17 October 2019

Training Guide for Using Random Forests to Classify Satellite Images

Introduction

This guide explains how to use an R script to classify satellite imagery using the random forest algorithm. To complete this guide you need to be able to digitize training areas and save them as an ESRI Shapefile. The output images will be in the GeoTiff format if the tutorial script is used but this script can be modified to work with other image formats.

The script included with this guide will need to be modified to use with your own data set. Near the top of the script are a few variables that must be edited to make it work for your data. The attributes that must be changed are noted throughout this guide.

The image that you use for the classification can include spectral data (image bands) and other continuous or categorical data such as DEM, climate layers, and soil maps.

The script has the following options:

- Use all samples in training polygons or randomly select a user-determined number of samples to train the random forests model.
- Use multiple cores to improve processing speed.
- Set a seed to ensure replicable results.
- Create a feature space plot to illustrate how well training data are distributed through features space. Different bands an be used for making the plots.
- Output an ESRI Shapefile or a GeoPackage file with an assessment of the quality of the training data (the margin).
- Output up to three images:
  - a classified image
  - a class probability image with the probability that the classification label is correct for each pixel of the class that got the most votes (i.e., the class that was selected for the classified image).
  - a classified image that has all pixels with a class probability below a user-defined threshold set to zero.

All of the options can be set in the variables section of the script. More information about these options and the variables used to set them can be found at the top of the R script. The script has gone through a number of revisions and although all of the new features

have been tested it is quite possible some bugs exist. Please use the contact information toward the end of this document to report any problems.

1. Download and install R on a windows computer

Instructions for downloading and installing R can be found on the CBC R scripts and guides website: http://biodiversityinformatics.amnh.org/index.php?section=R_Scripts. Install the following packages: snow, maptools, sf, randomforest, raster, rgdal, lwgeom.

2. Digitize training areas

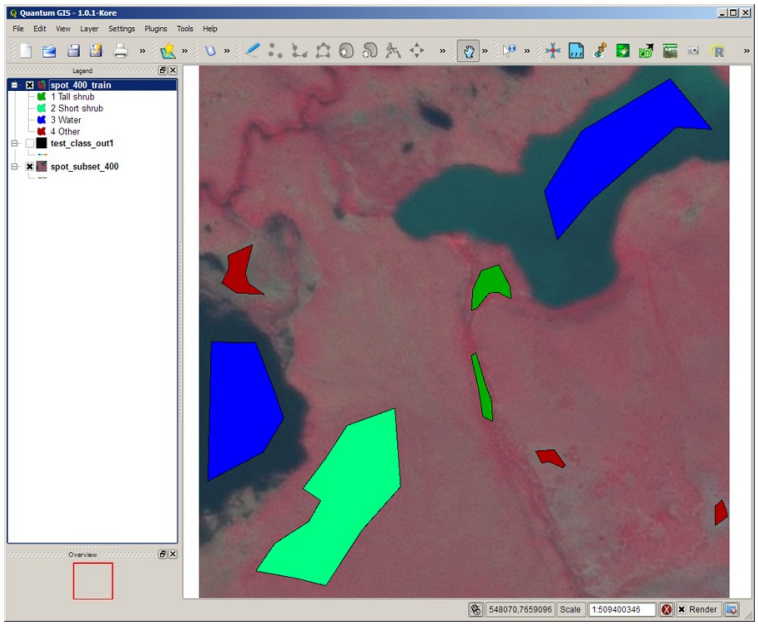
The script that goes with this tutorial requires that training areas are digitized as polygons and saved as an ESRI shapefile. Any GIS software can be used for this. If you do not have access to GIS software I suggest you try QGIS (http://www.qgis.org/), gvSIG (http://www.gvsig.com) or another open source GIS package but any GIS will work just fine.

