DePaul University – College of Computing and Digital Media
CSC 478 – Programming Data Mining Applications
Professor Bamshad Mobasher

Final Project Report

Project Title: Enhancement and Rebuild of Landsat 8 Imagery of Central Brazil, for Thermal Infrared Prediction

Student: Ricardo Barros Lourenço – DePaul ID# 1489379

Fall 2015

Introduction

Satellite imagery is a powerful data source that has been around since the early 1970's. Among several satellites, the Landsat project (formerly defined as Earth Resources Satellite) is the most known and used satellite imagery source. Formerly operated by the US National Aeronautics and Space Administration (NASA) and succeeded by the US National Oceanic and Atmospheric Administration (NOAA), is the longest running enterprise for acquisition of satellite imagery of Earth, beginning in 1972. On this satellite is a passive sensor, that through several wavelengths (bands), generate source information that is used in several domains, such as: agriculture, cartography, geology, geography, forestry, regional and urban planning, surveillance, education, among others.

On early 2015, Amazon have made a partnership with the US Geological Survey (USGS) to distribute images from Landsat 8 free of charge to anyone with internet access, just a few hours from the image being recorded. Also has made a compromise to distribute previous mission records in the long term. The very great advantage when having services hosted by Amazon AWS, is having those images stored on their repository, saves a lot of storage space, being just necessary to point to their S3 bucket, or internet URL and connect your systems to the data. Considering that each scene (group of images of a certain area) has 6GB of compressed GeoTIFF files, is interesting to save the source space, and persist just final and/or intermediate results.

Thermal Infrared (NIR) comprehend a spectrum ranging from 0.3µm to 0.3 cm of wavelength, being an energy emitted by all objects with temperature above zero Kelvin. Usage of this wavelength is broad, but one of the most used cases is on fire surveillance and associated deforestation surveillance. Despite having a sensor dedicated for it, images acquired on this specter have several problems on *scattering*, which is a problem related to multiple mutual interference patterns from bodies that emit radiation, including gases, so on certain weather and natural conditions those images get blurred and noisy on a high detail level.

A good approach would be if looking to other wavelengths, is possible to acquire relevant information about Thermal Behaviour, but without scattering concerns, and keeping the high resolution of images (pixel side of 30m).

Objectives

Considering this previous scenario, this project objective is to provide an accurate estimate of the Thermal Infrared values, in case of having too much scattered data on Band 9 (TIR 1 Band on Landsat 8).

To achieve this objective all Landsat 8 bands, except band 8 (panchromatic, overlapping with bands 2, 3 and 4) and Band 10 (TIR 2, transition band to Microwave domain) would gathered, normalized and object of Principal Component Analysis(PCA) for obtaining compressed features able to describe the different bands involved, and also another features that are able to describe each point on space.

After obtaining these features, both would be used for a Matrix Factorization aiming to build a synthetic data with accurate representativity of the original dataset, which would be used for regression analysis, taking Band 9 as target and Bands 1 to 7 as source attributes.

Depite PCA application, the objective is to rebuild a synthetic dataset, because of the possibility of image rebuilding.

Implementation and Processing

The mentioned objectives were implemented in Python, using an Anaconda-based Jupyter Notebook platform. For this implementation it was used Numpy, Pandas, Scikit-learn and GDAL libraries.

Aside of those three initial, GDAL library is not very often used on this class, and comprehend a very extensive library for geospatial data manipulation, including several algorithms used on GIS and Remote Sensing processing software, such as QGIS and ESRI ArcGIS. Basically on this work it was used their module to open GeoTIFF images, which is a sub-type of the TIFF image compression schema (based on a Huffman lossless data compression relying on a frequency-sorted binary tree), coupled with a header containing several metadata on acquisition parameters, projection, and geographical coordinate systems used for that image.

