NATIONAL TECHNICAL UNIVERSITY OF ATHENS

MSC DATA SCIENCE AND MACHINE LEARNING

Geospatial Big Data Analytics
Project 3

Pavlos Kalfantis
MSc Student
pavloskalfantis@mail.ntua.gr
Student ID: 03400134

June 2022

1 Introduction

The scope of the third semester project of the graduate course 'Geospatial Big Data Analytics' was to design and train machine learning and deep learning algorithms that classify hyperspectral images into 14 land cover classes. The dataset that is used for this project is the HyRANK Hyperspectral Dataset, which contains two training images (Dioni and Loukia) and three validation images (Erato, Nefeli and Kiriki). There are 176 different bands on each image of the dataset. The two training images are accompanied by the corresponding ground truth, which has the land cover classification of certain pixels of the image. An important thing to note is that the majority of the pixels in the training images does not have a corresponding land cover label and will not be used for supervised training. The two training images, the three validation images and the land cover classes are shown on figures 1, 2 and 3, as presented on the dataset website. This report is accompanied by a notebook that contains all the code used, and for that reason the code is not presented here as well. The report will explain all the steps that were taken on the notebook, and the reader is referred to the notebook for the exact implementation.

Firstly, the dataset is downloaded in the Google Colab environment. We then visualize the training images on a true color composite as shown on figures 4 and 5. True color composite uses visible light bands red, green and blue in the corresponding red, green and blue color channels, resulting in a natural colored result, that is a good representation of the image as humans would see it naturally.

The predictive task of the models that are developed as part of this project is to classify each pixel of the three validation images into one of the 14 land cover classes. There are many different approaches that can be taken to tackle this problem. The three main categories that these approaches fall into are Pixel-based Classification, Patch Classification and Semantic Segmentation. Each approach has a different way of loading the data into the classifier, as will be explained in the subsequent sections. For all the machine learning algorithms that are trained on our dataset for this project, the same results are presented. A Confusion Matrix showing the number of test data points that are classified on each class, and a Classification Report showing the classification metrics (precision, recall, f1-score and accuracy) for each category, as well as their averages.

2 Pixel-based Classification

The first approach that can be taken in order to classify the hyperspectral images into the 14 land cover categories, is to treat each pixel of the training image as an independent data point. That means that the dataset is not viewed as an image but rather as a vector of 176 features (equal to the number of bands). The first models that are trained for this task follow this approach, where we keep only the pixels of the two training images that have a corresponding land cover class assigned. The total number of training examples are 33527 for both of the training images.

After downloading the dataset, filtering out the pixels without an assigned class and

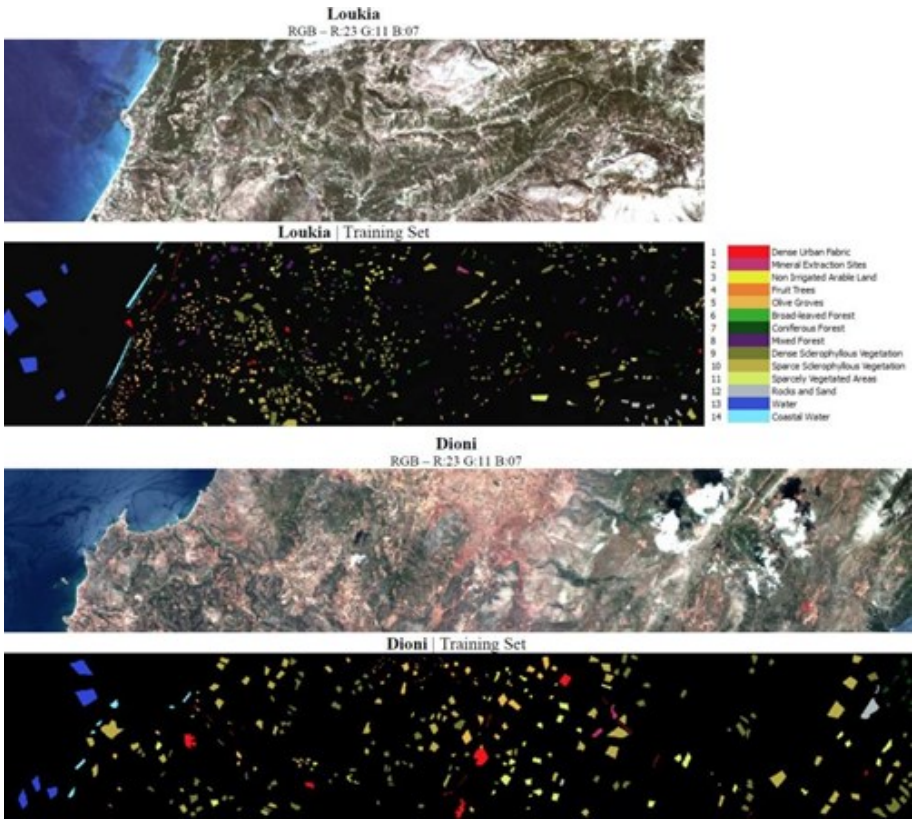

Figure 1: Training set images and corresponding ground truth

concatenating the pixels of the two images, we get our feature matrix of size (33527, 176) and our target vector of size (33527), which can be used in classic machine learning and deep learning algorithms. There are many different algorithms that can be explored in order to learn a suitable feature representation and in this project, three different algorithms are tested: A Random Forest Classifier, a Support Vector Machines Classifier and a Multi-Layer Perceptron classifier.

