

A background collage of various fashion items including handbags, shoes, dresses, and tops, arranged in a grid-like pattern.

ABSTRACT

For the Project Fashion MNIST Data Classifier

In this project, we have built a fashion apparel recognition using the Convolutional Neural Network (CNN) model. To train the CNN model, we have used the Fashion MNIST dataset. After successful training, the CNN model can predict the name of the class given apparel item belongs to. This is a multiclass classification problem in which there are 10 apparel classes the items will be classified. The fashion training set consists of 70,000 images divided into 60, 000 training and 10,000 testing samples. Dataset sample consists of 28x28 grayscale images, associated with a label from 10 classes. So the end goal is to train and test the model using Convolution neural network.

OBJECTIVE

This work is part of my experiments with Fashion-MNIST dataset using various Machine Learning algorithms/models. The objective is to identify (predict) different fashion products from the given images using various best possible Machine Learning Models (Algorithms) and compare their results (performance measures/scores) to arrive at the best ML model. I have also experimented with 'dimensionality reduction' technique for this problem.



INTRODUCTION

The MNIST dataset comprising of 10-class handwritten digits, was first introduced by LeCun et al. [1998] in 1998. At that time one could not have foreseen the stellar rise of deep learning techniques and their performance. Despite the fact that today deep learning can do so much the simple MNIST dataset has become the most widely used testbed in deep learning, surpassing CIFAR10 [Krizhevsky and Hinton, 2009] and ImageNet [Deng et al., 2009] in its popularity via Google trends¹. Despite its simplicity its usage does not seem to be decreasing despite calls for it in the deep learning community. The reason MNIST is so popular has to do with its size, allowing deep learning researchers to quickly check and prototype their algorithms. This is also complemented by the fact that all machine learning libraries (e.g. scikit-learn) and deep learning frameworks (e.g. Tensorflow, Pytorch) provide helper functions and convenient examples that use MNIST out of the box. Our aim with this work is to create a good benchmark dataset which has all the accessibility of MNIST, namely its small size, straightforward encoding and permissive license. We took the approach of sticking to the 10 classes 70, 000 grayscale images in the size of 28×28 as in the original MNIST. In fact, the only change one needs to use this dataset is to change the URL from where the MNIST dataset is fetched. Moreover, Fashion-MNIST poses a more challenging classification task than the simple MNIST digits data, whereas the latter has been trained to accuracies above 99.7% as reported in Wan et al. [2013], Ciregan et al. [2012]. We also looked at the EMNIST dataset

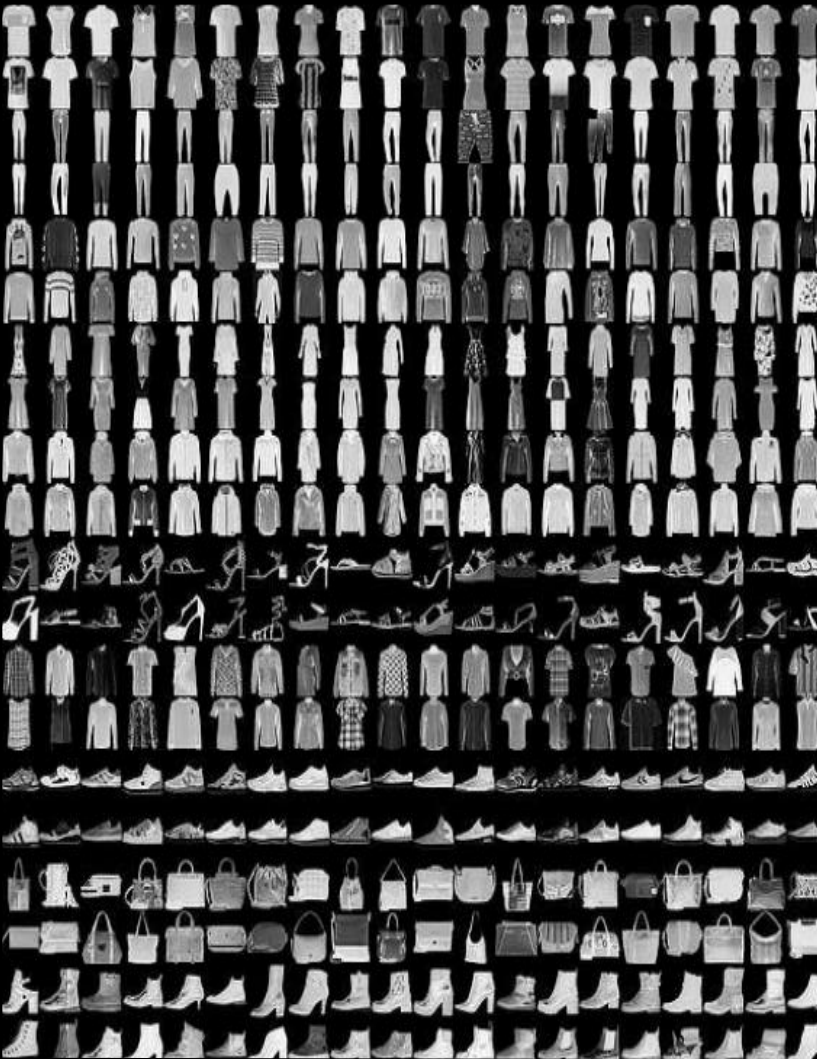
provided by Cohen et al. [2017], an extended version of MNIST that extends the number of classes by introducing uppercase and lowercase characters. However, to be able to use it seamlessly one needs to not only extend the deep learning framework's MNIST helpers, but also change the underlying deep neural network to classify these extra classes.

