GEOSPATIAL MACHINE LEARNING DATASETS STRUCTURING AND CLASSIFICATION TOOL: CASE STUDY FOR MAPPING LULC FROM RASAT SATELLITE IMAGES

Sohaib K. M. Abujayyab*, İsmail Rakıp Karaş

Dept. of Computer Engineering, Karabuk University, 78050 Karabuk, Turkey.
s.jayyab@hotmail.com and irkaras@gmail.com

KEY WORDS: Machine learning, neural networks, datasets structuring, LULC mapping, RASAT satellite images

ABSTRACT:

Remote sensing satellite images plays a significant role in mapping land use/land cover LULC. Machine learning ML provide robust functions for satellite image classification. The objective of this paper is to extend the capability of GIS specialists in geospatial area with minimum knowledge in computer science to easily perform ML satellite image classification. A framework consisting 7 stages established. Tools of steps developed in two programing environments, which are ArcGIS for geospatial datasets structuring and Anaconda for ML training and classification. During the development, authors constrained to reduce the complexity of big data of satellite images and limited memory of computers to make tools available for implementation in PC. In addition, automation and improving the performance accuracy. TensorFlow-Keras library employed to perform the classification using neural networks. A case study using RASAT satellite image in Ankara-Turkey utilized to perform the analysis. The developed classifier gained 80% performance accuracy. The complete RASAT satellite image processed and smoothly classified based on blocks methods. The developed tools successfully tested and applied in geospatial area and can be effectively execute in PC by GIS specialist.

1. INTRODUCTION

Observing land surface circumstance and changes are significant to evaluate current situation and transition in the environment (Reis 2008). Transition due to human activities, flooding disaster, drought, urban expansion, deforestation and global warming (Hamad, Balzter, and Kolo 2018; Paul et al. 2017). Such environmental problems often related to Land use/land cover (LULC) changes. Available relevant data to LULC changes can provide critical contribution to decision-making of environmental management and planning. In addition, LULC data helps to determine better land use policies, implement regional plans effectively, analyse environmental processes, and boosts living conditions (Ikiel et al. 2013).

Generally, LULC term is define through variable terms. Land cover indicate to physical material casing ground surface comprising water, soil, vegetation and artificial surfaces built by human habitats (Roy et al. 2015). In addition, land use indicates to how land is utilizing through people and their domestic activities such as agriculture activities (Haque and Basak 2017). The land could be allocated for agriculture activities but it is not consisting green cover. In this case, the land will be classifying as open area and not agriculture.

Geographical Information Systems (GIS), and remote sensing are powerful tools help to derive accurate and timely information about LULC changes over big zones (Pradhan and Lee 2010; Reis 2008). GIS provides a flexible software environment for gathering, storing, presenting and analysing digital data required for environmental management (Khamehchiyan, Nikoudel, and Boroumandi 2011). Remote sensing imagery plays a vital role as quantitative and qualitative data resource for GIS analysis. In addition, remote sensing imagery decreases difficulty of study time and field work.

Centres of satellite images receiving massive amount of data and size growing dramatically. Solid need for accurate method is existing to extract the valuable LULC data from enormous size of satellite images in efficient time. Recently studies have indicated that there has been a growing interest in developing methods and approaches for updating LULC database. (Abburu and Golla 2015). Machine learning (ML) provide some robust methods to efficiently classifying satellite images and extract LULC maps. Satellite images classification defined as representation of the land ordering or grouping into meaningful sets or groups based on their statistical relationships (Abburu and Golla 2015; Alqurashi and Kumar 2013). Various ML classification techniques were utilized to perform satellite images classification (Liao, Chu, and Hsiao 2012).

Classification techniques has been developed for a certain purpose, at a certain scale and utilizing different data type. Classification methods such as neural networks, random forest, support vector machine, regression, decision tree and so on (Li, Wang, and Li 2015). ML classification methods is new source, which is effective and very accurate, indirectly making them an attractive research area for LULC (Shekhar, Zhang, and Huang 2005).

Consequently, several studies have been reported. Studies involves utilization of ML classification methods to evaluate its performance for effective image classification and LULC database updating. Studies were trying to improve methods performance as well as facilitation and automation it utilization. In addition, Studies helps in dealing with big data (Abburu and Golla 2015). From the literature, mainly there are three

* Corresponding author

https://doi.org/10.5194/isprs-archives-XLII-4-W16-39-2019 | © Authors 2019. CC BY 4.0 License.

challenges exist related to satellite image and ML classification methods; [A], updating LULC at consistent time intervals in superior ground coverage, higher spatial and spectral resolution along with exponentially extra information content. Updating LULC from Satellite image using conventional methods for large areas relaying on several tools is often ineffective, infeasible, and time-consuming. This clearly represent the problem of big data complexity that arise in case the high need of computation time and memory to process the satellite data for classification, especially for neural networks NN. This issue became an obstacle for applying the analysis in normal personal computer, which limited the number of user and researcher.

