Semantic Segmentation with U-Net for Land Cover Mapping

In this exercise we will apply U-Net model on satellite images to map land cover.
The goal here is to evaluate the potential of the U-Net in developing detailed land use/land cover (LULC) classification in Alberta with ultimate production of forest fuels for our study area. Because the U-net summarizes both spectral and spatial features to perform the LULC classification, we expect that it will obtain higher accuracy than the traditional classification algorithms, which only uses spectral features.

Input data:

a) Features is a six-band stack of Sentinel-2 (2-8-9 bands) and Alos-2 dual polarization (HH,HV,HH/HV ratio)satellite images.
b) Labels is provincial Alberta Satellite Land Classification (ASLC)

There are twenty unique land cover types labels one band mask as below, -
Pixel value 0 - burned area clas
Pixel value 1 – Black Spruce
****
Pixel value 19 - Young regenerated mixedwood

Tiling the image

Before we come to model training stage, we need to prepare our data to feed the model. We need to split images into patches of predefined size. To do that , we first find out what is the maximum size of our features divisible with patch size. If our image is 1000 by 1000 and our patch size 256 by 256 , it means that there will be 3 patches along columns and 3 patches along rows. In total there will be 9 patches. After that we crop an image by maximum size divisible by patch size which is 3 in this case. Our cropped image's size will now be 768 by 768, instead of 1000 by 1000 pixels. Cropping images allows us to create more data for our model by breaking large images into small patches, and it also gives better predictions.
So we will break the images and masks as well into tiles. Tiles may be stored as tifs or kept in memory. To create tiles we use Python "patchify" library.

Every mask patch has to accompany the corresponding feature image patch when training the model. So, it is important to visualize random tiles of features (images) and labels (masks) (figure 1).

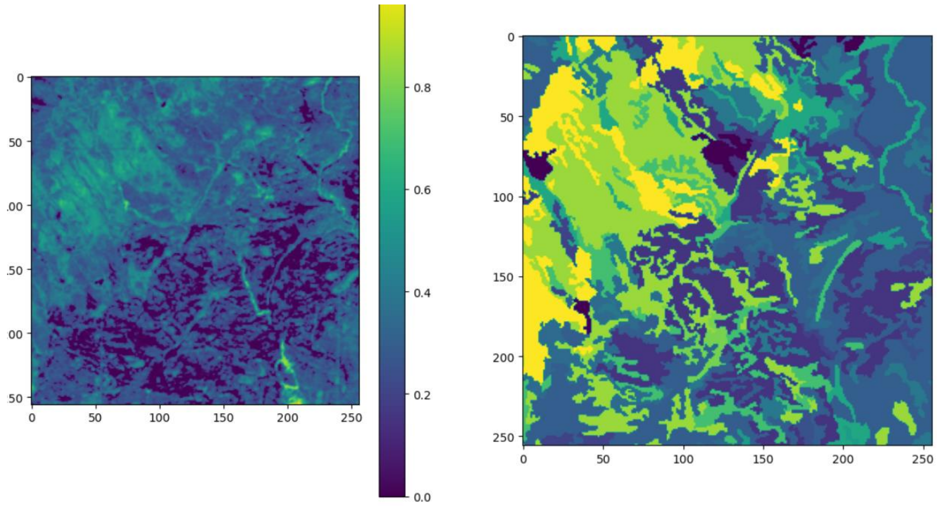


Figure 1. On the left is tiled image (Sentin2/Alos2) and on the right is tiled label (ASLC).

In the next step we did Hot encoding where we converted our classes (labels) to categorical.
After patching features and labels, then we split data into training an testing dataset. Here we reshape data using ratio 85:15% and define X_train, X_test, y_train, y_test to get data ready for training.

Before we start training model, we need to scale the data. Here we opted for MinMaxScaler().

Once we have completed previous tasks, we can train U-net model.

NPredicted Output image is land cover type..

