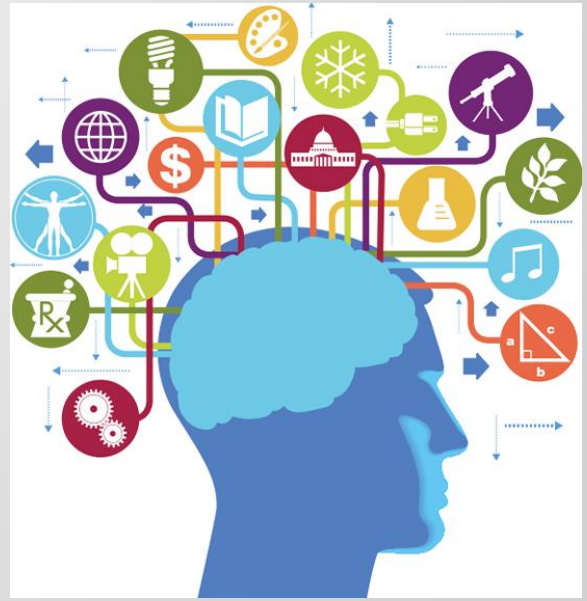
Capstone Project III
Establishing Images Classification System by Convolutional Neural Network (CNN)

Lok Hang Ronald, Wong

Background Information

- Image classification is one of the hottest topics in ML's world

- The usages of image classification are wide and significant in many academic subjects
  - Computer Science (Graphics, Algorithms, Theory, Systems, Architecture)
  - Mathematics (Information Retrieval, Machine Learning)
  - Engineering (Robotics, Speech, NLP, Image Processing)
  - Physics (Optics), Biology (Neuroscience)
  - Psychology (Cognitive Science).



Aims and Objectives

- Establish a CNN model which can classify 20 types of dog breeds from the labelled images
- The established model can serve as baseline model which can apply for different kinds of classification problems, i.e.
  - Vehicles Image Classification
  - Face Recognition
  - X-ray (or other medical images) Diagnosis

Dataset

- The original data source is found on http://vision.stanford.edu/aditya86/ImageNetDogs/
  - Created by Aditya Khosla et al.

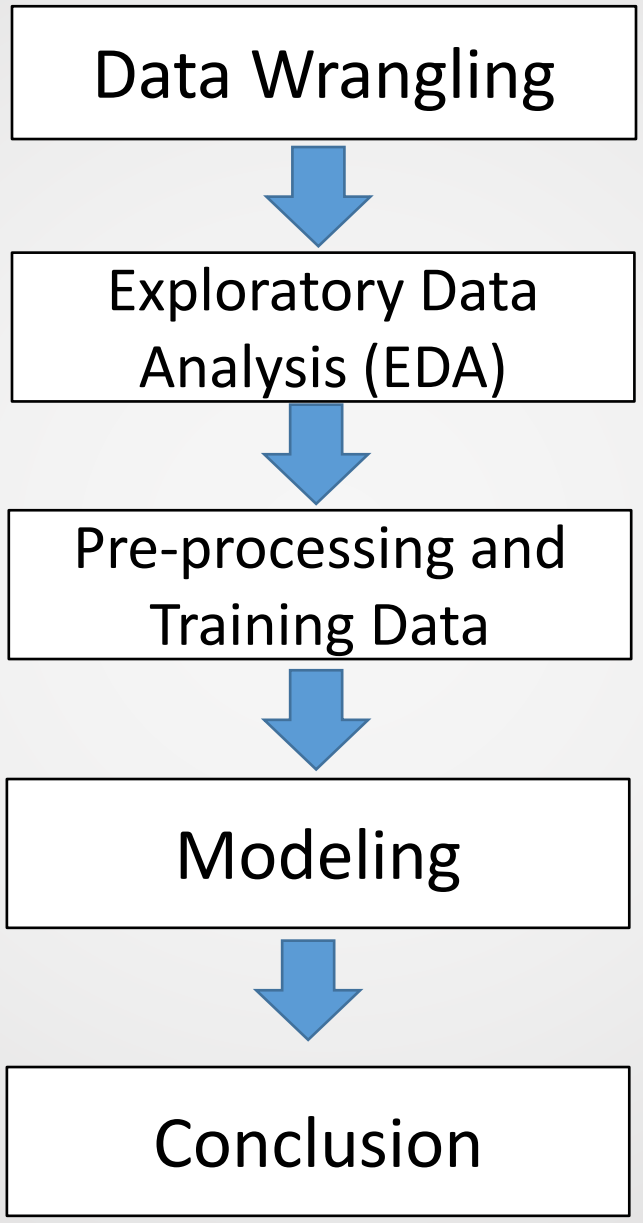
- The original data set contains:
  - Number of categories: 120
  - Number of images: 20,580
  - Annotations: Class labels, Bounding boxes

- Due to computation power, this project reduces the problems to:
  - Number of categories: 20
  - Number of images: 3,629

Dataset

Data Preview:

Project Outline



-Subset of 20 dog breeds
-Data Quality Checking

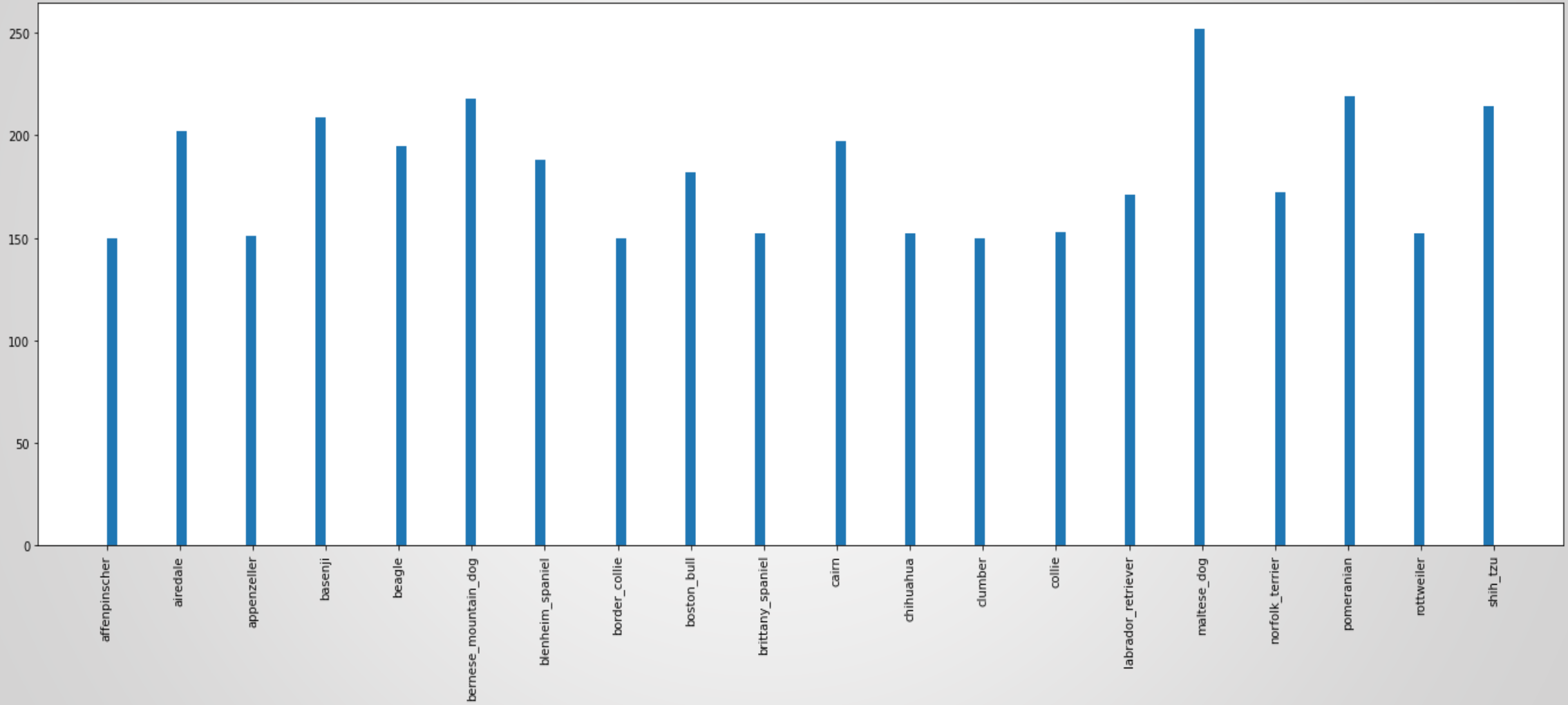
-Check for unbalance class
-Data Visualization