[B], updating LULC containing automation difficulty. For each time GIS specialists essential to execute all ML classification operations such as pre-processing, training, classification and post-processing from the beginning, which means the end-user (GIS specialists) will face problems and some restrictions shall evolve during the execution of established framework. Limitations such as big data, reading spatial data, coordinates systems problems, structure training data sets from geospatial data, human errors and manual application undermine this option. The GIS specialist essential to gather the and apply the analysis from the start each time. The earlier models suffer from dispersion due to utilizing of numerous geospatial tools. They are also subjected to impractical models due to poor repeatability. There is no flexibility for different geographic areas with different data sensors. Lastly, there is limited integration with other software to implement the advanced analysis ML training, tuning and sensitivity analysis. In other words, the previous tools have limited functionality.

[C], updating LULC consisting performance complexity. Several SDM methods were suffering from low performance accuracy compared with low processing time. Therefore, there is a need to enhance performance of the LULC mapping from satellite images.

The aim of this article is to extend the capability of researcher in geospatial area and non-computer science specialists to easily perform ML satellite image classification. Those specialists facing difficulty to prepare the ML dataset from satellite images and perform classification analysis. In this paper, two platforms and sub tools were developed. The first in ArcGIS-Arctoolbox. The second in Anaconda using TensorFlow library. The paper focus to produce a friendly GUI tools for the users. In addition, the tools use blocks system to handle the massive volume of data in light of using personal computers PC with limited memory.

2. METHODS

Applying machine learning in geospatial research area is a complex issue. This software platform generated to help the end-users (GIS specialist without strong programing experience) to preparing and structuring the geospatial datasets for machine learning analysis. Thus, users can easily utilize these datasets in the mapping and updating LULC datasets from RASAT satellite images.

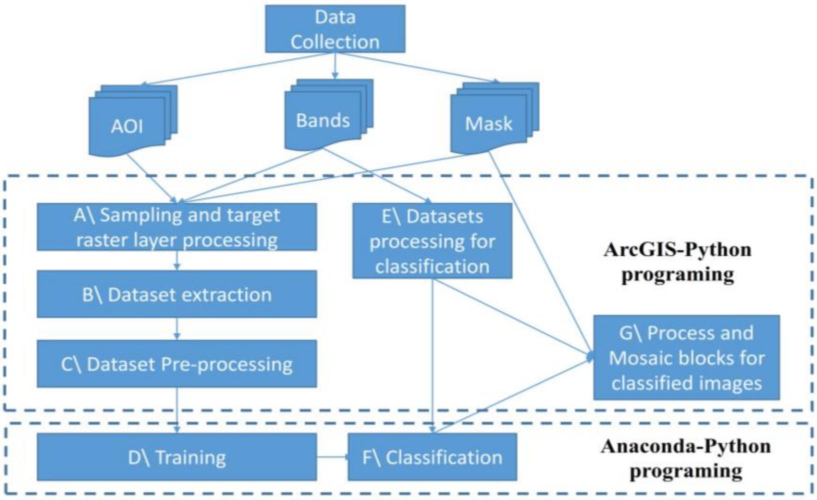

Data Collection
AOI
Bands
Mask
A\ Sampling and target raster layer processing
B\ Dataset extraction
C\ Dataset Pre-processing
E\ Datasets processing for classification
ArcGIS-Python programing
G\ Process and Mosaic blocks for classified images
D\ Training
F\ Classification
Anaconda-Python programing

Figure 1 processing workflow

According to programing environment, the development of software platform is divided to two sections. These two sections containing 7 steps from A to G (see **Figure 1**). All ArcGIS tools and Anaconda python scripts can be obtained from (https://github.com/SohaibAbujayyab. The first section is Python-ArcGIS environment. The final outcomes are a set of new GUI tools in ArcToolbox. This section consisting the steps A, B, C, E, and G. Second section is in Python-Anaconda environment. The final outcomes are a set of simple and readable string cods. The codes represent two steps fsrom processing workflow D and F.

In general, for both of the programing environment, the authors developed the tools considering several aspects for geospatial datasets structuring. The number of sub tools were reduced to minimum number in light of avoiding any complexity coming from data size and inputs parameters. Several number of input parameters were automatically extract to simplify the tools and reduce number of input parameters.The usage of memory in every tool released directly after each process. To extend the audience of the tool, all the codes designed to work with any kind of computer machine specification (High and Low computers specification). In addition, chunk computing concept applied to deal with big data complexity of RASAT satellite images.

This contribution has been peer-reviewed.
https://doi.org/10.5194/isprs-archives-XLII-4-W16-39-2019 | © Authors 2019. CC BY 4.0 License.

The output datasets from the tools can be in Numpy or TXT formats. The TXT format were developed in order to provide human-readable text files, which is make chance for the user to read and understand the datasets structures. While TXT output process made optional, the Numpy output format is the default format. Numpy format is binary-readable format, but it is faster in the processing and the user can decide which alternative is suitable. The next sections explained in details the sub tools and their processing content.

