# Spatial Analysis Using Convolutional Neural Networks

## Objective

Deep learning is a subset of machine learning which deals with Artificial Networks that are composed of many layers. It is a growing trend in the field of Machine Learning due to its effectiveness in applications where the target function is very complex and the datasets are large. Convolutional Neural Networks (**CNNs**) are a category of Neural Networks that have proven very effective in areas such as image recognition and classification. CNNs have been successful in identifying faces, objects and traffic signs apart from powering vision in robots and self-driving cars.

This project studies the use of CNNS to analyze aerial imagery from ArcGIS online, and reports the findings.



Figure : Neural Network Structure

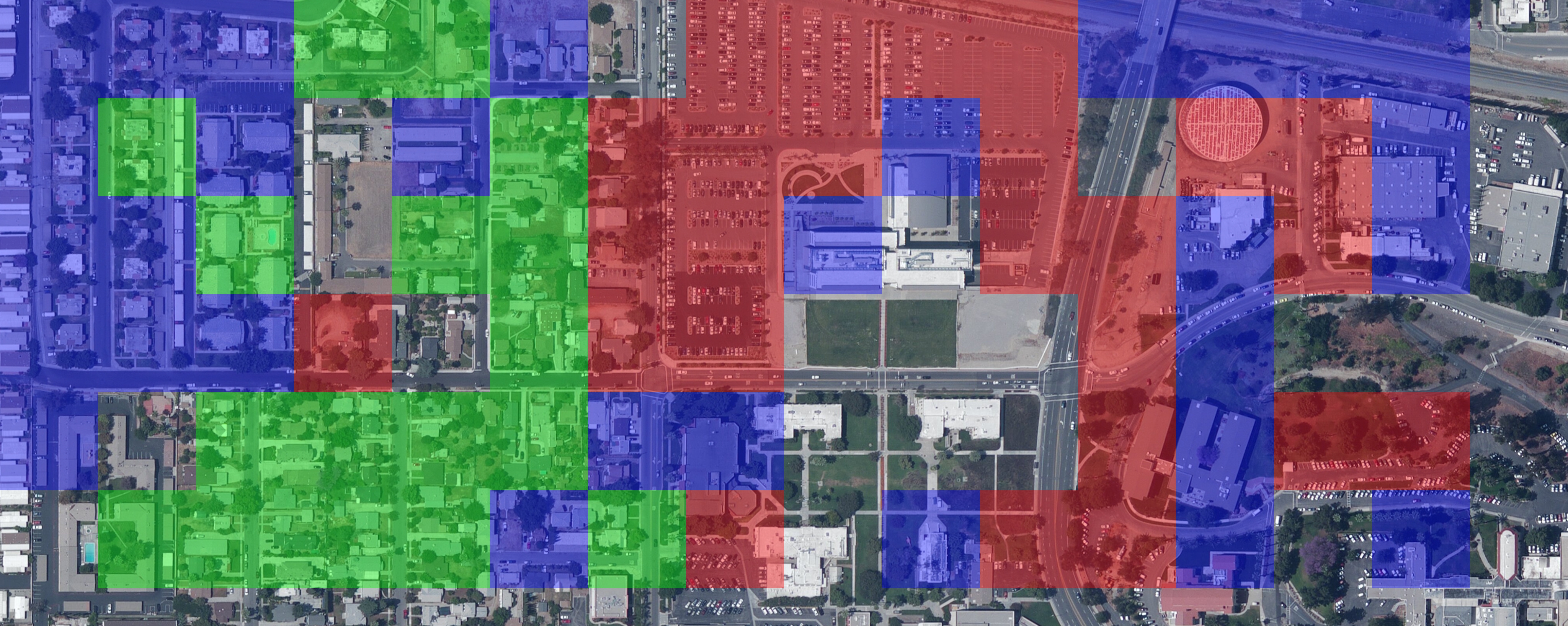


Figure : Sample Output

## Terminology

|  |  |
| --- | --- |
| Convolutional Neural Network or CNN. | A class of deep artificial neural network that is commonly used to analyze visual imagery |
| Tensorflow | Open source machine learning and deep learning library developed by google. |
| Keras | Open source deep neural network library in python that can run on top of Tensorflow. |
| Model | A neural network that takes in input and gives some output. In this project, this inputs are images and the outputs are probabilities of the image belonging to a class. |
| Datasets | A collection of images of uniform size used to train the Neural network model. |
| Jupyter Notebook | Editor that enables uses to author notebooks which include live code, interactive wedges, images and narrative text. This project was largely developed in Jupyter notebooks. |
| Anaconda | Open source distribution of python for scientific computing with build-in packet management capabilities. This project uses Anaconda distribution 4.4.0 |
| Numpy | High performant python library for dealing with large multi-dimensional arrays and matrices. |
| Neural Networks | Neural networks are Universal functions approximators. With enough training and given data, they can approximate any function very well. |
| Class | A category of the dataset. For example, a 5-class dataset for aerial imagery will be [roads, highways, buildings, oceans, forests. |
| Classification | The process of classifying an image by a model as belonging to a category. |
| Layers | A neural network is comprised of several layers. Layers are made up of interconnected nodes |
| Training | Programming a model to learn patterns from a dataset |
| Image Augmentation | Creates training images through different ways of processing or combination of multiple processing, such as random rotation, shifts, shear and flips, etc. |