2.1 Random Forest Classifier

For the Random Forest Classifier, the meta-estimator RandomForestClassifier, offered in the ensemble package of the scikit-learn toolbox, was implemented. The estimation is based on the combination of independent decision trees, built on different random subsets of the train set. The main hyperparameters of this classifier is the number of decision trees, their max depth and the criterion of the quality of each split. After a number of experiments the accuracy that this classifier achieved on the test set was 90%. Since the values of the data for each band range from 0 to 10517, scikit-learn's StandardScaler was used in order to standardize features by removing the mean and scaling to unit variance. The confusion matrix and the classification report are shown

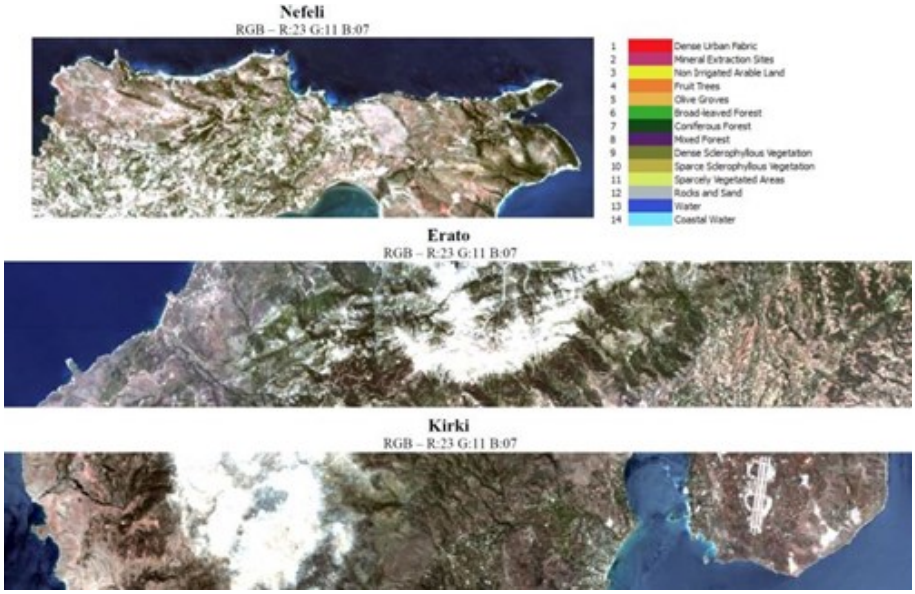

Figure 2: Validation set images

Figure 3: Land cover classes

on figures 6 and 7

2.2 Support Vector Machines Classifier

Another machine learning algorithm that is widely used for classification tasks is Support Vector Machines. The advantages of support vector machines are:

- Effective in high dimensional spaces.
- Still effective in cases where number of dimensions is greater than the number of samples.
- Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.

Figure 4: Loukia Training Image in True Color Composite

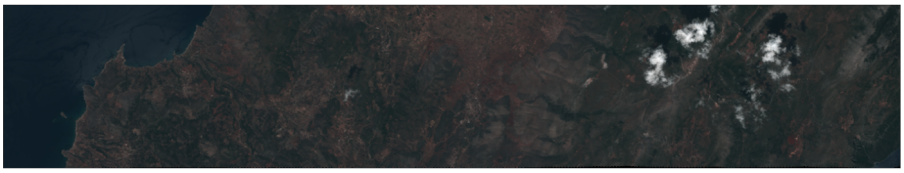

Figure 5: Dioni Training Image in True Color Composite

- Versatile: different Kernel functions can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels.

The radial basis function (RBF) kernel is used for our experiments and the classifier achieves an accuracy score of 87%.The confusion matrix and the classification report are shown on figures 8 and 9

2.3 Multi-layer Perceptron

The next machine learning model that is explored for the classification task of each pixel independently is the Multi-layer Perceptron. This model is a fully connected class of a feedforward artificial neural netrwork, with one ore more hidden layers of nodes. For this purpose, Pytorch Lightning is used for dataset preparation and model training. There are four total layers used in our network (the input layer that transforms the 176 features into 512, the first hidden layer that reduces the features to 256, the third layer that reduces the features to 128 and finally the output layer that classifies each input into the 14 classes). The rectified linear unit (ReLU) non linear function is selected and our architecture consists of two methods of regularization: Dropout and L1 regularization to the Neural Network's weights. The values of the dataset are again scaled using the standard scaler and then the dataset is split into train, validation and test datasets and loaded into DataLoaders (that are used for batch training). The validation set is used to tune the hyperparameters of the model, such the learning rate and the dropout probability and the test set is used to compute the metrics. The total number of trainable parameters for this model is 256 thousand. The accuracy achieved with this approach is 91% and the confusion matrix and the classification report are shown on figures 10 and 11.

