Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

BCSE332P

DEEP LEARNING-Lab

Experiment:5

(1.) Transfer Learning:

Transfer learning with **convolutional neural networks** (CNNs) has revolutionized the field of **computer vision** by enabling the ***reuse of pre-trained models on new, related tasks***.

This powerful technique leverages the **knowledge learned from large-scale datasets**, allowing for **faster** and **more accurate model** training, even with ***limited labeled data***

Experiment:5

(1.) Transfer Learning:

The ***reuse*** of a pre-trained model on a ***new problem*** is known as **transfer learning** in machine learning.

A machine uses the knowledge learned from a prior assignment to ***increase prediction*** about a new task in transfer learning.

Experiment:5

(1.) Transfer Learning:

For example:

use the information gained during training to distinguish beverages ***when training a classifier to predict*** whether an image contains cuisine.

Experiment:5

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

(1.) Transfer Learning:

Why?

Due to the substantial CPU power demand, practitioners typically apply transfer learning in tasks such as computer vision and natural language processing, such as sentiment analysis.

Experiment:5

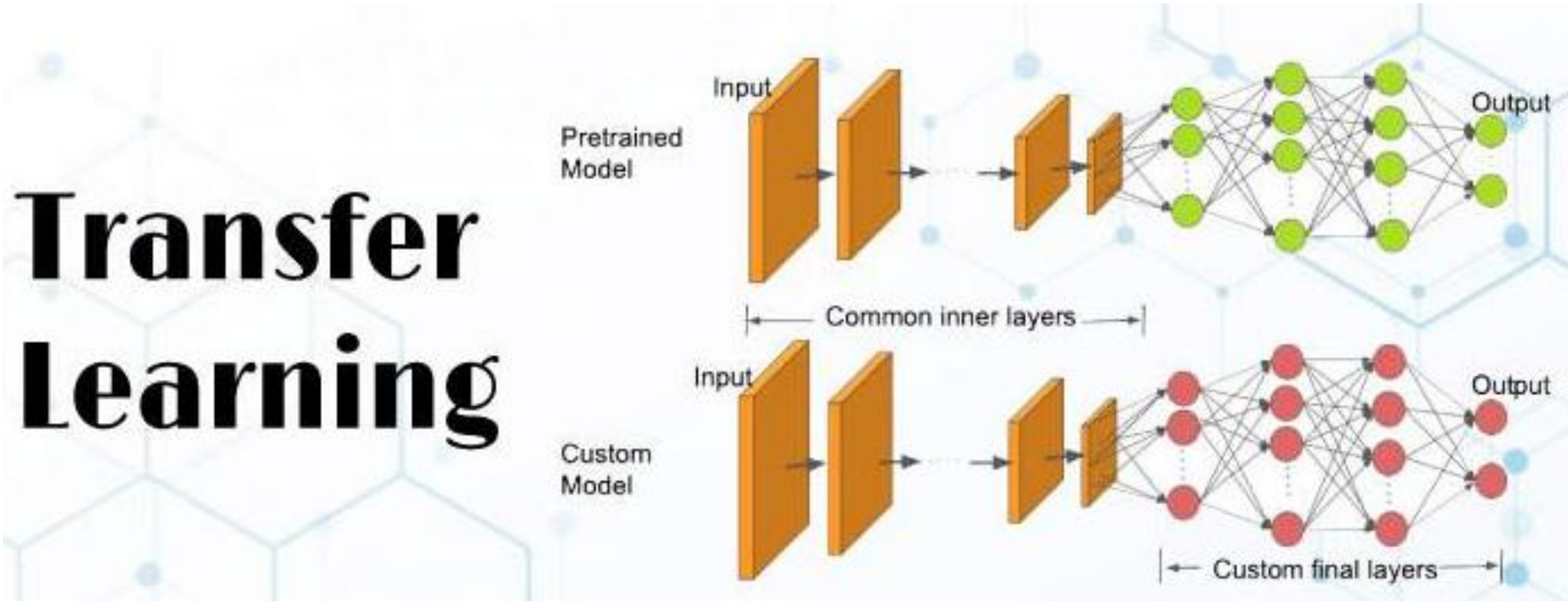
(1.) Transfer Learning:

How Transfer Learning Works?

- In computer vision, neural networks typically aim to ***detect edges in the first layer***, forms (structures/figures) in the middle layer, and task-specific features in the latter layers.
- In transfer learning in CNN, we utilize the **early** and **central layers**, while only retraining the latter layers.
- The model leverages labeled data from its original training task.