Figure Commonly used ML terms

## Install and Setup

This project uses ArcGIS API for Python which is distributed via Conda. Conda is a popular package and environment manager application that helps you install and update packages such as the ArcGIS Python API, Tensorflow. Keras etc. All required packages and their dependencies can be installed via Conda

1. Install the latest version of [Anaconda for Python](https://www.continuum.io/downloads) (for Python 3.x) .
2. Open conda in command prompt(This step depends on the installation process).
3. Install the following dependencies using the anaconda package manager. The format to install any anaconda package is conda install -c <channel> <package\_spec>
   1. [Jupyter](conda%20install%20-c%20conda-forge%20jupyter)
   2. [Tensorflow-gpu\*](https://anaconda.org/anaconda/tensorflow-gpu)
   3. [Keras](https://anaconda.org/conda-forge/keras)
   4. [Pillow](https://anaconda.org/conda-forge/pillow)
   5. [ArcGIS](https://anaconda.org/esri/arcgis)

It’s also possible to setup Keras using Tensorflow(CPU only version), but its highly recommended to setup the GPU version which is up to 10x faster than the GPU only version.

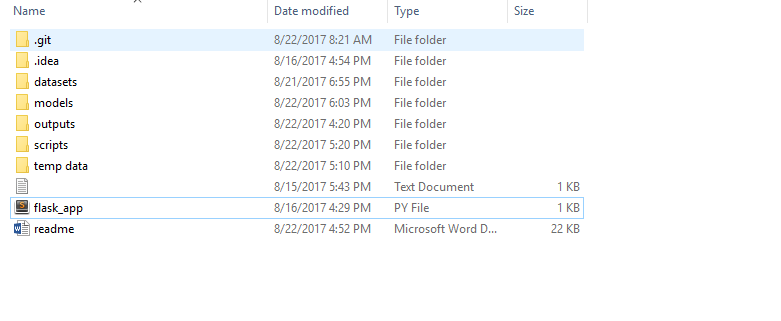


Figure : Listing of all the directories

## Running a Jupyter Notebook

1. In the command prompt (under anaconda), navigate to the project directory
2. Type <jupyter notebook> to open the jupyter notebook in your default browser
3. In the Jupyter Notebook, you can navigate the folder structure and click on a sample notebook. This opens the notebook in a new tab or window.
4. Run each cell by using **ctrl + enter** or **shift + enter**

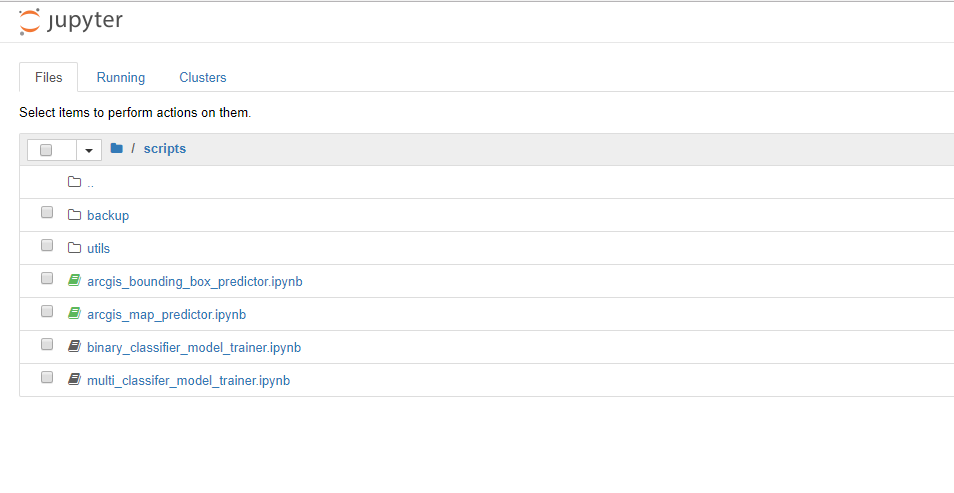
****

Figure 5: Jupyter Notebook running on chrome

## Datasets

This directory contains the datasets that were used to train the convolutional neural network. Each dataset contains subfolders that indicate the categories that were used to train the model.

