Integrating OpenStreetMap Crowdsourced Data for Land Cover Classification

Gaurav Tople, Mayur Vyas, Siddhant Ware

Abstract

We explored the potential for rapid land use/land cover (LULC) classification using OpenStreetMap (OSM) and Normalized Difference Vegetation Index (NDVI) Landsat time-series data. The main challenge with using these data for LULC classification was their high level of noise, as the Landsat images all contained varying degrees of cloud cover (causes of attribute noise) and the OSM polygons contained locational errors and class labeling errors (causes of class noise). A second challenge arose from the imbalanced class distribution in the extracted training data, which occurred due to wide discrepancies in the area coverage of each OSM-LU/OSM-N class. To address the first challenge, we used a relatively noise tolerant algorithm called Feedforward Artificial Neural Network for LULC image classification. To address the second challenge, we used weight balancing technique which balanced our data by altering the weight that each training example carries when computing the loss. Image classification accuracies were calculated for a four-class, five-class, and six-class LULC system to assess the capability of the proposed methods for mapping. The highest overall accuracy obtained was 76.33%, 71% and 65.33% for four-class, five-class and six-class system respectively by using the weight balancing technique.

1. Introduction

The objective of our study is to evaluate the potential for using freely available normalized difference vegetation index (NDVI) values extracted from satellite images in combination with OpenStreetMap (OSM) derived training data for land use/land cover (LULC) mapping. Although noise is often removed or corrected in remote sensing studies prior to LULC classification, these noise removal/correction procedures can be computationally- and/or labor-intensive. To simplify the LULC mapping process, we planned to explore the possibility of classifying the noisy data (i.e. without first removing/correcting the noise) using neural network and deep learning approach.

The dataset used for the project is downloaded from UCI Machine Learning Repository. The source of this dataset is unrestricted and publicly available for download. This dataset can be downloaded without any restrictions using the link given below:
http://archive.ics.uci.edu/ml/datasets/Crowdsourced+Mapping.

The dataset was derived from geospatial data from two sources: 1) Landsat

time-series satellite imagery from the years 2014-2015, and 2) crowdsourced georeferenced polygons with land cover labels obtained from OpenStreetMap. The crowdsourced polygons cover only a small part of the image area and are used to extract training data from the image for classifying the rest of the image. The main challenge with the dataset is that both the imagery and the crowdsourced data contain noise (due to cloud cover in the images and inaccurate labeling/digitizing of polygons).

Normalized Difference Vegetation Index(NDVI) is a simple graphical indicator that can be used to analyze remote sensing measurements, often from a space platform, assessing whether or not the target being observed contains live green vegetation. Negative values of NDVI indicate the presence of a waterbody. Similarly, values close to zero (-0.1 to 0.1) generally correspond to barren areas of rock, sand, or snow. Lastly, low, positive values represent shrub and grassland (approximately 0.2 to 0.4), while high values indicate temperate and tropical rainforests. When the data is compared with the OSM database for corresponding land cover, over 70% misclassified data was found.

In addition to attribute noise, class labeling errors in the training data (i.e. pixels with wrong class assignments), i.e. "class noise" (Frenay & Verleysen, 2013), can also have an impact on classification accuracy. Unfortunately, high quality training data for LULC classification can be time-consuming, difficult, and/or expensive to obtain, particularly if ground surveys are needed to collect the data. LULC is rapidly changing in many countries due to urbanization, so up to date LULC maps are needed in these areas for effective land use management and planning. However, there is not always sufficient time or funding available to gather high quality, up-to-date training data.