Experiment:5

(1.) Transfer Learning:



Experiment:5

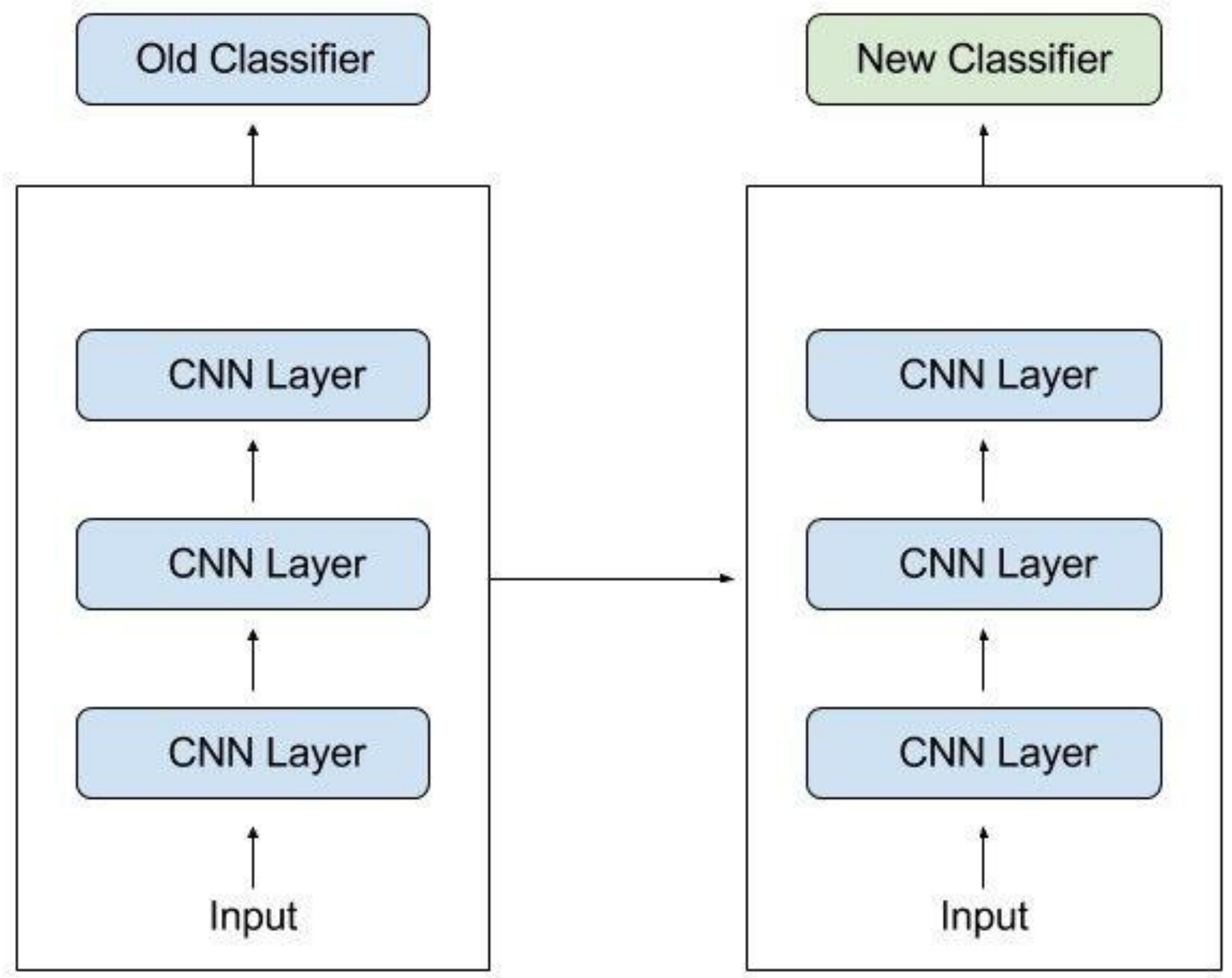
(1.) Transfer Learning:

Why Should You Use Transfer Learning?

- ***Transfer learning*** provides several advantages, including decreased ***training time***, ***enhanced neural network performance*** (in many cases), and the ***ability to work effectively with limited data***.
- Training a neural model from the ground up usually requires substantial data, which may not always be available.
- Transfer learning in CNN addresses this challenge effectively.

