Final Report for ECS289N Deep Learning

Satellite Image Object Recognition

Weizhuo Xiong 914443957
Shu Zhang 914472164

Abstract

This report describes our approach to the kaggele competition: Defense Science and Technology Laboratory (DSTL) Satellite Imagery Feature Detection challenge. The goal of this challenge is to accurately recognize and classify different object in satellite images into correct classess(crops, roads, etc.). In this project, we used fully connected convolutional neural network(CNN) and U-net structure. And we introduced several modifications to the training object and whole pipeline to imporve the effeciency and accuracy. We furthered our understanding of image analysis and deep learning in this project. Moreover, we gained experience of dealing with real-world data and solving practical deep learning problems, which is valuable and inspiring. Also our method might bring innovation to computer vision methodologies applied to satellite imagery.

Keywords : satellite imagery , image classification, U-net

1 Introduction

Image classification and object recognition is always a vital and important topic in the field of computer vision. And the rapid development of deep learning method make it possible to deal with large quantity of image data efficiently. As we know, the most widely used method for this kind of problem is deep convolutional neural network(CNN). Satellite image is an important kind of images with its own unique structure. And to apply CNN into satellite image dataset, we always need to do some modification.

This project origins from the Kaggle competition Dstl Satellite Imagery Feature Detection. Our dataset is about 15G and contains 25 images for training and 450 images for prediction. In training set, we has labeled class of each image pixel and we have 10 different classes of object label, and here is one example of the labeled training images.

As we can see, we have 10 different classes of objects.

1. Buildings: large buildings, residential, non-residential, fuel storage facilities, fortified building;
2. Misc. man-made structures;
3. Roads;
4. Track - poor/dirt/cart tracks, footpaths/trails;
5. Trees - woodland, hedgerows, groups of trees, stand-alone trees;
6. Crops - contour ploughing/cropland, grain (wheat) crops, row (potatoes) crops;
7. Running water;
8. Standing water;
9. Vehicle (Large) - large vehicle (e.g. lorry, truck, bus), logistics vehicle;
10. Vehicle (Small) - small vehicle (car, van), motorbike.

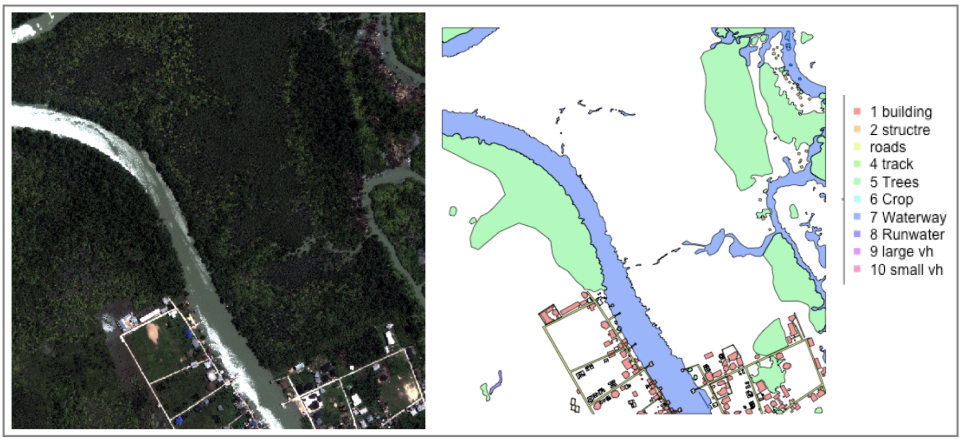

Figure 1: One example of labeled training set pictures

Our goal is to detect these kinds of objects in the prediction satellite images and assign them correct labels. To do so, we actually tried to extract suitable features from images and classify every pixel into right class.

As we mentioned earlier, the simple CNN is not perfectly suitable for this project. So we used U-net CNN structure and do some adaptation in the implementation to improve our results.The details of implementation is closely related to the image property so we would go through them in the later section.