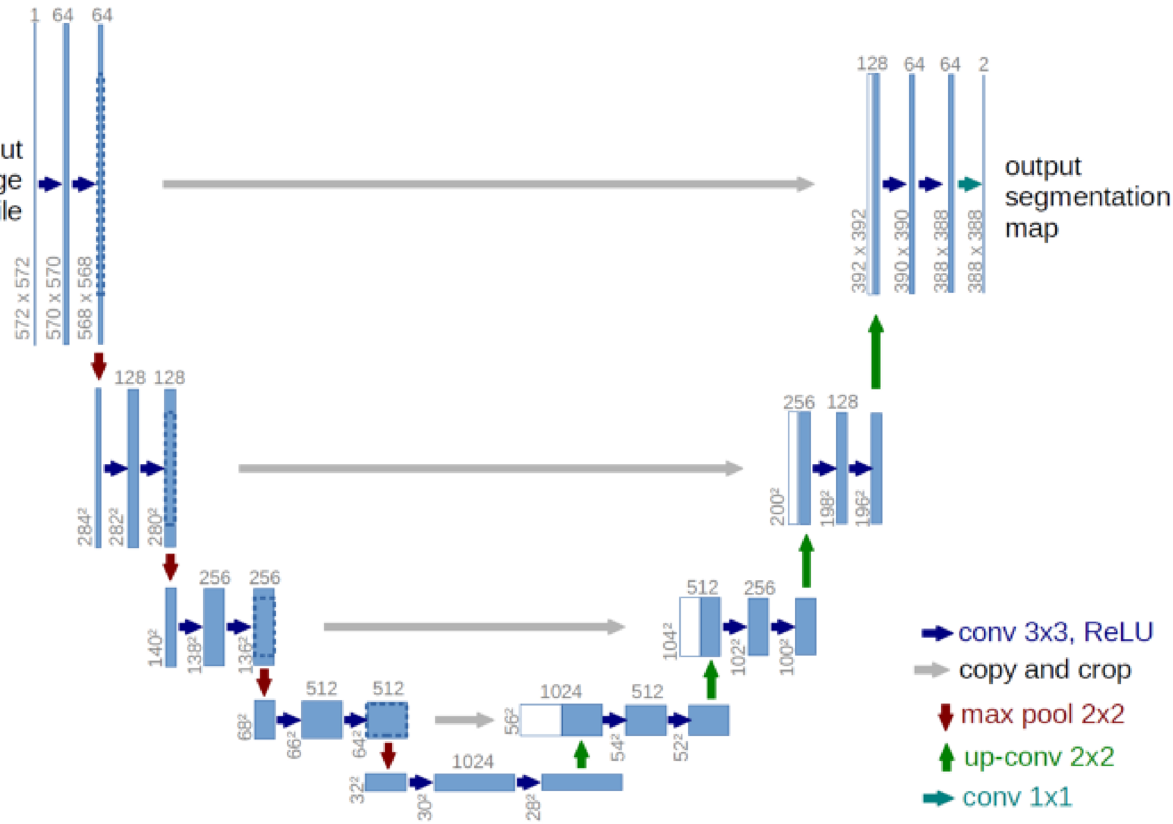


Figure 1 below. U-net architecture. Blue boxes represent multi-channel feature maps, while boxes represent copied feature maps. The arrows of different colors represent different operations

Diagnostics

We used IoU (Intersection over Union) as our Evaluating metric.
The Intersection over Union (IoU) metric, also referred to as the Jaccard index, This metric is often used with the dice coefficient, which is a loss function.

Loss Function

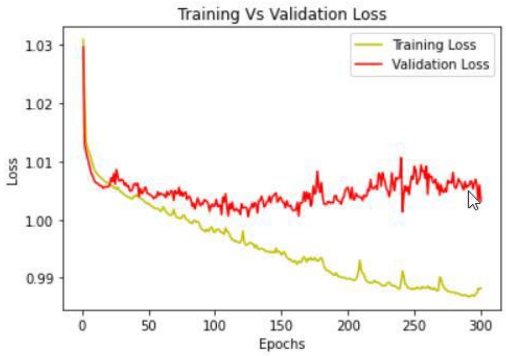
The loss function that we used was focal loss. So, a focal loss is an extension of cross-entropy loss. It improves the minority class classification, reduces the weight of easily clasified objects and mainly focuses on hard clasifiable objects. By default, it reduces the weights of well-clasified objects which have a probability greater than or equal to 0.5 and increases the weights of objects having a probability of less than 0.5.

Training Of Model

The training of this model has been done on Google Colab GPU, which works fast but with a limited amount of data and may be really frustrating relatively costly.

Visualising Loss And Accuracy

Some diagnostics in terms of loss and model accuracy can be assessed after fitting the model.



Predictions

If we are satisfied with the model we may approach to deploying the model on to some data that model has not seen it yet.

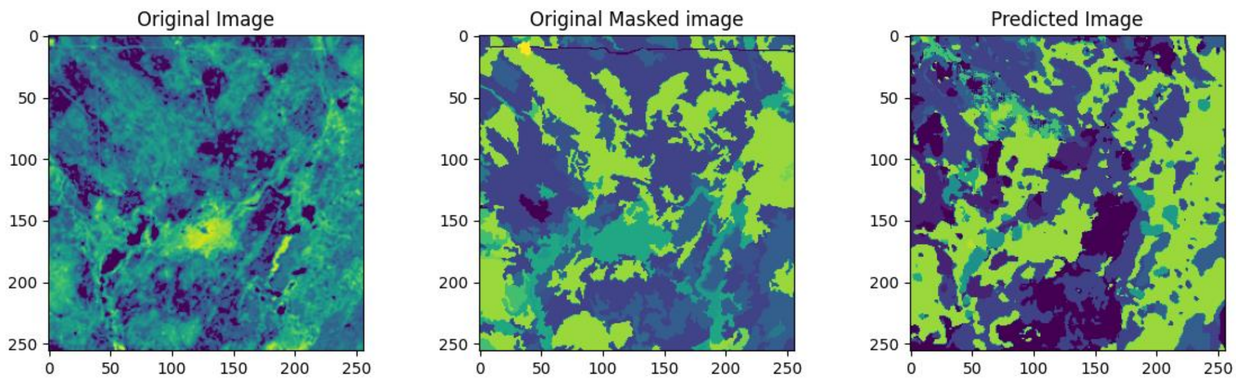


Figure 3. Using saved model to do prediction on test data.

<u>Results</u>

The most accurate results were obtained with the Sentinel-2 + Alos-2 dataset, where the highest overall accuracy (0.78) and Jackard coefficient of 59 were achieved at 300 epochs and batch size of 16.