduce the entropy will be put in the root node, and the second most important variable will be placed next after the root node. Thus, variables are sorted based on their capability to reduce the entropy. As a result, the importance and contribution of each variable to the model can be easily traced and explained. Split nodes are determined by their capability to reduce the entropy of the leaf node. Therefore, the importance rank of input variables in differentiating clusters can be traced and visualized. Figure 8 displays the importance rank plot for the three best performing ensemble models, which are the SOM model trained with sample 4, the five indices model trained on sample 3, and the base model with only 10 wavelength bands trained on sample 3. Among these tree models, band 7, band 1, band 5 and HOT have the most influence on the predicted model class. For the model incorporating the SOM classification, the SOM class is the fifth most important input feature.

Figure 8. The importance plot for three best models, which are (a) the model trained on sample 4 with SOM as input, (b) the model with 5 indices trained on sample 3 and (c) the base model with 10 wavelength bands trained on sample 3. Model (a) has the best performance on all metrics in the intra-group, while model (b,c) has the best performance on omission error and correctness, respectively. Red and yellow bars represent the top 5 important features and blue bars represent the importance of the rest variables.

3.4. Algorithm Complexity and Running Time Analysis

The proposed ensemble method is based on a decision tree model; thus, the algorithm complexity of decision tree is the conceptual base for the ensemble algorithm complexity

analysis. The depth complexity of a fully developed tree would be in $O(\log(n))$ for a balanced tree, and $O(n)$ for a extremely unbalanced tree, where n is the number of training samples. In practice, decision trees are rarely fully developed or extremely unbalanced. As a result, $O(\log(n))$ is a good approximation for the tree depth complexity. A quality function is calculated for every node for each feature at all levels. The algorithm complexity for a decision tree would be $O(m \times n \times \log(n))$, where m is the number of features. For bagging ensemble methods, random forest techniques are applied, where only the square root number of features are considered on each node for each tree. Thus, the training complexity is $O(k \times m^{\frac{1}{2}} \times n \times \log(n))$ where k is the number of trees in the ensemble method. For Boosting ensemble methods, all the features are considered for model training, thus the complexity is $O(k \times m \times n \times \log(n))$. Selection of the Boosting or Bagging approach are two of the ensemble hyperparameters which are determined by the hyperparameter optimization. Once the the model is trained and ready for running, the run time complexity for ensemble types are $O(k \times m)$.

The running time is summarized in Table 8. Training time is the total elapsed time for all the steps to generate a model including representative training subsets sampling, model training, hyper-parameter optimization, plotting and model saving under Intel(R) Xeon(R) CPU E7-4850 v3 @ 2.20 GHz. The Running time is the average elapsed time to make a cloud mask prediction and to plot the prediction masks under Intel(R) Xeon(R) CPU X5650 @ 2.67 GHz.

Table 8. The model training and running time.

Model Name	Training Time (s)	Running Time (s)
88Mdl_Sample1	971	836
88Mdl_Sample2	3037	133
88Mdl_Sample3	42,279	346
88Mdl_Sample4	694,051	1037
88Mdl_SOM_sample1	1587	268
88Mdl_SOM_sample2	5099	268
88Mdl_SOM_sample3	33,839	303
88Mdl_SOM_sample4	524,561	371
88Mdl_5Index_sample1	2162	326
88Mdl_5Index_sample2	4348	251
88Mdl_5Index_sample3	20,334	547
88Mdl_5Index_sample4	355,966	167

3.5. Hyperparameter Sensitivity Analysis

Hyperparameters are important parameters that control the behavior of empirical machine learning models. The hyperparameter optimization seeks to suppress the variance and bias of a model by finding a set of global optimal hyperparameters that minimize the objective function value. We proposed the Bayesian hyperparameters optimization which minimizes the cross-validation classification error with 30 iterations for each model. Optimized hyperparameters as well as the five-fold cross-validation objective function values measuring the model errors are summarized for each model. Hyperparameters at the initial iteration and at the best iteration are listed in the Table 9 to demonstrate how the optimization process improves the objective function value. Each model has two iterations result listed. The first iteration with the value one is the initial round of hyperparameter optimization, while the second number refers to the iteration where the best optimization results are achieved among the total 30 iterations. Ensemble column refers to the ensemble method including Bagging and Boosting two categories. Trees represents the number of single decision trees included in each model. LearnRate is a hyperparameter controlling the learning speed for the Boosting method only. MinLeafSize is the minimum number of samples required to be at a leaf node. The MaxNumSplits is the maximal number of decision splits for each tree. The MinLeafSize and MaxNumSplits together control the

