Land Cover Classification and Identification

Xintian (Stella) Li, David (Yuanrong) Pan, Jacey Chang

CONTENTS

01

Introduction

Problem and Dataset

Why Land Cover Classification?

Problem

Due to human activities, land cover is changing rapidly over the recent decades.

Motivations

Monitoring land cover changes help us understand and evaluate the urban development process and its associated environmental impacts

Technical Possibility

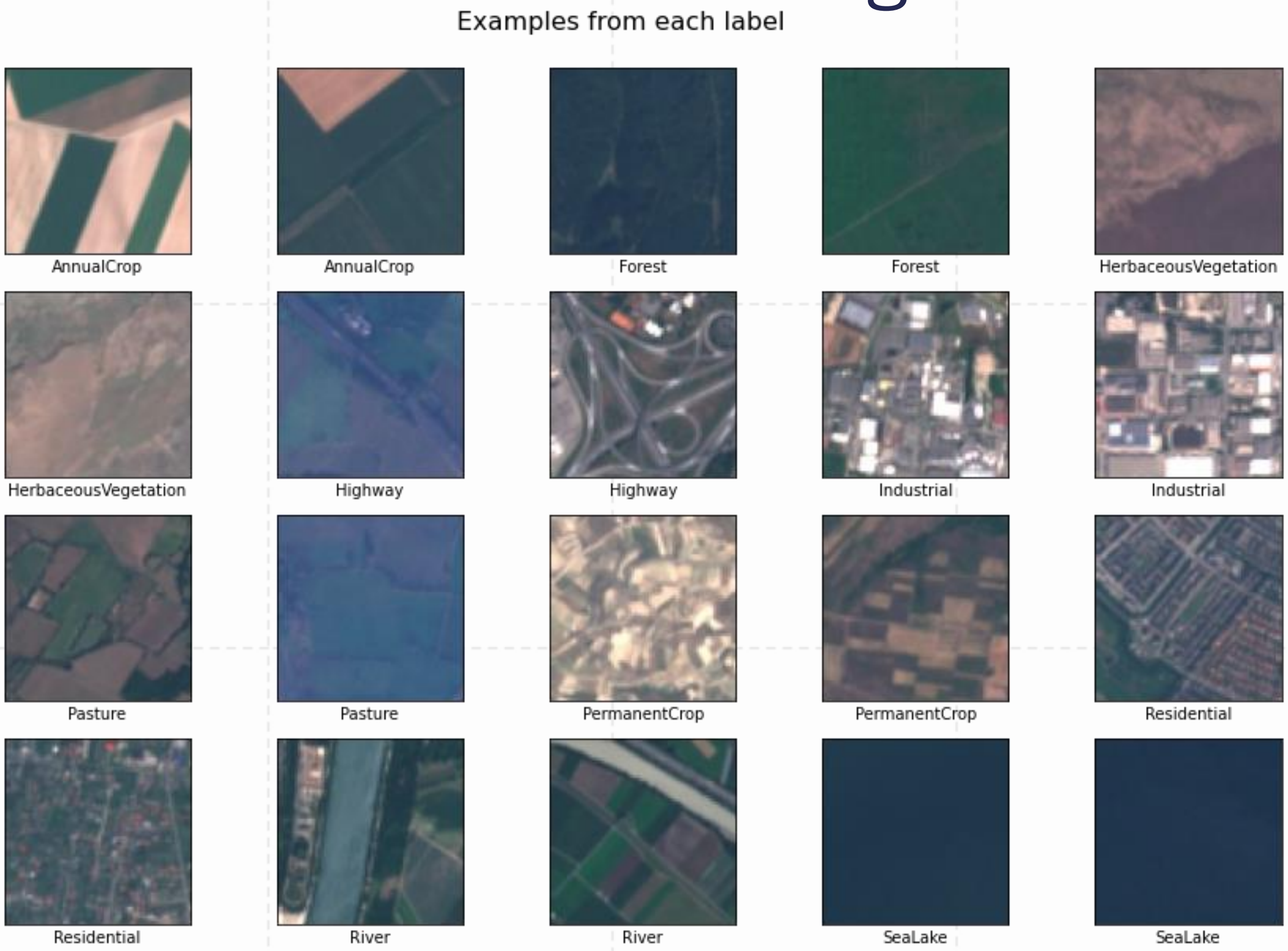
The proliferation of satellite image data and the emergence of advanced machine learning technologies make it possible to identify land cover and quantify the changes automatically.

