PinPlace: CNN based location image search And its adaptation to social network

TEAM H **Week 11**

CNN Build

Check confusion matrix

CHE SEUNG YUN

Modify image dataset

HONG SEONGJUN

Front end

Work on web app page

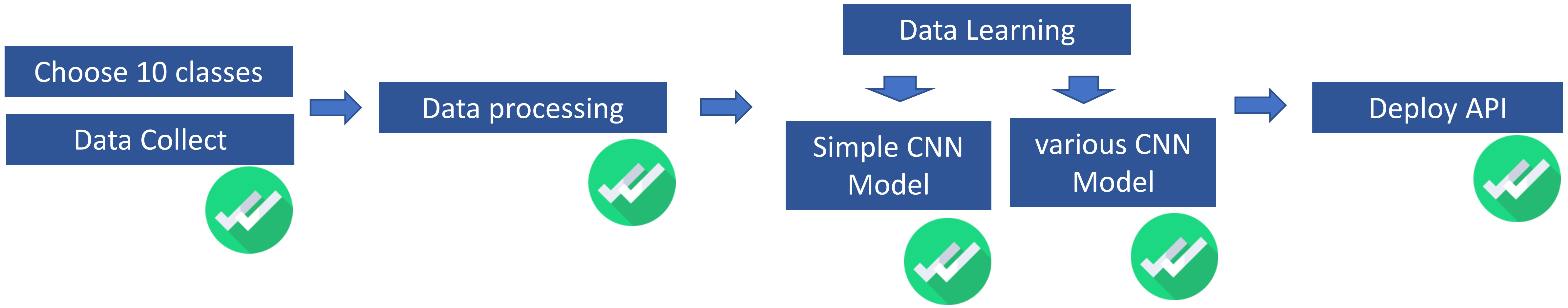
JEONG CHAEWON, LEE JI SEOP

CNN Model build > Week 9

CNN Build

“In this week?”

- **Select the model**
- **Implement confusion matrix and check which dataset needs to be modified**
- **Modify the dataset**