The attribute table for the vector training data must have an attribute of data type integer that stores the class number for each class to be included in the output classified map. Additional attributes will be ignored so if you have another attribute with a text description of each class that is fine. Here is an example of an attribute table (note that the "cover" attribute in the example below is optional and will be ignored by the script).

id	type_id	cover
0	3	Water
1	3	Water
2	4	Other
3	4	Other
4	4	Other
5	1	Tall shrub
6	1	Tall shrub
7	2	Short shrub

The random forest algorithm is non-parametric so it is not necessary to keep training areas homogeneous. For example, you can have a "cloud and shadow" class with both clouds and shadows in it. You can have as may polygons that you want for any class. All of the polygons for a single class (defined by type_id in this example) are combined when the script is run. In other words, all of the pixels that fall under a particular class, even if there are several training polygons for that class, will be grouped together.

17 October 2019



3. Processing options

A number of processing options have been incorporated into the R classification script. The intent is to provide options that facilitate the creation of an accurate classified map. All of the options can be skipped if they are not required. Here is a brief explanation of the different options.

Sample selection

Training samples are selected using polygons created in step 2. The script provides the option to either use all pixels covered by the polygons or to randomly select a user-defined number of samples under the polygons. One thing to keep in mind when deciding which option to use is that the more samples you use the longer it will take for the random forests model to be created while the script is running.

Create feature space plot

There is an option to create a feature space plot to visualize how your training data are distributed across feature space. The tools allows you to select two bands for the feature space plot axes. After the plot is displayed a dialog is printed in the console that gives you the option to use other bands, define a rectangle to locate gaps in feature space, cancel the script, or continue on to the random forest model creation. If the option to define a rectangle is selected then you need to click on the feature space plot to define the upper left and lower right corners of a rectangle that falls within a gap in the training data plotted on the feature space plot. After the rectangle is selected your image will be plotted using the first three bands and all pixels that were selected by the rectangle will be displayed in white. After the plot is displayed the script will stop so you can add more training data to cover the highlighted pixels.

17 October 2019

For more information about feature space look at the interactive tools available at the CBC web site: http://biodiversityinformatics.amnh.org/index.php?section=rsr_tools.

Training data assessment

There is an option to assess training data quality using a metric called the "margin". The margin of a training sample is the proportion of votes that sample got for the correct class minus maximum proportion of votes for the other classes for that sample. Margin values range from 1 to -1. If margin is zero, it means your training class, and another class received equal votes for that sample. If it is positive, your training class was voted higher than another class which is what we like to see. If margin is negative, your training class received fewer votes than another class. At the end of the script the margin data are written to a point ESRI Shapefile so they can be overlaid on the image and segment polygons to assess which segments need to be removed and relabeled in the training data and it can also help determine which classes needs additional training points.

Image output selection

Up to three image layers can be created in the output image. The first choice is to create a classified image. This is the typical output of a classification algorithm. The classified image will have pixels labeled by the class number used in the training data.

There is also an option to create a class probability image which gives the probability that the classification label is correct for each pixel. A low probability means there was significant confusion with other classes and a high probability indicated low confusion with other classes.

The third type of output is a classified image that has all pixels with a class probability below a user-defined threshold set to zero. In other words, this output will look similar to the first option (classified image) except pixels that have a low probability value will be classified as zero ("0"). This can be helpful in determining which areas can use better quality training data.

Overwrite existing files

This provide the option to overwrite existing image and margin files. If "TRUE" then it will be possible to replace a file with the same name as an existing file. If "FALSE" then an error will be thrown and you will enter Debug mode to alert you that the file exists.

Use multi-core processing

If you have multiple processing cores in your computer you can use these to speed up processing. If this is set to "TRUE" then multiple cores will be used and if "FALSE" a single core will be used. If "multiprocessing" is set to TRUE you can set the number of core you want to use for processing. It is good practice to set this to a number lower than the total number of core on your computer otherwise other programs might not respond quickly.