Hence, we aimed to study the behavior of deep neural networks when the labelling accuracy is low. With our findings, we aimed to help guide future studies in which large amount of data is easily available but is highly mislabeled and/or may be biased towards correct distribution

2. Background

In terms of past studies, the class noise-tolerance of four broad types of classification algorithms - probabilistic, decision tree (DT), instance-based, and support vector machines (SVM) algorithms – and found probabilistic methods, specifically the Naïve Bayes (NB) algorithm (John & Langley, 1995), best dealt with class noise, while DT and instance-based algorithms showed moderate performance, and SVM performed the worst due to its sensitivity to mislabeled training data located along the support vectors (Nettleton, et al., 2010). Another study, which considered the effects of class imbalance (i.e. discrepancy in the number of training samples per class) in addition to class noise and attribute noise, found that the Random Forest (RF) algorithm, an ensemble DT learner, achieved the highest classification accuracies

A previous study assessed OpenStreetMap (OSM) and NDVI Landsat

time-series data for LULC classification. The training data was extracted from OSM "land use" and "natural" polygon datasets. Noise-tolerant classification algorithms namely Naïve Bayes (NB), Decision Tree (C4.5 algorithm), and Random Forest (RF) were evaluated for dealing with data noise. Due to class imbalance synthetic minority oversampling technique SMOTE function was used to mitigate effects of class imbalance in the training dataset. Highest overall classification accuracy was obtained by random forest algorithm on using SMOTE function (Johnson & Iizuka, 2016).

3. Approach

3.1 Study area and data

The dataset used for this study is obtained from an area surrounding the Laguna de Bay of Philippines. This study area is approximately 125 km × 90 km, and includes the city of Manila as well as the surrounding suburban, agricultural, and forest areas. This area typically has at least some cloud cover, and it is often entirely covered by clouds during rainy season. It is also undergoing rapid LULC change due to urbanization (Tongson & Faraon, 2012), so training data collected in the past may be outdated. These factors make it a good study site for evaluating the tolerance of classification methods using neural networks and deep learning approach to attribute and class noise.

The NDVI values extracted from satellite images in combination with OSM derived land cover labels, were used to derive the dataset for this study. The dataset used was divided beforehand into training data (10545 exemplars) and testing data (300 exemplars) and were stored into two separate files namely 'training.csv' and 'testing.csv' respectively. The training data consisted of the data required to train the model for classification. Whereas, the testing data consisted of the data to evaluate the classification accuracy. The two files were then imported separately into Python Integrated development environment (IDE) in the form of DataFrames using pandas: A powerful Python data analysis toolkit (McKinney, 2012).

3.2 Forming a six-class, five-class and four-class LULC system

The dataset currently has six LULC classes: "impervious", "farm", "forest", "grass", "orchard", and "water", as shown in Table 2. Since the "forest" and "orchard" classes both contained tree cover (orchards consisted of mainly coconut, mango, and banana trees), and it is often difficult to distinguish between forests and orchards even when high quality training data is used (Oetter, et al., 2001), we also tested a five-class LULC system in which "forest" and "orchard" classes were merged into a more general "tree" class. Finally, an even broader four-class LULC system was tested in which the "farm" and "grass" classes were merged into an "other vegetation" class. Aside from common types of errors in the OSM polygon datasets (e.g. inaccurate boundary delineations of OSM-LU/OSM-N features), another challenge in using the data for LULC mapping came from the fact that the

OSM polygon could potentially contain multiple LULC types (e.g. an "impervious" OSM-LU polygon could potentially contain trees or grass in addition to impervious surfaces), so this type of mismatching can be considered as an additional source of class noise.

3.3 Data preprocessing

Once the datasets for six-class, five-class and four-class LULC systems are created, the next step is to separate the independent variables (or features) and dependent variables (or target label for classification) into two different NumPy arrays "X_train" and "y_train" for training data and "X_test" and "y_test" for testing data respectively, using NumPy toolkit in Python. 'max_ndvi' column was excluded from the set of input features as this column was highly correlated with some of the other input columns and hence was of least significance. No missing data were found in the feature sets of training and test data on performing check for missing values. As strings are a more ambiguous representation than integers in a program that relies on categorical data, label encoding was performed on classification labels by tuning string labels into integers with a 1:1 category to integer ratio (e.g. ['farm','forest','farm'] became [0,1,0] after label encoding). Hence, a six-class LULC system with classes: "farm", "forest", "grass", "impervious", "orchard", and "water" was converted into integer labels: 0, 1, 2, 3, 4, 5 respectively. A five-class and four-class LULC systems were label encoded in a similar way (Pedregosa, 2011).