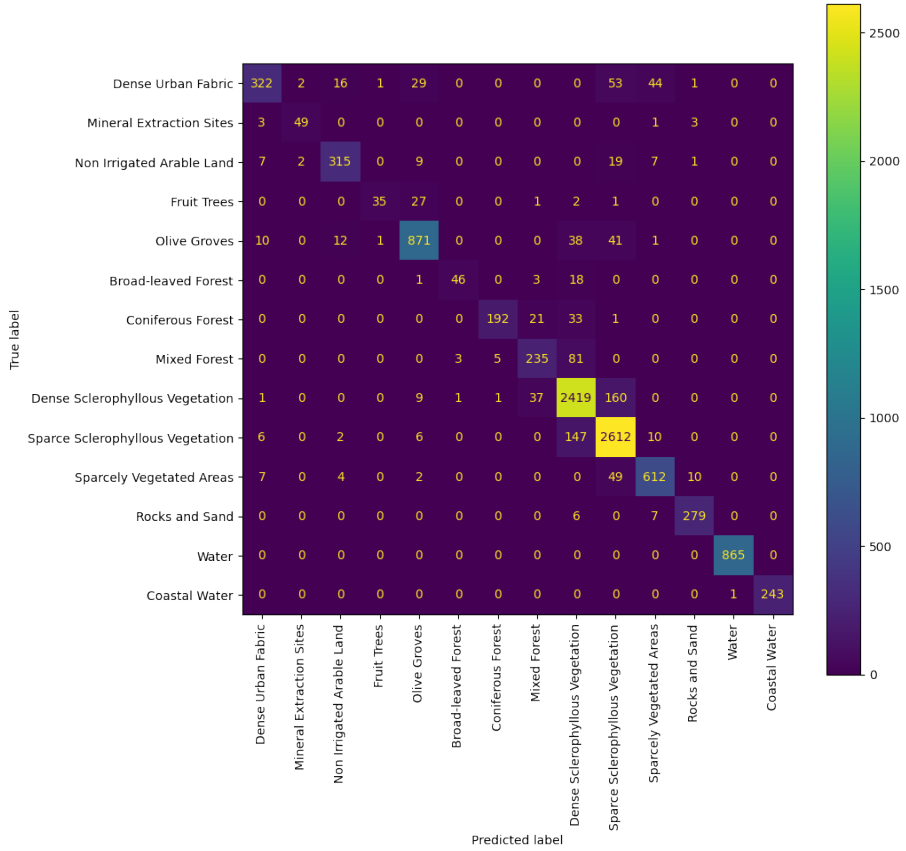

Figure 6: Random forest confusion matrix

3 Patch-based Classification

The next approach that is used for this problem is using Convolutional Neural Networks (CNNs) to classify sub-images of the training image. More specifically, the dataset that we create consists of small subsamples of the original image that take as a label the label of the center pixel of the sub-image. With this approach, we can make use of neighboring pixels of each target pixel in the predictive model and use the classic computer vision tool of the convolution. There are two model architectures that are explored. The first model is a novel CNN architecture that is trained from scratch, whereas the second model uses the technique of Transfer Learning in order to use a pretrained model to encode the input images.

3.1 CNN

The architecture of our convolutional neural network consists of 3 convolutional layers, followed by a ReLU nonlinear activation and max pooling layer. After the last convo-

	precision	recall	f1-score	support
Dense Urban Fabric	0.91	0.74	0.81	449
Mineral Extraction Sites	0.92	0.88	0.90	78
Non Irrigated Arable Land	0.91	0.89	0.90	335
Fruit Trees	1.00	0.45	0.62	83
Olive Groves	0.90	0.89	0.90	988
Broad-leaved Forest	0.78	0.63	0.70	75
Coniferous Forest	0.92	0.82	0.87	282
Mixed Forest	0.77	0.71	0.74	331
Dense Sclerophyllous Vegetation	0.88	0.91	0.90	2645
Sparce Sclerophyllous Vegetation	0.89	0.94	0.91	2726
Sparcely Vegetated Areas	0.91	0.91	0.91	653
Rocks and Sand	0.94	0.94	0.94	280
Water	0.99	1.00	1.00	899
Coastal Water	1.00	0.97	0.98	235
accuracy			0.90	10059
macro avg	0.91	0.83	0.86	10059
weighted avg	0.90	0.90	0.90	10059

Figure 7: Random forest classification report

lutional layer, an adaptive max pooling layer is used before the encoded image is fed into a fully connected feedforward neural network with 1 hidden layer. Different sizes of the layers are explored and another key parameter in these experiments is the size of the subsampled image. Sizes of 7x7, 11x11, 15x15 and 17x17 pixel images are tested in order to find the optimal solution. The best model with the batch size of 15x15 achieved an accuracy of 98% on the test set and the confusion matrix and the classification report are shown on figures 12 and 13. In addition, on figures 14 and 15, the training figures showing the loss and the accuracy for the validation set for all the different experiments are presented.

3.2 CNN with Transfer Learning

The second model based on the convolutional neural network architecture uses transfer learning for classifying the pixels into the 14 classes. Transfer learning is a research problem in machine learning that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem. In our case, the images are given as an input to the pretrained ResNet18 model and then the encoded information is given to a feedforward neural network with two hidden layers for the classification task. There are two different learning rates that are used while training, as the pretrained model weights do not need to be learned, but rather slightly changed for the specific task. Another important note is that the ResNet18 model is trained on three band RGB images, so when creating the dataset, only the Red, Green and Blue bands of the training images are used. The best model with the batch size of 17x17 achieved an accuracy of 98% on the test set and the confusion matrix and the classification report are shown on figures 16 and 17