2 Methods

2.1 Data exploratory analysis

Unlike most RGB or grayscale images, satellite images usually have more than 3 channels and a wider coverage of spectrum. Therefore, it carries more information, but also more difficult to be interpreted properly. In our DSTL dataset, we have 4 types of images of the same area: a high-resolution panchromatic (P), an 8-band image with a lower resolution (M-band), and a long-wave (A-band) that has the lowest resolution. And the RGB image is reconstructed based on these 3 images. The connection of 4 kinds of images and its spectrum coverage is shown in Figure 2.

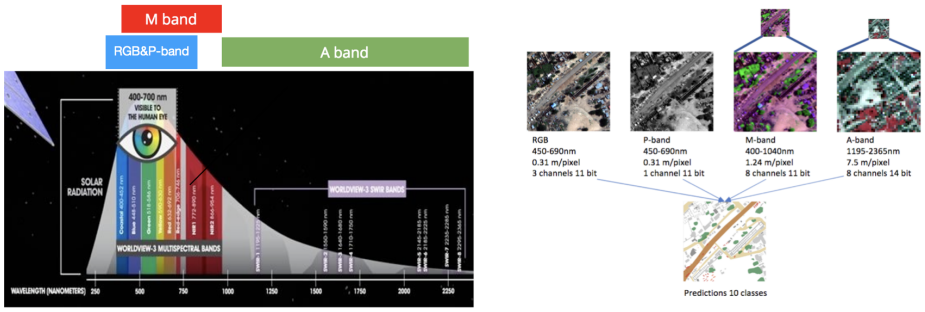

Figure 2: Image composition and spectrum coverage

Also, our images have color depth of 11 and 14-bit instead of a more common 8-bit for every pixel. Another noticeable feature of the images is that image channels(P, M and A band) are taken in slight different time points. So, the same moving object would appear twice in different image band, which would bring us big problems.

2.2 Specific process for certain classes of labels

We have totally 15GB data but only 475 different image, 25 for traning and 450 for prediction. So this is actually a very small dataset and we have relatively insufficient training data(25 images). Also the label distribution of different labels is very unbalanced. As it shown in Figure 3, most of the pixels are labeled as crop and tree both in training set and prediction set. And the vehicles are very rare compared to all other classes.

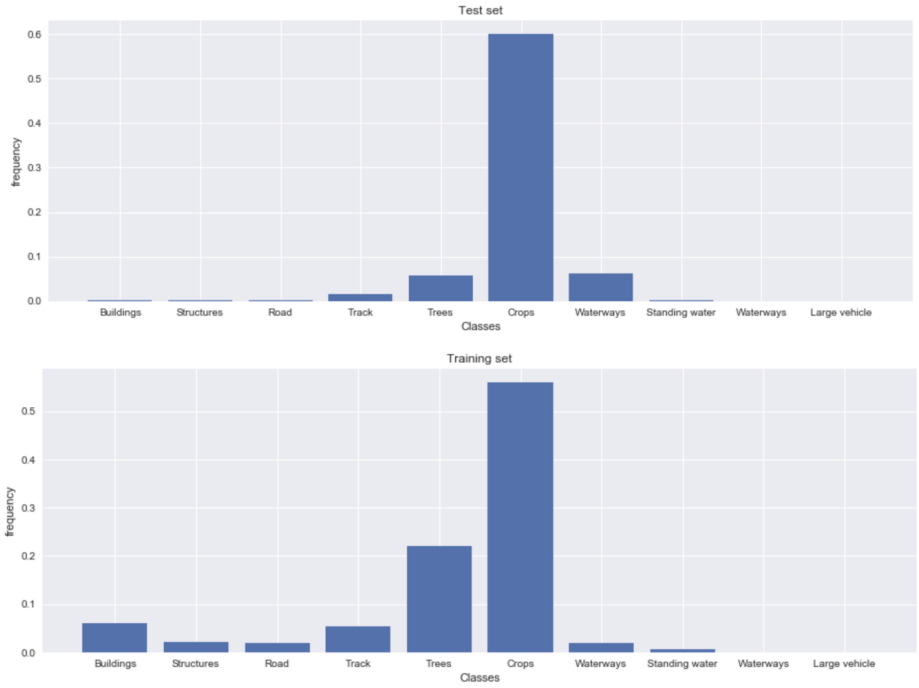

Figure 3: Histogram of pixel numbers in test and prediction set

Moreover, the labeling of objects is poor. Out of 10 labels, the label of vehicle is the most messy one and we can identify that some of them are incorrectly labels by eyes. Some vehicles are actually moving when taking the images, which would create two objects of one single vehicle in one image. That phenomenon makes it hard to do the prediction. Moreover, vehicles actually rarely appear in our dataset. So, we would leave vehicle out when training.

