Planet Labels - How do we use our planet?

Timon Ruban
Stanford University
timon@stanford.edu

Rishabh Bhargava
Stanford University
rish93@stanford.edu

Vincent Sitzmann
Stanford University
sitzmann@stanford.edu

1 Introduction

Accurate and up-to-date land cover classification data can be used for interesting applications in different fields; by tracking daily changes of forest cover in areas of interest it is possible to quickly detect illegal deforestation. Environmental scientists can further explore the consequences of climate change by measuring how fast ice cover is melting. Urban planners can track city growth to better anticipate future infrastructure requirements. With the ever increasing number of satellites with image capturing capabilities, it has become progressively easier to observe and act upon the changes that earths surface is undergoing. We use high-resolution satellite images to classify the land visible according to its cover such as forest, water or urban land. The input to our algorithm is a tile of 128 by 128 pixels cut out from a satellite image. We then extract features like Gabor filters and histograms of the HSV, red edge and near-infrared bands from the tile and use a random forest to output a predicted land cover class.

2 Related Work

In recent years, several papers have discussed the use of machine learning approaches to the land cover classification problem. Support Vector Machines [7] have achieved 60-75% accuracy using Radial Basis Function (RBF) and polynomial kernels. Other approaches have seen the use of Decision Trees [11], Random Forests [5] and Artificial Neural Networks [3, 8] employing a number of different hyper-parameters with very similar results (the maximum accuracy being 90%). However, the datasets used in all these papers tended to be much smaller than the ones we are using. Some papers [4] have argued for evaluation metrics other than accuracy, such as κ error. In terms of choosing features, pixel values are commonly used, however, other important descriptors used are SIFT and Gabor filters [14].

3 Dataset and Features

3.1 Dataset

Since there is no dataset of geospatial images with corresponding land cover classification information readily available, we assembled our own. To this end we collected a dataset of 78,082 samples consisting of pairs of image tiles of 128 by 128 pixels (input object) and labels denoting the true land cover seen in that image (desired output).

We cut out the tiles from 367 RapidEye satellite images, all depicting different areas in California. The RapidEye images are 5000 by 5000 pixels at a 5m resolution and consist of five spectral bands (red, green, blue, red edge and near-infrared). We downloaded the imagery from the Planet Explorers program [9], that the earth imaging start-up Planet Labs has kindly given us access to.

To label the tiles we make use of the most recent National Land Cover Database (NLCD 2011) [6]. It covers the entire U.S. (at a spatial resolution of 30m) and uses a 16-class land cover classification scheme adapted from the Anderson Land Cover Classification System [1]. We further aggregated these 16 classes into 7 important land cover classes and excluded wetlands, since we have almost no data for this class (see Figure 1).

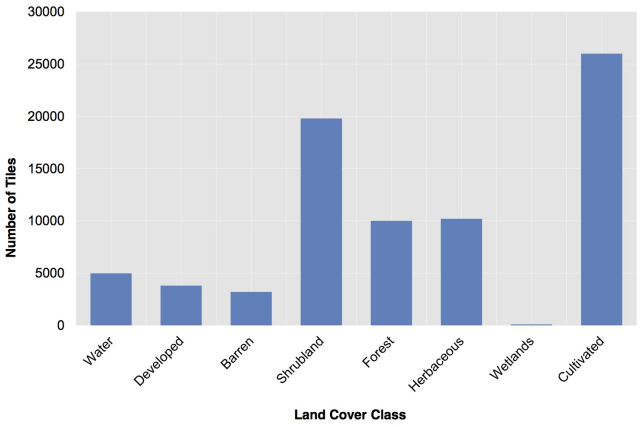


Figure 1: Histogram of the labels of the tiles in our dataset. Most images are of cultivated land. As the pictures are all from California, it is not surprising that almost no wetland is found.

