In [1]:
```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
```

Load Dataset

In [2]:
```
train_df = pd.read_csv('fashion-mnist_train.csv')
test_df = pd.read_csv('fashion-mnist_test.csv')
```

In [3]:
```
train_df.shape
```

Out[3]:
```
(60000, 785)
```

In [4]:
```
test_df.shape
```

Out[4]:
```
(10000, 785)
```

In [5]:
```
train_df.describe()
```

Out[5]:

	label	pixel1	pixel2	pixel3	pixel4	pixel5
count	60000.000000	60000.000000	60000.000000	60000.000000	60000.000000	60000.000000
mean	4.500000	0.000900	0.006150	0.035333	0.101933	0.247967
std	2.872305	0.094689	0.271011	1.222324	2.452871	4.306912
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	2.000000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	4.500000	0.000000	0.000000	0.000000	0.000000	0.000000
75%	7.000000	0.000000	0.000000	0.000000	0.000000	0.000000
max	9.000000	16.000000	36.000000	226.000000	164.000000	227.000000

8 rows × 785 columns

In [6]:
```
train_df.label.unique()
```

Out[6]:
```
array([2, 9, 6, 0, 3, 4, 5, 8, 7, 1])
```

Each row represents an grayscale image containing 784 pixels and each pixel having values in range from 0-255

The column label is a discrete value in the range 0 to 9 each value representing a specific category

In [7]:
```
class_names = ['T-shirt/top', 'Trouser', 'Pullover', 'Dress', 'Coat','Sanda
```

Preprocess Data

Convert each image of 784 into (28x28x1)(height x width x color_channels). Divide values by 255 to scale the values.

In [8]:
```
x_train = train_df.iloc[:,1:].to_numpy()
x_train = x_train.reshape([-1,28,28,1])
x_train = x_train / 255
```

In [9]:
```
y_train = train_df.iloc[:,0].to_numpy()
```

In [10]:
```
x_test = test_df.iloc[:,1:].to_numpy()
x_test = x_test.reshape([-1,28,28,1])
x_test = x_test / 255
```

In [11]:
```
y_test = test_df.iloc[:,0].to_numpy()
```

Visualization

In [12]:
```
plt.figure(figsize=(10,10))
for i in range(25):
    plt.subplot(5,5,i+1)
    plt.xticks([])
    plt.yticks([])
    plt.grid(False)
    plt.imshow(x_train[i], cmap=plt.cm.binary)
    plt.xlabel(class_names[y_train[i]])
plt.show()
```

Model Building

In [13]:
```
from keras.models import Sequential
from keras.layers import Dense,Conv2D,Flatten,MaxPooling2D,Dropout
```

```
2024-04-28 09:52:04.528874: I tensorflow/core/platform/cpu_feature_guard.
cc:210] This TensorFlow binary is optimized to use available CPU instruct
ions in performance-critical operations.
To enable the following instructions: AVX2 FMA, in other operations, rebu
ild TensorFlow with the appropriate compiler flags.
2024-04-28 09:52:05.275892: W tensorflow/compiler/tf2tensorrt/utils/py_ut
ils.cc:38] TF-TRT Warning: Could not find TensorRT
```

In [14]:
```
model = Sequential()

model.add(Conv2D(filters=64,kernel_size=(3,3),input_shape=(28,28,1),activat
model.add(MaxPooling2D(pool_size = (2,2)))
model.add(Dropout(rate=0.3))
model.add(Flatten())
model.add(Dense(units=32, activation='relu'))
model.add(Dense(units=10, activation='sigmoid'))
model.compile(loss='sparse_categorical_crossentropy',optimizer='adam',metri
model.summary()
```

```
/home/sanket/.local/lib/python3.10/site-packages/keras/src/layers/convolu
tional/base_conv.py:107: UserWarning: Do not pass an `input_shape`/`input
_dim` argument to a layer. When using Sequential models, prefer using an
`Input(shape)` object as the first layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
2024-04-28 09:52:05.908103: E external/local_xla/xla/stream_executor/cud
a/cuda_driver.cc:282] failed call to cuInit: CUDA_ERROR_UNKNOWN: unknown
error
2024-04-28 09:52:05.908132: I external/local_xla/xla/stream_executor/cud
a/cuda_diagnostics.cc:134] retrieving CUDA diagnostic information for hos
t: sanket
2024-04-28 09:52:05.908141: I external/local_xla/xla/stream_executor/cud
a/cuda_diagnostics.cc:141] hostname: sanket
2024-04-28 09:52:05.908238: I external/local_xla/xla/stream_executor/cud
a/cuda_diagnostics.cc:165] libcuda reported version is: 550.54.15
2024-04-28 09:52:05.908262: I external/local_xla/xla/stream_executor/cud
a/cuda_diagnostics.cc:169] kernel reported version is: NOT_FOUND: could n
ot find kernel module information in driver version file contents: "NVRM
version: NVIDIA UNIX Open Kernel Module for x86_64  550.54.15  Release Bu
ild  (dvs-builder@U16-A24-23-2)  Tue Mar  5 22:15:33 UTC 2024
GCC version:  gcc version 12.3.0 (Ubuntu 12.3.0-1ubuntu1~22.04)
"
```

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 26, 26, 64)	640
max_pooling2d (MaxPooling2D)	(None, 13, 13, 64)	0
dropout (Dropout)	(None, 13, 13, 64)	0
flatten (Flatten)	(None, 10816)	0
dense (Dense)	(None, 32)	346,144
dense_1 (Dense)	(None, 10)	330

Total params: 347,114 (1.32 MB)

Trainable params: 347,114 (1.32 MB)

Non-trainable params: 0 (0.00 B)

In [ ]:
```
model.fit(x_train,y_train,epochs=5,batch_size=1200,validation_split=0.05)
```

```
Epoch 1/5
48/48 ━━━━━━━━━━━━━━━━━━━━ 25s 513ms/step - accuracy: 0.8325 - loss: 0.47
35 - val_accuracy: 0.8463 - val_loss: 0.4236
Epoch 2/5
48/48 ━━━━━━━━━━━━━━━━━━━━ 43s 546ms/step - accuracy: 0.8569 - loss: 0.40
85 - val_accuracy: 0.8667 - val_loss: 0.3876
Epoch 3/5
```

Evaluation

In [ ]:
```
evaluation = model.evaluate(x_test,y_test)
```

In [ ]:
```
print(f"Accuracy: {evaluation[1]}")
```

In [ ]:
```
y_probas = model.predict(x_test)
```

In [ ]:
```
y_pred = y_probas.argmax(axis=-1)
```

In [ ]:
```
y_pred
```

In [ ]:
```
plt.figure(figsize=(10,10),)
for i in range(25):
    plt.subplot(5,5,i+1)
    plt.xticks([])
    plt.yticks([])
    plt.grid(False)
    plt.imshow(x_test[i], cmap=plt.cm.binary)
    #plt.xlabel(f"True Class:{y_test[i]}")
    plt.title(f"Pred:{class_names[y_pred[i]]}")
plt.show()
```

In [ ]:
```
from sklearn.metrics import classification_report
```

In [ ]:
```
num_classes = 10
class_names = ["class {}".format(i) for i in range(num_classes)]
cr = classification_report(y_test, y_pred, target_names=class_names)
print(cr)
```

In [ ]:
```
```