Another problematic label is the water. We have 2 types of water here, one is running water like river and one is standing water like lake. The temperature of water is usually much lower than ground and the structures on the ground. So it can be used as a very useful index to detect water. Also, by using M-band image, we can directly access to the infrared red spectrum which is closely related to temperate. So, we use a simple method to detect water in images.

As it shown in Figure 4, we calculated canopy chlorophyl content index (CCCI), and set boundary value as 0.1, when CCCI above 0.1 we classify it as water, which gives us good result. NIR_1 is near infrared have wavelength 772-890nm, NIR_2 is near infrared have wavelength 866-954nm, and red light has wavelength 620-750 nm.

$$CCCI = \frac{NIR_2 - NIR_1}{NIR_2 + NIR_1} \times \frac{NIR_1 - RED}{NIR_1 + RED}$$

In a word, to make problem easier to solve, we first rule out all the vehicles(both large and small) and water in our target labels when training the convolutional neural

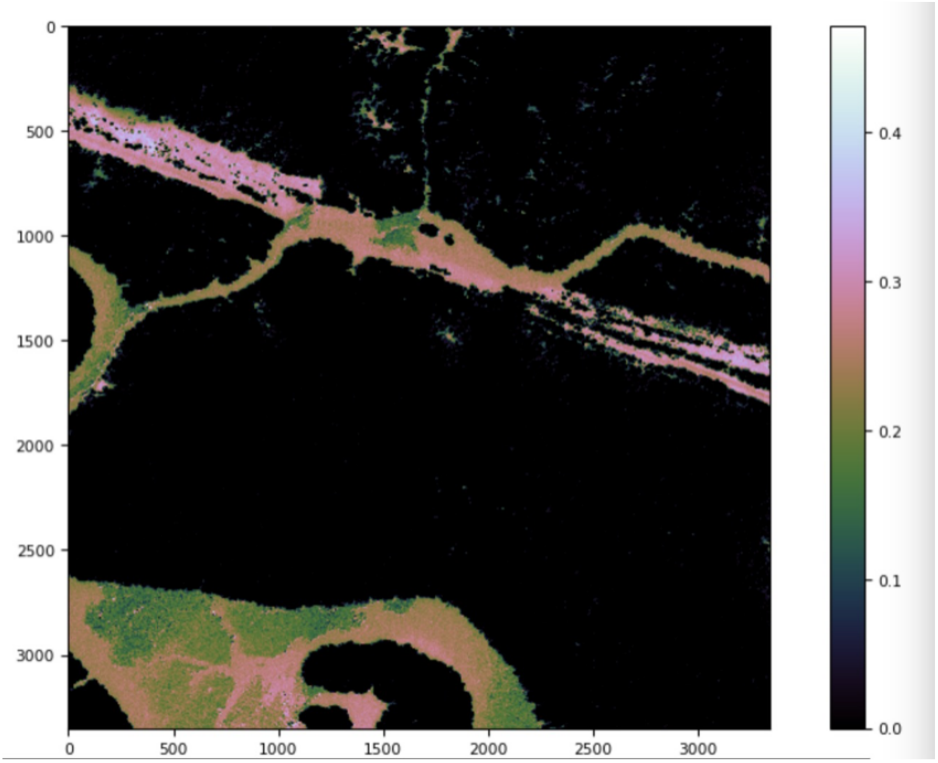

Figure 4: Water detection

network, leaving only first 6 kinds of objects to be recognized using CNN.

2.3 Evaluation

We used Jaccard index to evaluate our classification result. It is also called Intersection-over-Union and can be calculated in this way: $Score = J(A, B) = \frac{A \bigcap B}{A \bigcup B}$. Here A means the true pixels of target label while B means the predicted pixels of the target label.

