Classification of Satellite Images Using Deep Learning Models

Saurabh Toraskar
Roll no. 21290

Indian Institute of Science Education and Research Bhopal

Abstract

Satellite image classification has applications in environmental monitoring, urban planning, disaster response, and many other fields. Manual methods exist but are very time-consuming. Machine learning (ML) techniques provide an automated approach to this problem. Firstly, we analyzed the pixel intensity histograms of the classes and observed the non-similarity of their distributions. To feed our data into machine learning models, we separated the channels and flattened them. Further, to reduce the number of features from 40000 to 1000, we utilized Principal Component Analysis (PCA). 70% of data was utilized for training and 30% of data was separated for validation purposes. For hyperparameter tuning, we performed a three-fold cross-validated grid search on XGBoost, logistic regression, SVM classifier, decision tree, and random forest. XGBoost outperforms all the machine learning models and also Convolution Neural Networks (CNNs) by exhibiting an overall accuracy, precision, recall, and f1-score of 0.98 and a perfect score of 1 in identifying two classes. This study shows that tuned XGBoost model, even though simple, is reliable for satellite image classification.

Introduction

One of the most fundamental areas of research in remote sensing is satellite image classification. The applications of satellite image classification are not limited to earth and environmental sciences, it also finds its application in various fields for providing valuable insights and decision-making. These fields include environmental monitoring, urban planning, disaster management, climate change studies, security, and defense. In satellite image classification, pixels or images derived from satellite data are categorized into categories that give meaningful information. Some of the common categories are water bodies, vegetation, bare soil, clouds, ice, wetlands, and burned areas. The approach undertaken for the classification task can be manual or automated which leverages the effectiveness of different machine learning and deep learning models. Machine learning can be supervised which requires human interference for providing labeled data or it can be fully automated unsupervised learning. In this study, we focus on satellite image classification using supervised machine learning and deep learning algorithms.

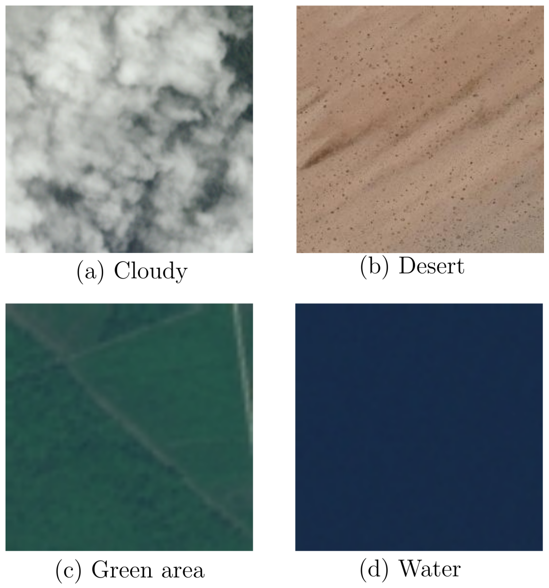


Figure 1: Representative images from each category: cloudy, desert, green area, and water.

Literature review

In recent times, machine learning models have been extensively used for satellite image classification. One such example is the study undertaken by Madhura and Venkatachalam where they compare various supervised classification methods such as the Maximum Likelihood classifier, Minimum Distance to Mean classifier, and Mahalanobis classifier on the study area of Chennai, India. The best performance is shown by the Maximum Likelihood classifier which gives an accuracy of 93.33% and kappa statistic of 0.92. Camps-Valls and Bruzzone utilize and compare different kernel-based methods for hyperspectral image classification. One application of image classification is crop mapping which is done by Pignatti et.al where they have compared Support Vector Machine (SVM) and 3-D Convolution Neural Networks (CNNs).

