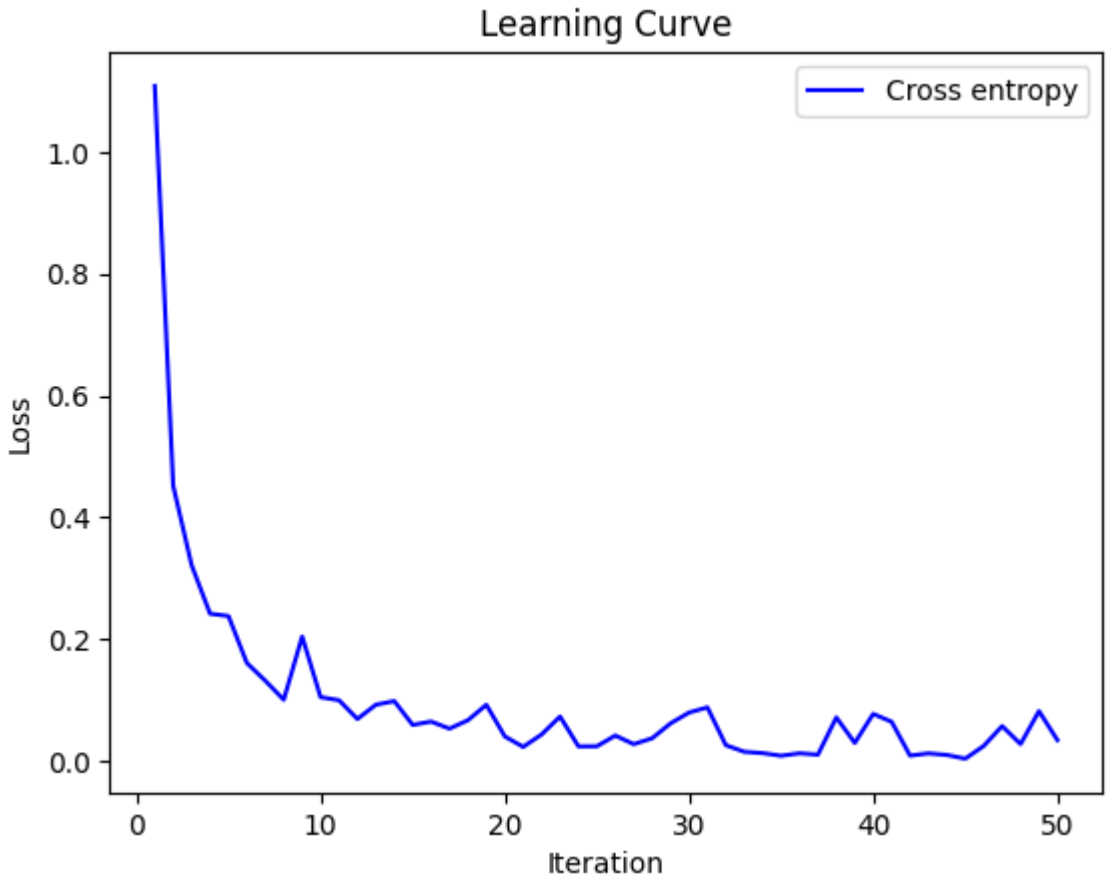
Learning Curve
Cross entropy
Loss
1.0
0.8
0.6
0.4
0.2
0.0
0
10
20
30
40
50
Iteration


Training Accuracy
Accuracy rate
1.0
0.9
0.8
0.7
0.6
Training Accuracy
Validation Accuracy
0
10
20
30
40
50
Iteration

```
## evaluate loss value of test dataset ##

# model.load_weights(ckpt_folder)
test_loss = model.evaluate(test_input, test_output)
print("test loss:", test_loss)

## end of evaluate loss value of test dataset ##
```

```
40/40 [==============================] - 1s 10ms/step - loss: 1.3314 - accuracy: 0.8445
test loss: [1.3314197063446045, 0.844531238079071]
```

—

rx_size改成8

tx_size改成3

```
rx_size = 8; % 可調，決定你要幾個rx的資料
tx_size = 3; % 可調，決定你要幾個tx的資料
cov_size = rx_size*tx_size; % covariance的大小為(tx*rx, tx*rx, 2)
Covariance = zeros(total_train_size,cov_size,cov_size,2);
Label = zeros(total_train_size,dir_idx_max);
%% load your train & test index
idxfolder = "student_train_test_idx/";
idxfilename = idxfolder + '109611036'; %%%%%%% idxfolder + 'YOUR ID'
```

Input_sqr改成24

```
learning_rate = 1e-4
Input_sqr = 24 ## Change this when you chane the size of covariance.
channel = 2
n_labels = 5 ## change it to number of area
keep_prob = 0.5

num_epochs = 50 ## number of epoch
num_batches = 64 ## size of batch, depend on your GPU memory
```

架構:

```
model = models.Sequential([
    layers.Conv2D(64, (2, 2), input_shape=(Input_sqr, Input_sqr, channel)),
    layers.BatchNormalization(),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64, (2, 2)),
    layers.BatchNormalization(),
    layers.MaxPooling2D((2, 2)),
    layers.Flatten(),
    layers.Dense(256),
    layers.Dropout(0.5),
    layers.Dense(400),
    layers.Dense(200),
    layers.Dense(n_labels, activation='softmax')
])
```