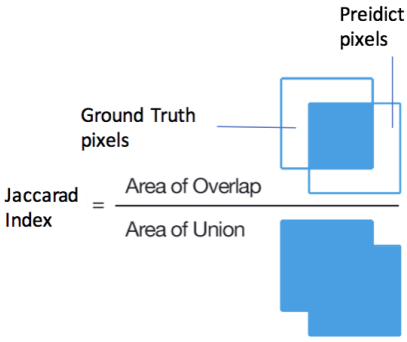

Figure 5: Jaccard coefficient

In order to evaluate the performance on our algorithm of all the labels, we have to calculate Jaccard index of each labels and take the average, which is $Score = \frac{1}{10}\sum_{i=1}^{10} Jaccard_i$ .Overall, the problem can be viewed as a classic supervised image

classificat and object recognition problem with multispectral input image channels and score function with Jaccard index.

2.4 Patches and Input

We want to use as many information as possible in our prediction. So, we tried to include as many channels as possible in input. However, since RGB image is reconstructed by 3 other kinds of images and the resolution of A-band image is too low, we didn't use them. And in practice, we found that the grayscale P-band didn't give us enough information and slowed the train process. Then we decided only use M-band image data as input. In order to make it compatible with the pixel labels in training set, we expand it to the same size of RGB images. On the other way, the training set is too small to extract useful features. So we tried to expand the training set and used patch trick. As it shown in Figure 6-1, We divided images into smaller patches and feed the patches into our model. Through this method, we got quite satisfactory result. Since the size of our patches is important for the performance of our network, we tried several different patch sizes in our training.

2.5 Boundary effect

One big problem for CNN is that when performing convolution of image and kernels, we downsampled the original image and lost the information of corner and boundary. To solve this, people usually use the padding trick to compensate for information loss. However, the common zero padding introduced man-made bias and caused huge error. So, we used reflection padding(shown in Figure 6-2.) instead, which is reflecting the image pixels near boundaries to expand the original image. This method successfully address the global boundary effect problem by transforming boundary pixels into pseudo-internal pixels without causing sudden change of pixels values. The boundary effect problem also exits for the small patches we used. In covolution, the number of ways to get from any input pixel to the central part of the output in a network is much higher than to get the edge ones. As a result, prediction quality is decreasing when it is away from the center. To resolve this problem, we trained with different division of patches and make predictions on overlapping patches, and crop them on the edges.

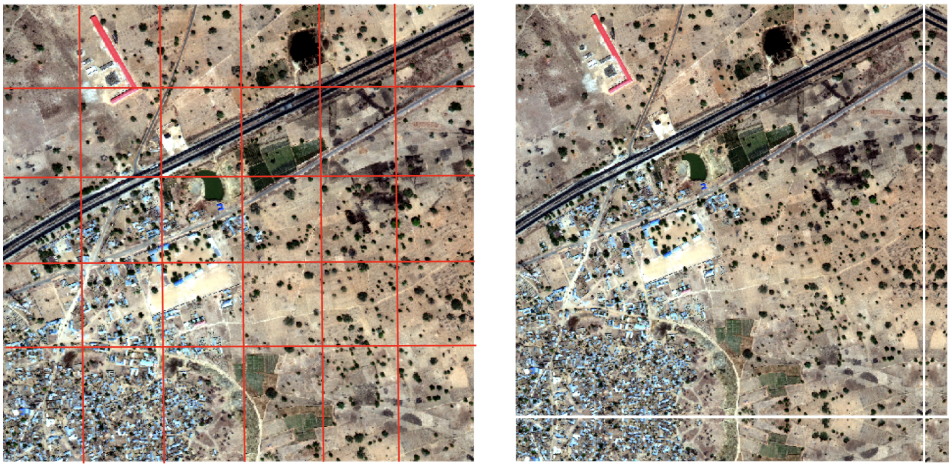

Figure 6: Patches and reflection padding

2.6 Fully connected deep convolutional neural network and U-net

Deep convolutional neural network(CNN) is the most important and widely used method in image classification and analysis. And since we need to predict every pixel in image and get a labeled image as output, we use fully connected CNN, and more specifically U-net structure. U-net structure is one special fully connected CNN architecture which is designed originally for biomedical image segmentation. tion. It allows combining low-level feature maps with higher-level ones, which enables precise localization and classification of each pixel. More importantly, this type of CNN can deal with large quantity of input images and feature channels quite well. So, we chose U-net as our final model in the project. Our network architecture is described in Figure 8.