2.1 ArcGIS-Python programing for datasets structuring

This section consisting the steps A, B, C, E, and G. Step (A) is for sampling and target raster layer processing. In this tool, area of interest AOI file need to be prepare. AOI is similar to signature file in ArcGIS. AOI can be made by using regular image classification toolbar, or as new Polygon file. Every group of polygon must represent one class in the satellite image. User need to be balanced between the number of Polygons in each class to avoid any bias during the modelling.

The overall tools support until 15 class. The user need to define the class of each Polygon in one field in the attribute table. This tool utilizes AOI Polygons to perform augmentation process for number of dataset samples. Augmentation for signature Polygons through converting each Polygon to grid of points. The user has carefully to defined cell size, which represent the distance between the points. Tiny distance can lead to massive number of training points, which could not be fit to the memory and sufficient number of samples for training. The first output from these processing will be a grid of points within the area of interest. The second output is the target raster layer. These two layers is the input of next step.

(B) The second step is dataset extraction. The inputs are the prepared points and target layer in the first stage. In addition, the user need to add the bands of the satellite images. The images must be well processed. The user need to handle the spatial reference issue and cell size. The tools developed to feed the target raster layer in different parameters to avoid the common problem in sorting the variables and the target. For each point from points grid file, the tool will extract all the overlay values from each band. The output will be grid of points with the extracted attributes. This structure of the attribute table is: fields represent the bands and rows represent the points. In addition, the target field will be added after the last field in the table.

(C) Dataset Pre-processing. The functionality of this stage is to store the structured attribute table as Numpy or TXT files. The output can be full dataset or separated band variables and target variable. This step employ “da.SearchCursor” function in Arcpy library to construct geospatial training dataset. The default output is Numpy files, which is fast in processing. But in case the user need to store a human-readable files, the user can check the Boolean box in the tools. TXT files help the GIS analysers to understand the structure of output datasets. The user need to select the input variables and target variable from list of fields. The list of fields appears automatically after adding the grid of points in the tool. Lastly, the output dataset migrated to ML environment for training and classification.

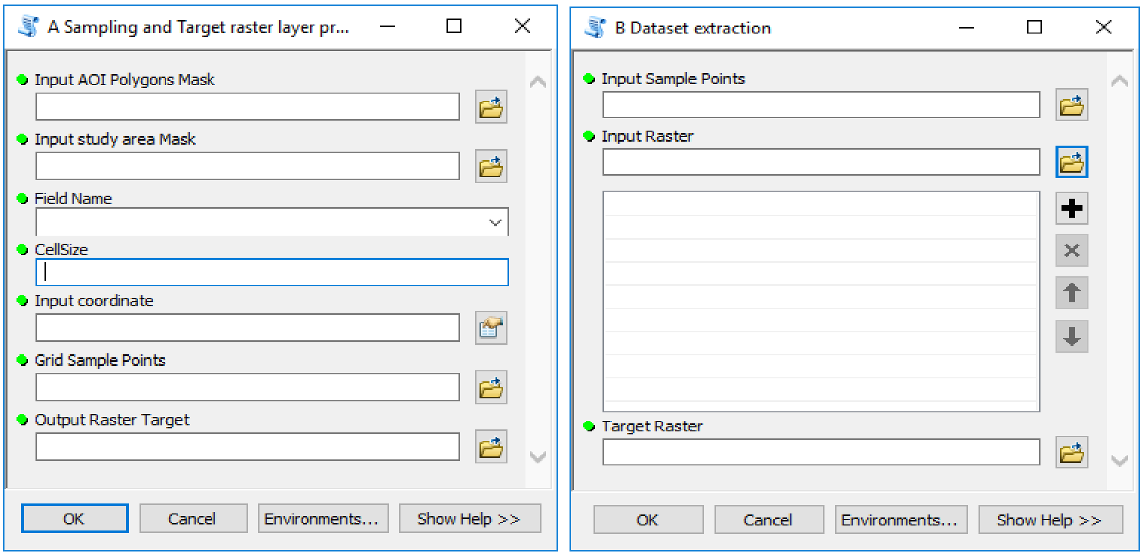
(E) Datasets processing for classification. The objective of this step is to avoid any crash in the systems during classification. Crash happen due to complex processing, big data and machines with limited internal memory. In order to perform classification for full satellite images after training, ML algorithms required to reshaping the images based their suitable structure. This structuring process is critical issue, especially in the wide geographic areas. The tool dividing full satellite images to blocks to avoid high usage of memory. Thus, block dataset fit into internal memory.

The user need to added the bands of the image identically with the same order in step B. In addition, size of the blocks need to be defined. The tool divides, read and store the blocks in suitable structure for ML. User need to define the spatial reference of the output datasets. The outcomes from this step are ML datasets of blocks as TXT files. Additionally, MetaData store for each block to define the block cell size, x,y coordinates, number of rows and columns.

(G) Process and Mosaic blocks for classified images. The objective of this stage to generate the final classified map. The tool used the classified TXT columns of each block to generate the final map. Throughout this step, the tool goes over the classified TXT columns and reshaping the TXT columns to Numby matrixes. The tool defined the shape and location of each block based on MetaData file. Then, the blocks of Numby matrixes Mosaic to full classified raster image. Finally, full classified image Masked based on area of the analysis.