Fashion-MNIST Dataset

1. Converting the input to a PNG image.
2. Trimming any edges that are close to the color of the corner pixels. The “closeness” is defined by the distance within 5% of the maximum possible intensity in RGB space.
3. Resizing the longest edge of the image to 28 by subsampling the pixels, i.e. some rows and columns are skipped over.
4. Sharpening pixels using a Gaussian operator of the radius and standard deviation of 1.0, with increasing effect near outlines.
5. Extending the shortest edge to 28 and put the image to the center of the canvas.
6. Negating the intensities of the image.
7. Converting the image to 8-bit grayscale pixels.

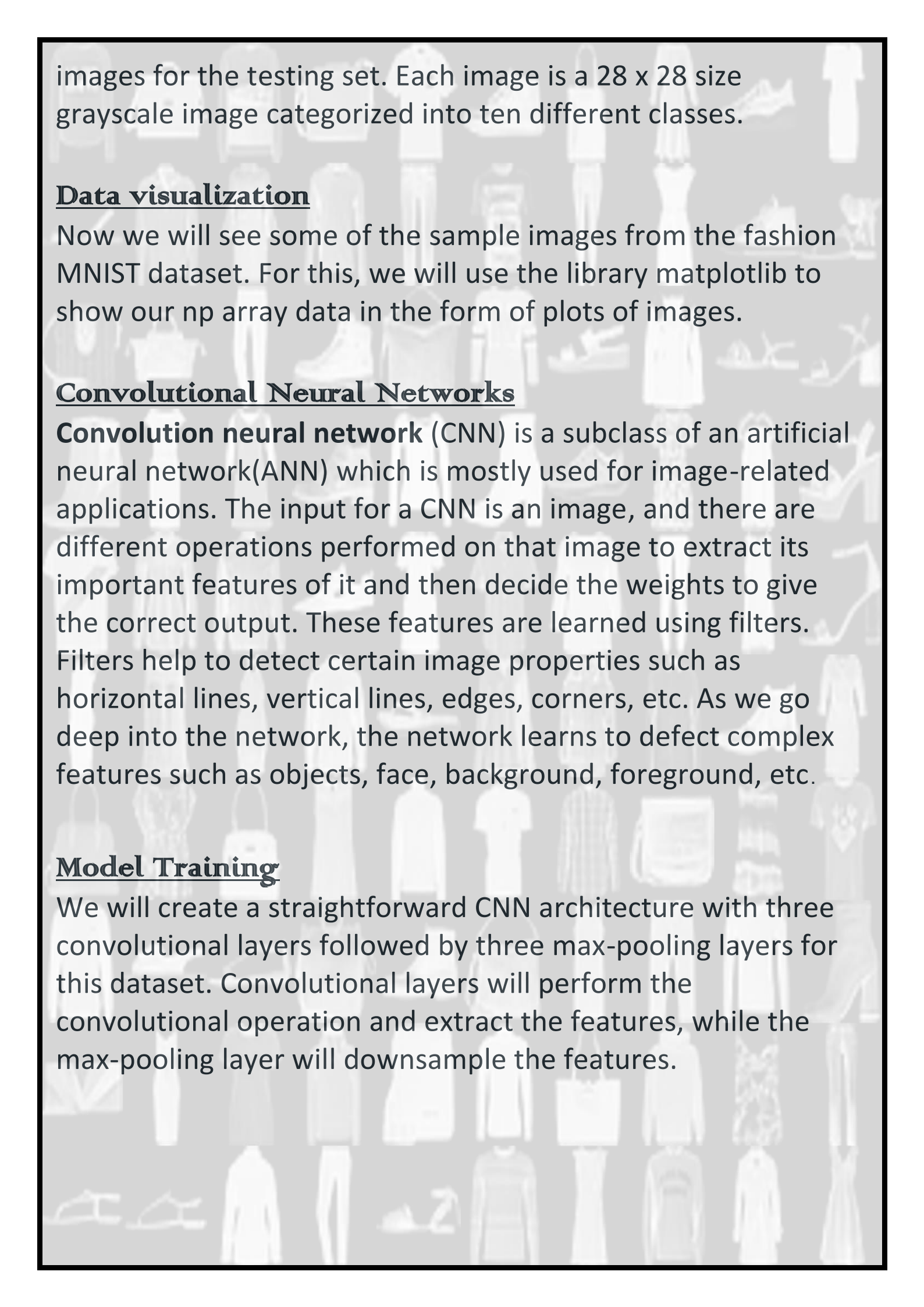
METHODOLOGY

Class names and example images in Fashion-MNIST dataset.

LABEL	DESCRIPTION	EXAMPLES
0	T-Shirt/Top	
1	Trouser	
2	Pullover	
3	Dress	
4	Coat	
5	Sandals	
6	Shirt	
7	Sneaker	
8	Bag	
9	Ankle boot	

Data Analysis

In the data analysis, we will see the number of images available, the dimensions of each image, etc. We will then split the data into training and testing. The fashion MNIST dataset consists of 60,000 images for the training set and 10,000



images for the testing set. Each image is a 28 x 28 size grayscale image categorized into ten different classes.

Data visualization

Now we will see some of the sample images from the fashion MNIST dataset. For this, we will use the library matplotlib to show our np array data in the form of plots of images.

Convolutional Neural Networks

Convolution neural network (CNN) is a subclass of an artificial neural network(ANN) which is mostly used for image-related applications. The input for a CNN is an image, and there are different operations performed on that image to extract its important features of it and then decide the weights to give the correct output. These features are learned using filters. Filters help to detect certain image properties such as horizontal lines, vertical lines, edges, corners, etc. As we go deep into the network, the network learns to detect complex features such as objects, face, background, foreground, etc.

Model Training