To obtain the label for a tile, we transformed the coordinates of its pixels to the projected coordinate reference system of the NLCD 2011 and then queried the NLCD 2011 to get a label for every pixel in the tile. If more than 90 percent of the pixels belong to one land cover class, we admitted the tile and its respective land cover class to the dataset. We used this 90 percent threshold to remedy the problem that the RapidEye images are more recent (from 2015) than the NLCD 2011 dataset and the land cover might have changed since then. By only using tiles that are dominated by one type of land cover we are more confident that the land cover type has remained unchanged.

After collecting the data we split it into a test, validation and training set. 60 percent of the tiles were used for the training set, 20 percent for the validation set used to select the best model and another 20 percent for the test set used to report the final prediction accuracy of our method.

3.2 Features

Three major considerations dictated the feature selection. First and foremost, different land cover classes differ in their color. To capture color information, we transform each tile to the Hue-Saturation-Value color space and extract a 50-bin histogram per channel, yielding 150 features. The HSV-color space was chosen due to the robustness of the Hue-channel to lighting differences caused by clouds or different angles of the sun. The second important differentiating factor of different land cover classes is texture - for instance, water and forest generally have a much smoother texture than urban landscapes. Texture information was extracted by filtering each tile with 16 Gabor filters. Gabor filters are linear edge detection filters that have been proven to perform well for the purpose of texture extraction in satellite imagery [10]. After filtering, the mean and variance of each of the 16 tiles was extracted as an indicator of the occurrence of the respective edge type in the tile, yielding 32 features. Lastly, the RapidEye images also feature near infrared and red edge bands. Chlorophyll absorbs red edge light and reflects near infrared light [2]. Thus, another 100 features were dedicated to two 50-bins histograms of the red edge and near infrared bands of the RapidEye imagery.

4 Methods and Metrics

The methods and metrics used in this project are briefly explained in the sections below. Our input, a tile, is represented by $x^{(i)}$, and its corresponding label is given by $y^{(i)}$, where $i \in \{1, 2, ..., m\}$ (we have m examples). Here, $x^{(i)} \in \mathbb{R}^n$ and $y^{(i)} \in \mathbb{R}$. Let $\hat{y}^{(i)}$ represent the predicted class from our algorithms.

4.1 Methods

Logistic regression is a binary classification model ($y^{(i)} \in \{0, 1\}$) that assumes a hypothesis $h_\theta(x) = g(\theta^T x) = \frac{1}{1+e^{(-\theta^T x)}} \in [0, 1]$. The cost function to be minimised using gradient descent is given by the following, where the first term is the likelihood of the data, and the second is the regularisation term:

$$l(\theta) = \sum_{i=1}^{m} -(y^{(i)} h_\theta(x^{(i)}) + (1 - y^{(i)})(1 - h_\theta(x^{(i)})) + \frac{1}{C}(|\theta|_p) \quad (1)$$

One vs Rest Classifiers: Both models described above are binary classification techniques, however, we face a multi-class problem. This is addressed by training k models for a k-class problem where the i^{th} model is given by w_i, and $f(w_i, x)$ is a measure of how likely it is that an example belongs to the i^{th} class. Prediction is done in the following manner:

$$y = argmax_{i \in 1,..,k} f(w_i, x) \tag{2}$$

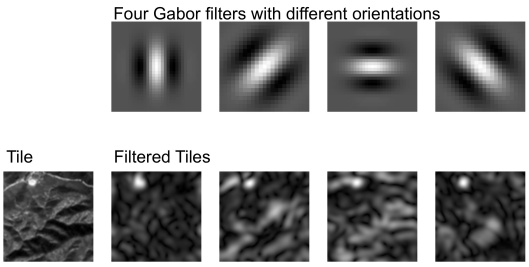


Figure 2: Example of a tile filtered with four Gabor filters of different orientations.

