Road Segmentation from Satellite Images

Shuai Jia (sj39), Zhenwei Feng (zf12)
Rice University

Introduction

The goal of this Kaggle project is to build a model that can segment satellite images to identify roads. We observed that the images have size 512*512*3 and at first we thought that this size is too big to train a model so we downsize the images to 256*256*3. However, after we trained a model we found that the prediction accuracy is not ideal. So we decided to go with the initial-size images because the model can be trained without any loss of information.

Data Augmentation

In order to train a good model, it is crucial to provide enough training data. So we implemented a series of image augmentation strategies as follows:

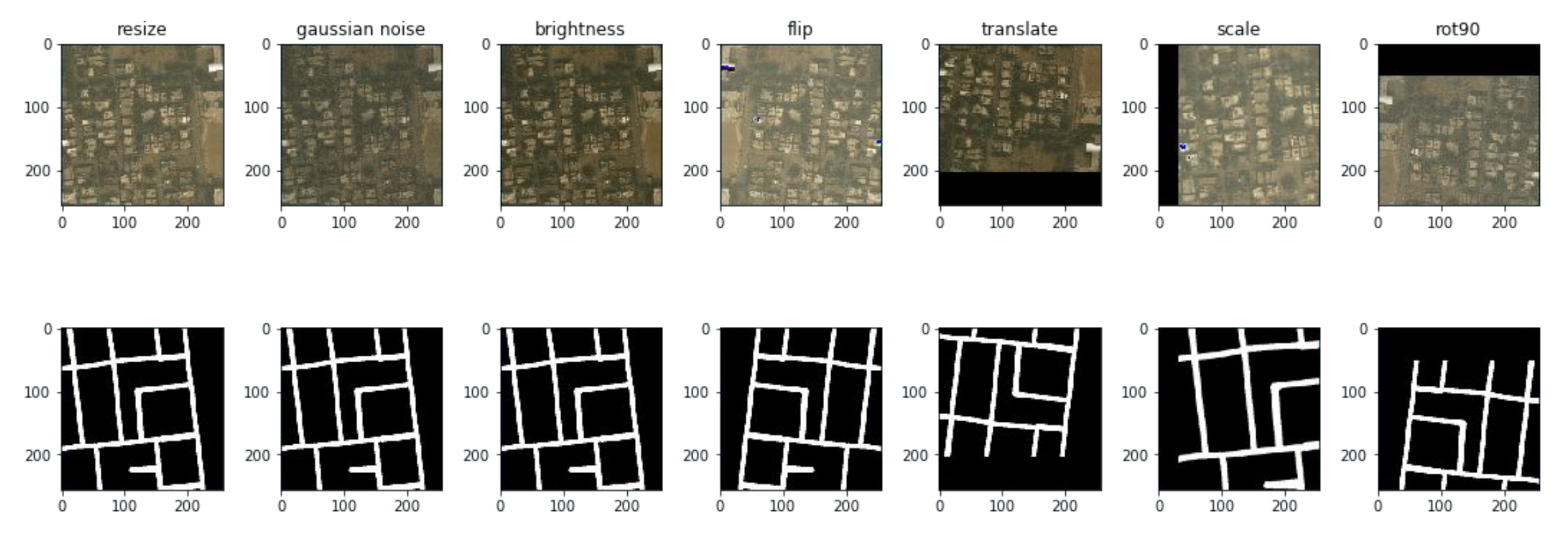

Figure 1. Augmented satellite images and masks

Model Architecture

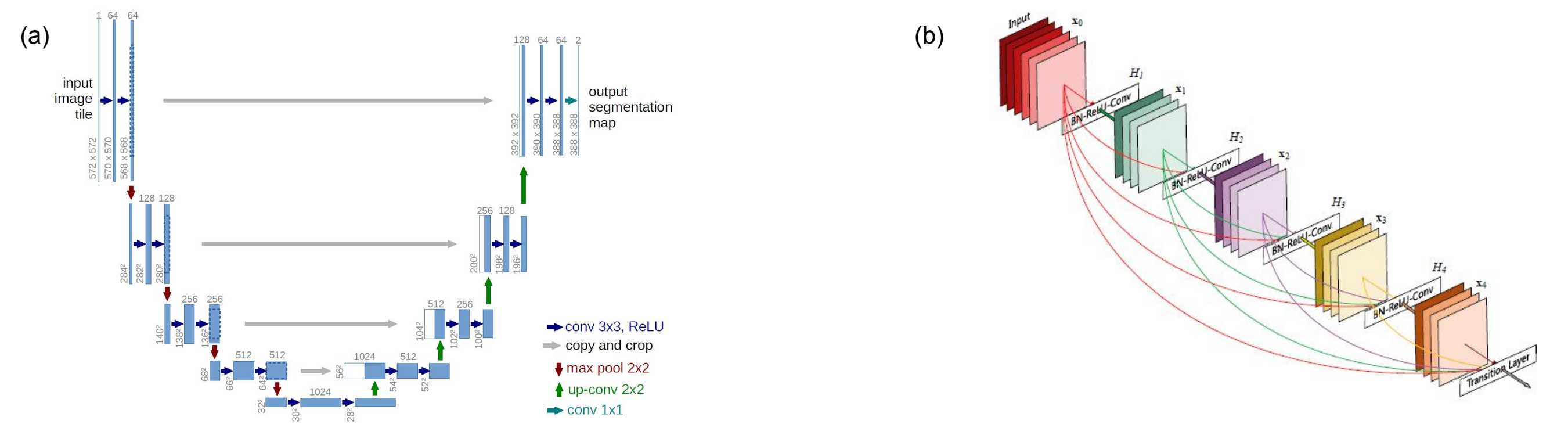

Figure 2. (a) UNet Model architecture. (b) dense block convolution

UNet is a sophisticated model in segmenting images. One of its unique characteristics is that there are a lot of feature channels in the upsampling part, which allow the network to propagate context information to higher resolution layers. Moreover, UNet only makes use of valid part of each convolution so that there are only around 7,000,000 learning parameters in our model, which is much less than a typical fully connected network. That is why UNet is much easier to train.

Here we use a modified unet as our model. In our model, the initial filter number is 16 since only roads need to be recognized. The filters are doubled after every convolution block. More importantly, to extract more abstract information while not increasing parameters significantly, dense dilation convolution block is used in the last encoder layer. The dense block and dilated convolution can make us extract different context information and simplify the training and convergence.

Regularization Techniques

From the dice coefficient curve throughout training, it is obvious that the model is undergoing overfitting, so we added L2 regularization in each UNet layer. Dropout layer is another regularization technique that we used. The more filters that we used in a convolutional layer, higher the dropout rate is. The data augmentation is also helpful to avoid overfitting.

Model Implementation

The model was trained on AWS p2.xlarge with K80 GPU. We use bce_dice_loss as the loss function which is the sum of half of binary cross-entropy and dice loss. The batchsize is 16 if it is not explicitly specified. Adam optimizer is used for accelerated convergence.