We will create a straightforward CNN architecture with three convolutional layers followed by three max-pooling layers for this dataset. Convolutional layers will perform the convolutional operation and extract the features, while the max-pooling layer will downsample the features.

CODE

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

(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[1]

Y_train[1]

class_labels = [ "T-shirt/top", "Trouser", "Pullover", "Dress", "Coat", "Sandal", "Shirt", "Sneaker", "Bag", "Ankle boot"]

class_labels

plt.figure()
plt.imshow(X_train[1])
plt.colorbar()

X_train[0]

X_train = X_train/255
X_test = X_test/255

X_train[0]

plt.imshow(X_train[1],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=0.2,random_state=2020)

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

model=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.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=['accuracy'])

model.fit(X_train,y_train,epochs=10,batch_size=512,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, 1000,25):
    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))
    plt.axis('off')

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

j=1
for i in np.random.randint(0, 1000,60):
    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))

```



```

plt.axis('off')

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)

model.save("Fashion_mnist_cnn_model.h5")

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')
])

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

cnn_model2.fit(X_train, y_train, epochs=50, batch_size=512, verbose=1, validation_data=(X_Validation, y_Validation))

cnn_model2.save('fashion_mnist_cnn_model3.h5')

cnn_model2.evaluate(X_test, Y_test)

```

CONCLUSION

This paper introduced Fashion-MNIST, a fashion product images dataset intended to be a dropin replacement of MNIST and whilst providing a more challenging alternative for benchmarking machine learning algorithm. The images in Fashion-MNIST are converted to a format that matches that of the MNIST dataset, making it immediately compatible with any machine learning package capable of working with the original MNIST dataset.

GOOGLE DRIVE LINK :-

[https://colab.research.google.com/drive/1nroZXWM0Vi2PxTayhIZol713TqlxTFMK?usp=share link](https://colab.research.google.com/drive/1nroZXWM0Vi2PxTayhIZol713TqlxTFMK?usp=share_link)



Thank You