Table 1 Example part of training dataset of sample points

Input (explanatory parameters)				**Target (LULC class)**
Band 1	**Band 2**	**Band 3**	**Band 4**	
38	59	55	65	GreenCover
59	98	84	88	OpenAreas
56	97	82	87	OpenAreas
38	58	55	66	GreenCover
33	47	48	63	Water
33	46	48	63	Water
57	98	85	88	OpenAreas
60	98	83	87	OpenAreas
63	113	84	91	Urban
67	120	89	89	Urban
32	47	49	63	Water



A Sampling and Target raster layer pr...

- ● Input AOI Polygons Mask
- ● Input study area Mask
- ● Field Name
- ● CellSize
- ● Input coordinate
- ● Grid Sample Points
- ● Output Raster Target

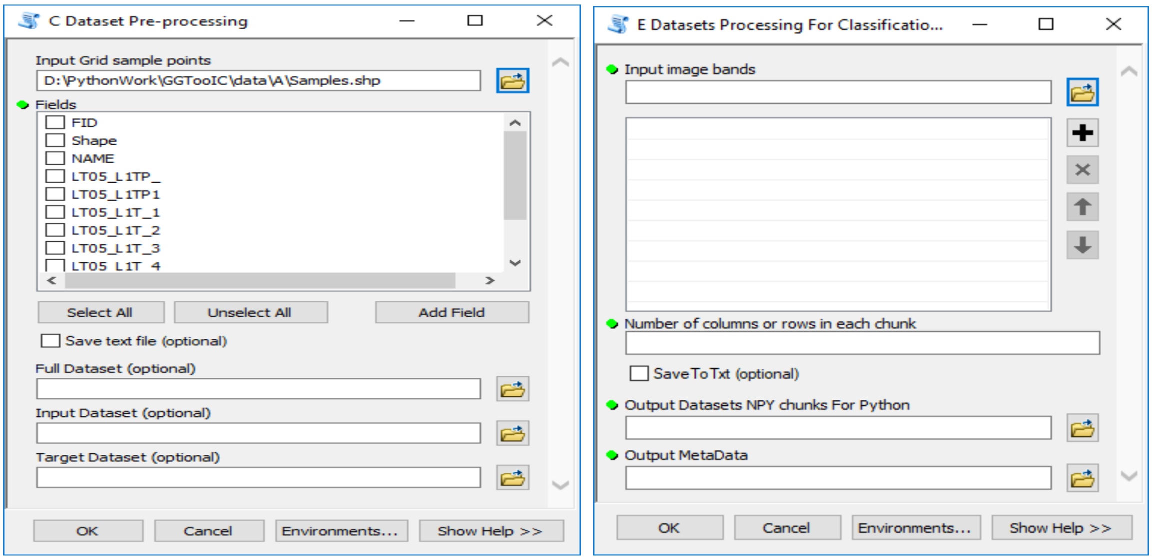
OK | Cancel | Environments... | Show Help >>

B Dataset extraction

- ● Input Sample Points
- ● Input Raster
- ● Target Raster

OK | Cancel | Environments... | Show Help >>

Figure 2 Tool Parameterization (A) Sampling and Target raster layer process. (B) Dataset extraction.



C Dataset Pre-processing

Input Grid sample points
D:\PythonWork\GGTooIC\data\A\Samples.shp

● Fields

- □ FID
- □ Shape
- □ NAME
- □ LT05_L1TP_
- □ LT05_L1TP1
- □ LT05_L1T_1
- □ LT05_L1T_2
- □ LT05_L1T_3
- □ LT05_L1T_4

Select All | Unselect All | Add Field

□ Save text file (optional)

Full Dataset (optional)

Input Dataset (optional)

Target Dataset (optional)

OK | Cancel | Environments... | Show Help >>

E Datasets Processing For Classificatio...

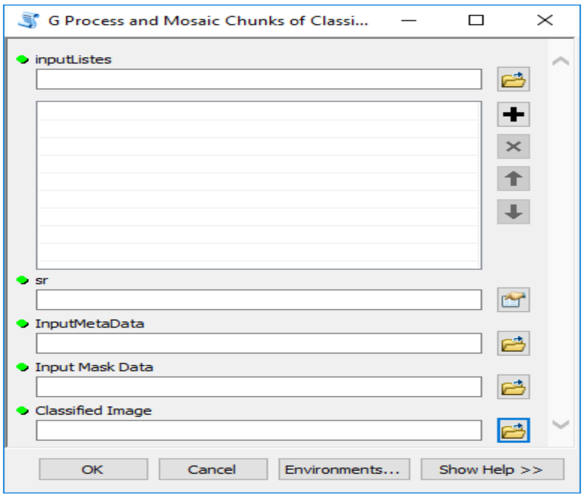
- ● Input image bands
- ● Number of columns or rows in each chunk

□ SaveToTxt (optional)

- ● Output Datasets NPY chunks For Python
- ● Output MetaData

OK | Cancel | Environments... | Show Help >>

Figure 3 C Tool Parameterization (Dataset Pre-processing) E (Datasets Processing For Classification based on chunks)



G Process and Mosaic Chunks of Classi...