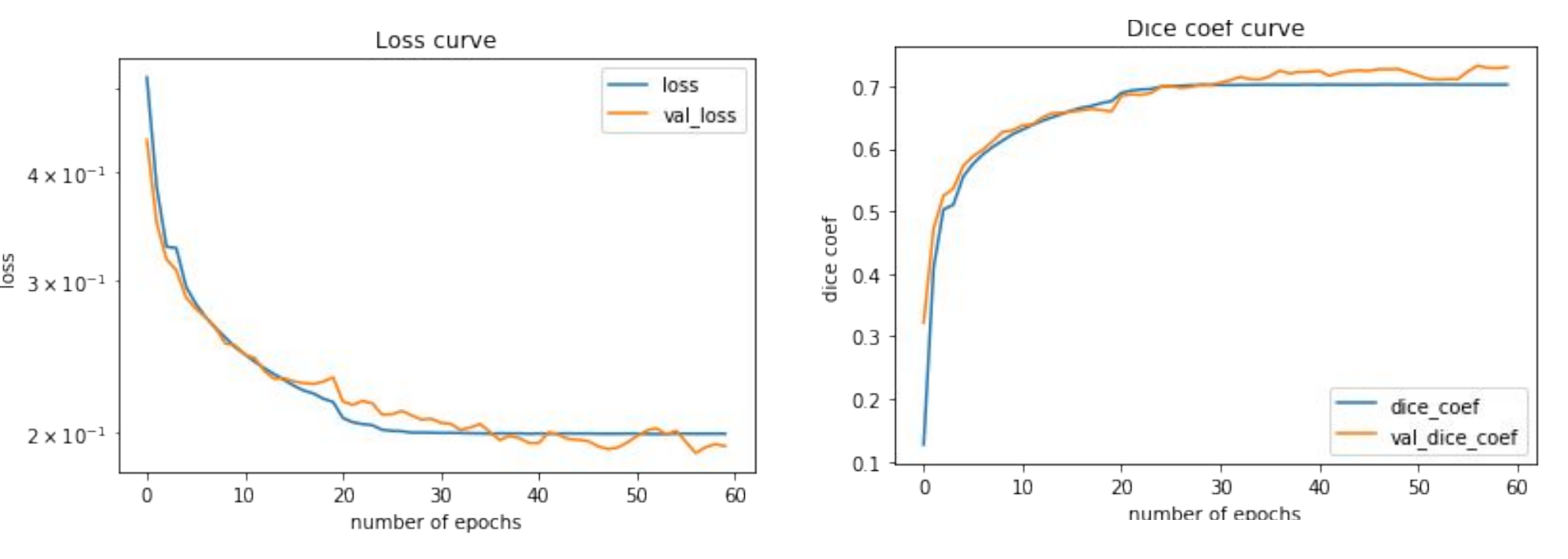

Figure 3. Loss and dice coef of training history

The initial learning rate is e-3, and will decrease by five times if validation loss does not change in three epochs. Early stops will be called if validation loss does not decrease in consecutive 10 epoches.

Model Analysis

It is worthy to note that some images could not be learned well, even for long time training. These images could be roughly divided into five classes, as shown in Figure 4. The first class is that small part of roads lies on the edges. The second class is that some roads are covered by trees, which makes it hard to get a continuous road. The third class is some materials, such as canal, is easily recognized as road because of their similar shape. The fourth class is that some roads have similar color with its surroundings (Figure 4d). The last one is the slight road trence, on farm, or desiret, has similar color and shape with roads (Figure 4e). The first three classes are hard to recognize even for naked eyes. But the last two classes could be learned by model with finer training.