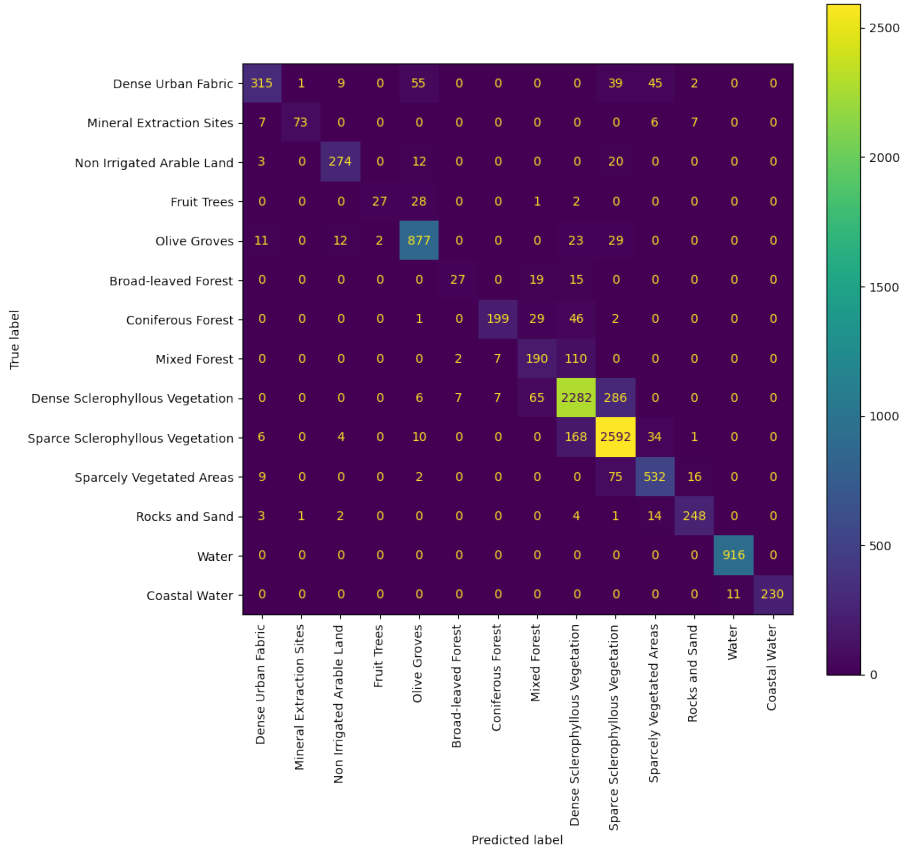

Figure 8: SVM confusion matrix

4 Semantic Segmentation

The last approach that can be explored in order to classify the satellite images into the 14 land classes is Semantic Segmentation. In a Semantic Segmentation model, each pixel of the input image is classified to one of the categories. One of the most common architectures in Semantic Segmentation problems is U-Net. It stems from the so-called 'fully convolutional network" and the main idea is to supplement a usual network by successive layers, where pooling operations are replaced by upsampling operators. Hence these layers increase the resolution of the output. A successive convolutional layer can then learn to assemble a precise output based on this information.

One important modification in U-Net is that there are a large number of feature channels in the upsampling part, which allow the network to propagate context information to higher resolution layers. As a consequence, the expansive path is more or less symmetric to the contracting part, and yields a u-shaped architecture. The network only uses the valid part of each convolution without any fully connected layers.[2] To predict the pixels in the border region of the image, the missing context is extrapolated

	precision	recall	f1-score	support
Dense Urban Fabric	0.89	0.68	0.77	466
Mineral Extraction Sites	0.97	0.78	0.87	93
Non Irrigated Arable Land	0.91	0.89	0.90	309
Fruit Trees	0.93	0.47	0.62	58
Olive Groves	0.88	0.92	0.90	954
Broad-leaved Forest	0.75	0.44	0.56	61
Coniferous Forest	0.93	0.72	0.81	277
Mixed Forest	0.62	0.61	0.62	309
Dense Sclerophyllous Vegetation	0.86	0.86	0.86	2653
Sparce Sclerophyllous Vegetation	0.85	0.92	0.88	2815
Sparcely Vegetated Areas	0.84	0.84	0.84	634
Rocks and Sand	0.91	0.91	0.91	273
Water	0.99	1.00	0.99	916
Coastal Water	1.00	0.95	0.98	241
accuracy			0.87	10059
macro avg	0.88	0.79	0.82	10059
weighted avg	0.87	0.87	0.87	10059

Figure 9: SVM classification report

by mirroring the input image.

In this project, a U-Net architecture is implemented in Pytorch Lightning. There are 4 downsampling and upsampling layes connected with skip connections. In order to create the dataset to be used for this task, we use the same patch based approach that was used for the CNNs, with a large image size (200x200 pixels). Because of the large number of images that can be created with this image size and a stride of 1, we only take a random sample of user-specified size out of all the possible sub images. The difference of this dataset to the one used in CNNs is that now the output y is an array of labels (called a mask) instead of a single label of each image. We create 3000 images of the specified size before splitting them into training, validation and test sets. Since as we discussed in the beginning of this report, many pixels in the training images do not have an assigned label, these pixels that are kept in the training set should not contribute to the calculation of the loss and that is why the cross entropy loss of the training_step and validation_step is changed to reflect that.

After the model is trained on the GPU of Colab for 10 epochs, we print the classification report and confusion matrix of the test set on figures 18 and 19. We can see that since each pixel is a different data point, the 300 images of size 200x200 that were used as the test dataset (10% of the total 3000 images of the dataset) correspond to 730000 data points (after the ones that do not have a corresponding label are deleted).

5 Prediction of Validation Images

We select the custom convolutional neural network (without using the pretrained Resnet18 model for transfer learning) with patch size equal to 15 as the best model of this project. The final step of this project is to predict the class labels of the three validation images (Nefeli, Erato and Kirki) and save them in tif format. In order to do that, a custom function is created that loads the validation images, creates batches equal to 15x15 in order to predict the class of each label and saves the output in the required format. Figures 20, 21, 22, 23, 24 and 25 show the three validation images and the corresponding class

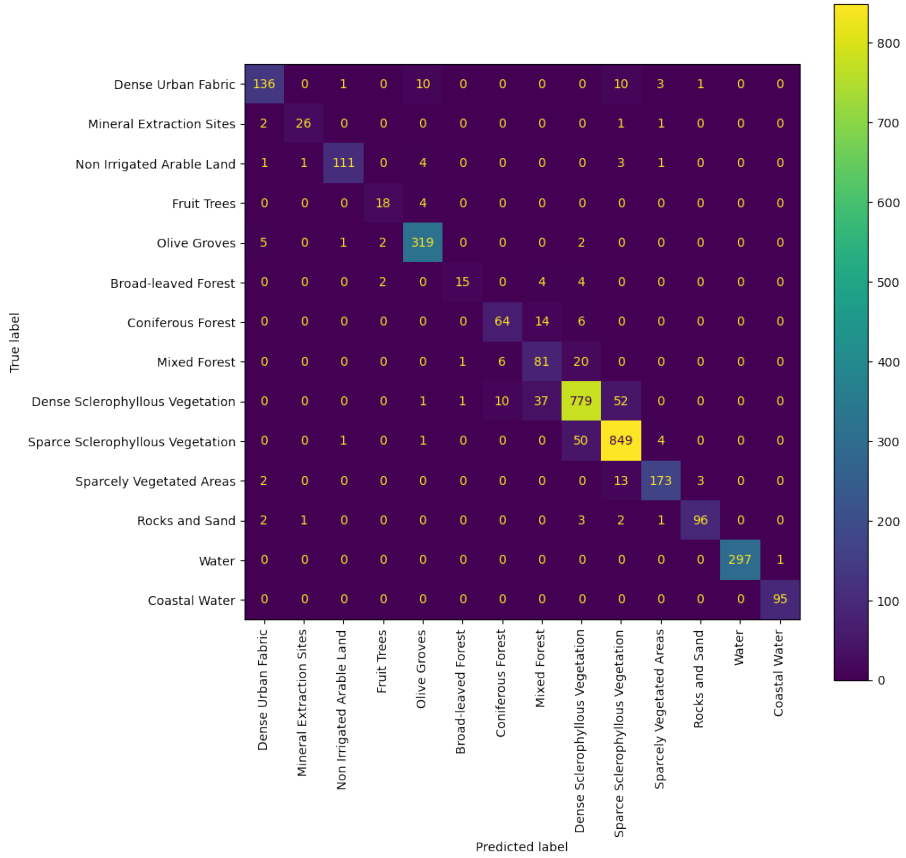

Figure 10: Multi-layer perceptron confusion matrix

predictions.

