**README:**
**Name: Crystal Low**
**Class: EE 104**

**LINK TO VIDEO:**
**Video 1:** https://www.youtube.com/watch?v=scXhICnOWxk (This video is to show that I ran the CNN and got the results shown in the github and will be explained what I did in the next video)
**Video 2:** https://www.youtube.com/watch?v=RlymMF6dBVA (This video is more in-depth with CNN and the image recognition)
**Video 3:** https://www.youtube.com/watch?v=gfnq_h-kAMU (This is a video for the balloon flight demonstration)

**Github Link:** https://github.com/crystalalow/EE104Lab8_Low_Crystal/tree/main
**Download and Extract File:** Lab8_Low_Crystal

**CNN**

In this code, the goal was to improve the accuracy of the images that were provided. The layers were fixed to accommodate the accuracy. The results will show the final accuracy that was captured. I have run the code multiple times and this is the best I could get.

**Import Packages:**

Import TensorFlow

```
[ ] import tensorflow as tf
    import sys
    import numpy as np
    # baseline model with dropout
    #data augmentation on the cifar10 dataset
    from keras.datasets import cifar10
    from tensorflow.keras.utils import to_categorical
    from keras.models import Sequential
    from keras.layers import Conv2D
    from keras.layers import MaxPooling2D
    from keras.layers import Dense
    from keras.layers import Flatten
    from tensorflow.keras.optimizers import SGD
    from keras.preprocessing.image import ImageDataGenerator
    from keras.layers import Dropout
    from keras.layers import BatchNormalization
    from tensorflow.keras import datasets, layers, models
    import matplotlib.pyplot as plt
```

**References:**

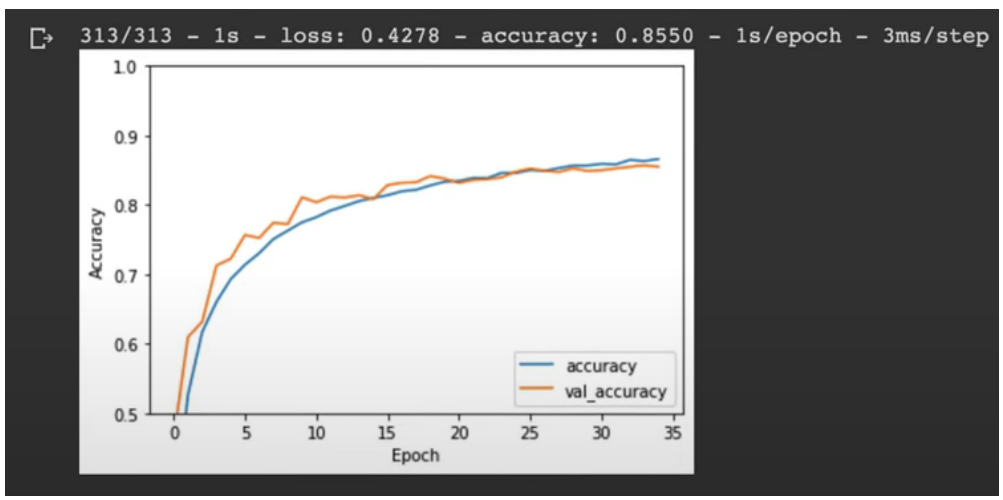
https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/tutorials/images/cnn.ipynb#scrollTo=WRzW5xSDDbNF

https://machinelearningmastery.com/how-to-develop-a-cnn-from-scratch-for-cifar-10-photo-classification/

**Instructions:**

Download code: Low_cnn or open from github on google collab

1) Run the code and see the resulting accuracy
    - Altered the layers with the baseline method to obtain a better accuracy
    - Altered the number of epochs to get a better accuracy
2) Results with the accuracy of 85.5%




**<u>CNN-Challenge Test</u>**

This code is used to identify the image and classify the category the picture belongs to. There is definitely more room for improvement. I have run the code multiple times and this is the best I could get.

**Import Packages:**

## Import TensorFlow

```
[ ] import tensorflow as tf
    import sys
    import numpy as np
    # baseline model with dropout
    #data augmentation on the cifar10 dataset
    from keras.datasets import cifar10
    from tensorflow.keras.utils import to_categorical
    from keras.models import Sequential
    from keras.layers import Conv2D
    from keras.layers import MaxPooling2D
    from keras.layers import Dense
    from keras.layers import Flatten
    from tensorflow.keras.optimizers import SGD
    from keras.preprocessing.image import ImageDataGenerator
    from keras.layers import Dropout
    from keras.layers import BatchNormalization
    from tensorflow.keras import datasets, layers, models
    import matplotlib.pyplot as plt
```

**Instructions:**

Download code: Low_cnnimage or open from github on google collab

1) Run the code
2) At the last few cells look at the code and see what it is results, there is still room for improvement for the code

Low_cnn.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
[27]
    0.8539999723434448