- ● inputListes
- ● sr
- ● InputMetaData
- ● Input Mask Data
- ● Classified Image

OK | Cancel | Environments... | Show Help >>

Figure 4 G Tool Parameterization (Process and Mosaic Chunks of Classified Image)

2.2 Anaconda-Python programing for ML

This section mainly containing two stages, which are (D) training and (F) classification. The first step (D step the overall workflow) is training. This step considers as most critical stage among the steps. During this step, the user required to develop a classifier and store it for next steps. The classifier need to be evaluate and gain suitable or high performance accuracy. This stage read TXT dataset of geospatial data, then perform the training and generate the classifier.

Table 2 Step D, Python TF classifier programming and development in Anaconda environment

```
# Import TensorFlow
import tensorflow as tf
# Import helper libraries
import numpy as np
import seaborn as sns
import pandas as pd
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt

# Import dataset
X= np.loadtxt('E:\Ankaradata\C\Input5.txt')
y= np.loadtxt('E:\Ankaradata\C\Trget5.txt')

# Pre-process the data
x_train, x_test, y_train, y_test = train_test_split(X,y, test_size= 0.30, random_state=27)
x_train = tf.keras.utils.normalize(x_train, axis=1)  # scales data between 0 and 1
x_test = tf.keras.utils.normalize(x_test, axis=1)  # scales data between 0 and 1

# Build the classifier
model = tf.keras.models.Sequential()
model.add(tf.keras.layers.Dense(1000, tf.nn.tanh))
model.add(tf.keras.layers.Dense(100, activation=tf.nn.selu))
model.add(tf.keras.layers.Dense(5, activation=tf.nn.softmax))

# Compile the classifier
model.compile(optimizer='adam',
        loss='sparse_categorical_crossentropy',
        metrics=['accuracy'])
# Train the classifier
model.fit(x_train, y_train, epochs=5, batch_size=32,
          validation_data=(x_test, y_test))

# Evaluate accuracy
test_loss, test_acc = model.evaluate(x_test, y_test)
print('Test loss:', test_loss) # model loss (error)
print('Test accuracy:', test_acc)  # model accuracy

# Save the model
ModelPath = 'E:\Ankaradata\D\my_model5.h5'
model.save(ModelPath)
```

Earlier research tried to implement NN using stand-alone codes or software’s. the main criticism for these codes that were gaining humble performance. In addition, ML codes are very complex due to low level of programing, which is difficult to be understood or used by GIS user. Out of these codes, verel are commercially. Additional, it is neither scientific nor practical to ignore the most modern and open-source machine learning libraries. Furthermore, researchers tried to deploy ML Anaconda codes in ArcGIS environment. But in this case, developer waste the tuning capability to improve the performance accuracy and produce limited functions. Lastly, working with Anaconda became smoother after developing anaconda GUI navigator and Spider IDLE.

TensorFlow library employed to develop the NN classifier in this study. TensorFlow library produced by Google Brain team in Google. TensorFlow is one of most common open-source and up to date library worldwide for high performance numerical computation. TensorFlow library is available in Anaconda environment. Keras library from Tensorflow is a high level Python programming library that almost readable. It is a high-level neural networks API able to running on top of TensorFlow Keras design for ML beginners to build and design a Neural Network.In addition, several extra classifiers can be developed in Anaconda environment using Scikit-learn library. Varied available algorithms from Scikit-learn library can be implement in geospatial area such as random forest, support vector machine, regression, decision tree and MLP neural networks.

Table 3 Step F, image datasets classification in Anaconda environment using the TF developed classifier

```
# Import TensorFlow and helper libraries
import tensorflow as tf
import os
import glob
import numpy as np

# Import trained model
ModelPath = 'E:\Ankaradata\D\my_model5.h5'
new_model = tf.keras.models.load_model(ModelPath)

#Define location of output files of classified values
folderOfOutput = 'E:\Ankaradata\F_TFClassifireOutput'

#Define location of full image blocks datasets
folderpath = 'E:\Ankaradata\E'
folder2 = folderpath + '\*.txt'
folderOfOutput2= os.path.join(folderOfOutput, '')
filenames = glob.glob(folder2)

#Classify input datasets, and store classified output
for f in filenames:
  print (f)
  X = np.loadtxt(fname=f)
  normalizedX = tf.keras.utils.normalize(X, axis=1)
  del X
  y_pred = new_model.predict(normalizedX)
  del normalizedX
  # Convert binary class matrix to integer
  length_of_y_pred=len(y_pred)
  classifiedSet = []
  for i in range(length_of_y_pred):
    classifiedValue = np.argmax(y_pred[i])
    classifiedSet.append(classifiedValue)
  # Store classified Dataset
  basenameOfInputDatasetBlock = os.path.basename(f)
  FullPathOfOutputFile= folderOfOutput2 +
        basenameOfInputDatasetBlock[0:-4] + '.txt'
  print (FullPathOfOutputFile)
  np.savetxt(FullPathOfOutputFile, classifiedSet, fmt="%s")
  del y_pred
```

However, in this stage, Sequential model from TensorFlow-Keras used to developed the classifier. Sequential is a feed forward neural network model. Table 2 showing in details the real code of the classifier development in Anaconda environment. The code starting from importing the required libraries, processing the datasets, building, training, and evaluation the classifier. At the end of the code if the classifier is suitable, the user can store it for next stages in (h5) file format.

