2022

Machine Learning Land Cover and Land Use Classification of 4-band Satellite Imagery

Lorelei Turner [*]
Air Force Institute of Technology

Torrey J. Wagner
Air Force Institute of Technology

Paul Auclair
Air Force Institute of Technology

Brent T. Langhals
Air Force Institute of Technology

Machine Learning Land Cover and Land Use Classification of 4-band Satellite Imagery

Lorelei Turner[1], Torrey Wagner[1,2,*], Paul Auclair[1,2], Brent Langhals[1]

[1] Data Analytics Certificate Program, Graduate School of Engineering and Management, Air Force Institute of Technology, 2950 Hobson Way, Wright-Patterson AFB, OH 45433, USA

[2] LinQuest Integrated Analytics & Support (Perduco), 2647 Commons Blvd, Beavercreek, OH 45431, USA

[*] Correspondence: torrey.wagner.ctr@afit.edu

Abstract — Land-cover and land-use classification generates categories of terrestrial features, such as water or trees, which can be used to track how land is used. This work applies classical, ensemble and neural network machine learning algorithms to a multispectral remote sensing dataset containing 405,000 28x28 pixel image patches in 4 electromagnetic frequency bands. For each algorithm, model metrics and prediction execution time were evaluated, resulting in two families of models; fast and precise. The prediction time for an 81,000-patch group of predictions was <1 s for the fast models, and >5s for the precise models, and there was not a significant change in prediction time when a GPU was used. Logistic regression was the best fast model, with >91% accuracy & f1 on the holdout dataset. While this model struggled with the trees and grassland classes, it provided inferences that the green wavelength band and the perimeter of the blue wavelength were important. All precise models had better performance than prior work, and the best precise model resulted from an 8-hyperparameter simultaneous neural network optimization. This model possessed >97% accuracy & f1 on the holdout dataset, and >97% accuracy & f1 on each of the 6 land categories. For both best-of-family models, there was little overfitting. Finally, the model was validated by predicting the land categories of a sample 159 MB multispectral image and visually verifying correct predictions. Using these classification models to automatically monitor land-cover and land-use classification is a promising approach for tracking changes over time and potentially reducing analyst workload.

Keywords — Machine learning, Remote sensing, Multispectral, Land-use classification, Land-cover classification, Neural networks, Random forest

I. Introduction & Background

Land-cover maps show regions of different kinds of land cover, such as water, vegetation, or dirt. Land-use maps additionally contain categories that show how people use the land, with categories such as crop land or urban areas. While these maps could be created by manual inspection of the regions, they are commonly created using multispectral or hyperspectral remote sensing data. Land cover and land use have numerous applications, especially when land-cover and land-use maps are collected over time and changes are examined. For civilian usage, the most important aspect of land-cover and land-use maps is monitoring the environment. Examples include monitoring deforestation, urbanization, and natural disasters such as flooding or wildfires. For military applications, knowing the terrain may be important knowledge for ground forces. Additionally, identifying change in land-use and land-cover maps may reveal new roads and facilities that have not been publicly announced.

Multispectral remote sensors are electro-optical sensors that collect images of the Earth's surface (from air or space) in at least four different spectral bands. Since material reflection depends on wavelength, multispectral images allow for discrimination between different materials when comparing spectral bands. Multispectral remote sensors have a long history; the first satellite with a multispectral imaging system was Landsat, which was launched in 1972 [1]. There are many traditional methods of identifying the land-cover category of each pixel. One of these is band ratios. The most well-known band ratio is the Normalized Difference Vegetation Index (NDVI), which compares the red and near infrared bands, and is a measure of the health and density of vegetation [1]. Another traditional method involves comparing the spectra of each pixel with reference spectra, which can be done by minimizing the least squares fit or by using vector algebra to calculate the spectral angle of the pixel and the reference spectra [2]. A more modern approach is the use of various machine learning algorithms to classify the land cover and land use of different regions. Many different algorithms have been tested, including random forests (RF), support vector machines (SVM), and artificial neural networks (ANN). A summary of recent machine learning land-cover and land-use research is presented in Table 1.

