AI- DEEP LEARNING PROJECT
FASHION MNIST CLASSIFICATION

Name Ponnuri Aniruddha

ABSTRACT:

In this project, we used the Convolutional Neural Network (CNN) model to create a fashion apparel detection system. We used the Fashion MNIST dataset to train the CNN model. Following a good training. The class name for a given piece of clothing can be predicted by the CNN model. The articles will be categorised into ten different types of clothing in this multiclass classification issue. The fashion training set comprises of 70,000 photos, divided among 10,000 testing samples and 60,000 training samples. Grayscale photos measuring 28x28 that have a label from one of ten classifications make up the dataset sample. Using a convolution neural network to train and test the model is the final objective.

OBJECTIVE:

Real-time searching and recognition are exceedingly challenging tasks. There hasn't been an efficient solution discovered for this issue yet. Despite extensive study in this field, the approaches that have so far been created are ineffective, call for lengthy training sessions, cannot be used in the real world, and cannot scale to a large number of classes. If a machine is trying to identify a specific thing, object detection is comparatively easier. Although they might be of the same type, recognising all the things requires the ability to tell one from the other. For machines, this challenge is especially challenging if they are unaware of the numerous possibilities of the objects.

INTRODUCTION:

Finding and identifying instances of real-world objects—like cars, bikes, TVs, flowers, and people—in photos or videos is a process known as object detection. An object detection technique enables the identification, localisation, and detection of numerous things inside an image, allowing you to comprehend the specifics of an image or video.

It is usually utilized in applications like image retrieval, security, surveillance, and advanced driver assistance systems (ADAS).Object Detection is done through many ways:

- Feature Based Object Detection
- Viola Jones Object Detection
- SVM Classifications with HOG Features
- Deep Learning Object Detection

METHODOLOGY:

A picture can be seen of as a two-dimensional capacity with the formula f(x, y), where x and y are spatial directions. The power or dark level of the picture is then the adequacy off at any combination of x and y. We refer to a picture as computerised if x, y, and the estimation of the abundance of are all constrained discrete amounts. The term "DIP" refers to the field of creating enhanced images using computerised PC techniques. A sophisticated image is composed of a small number of parts, each of which is valued and positioned in a certain location. Pixels are the name of the parts.

CODE:

Step 1) Import Libraries

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import tensorflow as tf
import keras
```

Step 2) Load data

```
(X_train, y_train), (X_test, y_test)=tf.keras.datasets.fashion_mnist.load_data()

X_train.shape,y_train.shape, "***************" , X_test.shape,y_test.shape

X_train[0]

y_train[0]

class_labels = ["T-shirt/top", "Trouser", "Pullover", "Dress", "Coat", "Sandal",
"Shirt",'Sneaker','Bag',"Ankle boot"]

class_labels

plt.imshow(X_train[0],cmap='Greys')

plt.figure(figsize=(16,16))

j=1
for  i in np.random.randint(0,1000,25):
  plt.subplot(5,5,j);j+=1
  plt.imshow(X_train[i],cmap='Greys')
  plt.axis('off')
  plt.title('{} / {}'.format(class_labels[y_train[i]],y_train[i]))
```

```
X_train.ndim

X_train = np.expand_dims(X_train,-1)

X_train.ndim

X_test=np.expand_dims(X_test,-1)

X_train = X_train/255
X_test= X_test/255

from sklearn.model_selection import  train_test_split
X_train,X_Validation,y_train,y_Validation=train_test_split(X_train,y_train,test_size=
0.4,random_state=210)

X_train.shape,X_Validation.shape,y_train.shape,y_Validation.shape
```

Step 3) Buiding the CNN model

```
model=keras.models.Sequential([

keras.layers.Conv2D(filters=32,kernel_size=3,strides=(1,1),padding='valid',activatio
n='relu',input_shape=[28,28,1]),
                    keras.layers.MaxPooling2D(pool_size=(2,2)),
                    keras.layers.Flatten(),
                    keras.layers.Dense(units=128,activation='relu'),
                    keras.layers.Dense(units=10,activation='softmax')
])

model.summary()

model.compile(optimizer='adam',loss='sparse_categorical_crossentropy',metrics=['a
ccuracy'])

model.fit(X_train,y_train,epochs=50,batch_size=1024,verbose=1,validation_data=(X
_Validation,y_Validation))

y_pred = model.predict(X_test)
y_pred.round(2)

y_test

model.evaluate(X_test, y_test)

plt.figure(figsize=(16,16))

j=1
for i in np.random.randint(0, 2000,15):
  plt.subplot(5,5, j); j+=1
  plt.imshow(X_test[i].reshape(28,28), cmap = 'Greys')
  plt.title('Actual = {} / {} \nPredicted = {} / {}'.format(class_labels[y_test[i]],
y_test[i], class_labels[np.argmax(y_pred[i])],np.argmax(y_pred[i])))
  plt.axis('off')

plt.figure(figsize=(16,30))

j=1
for i in np.random.randint(0, 1500,90):
  plt.subplot(10,6, j)
  j+=1
  plt.imshow(X_test[i].reshape(28,28), cmap = 'Greys')
  plt.title('Actual = {} / {} \nPredicted = {} / {}'.format(class_labels[y_test[i]],
y_test[i], class_labels[np.argmax(y_pred[i])],np.argmax(y_pred[i])))
  plt.axis('off')
```

Confussion matrix

```
from sklearn.metrics import confusion_matrix
plt.figure(figsize=(16,9))
y_pred_labels = [ np.argmax(label) for label in y_pred ]
cm = confusion_matrix(y_test, y_pred_labels)


sns.heatmap(cm, annot=True, fmt='d',xticklabels=class_labels,
yticklabels=class_labels)

from sklearn.metrics import classification_report
cr= classification_report(y_test, y_pred_labels, target_names=class_labels)
print(cr)
```

Saving Model

```
model.save('fashion_mnist_cnn_model.h5')
```

Building Complex CNN 2

```
#Building CNN model
cnn_model2 = keras.models.Sequential([
                    keras.layers.Conv2D(filters=32, kernel_size=3, strides=(1,1),
padding='valid',activation= 'relu', input_shape=[28,28,1]),
                    keras.layers.MaxPooling2D(pool_size=(2,2)),
                    keras.layers.Conv2D(filters=64, kernel_size=3, strides=(2,2),
padding='same', activation='relu'),
                    keras.layers.MaxPooling2D(pool_size=(2,2)),
                    keras.layers.Flatten(),
                    keras.layers.Dense(units=128, activation='relu'),
                    keras.layers.Dropout(0.25),
                    keras.layers.Dense(units=256, activation='relu'),
                    keras.layers.Dropout(0.25),
                    keras.layers.Dense(units=128, activation='relu'),
                    keras.layers.Dense(units=10, activation='softmax')
                    ])

# complie the model
cnn_model2.compile(optimizer='adam', loss= 'sparse_categorical_crossentropy',
metrics=['accuracy'])

#Train the Model
cnn_model2.fit(X_train, y_train, epochs=20, batch_size=512, verbose=1,
validation_data=(X_Validation, y_Validation))

cnn_model2.save('fashion_mnist_cnn_model2.h5')

"""######## very complex model"""

#Building CNN model
cnn_model3 = keras.models.Sequential([
                    keras.layers.Conv2D(filters=64, kernel_size=3, strides=(1,1),
padding='valid',activation= 'relu', input_shape=[28,28,1]),
                    keras.layers.MaxPooling2D(pool_size=(2,2)),
                    keras.layers.Conv2D(filters=128, kernel_size=3, strides=(2,2),
padding='same', activation='relu'),
                    keras.layers.MaxPooling2D(pool_size=(2,2)),
                    keras.layers.Conv2D(filters=64, kernel_size=3, strides=(2,2),
padding='same', activation='relu'),
                    keras.layers.MaxPooling2D(pool_size=(2,2)),
                    keras.layers.Flatten(),
                    keras.layers.Dense(units=128, activation='relu'),
                    keras.layers.Dropout(0.25),
                    keras.layers.Dense(units=256, activation='relu'),
                    keras.layers.Dropout(0.5),
                    keras.layers.Dense(units=256, activation='relu'),
                    keras.layers.Dropout(0.25),
                    keras.layers.Dense(units=128, activation='relu'),
                    keras.layers.Dropout(0.10),
                    keras.layers.Dense(units=10, activation='softmax')
                    ])

# complie the model
cnn_model3.compile(optimizer='adam', loss= 'sparse_categorical_crossentropy',
metrics=['accuracy'])

#Train the Model
```

```
cnn_model3.fit(X_train, y_train, epochs=50, batch_size=1024, verbose=1,
validation_data=(X_Validation, y_Validation))

cnn_model3.save('fashion_mnist_cnn_model3.h5')

cnn_model3.evaluate(X_test, y_test)
```

CONCLUSION:

In recent years, deep learning-based object recognition has attracted a lot of academic attention. Generic object detection pipelines serve as the foundation for this project and other similar jobs by acting as the building blocks for such tasks. These three more typical tasks—detecting objects, faces, and pedestrians—can all be completed with the aid of technology. Combining deep learning for object detection with OpenCV and efficient, threaded video streams with OpenCV, the authors were able to achieve their goal. Noise from the camera sensor and lighting conditions can alter the outcome since they make it difficult to identify objects. The outcome is an object detector powered by deep learning that can handle data at a rate of 6–8 frames per second.