Finally, the last step in Anaconda environment is step (F). Step F classifying the prepared datasets of each block using the developed classifier. Table 3 explained in details the process of image datasets classification in Anaconda environment using the TF developed classifier. The code start by importing the developed method and required libraries. then, defining the files location of input dataset and output values. Lastly, classify input datasets, and store classified outputs as TXT files.

2.3 Tool implementation

The developed tools and codes were employed based on case study. RASAT satellite image used to implement the analysis. RASAT is first Earth observation satellite in Turkey and launched in 17 August 2011. RASAT is designed, manufactured and tested by TUBITAK Space Technologies Research Institute (TUBITAK UZAY), with funding from Turkish State Planning Office (DPT).

RASAT, providing high resolution optical imaging. The altitude is 700 km, orbital period 98.8 minutes, spatial resolution (PAN 7.5 m, RGB: 15 m), swath width 30 km and temporal resolution is 4 days. RASAT images are available in GEZGİN GeoPortal. GEZGİN is internet service authorized to deliver RASAT images to stakeholders. Processed RASAT products are available to public institutions and universities. RASAT images are shared in several raster and vector levels.

In this study, processed RASAT image used (see Figure 5) within Ankara region in Turkey. Ankara is the capital city in Turkey. The obtained image within L1RB category (radiometric corrected image, band registered image and cloud flares removed image). This image processed using the developed codes and tools. First, using developed tools in ArcToolbox, the signature AOI file generated for 4 classes (urban, agriculture, water and open areas) (see Figure 5). Then, augmentation processes applied and grid of points generated. Based on the points grid, each point extract the spectral content from the 4 RASAT image bands. The extracted attribute table processed and the ML training dataset created. Sample points of training data set was consisting 392 points. an example extract from the case study data illustrated in Table 1. The table show the explanatory parameters data and their target values. In addition, the image divided to several blocks and input bands datasets were established.

Second, in Anaconda environment, TensorFlow-Keras library employed to develop the classifier. The model classifier trained, tested and model performance accuracy evaluated. The trained model stored and load again for full image classification. The tool read and process over the dataset files to classify each dataset and store the classified values. Lastly, the classified files of blocks were processed and Mosaic, then full classified LCLU map Masked based on the border of study area.

