Exercise

Image classification using a Neural Network
Exercise outline

Time
0.25 day.

Objectives
To provided practice in procedures to infer land cover information from optical polar-orbiter satellite observations and artificial neural networks.

Software
IDL/ENVI

Data – located on Blackboard “Exercise data” or network drive
- Landsat ETM+ (2002) imager data, bands 1,2,3,4,5 and 6 covering part of our study area in Uganda.
- Region of Interest (ROI) listing all the ground observations.
Introduction

This exercise falls within the “advanced” group of image processing techniques, and focuses on classifying land cover using optical satellite imagery and a neural network. In the next sections we will describe the datasets, which are available for this exercise, explain the relevance of certain classification steps and follow these for each input. Although some principles also apply to infrared and radar data, this exercise deals with optical data only, which we will call “VNIR” for simplification (VNIR = Visible and Near-Infrared).

For this exercise NeuralNet, a supervised classification method according to the back propagation method for artificial neural networks is used. The full method is described at page 137, chapter 5 of the book “Explorations in parallel distributed processing: a handbook of models, programs, and exercises” by McClelland, J.L., Rumelhart, D.E.

Data description

The imager sensor aboard the Landsat 7 spacecraft is called the Enhanced Thematic Mapper Plus (ETM+), see also http://landsat.gsfc.nasa.gov/. The Landsat 7 satellite was launched into a 705 km high orbit. This lower orbit was chosen to make the satellite potentially retrievable by the space shuttle and to improve the ground resolution of the sensors on board. Each orbit takes approximately 99 minutes with just over 14.5 orbits completed each day. This orbit results in a 16 day repeat cycle, meaning a single location on the earth's surface can be imaged every 16 days. As with the previous Landsat satellites, Landsat 7 does not have off-nadir viewing so a daily world-wide coverage is not possible. Landsat 7 ETM+ imagery looks much the same as previous Landsat TM data as they both have a spatial resolution of 25m. A full scene makes up 185 km2, therefore this medium resolution sensor covers a large surface area.

Radiometrically, the ETM+ sensor has a quantization range of 256 digital numbers (8 bits) which permits observation of small changes in radiometric magnitudes in a given band and sensitivity to changes in relationships between bands.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>File name(s)</th>
<th>Date</th>
<th>Pixel size (nadir)</th>
<th>Number of layers</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROI</td>
<td>Roi_with_one_word_classes.roi</td>
<td></td>
<td>1– n pixels</td>
<td>-</td>
</tr>
<tr>
<td>Imager data</td>
<td>etm_envi</td>
<td>February 2002</td>
<td>30 m</td>
<td>6</td>
</tr>
</tbody>
</table>

Table 1. Dataset description

The ETM+ bands are useful for water penetration, discriminating vegetation types and vigour, plant and soil moisture measurements, differentiation of clouds, snow and ice and identifying rock types. Similar to Landsat TM, Landsat ETM+ can be used for urban applications but its high spectral resolution makes it more suitable for making the natural characteristics of the landscape. Normally, the data are distributed in the HDF format, but for your convenience already converted to ENVI/IDL raster files.

Budongo Forest Reserve is located in the northwestern part of Uganda and it consists of two forest blocks, Main Budongo (made up of Budongo, Siba, Busaja forests) and Kaniyo-Pabidi forest blocks (Figure 1). It is situated in the districts of Masindi and Hoima with the largest part falling in Masindi. It is located between 1° 35′–1° 55′ N and 31° 18′ – 31° 42′ E on the edge of the western rift valley. Budongo forest is classified as a medium altitude, moist semideciduous forest (Langdale-Brown et al., 1964). Budongo Forest Reserve was gazetted as a central forest reserve in 1932.
The reserve, which is a mixture of tropical high forest with a large population of mahogany species e.g. Khaya spp. and Entandrophragma spp., woodlands and savanna grasslands (Hamilton, 1984), covers 82,530 ha, making it Uganda’s largest forest reserve. It consists of 53.7% forest and 46.3% “grassland”. A woodland-savanna area, interspersed with forest patches, commonly referred to as Kaniyo-Pabidi Woodland, separates the Main Budongo forest block from the Kaniyo-Pabidi forest block.

**Software Notes**

The C code covered in this book has been translated into IDL code. The calculation method has only been changed to exploit the power of IDL. It avoids the use of matrix indexes and the loops have been replaced by matrix operations. The result is a reasonable fast learning process.

The source code is also available in the distribution.

The neural net variables are read and write by access functions, which are only working with linear arrays. A Neural Network is mapped on a linear array by the functions `Create_descriptor` and `Create_network`.

- The function `Create_descriptor` defines a linear array with the offsets to access the values of one layer relative to the start location of a layer
- The function `Create_network` fills the neural net variables partly by preset values, partly by random values.