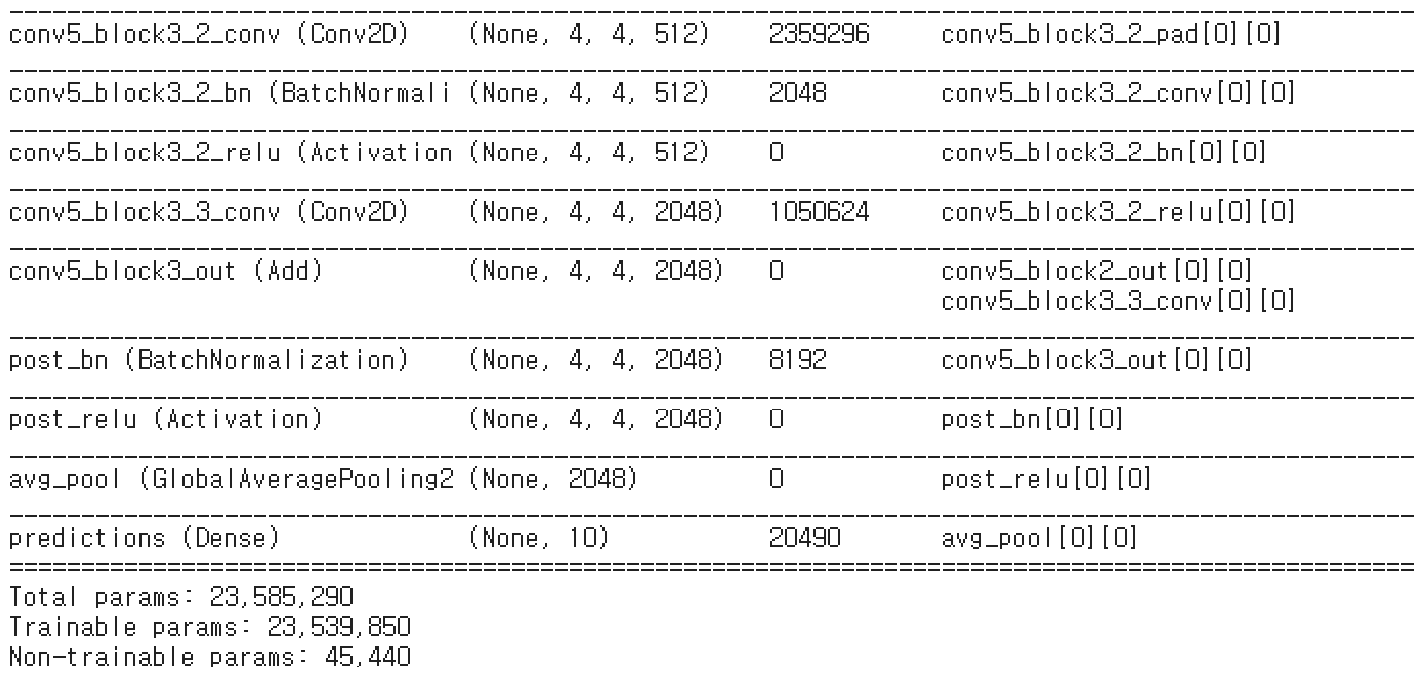
CNN Model build > Week 9

Select model

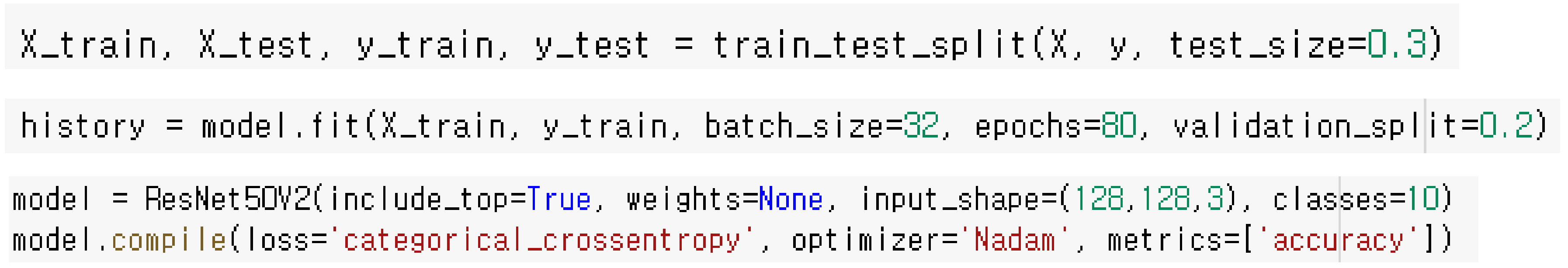
➢ Model spec

- ResNet50 model is adopted.
- Total image data : 25,450
- Training & validation data: 17,815
- Input Size : 128 * 128
- Train set, Validation set, Test set : 5:2:3
- Classes : 10
- Batch size : 32 epoch : 80
- Optimizer : Nadam

“Our selected model?”



```
________________________________________________________________________________________________________
conv5_block3_2_conv (Conv2D)    (None, 4, 4, 512)    2359296     conv5_block3_2_pad[0][0]
________________________________________________________________________________________________________
conv5_block3_2_bn (BatchNormali (None, 4, 4, 512)    2048        conv5_block3_2_conv[0][0]
________________________________________________________________________________________________________
conv5_block3_2_relu (Activation (None, 4, 4, 512)    0           conv5_block3_2_bn[0][0]
________________________________________________________________________________________________________
conv5_block3_3_conv (Conv2D)    (None, 4, 4, 2048)   1050624     conv5_block3_2_relu[0][0]
________________________________________________________________________________________________________
conv5_block3_out (Add)          (None, 4, 4, 2048)   0           conv5_block2_out[0][0]
                                                                 conv5_block3_3_conv[0][0]
________________________________________________________________________________________________________
post_bn (BatchNormalization)    (None, 4, 4, 2048)   8192        conv5_block3_out[0][0]
________________________________________________________________________________________________________
post_relu (Activation)          (None, 4, 4, 2048)   0           post_bn[0][0]
________________________________________________________________________________________________________
avg_pool (GlobalAveragePooling2 (None, 2048)         0           post_relu[0][0]
________________________________________________________________________________________________________
predictions (Dense)             (None, 10)           20490       avg_pool[0][0]
========================================================================================================
Total params: 23,585,290
Trainable params: 23,539,850
Non-trainable params: 45,440
```



```python
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)
```

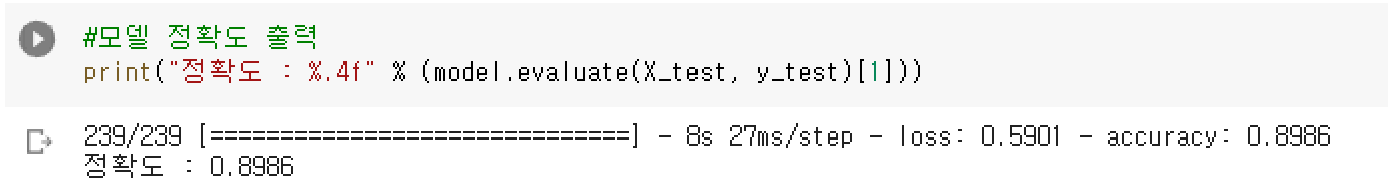
```python
history = model.fit(X_train, y_train, batch_size=32, epochs=80, validation_split=0.2)
```

```python
model = ResNet50V2(include_top=True, weights=None, input_shape=(128,128,3), classes=10)
model.compile(loss='categorical_crossentropy', optimizer='Nadam', metrics=['accuracy'])
```

CNN Model build > Week 9

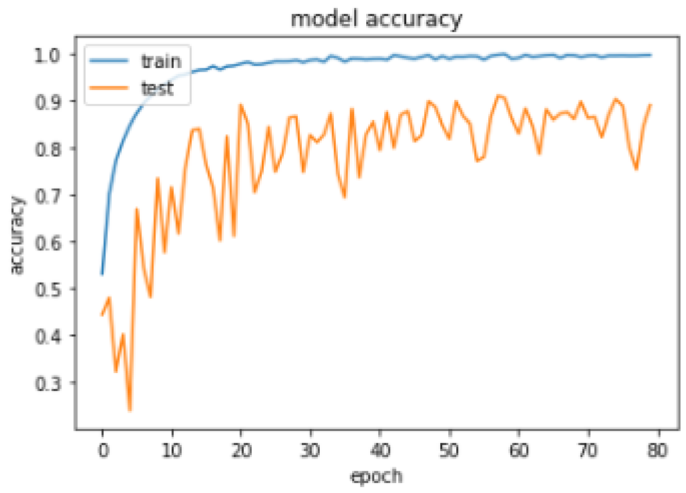
Result of Select model

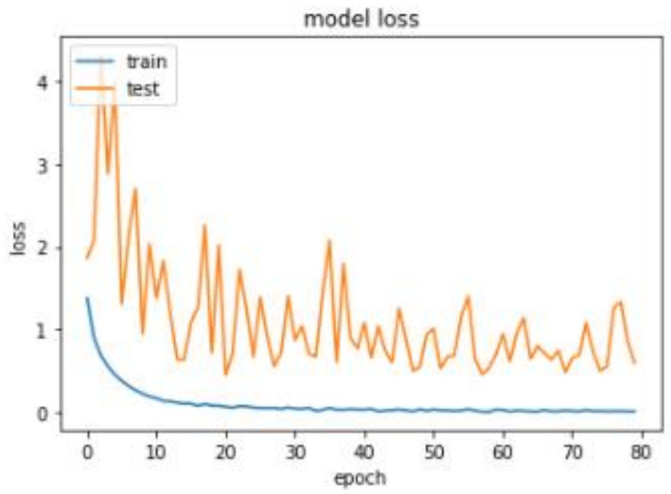
Accuracy of model is 89.86%



```
#모델 정확도 출력
print("정확도 : %.4f" % (model.evaluate(X_test, y_test)[1]))
```

```
239/239 [==============================] - 8s 27ms/step - loss: 0.5901 - accuracy: 0.8986
정확도 : 0.8986
```