On Landsat 8 GeoTIFF images, there is not a stacked image, so, when you receive a file, it just contains one, regarding to a determined Band. Each image contains 58,304,271 pixels, each one regarding to a squaroid (not perfect, because of projection issues) position on space, with size of 30 meters. Band 9, which is also of 30 m of side, but uses data from a sensor acquiring with 100m of side, and applying decimation of data. This is not a very good practice, because usually introduce artifacts on images, and is probably done to fulfill user convenience. On this aspect, the approach of estimating TIR values from other bands is interesting on the sense that those are acquired natively on 30m.

The developed algorithm loads all bands on-the-fly from an Amazon S3 previously chosen bucket. The scene comprehends an area that includes the Great Brasília Region, on central Brazil, on a cloudless day (0% of clouds) , more precisely on September, 18th 2014, being the most recent cloudless scene available (Initially this project aimed on image reconstruction from cloud removal, but this seems to be tricky if not includes active sensors products, such as polarimetric radars, that are able to cross clouds, on the training phase as targets – and they are not provided on a free cost such as Landsat).

Those images are opened and the image matrices serialized to vectors, being each one representative of an image. So, the initial load uses bands 1-7 and 9-10 and form a numpy matrix of 58,304,271 lines (each one a specific point on space) per 9 collumns (each one being a Landsat 8 band, discarded

Band 8). Once loaded, this matrix is duplicated, being one kept for benchmark as the original data, and the other would be submitted to PCA analysis, for feature extraction, both on Band domain, and on the spatial domain.

I would like to talk about the problems I've faced while processing. When thinking on this initial approach, on applying matrix factorization for remote sensing data, I didn't realize that if I would include all Bands at once for processing, I would be dealing with several problems on data manipulation, environment tuning, and algorithm design. An initial problem emerged after matrix replication, when running PCA for the first domain, being the lack of memory. When proceeding to PCA, the amount of memory used was above 10 GB. My personal notebook has 16GB, but after running the processes, my entire machine froze. Discussing this with friends, I've tried to force garbage collection, but with minor improvement, and still being locked. I also tried to use cPickle library to dump memory objects on disk and force garbage collection, but it was very slow, taking more than half an hour just to dump a variable. Looking on documentation and on forums of QGIS, they usually process data on several batches, and dumping and restoring them while needed. However this was just done for visualization purposes, and not for machine learning. The found solution was to use an Amazon EC2 instance. I've picked a strong cr1.8xlarge, with 244 GB of RAM and 32 cores. My code was not parallelized , but it was a good solution for my RAM bottleneck (processing sometimes had peaks of 80GB of RAM while running) and on a Spot Market it just costed $0.70/hour, with the minor tradeoff of rebuilding it if a competitor bid was higher, but using Anaconda helped a lot.

The Principal Component Analysis was done using the implementation available on scikit-learn, *decomposition.PCA*. It was not indicated for large matrices, but after tackling the problems with the Amazon instance, it was feasible, calculating the attributes in less then 2 minutes. I've tried the *IncrementalPCA* method, but aside not improving much the performance of run, it didn't provide the *explained_variance_ratio_* method, which is necessary to evaluate the explained variance of attributes generated. Prior to PCA runs, each band-vector was normalized using scikit-learn Min-Max Normalization method

For both domains, Bands and Space, it was extracted three principal components, being the explained variance 98.01% and 99.28% respectively. This is a very good result, because corroborates on several assumptions when recommending the usage of PCA that much of what you could have on your dataset is noise, because it was reduced to three variables one domain with 9 factors, and other with enormous 58 million. Even considering that those would be used for synthetic rebuilding the data on its original size, it is worth to consider that the data was enhanced for each point on space, for each band.

The data rebuilding was done by a dot potroduct among the factor matrices as :

$$(58304271 , 3) \times (3 , 9) = (58304271 , 9)$$

Then those matrices were compared in terms of cosine similarity between each pair of collumns(band pairs), with results being:

```
Cosine similarity between original, and rebuild images:
Band 1 : [[-0.52924785]]
Band 2 : [[ 0.52532227]]
Band 3 : [[-0.54002619]]
Band 4 : [[-0.53923166]]
Band 5 : [[ 0.50853795]]
Band 6 : [[-0.53509485]]
Band 7 : [[-0.5387424]]
Band 9 : [[ 0.52819287]]
Band 10 : [[ 0.53043943]]
```

These were not probably the best results, possible, on direct comparison, and perhaps susceptible to enhancements in terms of de-normalize the data, but this could also represents a reflection on the noise reduction, from original to the PCA dataset.

