Selected-2 Project Documentation

-Faculty Name: Faculty of computers and artificial Intelligence, Helwan University

-Course Name: Selected Topics in Computer Science

-Team number: 17

Team Members:

1) يوسف أحمد عبد الغني رجب	*ID: 201900975*
2) مصطفى عامر محمد محمود	*ID: 201900821*
3) معتصم محمود احمد احمد محرم	*ID:201900843*
4) مصطفي ابراهيم حسن ابراهيم	*ID: 201900807*
5) يوسف زكريا محمد عبد الله	*ID: 201900981*
6) محمد وائل محمد أحمد	*ID: 201900750*

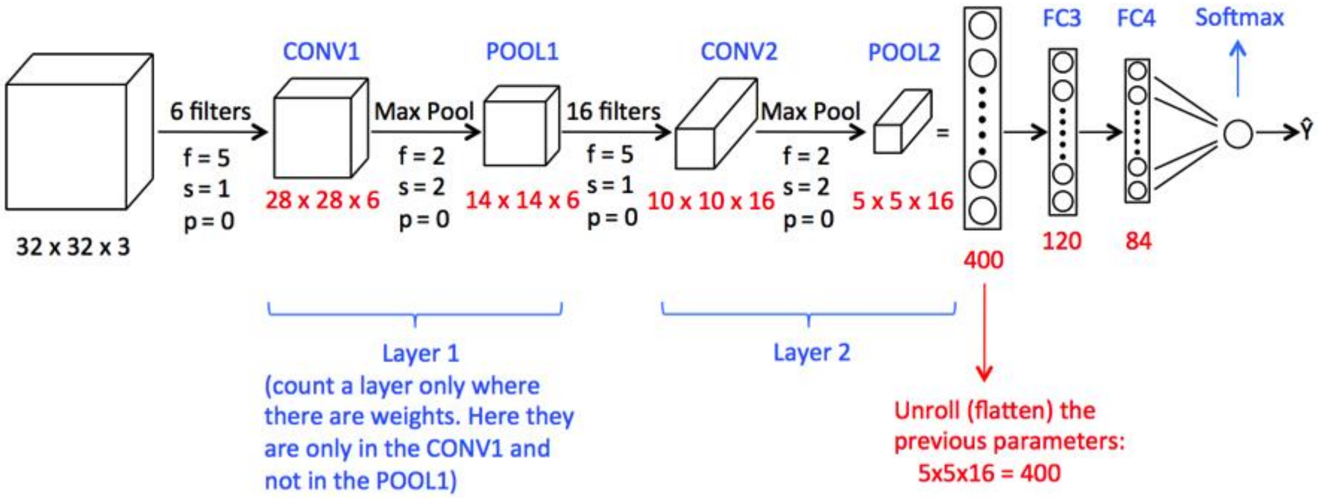
-Paper Details:

- *Author Name: Amruta Kadlaskar*

- *Paper Name: Image Classification using Convolutional Neural Network with Python*

- *Year of Publication: May 2021*

1) Architecture used in the paper:



-The first thing you should do is feed the pixels of the image in the form of arrays to the input layer of the neural network (MLP networks used to classify such things). The hidden layers carry Feature Extraction by performing various calculations and operations. There are multiple hidden layers like the convolution, the ReLU, and the pooling layer that performs feature extraction from your image. So finally, there is a fully connected layer that you can see which identifies the exact object in the image.

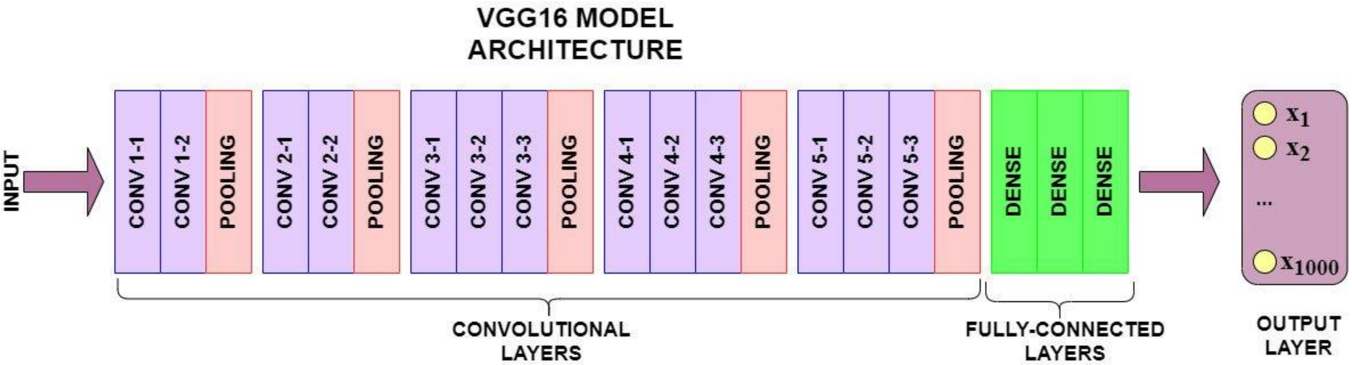
2) Dataset Details:

- The dataset detects different car brand based on image classification.