17 October 2019

4. Editing the script

The random forest classification process is controlled through the use of a script. By modifying this script you are able to customize it for your own application. Near the top of the script there are a number of attributes that can be changed to customize it for your application. Here is a list of the attributes with an explanation about the attribute:

setwd() = This is where you specify the working directory where your files reside. If the files specified in the script are in the directory specified by setwd then only the file name is needed. Otherwise the full or relative path must be input when specifying a file in this script.
set.seed() = This is used to set a seed so your results can be replicated. As long as the number entered as the function argument is the same between run the results should be the same.
overwrite = Allows existing file to be overwritten when the same file names are used.
multiprocessing = Control if multiple processors should be used to speed up processing.
cores = If multiprocessing is set to "TRUE" this specifies how many cores should be used.
shapefile = Name and path for the Shapefile or GeoPackage file with the training data. The filetype (Shapefile or GeoPackage) will determine the output file type of the margin file. GeoPackage files are the preferred format.
classNums = Class numbers that you want to select fraining sample from. This value must be specified as a vector (e.g., c(1,2,3,4,5)).
classSampNums = For each land cover class specify the approximate number of training samples to be randomly selected. If a value is "0" then all pixels in all of the polygons for that classwill be used. This value must be specified as a vector and the vector length must be the same as the vector length for classNums.
attName = Name of the attribute in the ESRI Shapefile that holds the integer land cover type identifier.
nd = No-data value for the input image
inImageName = Name and path for the input satellite image (GeoTiff works and other formats probably do as well)
marginFile = TRUE or FALSE option to specify if you want to output a margin file. If set to TRUE the margin file will have the same base name as the image with "_Margin" appended to the end. The output file type will be the same as the training data vector file.
classImage = Output classification image without applying threshold (enter TRUE or FALSE)
probImage = Output probability image (enter TRUE or FALSE)
threshImage = Output classification image and set pixels with probability less than "probThreshold" to 0 (enter TRUE or FALSE)
probThreshold = Enter threshold probability in percent (values must be between 0 and 100) only used if threshImage=TRUE

xBand and **yBand** = Layer number (band number) for the X and Y axis of the feature space plot. If you do not want to calculate a feature plot enter zero ("0") as the layer number

Notes: When specifying the directory path in R for a Windows computer it is necessary to use a double-backslash ("\\") instead of a single backslash ("\") or you can use a forward slash ("/") which will also work on Apple and Linux operating systems. For example the directory path: C:\Data would be typed C:\\Data or C:/Data.

R is case sensitive. In other words "a" is different from "A". Make sure the directory path and file name is exactly as it appears when you list the directory contents. Also, note that in R variable or file names should not start with a number or most special characters. It is best to start variable names with an upper or lower-case letter.

Other parts of the script can also be modified but you will need to have a good understanding of how the different commands work. The script file must be saved as an ASCII text file.

4. Running the random forest script

To run the random forest classification click on "File => Open script" and then navigate to the script and click on "Open". The script will open in the R Editor. You can run the script one, or several lines at a time or you can run the entire script by Edit => Run all (note that the R Editor window must be selected for this option to appear).

You can also type the path and the name of the script file into the R Console using this syntax: source("path and filename"). For example:
source("C:\\IPY\\R_scripts\\rf_classification_windows.R"). Remember to use double backslash instead of single backslash in the directory path unless you are running the script in Linux in which case you would use a single forward slash.

When the processing starts messages are printed in the R console and once the image classification step begins a status bar is displayed so you can monitor the progress. Large images can take several hours to process.

The output image name(s) will be the same as the input image name with the text "_Class", "_Prob" and/or "_Thresh" appended to the and of the file name. When the classification is finished you will need to check the result to see if it is okay. If it is not you will need to select more training sites and/or edit some you have already defied.

Appendices:

A – Citations and license information

If you cite this document we ask that you include the following information:

17 October 2019

Horning, N. 2019. Training Guide for Using Random Forests to Classify Satellite Images. American Museum of Natural History, Center for Biodiversity and Conservation. Available from http://biodiversityinformatics.amnh.org/. (accessed on *the date*).

B – Acknowledgements

I would like to thank the The John D. and Catherine T. MacArthur Foundation and Google.org for supporting this effort. I would also like to thank Jonas Viehweger for his work to add multicore processing and rework the algorithm for selecting training data as well ass the random forest prediction approach. All of these changes resulted in a many-fold reduction in processing time.