CNN Model build > Week 9

Confusion matrix

Confusion Matrix

```
[[627   5   4   2   2   1   0   8   7   3]
 [  5 814   8   1   4   3   4   2   1   8]
 [  4  14 586  11   1   3   4   4  11  14]
 [  9  11  13 640   2   6   6  30   7   7]
 [ 20  15   6   2 776   1  10   3   5  31]
 [ 10  15   8   3   0 586   4  13   1  17]
 [  3  19   4   1   7   0 789   3   3  36]
 [ 12   9   1   2   1   5   0 701  12   9]
 [ 11   6   4   3   1   1   1   6 872   6]
 [ 12  50  14   6  12  17  86  11  11 471]]
```

```python
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt
import itertools
labels = ["DDP", "GLFP", "NP", "NST","THSM",
              "MC", "IHV", "JLT", "HRS", "HBC"]

y_pred=model.predict(X_test)
y_test=np.argmax(y_test, axis=1)
y_pred=np.argmax(y_pred, axis=1)
cm = confusion_matrix(y_test, y_pred)
print(cm)
```

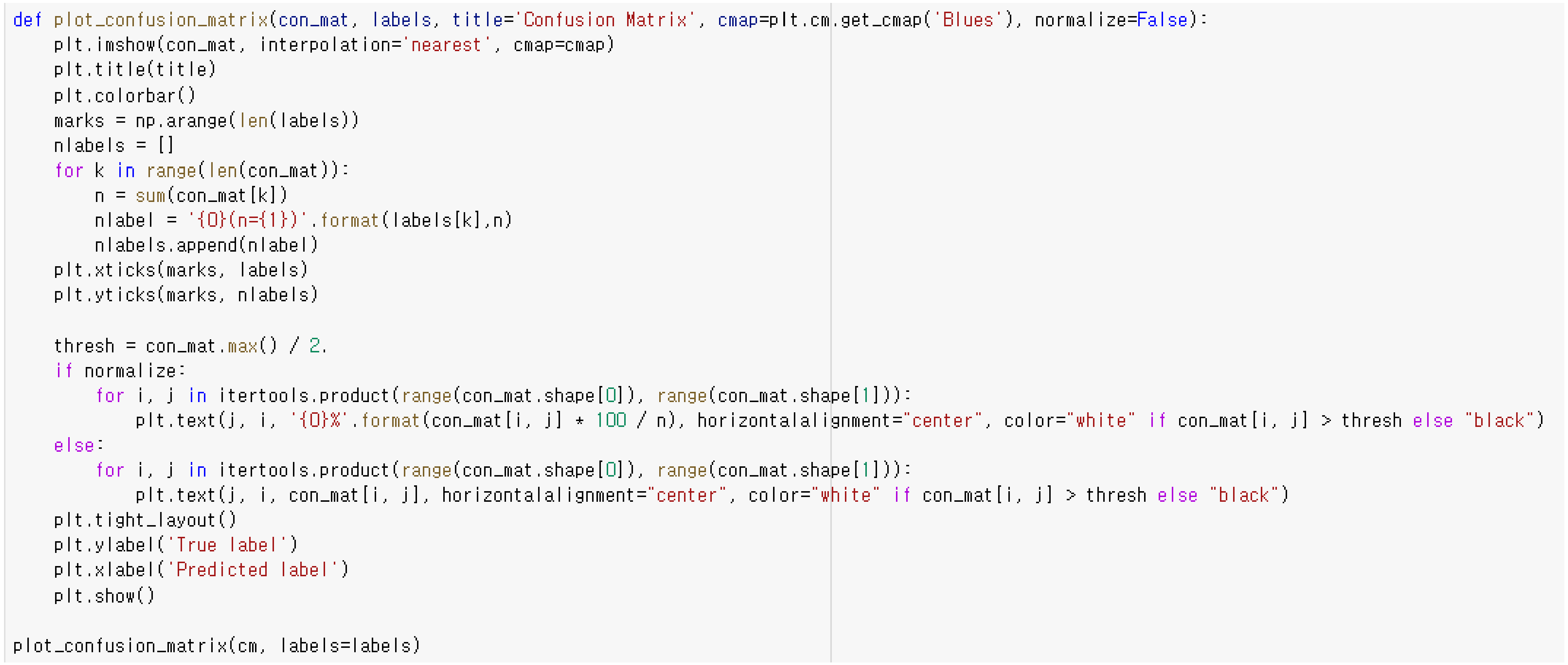
Simple confusion matrix

Code for confusion matrix

CNN Model build > Week 9

Confusion matrix

Code for Confusion Matrix by using matplot



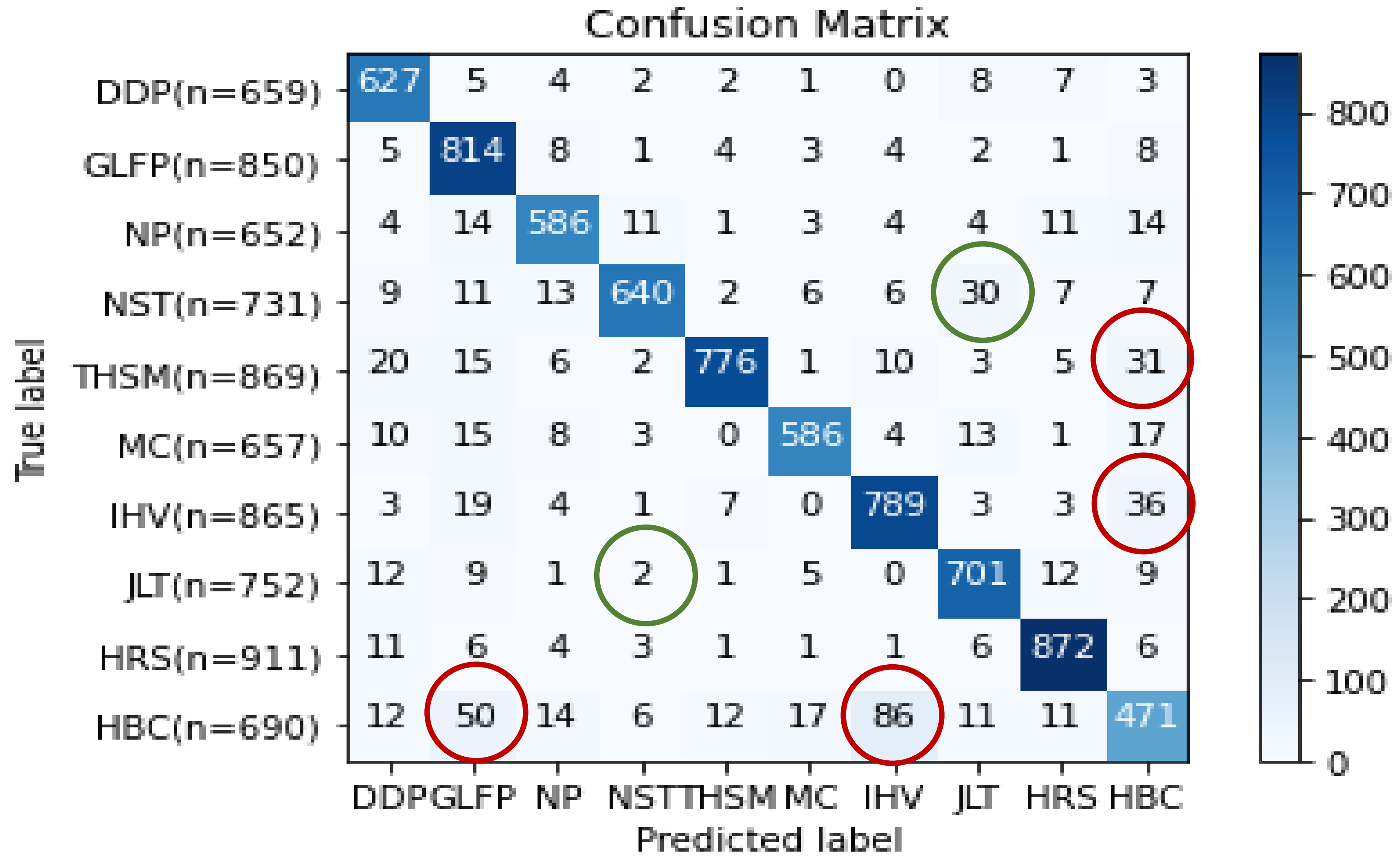
```python
def plot_confusion_matrix(con_mat, labels, title='Confusion Matrix', cmap=plt.cm.get_cmap('Blues'), normalize=False):
    plt.imshow(con_mat, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    marks = np.arange(len(labels))
    nlabels = []
    for k in range(len(con_mat)):
        n = sum(con_mat[k])
        nlabel = '{0}(n={1})'.format(labels[k],n)
        nlabels.append(nlabel)