-Re-size & augmentation
-Data splitting

-Models Comparison
-VGG16, Inception V3 & Self - customized CNN

-Confusion Matrix
-Precision & Recall

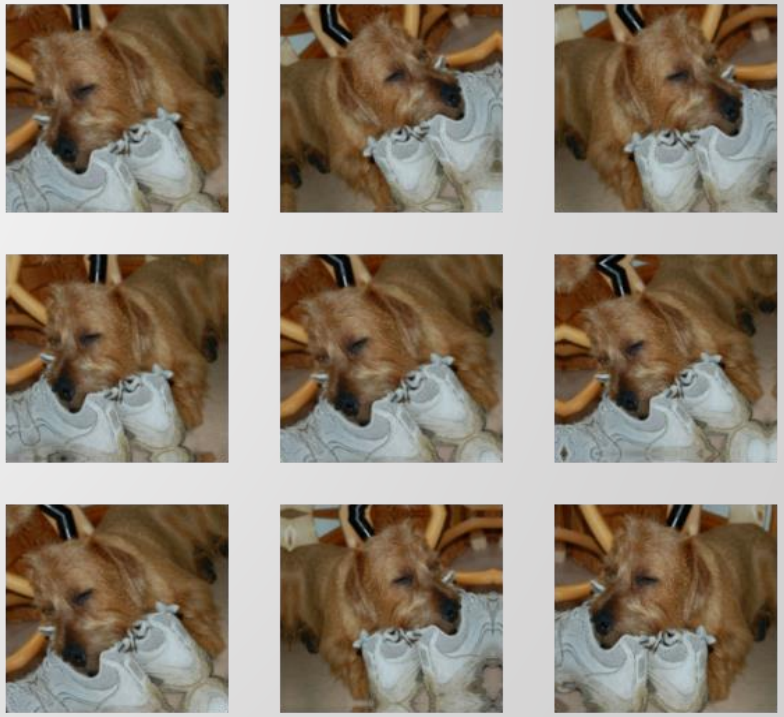
Exploratory Data Analysis (EDA) – Balanced Data



Data Preprocessing

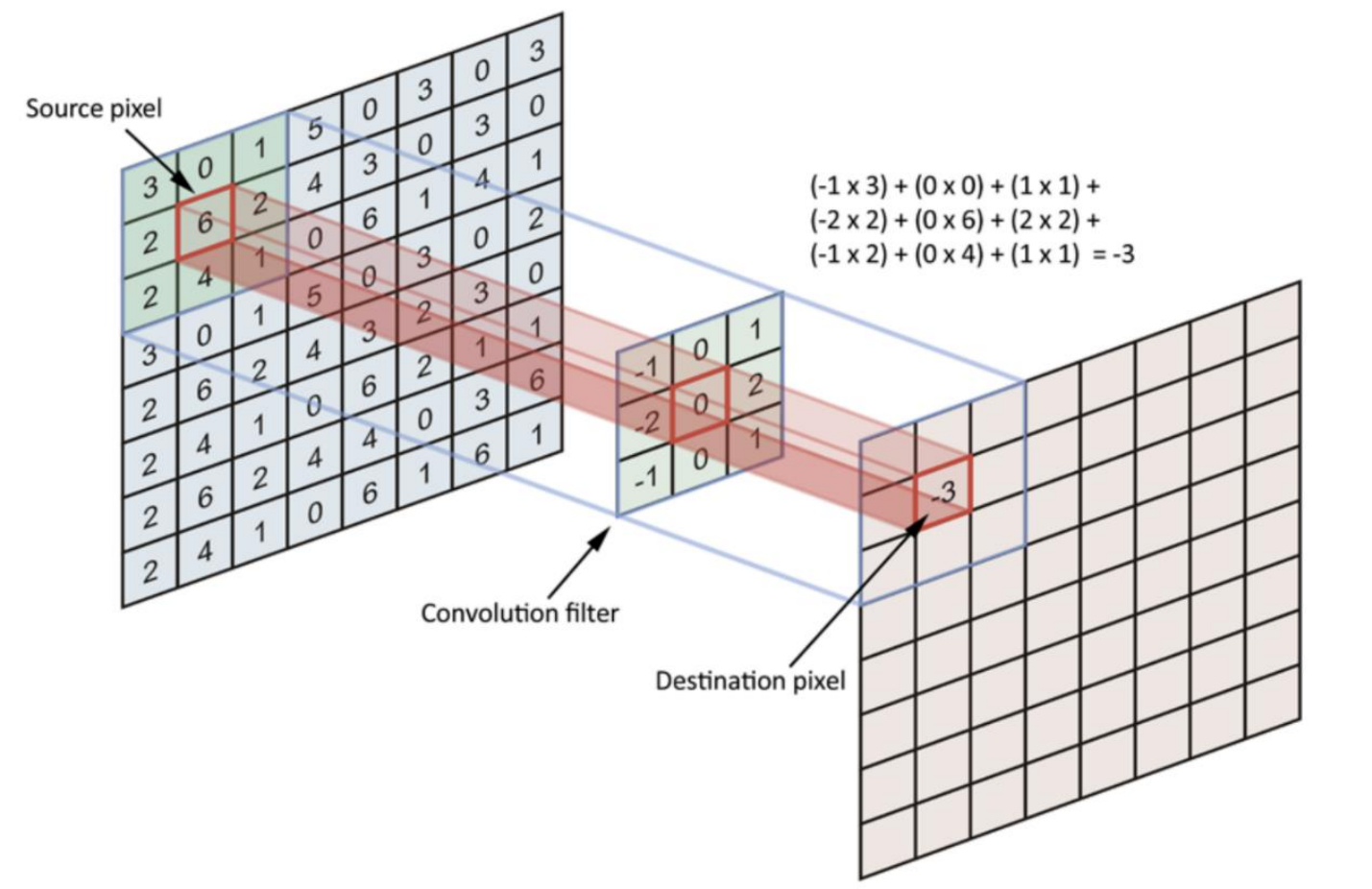
- Image Resize
  - Resize all image into identical width & height
  - Ensure all data will generate same number of neural nodes