Table 1. Recent machine learning research in land cover and land use classification

Description of work	Method used	Approach	Performance	Ref
Land cover classification with deep learning, using texture as well as spectral information (2021)	ANN	Pixel	Accuracy: 86%	[3]
Evaluation of performance of several algorithms for land cover and land use classification (2021)	SVM ANN	Pixel	Accuracy: 95% Accuracy: 90%	[4]
Several machine learning algorithms applied to land use and land cover classification (2020)	RF SVM ANN	Pixel	AUC: 0.91 AUC: 0.87 AUC: 0.89	[5]
Land use/land cover classification focusing on temporary inundated areas (2020)	RF SVM ANN	Pixel	Accuracy: 94% Accuracy: 97% Accuracy: 96%	[6]
Land cover classification of perennial crops (2019)	SVM	Segment	Accuracy: 91%	[7]

In addition to numerous techniques and algorithms that can be used to solve the land-cover and land-use classification problem, there are also several different ways to approach dividing an image into land-cover and land-use categories. As mentioned before, the traditional method was the pixel-based approach, where each pixel is classified independently. Another approach is to segment the image into similar regions, then classify the regions. A third approach is patch-based classification, in which a remote sensing image is divided into patches, and each patch is classified independently.

The research focus explored in this paper is related to patch-based classification, to determine how well classical and neural network machine learning algorithms can predict land-cover classifications of the multispectral image patches in the SAT-6 dataset. The performance of the model will be measured using the macro averaged f1 score. The two criteria for success are to exceed the accuracy & f1 metric of a trivial model that predicts the majority class, and also exceeds the 91.6% average accuracy of the prior work shown in Table 1.

Data Acquisition

The process used to acquire the data is described in [8]. The first step was to extract images from the 65 terabyte National Agriculture Imagery Program

(NAIP) dataset. The NAIP dataset contains airborne images covering the entire continental United States. These images measure 6000x7000 pixels each and represent a ground sample distance of 1 meter in four spectral bands: red, green, blue, and near infrared (NIR). To develop the SAT-6 dataset, 1500 image tiles were sampled from a variety of scenes covering the state of California. Image patches from a land-cover class were manually labeled as barren land, trees, grassland, roads, buildings, and water bodies. Then, nonoverlapping, 28x28 pixel image patches were selected from the labeled image patches, making sure that there were no interclass patches selected. A total of 405,000 image patches are in the dataset. For the first image in the dataset, the 28x28 pixel patches from each wavelength band are shown in Fig. 1. The RGB wavelength bands from the first 25 images in the dataset are shown in **Error! Reference source not found.**, along with their classification.

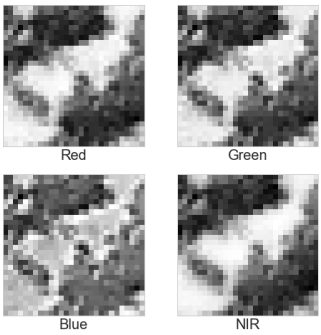

Fig. 1. First 28x28 pixel patch image in training set, showing the four wavelength bands. Acronym: Near infrared (NIR).

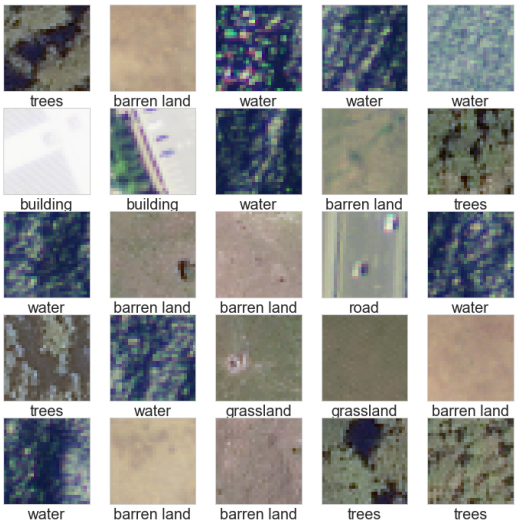

Fig. 2. First 25 images in training set, as RGB images.