6 Potential Improvements

The approach that was taken for this project was combining both training images and then splitting them into a train set and a validation set. This was the case for all different models that were created over the course of this exercise, from Pixel-based Classification, to the Convolutional Neural Networks and the Semantic Segmentation model. This approach, however, has some limitations on the predictive power that these models can have on new images.

In the case of pixel-based classification, neighboring pixels can contain essentially the same information on a satellite image. This means that if we have a large area with, for example, coastal water, two pixels that are next to each will have almost the same values for all 176 bands. By splitting randomly the dataset into train and validation sets, there is a large possibility that one of the two example pixels will end up on the training

	precision	recall	f1-score	support
Dense Urban Fabric	0.92	0.84	0.88	161
Mineral Extraction Sites	0.93	0.87	0.90	30
Non Irrigated Arable Land	0.97	0.92	0.94	121
Fruit Trees	0.82	0.82	0.82	22
Olive Groves	0.94	0.97	0.96	329
Broad-leaved Forest	0.88	0.60	0.71	25
Coniferous Forest	0.80	0.76	0.78	84
Mixed Forest	0.60	0.75	0.66	108
Dense Sclerophyllous Vegetation	0.90	0.89	0.89	880
Sparce Sclerophyllous Vegetation	0.91	0.94	0.93	905
Sparcely Vegetated Areas	0.95	0.91	0.93	191
Rocks and Sand	0.96	0.91	0.94	105
Water	1.00	1.00	1.00	298
Coastal Water	0.99	1.00	0.99	95
accuracy			0.91	3354
macro avg	0.90	0.87	0.88	3354
weighted avg	0.91	0.91	0.91	3354

Figure 11: Multi-layer perceptron classification report

set, whereas the other on the validation set. This can cause overfitting on the model, as we are testing and training on very similar data points. In addition this can cause the results that were obtained (90% for the Multi Layer Perceptron) to be misleading and most probably will not be achieved on a completely new training image.

Similar is the case in the Patch Based classification, whether it is with Convolutional Nets or with the Semantic Segmentation approach. By using a sliding window of one, we are essentially creating patch images that are very similar. We are then splitting these images into the train and validation set and that can cause the same problems we described above for the pixel-based classification.

In fact some tests were run where we use only one of the images on the training set and the other on the test set and the same algorithms that we described and had accuracy more than 90& ended up with very poor results, as seen for example on figures 26 and 27, where we trained the exact same CNN we picked as the best in the previous paragraph, this time only on one of the images. The figure represents the results of using this model on the other image for validation and we can see the poor results it achieved.

For a better predictive model, some potential steps that can improve the performance of our predicting models are presented:

First approach can be to keep tuning our machine learning models and training them only on one of the images. By adding more layers or tweaking the architecture of the model there is a possibility that the accuracy on the other image can increase. We can use methods like skip connections to battle overfitting. However in this approach it is hard for the model to escape overfitting, as the model is learning only on one image with specific morphology and geography.

Another approach can be to take fewer training data points out of the two images. In the pixel-based classification, we can choose only pixels with a certain distance to each other and put them on our training set, thus decreasing the possibility of having very similar data points go to the train and validation set. In the patch-based classification, we can use a stride on our sliding window that is greater than one (or even equal to the

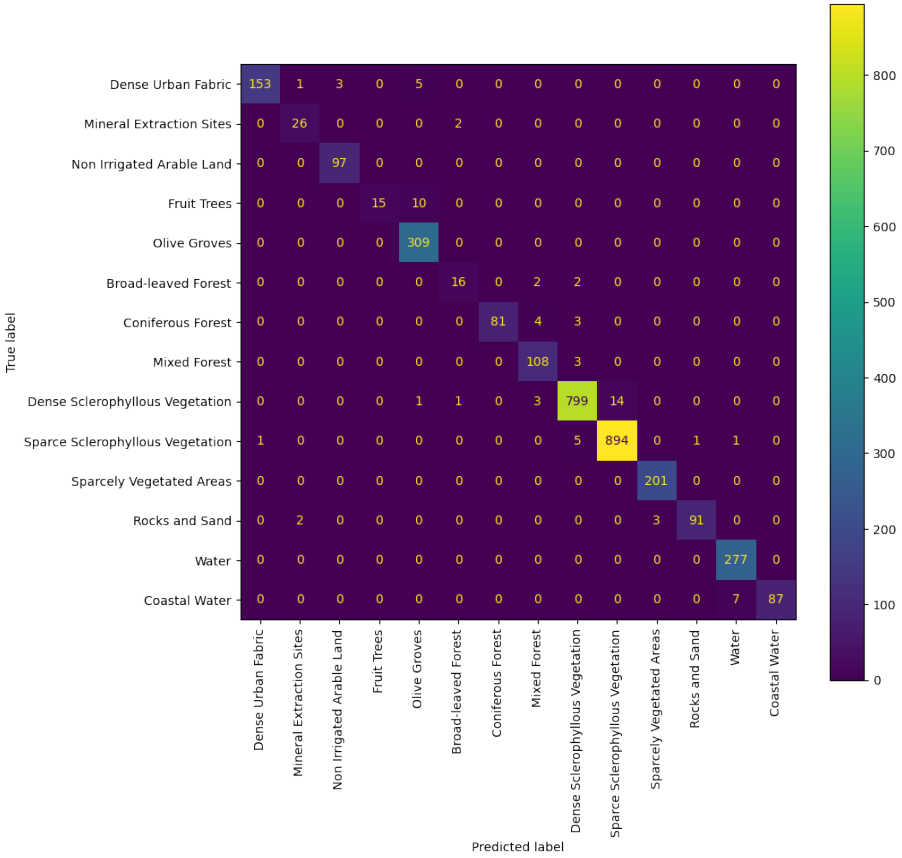

Figure 12: CNN confusion matrix

window size) so that same pixels do not end up on two different training patch images. These approaches are left as potential future improvements on this project and may be explored at a later date.