In the next step we applied OneHotEncoder on the label encoded class which created 6 binary valued columns for 6 class categorical features (Pedregosa, 2011). Similar steps were performed for a five-class and four-class systems. The data in the training set was then standardized so that the distribution for each standardized feature was zero-mean and unit-variance. The scalers generated from the abovementioned procedure was then applied to the testing set (Pedregosa, 2011).

3.4 Applying Deep Learning for LULC classification

Deep learning model using Feedforward Artificial Neural Networks (ANN) was developed using keras API on TensorFlow backend to perform land cover classification. The model consisted of 3 layers in total with 1 input layer, 2 hidden layers and 1 output layer. The input layer consisted of 27 neurons (27 input features as "max_ndvi" feature was excluded). Each of the hidden layer consisted of 20 neurons each with an activation function of 'softplus' used in each neuron of the hidden layer. 'uniform' *kernel_initializer* was used and a 'GaussianDropout' regularizer was added after each hidden layer to prevent the model from overfitting the training data. On applying 'GaussianDropout' each input activation x is independently set to: x <- x * y, where y ~ N(1, stdev = sqrt((1-rate)/rate)). The dropout rate in this case is set to 0.3 for each dropout layer added.

The output layer of the ANN model consisted of 6, 5 and 4 neurons for a six-class, five-class and four-class system respectively. As this is a multiclass classification problem, we used 'softmax' as an activation function for each neuron at the output layer. Moreover, we used 'adam' as an optimizer with a learning rate of 0.0008 and a loss function of 'categorical_crossentropy' for the multiclass LULC classification. The above built model was then fitted on training data with number of epochs set to 200. The batch_size was set to 16 for a six-class LULC system. For five-class and four-class system, it was set to 32.

Classification was initially done using the original imbalanced training data. However, the model failed to predict the minority class as the number of exemplars available in the training dataset were significantly lower. (e.g. the orchard class in the six class LULC system only has about 53 exemplars out of 10545 exemplars). Hence, such minority classes are very difficult to predict using the original data. One of the class balancing techniques used in the previous research work was the synthetic minority oversampling technique (SMOTE). However, in our work we used another class balancing technique called 'weight balancing' which balanced our data by altering the weight that each training example carries when computing the loss. Normally, each example and class in our loss function will carry equal weight. But sometimes we might want certain classes or certain training examples to hold more weight if they are important. In our case, we allowed the minority classes to hold more weight than the majority classes. This extra parameter of class_weight was passed to the model while fitting on the training data. This model was then used to predict the LULC classes for test data and the overall accuracy was calculated to evaluate the model performance.

4. Results and Discussion

The model performance was evaluated using K-fold cross-validation technique by setting the value of k=10. On a six-class LULC system, the model achieved a mean accuracy of 0.8614 with a standard deviation of 0.18 on training data. Considering the label noise in the dataset and class imbalance in the training data, the model performed fairly well under noisy conditions.

The overall accuracy (OA) of 59.67% was achieved on a six class LULC system without using any class balancing technique on training data. Similarly, an overall accuracy of 68.67% and 75.67% was achieved for a five-class and four-class LULC system under the same conditions. As seen in the figure 1, we can determine that the model could not classify the minority class well for a six class LULC system. Similar behavior can be observed in Figure 3 and Figure 5 as well.

On using weight balancing technique on training data, the model accuracy fairly improved on test data and as seen in the confusion matrix in figure 2, figure 4 and figure 6, the model was able to classify the minority class fairly well. The highest overall accuracy of 65.33% was achieved on a six-class LULC system by using weight

balancing technique on imbalanced training dataset. Similarly, an overall accuracy of 71% and 76.33% was achieved on five-class and four-class system respectively using the similar technique.