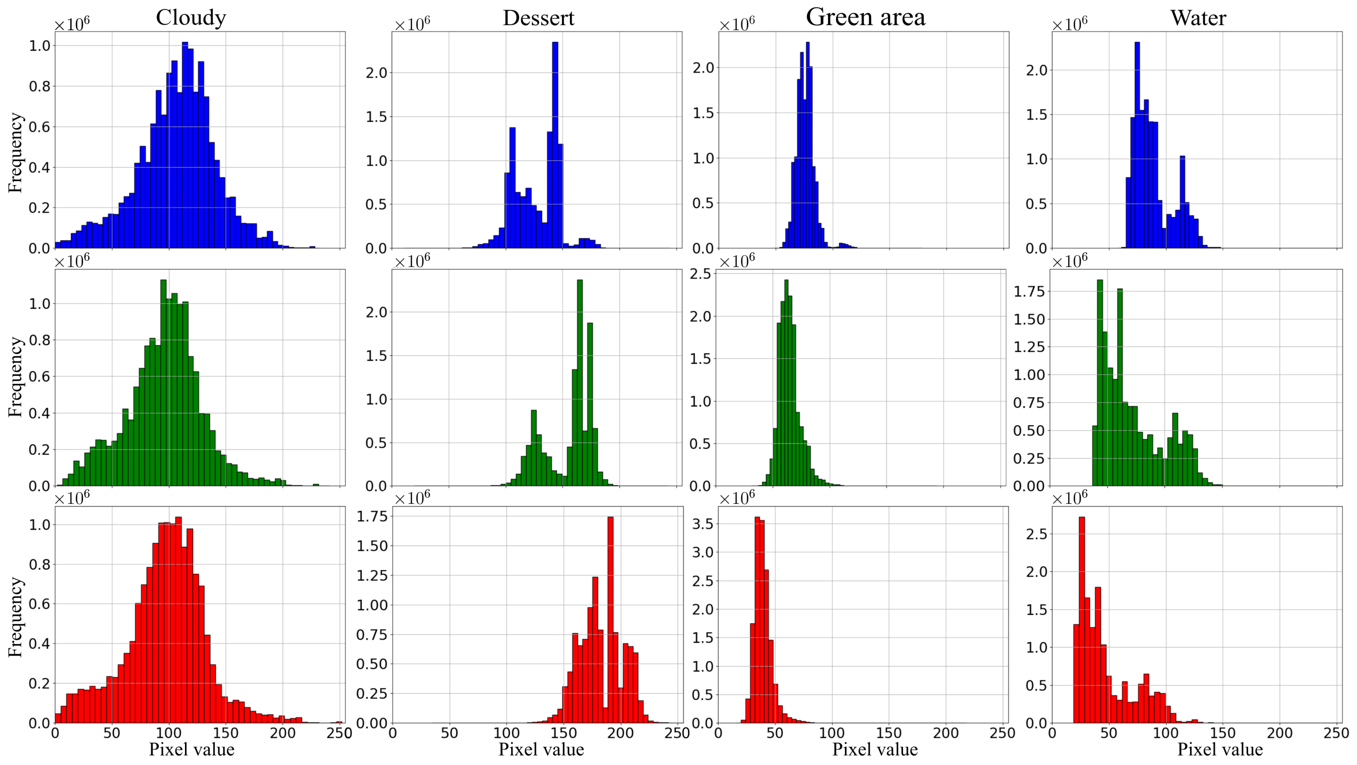


Figure 2: Pixel intensity histogram of the red, blue, and green channels for each class. The color of the histogram corresponds to the color of the channel it represents. It can be observed that the distribution of histograms for channels for a particular class is similar with their peaks shifting. The distribution across classes is significantly different. The peaks of "Green area" and "Water" for each channel are approximately coinciding

Table 1: Distribution of the dataset

Label	Number of Images	Number of channels
Cloudy	1500	4
Desert	1131	3
Green area	1500	3
Water	1500	3

Dataset

For supervised machine learning, the quality and consistency of the labeled data play a critical role. The dataset used in this study was provided as part of the course project. It is a subset of the RSI-CB256 dataset. The dataset has annotated images in *.jpg* format with the following classes: cloudy, desert, green area, and water. The distribution of the number of images and number of channels in each class is given in Table 1. One thing to note is that only the cloudy channel has a fourth channel, even though the value of all the pixels is 0, the presence of this channel can be exploited for feature extraction. A representative image of each class is shown in Fig 1

Exploratory data analysis

Exploratory data analysis (EDA) is a critical pre-processing step. In EDA, data is examined to uncover any underlying patterns, trends, and relationships that can be useful in the model development process. Pixel intensity histograms are used as the first step of EDA. Pixel intensity histograms show the distribution of pixel brightness in an image or a group of images. We have plotted the histograms for the red, blue, and green channels for each class in Fig. 2. For data handling, *pandas* and for visualization and plotting, *matplotlib* libraries were used.