Data Understanding

A summary of the training and holdout datasets is presented in Table 2. As discussed in the data acquisition section, these input variables are 28x28 pixel image patches, with 4 spectral bands, for a total of 3136 features. The labels represent the 6 land-cover class categories and are one-hot encoded as indicator variables.

Table 2. Variables Summary

Variables	Shape	Observations	Data Distribution
Train dataset	28x28x4	324000	Varies by class
Holdout dataset	28x28x4	81000	Varies by class
Train label	1x6	324000	Unbalanced
Test label	1x6	81000	Unbalanced

As shown in **Error! Reference source not found.**, the classes are unbalanced – the water class has more than one third of the observations, while the building and road classes each have less than 5%.

Table 3. Class statistics.

Class	Percent of Total	Mean of pixels	Standard Deviation

			of pixels
water	37.1	70.1	46.5
barren land	22.7	162.3	28.4
trees	17.5	103.2	40.8
grassland	15.5	127.0	36.2
building	4.6	174.1	55.4
road	2.5	135.7	47.3

The dynamic range for all pixels is 0 to 255, and the top subplot of Fig. 3, which shows a histogram of all pixels in the training data, separated out by spectral band. The distribution of pixels is very different for the NIR band than the other bands; while the red, green, and blue bands possess a quasi-normal distribution, the NIR band does not. The bottom part of Fig. 3 shows the same histogram separated by land-use class, and not all classes follow a normal distribution. For example, the road class is spiky and does not appear to follow a standard distribution. The NIR band is distinct from the other bands for every class, while the red, green, and blue bands often have significant overlap.