- Data Augmentation
  - cropping, padding, and rotation



Convolution Neural Network (CNN) Model

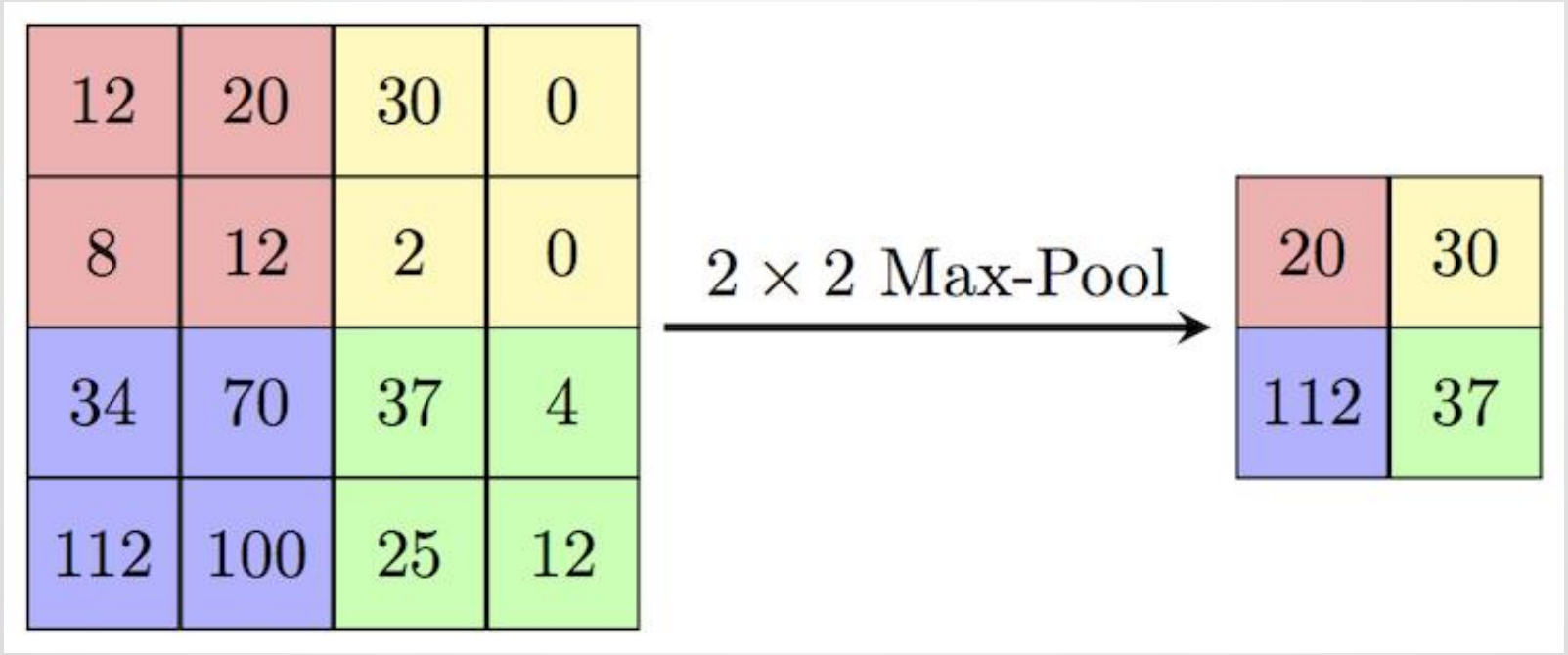
☐ Step 1 – Apply filter and activation function to input image



Source: https://towardsdatascience.com/simple-introduction-to-convolutional-neural-networks-cdf8d3077bac

Convolution Neural Network (CNN) Model

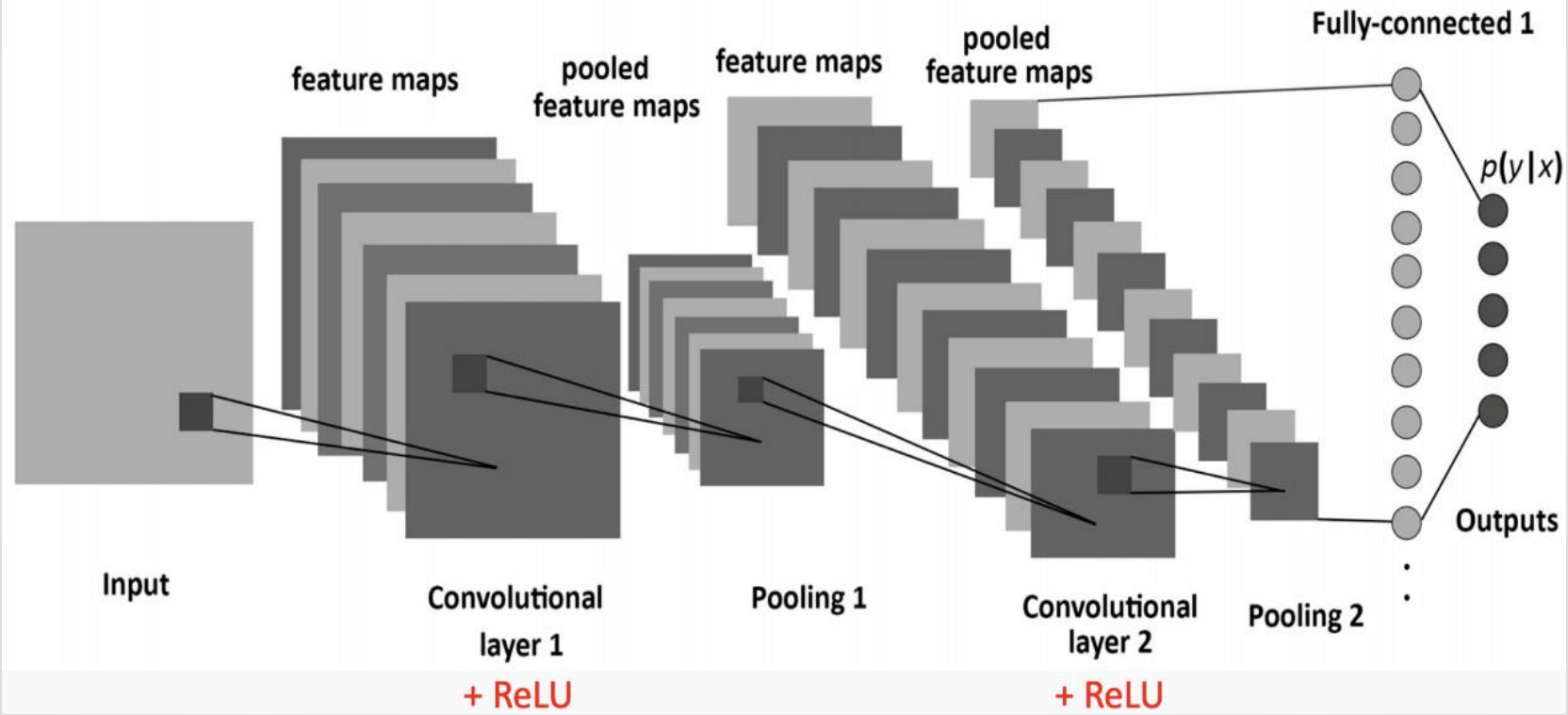
☐ Step 2 – Pooling



Source: https://computersciencewiki.org/index.php/Max-pooling_/_Pooling

Convolution Neural Network (CNN) Model

☐ Step 3 – Flattening (Fully Connected)