We can observe in a particular class, the images have similar distribution across channels with their peaks being shifted. Another key observation from this analysis is that different classes have notably different distributions and peaks. Thus, models relying on pixel intensity values can be expected to give good results. It may be also noted that even though "Forest" and "Water" have different distributions, their peaks are around the same pixel intensity value.

Methods

Data preprocessing

Most machine learning models expect input data in tabular form. These models may not be designed for image classification tasks, but their effectiveness and robustness make

Table 2: The hyperparameters considered for grid search for each model and its best-performing set of hyperparameters.

Model	Hyperparameters	Values considered	Best parameters
XGBoost	n_estimators	{100, 150, 200}	200
	max_depth	{3, 6, 10}	6
	learning_rate	{0.01, 0.1}	0.1
	gamma	{0,0.1}	0
	subsample	{0.8, 1}	1
Logistic Regression	C	{0.1, 1, 10	0.1
	penalty	{'l1', 'l2', 'elasticnet'}	'l1'
	solver	{'liblinear', 'lbfgs'}	'liblinear
SVC	C	{1, 10, 20}	10
	kernal	{'linear', 'rbf'}	'rbf'
	gamma	{'scale', 'auto'}	'auto'
Decision Tree	criterion	{'gini', 'entropy'}	'entropy'
	max_depth	{None, 5, 10, 20}	None
	min_samples_split	{2, 10, 20}	10
	min_samples_leaf	{1, 5, 10}	1
Random Forest	n_estimators	{50, 100, 200}	200
	max_depth	{10, 20, 30}	20
	min_samples_split	{2, 10, 20}	2
	min_samples_leaf	{1, 5, 10}	1
	max_features	{'auto', 'sqrt', 'log2'}	'sqrt'

them a probable candidate. Also, we see in Section that a model relying solely upon pixel intensities could give good results. Thus to harness the capabilities of these models we need to convert our data into a tabular form. Initially, all the images were resized to a resolution of 100×100, then the available channels were separated. Were the 4^{th} channel was not present, an artificial value of 255 signifying opaqueness was introduced. Consequently, all the channel grids were flattened resulting in a single row vector of 40000 columns.

To reduce the computational expense and training times of the model, we utilized Principal Component Analysis (PCA) to reduce the dimensionality of the data while avoiding significant data loss. PCA is a linear transformation technique that projects high-dimensional data onto a lower-dimensional orthogonal subspace by identifying the eigenvectors of the covariance matrix. For determining the number of principal components (PCs) to be used for this study, one dataset comprising of 100 PCs and one of 1000 PCs were used. Upon performing the initial computations, the dataset with 1000 PCs yielded more favorable and satisfactory results and thus was finalized. Finally, the scaling of the data was done using *StandardScalar* from the scikit-learn library of Python which scales the feature in such a way that each feature has a mean of 0 and standard deviation of 1. This is done there is no bias of feature value scales.

Model development

As inferred from previous sections, machine learning models that take tabular input and rely solely upon pixel intensities could also give good results. For this project, we have used Python's *scikit-learn* library for model implementation and evaluation. The machine learning models used for this study are:

- **Decision Tree (DT)** is a tree-based model that iteratively splits the dataset based on feature values.
- **Logistic Regression (LR)** is a linear model which can also be used for multiclass classification using techniques like softmax and one vs rest for predicting probabilities of classes.
- **Random Forest (RF)** is an ensemble model that constructs multiple decision trees and aggregates their prediction.
- **Extreme Gradient Boosting (XGBoost)** is a boosting algorithm that iteratively builds decision trees, improving each tree based on mistakes of the previous ones.

Table 3: Evaluation metrics of the tuned models.

Model	Accuracy	Precision	Recall	F1-Score
Xgboost	98	98	98	98
Losgistic Regression	86	87	87	87
SVC	79	81	80	80
Decision Tree	94	95	95	95
Random Forest	94	95	95	95
CNN	87	90	88	87

- **Support Vector Classifier (SVC)** constructs a hyperplane that splits the dataset into classes.

Hyperparameters of all the above models are tuned using GridSearch with a 3-fold cross-validation. The details of the hyperparameters considered for the models are reported in the Table (to be added)

Convolution Neural Networks (CNNs) are deep learning models developed for image classification tasks. These models take images as inputs directly and thus pre-processing of images is not required for these models. However, the performance of these models is heavily dependent on its architecture, and constructing an appropriate architecture for a particular problem is very complex. Additionally, deep learning models are recognized to perform better on large datasets and there is a possibility the number of data points in our dataset may not be adequate.