**Located at:** Esri Project/datasets

1. **Sample Map Dataset:** This dataset was collected from <http://vision.ucmerced.edu/datasets/landuse.html>. The dataset is divided into 21 classes, each class containing **100** images. All images are size **256 \* 256** in **jpeg** format.
2. **Redlands map Dataset:** This dataset collected manually from <http://services.arcgisonline.com/arcgis/rest/services/ESRI_Imagery_World_2D/MapServer> , from the area of Redlands. The dataset is divided into 7 classes
   1. **Agricultural:** 270 images
   2. **Buildings:** 230 images
   3. **Land:** 312 images
   4. **Others:** 2541 images
      1. This class contains images from all classes except roads and highways. This class was created for training a binary classification model to identify roads in the map.
   5. **Parking:** 290 images.
   6. **Residential:** 527 images
   7. **Roads:** 1570 images
      1. This class contains a large number of images because it was used to train a binary classification model to identify roads in the map.

## Temp data

**Located at:** Esri Project/temp data

This folder contains three temporary directories

1. **Training data set:** This directory contains 60-70% of the number of image from a class.
2. **Validation data set:** The directory remaining 30% of the images from a class
3. **Test data set:** This directory contains the same images as the validation directory.

## Models

This directory contains various models that were trained during this project. The models are saved as **hdf5** which is a file format, which is designed for high volume and complex data types and the default format used by **Keras** for saving models.

This directory generally contains two types of models

1. **road\_binary\_classifer\_model:** This is an example of a binary classification model. Given a map tile, this model will output.
2. **redlands\_multi\_classifier\_model:** This model will output an array of size 5(the number of classes in a multi model classification problem). Each index of the array indicates the probability that given image belongs to the class. The classes are in the order ['agricultural', 'buildings', 'land', 'parking', 'residential', 'roads']

## Scripts

This directory contains the all the python scripts and Jupyter notebook that were developed during the project.

**Scripts for training model:**

1. **Multiclass model trainer:** This script is for training a multi-class classification CNN from the given dataset using Keras + Tensorflow stack.
   1. **Input:** Directory of images, arranged into more than two classes. (in datasets folder).
   2. **Output:** A h5 file containing the trained model (saved in models folder)
2. **Binary model trainer:** This script is for training a binary classification model CNN from the given dataset using Keras + Tensorflow stack.
   1. **Input:** Directory of images, arranged into two classes. (in datasets folder).
   2. **Output:** A h5 file containing the trained model (saved in models folder)

**Scripts for testing models:**

1. **arcgis map predictor:** This script will run a given model on a given extent map, selected from the map, tile by tile and output an jpeg image showing the prediction.
   1. **Input:** Use **the arcgis map widget** to point to the extent that needs to analyzed.
   2. **Output:** Jpeg image marking the classes is saved in the **output** folder.

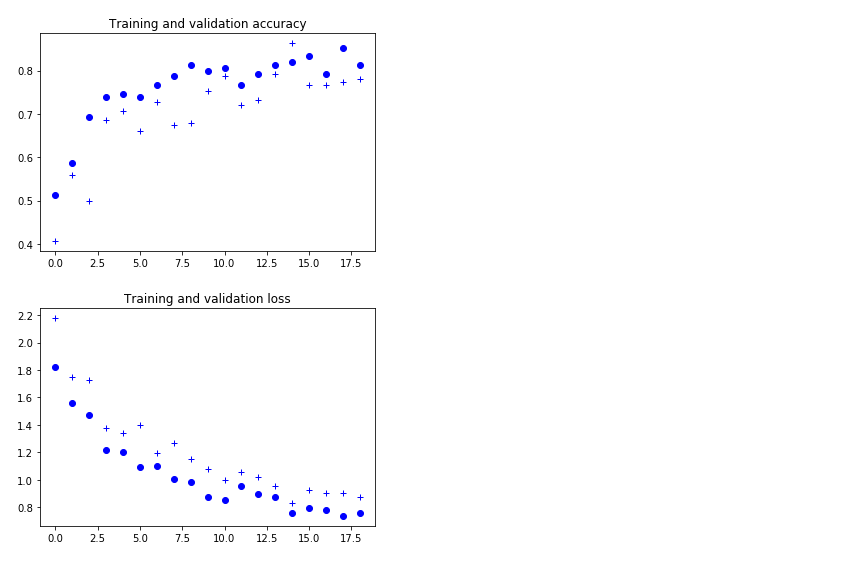
****

Figure 6: accuracy graph for Training and Validation set

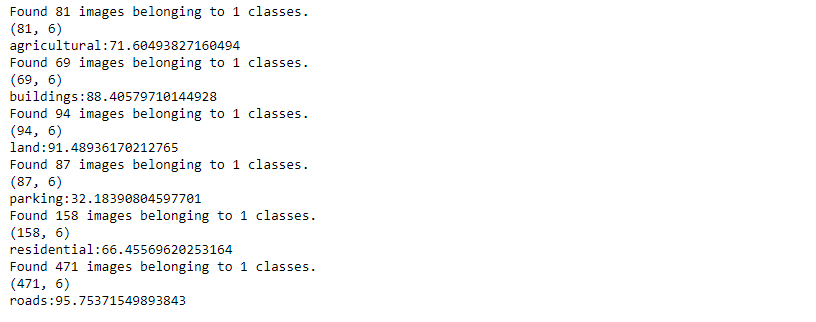
****

Figure 7: Showing accuracy per class