FASHION-MNIST IMAGE CLASSIFICATION PROJECT

TEAM:- Teachbots

Vibhu Pandey
GLA University,Mathura
vpskmic@gmail.com

Shaswat Tripathi
GLA University,Mathura
shaswat.traipathi_da18@gla.ac.in

Divyansh Mishra
GLA University,Mathura
divyanshmishra29@gmail.com

Abstract

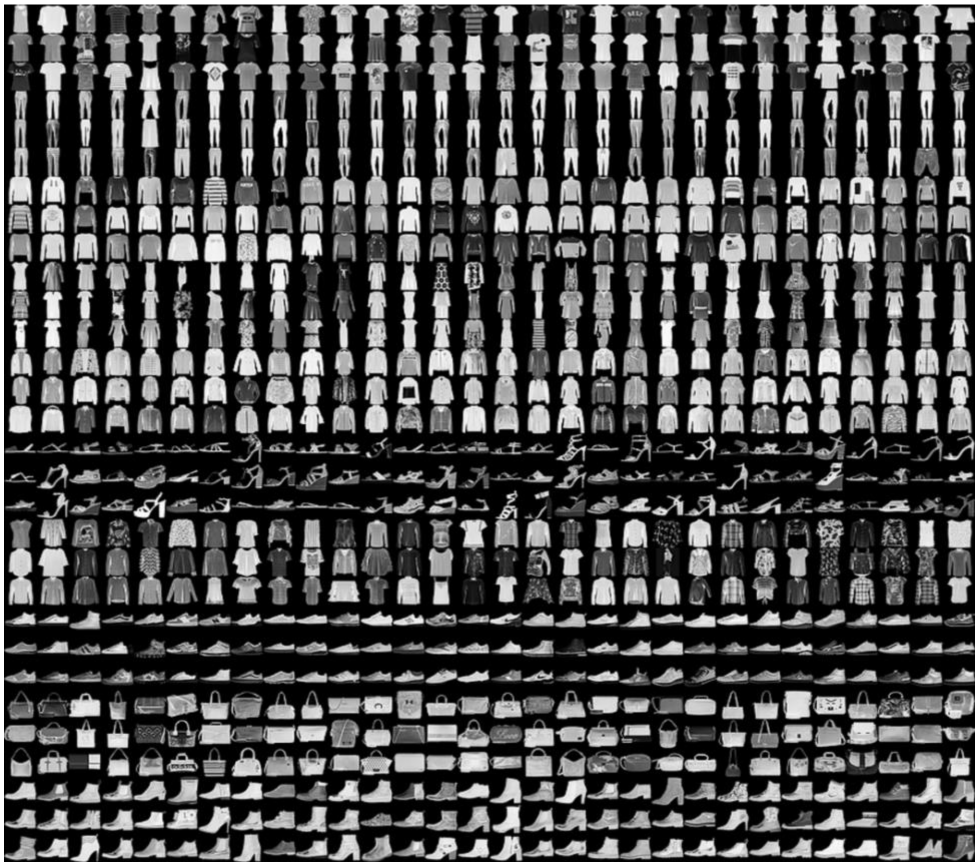
We tend to build a neural network (NN) and train it with the "Fashion MNIST Dataset". Fashion MNIST Dataset is a data with 70.000 images and contains 10 classes of clothing with a dimension of 28 by 28 grayscale image color.

Introduction

One of the most excellent things concerning deep learning is that it assists you to match the x and y relationship and that's what enables you to do magnificent things like have computers examine an image and do recognition or even examine an image and tells you that's a dress or a combination of pants.
The Fashion MNIST dataset is meant to be a (slightly more challenging) drop-in replacement for the (less challenging) MNIST dataset.

Why is this of interest for the scientific community?

The original MNIST Dataset contains a lot of handwritten digits. People from AI/ML/Data Science community love this dataset and use it as a benchmark to validate their algorithms. In fact, MNIST is often the first dataset they would try on. "If it doesn't work on MNIST, it won't work at all", they said. "Well, if it does work on MNIST, it may still fail on others."
Fashion-MNIST is intended to serve as a direct drop-in replacement for the original MNIST dataset to benchmark machine learning algorithms, as it shares the same image size and the structure of training and testing splits.



There are a few problems with the standard MNIST digit recognition dataset:

It’s far too easy for standard machine learning algorithms to obtain 97%+ accuracy.