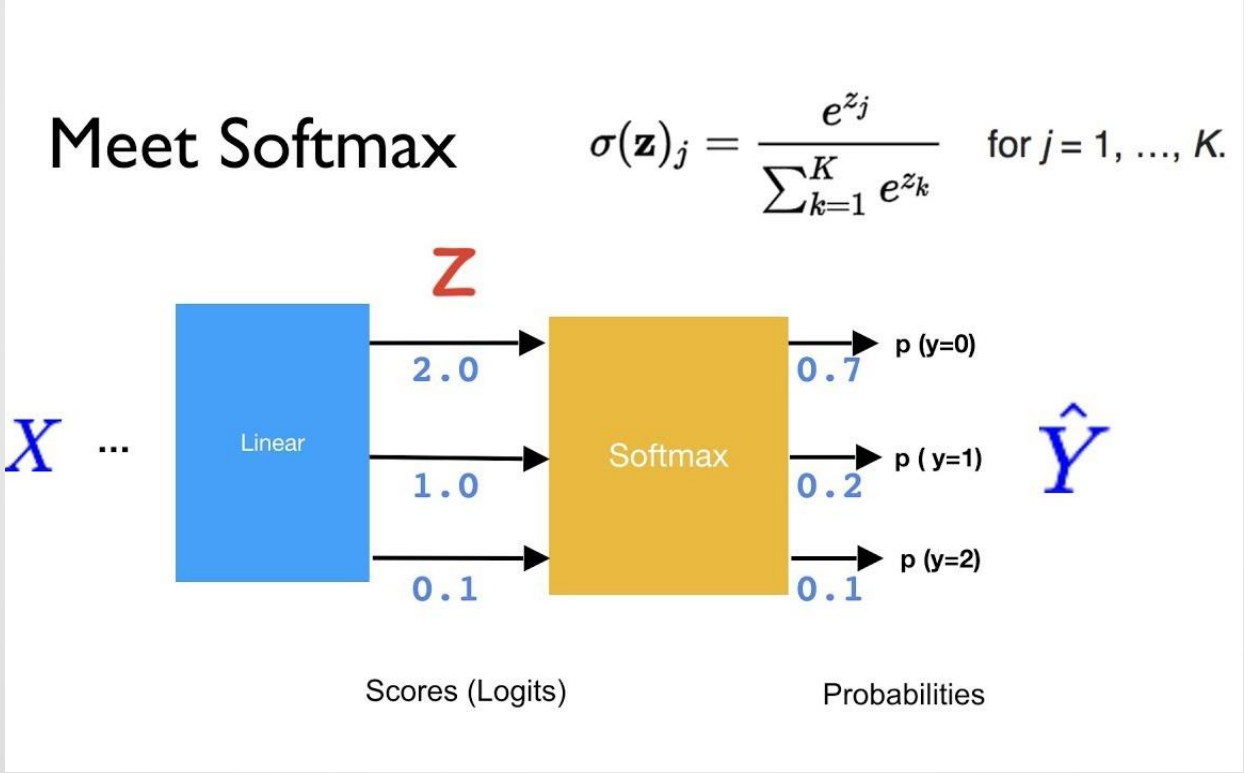


Source: https://towardsdatascience.com/simple-introduction-to-convolutional-neural-networks-cdf8d3077bac

Convolution Neural Network (CNN) Model

☐ Step 4 – Model training and Backpropagation

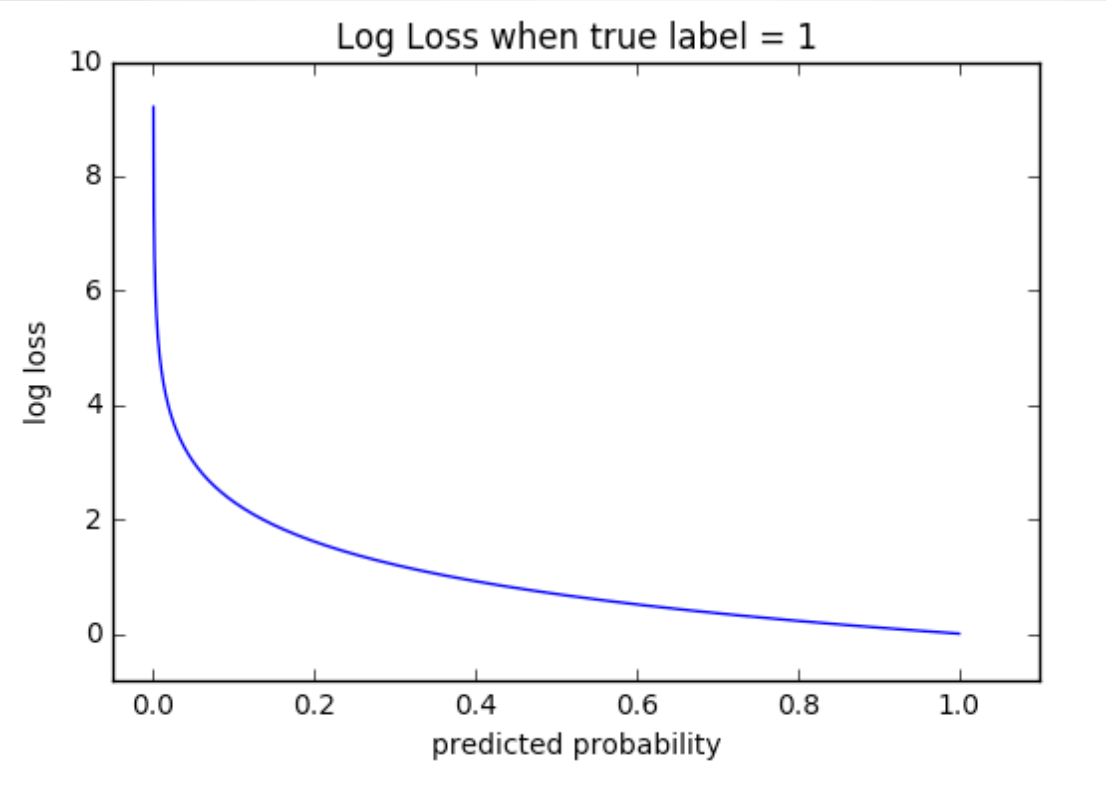
➢ Softmax



Source: https://github.com/hunkim/PyTorchZeroToAll

Convolution Neural Network (CNN) Model

- **Step 4 – Model training and Backpropagation**
  - **Crossentropy**



Convolution Neural Network (CNN) Model

- **Step 5 – Tunning of Hyperparameter**

Hyperparameter	Description
No. of Epoch	No. of times updating bias and weighting of CNN
Learning rate	Step size of each epoch
Activation Function	Fitting function mapping input and output
Dropout	%, no. and position of network dropout
No. of layer	Combination of Step 1 & Step 2
Filter	Size & padding of filters