Experiment:5

(1.) Transfer Learning:



Experiment:5

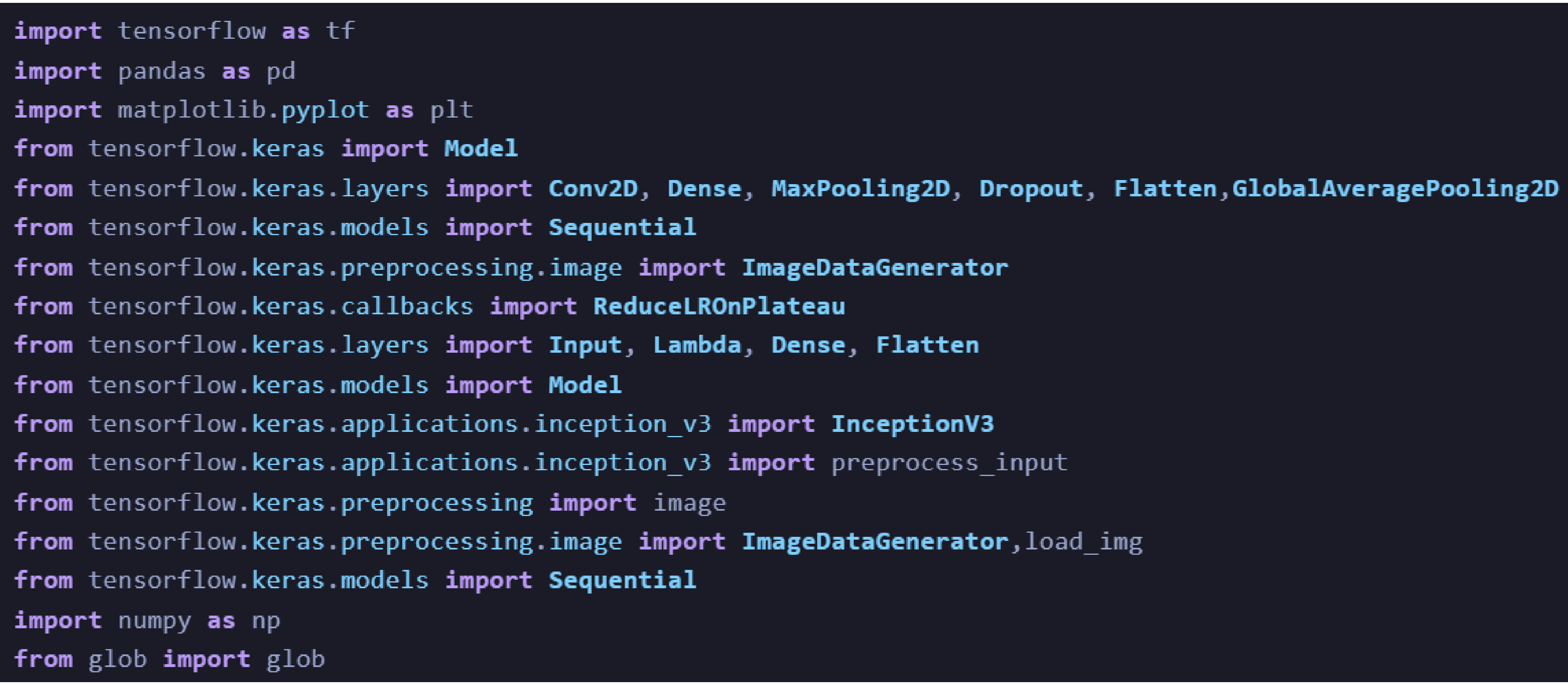
(1.) Transfer Learning:

Transfer learning in CNN leverages pre-trained models to achieve strong performance with limited training data, crucial in fields like natural language processing with vast labeled datasets.

It reduces training time significantly compared to building complex models from scratch, which can take days or weeks.

Experiment:5

Code Implementation of Transfer Learning with Python



```python
import tensorflow as tf
import pandas as pd
import matplotlib.pyplot as plt
from tensorflow.keras import Model
from tensorflow.keras.layers import Conv2D, Dense, MaxPooling2D, Dropout, Flatten,GlobalAveragePooling2D
from tensorflow.keras.models import Sequential
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.callbacks import ReduceLROnPlateau
from tensorflow.keras.layers import Input, Lambda, Dense, Flatten
from tensorflow.keras.models import Model
from tensorflow.keras.applications.inception_v3 import InceptionV3
from tensorflow.keras.applications.inception_v3 import preprocess_input
from tensorflow.keras.preprocessing import image
from tensorflow.keras.preprocessing.image import ImageDataGenerator,load_img
from tensorflow.keras.models import Sequential
import numpy as np
from glob import glob
```

Experiment:5

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

Code Implementation of Transfer Learning with Python

Uploading Data via Kaggle API

```
from google.colab import files
files.upload()
```