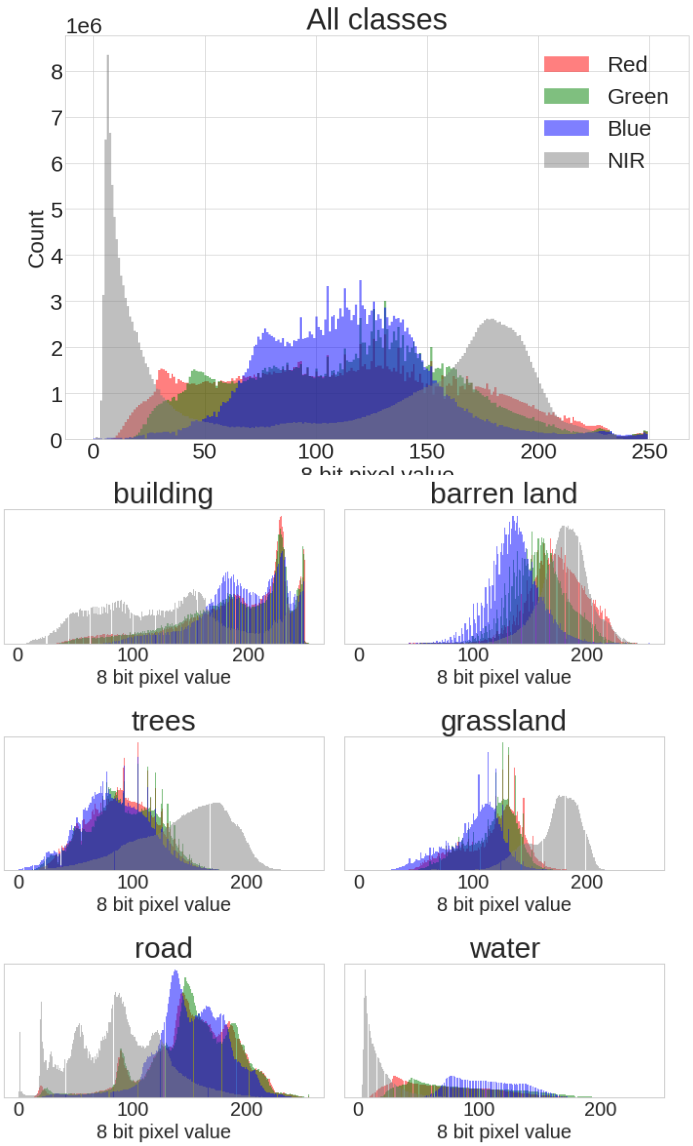

Fig. 3. Histograms of pixels per band.

II. Method

Before modeling, the data were prepared, comparative metrics were established, and several machine learning algorithms were modeled using the Scikit-learn 1.0.1 and Keras/TensorFlow 2.7 frameworks within a GPU-enabled Python 3.7.12 environment. The cross-industry standard process for data mining (CRISP-DM) was followed, with the phases of data understanding, data preparation, modeling, and evaluation [9]. Since this model may be applied to vast quantities of data, prediction time for each algorithm was also measured.

Data Preparation

The data did not contain missing values or noticeable outliers. The input variables have the same scale and units, so while scaling and normalization are not strictly necessary, some algorithms benefit from those data preparation steps. For example, feature scaling is required by support vector machines and can help gradient descent converge faster [10]. As a result, the data were scaled using scikit-learn's StandardScaler. As mentioned in the prior section, the data has some non-normal characteristics, but it was not possible to transform the data to make it more normal as those characteristics vary by class. In preparation for training the data were flattened from a 28x28x4 array to a 1x3136 array. The dataset was randomized then split into 80/20% train/test datasets to monitor overfitting. For neural networks the train data was further split into 80/20% train/validation.

Metrics

The primary metric used to evaluate the performance of both the classical and neural network models are f1, as the data labels are unbalanced, and the penalty for misclassifying a land area is even between classes. f1 gives a higher score to models that have similar (and higher) precision and recall, which is desired in this problem. While accuracy is not as appropriate since the labels are unbalanced, it is calculated to enable comparison to prior work. Since this is a multiclass problem, the f1 metric reported are calculated as the macro average, which weighs each class equally and rewards models that have better performance on the all classes, not just the larger classes.

Classical Modeling

The classical modeling implemented four different algorithms: logistic regression (LR), support vector machine (SVM), random forest (RF), and the ensemble

XGBClassifier algorithm. The performance for all classical algorithms is presented in Table 4 in Section III.

- Logistic regression is a linear algorithm used for classification, and was selected because it a simple algorithm and can be used as a baseline for the other algorithms [10] [11]. 6-class multinomial logistic regression was implemented within the sklearn *linear_model* framework, which natively supports multi-class classification. The adaptive Stochastic Average Gradient (SAGA) solver was used with a maximum of 100 iterations to minimize the cross-entropy loss associated with the multinomial probability distribution.
- SVMs are classifiers that can produce non-linear decision boundaries, however linear SVM was used due to the large dataset and because it has often been used for land use and land cover classification. The Python SGDClassifier SVM algorithm was used with a maximum of 100 iterations and a tolerance of 0.001.
- Random forests were also selected as they are another commonly used technique in land cover and land use classification and are capable of learning complex functions [11]. The Python RandomForestClassifier algorithm was used with 100 trees, gini criterion and the default settings for other parameters.
- The Python extreme gradient boosting algorithm XGBClassifier was selected to investigate the performance of an ensemble algorithm, which was not applied in the literature surveyed for this work [3] - [7]. A maximum depth of 6 was selected, and the algorithm exceeded the available 25 GB RAM if the full 324,000-patch dataset was used. For this algorithm, the training dataset size used the first 121,500 patches to fit within this memory constraint.