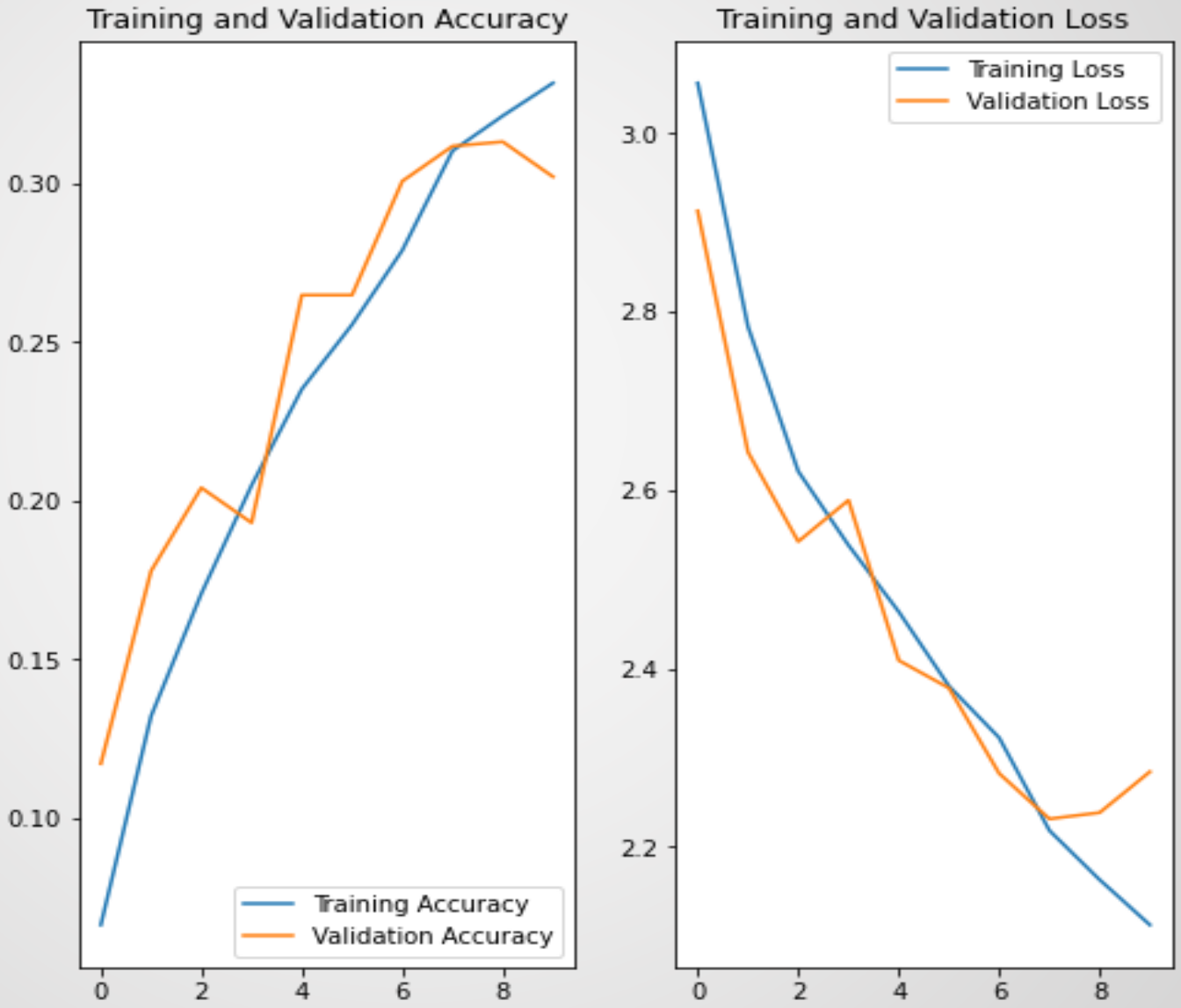
Model 1 - Self-Customized CNN Model

- Model Setting

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
sequential (Sequential)      (None, 224, 224, 3)       0
_________________________________________________________________
rescaling (Rescaling)        (None, 224, 224, 3)       0
_________________________________________________________________
conv2d (Conv2D)              (None, 224, 224, 16)      448
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 112, 112, 16)      0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 112, 112, 32)      4640
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 56, 56, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 56, 56, 64)        18496
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 28, 28, 64)        0
_________________________________________________________________
dropout (Dropout)            (None, 28, 28, 64)        0