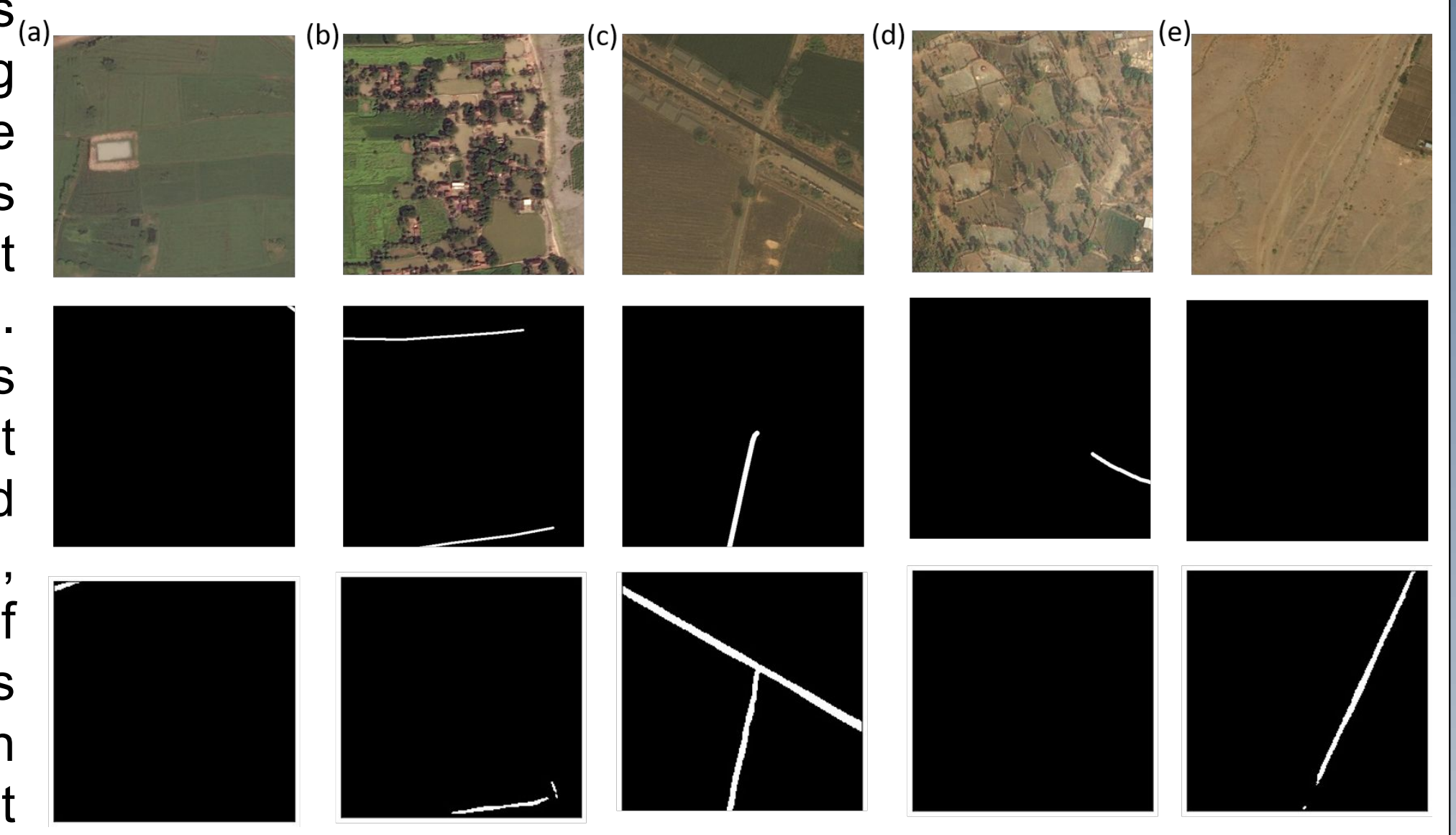

Figure 4. Five classes of ‘outlier’ images that model could not learn very well. Satellite images, true mask and predicted mask for each row.

Fine Tuning

The loss (bce_dice_loss) for image without road has a significantly larger loss than image with road, since their intersection is zero. This means it is very dangerous to predict a image with some roads while there is no road on true mask. This might be one of the reasons the model could not learn well on some images without any obvious roads, such as the images in Figure 4d and 4e.

To make models learn better on the images without any roads, two approaches are used here. The first one is to fine tune a model which is trained with a large batchsize (16) to ensure convergence. Thus we reduce batchsize to train the model and force it to learn well on images without any roads. For example, when the batchsize reduces to 4, the test dice coef increase to 0.72 from 0.68 (for batchsize=16). The detailed model performance is listed in the following table 1. The second approach is to penalize images without roads. After we replaced bce_dice_loss with dice_loss, It is clear to see that the model test score was improved from 0.732 to 0.738.

Table 1: batchsize fine tuning

batchsize	test dice coef
16	0.682
4	0.725
2	0.729
1	0.732

Table 2: loss function fine tuning

loss function	test dice coef
bce_dice_loss	0.732
soft_dice_loss	0.738

Model Predictions:

The fine tuning makes model capable of predicting images without any roads, and also preserve the initial good capability for road predicting. For example, some slight road trances are shown in fourth image of Figure 5. However, the model could successfully recognize these as roads trance, not real roads, and the predicted mask and true mask are exactly the same. This gives model better performance for all images. The final test dice score is around 0.739.