The values of a neural net can be read / modified by the access functions:

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>get_weights/set_weights</td>
<td>get or set all the weights of one layer</td>
</tr>
<tr>
<td>get_act/set_act</td>
<td>get or set all the activations of one layer</td>
</tr>
<tr>
<td>get_delw/set_delw</td>
<td>get or set all the delta weights of one layer</td>
</tr>
<tr>
<td>get_error/set_error</td>
<td>get or set all the errors of one layer</td>
</tr>
</tbody>
</table>
The training of a neural net is done by the procedures:

<table>
<thead>
<tr>
<th>Procedure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preprocess vector</td>
<td>normalizes the input activation by normalizing the input vector</td>
</tr>
<tr>
<td>Forward network</td>
<td>calculates the output activation vector given the input activation</td>
</tr>
<tr>
<td>Rumelhart</td>
<td>corrects weights by the back propagation algorithm</td>
</tr>
</tbody>
</table>

The program NeuralNet contains many other non specific function and procedures for the realization of the interface.

For general use CW_SelectFromDomain.pro file contains the procedures and functions of a compound widget useful for the interface function for moving items from one list to another list.

The software is packaged in the neuralnet.zip file, containing:
- neuralnet.sav  The compiled module
- source.zip  All the source code
- data  A folder with some example data:
  - etm_envi
  - etm_envi.hdr
  - woodland.roi
Installation and configuration Notes

To install the software, extract "neuralnet.sav" from the zip file into the "save_add" folder in ENVI. (You can find the location of the "save_add" folder in ENVI from the "File | Preferences" menu on the "Default Directories" tab page; see the highlighted part on the image.) After placing the "neuralnet.sav" file into this folder, you need to restart ENVI.

Now open IDL. In the message pane check if you see the message highlighted below:

If this line is there you are setup.
Now extract the data folder to any folder of your liking.

Operation

There are two ways to start the module:

- On the ENVI command line in IDL, type: "neuralnet". (If you don't see a command line, toggle the setting of the "Window | Command Input" menu)
- From the ENVI menu, click “Classification | Supervised | Neural Net (NRS)”: 
The interface of the NeuralNet is relatively straightforward:

For “Input Layers Filename” click on the “Browse” button to enter the satellite image and ground observation file (ROI). You will see a regular ENVI open dialog: open your image and then via “Open | ROI file…” select the ROI file. The module automatically proposes a name for the output. Also default parameters are proposed.

The user interface show several fields and parameters:
- Numbers of hidden layers,
- Hidden layers for layer 1 (to max 3)
- Learning rate and Momentum
- Max Epochs
- Maximum System error
- Individual Error
- Facility for input layer selection, and selection of participating classes.

The number of hidden layers and the number of nodes / layers are related. If the number of hidden layers is $N$, then only the nodes of layer $1..N$ will be read.

Note: Changing the number of (hidden) layers or the number of classes in your ROI voids a previous training process.

Press button “Learn” to start training the network. The progress of the learning process is displayed in the progress window and can be interrupted by the user. When interrupted a report is displayed so it is possible to check the learning quality. If it is not high enough, the learning phase can be resumed.

After the learning phase has finished click “Apply” to start the classification based on the neural network. The progress can be monitored in the progress window. When interrupted, the application will store the classification up to the point of interruption. The remainder is then filled with zero valued pixels.
Explore the performance of a Neural Network.

1) Stochactical Test
Train a Neural Net 2 times with the same parameters:
Maximum System error = 0.01

Store classification 1 as [your directory]Nrstudy01
Store classification 2 as [your directory]Nrstudy01

With the selection in the ENVI menu you can see the differences of the two classifications by a confusion matrix: Go to: “Classification | Post Classification | Using Ground Truth Image”. Explain the difference below:

2) The “Learning Rate” parameter
Keep the other parameters constant at the following values:
• Number of input nodes = 6
• Maximum system error = 0.001
• Momentum = 0.7
And vary the learning rate from 0.9, 0.5, and 0.1
Plot the results below and explain the influence of the learning rate.

3) The “Momentum” parameter
Keep the other parameters constant at the following values:
• Number of input nodes = 6
• Maximum system error = 0.01
• Learning rate = 0.9

And vary the momentum from 0.9, 0.5, and 0.1
Plot the result and explain the result

Nr of epochs

<table>
<thead>
<tr>
<th>Band Count</th>
<th>#Epochs</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

4) The “Number of Layers” parameter
Reduce the input layers from 6 bands to 3 bands
Set the other parameters to the following values:
• Number of input nodes = 6
• Maximum system error = 0.01
• Learning rate = 0.9
• Momentum = 0.7
Record the number of epochs required for 6 bands and for 3 bands. Is there a difference?

<table>
<thead>
<tr>
<th>Band Count</th>
<th>#Epochs</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

Explain the result:

5) The “Number of epochs” parameter
Fix all parameters and learn with 25, 50, 100, 200 and 400 epochs.
Record the total system error and the % correctly classified pixels.

<table>
<thead>
<tr>
<th>#Epochs</th>
<th>System error</th>
<th>% Correct</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td></td>
<td></td>
</tr>
<tr>
<td>50</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>200</td>
<td></td>
<td></td>
</tr>
<tr>
<td>400</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Explain the result:
References

“Explorations in parallel distributed processing: a handbook of models, programs, and exercises” by McClelland, J.L., Rumelhart, et al.