_________________________________________________________________
flatten (Flatten)            (None, 50176)             0
_________________________________________________________________
dense (Dense)                (None, 128)               6422656
_________________________________________________________________
dense_1 (Dense)              (None, 20)                2580
=================================================================
Total params: 6,448,820
Trainable params: 6,448,820
Non-trainable params: 0
_________________________________________________________________
```

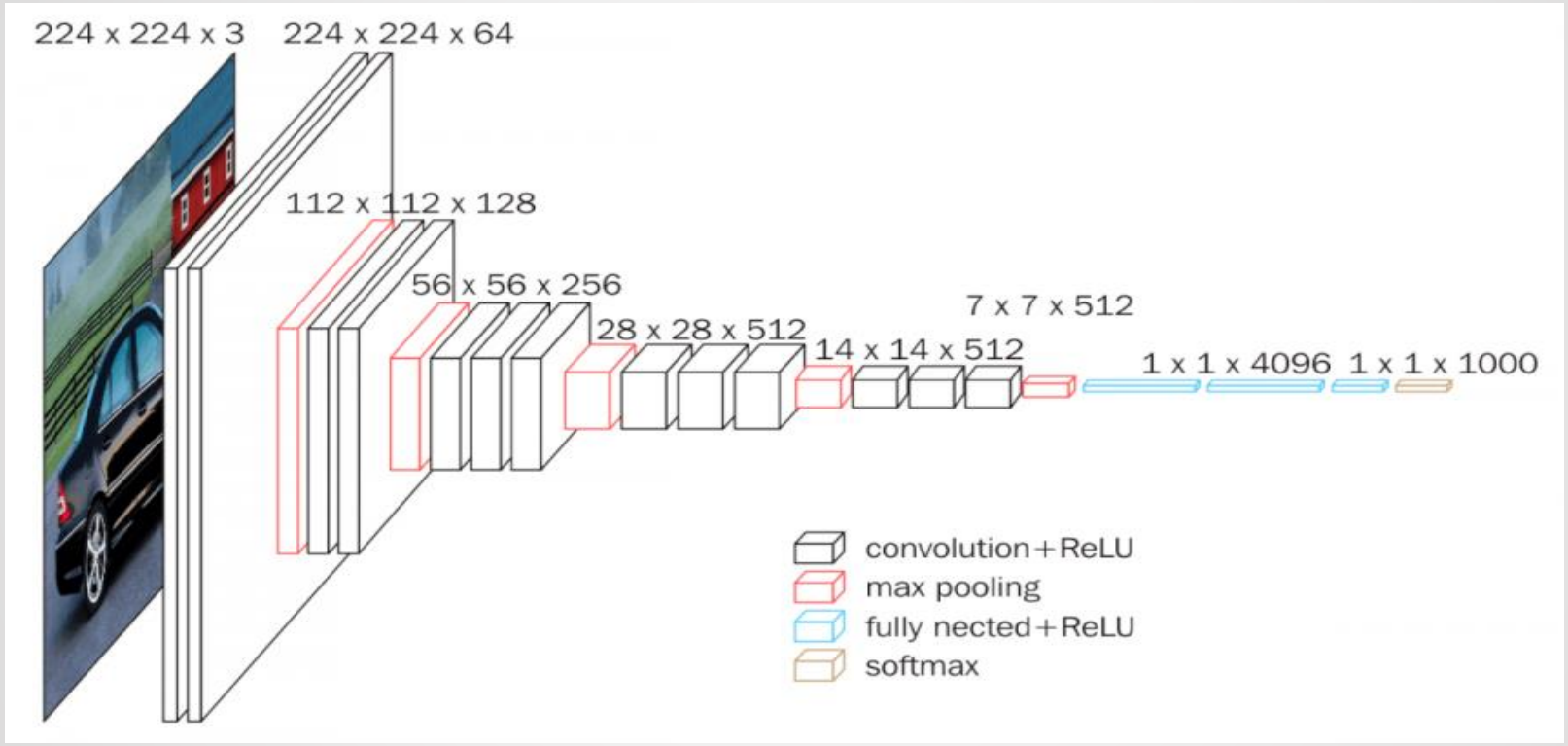
Model 1 - Self-Customized CNN Model

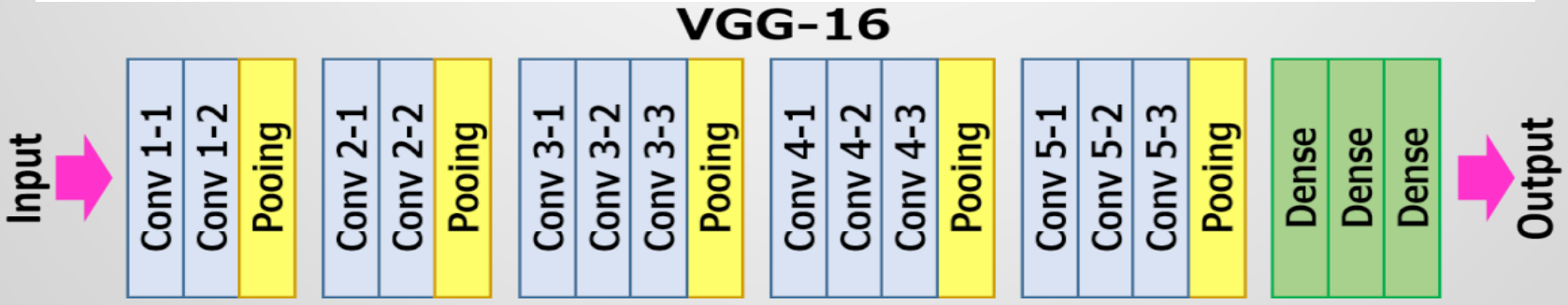
- Result
  - 30.2% Accuracy on validation data



Model 2 - VGG16 Model

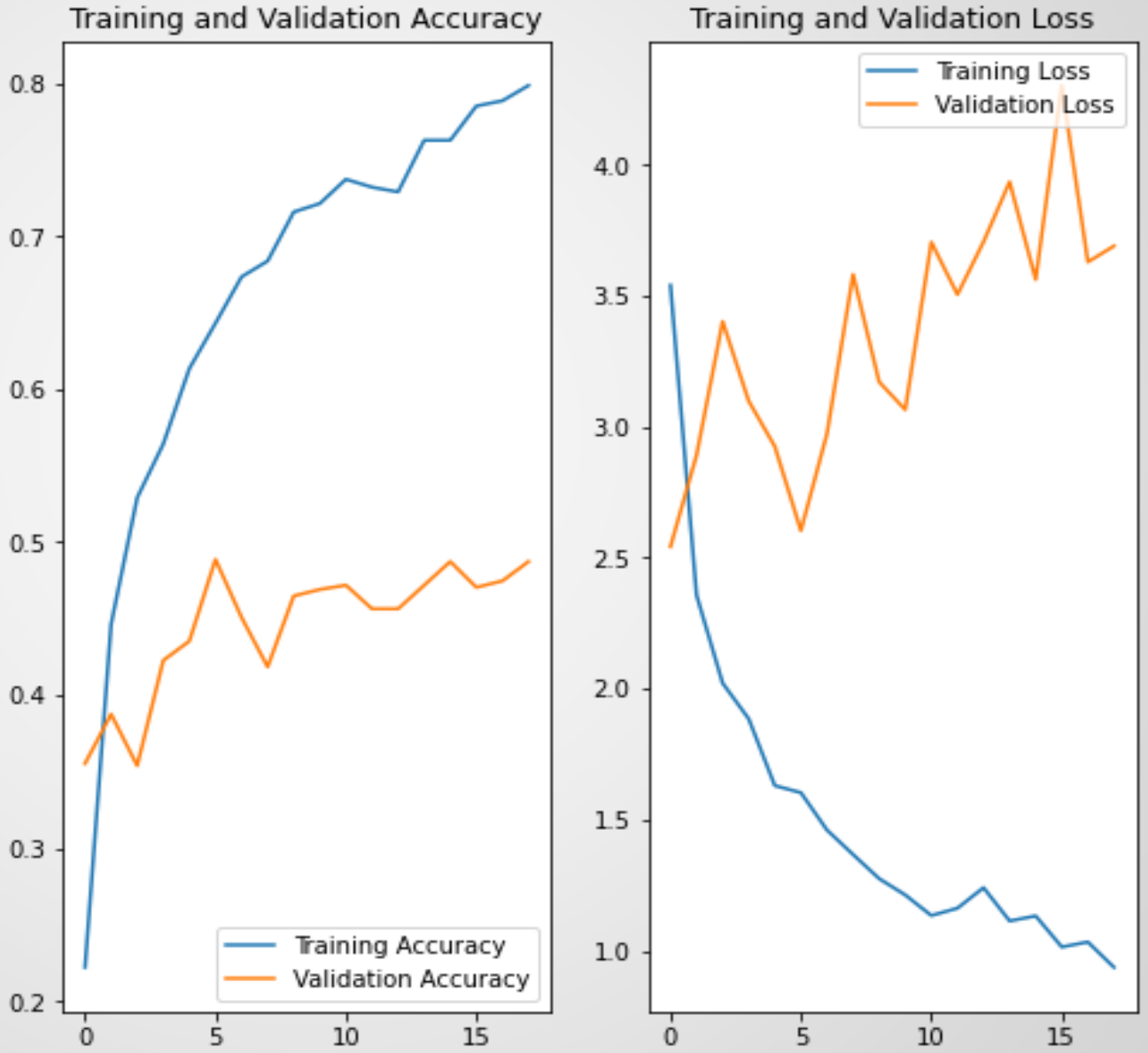
- Developed by by K. Simonyan and A. Zisserman from the University of Oxford