EuroSAT Dataset

a dataset based on Sentinel-2 satellite images and consisting out of 10 classes with in total 27,000 labeled images, each 64 pixel * 64 pixel * 3 channels.

The 10 labels: Industrial Buildings, Residential Buildings, Annual Crop, Permanent Crop, River, Sea & Lake, Herbaceous Vegetation, Highway, Pasture, Forest

Visualize the Images



02

Data Preprocessing

Relabel and PCA

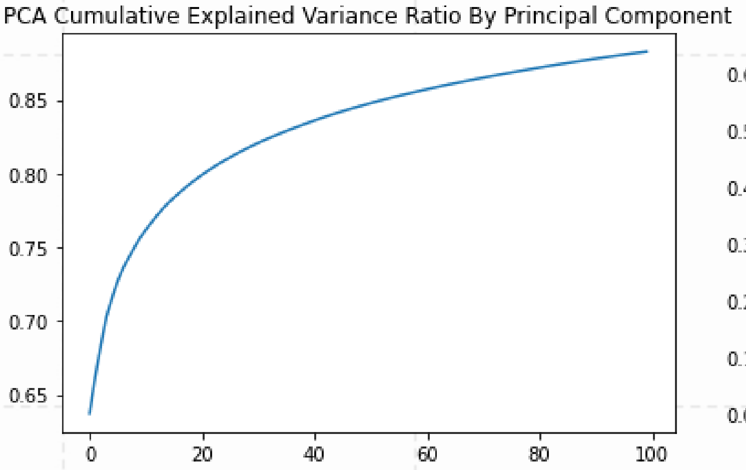
Relabel the Data

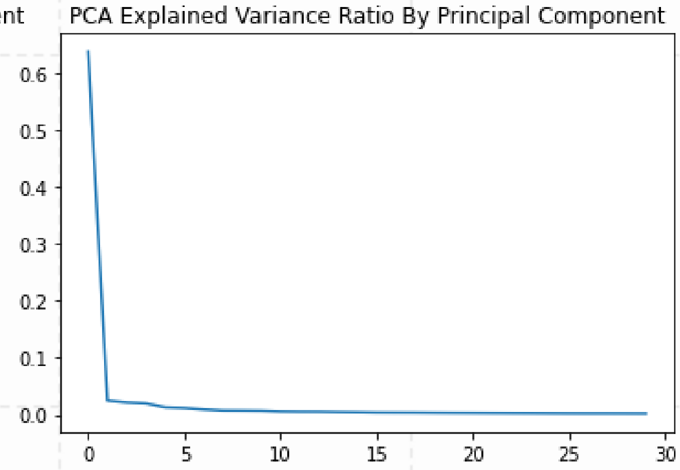
We relabel the data to make the data **balanced** and faster to train.

Original Label	New Label
AnnualCrop	Agriculture
PermanentCrop	
Pasture	
River	Water
SeaLake	
Residential	Building
Industrial	
Forest	Forest
HerbaceousVegetation	HerbaceousVegetation
Highway	Highway

Dimensionality Reduction: PCA

Dimensionality reduction is needed because huge data size and thus a long training time with the full dataset.