Considering this need for benchmarking, it was applied on both datasets, original and PCA-generated, a feature extraction algorithm to calculate Normalized-Difference Vegetation Index(NDVI), which is a normalized measure of biomass, using bands 4 and 5 for this calculation, which formula is represented below:

$$\mathbf{NDVI} = \frac{\mathbf{(NIR - Red)}}{\mathbf{(NIR + Red)}}$$

Where NIR corresponds to Landsat 8 - Band 5, and Red (which is the visible specter Red) corresponds to Band 4.

After this, it was also compared the cosine similarity which presented impressive 1.0 value, which determines full similarity. This is quite impressive in terms that even with massive compression of PCA, it was preserved one of the most important usages of Landsat Imagery, that is related to biomass level, and directly related to forestry and agricultural surveillance, being used worldwide for years.

Finally, using both original and PCA datasets, it was proceeded to regression analysis, aiming to establish a model able to predict Thermal Infrared values as like of Band 9. Band 10, also a TIR band, was not used due to its wavelength proximity to Microwave spectrum, which could led to biased results.

For this I tried to review regression algorithms already implemented on scikit-learn. It was chosen a Ridge Regression, due to the risk of overfitting the dataset, due to the large amount of samples.

The dataset was split on a randomized test-train split being kept for testing 90% of the dataset, and 10% for training. This is quite sufficient, because the 10% is about 5 million samples. Also to improve model stability, it was chosen a RidgeCV implementation, on which was used 10-Fold Cross-validation.

For this case it was calculated initially the original dataset, for benchmark comparison which had a cosine similarity between estimated predictor and true predictor on testing of 0.9873 (on training the same value), which is quite good. However, the calculated Mean Squared Error (MSE) was really high, with value of 17500855.73 (17527832.80 on training).

On a second round it was used the PCA treated dataset, which had both on training and testing a cosine similarity of 1.0. Despite expectations on overfitting(due to cosine similarity value), the MSE have been ranked at impressive 6.9071540716417226 e^{-18}.

Final Remarks

On this final project I could remark the gain on experience of dealing with a data that doesn't fit on a consumer PC. Despite of Amazon availability, this kind of processing is a very good candidate for distributed processing, specially on spark environment, which has support for abstractions on distributed matrices and support for all python packages.

In terms of method implementation, despite of final good results, perhaps they should be still under evaluation. A model produced for estimating thermal infrared values, is worth in the sense of its spatial coverage, and perhaps not suiting to a more general case, where imaging conditions could be different, such as sunlight exposure, date of the year, or even scattering effects from vapors. Perhaps working on a distributed framework with more regional data, helps on understanding better the results.

Another good improvement should be testing other regression methods, special ones that have possibility to be implemented in parallel, avoiding data serialization, and applying stochastic gradient descent for model discovery. However, if not making the algorithm scalable, this approach could turn into a very strong bottleneck for performance.

References

Amazon Landsat Blog Post: https://aws.amazon.com/fr/blogs/aws/start-using-landsat-on-aws/?sc_campaign=launch&sc_category=dataset&sc_channel=SM&sc_content=landsat&sc_detail=std&sc_medium=aws&sc_publisher=twitter&adbsc=social_launches_20150319_42385646&adbid=578650521231319041&adbpl=tw&adbpr=66780587

USGS Landsat 8 Band Description: http://landsat.usgs.gov/band_designations_landsat_satellites.php

Geospatial Data Abstraction Library - GDAL : http://www.gdal.org/

Used Landsat Repository: https://s3-us-west-2.amazonaws.com/landsat-pds/L8/221/071/LC82210712014261LGN00/index.html