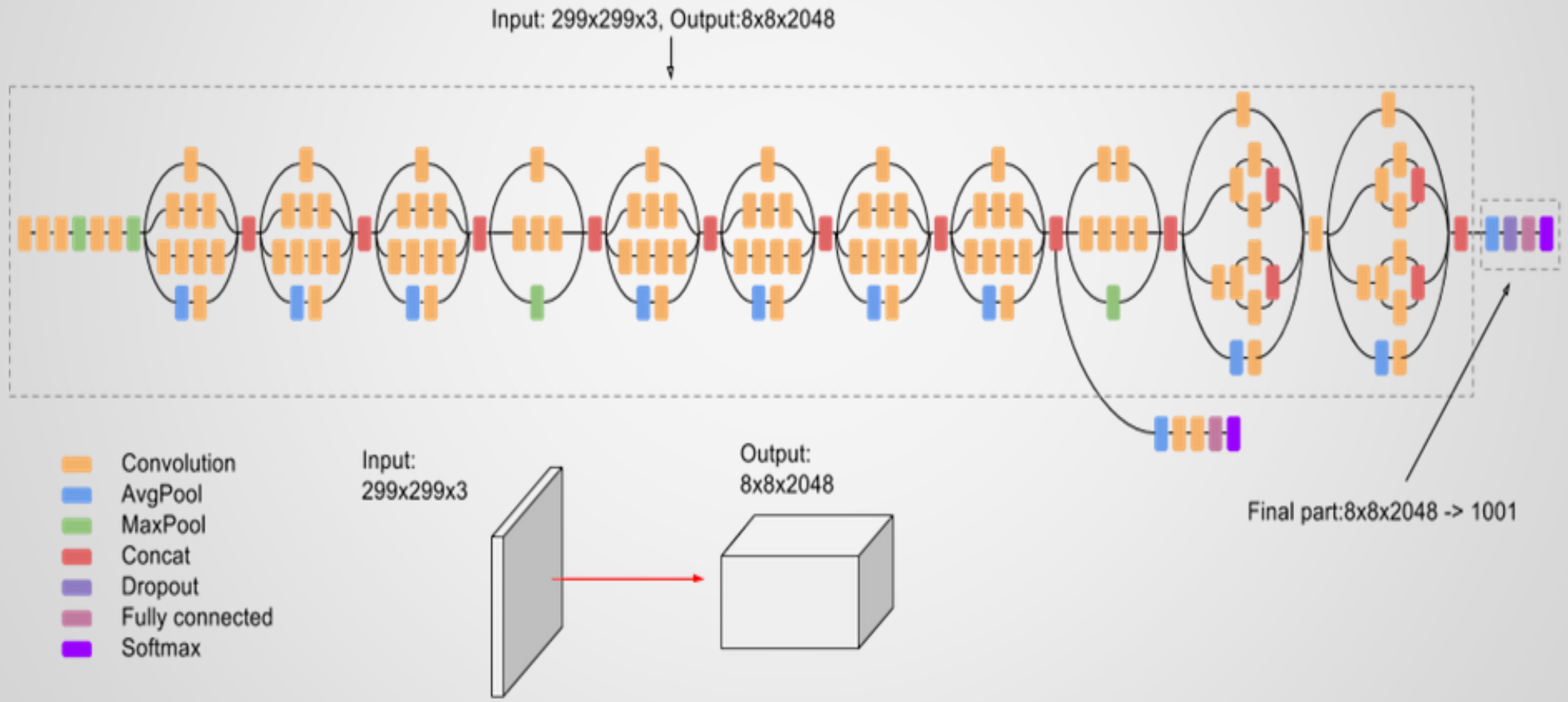
Model 2 - VGG16 Model

- Result
  - 48.7% Accuracy on validation data



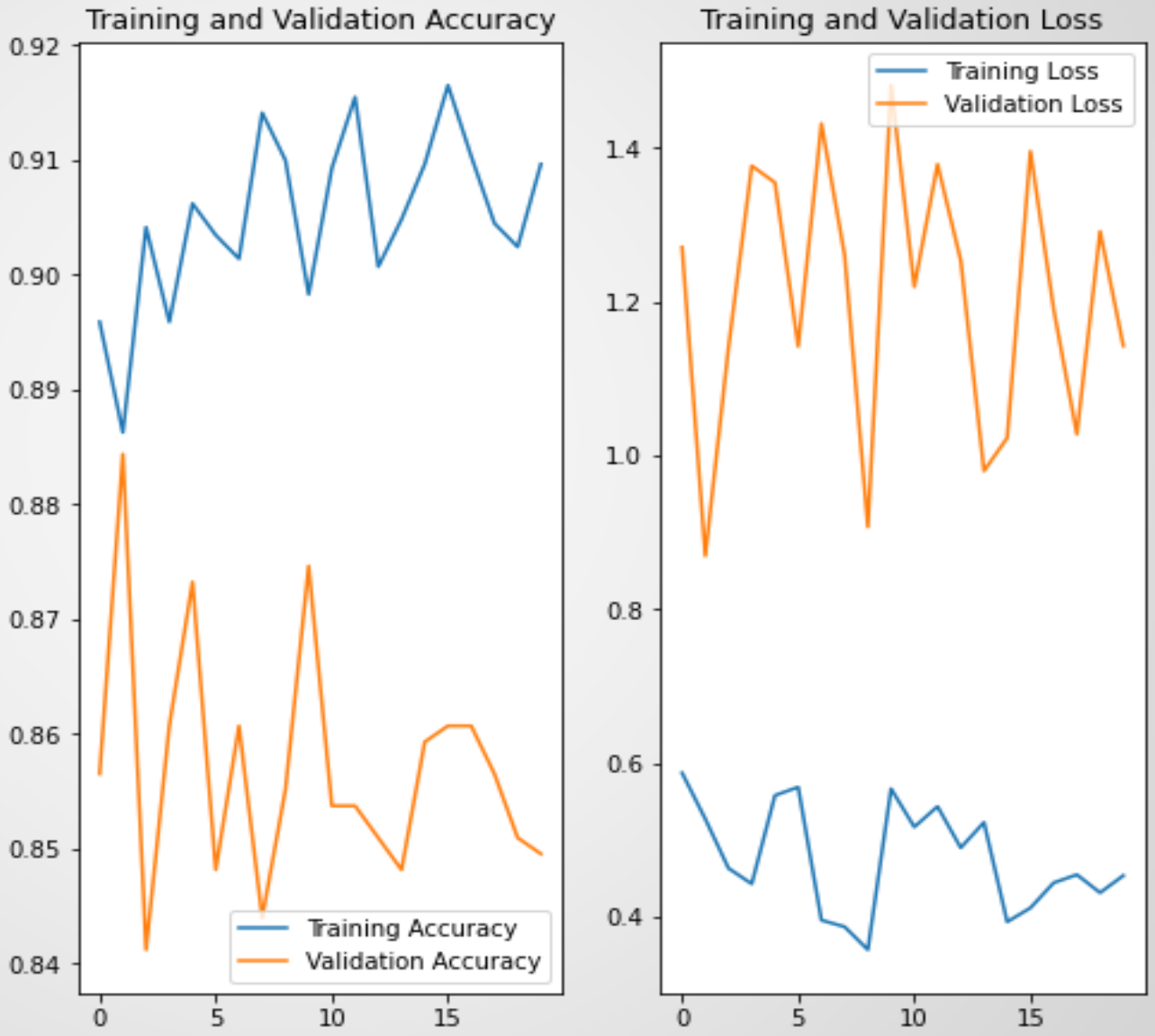
Model 3 – Inception V3 Model

- The third edition of Google's Inception Convolutional Neural Network



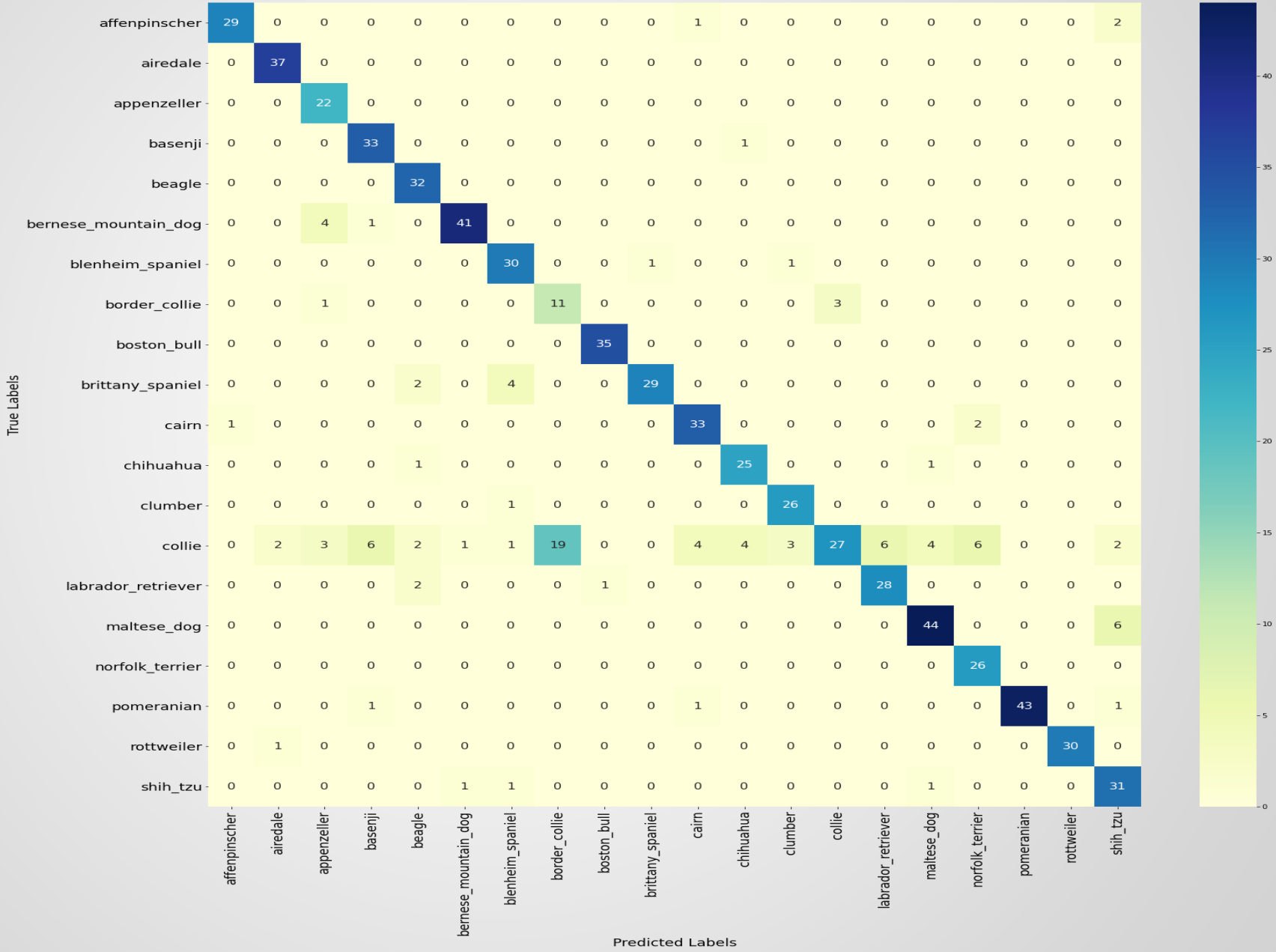
Model 3 – Inception V3 Model

- Result
  - 87.5% Accuracy on validation data



Model 3 – Inception V3 Model

- Result
  - Confusion Matrix



Model 3 – Inception V3 Model

- Result
  - Classification Report

	precision	recall	f1-score	support
0	0.97	0.91	0.94	32
1	0.93	1.00	0.96	37
2	0.73	1.00	0.85	22
3	0.80	0.97	0.88	34
4	0.82	1.00	0.90	32
5	0.95	0.89	0.92	46
6	0.81	0.94	0.87	32
7	0.37	0.73	0.49	15
8	0.97	1.00	0.99	35
9	0.97	0.83	0.89	35
10	0.85	0.92	0.88	36
11	0.83	0.93	0.88	27
12	0.87	0.96	0.91	27
13	0.90	0.30	0.45	90
14	0.82	0.90	0.86	31
15	0.88	0.88	0.88	50
16	0.76	1.00	0.87	26
17	1.00	0.93	0.97	46
18	1.00	0.97	0.98	31
19	0.74	0.91	0.82	34
accuracy			0.85	718
macro avg	0.85	0.90	0.86	718
weighted avg	0.87	0.85	0.84	718

Future Studies

- Google Colab or other methods can be applied to solve the computation problems.

- Collie and border collie are performed bad in the best mode. Further studies can be carried for further improvement.

- More time and effort can be used for tuning of hyperparameter

- More problems can be solved by transforming models in this project. This project only provides baseline models for study and reference.

Conclusion

- This project adopts CNN model to identify 20 dog breeds from their images and obtains an accuracy of 84.5% on validation data.
- With the same CNN concepts but different combination of layers and hyperparameters. The result of model will have huge different.
- Establish a baseline model and further transformation of the model can be performed to solve other real-world problems

Q & A