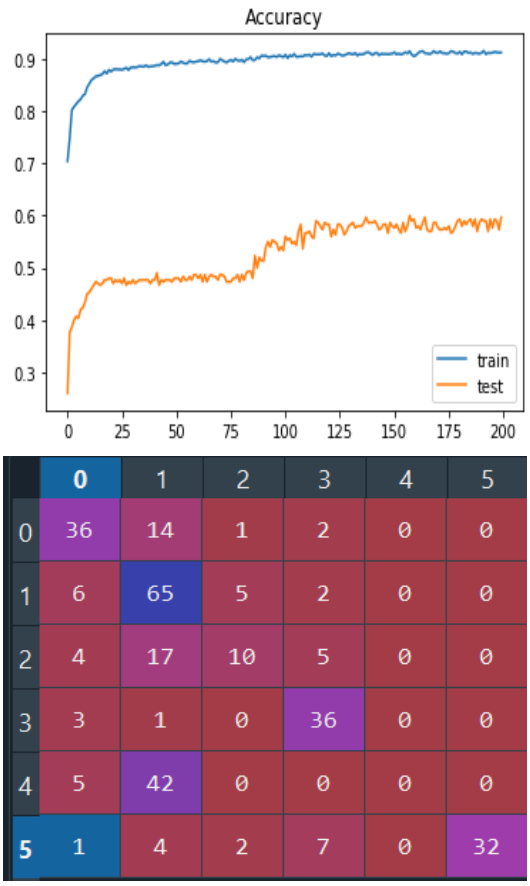


Figure 1: train vs test accuracy for six-class LULC system without using weight balancing technique (See top image) and confusion matrix for the same (See bottom image). OA = 59.67%.

Classes	Labels	Exemplars	Class Weight
0	Farm	1421	1
1	Forest	7431	1
2	Grass	446	1
3	Impervious	969	1
4	Orchard	53	5
5	Water	205	1

Table 1: Integer mapping of six-classes with class weights.

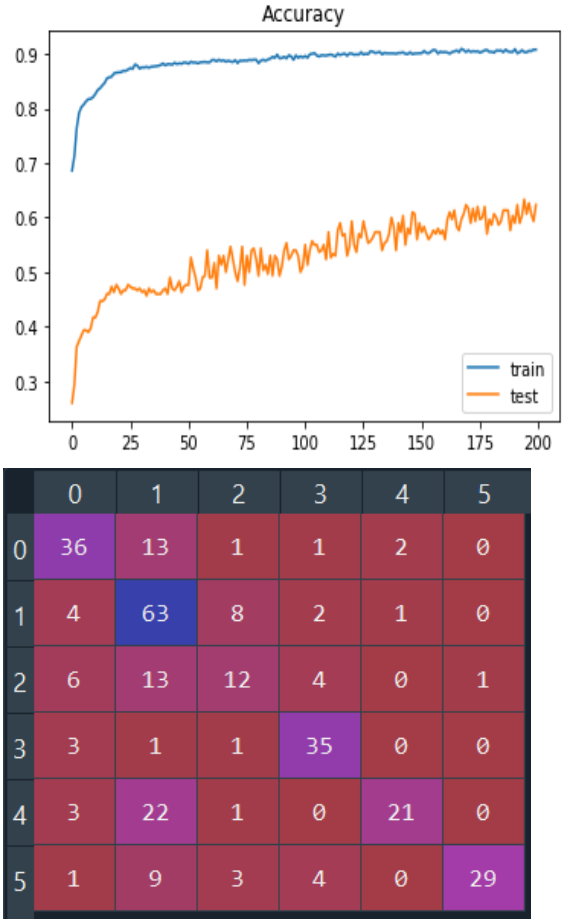
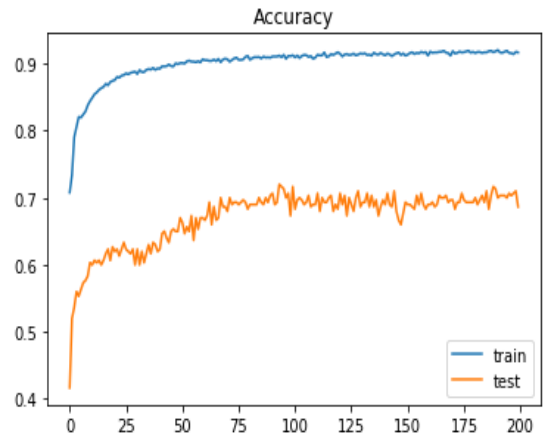