Neural Network Modeling

For the neural network modeling, a baseline model was created followed by a simultaneous hyperparameter optimization using the Adaptive Experimentation Platform (Ax) within Python. A train/validate/holdout split was used to monitor overfitting, since there are many datapoints available. The model is trained with the train set, the validation set is used for making decisions about hyperparameters, and the test set is used to evaluate the chosen models with the best performance. A stratified split is used to make sure minority classes are represented due to the class imbalance.

The Widrow recommendation for the number of neurons is related to the number of datapoints *P*, the number of weights (neurons * (inputs + 1)), and the desired error level according to Equation 1 below [12].

$$P = \frac{neurons*(inputs+1)}{error} \quad (1)$$

According to the Widrow recommendation, for the 8.4% error level goal of this work, 3136 inputs and 6 outputs, even a model with a total of 100 neurons would require a dataset of 22,000,000 patches. As the present dataset is only 405,000 patches, all models created in this work are overparameterized. The only limitation placed on the models is by the available RAM of 25 GB, where a neuron count greater that 280 causes a memory crash.

The baseline neural network model was chosen to have a sequential architecture, a single hidden layer with 100 ReLU neurons, and an output layer of 6 softmax neurons due to the multiclass classification problem. The optimization algorithm chosen was stochastic gradient descent (SGD) and sparse categorical cross-entropy loss function as neural network was shallow and this was a multiclass problem. No regularization technique was applied, and the model was fit with 100 epochs of training and a batch size of 2048 for speed.

The Ax framework simultaneous hyperparameter optimization used Bandit optimization for categorical hyperparameters (activation function and optimizer) and Bayesian optimization for numeric hyperparameters (all others). Based on the results of the baseline neural network model, the hyperparameter search space with Ax was defined as:

- Learning rate (0.001 – 0.5)
- Dropout rate (0.01– 0.5)
- L2 regularization rate (0.0001 – 0.2)
- Number of hidden layers (1-3)
- Neurons per layer (1 – 70)
- Batch size (64 – 2048)
- Hidden layer activation function (relu, sigmoid or tanh)
- Optimizer (adam or sgd)

The tuning methodology varies hyperparameters between optimization loops while monitoring performance on the training and validation datasets. Each optimization loop was trained with 100 epochs, and an adaptive learning rate was applied that halved the learning rate every 30 epochs. Convergence on a set of hyperparameters and optimizer was noted after 50 optimization loops and 5 hours of GPU-assisted training, and then the neural network was remodeled with the optimal hyperparameters and optimizer. The performance for all neural networks is presented in Table 4 in Section III.

III. Analysis & Results

In this section the performance of six machine learning algorithms is presented in Table 4. Performance metrics and prediction speed of each model variation are compared to determine the best model. The criterion for a successful model was accuracy & f1 $\geqslant$0.916, which is the average performance of the prior work surveyed. Successful models also required a greater f1 than the trivial majority-class and random models (f1 metric of 0.09 and 0.15, respectively) which are shown in grey text at the bottom of Table 4. All models except for logistic regression and support vector machine met these criteria.

Table 4. Model Results for the training and holdout datasets.

Model	Train Accuracy	Train f1	Holdout Accuracy	Holdout f1	Time to evaluate None/GPU (s)
Goal	--	--	0.916	0.916	--
Logistic Regression	0.933	0.919	0.928	0.910	0.9 / 0.7
Support Vector Machine	0.899	0.841	0.899	0.839	0.9 / 0.7
Random Forest	1.000	1.000	0.980	0.969	6.9 / 5.8
XGBClassifier* (max_depth=6)	0.942	0.997	0.983	0.973	7.2 / 6.0
Preliminary NN model	0.977	0.966	0.975	0.962	6.1 / 5.1
Ax HP sweep	0.991	0.987	0.982	0.973	5.9 / 5.8
Always predicts water	0.371	0.090	--	--	--
Predicts randomly	0.167	0.145	--	--	--

The f1 metric is calculated as the macro average of all classes. Red text indicates performance that did not meet study goals, and grey text designates the performance of the trivial models.