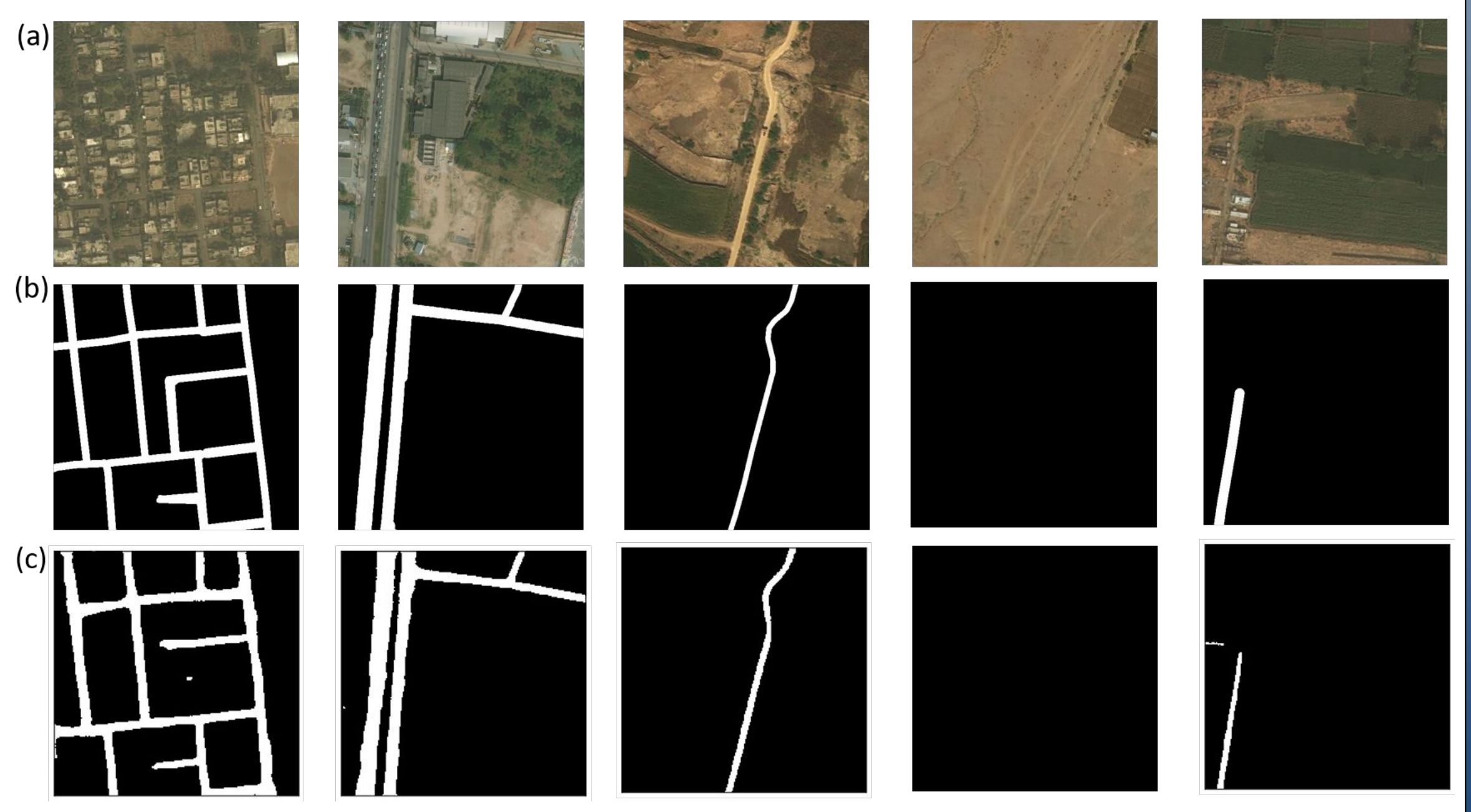

Figure 5.Satellite images (rows a) and ground truth masks(rows b) and predicted masks (rows c).

Custom approach - dense dilation block

After we apply a couple of convolutional layers with increasing number of filters, the initial image has been downsize to 32*32. However, this still looks a little too big as a starting point of the upsampling process. If we add another layer of convolution, the parameters to be learned would blow up. To solve this problem, we used a relatively new technique called dilation. Specifically,it introduces gaps into it's kernels so that it can cover a larger section of the image while still only have as many weights as the standard form. Furthermore, to extract information from different context, different dilation rate (1,2,4,8) convolution is used for the last layer in Unet. And these dilated convolution will be added to upsampling.