Figure 2: train vs test accuracy for six-class LULC system using weight balancing technique (See top image) and confusion matrix for the same (See bottom image). OA = 65.33%.

Classes	Labels	Exemplars	Class Weight
0	Farm	1421	1
1	Grass	446	5
2	Impervious	969	1
3	Tree	7484	1
4	Water	205	2

Table 2: Integer mapping of five-classes with class weights.

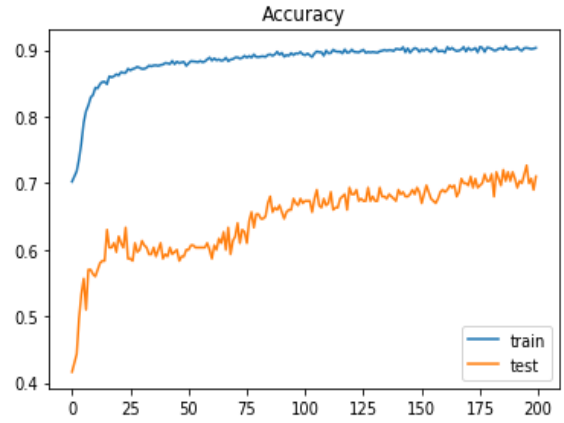


	0	1	2	3	4
0	35	1	2	15	0
1	8	9	3	16	0
2	4	0	35	1	0
3	8	11	1	105	0
4	1	2	10	11	22

Figure 3: train vs test accuracy for five-class LULC system without using weight balancing technique (See top image) and confusion matrix for the same (See bottom image). OA = 68.67%.

Classes	Labels	Exemplars	Class Weight
0	impervious	969	2
1	Other Veg.	1867	3
2	Tree	7484	2
3	Water	205	3

Table 3: Integer mapping of four-classes with class weights.



	0	1	2	3	4
0	33	2	2	16	0
1	2	19	2	13	0
2	2	9	28	0	1
3	6	19	1	99	0
4	1	4	5	2	34

Figure 4: train vs test accuracy for five-class LULC system using weight balancing technique (See top image) and confusion matrix for the same (See bottom image). OA = 71%.

In terms of the classification accuracies that could be achieved for the different LULC classification systems, none of the methods evaluated in this study were very successful for LULC mapping under the six- and five-class systems due to the high level of confusion between the "orchard" and "forest" classes in the six-class system and the confusion between the "grass" and "farm" classes in the five-class system. However, the four-class system achieved moderate accuracy by using weight balancing technique on ANN.

Based on our results, the classification methods presented in this study

are recommended mainly for mapping relatively broad LULC classes similar to the ones in our four-class LULC classification system. Of course, the usefulness of the LULC data for some applications (e.g. forest or crop monitoring) is limited when only a few broad LULC classes are mapped, but this can potentially be overcome if an older but more detailed LULC map is available. For example, a current tree cover map can be used for monitoring forest change over time if a historical map of forest boundaries exists