* trained on first 121,500 elements of train dataset due to 25 GB RAM limitation

In Table 4, the prediction time for the 81,000-patch training dataset was measured when using a regular processor and a GPU. The GPU did not have a significant impact on runtime. There were two families of models that resulted – *fast* models that were less accurate and required < 1 s to execute, and *precise* models that possessed increased accuracy but required > 5 s to execute. As each category may have potential applications based on user needs, a best model was selected from each category.

Fast Models

The best-performance fast model was the logistic regression model that used the adaptive SAGA solver and a maximum of 100 iterations. The accuracy met the study goals, however the f1 metric did not, primarily due to poor performance (78% accuracy) on the *grassland* class.

Model inferences from the logistic regression model were obtained by plotting the absolute value of the model coefficient for each pixel. When those coeffects are plotted for each band with a common z axis in Fig. 4, it was apparent that the green and blue bands were more important. There were numerous green-band pixels that were viewed as important, and an outline of important pixels were important for the blue band. The blue-band outline inference could be because water patches would be mostly uniform across the outline of the patch. Due to the large number of features and observations, feature selection, such as an input variables p-value technique, was not applied.

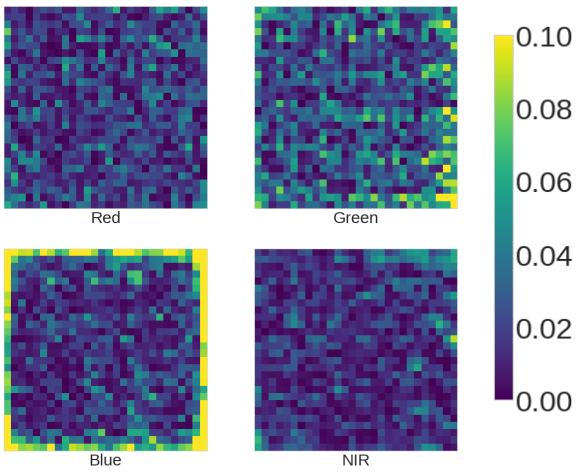

Fig. 4. Feature importance from logistic regression model.
For each band, the absolute value of the model coefficient for each pixel was plotted on a common z axis.

The multi-class confusion matrix for the logistic regression model is shown in Fig. 5, and which shows that the model had difficulty predicting the trees and grassland classes. The classification report confirmed this difficulty, and showed that all classes possessed >90% accuracy, except for trees (12,128/14,158 = 85%) and grassland (9,876/12,596 = 79%).

Fig. 5. 6-class confusion matrix for the logistic regression model.

Precise Models

For the precise models, the XGBClassifier and Ax hyperparameter sweep models had nearly-tied performance on the holdout dataset. The best performing fast model was judged to be the Ax neural network due to its faster prediction execution time. The optimal neural network hyperparameters and performance metrics are presented in Table 5.

Table 5. Hyperparameters and performance of the final NN model

Hyperparameter or metric	Value
Learning Rate	0.026
Dropout Rate	10.4%
# Hidden layers	1
Neurons/layer	56
L2 regularization	0.001
Batch size	501
Activation Function	tanh
Optimizer	SGD
f1(train)	0.987
f1 (holdout)	0.973

A neural network was trained on the optimal Ax hyperparameters, with the training extended to 200 epochs, instead of the 100 epochs that was used during the sweep. The training curves for this neural network are shown in Fig. 6, which show that past 100 epochs of training, the train accuracy increases slightly but the validation accuracy remained constant.