It’s even easier for deep learning models to achieve 99%+ accuracy.

The dataset is overused.

MNIST cannot represent modern computer vision tasks.

Zalando, therefore, created the Fashion MNIST dataset as a drop-in replacement for MNIST.
The Fashion MNIST dataset is identical to the MNIST dataset in terms of training set size, testing set size, number of class labels, and image dimensions:
60,000 training examples

10,000 testing examples

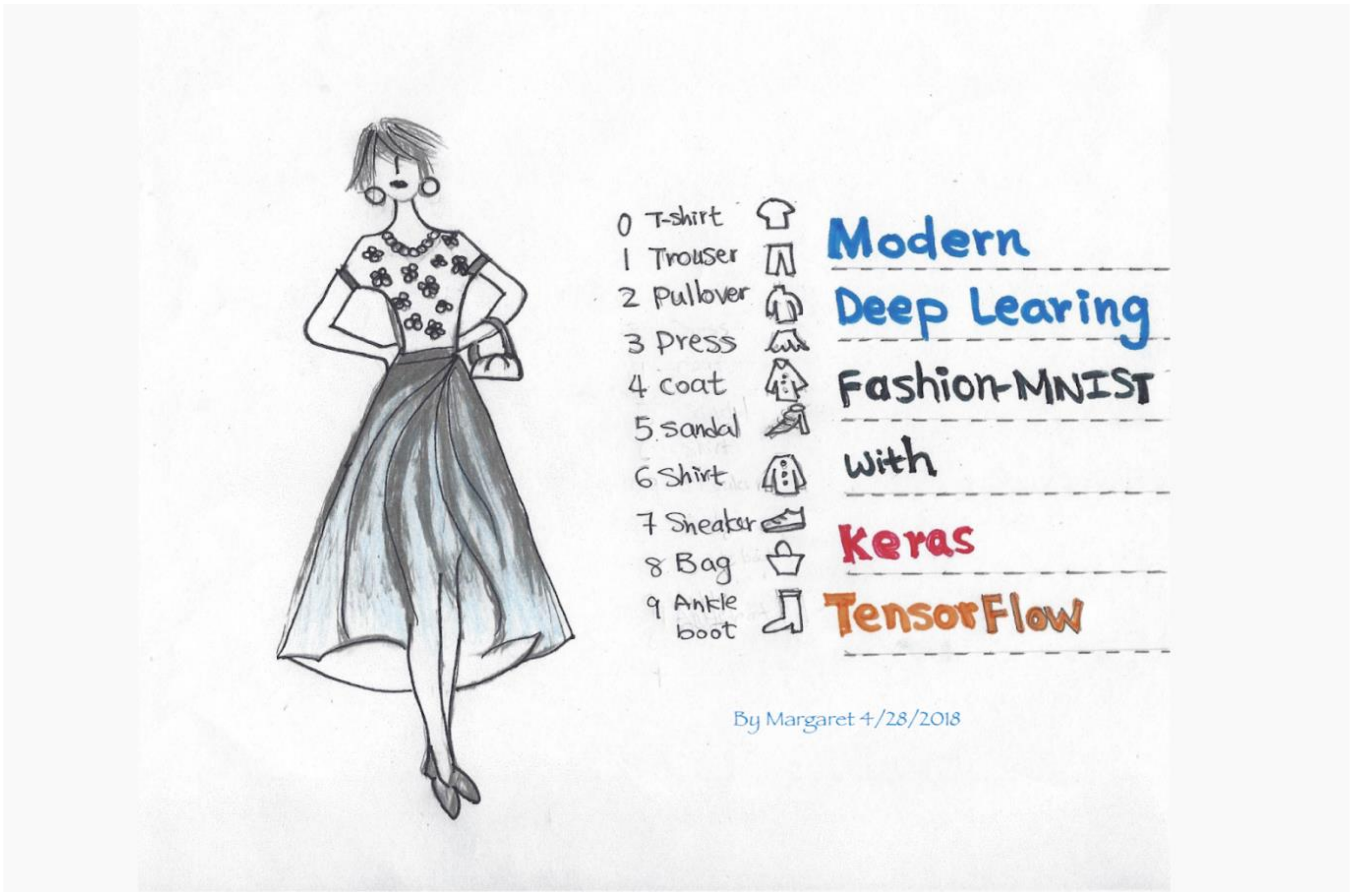
10 classes

28×28 grayscale images

DATASET DESCRIPTION

The ten fashion class labels include:
T-shirt/top
Trouser/pants
Pullover shirt
Dress
Coat
Sandal
Shirt
Sneaker
Bag
Ankle boot