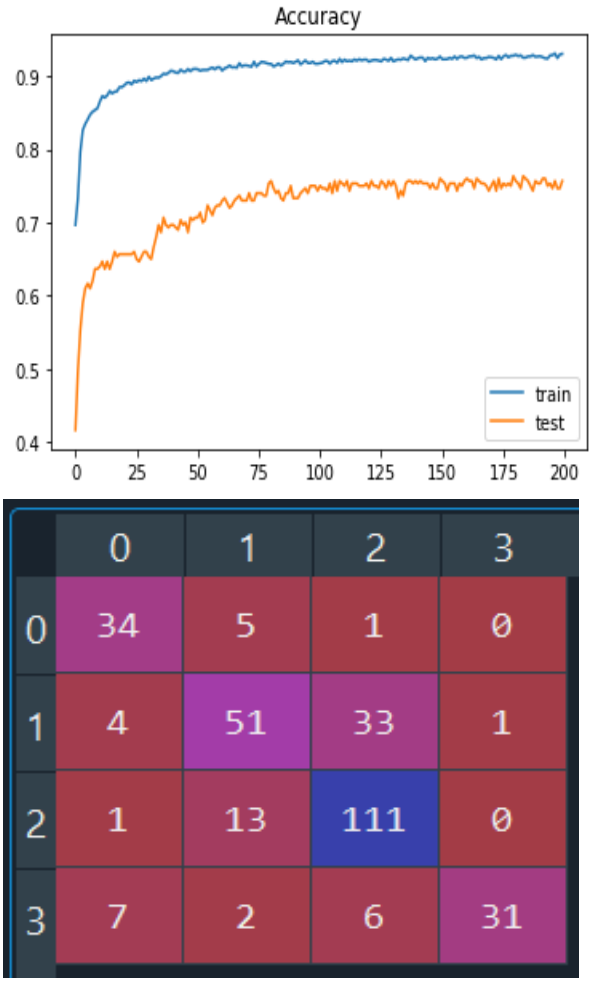


Figure 5: train vs test accuracy for four-class LULC system without using weight balancing technique (See top image) and confusion matrix for the same (See bottom image). OA = 75.67%.

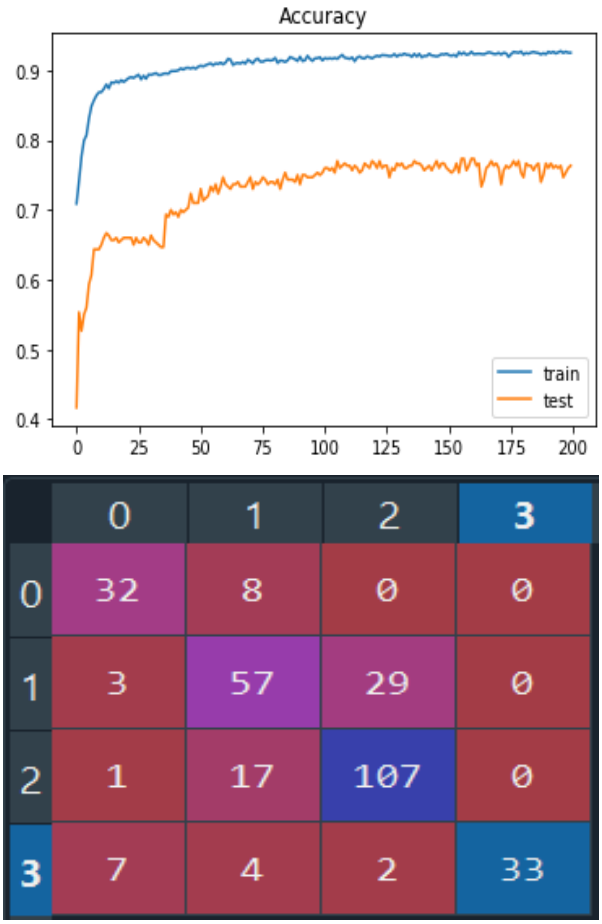


Figure 6: train vs test accuracy for four-class LULC system using weight balancing technique (See top image) and confusion matrix for the same (See bottom image). OA = 76.33%.