    plt.xticks(marks, labels)
    plt.yticks(marks, nlabels)

    thresh = con_mat.max() / 2.
    if normalize:
        for i, j in itertools.product(range(con_mat.shape[0]), range(con_mat.shape[1])):
            plt.text(j, i, '{0}%'.format(con_mat[i, j] * 100 / n), horizontalalignment="center", color="white" if con_mat[i, j] > thresh else "black")
    else:
        for i, j in itertools.product(range(con_mat.shape[0]), range(con_mat.shape[1])):
            plt.text(j, i, con_mat[i, j], horizontalalignment="center", color="white" if con_mat[i, j] > thresh else "black")
    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')
    plt.show()

plot_confusion_matrix(cm, labels=labels)
```

CNN Model build > Week 9

Confusion matrix



True label \ Predicted label	DDP	GLFP	NP	NST	THSM	MC	IHV	JLT	HRS	HBC
DDP(n=659)	627	5	4	2	2	1	0	8	7	3
GLFP(n=850)	5	814	8	1	4	3	4	2	1	8
NP(n=652)	4	14	586	11	1	3	4	4	11	14
NST(n=731)	9	11	13	640	2	6	6	30	7	7
THSM(n=869)	20	15	6	2	776	1	10	3	5	31
MC(n=657)	10	15	8	3	0	586	4	13	1	17
IHV(n=865)	3	19	4	1	7	0	789	3	3	36
JLT(n=752)	12	9	1	2	1	5	0	701	12	9
HRS(n=911)	11	6	4	3	1	1	1	6	872	6
HBC(n=690)	12	50	14	6	12	17	86	11	11	471

- DDP : Dongdaemun Design Plaza
- GLFP : Gyeongui Line Forest Park
- NP : Naksan Park
- NST : Namsan Seoul Tower
- THSM : The Hyundai Seoul Mall
- MC : Myeongdong Cathedral
- IHV : Ikseon Dong Hanok Village
- JLT : Jamsil Lotte Tower
- HRS : Han River Sebitseom
- HBC : Haebangchon