The CNN architecture used in this study consists of three convolutional layers with a ReLU activation function, each followed by max-pooling layers that extract hierarchical features. It then includes a fully connected dense layer with 128 neurons and dropout for regularization, the architecture ends with a softmax output layer for multi-class classification

Model evaluation

Model evaluation tests the model on unseen data. From the original dataset, 30% of the data is kept for evaluation purposes. The evaluation metrics used in this study are:

- **Accuracy**: It is the proportion od correctly identified instances from the data. The formula is given by:

$$Accuracy = \frac{Correct\,prediction}{All\,instances} \tag{1}$$

- **Precision**: The proportion of true positive predictions out of all positive predictions made by the model.

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

This is the equation for calculating the precision for a single class in a binary class problem. In a multi-class problem, the precision of each class is calculated by the one-vs-rest approach and then the arithmetic mean is calculated.

- **Recall**: The proportion of true positive predictions out of all actual positive instances. The formula is given by:

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

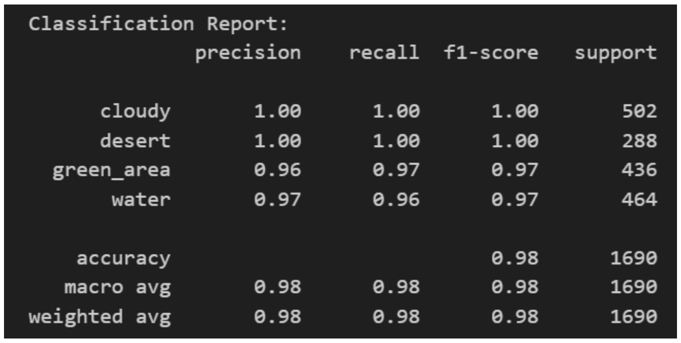
Similar to precision, for multi-class problems, we take the arithmetic mean of the recall of all classes.

- **F1-Score**: It is the harmonic mean of precision and recall. Takin the harmonic mean helps to maintain a balance between the two. The formula for it is:

$$F1 - Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{4}$$

Results

The best-performing hyperparameters of each model are reported in Table 2. Table 3 gives all the tuned models' evaluation metrics. We can see that XGBoost outperforms all the considered algorithms with 0.98 accuracy, 0.98 macro-averaged precision, and recall, and 0.98 f1-measure. It indicates that the model's predictions are balanced and it is not biased to any class in particular. Thus we can conclude that XGBoost is the most suitable model for such satellite image classification tasks.



```
Classification Report:
              precision    recall  f1-score   support

      cloudy       1.00      1.00      1.00       502
      desert       1.00      1.00      1.00       288
  green_area       0.96      0.97      0.97       436
       water       0.97      0.96      0.97       464

    accuracy                           0.98      1690
   macro avg       0.98      0.98      0.98      1690
weighted avg       0.98      0.98      0.98      1690
```

Figure 3: Classification report of the best performing tuned model. It illustrates the class-wise, macro-averaged, and weighted average of precision, recall, and f1-score. We can observe that the selected XGBoost can identify 2 classes perfectly.

To get an in-depth analysis, we further analyzed the results of our best-performing model by calculating the class-wise precision, recall, and f1-score. These results are illustrated in Fig. 3, where we can see that our model scores perfectly for the ”Cloudy” and ”Desert” classes. The precision and recall of the remaining two classes are complimentary which indicates that the model is confused in these two classes. A possible reason could be that as observed in Fig. 2, even though the distributions may be different the peaks of the pixel histograms of the ”Green Area” and ”Water” coincide.

Limitations and future work

We have demonstrated that our selected model can accurately classify the given satellite images. Still, it has some limitations. First, no spatial or geographical context was provided with the dataset’s images. Second, the number of images is less to employ deep-learning models for significantly better results.

To further develop this work, the machine learning framework could be employed for the different datasets with a higher number of classes and also across different regions or on a global scale for satellite image classification.

Conclusion

In this study, we applied various preprocessing techniques and developed a highly accurate and robust model to classify satellite images using machine learning and deep learning models.