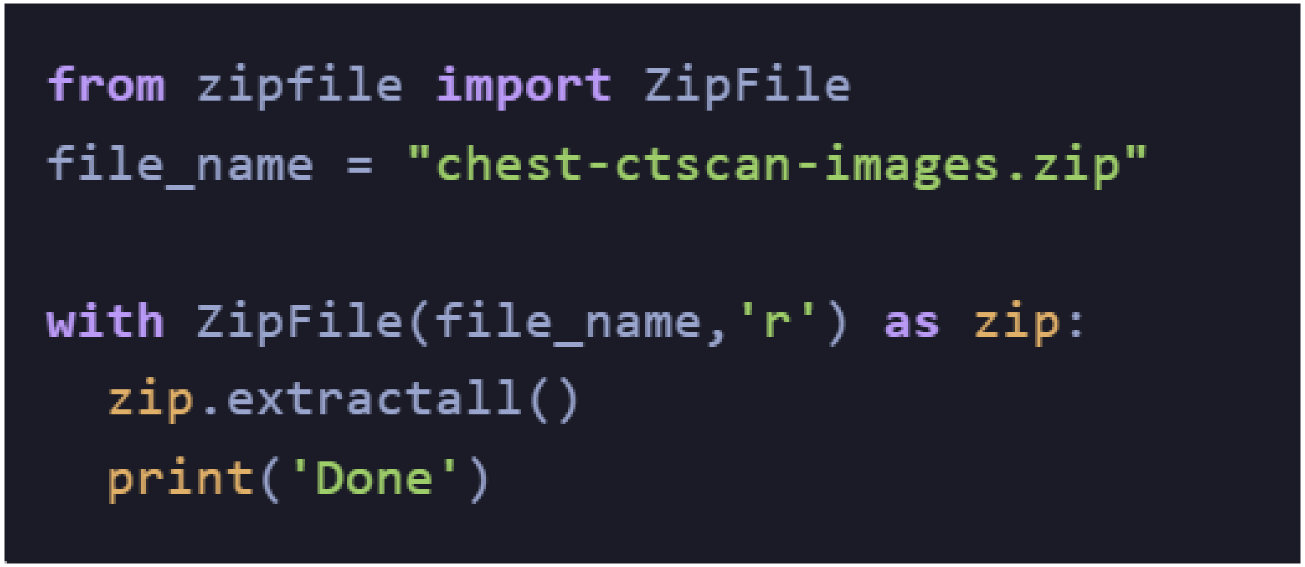
Saving kaggle.json to kaggle.json

```
!mkdir -p ~/.kaggle
!cp kaggle.json ~/.kaggle/

!chmod 600 ~/.kaggle/kaggle.json
```

```
!kaggle datasets download -d mohamedhanyyy/chest-ctscan-images #downloading data from kagg
```

Experiment:5

Code Implementation of Transfer Learning with Python



```python
from zipfile import ZipFile
file_name = "chest-ctscan-images.zip"

with ZipFile(file_name,'r') as zip:
  zip.extractall()
  print('Done')
```