LULC system	ANN accuracy	ANN accuracy (w/ class weight)	RNN accuracy
6 class	59.67%	65.33%	61%
5 class	68.67%	71%	72.33%
4 class	75.67%	76.33%	77%

Table 4: Overall Accuracies for LULC classes

Neural learning from imbalanced trained data can result in totally ignoring the minority classes. Our research showed that a neural network trained from imbalanced data using the ANN algorithm as well as RNN algorithm can be biased toward the majority

class, when training data is noisy. Another problem associated with imbalanced training data is the rate of convergence. Since the majority class contains far more samples than the minority class, the rate of convergence of the neural network output error is very low. This is because the gradient vector computed by the ANN and RNN algorithm for an imbalanced training set responds slowly to the error generated by the training examples of the minority class. To resolve this issue weight balancing technique can prove to be a promising solution if optimal class weight values are set and passed through the model. Applying this technique on many-to-one LSTM architecture using RNN would be an interesting direction for future research.

5. Conclusion

In the scope of the project, we evaluated the potential for rapid and automated land use/land cover (LULC) map production using Landsat time-series NDVI imagery and training data derived from OpenStreetMap (OSM) "landuse" and "natural" polygons. The Landsat data contained attribute noise due to cloud cover in all images, while the OSM data contained class noise due to inaccurate polygon boundary delineations and because the OSM classes did not exactly match our the LULC classes they were used to map. Extracting training data from the OSM data also led to a class imbalance in the training set since some OSM classes had much greater area coverage than others.

The performance of tested the performance of our classification system under noisy conditions. In a series of experiments, we tested the performance with respect to the number of classes and then assigned weights to respective classes to balance the weights of the classes. During the course of our research, we tried to perform the classification using imbalanced data. We could not achieve the desired accuracy on minority class labels using this method. On the contrary after using weight balancing technique, we got better accuracy and could make the system learn better. We also observed that accuracy increased as a result of merging two classes into one.

Some future research avenues to build on this study include: (i) evaluating the performance of many-to-one LSTM architecture using RNN on applying weight balancing technique, (ii) incorporating other image datasets containing less cloud contamination, e.g. Sentinel-1 synthetic aperture radar data, (iii) evaluating automated filtering approaches to remove mislabeled training samples from the training dataset.

6. Acknowledgments

We wish to express our sincere thanks to Dr. Jerome J. Braun, course instructor, for his inspiring lectures which elevated our interest in Neural Networks and Deep Learning. We also place on record, our sense of gratitude to one and all, who directly or indirectly have lent their hand in this project.

References

Beigman, E. & Klebanov, B., 2009. *Learning with annotation noise.* s.l., s.n.

John, G. & Langley, P., 1995. *Estimating continuous distributions in Bayesian classifiers.* s.l., Morgan Kaufmann Publishers Inc.

Johnson, B. A. & Iizuka, K., 2016. Integrating OpenStreetMap crowdsourced data and Landsat time-series imagery for rapid land use/land cover (LULC) mapping. *Case study of the Laguna de Bay area of the Philippines, Applied Geography,* Volume 67, pp. 140-149.

McKinney, W., 2012. *pandas: powerful Python data analysis Release 0.7.3.* s.l.:pydata.org.

Nettleton, D., Orriols-Puig, A. & Fornells, A., 2010. A study of the effect of different types of noise on the precision of supervised learning techniques. *Artificial Intelligence Review,* 33(4), pp. 275-306.

Oetter, D. R. et al., 2001. Land cover mapping in an agricultural setting using multiseasonal Thematic Mapper data. *Remote Sensing of Environment 76(2),* pp. 139-155.

Pedregosa, e. a., 2011. Scikit-learn: Machine Learning in Python. *JMLR 12.*

Tongson, E. & Faraon, A., 2012. *Hydrologic atlas of Laguna de Bay,* Quezon City, Philippines: Laguna Lake Development Authority and WWF-Philippines.

Team Member Roles:

Siddhant worked on preprocessing of data (data cleaning, application of SMOTE function, etc.). Gaurav worked on building the deep learning model using preprocessed train data. Mayur validated the model built by using the test dataset and derived results. Similarly, the paper and presentation work was equally divided among group mate