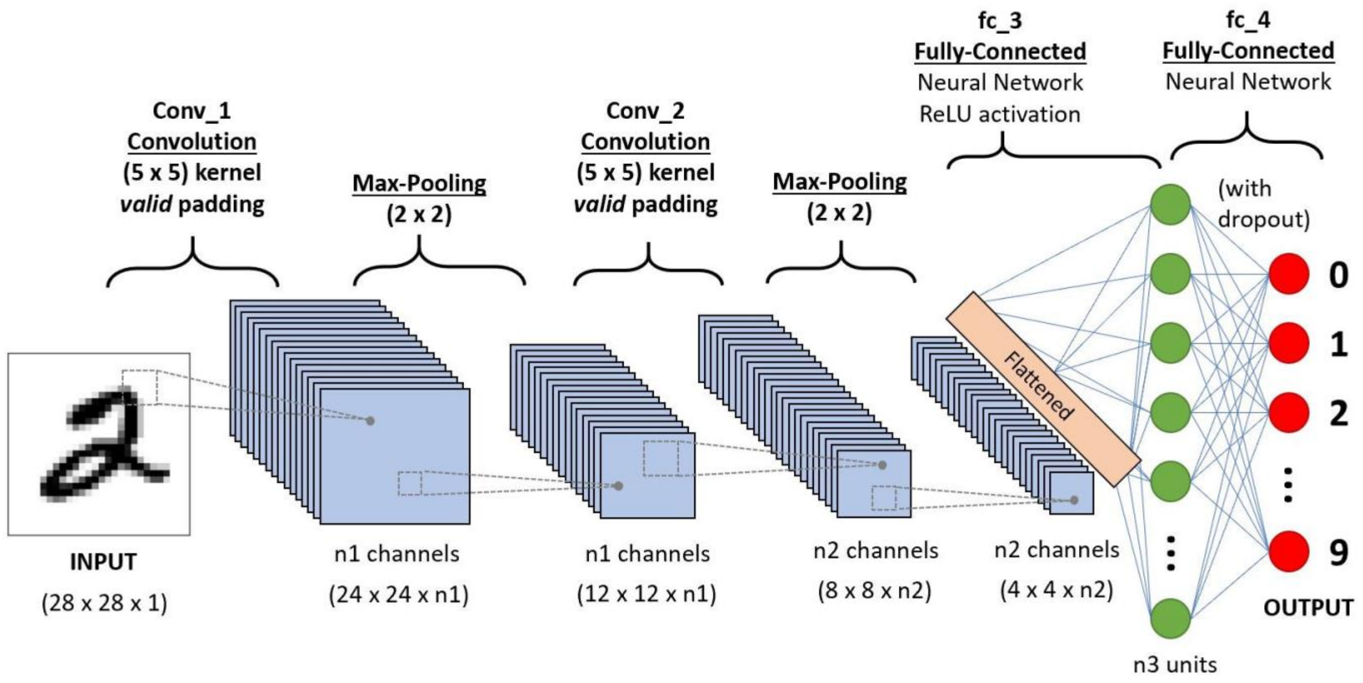
Convolutional Neural Network (CNN) with Keras on the Fashion MNIST dataset, giving you not only hands-on experience working with the Keras library but also your first taste of clothing/fashion classification.