	precision	recall	f1-score	support
Dense Urban Fabric	0.99	0.94	0.97	162
Mineral Extraction Sites	0.90	0.93	0.91	28
Non Irrigated Arable Land	0.97	1.00	0.98	97
Fruit Trees	1.00	0.60	0.75	25
Olive Groves	0.95	1.00	0.97	309
Broad-leaved Forest	0.84	0.80	0.82	20
Coniferous Forest	1.00	0.92	0.96	88
Mixed Forest	0.92	0.97	0.95	111
Dense Sclerophyllous Vegetation	0.98	0.98	0.98	818
Sparce Sclerophyllous Vegetation	0.98	0.99	0.99	902
Sparcely Vegetated Areas	0.99	1.00	0.99	201
Rocks and Sand	0.99	0.95	0.97	96
Water	0.97	1.00	0.99	277
Coastal Water	1.00	0.93	0.96	94
accuracy			0.98	3228
macro avg	0.96	0.93	0.94	3228
weighted avg	0.98	0.98	0.98	3228

Figure 13: CNN classification report

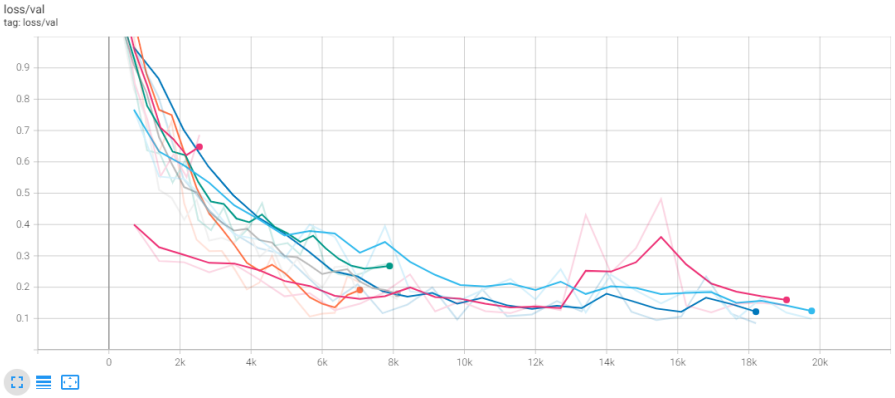

Figure 14: Validation Set Loss during Training

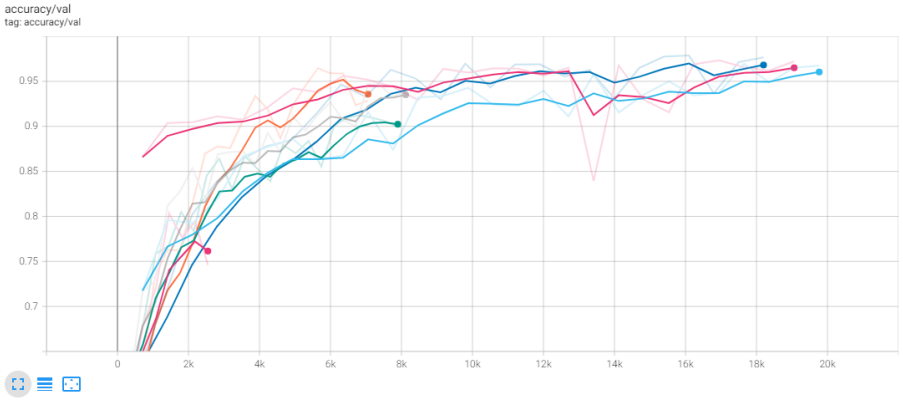

Figure 15: Validation Set Accuracy during Training

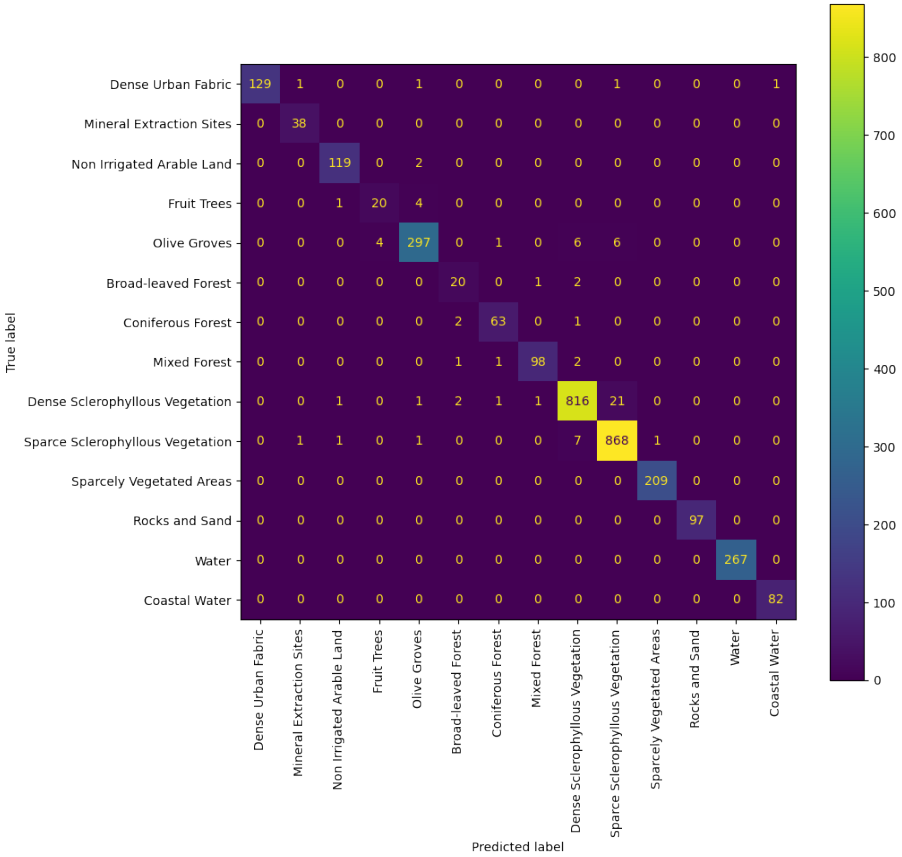

Figure 16: Transfer CNN confusion matrix

	precision	recall	f1-score	support
Dense Urban Fabric	1.00	0.97	0.98	133
Mineral Extraction Sites	0.95	1.00	0.97	38
Non Irrigated Arable Land	0.98	0.98	0.98	121
Fruit Trees	0.83	0.80	0.82	25
Olive Groves	0.97	0.95	0.96	314
Broad-leaved Forest	0.80	0.87	0.83	23