Random Forests is a supervised ensemble method based on decision trees. A decision tree divides the feature space into distinct, non-overlapping regions, that are then used to predict the class of an input sample. Growing a decision tree for classification involves making recursive binary splits, where every split tries to minimize the classification error rate of the tree until a specified depth or until every leaf is pure (fully grown tree). Random Forests build a number of decision trees from new datasets that were sampled with replacement from the original one (bootstrapping) and predict a class by taking the trees' majority vote. They also restrict the features considered in every split to a random subset of a certain size.

4.2 Metrics

Accuracy and Error are given as follows:

$$Accuracy = \frac{1}{m} \sum_{i=1}^{m} \hat{y}^{(i)} \neq y^{(i)} \tag{3}$$

$$Error = 1 - Accuracy \tag{4}$$

Precision and Recall: For the k^{th} class, precision and recall are given as

$$Precision = \frac{\sum_{i=1}^{m} 1[\hat{y}^{(i)} = k] \times [y^{(i)} = k]}{\sum_{i=1}^{m} 1[\hat{y}^{(i)} = k]} \tag{5}$$

$$Recall = \frac{\sum_{i=1}^{m} 1[\hat{y}^{(i)} = k] \times [y^{(i)} = k]}{\sum_{i=1}^{m} 1[y^{(i)} = k]} \tag{6}$$

5 Experiments, Results and Discussion

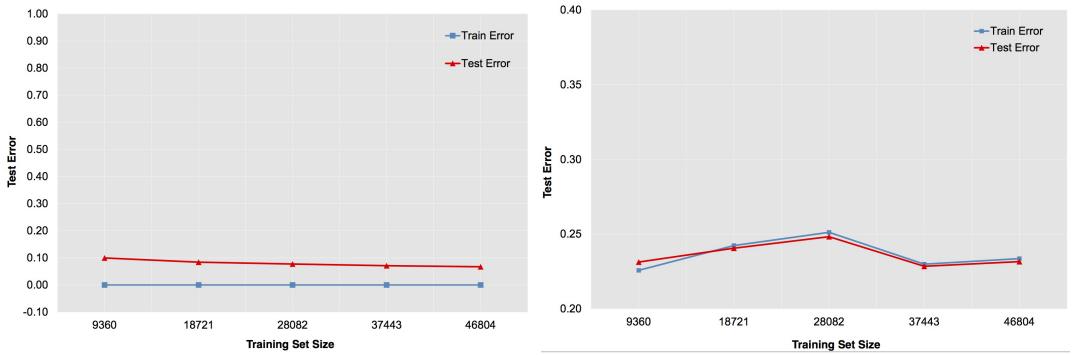
We ran our experiments with the implementation of methods done using `scikit-learn`[12] and other Python libraries. In most cases, all the features were used (colour and infrared spectra histograms and Gabor filters) unless otherwise stated.

5.1 Logistic Regression

We began with implementing Logistic Regression with L2 regularisation; this was our baseline model. We trained our models on 60% of the total data, which comprised $\sim$ 40000 images and 7 classes. When we trained with all the possible features, we got a training accuracy of 76.4%, and a validation accuracy of 76.8%. The learning curve associated with Logistic regression is shown in Figure 3b which demonstrates that this model has a high bias. We thus turned to a more flexible model, Random Forest (RF).

5.2 Random Forests

The first Random Forest (RF) model, using 500 trees and a random subset of 16 ($\approx \sqrt{282}$) features considered at every split, gives us an excellent validation accuracy of 92.5%. The learning curve is shown in Figure 3a, demonstrating that the training error is 0% with a validation error of 7.5%, implying that this is a high performing, high variance model. The confusion matrix obtained from



(a) Learning curve for Random Forest model

(b) Learning curve for Logistic Regression

Figure 3: Different measures for feature importance

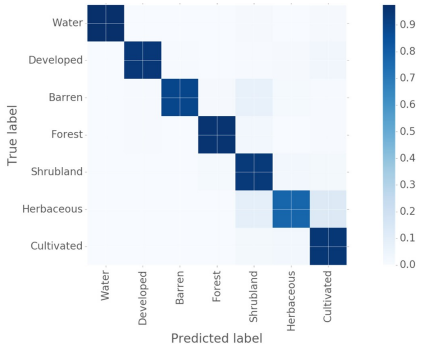


Figure 4: Confusion Matrix for Random Forest validation set.