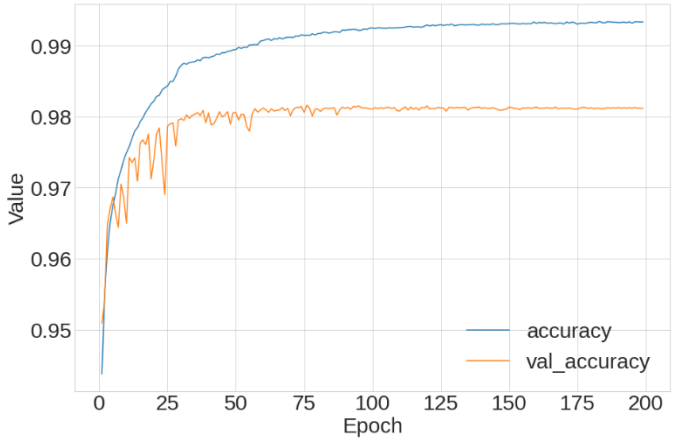

Fig. 6. Training and validation accuracy for neural network model after hyperparameter sweep.

The multi-class confusion matrix for the Ax neural network model is shown in Fig. 7, and the classification report showed a >97.6% accuracy for all classes.

Fig. 7. 6-class confusion matrix for the Ax neural network model.

Model overfitting is a common concern, especially as model complexity increases with ensemble and neural network methods. However, all models developed in this work possessed <3% overfitting, as measured by the f1 metric change between the training and holdout datasets in Table 4. All models except for lo-

gistic regression and support vector machine met the success criteria of this study, but the logistic regression model is still valuable due to its speed of execution and the ability to gather inferences.

Model Application

The neural network model was applied to classify regions of a sample NAIP image from California collected in 2016 [13], which is shown on the left side of **Error! Reference source not found.**. Each wavelength band of the 159 MB image was divided into 28x28x4 pixel patches and preprocessed as discussed in the Data Preparation section. The land category of each patch was predicted using the best model, and is shown on the right side of **Error! Reference source not found.**. The classes look appropriate; for example the lake in the upper left corner of each image can be clearly identified.

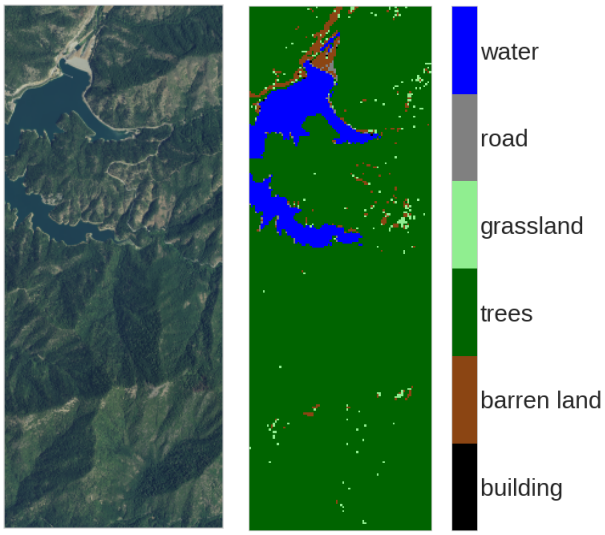

Fig. 8. . Model application showing original image (left) and model predictions where colors are mapped according to land use class (right).

The model weaknesses provided further insight into what kinds of land-use categories are harder to classify than others. It is useful to know which classes may be misidentified – for example in this work *grasslands* were harder to distinguish from *barren land* and *trees*. Additional inferences include the finding from the logistic regression model that the green band seemed to be the most important band, and that the outer edges of the blue band were the most influential.

Known limitations of the current models include the inability to correctly predict a class they have not been trained on, for example, snow or cropland. Also, since all the data was from a specific group of sensors at a specific resolution, data from other sensors will most likely give poor results. As a result, suggestions for future work include expanding the number of classes, testing the impact of mixed classes in an image patch, and investigating land use and land cover classification with other sensors. Also, a convolutional neural network architecture is commonly used for image recognition, and its performance could be explored.

IV. Conclusion

Land-use and land-cover multiclass classification on the 1.1 GB SAT-6 dataset of image patches was modeled with both classical, ensemble and neural network machine learning algorithms, using the CRISP-DM method. For each algorithm, model accuracy, f1 and prediction execution time were evaluated. This evaluation resulted in two families of models; fast and precise. Logistic regression was the best fast model, with a >91% accuracy & f1 on the holdout dataset, and < 1 s prediction time for a batch of 81,000 predictions. While the logistic regression model struggled with the trees and grassland classes, it provided inferences that the green wavelength band was important, and the perimeter of the blue wavelength band.