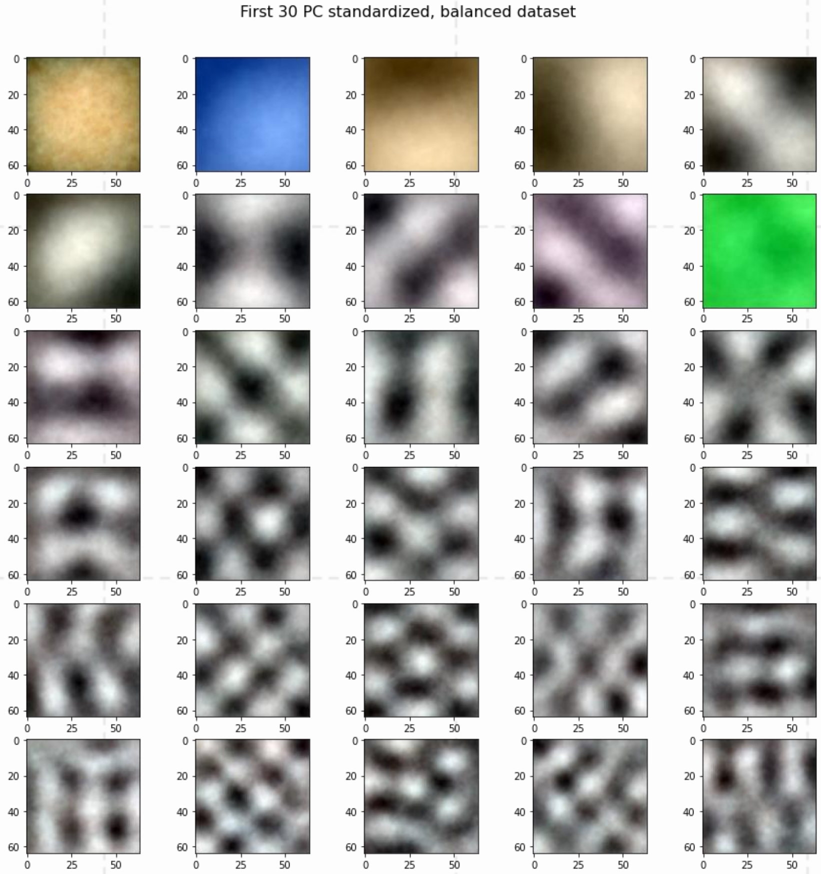
The first PC explains over 60% of the variance

The first 30 PCs explain over 82% of the variance

The first 100 PCs explain about 88% of the variance

We choose to use first 30 PCs in our models.

Visualize PCA Components



It is shown in the PCA that the average hue of images matters, especially how much “yellowness” in the image.

03

Machine Learning

K-Means, SVM and Random Forest

KMeans Clusters

We first split the 30-pc dimensionality-reduced data into a 70% training set and a 30% testing test.

Then we tried 6-cluster KMeans as an unsupervised model to see whether the data were easy to separate.

cluster	0	1	2	3	4	5
actual_label						
Agriculture	129	219	146	53	166	7
Building	202	24	170	67	257	0
Forest	8	136	0	0	0	576
HerbaceousVegetation	242	161	124	18	151	24
Highway	235	232	38	41	135	39
Water	160	235	4	0	102	219

Except for recognizing "Forest" as a distinct cluster (cluster 5) with an rough accuracy around **67%,**

K-Means did a poor job to distinguish different labels.

SVM Classifier

Then we used SVM classifier. We did cross-validation, and used grid search several times to narrow down the best parameters.

```
pd.DataFrame(svc_grid2.cv_results_)[
    ['params','mean_test_score','std_test_score','rank_test_score']
].sort_values(by='rank_test_score')
```

	params	mean_test_score	std_test_score	rank_test_score
2	{'estimator__C': 15, 'estimator__kernel': 'rbf'}	0.671429	0.012084	1
3	{'estimator__C': 20, 'estimator__kernel': 'rbf'}	0.669841	0.011362	2
4	{'estimator__C': 30, 'estimator__kernel': 'rbf'}	0.669544	0.011607	3
1	{'estimator__C': 10, 'estimator__kernel': 'rbf'}	0.669345	0.009516	4
0	{'estimator__C': 1, 'estimator__kernel': 'rbf'}	0.629266	0.012061	5

