GDAL OTB QGIS in a virtual environment

The Orfeo (https://www.orfeo-toolbox.org/tag/machine-learning/) machine learning library and the translator library for raster and vector geospatial data formats GDAL (https://gdal.org/) do not play well with the open source geographic information system QGIS (https://qgis.org) outside of the resource intensive GUI environment.

The systems have differing and partially conflicting dependencies, and the GUI environment makes rapid testing across the various tool boxes cumbersome. This fact becomes apparent, for example, when developing a pipeline to perform vector-based satellite image classification with a neural network under OTB.

This repository demonstrates how to setup a virtual computer that can support GDAL, OTB and QGIS functionality seamlessly. It also includes examples of how to use the setup to apply various classifiers including vector support machines, random forests and neural networks across GDAL, OTB and QGIS.

1 - VM

Create a virtual computer.
Choose Ubuntu 18.04 LTS, with at least 16GB of RAM

2 – Get the files from GitHub

At the prompt type:	*git clone https://github.com/realtechsupport/gdal-otb-qgis-combo.git*
Enter the username:	*realtechsupport*
Enter the access token:	*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*

3 - Install QGIS

Make the file basics+qgis.sh executable.

```
chmod +x basics+qgis.sh
```

Run that file

```
sh basics+qgis.sh
```

4 - Install OTB in a virtual environment with conda

Make the file conda-packages.sh executable.

```
chmod +x conda-packages.sh
```

Run that file

```
sh conda-packages.sh
```

5 - Create a conda environment with defined dependencies.

(OTB 7.2 requires python 3.7, for example)
Create a conda environment with the settings in the environment.yml file.

```
conda env create -f environment.yml
```

6 - Customize
Add other libraries to the OTB environment as needed

conda install -c conda-forge pillow

(Enable the OTB environment to install geojson and geopandas.
The geopandas install may take some time…)

conda install -c conda-forge geojson
conda install -c conda-forge geopandas

7 – Setup the directory structure
The directories code, data, results should be ready made in your repository directory.
Put the rasterimages and vectorfiles directories into the data directory. Place image assets (.tif) in the rasterimages directory and vector assets (.shp) into the vectorfiles directory. Add the corresponding .dbf, .prj, .shx files for each .shp file.

Use the settings.txt file (in the data directory) to set your file names, process preferences and classification parameters. The content of this file is parsed and passed to the classification steps.

8 - Test
Adjust the parameters in the file data/settings.txt to work with your data directory.

a) Test QGIS

Run the qgis test in the base environment.

python3 qgis_test.py

You should see a list of operations supported by QGIS.

b) Test OTB

Activate the OTB environment.

conda activate OTB

Run the otb test

python3 otb_test.py

You should see a list of operations supported by OTB as well as the output of the settings.txt file.

You can now use QGIS in the base installation and enable the conda environment to access OTB functionality. You can also move across environments to access libraries from either environment with python scripts as outlined below (see vector_classify_top.sh):

```
echo "Starting OTB-QGIS pipeline...\n\n"
conda run -n OTB python3 /home/code/otb_code_A.py
python3 /home/code/qgis_code_A.py
conda run -n OTB python3 /home/code/otb_code_B.py
python3 /home/code/qgis_code_B.py
```

Since the directory structure you setup will be identical in both the QGIS and OTB environments, intermediate filed produced will be available to processes running in either environment, allowing for data to be shared at no extra computational cost. All settings across the script modules are stored in the 'settings.txt' resource file and imported to the individual modules.

Examples:

After installation and testing you can perform operations as such (provided you have already prepared your data accordingly.)

If you are using pCloud to move results from the VM to the pCloud server, install the library in the OTB environment (to this only once).

```
conda activate OTB
python3.7 -m pip install pcloud
```

Pixel based classification (SVM or RF)

1) Open the settings.txt file and change the parameters “raster image” and “raster shapefile” to your data. Make sure all the data paths are correct.

2) Enable the OTB environment:

```
conda activate OTB
```

3) Run the program with the selected classifier as input argument (libsvm or rf):

```
python3 otb_raster_classify.py libsvm
```

4) You can also vary the parameters of the classifiers and batch execute either rf or libsvm classifiers for each of the possible combinations. Edit the file otb_vary raster_classify.py to edit the parameters you want to include and iterate across and select the classifier. Then run the script:

python3 otb_vary_raster_classify.py

Results from all permutations of the selected parameters under the chosen classifier will be saved to the results folder. If you installed pCloud and enabled uploading, these results can be saved to that environment.

Object based classification (ANN)

1) Deactivate the OTB environment first

conda deactivate

2) Open the settings.txt file and change the parameters "raster image" and "points file" to your data.
3) Check the paths in the vector_classify_top.sh file. Run the program:

sh vector_classify_top.sh

This script runs all the processes across multiple otb and qgis scripts to perform the object-based classification.

In both pixel and object-based classification, the performance of the classifier is evaluated (recall, precision, f1-score) for each category and saved to the results folder.