Experiment:5

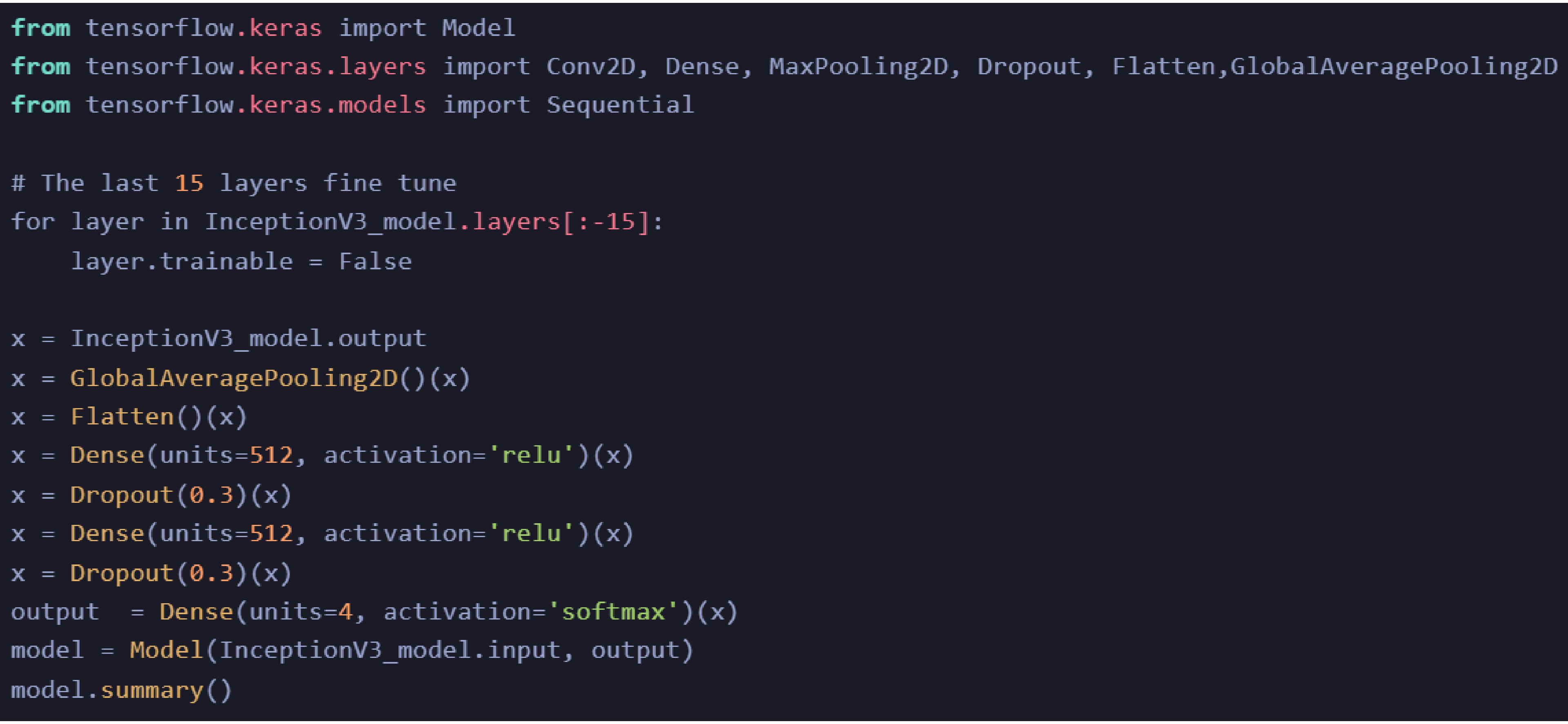
Code Implementation of Transfer Learning with Python

Designing Our CNN Model with Help of Pre-Trained Model

```
InceptionV3_model = tf.keras.applications.InceptionV3(weights='imagenet',
                                                      include_top=False, input_shape=(224, 224, 3))
```

Experiment:5

Code Implementation of Transfer Learning with Python

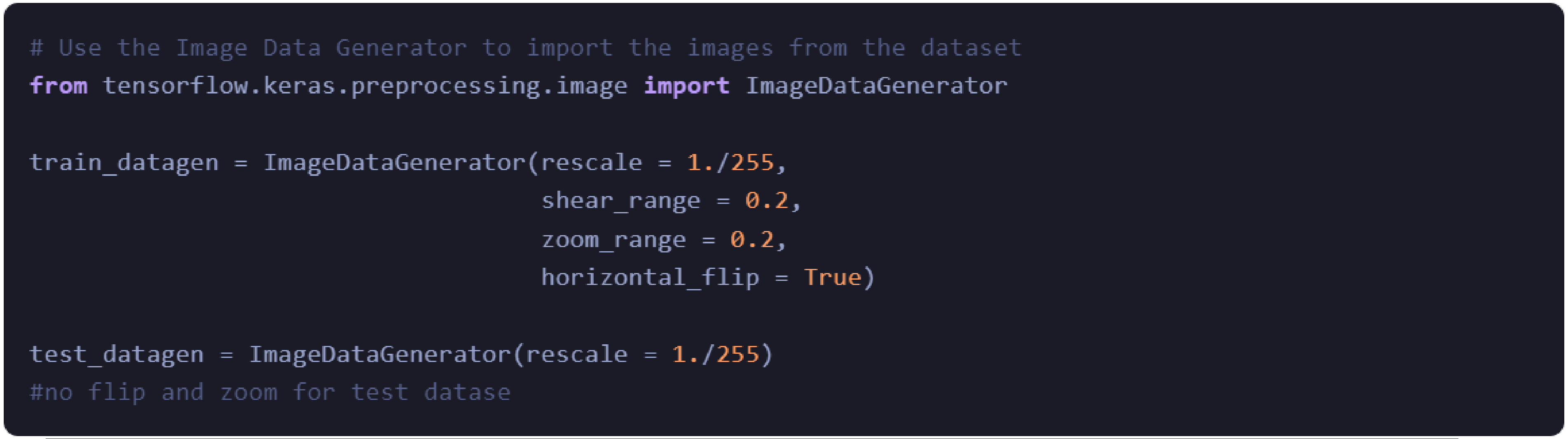


```python
from tensorflow.keras import Model
from tensorflow.keras.layers import Conv2D, Dense, MaxPooling2D, Dropout, Flatten,GlobalAveragePooling2D
from tensorflow.keras.models import Sequential

# The last 15 layers fine tune
for layer in InceptionV3_model.layers[:-15]:
    layer.trainable = False

x = InceptionV3_model.output
x = GlobalAveragePooling2D()(x)
x = Flatten()(x)
x = Dense(units=512, activation='relu')(x)
x = Dropout(0.3)(x)
x = Dense(units=512, activation='relu')(x)
x = Dropout(0.3)(x)
output  = Dense(units=4, activation='softmax')(x)
model = Model(InceptionV3_model.input, output)
model.summary()
```