The best accuracy is about **67%,** with an rbf kernel and parameter C=15

predicted_label	Agriculture	Building	Forest	HerbaceousVegetation	Highway	Water
actual_label						
Agriculture	411	66	16	103	85	39
Building	23	559	1	70	53	14
Forest	14	1	698	1	0	6
HerbaceousVegetation	60	94	10	489	41	26
Highway	103	106	11	61	331	108
Water	33	22	96	21	87	461

The confusion matrix for the prediction of the best SVM model

Random Forest Classifier

Random forest classifier had a similar performance with SVM classifier: The best model has a **68%** testing accuracy

predicted_label	Agriculture	Building	Forest	HerbaceousVegetation	Highway	Water
actual_label						
Agriculture	454	86	10	80	66	24
Building	16	583	0	61	47	13
Forest	19	0	678	2	0	21
HerbaceousVegetation	101	90	6	440	65	18
Highway	128	146	2	32	359	53
Water	37	41	45	15	147	435

Confusion matrix

	params	mean_test_score	std_test_score	rank_test_score
8	{'max_features': 'auto', 'n_estimators': 400}	0.680357	0.007448	1
1	{'max_features': 'sqrt', 'n_estimators': 200}	0.678869	0.007517	2
5	{'max_features': 'log2', 'n_estimators': 400}	0.677282	0.011021	3
2	{'max_features': 'sqrt', 'n_estimators': 400}	0.676984	0.010959	4
7	{'max_features': 'auto', 'n_estimators': 200}	0.675794	0.008566	5
4	{'max_features': 'log2', 'n_estimators': 200}	0.672024	0.009893	6
6	{'max_features': 'auto', 'n_estimators': 100}	0.670734	0.007979	7
3	{'max_features': 'log2', 'n_estimators': 100}	0.667956	0.013266	8
0	{'max_features': 'sqrt', 'n_estimators': 100}	0.666865	0.012826	9