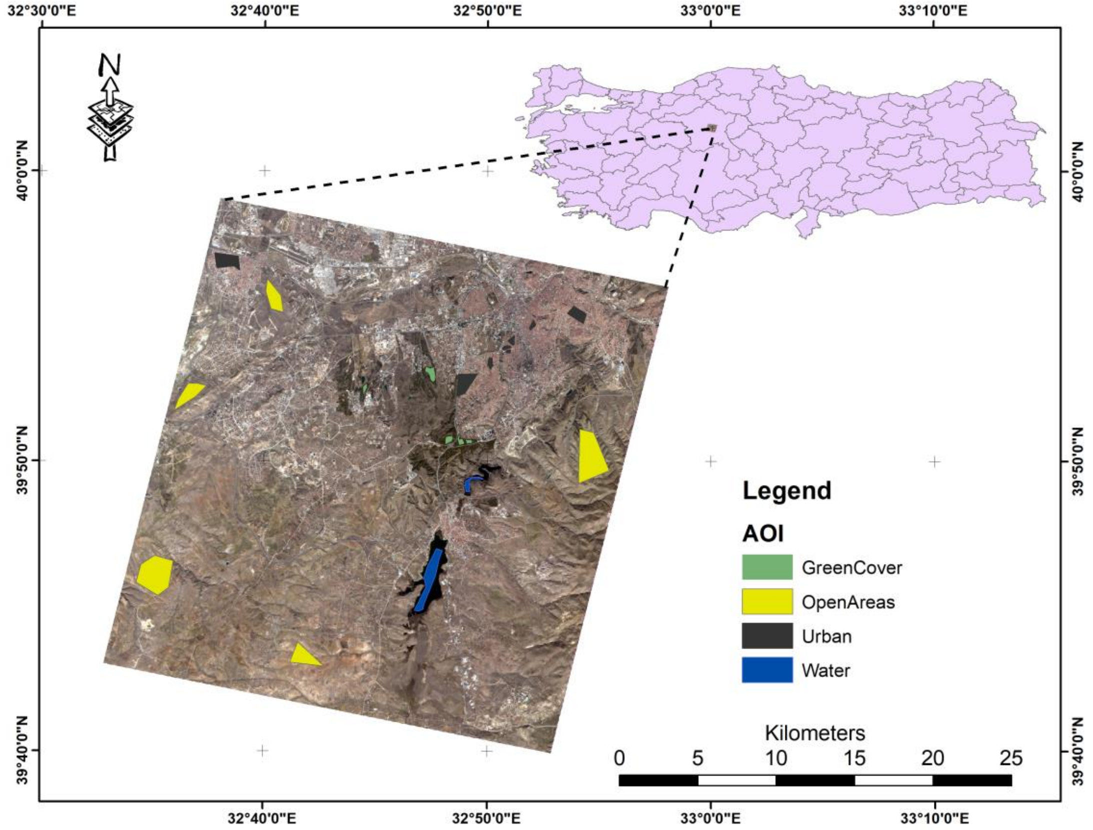


Figure 5 RASAT satellite image within Ankara region.

3. RESULT

In this section, authors present the result of the case analysis area. The case study applied using the developed tools in the tow software environments. Initially, authors evaluate applying neural network as ML algorithm to perform the classification. The RASAT satellite image classified to extract the LULC map. From the RASAT image in Ankara area, bands were obtained and processed until generating the training dataset. Then the dataset used to train the classifier using Keres library and neural network.

The classifier trained to achieve minimum error and high accuracy. Several training algorithms were implemented to gain the optimal performance accuracy. In addition, several number of neurons were applied. The optimal numbers were 1000 nodes in the first hidden layer and 100 nodes in the second hidden layer. The developed classifier gained 80% performance accuracy. Based on only 4 bands and compare to several past research, this accuracy can consider as high accuracy.

In order to classify the full image and extract LULC map without facing any problem with the memory, the full image divided to 13 blocks. The size of each block was (500 columns * 500 rows). The size of blocks determined after several attempts in personal computer.

After confirming the accuracy of the classifier, blocks of datasets were classified by the model. The classified values migrated to ArcGIS environment and combined in one raster map. The combined data generated the LCLU map as illustrated in Figure 6.

The complete RASAT satellite image processed and smoothly classified based on blocks methods, which handled the big data issue in satellite image classification. The developed tools can be easily applied from the side of GIS specialist without deep knowledge in computer programming. The tools reduced the number of processing steps to minimum number of steps.