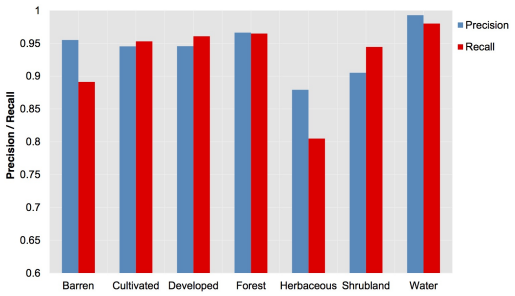
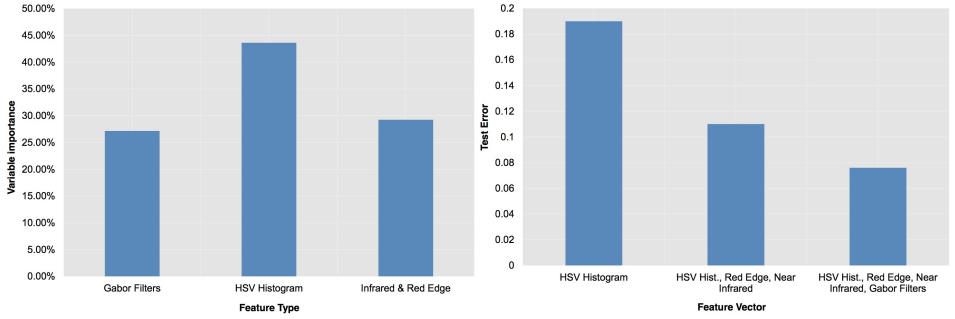


Figure 5: Precision Recall values for the different classes in the validation set.



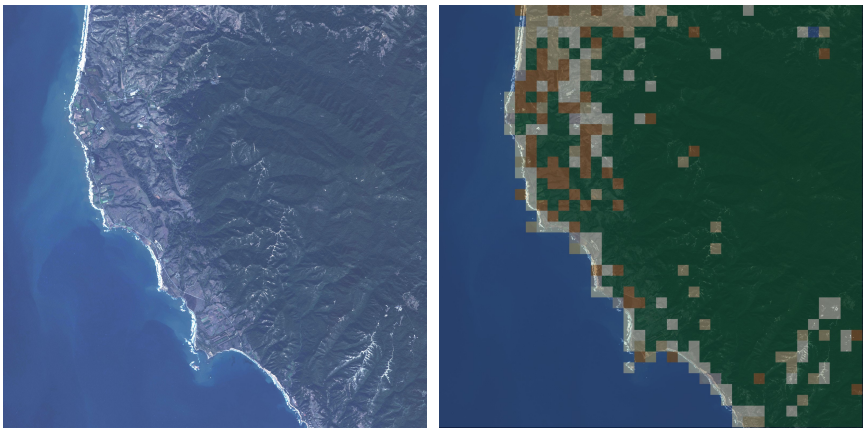
(a) Variable importance found from our RF model

(b) Drop in error as features are added sequentially in our RF model

Figure 6: Different measures for feature importance

this classification is shown in Figure 4 and a plot with the corresponding precision recall values for each class is given in Figure 5. We find that herbaceous areas (or grasslands) are most often misclassified. They are often confused with shrub or cultivated land - two very similar types of land cover, even to the human eye.

We further tested the importance of the three sets of features (colour histograms, infrared histograms and Gabor filters). The same RF model was trained on the same tile data, but the different feature sets were successively added. This gives us Figure 6b, demonstrating that all three sets of features contribute significantly to the overall accuracy. This is further backed up by the importance of different features extracted from the RF model with all three sets, which is shown in Figure 6a.



(a) An original satellite image from RapidEye

(b) The original image overlaid with our predicted class (using our RF model).