Coniferous Forest	0.95	0.95	0.95	66
Mixed Forest	0.98	0.96	0.97	102
Dense Sclerophyllous Vegetation	0.98	0.97	0.97	843
Sparce Sclerophyllous Vegetation	0.97	0.99	0.98	879
Sparcely Vegetated Areas	1.00	1.00	1.00	209
Rocks and Sand	1.00	1.00	1.00	97
Water	1.00	1.00	1.00	267
Coastal Water	0.99	1.00	0.99	82
accuracy			0.98	3199
macro avg	0.96	0.96	0.96	3199
weighted avg	0.98	0.98	0.98	3199

Figure 17: Transfer CNN classification report

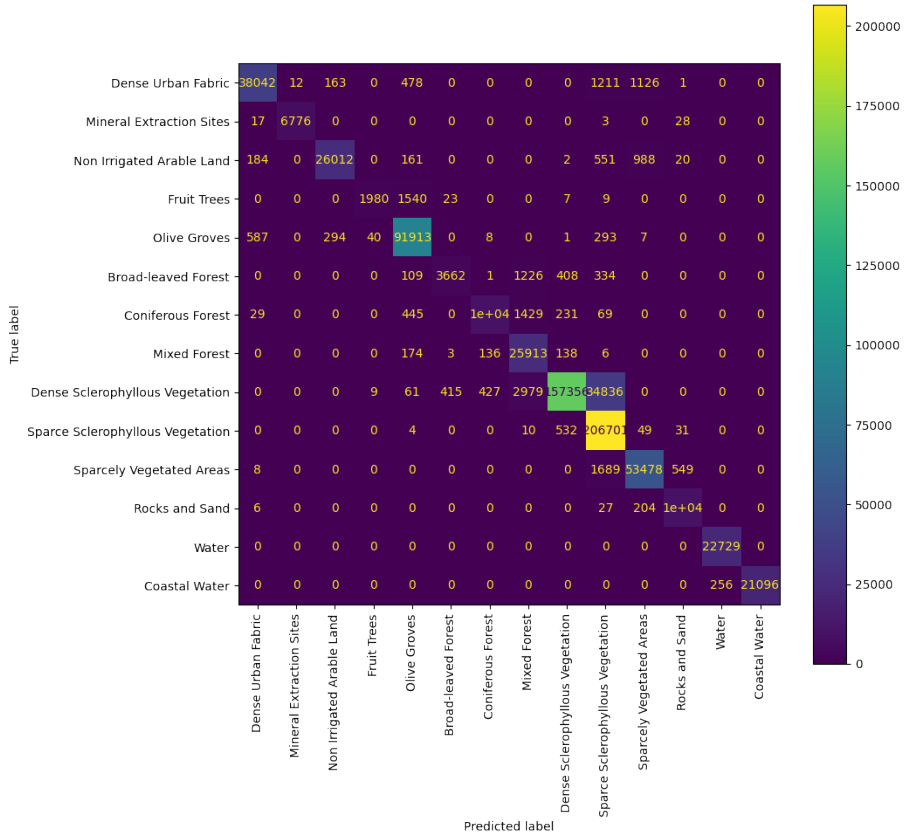

Figure 18: U-Net confusion matrix

	precision	recall	f1-score	support
Dense Urban Fabric	0.98	0.93	0.95	41033
Mineral Extraction Sites	1.00	0.99	1.00	6824
Non Irrigated Arable Land	0.98	0.93	0.96	27918
Fruit Trees	0.98	0.56	0.71	3559
Olive Groves	0.97	0.99	0.98	93143
Broad-leaved Forest	0.89	0.64	0.74	5740
Coniferous Forest	0.95	0.83	0.88	12638
Mixed Forest	0.82	0.98	0.89	26370
Dense Sclerophyllous Vegetation	0.99	0.80	0.89	196083
Sparce Sclerophyllous Vegetation	0.84	1.00	0.91	207327
Sparcely Vegetated Areas	0.96	0.96	0.96	55724
Rocks and Sand	0.94	0.98	0.96	10365
Water	0.99	1.00	0.99	22729
Coastal Water	1.00	0.99	0.99	21352
accuracy			0.93	730805
macro avg	0.95	0.90	0.92	730805
weighted avg	0.93	0.93	0.92	730805