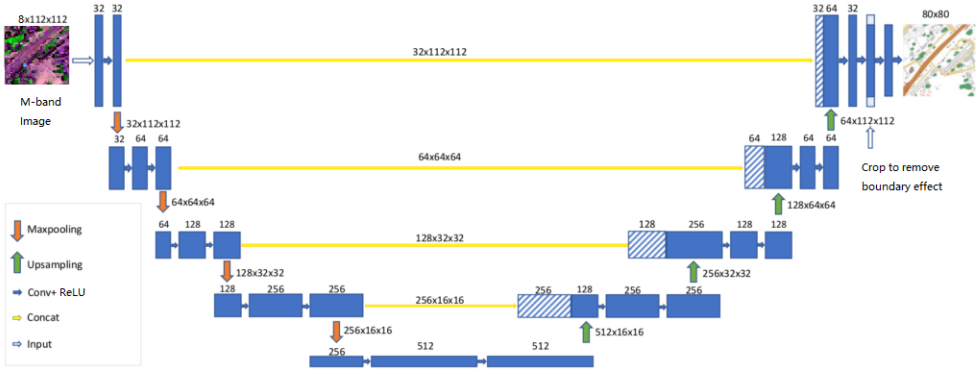

Figure 7: U-net structure

As its name, this architecture has a U shape structure. It consists of decomposition path and and reconstruction path. The decomposition path is also called contracting path follows the typical architecture of convolutional network. In our implementation, we used convolution layer followed by a ReLU(Rectified Linear Unit) non-linear layer. And we did a max-pooling to downsample images to enter into next layer of decomposition path. Finally in the bottom layer we use 1×1 convolution to enter into reconstruction path. Then in the reconstruction path, we upsample each feature map by de-convolution and halves the number of feature channels.Then we repeat the similar convolution, RLU in each layer of reconstruction path accordingly. Most importantly, in each layer of reconstruction path, the corresponding feature map from decompostion path is concatenated to the corresponding feature map of the decomposition path. When training the U-net, we also have to consider some aspects that might influence the performance. As I mentioned earlier, one important aspect is the size of patches. Smaller patches normally means more patches, and that would lead to high accuracy and slow speed. Also, larger patches require more max-pooling layers that reduce the localization accuracy, while small patches allow the network to see only little context. We really need to consider this trade-off in model training. Also, the choices of non-linear unit, convolution kernels, up-sampling and down-sampling process are also important.

3 Results

In conclusion, based on all the methods and adaptation above, such as the adaptation of fully convolutional network to multispectral satellite images , using patches and reflection padding, spectrum feature method for water detection, we successfully trained a U-net structure and get the final result.

The accuracy shown above comes from kaggle system.The public set and private set are both part of the test set(public is 19% test set while private set is the remaining

Table 1: Results for each class in terms of Jaccard coeffecient

Class	Public Set	Private Set
Buildings	0.4017	0.3288
Structure	0.197	0.209
Road	0.388	0.3987
Track	0.1696	0.1596
Trees	0.4365	0.2861
Crops	0.6327	0.6945
Waterway	0.8416	0.9486
Standing water	0.1531	0.4446
Vehicle large	0	0
Vehicle small	0	0

81%). Kaggle divided it in this way to test model more robustly. Also, the Roc Curve of 10 kinds of classes is as following:

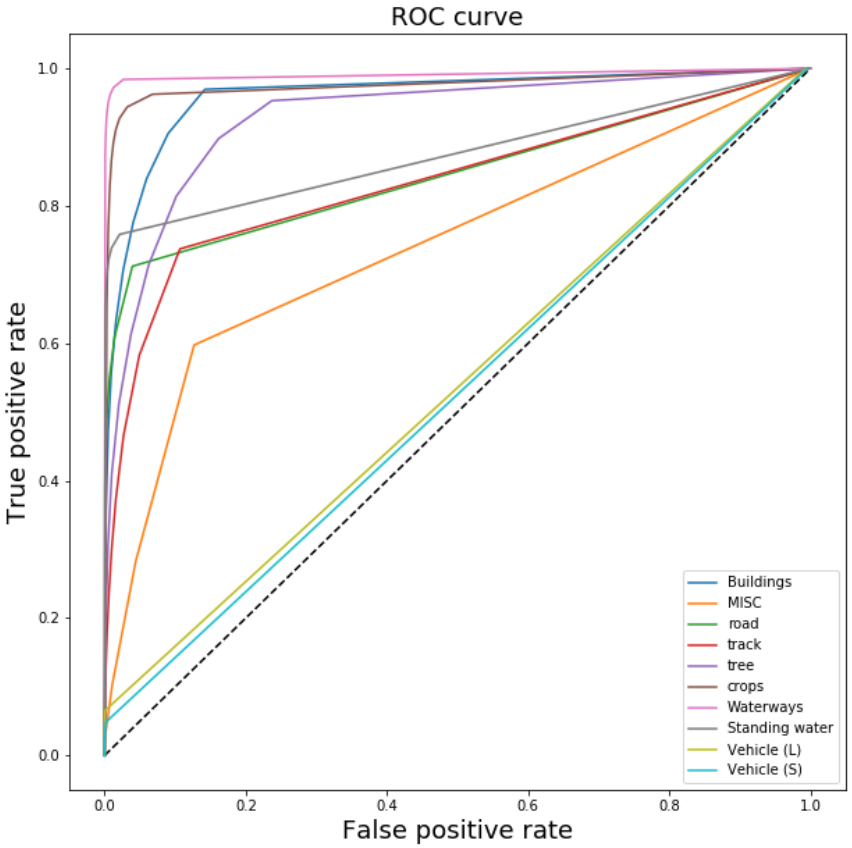

Figure 8: Patches and reflection padding

From the Kaggle public set and private set we can see the jaccrad coefficient for different classes. The answer between two set and reasonable close, which means there is no severe overfitting. For the roc curve, since we lack of train data, their classification accuracy is low and for other objects we have fairly good result. Moreover, the simple method we used for water detection yield good result, suggesting simple method

sometimes can also be useful.

4 Discussion

In this project, we need to use real world data to deal with problems and try to solve it with deep learning method. We analyzed a dataset of satellite image and used U-net CNN to recognize different objects in those images and classified them into right classes. The dataset for this problem is much pretty messy and we spend most of time in the project to deal with data preprocessing.

Since its the kaggle competition, our final result is not as good as some teams in the top ranks of leaderboard. So, there is still much can be done in the future. For example, we trained the CNN for 6 classes together to speed up the training process, but for the imbalance of class distribution, we may consider training 6 classes separately and combine them afterwards. Also, we haven't test enough combination of optimization and activation method, and the introduction of batch normalization together with higher epoch would very possibly increase the accuracy.

5 References

[1] Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. *"U-net: Convolutional networks for biomedical image segmentation."*. International Conference on Medical Image Computing and Computer-Assisted Intervention. Springer, Cham, 2015.

[2] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. *"Imagenet classification with deep convolutional neural networks."* Advances in neural information processing systems. 2012.

[3] Albert, Adrian, Jasleen Kaur, and Marta Gonzalez. *"Using convolutional networks and satellite imagery to identify patterns in urban environments at a large scale."* arXiv preprint arXiv:1704.02965(2017).

[4] Ciresan, Dan, et al. *"Deep neural networks segment neuronal membranes in electron microscopy images."* Advances in neural information processing systems. 2012.

[5] Iglovikov, Vladimir, Sergey Mushinskiy, and Vladimir Osin. *"Satellite Imagery Feature Detection using Deep Convolutional Neural Network: A Kaggle Competition."* arXiv preprint arXiv:1706.06169 (2017).

[6] Cammarano,Davide,etal. *Useofthecanopychloro- phyl content index (CCCI) for remote estimation of wheat nitrogen content in rainfed environments.* Agronomy journal 103.6 (2011): 1597-1603.

[7] Dstl Satellite Imagery Competition, 1st Place Winner's Interview: Kyle Lee
`http://blog.kaggle.com/2017/04/26/dstl-satellite-imagery-competition-1st-place-winners`

[8] Dstl Satellite Imagery Feature Detection
`https://www.kaggle.com/c/ dstl-satellite-imagery-feature-detection`