CNN Model build > Week 9

Confusion matrix

All of these are Haebangchon

Hanok village

Gyeongui Line Forest Park

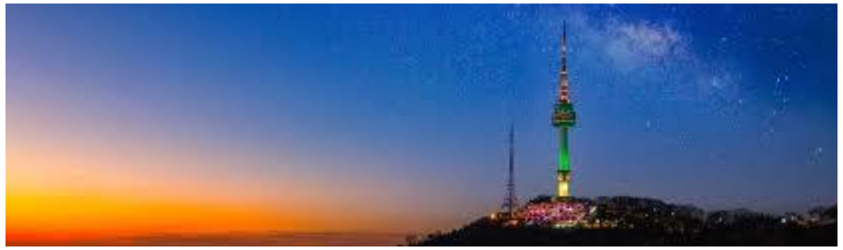
The Hyundai Seoul Mall

➢ Cause of problem

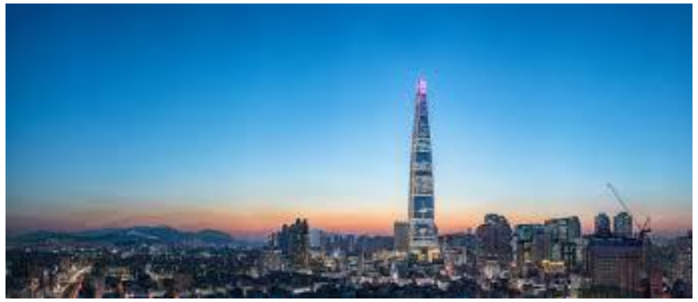
- These place has too much broad regional range
- Especially, Haebangchon has so many different pictures of various location

CNN Model build > Week 9

Confusion matrix



Namsan Seoul Tower



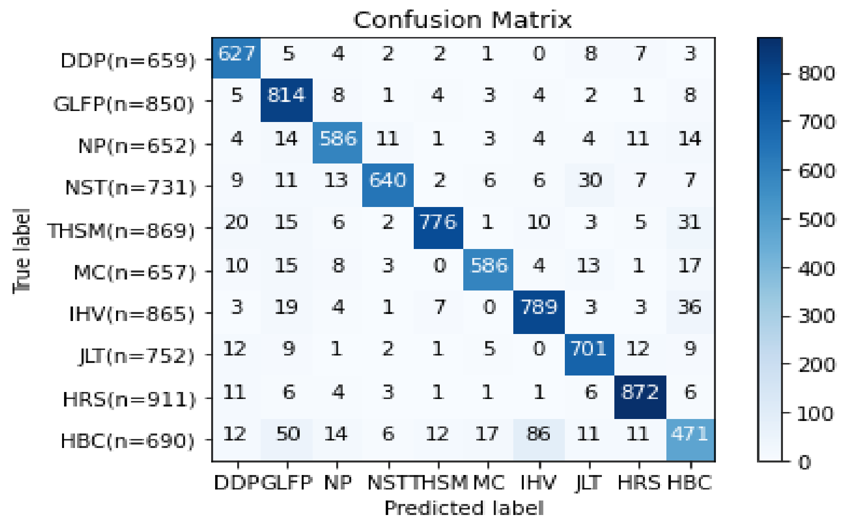
Jamsil lotte tower

➢ **Cause of problem**

- If we take the picture far from the tower, the features of them is very similar as distance is increasing
- Compare two towers above

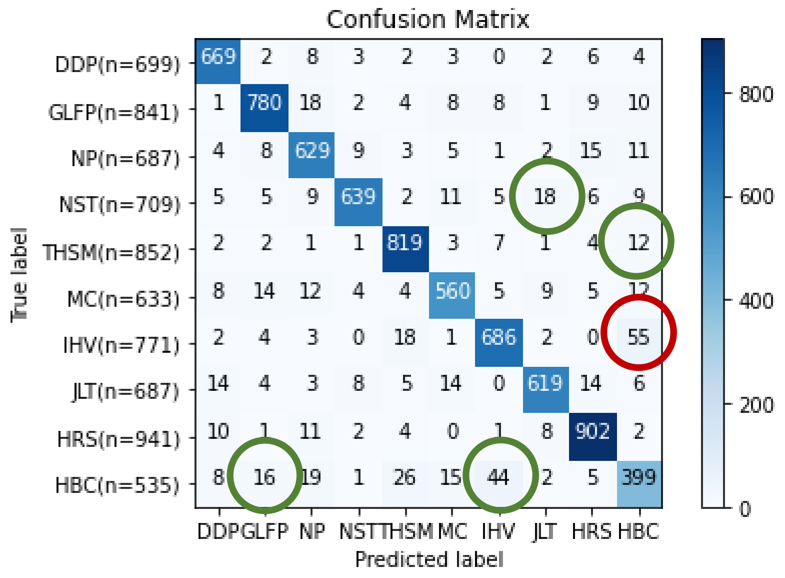
CNN Model build > Week 9

Confusion matrix



True label \ Predicted label	DDP	GLFP	NP	NST	THSM	MC	IHV	JLT	HRS	HBC
DDP(n=659)	627	5	4	2	2	1	0	8	7	3
GLFP(n=850)	5	814	8	1	4	3	4	2	1	8
NP(n=652)	4	14	586	11	1	3	4	4	11	14
NST(n=731)	9	11	13	640	2	6	6	30	7	7
THSM(n=869)	20	15	6	2	776	1	10	3	5	31
MC(n=657)	10	15	8	3	0	586	4	13	1	17
IHV(n=865)	3	19	4	1	7	0	789	3	3	36
JLT(n=752)	12	9	1	2	1	5	0	701	12	9
HRS(n=911)	11	6	4	3	1	1	1	6	872	6
HBC(n=690)	12	50	14	6	12	17	86	11	11	471

BEFORE



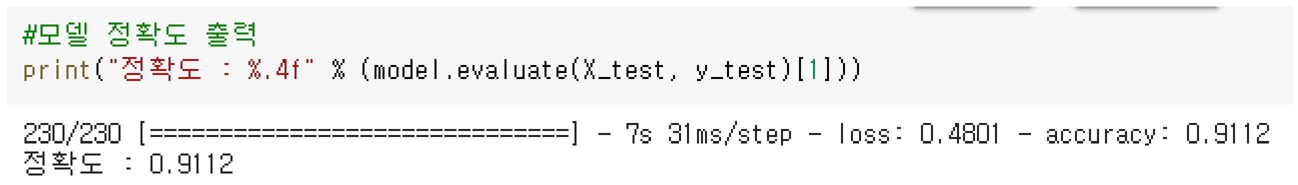
True label \ Predicted label	DDP	GLFP	NP	NST	THSM	MC	IHV	JLT	HRS	HBC
DDP(n=699)	669	2	8	3	2	3	0	2	6	4
GLFP(n=841)	1	780	18	2	4	8	8	1	9	10
NP(n=687)	4	8	629	9	3	5	1	2	15	11
NST(n=709)	5	5	9	639	2	11	5	18	6	9
THSM(n=852)	2	2	1	1	819	3	7	1	4	12
MC(n=633)	8	14	12	4	4	560	5	9	5	12
IHV(n=771)	2	4	3	0	18	1	686	2	0	55
JLT(n=687)	14	4	3	8	5	14	0	619	14	6
HRS(n=941)	10	1	11	2	4	0	1	8	902	2
HBC(n=535)	8	16	19	1	26	15	44	2	5	399