3) Implementation Details:

- we are going to import the necessary libraries which are required for performing CNN tasks.

- we are using VGG16 architecture



VGG16 is a convolution neural net (CNN) architecture which was used to win ILSVR(Imagenet) competition in 2014. It is considered to be one of the excellent vision model architecture till date. Most unique thing about VGG16 is that instead of having a large number of hyper-parameter they focused on having convolution layers of 3x3 filter with a stride 1 and always used same padding and maxpool layer of 2x2 filter of stride 2. It follows this arrangement of convolution and max pool layers consistently throughout the whole architecture. In the end it has 2 FC(fully connected layers) followed by a softmax for output. The 16 in VGG16 refers to it has 16 layers that have weights. This network is a pretty large network and it has about 138 million (approx) parameters.

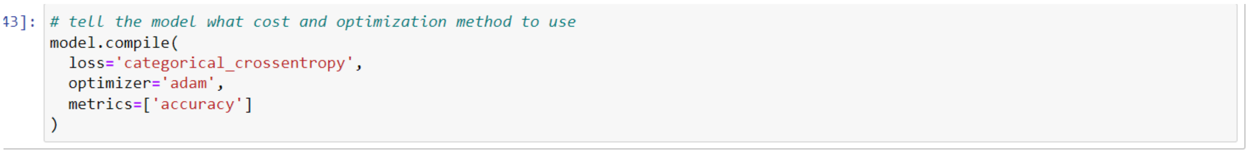
-We make a model summary

Total params: 14,789,955

Trainable params: 75,267

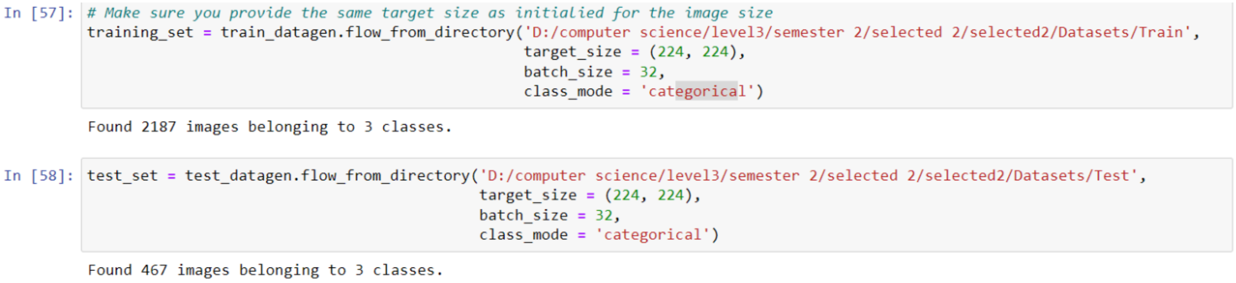
Non-trainable params: 14,714,688

-We are required to specify optimizers.



```
43]: # tell the model what cost and optimization method to use
     model.compile(
       loss='categorical_crossentropy',
       optimizer='adam',
       metrics=['accuracy']
     )
```

-We will see how to set the data directory and generate image data.



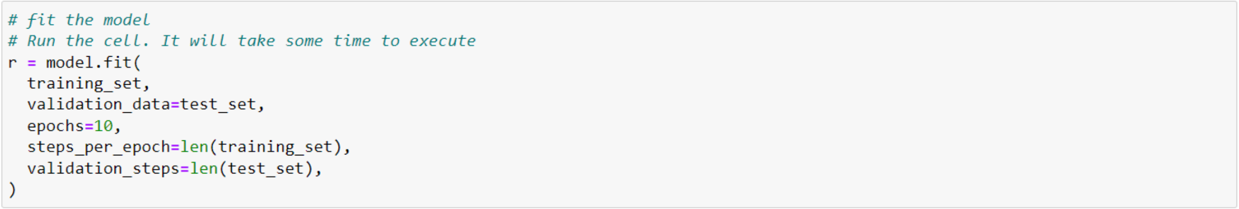
```
In [57]: # Make sure you provide the same target size as initialied for the image size
         training_set = train_datagen.flow_from_directory('D:/computer science/level3/semester 2/selected 2/selected2/Datasets/Train',
                                                          target_size = (224, 224),
                                                          batch_size = 32,
                                                          class_mode = 'categorical')

         Found 2187 images belonging to 3 classes.

In [58]: test_set = test_datagen.flow_from_directory('D:/computer science/level3/semester 2/selected 2/selected2/Datasets/Test',
                                                     target_size = (224, 224),
                                                     batch_size = 32,
                                                     class_mode = 'categorical')

         Found 467 images belonging to 3 classes.
```

-Final step of the fitting model.