[28] automobile_url = "https://hips.hearstapps.com/hmg-prod.s3.amazonaws.com/images/devel-motors-sixteen-1540564064.jpg"
     automobile_path = tf.keras.utils.get_file('devel-motors-sixteen-1540564064.jpg', origin=automobile_url)

     img = tf.keras.utils.load_img(
         automobile_path, target_size=(32, 32)
     )
     img_array = tf.keras.utils.img_to_array(img)
     img_array = tf.expand_dims(img_array, 0) # Create a batch

     predictions = model.predict(img_array)
     score = tf.nn.softmax(predictions[0])

     print(
         "This image most likely belongs to {} with a {:.2f} percent confidence."
         .format(class_names[np.argmax(score)], 100 * np.max(score))
     )
     print(score)

     This image most likely belongs to automobile with a 99.84 percent confidence.
     tf.Tensor(
     [1.51132845e-05 9.98428643e-01 1.32753416e-06 9.97377811e-07
      5.18266710e-08 7.22688310e-06 4.77880121e-06 4.82860003e-07
      1.42707097e-04 1.39863719e-03], shape=(10,), dtype=float32)
```

Low_cnn.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```python
car_url = "https://images.all-free-download.com/images/graphiclarge/classic_jaguar_210354.jpg"
car_path = tf.keras.utils.get_file('classic_jaguar_210354.jpg', origin=car_url)

img = tf.keras.utils.load_img(
    car_path, target_size=(32, 32)
)
img_array = tf.keras.utils.img_to_array(img)
img_array = tf.expand_dims(img_array, 0) # Create a batch

predictions = model.predict(img_array)
score = tf.nn.softmax(predictions[0])

print(
    "This image most likely belongs to {} with a {:.2f} percent confidence."
    .format(class_names[np.argmax(score)], 100 * np.max(score))
)
print(score)
```

```
Downloading data from https://images.all-free-download.com/images/graphiclarge/classic_jaguar_210354.jpg
98304/93791 [==============================] - 0s 4us/step
106496/93791 [==================================] - 0s 4us/step
This image most likely belongs to automobile with a 99.84 percent confidence.
tf.Tensor(
[1.4931861e-05 9.9844068e-01 1.3129223e-06 9.8676685e-07 5.1398274e-08
 7.1654108e-06 4.7521989e-06 4.7763143e-07 1.4223462e-04 1.3874526e-03], shape=(10,), dtype=float32)
```

Low_cnn.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```python
[30] cars_url = "https://amsc-prod-cd.azureedge.net/-/media/aston-martin/images/default-source/models/valkyrie/new/valkyrie-spid
     cars_path = tf.keras.utils.get_file('valkyrie-spider_f02-169v2.jpg', origin=cars_url)

     img = tf.keras.utils.load_img(
         cars_path, target_size=(32, 32)
     )
     img_array = tf.keras.utils.img_to_array(img)
     img_array = tf.expand_dims(img_array, 0) # Create a batch

     predictions = model.predict(img_array)
     score = tf.nn.softmax(predictions[0])

     print(
         "This image most likely belongs to {} with a {:.2f} percent confidence."
         .format(class_names[np.argmax(score)], 100 * np.max(score))
     )
     print(score)
```

```
Downloading data from https://amsc-prod-cd.azureedge.net/-/media/aston-martin/images/default-source/models/valkyrie/new/val
729088/727309 [==============================] - 0s 0us/step
737280/727309 [==============================] - 0s 0us/step
This image most likely belongs to automobile with a 99.83 percent confidence.
tf.Tensor(
[1.8177538e-05 9.9830317e-01 1.5855538e-06 1.2335146e-06 5.9273333e-08
 8.7918352e-06 5.1984143e-06 5.6935158e-07 1.4943945e-04 1.5116582e-03], shape=(10,), dtype=float32)
```

Low_cnn.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```python
birdtwo_url = "https://images.all-free-download.com/images/graphiclarge/flying_bird_201952.jpg"
birdtwo_path = tf.keras.utils.get_file('flying_bird_201952.jpg', origin=birdtwo_url)

img = tf.keras.utils.load_img(
    birdtwo_path, target_size=(32, 32)
)
img_array = tf.keras.utils.img_to_array(img)
img_array = tf.expand_dims(img_array, 0) # Create a batch

predictions = model.predict(img_array)
score = tf.nn.softmax(predictions[0])

print(
    "This image most likely belongs to {} with a {:.2f} percent confidence."
    .format(class_names[np.argmax(score)], 100 * np.max(score))
)
print(score)
```

```
This image most likely belongs to ship with a 99.83 percent confidence.
tf.Tensor(
[7.8882376e-04 5.1609770e-04 2.8368187e-05 6.3221087e-05 2.6592275e-05
 7.6668475e-06 5.1191804e-05 8.4644571e-06 9.9832994e-01 1.7964470e-04], shape=(10,), dtype=float32)
```

## Balloon Flight

The goal of the game is avoid the following objects by clicking the hot-air balloon to move up.

**Import Packages:**

```
import pgzrun
from pgzero.builtins import Actor
from random import randint
```

**Instructions:**

Open code: Low_balloonflight.py in the folder Game_dev

1) Run the code
2) Click on the balloon and avoid the obstacles
3) Try to score a high score