Rank of model accuracies using different parameters

04

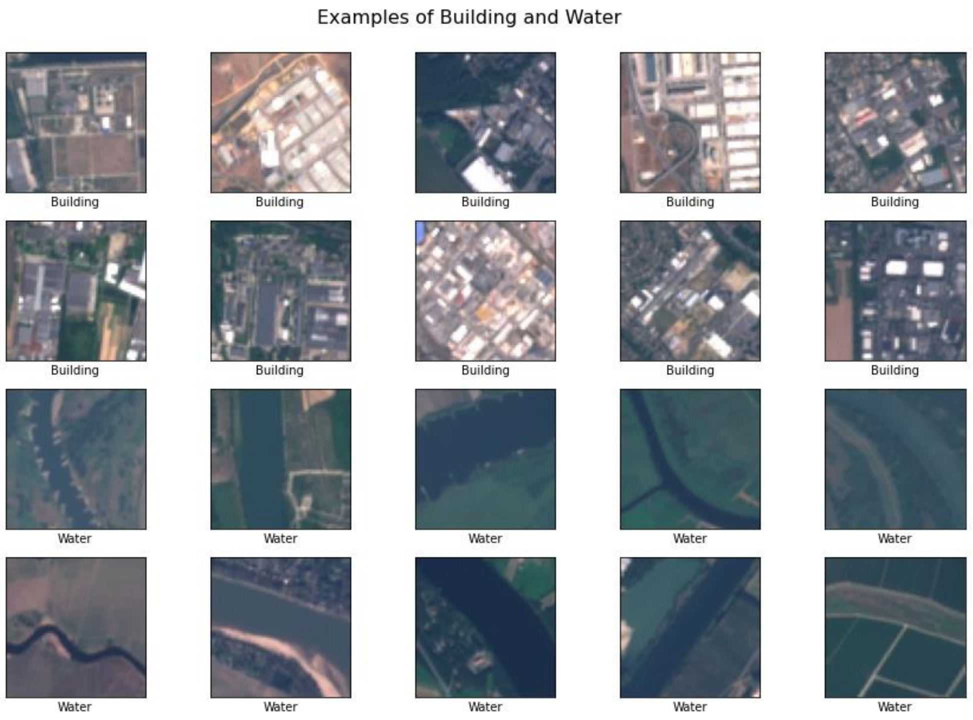
Deep Learning

CNN, Transfer Learning Models

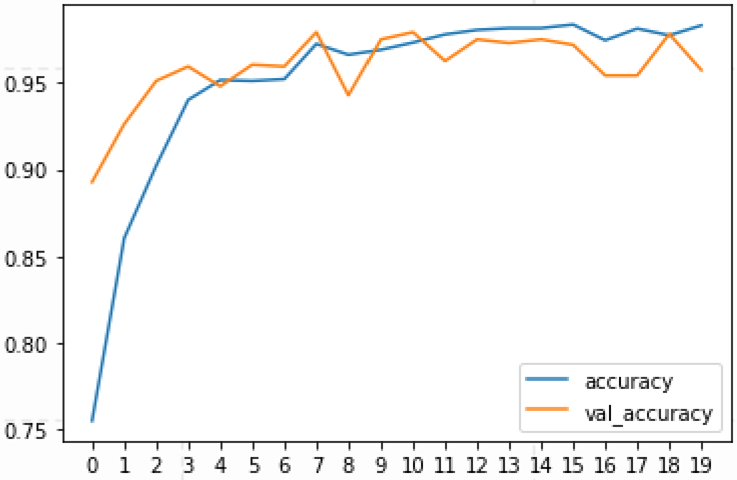
Convolutional Neural Network

First, we tried to use CNN to solve a simple 2-label classification question. We picked two classes which are very different from each other, buildings and water. Then we created a subset of data with 2,400 images in each of the two categories and perform CNN on the dataset.

Building vs. Water



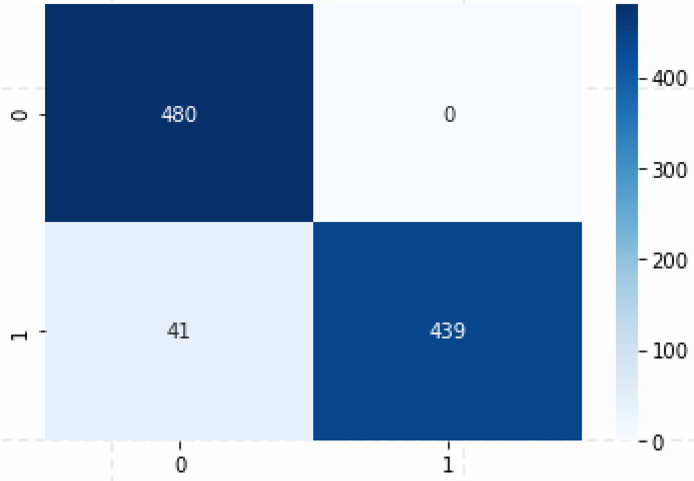
Convolutional Neural Network (Binary Classification)

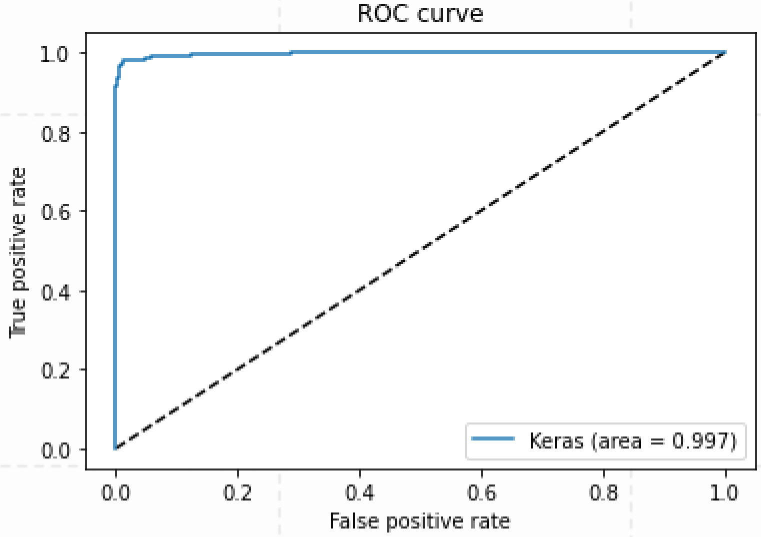



```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_8 (Conv2D)            (None, 64, 64, 32)        896
_________________________________________________________________
max_pooling2d_8 (MaxPooling2 (None, 32, 32, 32)        0
_________________________________________________________________
dropout_10 (Dropout)         (None, 32, 32, 32)        0
_________________________________________________________________
flatten_2 (Flatten)          (None, 32768)             0
_________________________________________________________________
dense_4 (Dense)              (None, 100)               3276900
_________________________________________________________________
dropout_11 (Dropout)         (None, 100)               0
_________________________________________________________________
dense_5 (Dense)              (None, 2)                 202
=================================================================
Total params: 3,277,998
Trainable params: 3,277,998
Non-trainable params: 0
_________________________________________________________________
```