Related Work

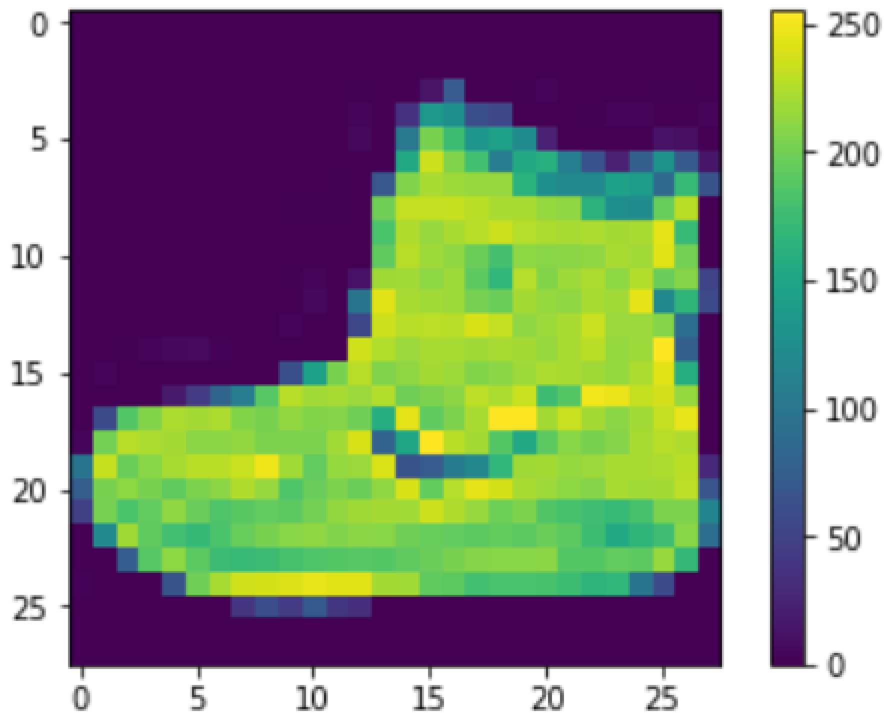
Convolutional Neural Network

CNN plays an important role in computer vision related pattern recognition tasks, because of its superior spatial feature extraction capability and fewer computation cost. CNN uses convolution kernels to convolve with the original images or feature maps to extract higher-level features. However, how to design better convolutional neural network architectures still remains as an opening question. Inception network proposed in allows the network to learn the best combination of kernels. In order to train much deeper neural networks, K. He et al. propose the Residual Network (ResNet) , which can learn an identity mapping from the previous layer. As object detectors are usually deployed on mobile or embedded devices, where the computational resources are very limited, Mobile Network (MobileNet) is proposed. It uses depth-wise convolution to extract features and channel wised convolutions to adjust channel numbers, so the computational cost of MobileNet is much lower than networks using standard convolutions.



Preprocess the data

The MNIST dataset or any kind of data needs to be preprocessed before you feed it to the neural network for training, and if you plot the first image in the training set by using matplotlib, you will see all the pixel values fall between 0 to 255:



Now, we normalized the data before feeding it to the model so that the neural network will require less computational power for training. To do that we divide all the training image and the test images by 255 so it will be scaled to a range of 0 to 1:

Build the model

The most important theory of creating block neural networks is the layers. Layers will extract a pattern from the data that have fed it into them.
Most of the neural network models contain multiple layers that are connected to each other and most of them such as tf.keras.layers.Dense has parameters that are learned during training.

The first layer in this model: tf.keras.layers.Flatten transforms the format of each picture to a one-dimensional array.
Upon the pixels of pictures that have been flattened by the Flatten layer, the rest of the neural network contains a sequence of two tf.keras.layers.Dense layers and that is what we call the Fully connected layers.

The first layer which is the Dense has 128 neurons and the second one has 10 neurons.
You can use this line of code to get a great understanding of the architecture of your own neural network:

Compile the model

Before we start running our neural networks, we need to make sure that we have set up a few more things and the first thing is:
The Loss function that Will detect how precise the neural network is during the compile process or in other words how the model is doing well.
The second one is Metrics which will observe the testing and training during the running of the model.
The third one is the Optimizer that will make the neural network update the parameters for our neural network during the training process.

Train the model

Train the artificial neural network demands the next steps:

Feed the training data to the model.

The artificial neural network learns to link images and labels.