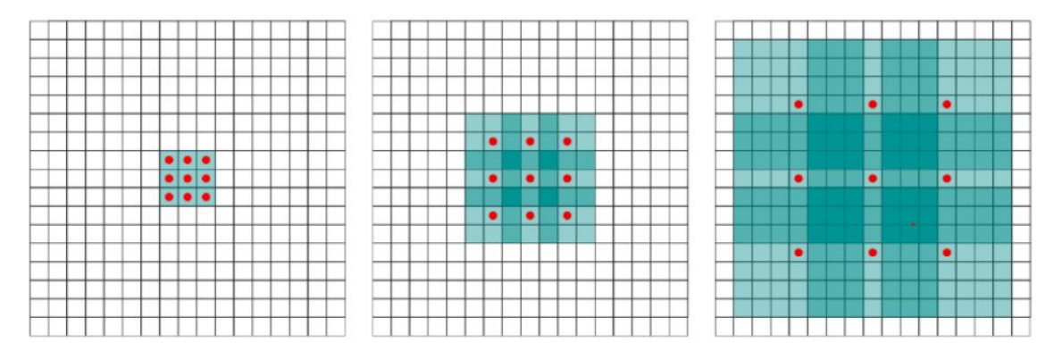

Figure 6. Dilation illustration

Submission Timeline

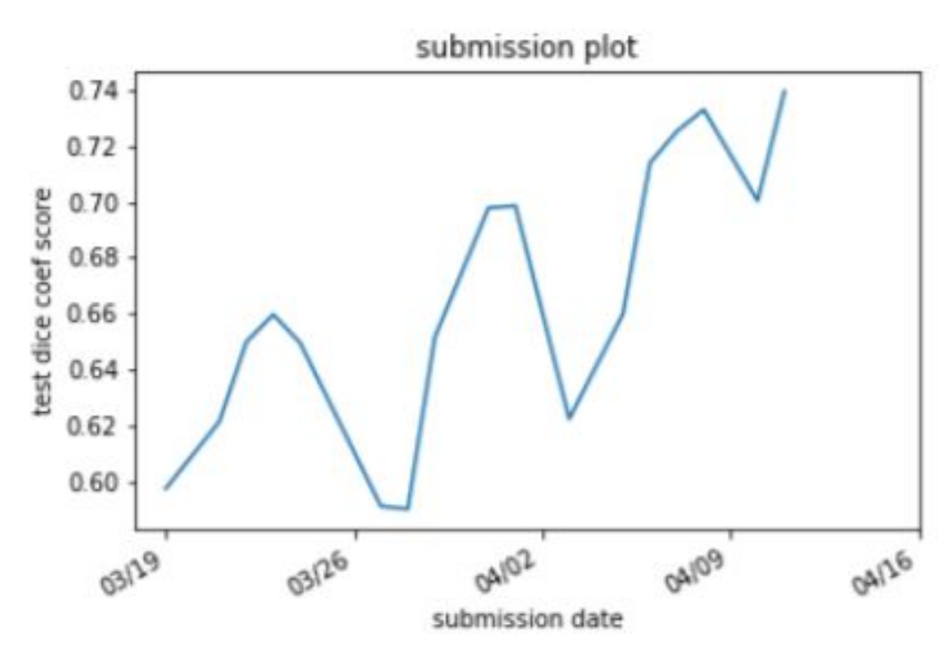

Figure 7. Dice coefficient of testing VS. date

Conclusions

We learned a lot of things from this Kaggle project:

1. How to use cloud computing services like AWS and Google Colab.
2. How to use Tensorflow and Keras to implement deep learning model.
3. How to do data analysis whenever we are given a practical problem.
4. How to look for high-quality models by looking into all kinds of resources like academic papers and data science posts.
5. How to detect defects in the model and improve them with various techniques.

Next steps

There is still a lot of room for our model to improve. From our observation on the training data, there are a lot of images that have no roads in it at all, whose dice loss is around 1. Those images will prevent our model from converging. Our idea to solve this problem is that first we train a binary classifier that identify if there is road in the image. Then we will only use those images with road to train our model. We believe that it will take much faster for out model to converge.

Secondly, it occurred to us that we may have used too many filters to identify the road structure in the images. Too many filters yield too many parameters, which have the potential to overfit. We would try to use 8 filters in the first convolutional layer so that the total number of filters would reduce considerably, which can make our model much easier to train.

References

1. Guillaume Chhor, Cristian Bartolome Aramburu, Ianis Bougal-lambert, 'Satellite Image Segmentation for building Detection using U-net'
2. Gao Huang, Zhuang Liu, Laurens van der Matten, 'Densely Connected Covolutional Networks', arXiv:1608.06993v5
3. Liang Chieh Chen, George Papandreou, Iasonas Kokkinos,Kevin Murphy, and Alan L. Yuille, 'DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs', arXiv:1606:00915v2
4. Liang-Chieh Chen, George Papandreou, Florian Schroff, Hartwig Adam, 'Rethinking Atrous Convolution for Semantic Image Segmentation',arXiv:1706.05587v3
5. Olaf Ronneberger, Philipp Fischer, and Thomas Brox, 'U-Net: Convolutional Networks for Biomedical Image Segmentation',arXiv:1505.04597v1
6. https://www.kaggle.com/abhmul/python-image-generator-tutorial