Figure 8: This can be done for satellite images across the world.

By looking at misclassified tiles, we find that many incorrect classifications involve a disparity between the NLCD dataset and the recently taken RapidEye images. An example is shown in Figure 7.

Figure 7: The image taken November 13th, 2015, depicts the Yolo Bypass located in the Sacramento Valley. The red box indicates a tile, that our algorithm classifies as water, but the NLCD dataset says is cultivated land. According to Wikipedia: "The entire bypass forms a valuable wetland habitat when flooded during the winter and spring rainy season. In the summer, areas of the bypass are used for agriculture. [13] This is a good example for a seasonal change that is correctly detected by our algorithm, but not by the NLCD dataset.

There are a number of hyperparameters to tune, and after computing validation errors for varying depth of trees, number of trees and number of features per split, we find that the best model has the following hyperparameters: 100 fully grown trees and 125 features per split. The final test error achieved is **6.9%**. This shows that we have trained a model that generalises well.

6 Conclusions and Future Work

In this project, we obtained high resolution satellite images and predicted their land cover. We explored a number of features related to colour, infrared reflectance and texture, and found all of them to be relevant. Random Forests outperformed other models, achieving an accuracy of 93.1% on the test set.

In the future, we would like to improve the granularity of our algorithm, by classifying images on a per pixel basis. This would help us achieve long term goals such as creating land cover databases for countries where this hasn't been done before. This would be achieved in a fashion similar to Figure 8. We also want to look at temporal differences between images, which would enable the identification of illegal deforestation and the tracking of urban land growth.

References

[1] James Richard Anderson. *A land use and land cover classification system for use with remote sensor data*, volume 964. US Government Printing Office, 1976.

[2] BlackBridge. The rapideye red edge band. [Online; accessed 09-December-2015].

[3] Daniel L Civco. Artificial neural networks for land-cover classification and mapping. *International Journal of Geographical Information Science*, 7(2):173–186, 1993.

[4] RS DeFries and Jonathan Cheung-Wai Chan. Multiple criteria for evaluating machine learning algorithms for land cover classification from satellite data. *Remote Sensing of Environment*, 74(3):503–515, 2000.

[5] Pall Oskar Gislason, Jon Atli Benediktsson, and Johannes R Sveinsson. Random forests for land cover classification. *Pattern Recognition Letters*, 27(4):294–300, 2006.

[6] Collin Homer, Jon Dewitz, Limin Yang, Suming Jin, Patrick Danielson, George Xian, John Coulston, Nathaniel Herold, James Wickham, and Kevin Megown. Completion of the 2011 national land cover database for the conterminous united states–representing a decade of land cover change information. *Photogrammetric Engineering & Remote Sensing*, 81(5):345–354, 2015.

[7] C Huang, LS Davis, and JRG Townshend. An assessment of support vector machines for land cover classification. *International Journal of remote sensing*, 23(4):725–749, 2002.

[8] T Kavzoglu and PM Mather. The use of backpropagating artificial neural networks in land cover classification. *International Journal of Remote Sensing*, 24(23):4907–4938, 2003.

[9] Planet Labs. Planet explorer program, 2015. URL: https://www.planet.com/explorers/.

[10] Shawn Newsam, Lei Wang, Sitaram Bhagavathy, and Bangalore S Manjunath. Using texture to analyze and manage large collections of remote sensed image and video data. *Applied optics*, 43(2):210–217, 2004.

[11] Mahesh Pal and Paul M Mather. An assessment of the effectiveness of decision tree methods for land cover classification. *Remote sensing of environment*, 86(4):554–565, 2003.

[12] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[13] Wikipedia. Yolo bypass — Wikipedia, the free encyclopedia, 2014. URL: https://en.wikipedia.org/wiki/Yolo_Bypass.

[14] Yi Yang and Shawn Newsam. Comparing sift descriptors and gabor texture features for classification of remote sensed imagery. In *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, pages 1852–1855. IEEE, 2008.