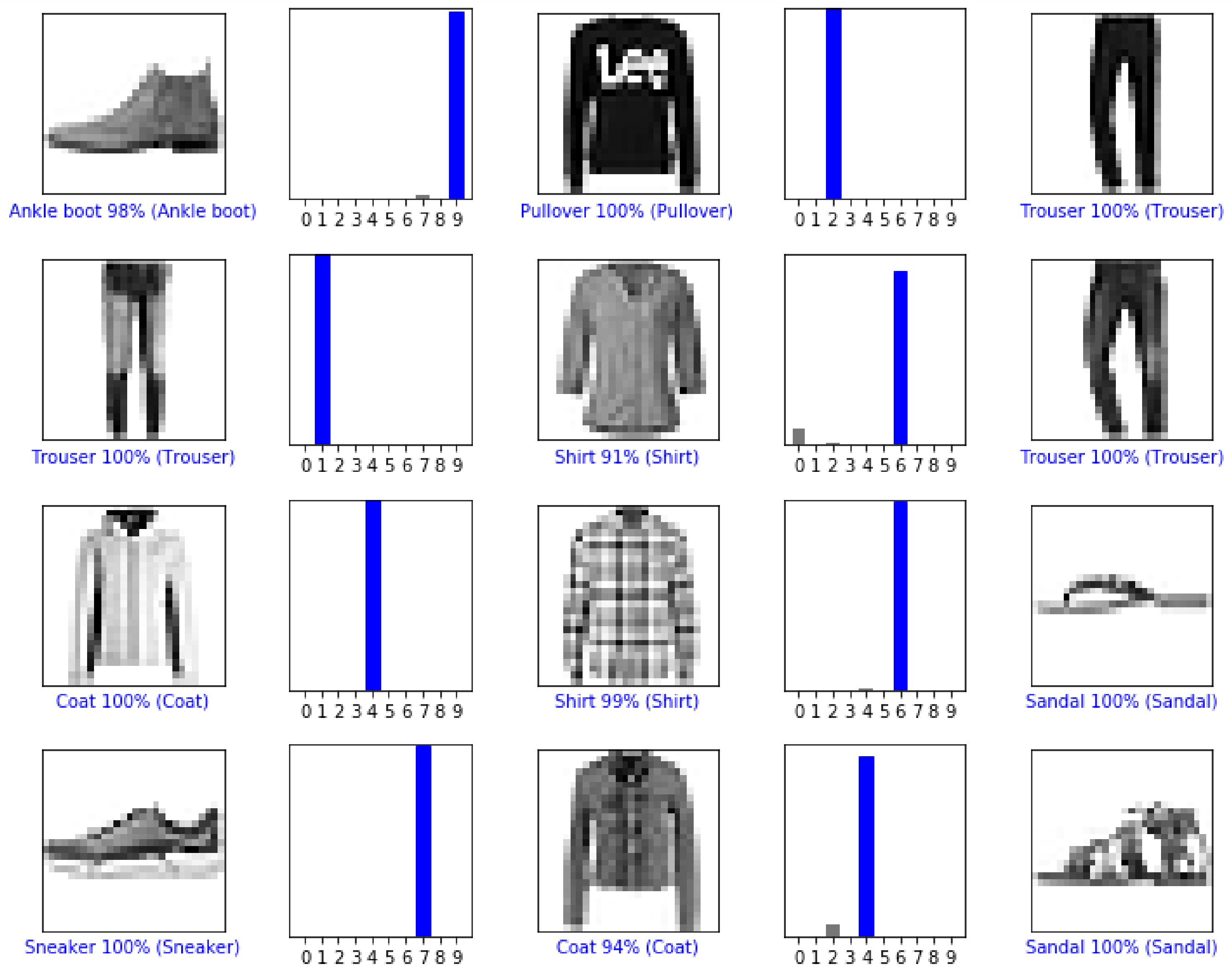
You ask the neural network to make predictions about a test set.

Evaluation Metrics

We employed precision and recall as metrics, they are defined as follows.
precision = T P /T P + F P
recall = T P /T P + F N
where T P, F P and F N denoted the true positive, false positive and false negative, respectively.

Visualize the predictions

Now we can use the trained model to make predictions / classifications on the test datasetmodel.predict(x_test) and visualize them.



CONCLUSION

With CNN model using Tensorflow Keras, we got 86% accuracy on test data in classification of our fashion dataset using different labels mentioned above.
Only few classes are not correctly classified all the time, especially Shirt and Coat.
And with these satisfactory result, we mark the end of project.

REFERENCES

A. Schindler, T. Lidy, S. Karner, and M. Hecker, “Fashion and apparel classification using convolutional neural networks,” arXiv preprint arXiv:1811.04374, 2018.

B.Research Project by Kashif Rasul & HanXiao (ex-member)

Wikipaedia

Discover The Neural Network Power For Classifying Images by Abdelhakim Oh