The best precise model resulted from an 8-hyperparameter simultaneous neural network optimization using the Bayesian and Bandit optimization within the Ax framework. This neural network had better performance than prior work, with >97% accuracy & f1 on the holdout dataset, and >97% accuracy & f1 on each of the 6 land categories. The prediction time for a 81,000-patch group of predictions was <6 s.

For both best-of-family models, there was little overfitting. A fast prediction time is important when considering remote sensing images with millions of pixels, and there was not a significant change in prediction time when a GPU was used. Finally, the model was validated by predicting the land categories of a sample 159 MB multispectral image and visually verifying correct predictions. The impact of applying the model to classify regions of multispectral remote sensing images could reduce analyst workload by tracking land-use and land-cover classes automatically.

References

[1] R. C. Olsen, Remote Sensing from Air and Space, 2nd ed., Bellingham, Washington: SPIE, 2016.

[2] H. Evans, J. Lange and J. Schmitz, The Phenomenology of Intelligence-focused Remote Sensing, New York: Riverside Research, 2014.

[3] T. He and S. Wang, "Multi-spectral Remote Sensing Land-Cover Classification Based on Deep Learning Methods," *Journal of Supercomputing,* vol. 77, no. 3, pp. 2829-2843, 2021.

[4] L. Ghayour, A. Neshat, S. Paryani, H. Shahabi, A. Shirzadi, W. Chen, N. Al-Ansari, M. Geertsema, M. P. Amiri, M. Gholamnia, J. Dou and A. Ahmad, "Performance Evaluation of Sentinel-2 and Landsat 8 OLI Data for Land Cover/Use Classification Using a Comparision between Machine Learning Algorithms," *Remote Sensing,* vol. 13, no. 7, p. 1349, 2021.

[5] S. Talukar, P. Singha, S. Mahato, P. S. Shahfahad, Y.-A. Liou and A. Rahman, "Land-Use Land-Cover Classification by Machine learning Classifiers for Satellite Observations - A Review," *Remote Sensing,* vol. 12, no. 7, p. 1135, 2020.

[6] B. van Leeuwen, Z. Tobak and F. Kovács, "Machine Learning Techniques for Land Use/Land Cover Classification of Medium Resolution Optical Satellite Imagery Focusing on Temporary Inundated Areas," *Journal of Environmental Geography,* Vols. 1-2, no. 13, pp. 43-52, 2020.

[7] J. Brinkhoff, J. Vardanega and A. Robson, "Land Cover Classification of Nine Perennial Crops Using Sentinel-1 and -2 Data," *Remote Sensing,* vol. 12, no. 1, p. 96, 2019.

[8] S. Basu, S. Ganguly, S. Mukhopadhyay, R. DiBiano, M. Karki and R. Nemani, "DeepSat - A Learning framework for Satellite Imagery," *ACM SIGSPATIAL,* 2015.

[9] IBM Corporation, "IBM SPSS Modeler CRISP-DM Guide," IBM Corporation, 2011.

[10] A. Géron, Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow, 2nd ed., Sebastopol, CA: O'Reilly Media, 2019.

[11] G. James, D. Witten, T. Hastie and R. Tibshirani, An Introduction to Statistical Learning: with Applications in R, 1st ed., New York: Springer, 2013.

[12] B. Widrow, "ADALINE and MADALINE," in *1st International Conference on Neural Networks*, San Diego, CA, 1987.

[13] California Natural Resources Agency, "NAIP 2016," 25 April 2017. [Online]. Available: http://gisarchive.cnra.ca.gov/iso/ImageryBaseMapsLandCover/NAIP/naip2016/. [Accessed 25 August 2021].