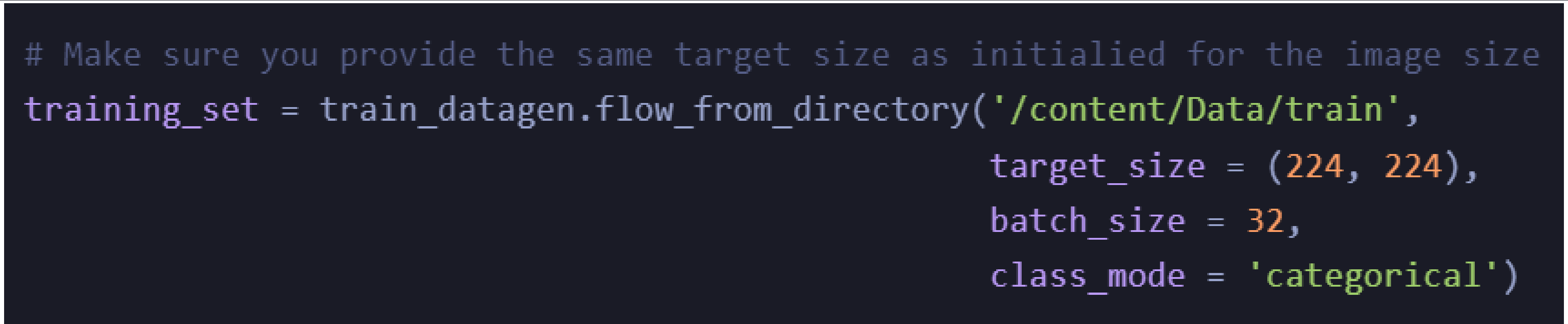
Experiment:5

Code Implementation of Transfer Learning with Python

Image Augmentation(For Preventing the Issue of Overfitting)



```
# Use the Image Data Generator to import the images from the dataset
from tensorflow.keras.preprocessing.image import ImageDataGenerator

train_datagen = ImageDataGenerator(rescale = 1./255,
                                   shear_range = 0.2,
                                   zoom_range = 0.2,
                                   horizontal_flip = True)

test_datagen = ImageDataGenerator(rescale = 1./255)
#no flip and zoom for test datase
```