depth of tree models. As in Table 9, the objective function values decreased significantly after hyperparameter optimization for all the 12 ensemble models. Compared to the initial model, the optimized models are dominated by employing the Boosting method, with larger numbers for the MinLeafSize and more MaxNumSplits.

Table 9. Hyperparameters sensitivity analysis.

Model Name	Iteration	Objective	Ensemble	Trees	LearnRate	MinLeafSize	MaxNumSplits
88Mdl_Sample1	1	0.99	RUSBoost	91	0.07	8435	1741
	25	0.03	Bag	31	NA	1	124,550
88Mdl_Sample2	1	0.99	RUSBoost	25	0.00	153,080	19
	28	0.02	AdaBoostM2	25	0.40	39	3153
88Mdl_Sample3	1	0.99	RUSBoost	13	0.02	51,723	61,518
	11	0.02	RUSBoost	12	0.02	7	647,140
88Mdl_Sample4	1	0.13	RUSBoost	17	0.05	5433	37,578
	19	0.02	RUSBoost	145	0.02	1	1.77×10^7
88Mdl_SOM_sample1	1	0.23	RUSBoost	12	0.71	13	2
	27	0.03	Bag	15	NA	10	177,740
88Mdl_SOM_sample2	1	0.16	RUSBoost	13	0.02	4	9
	24	0.03	AdaBoostM2	27	0.00	11	7937
88Mdl_SOM_sample3	1	0.06	RUSBoost	19	0.01	4	446,920
	21	0.02	RUSBoost	218	0.05	1	28,364
88Mdl_SOM_sample4	1	0.09	RUSBoost	23	0.01	369	3.07×10^6
	23	0.02	RUSBoost	88	0.13	1	1.32×10^7
88Mdl_5Index_sample1	1	0.22	Bag	14	NA	62,299	65,065
	26	0.03	Bag	29	NA	22	2138
88Mdl_5Index_sample2	1	0.99	RUSBoost	28	0.04	542,860	95
	26	0.02	AdaBoostM2	12	0.80	16	418,850
88Mdl_5Index_sample3	1	0.99	RUSBoost	28	0.04	542,860	95
	30	0.02	RUSBoost	412	0.11	2	255,040
88Mdl_5Index_sample4	1	0.99	RUSBoost	18	0.27	1.78×10^6	2.89×10^6
	16	0.01	RUSBoost	276	0.60	7	420,910

4. Discussion

This paper established and evaluated 12 ensemble machine learning models for pixel-based cloud classification trained and tested using the labeled Landsat 8 Biome dataset. In addition to the information from the ten Landsat bands some additional input features were used, including an unsupervised self-organizing map classification and five indices derived from various band combinations. This feature engineering improved the performance of the machine learning cloud pixel classification.

Typically, training a machine learning model by using as much data as possible enables a better model performance. However, when it comes to supervised cloud classification, we are often constrained by the availability of labeled images and sometimes by the available computing power. We therefore designed a strategy to use only a tiny subset of the total number of pixels available for model training.

The objective of this research is three-fold. First, we described, implemented and validated an ensemble approach to build supervised classification model for cloud detection, including model selection, selection of representative training data, and feature engineering. Second, we explored the importance of the input features. Last but not least, we investigated how the size of the training subsets influenced the model performance.

Generally, a larger number of training samples enables better model performance when using the intra-group, while the training size has less influence on the model performance of the ultra-group. In addition, our ensemble models had consistent performance

on all land types in contrast to Fmask which had trouble in differentiating cloud pixels in some “bright” and “dark” area such as snow/ice, shrub, and some forest area. The fact that the overall model performances on the intra-group are better than the performance on the ultra-group is reasonable because all the training data are sampled from the intra-group, even though just a small fraction of the total available pixels were used. As shown in the Tables 2 and 5, models trained on subsets that only accounted for 1.3% of the total pixels available could achieve an accuracy as high as 98.57% when applied to the whole 88 intra-group images. For the independent validation dataset that our machine learning models had never previously seen, even though the performance decreased, they still out-performed Fmask.