AFTER

- DDP : Dongdaemun Design Plaza
- GLFP : Gyeongui Line Forest Park
- NP : Naksan Park
- NST : Namsan Seoul Tower
- THSM : The Hyundai Seoul Mall
- MC : Myeongdong Cathedral
- IHV : Ikseon Dong Hanok Village
- JLT : Jamsil Lotte Tower
- HRS : Han River Sebitseom
- HBC : Haebangchon

➢ Improved performance

- We remove ambiguous images
- Then, we can get 91% accuracy as same condition
- Also, we get improved result in confusion matrix



```
#모델 정확도 출력
print("정확도 : %.4f" % (model.evaluate(X_test, y_test)[1]))
```

```
230/230 [==============================] - 7s 31ms/step - loss: 0.4801 - accuracy: 0.9112
정확도 : 0.9112
```

CNN Model build > Week 9

Next week

1. Finally modify the dataset

   HONG SEONG JUN

2. Choose the final model & check the accuracy

   CHE SEUNG YUN

3. Sum up the CNN building process for final report

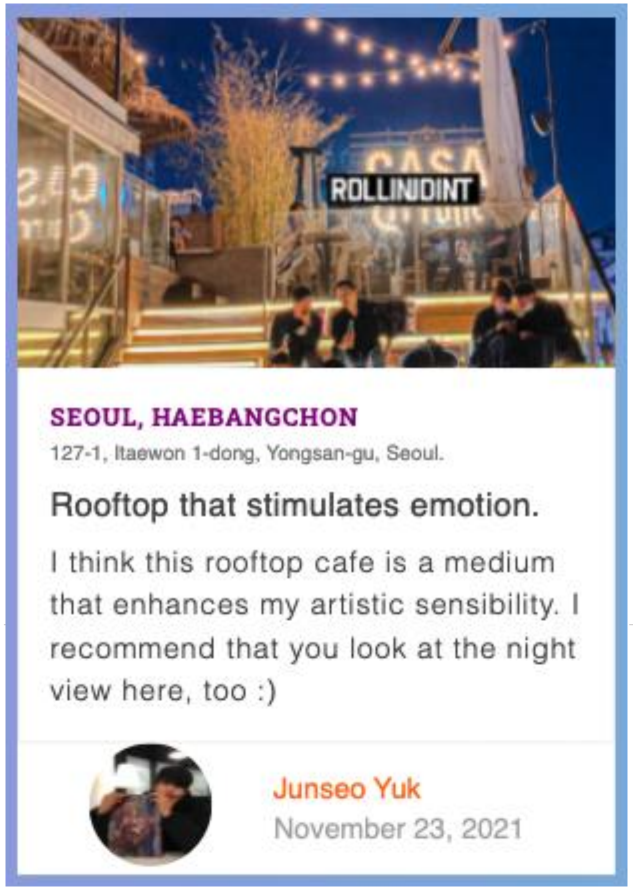
   CHE SEUNG YUN, UHM JI YONG

Frontend

We are near the end!

- **Last week, we did…**

More on SNS page



SEOUL, HAEBANGCHON

127-1, Itaewon 1-dong, Yongsan-gu, Seoul.

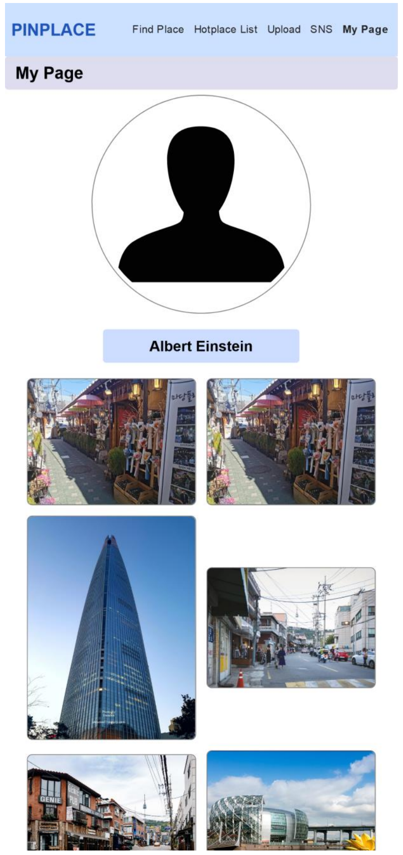
Rooftop that stimulates emotion.