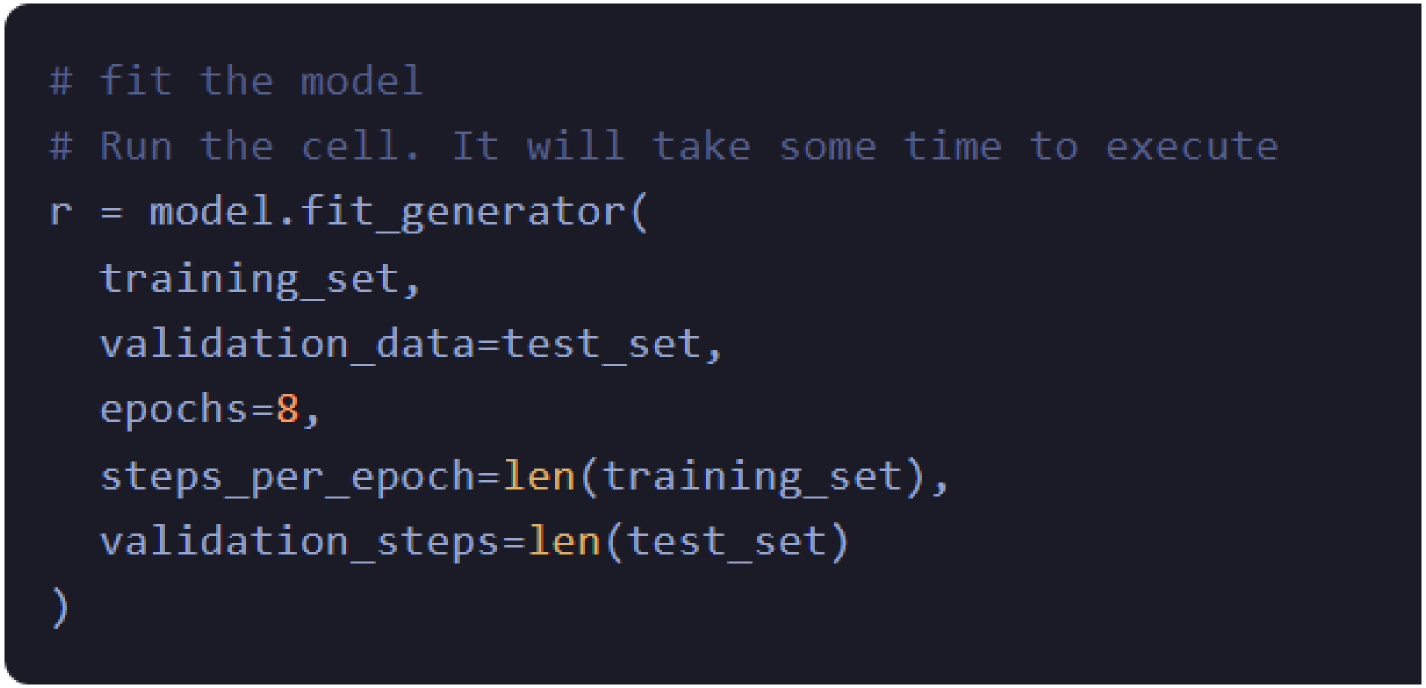
```
# Make sure you provide the same target size as initialied for the image size
training_set = train_datagen.flow_from_directory('/content/Data/train',
                                                 target_size = (224, 224),
                                                 batch_size = 32,
                                                 class_mode = 'categorical')
```

Experiment:5

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

Code Implementation of Transfer Learning with Python

Training Our Model

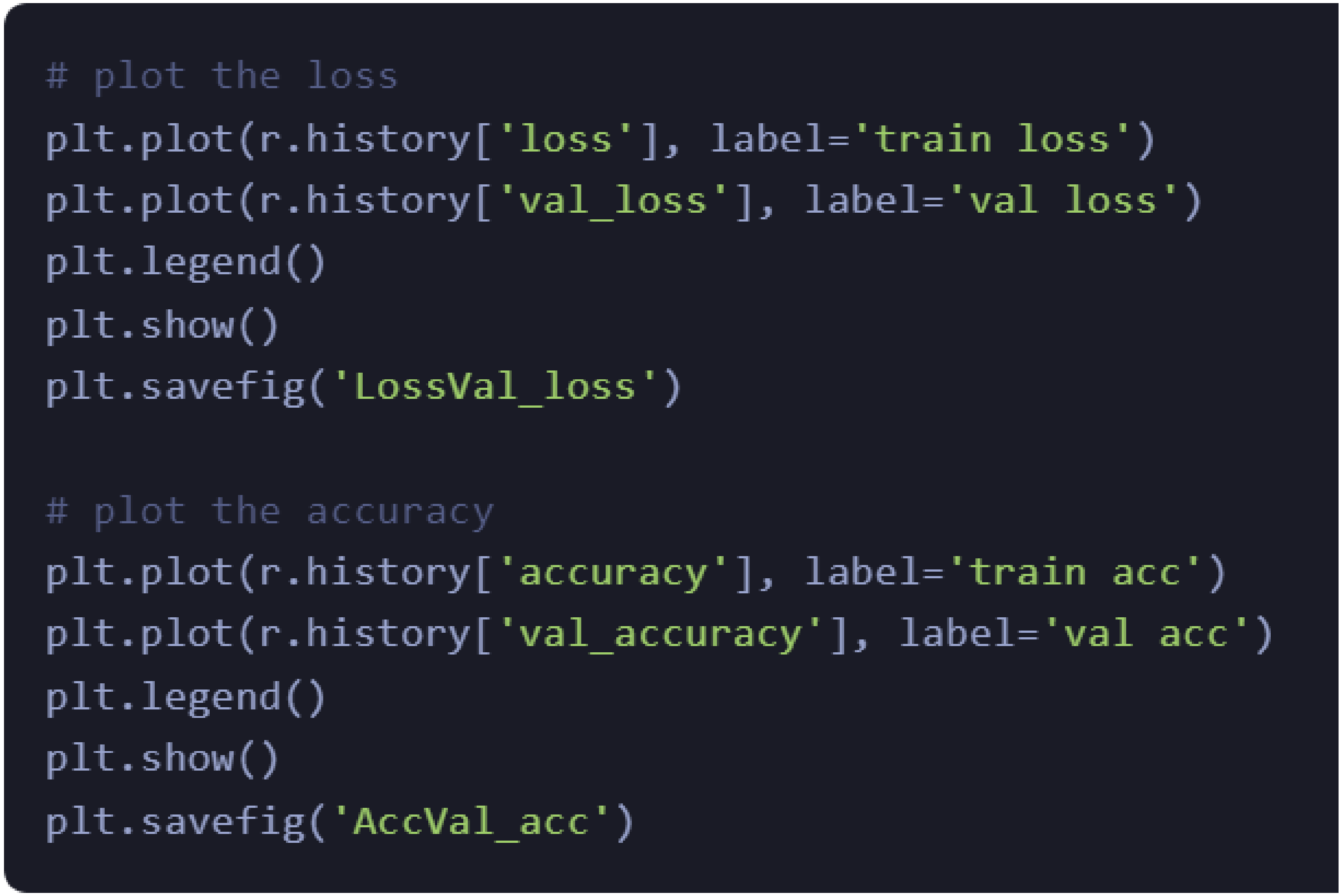


```
# fit the model
# Run the cell. It will take some time to execute
r = model.fit_generator(
  training_set,
  validation_data=test_set,
  epochs=8,
  steps_per_epoch=len(training_set),
  validation_steps=len(test_set)
)
```