Landsat 8 images could suffer from various degradation, noise, or variability during the image processing. The generation capability of the machine learning models have been largely determined by the representatives of the training samples. In this study, the LC8Biome cloud validation data are designed to cover the geographical scope of the world and the eight biome types, and includes three levels of cloud coverage (clear, mid, cloudy). This approach makes the training set representative enough to allow a model with good generalization for the normal radiance variability of LC8 data. However, the LC8 Biome dataset is free from severe noise, which is caused by such as equipment failure, which makes the cloud detection on images with severe noise out of the scope of the proposed machine learning models. Nevertheless, the only well-known issue so far for LC8 is the thermal band failure. Although the lack of cloud validation masks for severely distorted image stops us from doing an accuracy assessment; the incorporation of 10 wavelength bands as predictors in the machine learning models allows some degree of resistance to certain band failure. In addition, the proposed machine learning method is also suitable to establish a cloud detection model for images with high noise once the training samples with severe noise are available.

5. Conclusions

Overall, the tree-based ensemble machine learning models with Bayesian optimization achieved better performance than Fmask for all metrics over the 88 images in the intra-group. On the eight images in the ultra-group, 88Mdl_5index_sample3 has the lowest cloud omission error and the 88Mdl_sample3 has the best correctness. The base model with 10 wavelength bands as the input variable could achieve good performance for both the intra-group and ultra-group. Fmask had issues in differentiating cloud pixels mainly in ice/snow images, and some in the water, forest, and shrub images.

The empirical pixel-based machine learning models discussed in this paper could achieve very good and consistent performance when using only a tiny portion of the total available training data. This result indicates that a pixel-based method can perform well, and that each scene obviously has unique spectral characteristics, and having a small portion of example pixels from each of the sub-regions in a scene can improve the model accuracy significantly. Integrating the self-organizing map results and five band derived indices as a part of input variables could further help to reduce the cloud commission error and cloud omission error compared to the base models. Based on hyperparameter sensitivity analysis, the Boosting ensemble method with a small MinLeafSize and large MaxNumSplits are effective settings for high accuracy model training. Among all the input variables investigated in this research, band 1, band 5, band 7, and HOT are the five most important variables for the best three models; Band 10, band 6, and SOM are in the second tier; band 4, band 6, band 9, NDVI, whiteness, B5/B6 ratio are in the third tier which also contributes to the model performance.

Author Contributions: Conceptualization, X.Y. and D.J.L.; methodology, X.Y. and D.J.L.; software, X.Y. and D.J.L.; validation, X.Y.; formal analysis, X.Y.; investigation, X.Y.; resources, X.Y. and D.J.L.; data curation, X.Y.; writing—original draft preparation, X.Y.; writing—review and editing, D.J.L.; visualization, X.Y.; supervision, D.J.L.; project administration, X.Y. and D.J.L.; funding acquisition, D.J.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The LC8-Biome cloud validation dataset is available at URL: <https://landsat.usgs.gov/landsat-8-cloud-cover-assessment-validation-data> (accessed on 18 August 2021). The code for image processing, model training and model application is available at URL: <https://github.com/xiaoheyu> (accessed on 18 August 2021).

Acknowledgments: The concept for this paper was first developed as an example of the use of machine learning for the ESA ϕ -lab machine learning bootcamp for remote sensing in January 2019. URL: <https://philab.phl.esa.int/> (accessed on 18 August 2021). Christopher Simmons is gratefully acknowledged for his computational support. The authors acknowledge the Texas Research and Education Cyberinfrastructure Services (TRECIS) Center, NSF Award #2019135, and the University of Texas at Dallas for providing HPC, visualization, database, or grid resources and support that have contributed to the research results reported within this paper. URL: <https://trecis.cyberinfrastructure.org/> (accessed on 18 August 2021). The authors acknowledge the Texas Advanced Computing Center (TACC) at the University of Texas at Austin for providing HPC, visualization, database, or grid resources that have contributed to the research results reported within this paper. URL: <http://www.tacc.utexas.edu> (accessed on 18 August 2021).

Conflicts of Interest: The authors declare no conflict of interest.

References

- King, M.D.; Platnick, S.; Menzel, W.P.; Ackerman, S.A.; Hubanks, P.A. Spatial and Temporal Distribution of Clouds Observed by MODIS Onboard the Terra and Aqua Satellites. *IEEE Trans. Geosci. Remote Sens.* **2013**, *51*, 3826–3852. [CrossRef]
- Arvidson, T.; Goward, S.; Gasch, J.; Williams, D. Landsat-7 long-term acquisition plan. *Photogramm. Eng. Remote Sens.* **2006**, *72*, 1137–1146. [CrossRef]
- Irish, R.R. Landsat 7 automatic cloud cover assessment. Algorithms for Multispectral, Hyperspectral, and Ultraspectral Imagery VI. *Int. Soc. Opt. Photonics* **2000**, *4049*, 348–355.
- Zhu, Z.; Woodcock, C.E. Object-based cloud and cloud shadow detection in Landsat imagery. *Remote Sens. Environ.* **2012**, *118*, 83–94. [CrossRef]
- Zhu, Z.; Wang, S.; Woodcock, C.E. Improvement and expansion of the Fmask algorithm: Cloud, cloud shadow, and snow detection for Landsats 4–7, 8, and Sentinel 2 images. *Remote Sens. Environ.* **2015**, *159*, 269–277. [CrossRef]
- Foga, S.; Scaramuzza, P.L.; Guo, S.; Zhu, Z.; Dilley, R.D., Jr.; Beckmann, T.; Schmidt, G.L.; Dwyer, J.L.; Hughes, M.J.; Laue, B. Cloud detection algorithm comparison and validation for operational Landsat data products. *Remote Sens. Environ.* **2017**, *194*, 379–390. [CrossRef]
- Hughes, M.J.; Hayes, D.J. Automated detection of cloud and cloud shadow in single-date Landsat imagery using neural networks and spatial post-processing. *Remote Sens.* **2014**, *6*, 4907–4926. [CrossRef]
- Li, Z.; Shen, H.; Li, H.; Xia, G.; Gamba, P.; Zhang, L. Multi-feature combined cloud and cloud shadow detection in GaoFen-1 wide field of view imagery. *Remote Sens. Environ.* **2017**, *191*, 342–358. [CrossRef]
- Zhu, Z.; Woodcock, C.E. Automated cloud, cloud shadow, and snow detection in multitemporal Landsat data: An algorithm designed specifically for monitoring land cover change. *Remote Sens. Environ.* **2014**, *152*, 217–234. [CrossRef]
- Candra, D.S.; Phinn, S.; Scarth, P. Automated Cloud and Cloud-Shadow Masking for Landsat 8 Using Multitemporal Images in a Variety of Environments. *Remote Sens.* **2019**, *11*, 2060. [CrossRef]
- Zhu, X.X.; Tuia, D.; Mou, L.; Xia, G.S.; Zhang, L.; Xu, F.; Fraundorfer, F. Deep learning in remote sensing: A comprehensive review and list of resources. *IEEE Geosci. Remote Sens. Mag.* **2017**, *5*, 8–36. [CrossRef]
- Xie, F.; Shi, M.; Shi, Z.; Yin, J.; Zhao, D. Multilevel cloud detection in remote sensing images based on deep learning. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2017**, *10*, 3631–3640. [CrossRef]
- An, Z.; Shi, Z. Scene learning for cloud detection on remote-sensing images. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2015**, *8*, 4206–4222. [CrossRef]
- Zhang, Q.; Xiao, C. Cloud detection of RGB color aerial photographs by progressive refinement scheme. *IEEE Trans. Geosci. Remote Sens.* **2014**, *52*, 7264–7275. [CrossRef]

15. Li, Z.; Shen, H.; Wei, Y.; Cheng, Q.; Yuan, Q. Cloud detection by fusing multi-scale convolutional features. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* **2018**, *4*, 149–152. [[CrossRef](#)]
16. Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, Boston, MA, USA, 7–12 June 2015; pp. 3431–3440.
17. Segal-Rozenhaimer, M.; Li, A.; Das, K.; Chirayath, V. Cloud detection algorithm for multi-modal satellite imagery using convolutional neural-networks (CNN). *Remote Sens. Environ.* **2020**, *237*, 111446. [[CrossRef](#)]
18. Zi, Y.; Xie, F.; Jiang, Z. A cloud detection method for Landsat 8 images based on PCANet. *Remote Sens.* **2018**, *10*, 877. [[CrossRef](#)]
19. Yang, J.; Guo, J.; Yue, H.; Liu, Z.; Hu, H.; Li, K. CDnet: CNN-based cloud detection for remote sensing imagery. *IEEE Trans. Geosci. Remote Sens.* **2019**, *57*, 6195–6211. [[CrossRef](#)]
20. Guo, J.; Yang, J.; Yue, H.; Tan, H.; Hou, C.; Li, K. CDNetv2: CNN-Based cloud detection for remote sensing imagery with cloud-snow coexistence. *IEEE Trans. Geosci. Remote Sens.* **2020**, *59*, 700–713. [[CrossRef](#)]
21. Shahraki, F.F.; Prasad, S. Graph convolutional neural networks for hyperspectral data classification. In Proceedings of the 2018 IEEE global conference on signal and information processing (GlobalSIP), Anaheim, CA, USA, 26–29 November 2018; pp. 968–972.
22. Qin, A.; Shang, Z.; Tian, J.; Wang, Y.; Zhang, T.; Tang, Y.Y. Spectral-spatial graph convolutional networks for semisupervised hyperspectral image classification. *IEEE Geosci. Remote Sens. Lett.* **2018**, *16*, 241–245. [[CrossRef](#)]
23. Pu, S.; Wu, Y.; Sun, X.; Sun, X. Hyperspectral Image Classification with Localized Graph Convolutional Filtering. *Remote Sens.* **2021**, *13*, 526. [[CrossRef](#)]
24. Hong, D.; Gao, L.; Yao, J.; Zhang, B.; Plaza, A.; Chanussot, J. Graph convolutional networks for hyperspectral image classification. *IEEE Trans. Geosci. Remote Sens.* **2020**, *59*, 5966–5978. [[CrossRef](#)]
25. Wang, B.; Ono, A.; Muramatsu, K.; Fujiwara, N. Automated detection and removal of clouds and their shadows from Landsat TM images. *IEICE Trans. Inf. Syst.* **1999**, *82*, 453–460.
26. Mahajan, S.; Fataniya, B. Cloud detection methodologies: Variants and development—A review. *Complex Intell. Syst.* **2020**, *6*, 251–261. [[CrossRef](#)]
27. U.S. Geological Survey. *L8 Biome Cloud Validation Masks*; U.S. Geological Survey: Reston, VA, USA, 2016. [[CrossRef](#)]
28. Gómez-Chova, L.; Camps-Valls, G.; Calpe-Maravilla, J.; Guanter, L.; Moreno, J. Cloud-screening algorithm for ENVISAT/MERIS multispectral images. *IEEE Trans. Geosci. Remote Sens.* **2007**, *45*, 4105–4118. [[CrossRef](#)]
29. Zhang, Y.; Guindon, B.; Cihlar, J. An image transform to characterize and compensate for spatial variations in thin cloud contamination of Landsat images. *Remote Sens. Environ.* **2002**, *82*, 173–187. [[CrossRef](#)]
30. Kohonen, T. Self-organized formation of topologically correct feature maps. *Biol. Cybern.* **1982**, *43*, 59–69. [[CrossRef](#)]
31. Scaramuzza, P.L.; Bouchard, M.A.; Dwyer, J.L. Development of the Landsat data continuity mission cloud-cover assessment algorithms. *IEEE Trans. Geosci. Remote Sens.* **2011**, *50*, 1140–1154. [[CrossRef](#)]
32. Kiviluoto, K. Topology preservation in self-organizing maps. In Proceedings of the International Conference on Neural Networks (ICNN'96), Washington, DC, USA, 3–6 June 1996; Volume 1, pp. 294–299.