Even a simple CNN network with only 1 convolutional layer and 1 dense layer can predict the 2 labels with a test accuracy of **96%**

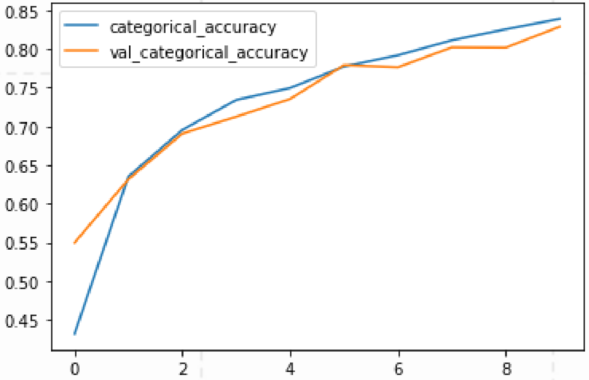
Convolutional Neural Network (Binary Classification)

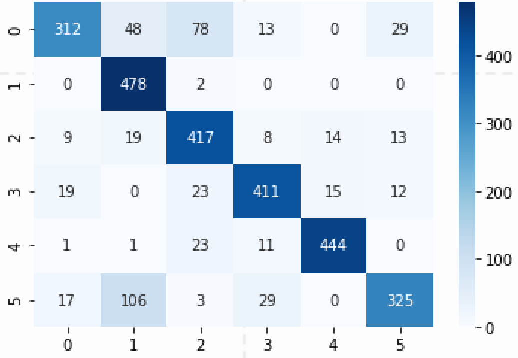




The model predict all the buildings correctly. The ROC curve is very close to the upper left corner, also indicating a good diagnosis ability of the 2-label classifier.

Convolutional Neural Network (Multi-class Classification)






```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 64, 64, 32)        896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 32, 32, 32)        0
_________________________________________________________________
dropout (Dropout)            (None, 32, 32, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 32, 32, 64)        18496
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 16, 16, 64)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 16, 16, 64)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 16, 16, 128)       73856
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 8, 8, 128)         0
_________________________________________________________________
dropout_2 (Dropout)          (None, 8, 8, 128)         0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 8, 8, 256)         295168
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 4, 4, 256)         0
_________________________________________________________________
dropout_3 (Dropout)          (None, 4, 4, 256)         0
_________________________________________________________________
flatten (Flatten)            (None, 4096)              0
_________________________________________________________________
dense (Dense)                (None, 120)               491640
_________________________________________________________________
dropout_4 (Dropout)          (None, 120)               0
_________________________________________________________________
dense_1 (Dense)              (None, 6)                 726
=================================================================
Total params: 880,782
Trainable params: 880,782
Non-trainable params: 0
_________________________________________________________________
```

The multi-class classification (6 labels) has a test accuracy of **83%**. The model is best at predicting Label 1 (Forest), not very good at predicting Label 0 and 5 (Agriculture and Water).