Experiment:5

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

Code Implementation of Transfer Learning with Python



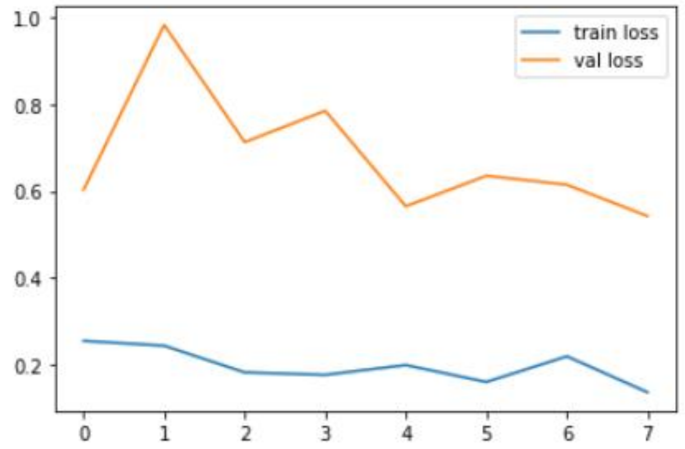
```python
# plot the loss
plt.plot(r.history['loss'], label='train loss')
plt.plot(r.history['val_loss'], label='val loss')
plt.legend()
plt.show()
plt.savefig('LossVal_loss')

# plot the accuracy
plt.plot(r.history['accuracy'], label='train acc')
plt.plot(r.history['val_accuracy'], label='val acc')
plt.legend()
plt.show()
plt.savefig('AccVal_acc')
```

Experiment:5

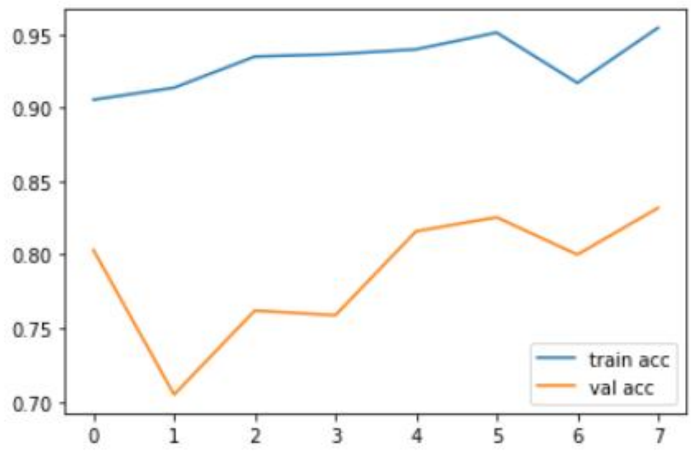
Code Implementation of Transfer Learning with Python

Making Predictions

```python
import numpy as np
y_pred = np.argmax(y_pred, axis=1)
y_pred
```



Source: Loss plot


train acc
val acc