Figure 19: U-Net classification report

Figure 20: Nefeli Validation Image in True Color Composite

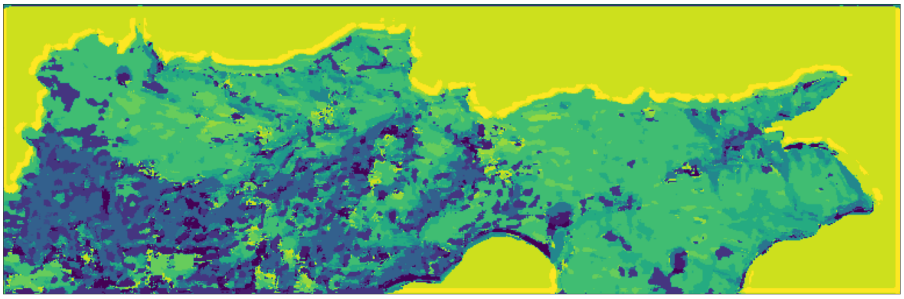

Figure 21: Nefeli Validation Image Predicted Labels

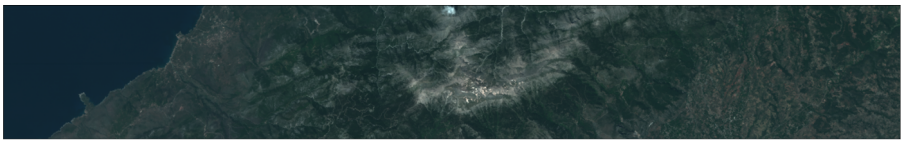

Figure 22: Erato Validation Image in True Color Composite

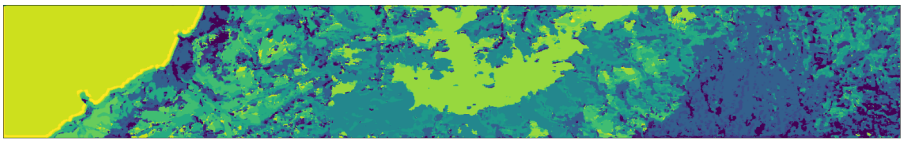

Figure 23: Erato Validation Image Predicted Labels

Figure 24: Kirki Validation Image in True Color Composite

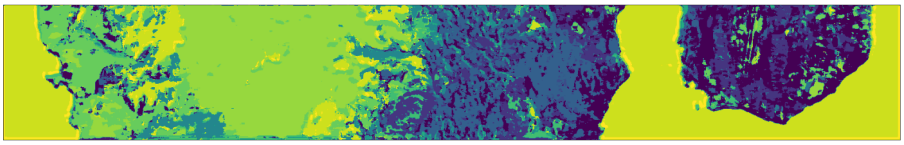

Figure 25: Kirki Validation Image Predicted Labels

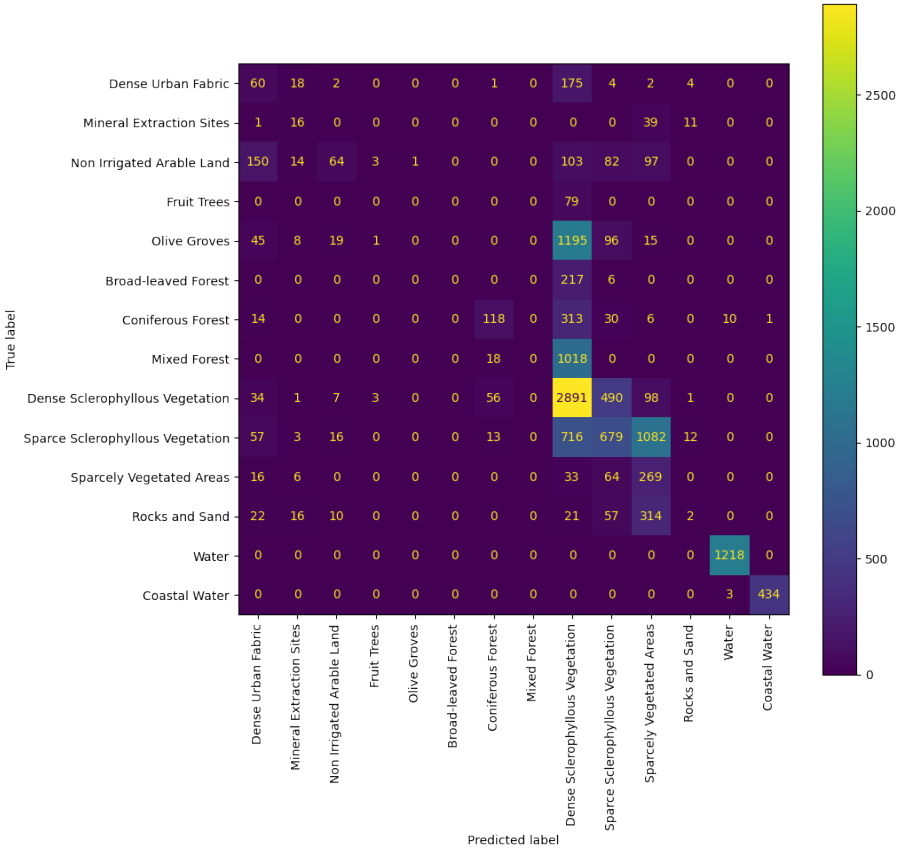

Figure 26: CNN trained on one image confusion matrix

	precision	recall	f1-score	support
Dense Urban Fabric	0.15	0.23	0.18	266
Mineral Extraction Sites	0.20	0.24	0.21	67
Non Irrigated Arable Land	0.54	0.12	0.20	514
Fruit Trees	0.00	0.00	0.00	79
Olive Groves	0.00	0.00	0.00	1379
Broad-leaved Forest	0.00	0.00	0.00	223
Coniferous Forest	0.57	0.24	0.34	492
Mixed Forest	0.00	0.00	0.00	1036
Dense Sclerophyllous Vegetation	0.43	0.81	0.56	3581
Sparce Sclerophyllous Vegetation	0.45	0.26	0.33	2578
Sparcely Vegetated Areas	0.14	0.69	0.23	388
Rocks and Sand	0.07	0.00	0.01	442
Water	0.99	1.00	0.99	1218
Coastal Water	1.00	0.99	1.00	437
accuracy			0.45	12700
macro avg	0.32	0.33	0.29	12700
weighted avg	0.40	0.45	0.39	12700

Figure 27: CNN trained on one image classification report