I think this rooftop cafe is a medium that enhances my artistic sensibility. I recommend that you look at the night view here, too :)

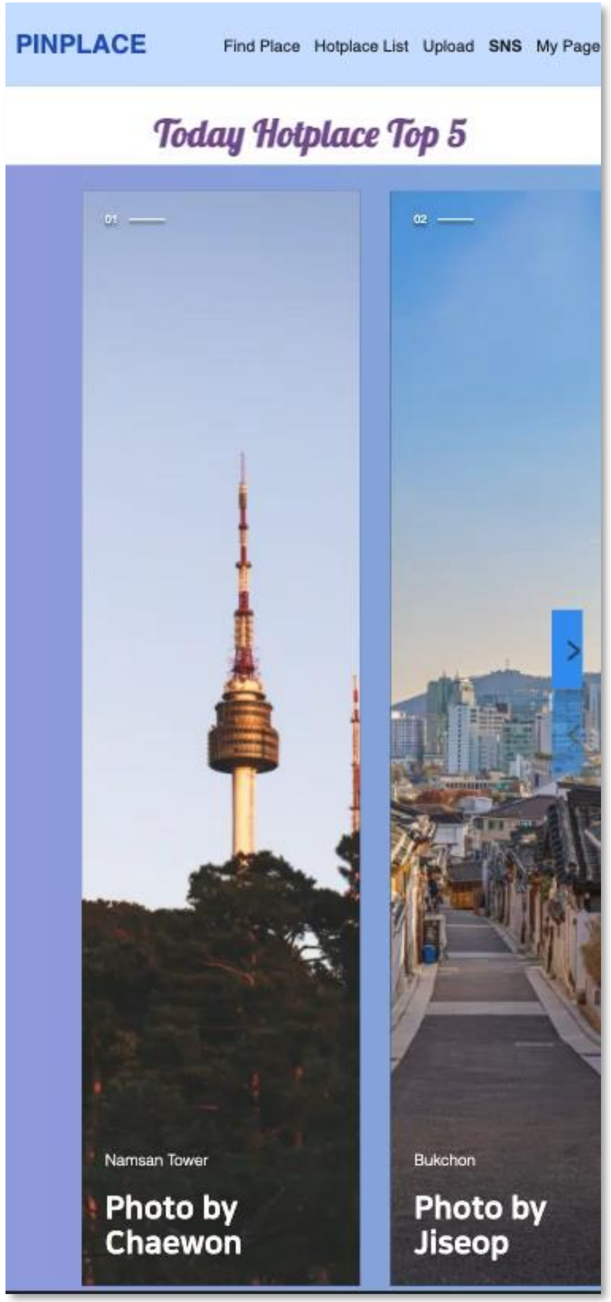
Junseo Yuk

November 23, 2021

"My Page"

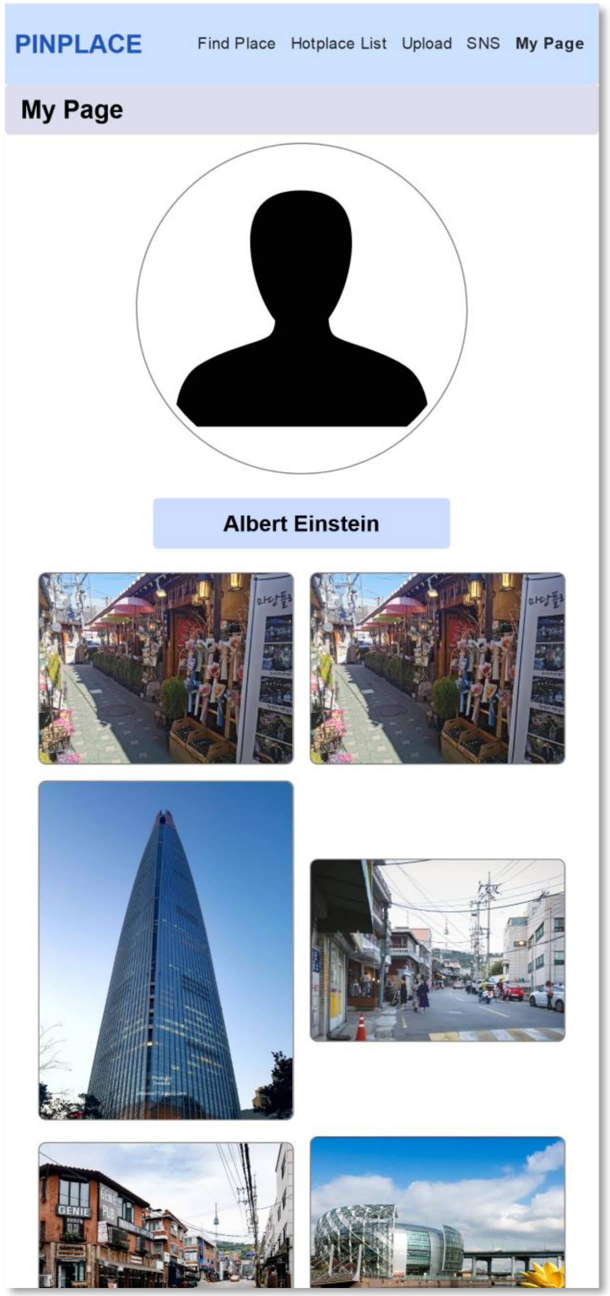


Frontend > SNS Page



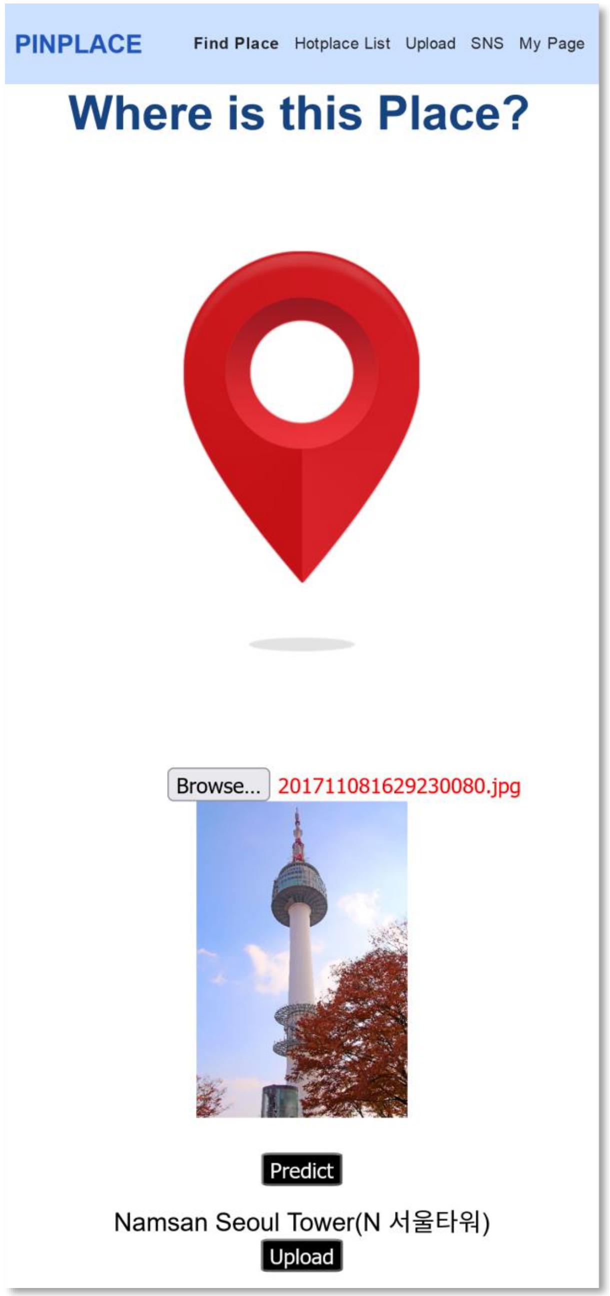
- Posts by users are shown under the “Top” section
- Posts will consist of an image and some text
- One may keep scrolling to see more posts hypothetically

Frontend > My Page



- Images uploaded by oneself is shown
- Design is similar to hotplace pages
- Needs to be tied with user credential

Frontend > Find Place


PINPLACE

Find Place Hotplace List Upload SNS My Page

Where is this Place?

Browse... 201711081629230080.jpg

Predict

Namsan Seoul Tower(N 서울타워)

Upload

- After the AI model predicts, an upload button shows
- The uploaded photo will be shown in My Page

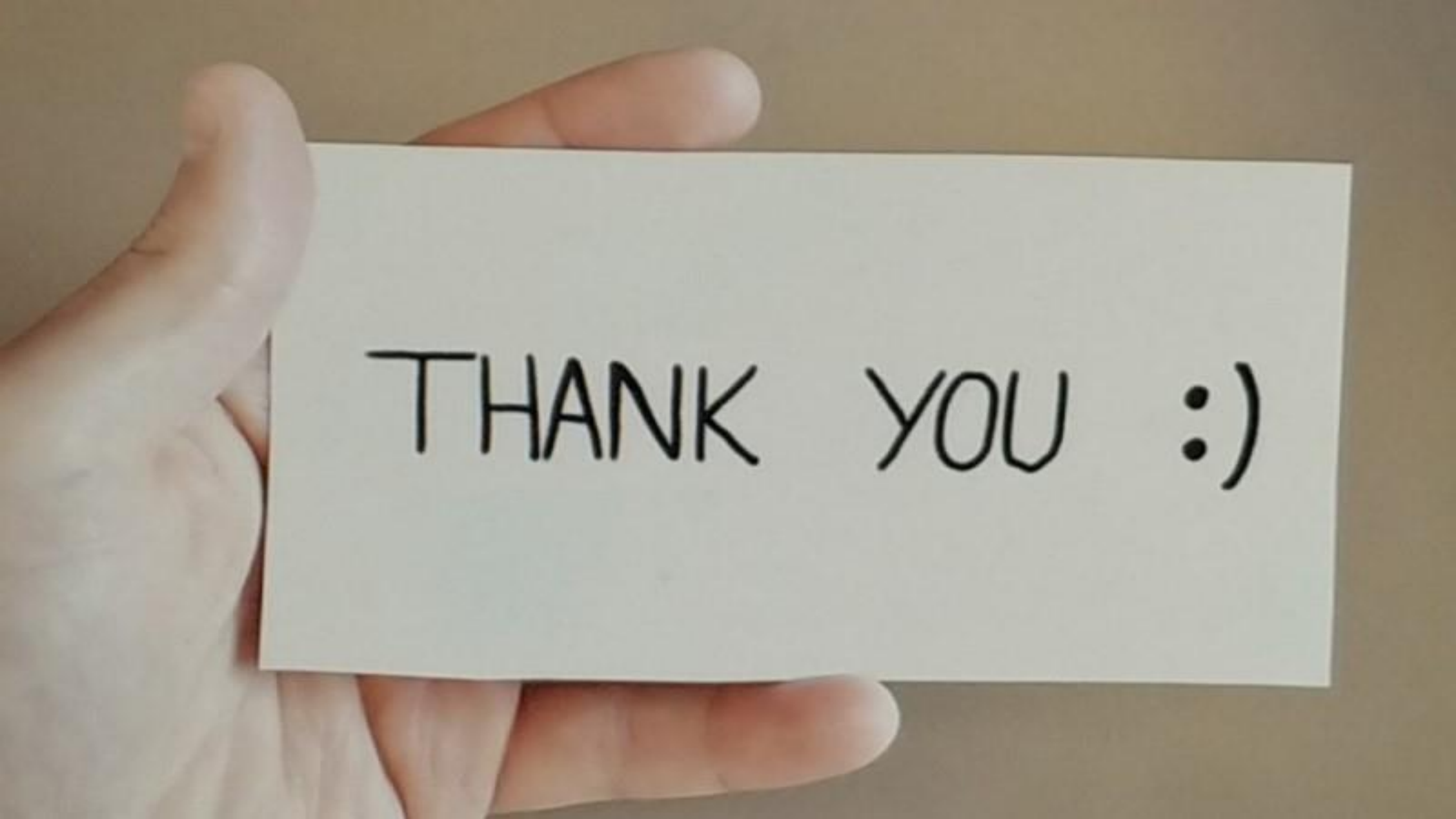
Frontend > To do

- **Refactoring**

- Writing style is quite different
- Website structure is like a spaghetti code
- For easy maintenance, we need to focus on refactoring

- **Design fix**

- There may still exist still some visual errors and lack of styling
- We have fixed what we discovered before, but some remains


THANK YOU :)