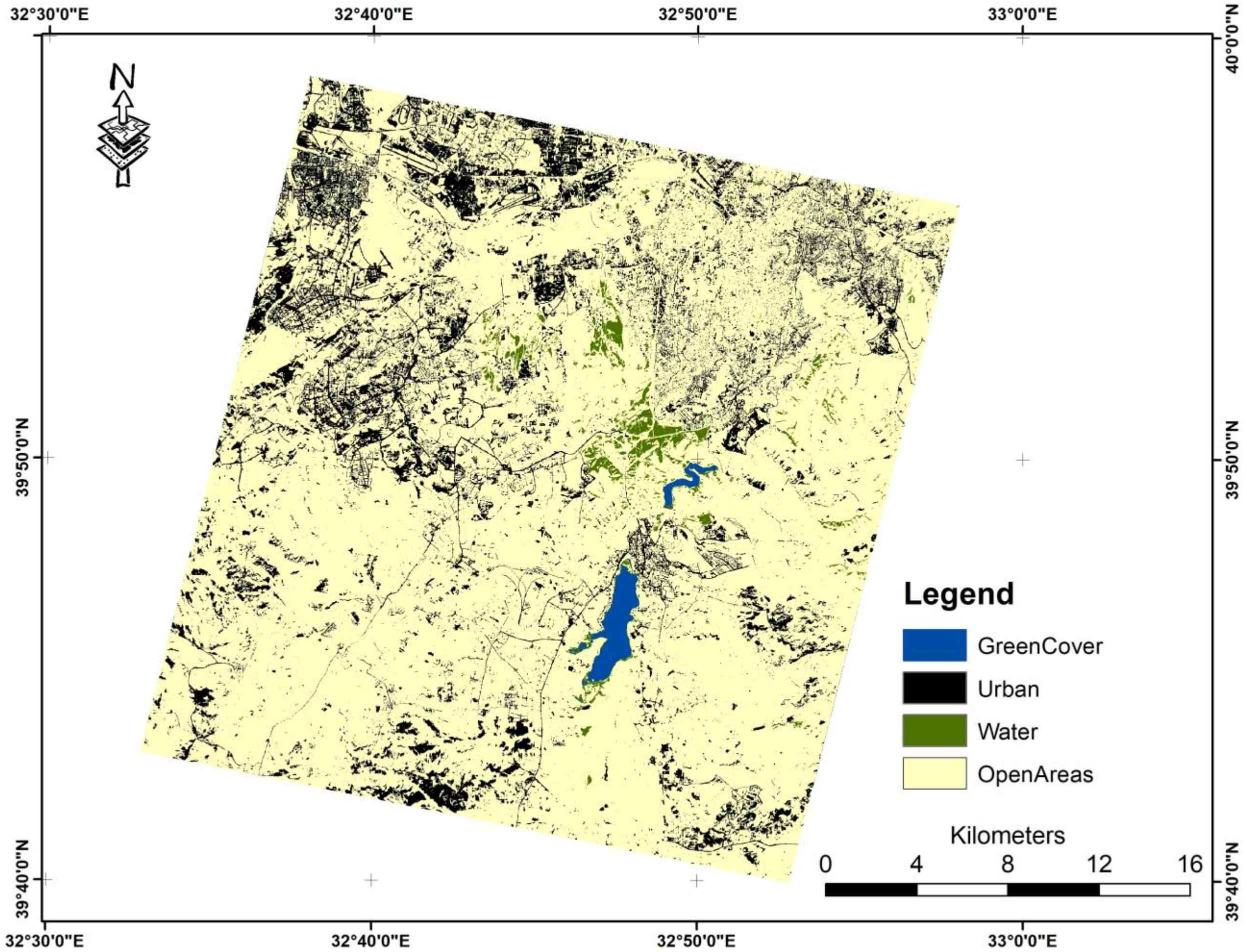


Figure 6 RASAT image and classified LULC map

4. CONCLUSION

This paper introduced a developed Python tools in ArcGIS and Anaconda software environments. The generated tools help the GIS users to easily structuring machine learning datasets, applying training analysis, developing classifiers and mapping LULC satellite images. In this paper, a case study applied based on RASAT images. The analysis executed in Ankara area in Turkey. The outcomes show that users can easily structuring machine learning datasets by using the automated tools. In addition, user can develop the classifiers and generate LULC maps with high accuracy. The authors recommend to test these tools using different satellite images in different geographic region.

ACKNOWLEDGEMENTS

This study has been supported by 2221 – Fellowship Program of TUBITAK (The Scientific and Technological Research Council of Turkey). We are indebted for their supports.

REFERENCES

Abburu, Sunitha and Suresh Babu Golla. 2015. “Satellite Image Classification Methods and Techniques: A Review.” *International Journal of Computer Applications* 119(8).

Alqurashi, Abdullah F. and Lalit Kumar. 2013. “Investigating the Use of Remote Sensing and GIS Techniques to Detect Land Use and Land Cover Change: A Review.” *Advances in Remote Sensing* 2(02):193.

Hamad, Rahel, Heiko Balzter, and Kamal Kolo. 2018. “Predicting Land Use/Land Cover Changes Using a CA-Markov Model under Two Different Scenarios.” *Sustainability* 10(10).

Haque, Md. Inzamul and Rony Basak. 2017. “Land Cover Change Detection Using GIS and Remote Sensing Techniques: A Spatio-Temporal Study on Tanguar Haor, Sunamganj, Bangladesh.” *The Egyptian Journal of Remote Sensing and Space Science* 20(2):251–63.

Ikiel, Cercis, Beyza Ustaoglu, Ayse Atalay Dutucu, and Derya Evrim Kilic. 2013. “Remote Sensing and GIS-Based Integrated Analysis of Land Cover Change in Duzce Plain and Its Surroundings (North Western Turkey).” *Environmental Monitoring and Assessment* 185(2):1699–1709.

Khamehchiyan, Mashalah, MohammadReza Reza Nikoudel, and Mehdi Boroumandi. 2011. “Identification of Hazardous Waste Landfill Site: A Case Study from Zanjan Province, Iran.” *Environmental Earth Sciences* 64(7):1763–76.

Li, Deren, Shuliang Wang, and Deyi Li. 2015. “Spatial Data Mining.”

Liao, Shu-Hsien, Pei-Hui Chu, and Pei-Yuan Hsiao. 2012. “Data Mining Techniques and Applications – A Decade Review from 2000 to 2011.” *Expert Systems with Applications* 39(12):11303–11.

Paul, Bimal Kanti, Harun Rashid, Bimal Kanti Paul, and Harun Rashid. 2017. “Land Use Change and Coastal Management.” *Climatic Hazards in Coastal Bangladesh* 183–207.

Pradhan, Biswajeet and Saro Lee. 2010. “Landslide Susceptibility Assessment and Factor Effect Analysis: Backpropagation Artificial Neural Networks and Their Comparison with Frequency Ratio and Bivariate Logistic Regression Modelling.” *Environmental Modelling & Software* 25(6):747–59.

Reis, Selçuk. 2008. “Analyzing Land Use/Land Cover Changes Using Remote Sensing and GIS in Rize, North-East Turkey.” *Sensors* 8(10):6188–6202.

Roy, Sanjoy, Kaniz Farzana, Mossammat Papia, and Mehedi Hasan. 2015. “Monitoring and Prediction of Land Use/Land Cover Change Using the Integration of Markov Chain Model and Cellular Automation in the Southeastern Tertiary Hilly Area of Bangladesh.” *Int. J. Sci. Basic Appl. Res* 24:125–48.

Shekhar, Shashi, Pusheng Zhang, and Yan Huang. 2005. “Spatial Data Mining.” Pp. 833–51 in *Data Mining and Knowledge Discovery Handbook*, edited by O. Maimon and L. Rokach. Boston, MA: Springer US.

Revised August 2019