The accuracy is much higher relative to the ML classifiers.

Transfer Learning Models

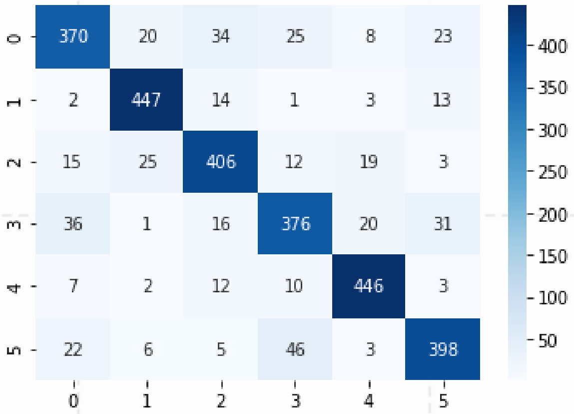
Then we imported two pre-trained models (VGG16 and MobileNetV2) to fit and test on our own datasets.

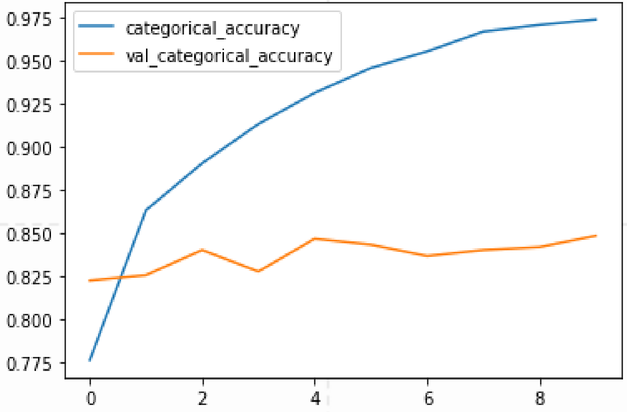
We set input shape to (64, 64, 3), froze the convolutional layers of the pre-trained model, transferred to our datasets and updated the dense layers to get the output labels.

VGG16 accuracy: 85.5%
MobileNetV2: accuracy: 84.8%

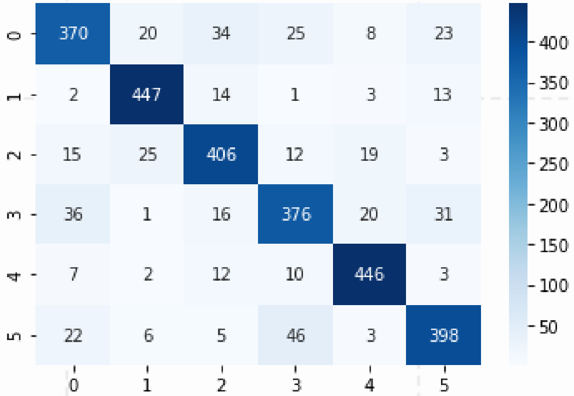
Transfer Learning Models

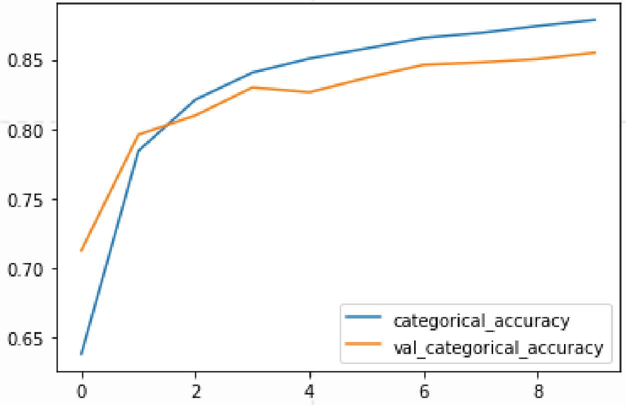
MobileNetV2: accuracy: 84.8%





VGG16 accuracy: 85.5%





Thank you!

If you have any questions, please contact us at:

xintianl@upenn.edu
yrpan@upenn.edu
jzchang@upenn.edu