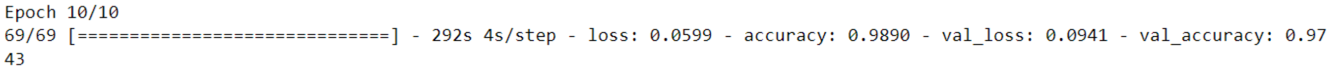
```
# fit the model
# Run the cell. It will take some time to execute
r = model.fit(
  training_set,
  validation_data=test_set,
  epochs=10,
  steps_per_epoch=len(training_set),
  validation_steps=len(test_set),
)
```

-Ratio Used in Training: 80% (2187 Images)

-Ratio Used in Test: 20% (467 Images)

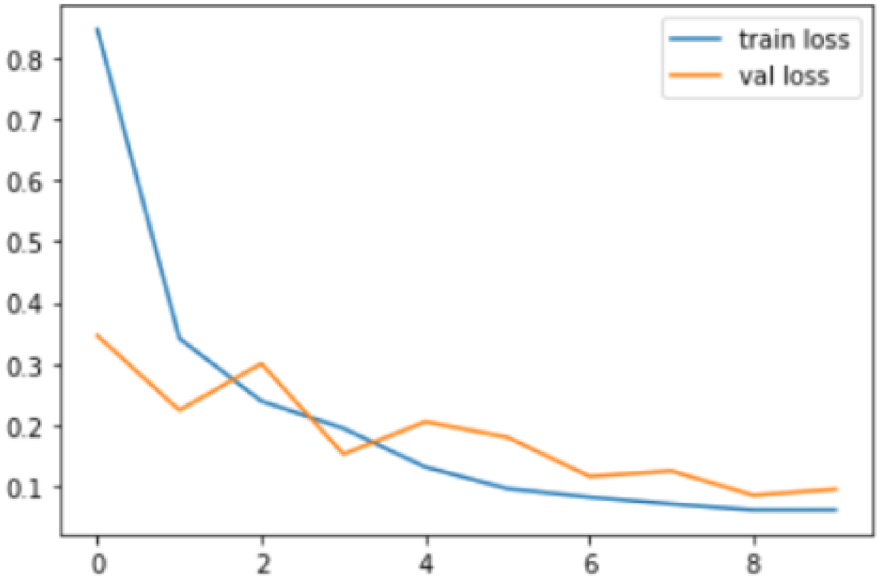
4) Results and Visualization:

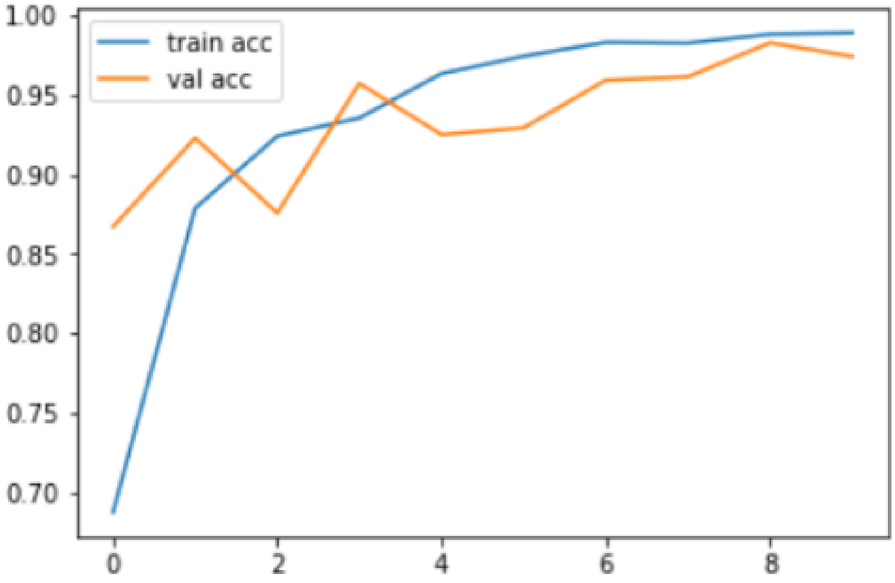
- Results:



```
Epoch 10/10
69/69 [==============================] - 292s 4s/step - loss: 0.0599 - accuracy: 0.9890 - val_loss: 0.0941 - val_accuracy: 0.97
43
```

- Visualization:





-Feature Extracting: