IMAGE CLASSIFICATION

FASHION-MNIST

OBJECTIVE

- To classify images of different pieces of clothing.

CLIENT & DATA-SET

- Fashion-MNIST is a dataset of Zalando's article images—consisting of a training set of ***60,000*** examples and a test set of ***10,000*** examples. Each example is a 28x28 grayscale image, associated with a label from ***10 classes***.

- Data-set is publicly available on Kaggle and Zalando Fashion MNIST repository on GitHub.

- Fashion-MNIST is intended as direct drop-in replacement for the original MNIST dataset. It shares the same image size and structure of training and testing splits.

BUSINESS IMPACT

- E-commerce companies have lots of items for sale online which requires lots of images to be displayed on their websites, applications and on social media. And it takes lot of human power and time to separate these images into respective groups. This classifier which we are going to build helps businesses to categorize images into respective groups.

METHODOLOGY

- For this project we will be going to use deep learning concepts like artificial neural networks and convolutional neural networks to build an image classification model which will learn to distinguish 10 different item images into their respective categories.

LABELS

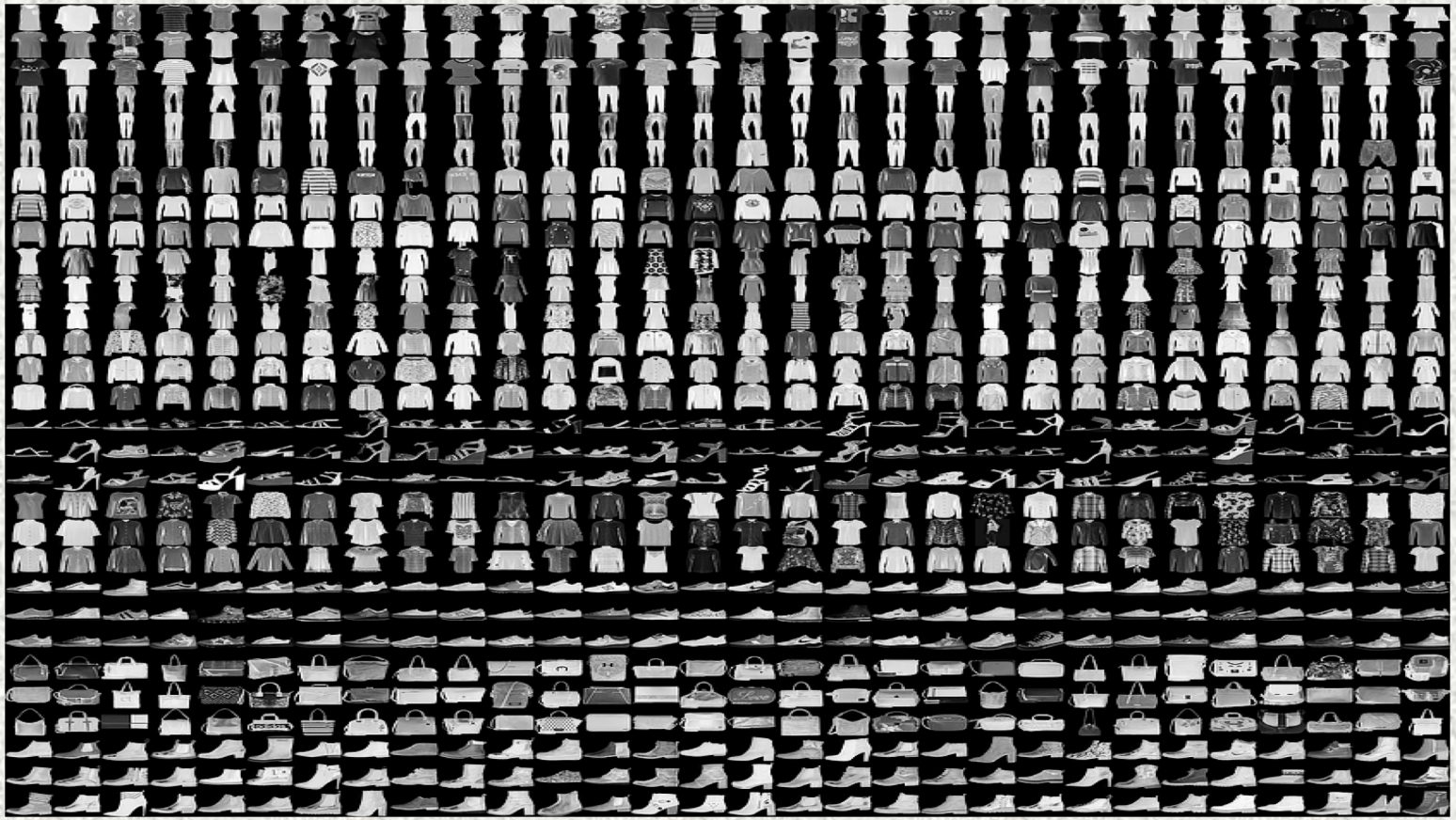
- 0 - T-shirt/top
- 1 - Trouser
- 2 - Pullover
- 3 - Dress
- 4 - Coat
- 5 - Sandal
- 6 - Shirt
- 7 - Sneaker
- 8 - Bag
- 9 - Ankle boot

DATA WRANGLING

- Each row is a separate image.
- Column 1 is the class label.
- Remaining columns are pixel numbers (784 total).
- Each value is the darkness of the pixel (1 to 255).
- We normalized the pixel data between 0 and 1.
- Then we split the original training data into 80% training and 20% validation.
- Later we reshaped the the image array (60000, 785) into original image size which is (60000, 28, 28, 1).
- Finally we apply one-hot encoding to our class variables to convert them into a format that works better with machine learning algorithms.

DATA EXPLORATION

- Here's an example how the data looks (*each class takes three-rows*)



MODELLING

- For building an image classifier here we use deep convolutional neural networks (CNN's).

Layers in a CNN:

- Convolution Layer: The primary purpose of Convolution in case of a CNN is to extract features from the input image.
- Pooling Layer: Pooling reduces the dimensionality of each feature map but retaining the most important information.
- Fully Connected Layer/Dense Layer: "Fully Connected" implies that every neuron in the previous layer is connected to every neuron on the next layer.

CONVOLUTIONAL NETWORK

Training process in a Convolution Network:

- We initialize all filters and parameters / weights with random values.
- The network takes a training image as input, goes through the forward propagation step (convolution, ReLU and pooling operations along with forward propagation in the Fully Connected layer) and finds the output probabilities for each class.
- Calculate the total error at the output layer (summation over all 10 classes).
- Use Backpropagation to calculate the gradients of the error with respect to all weights in the network and use gradient descent to update all filter values / weights and parameter values to minimize the output error.

EVALUATION METRIC

- Evaluation metrics explain the performance of a model.
- Categorical Cross-Entropy & Accuracy are our metrics in this project.
- **Categorical Cross-Entropy:** Cross-entropy loss, or log loss, measures the performance of a classification model whose output is a probability value between 0 and 1. Cross-entropy loss increases as the predicted probability diverges from the actual label.

GENERALIZATION & REGULARIZATION

- Generalization refers to how well the concepts learned by the model apply to new unseen data.
- Overfitting happens when the models learns too well the details and the noise from training data, but it doesn't generalize well, so the performance is poor for testing data.
- Regularization is a key component in preventing overfitting.

Regularization techniques:

1. Data Augmentation.
2. Dropout.
3. Batch Normalization.

HYPER PARAMETERS

- Hyper parameters are the variables which determines the network structure and the variables which determine how the network is trained.

Hyper parameters related to Network structure:

- Number of Hidden-Layers & Units
- Dropout

Hyper parameters related to Training Algorithm:

- Learning Rate
- Number of Epochs
- Batch size

MODELLING STRATEGY

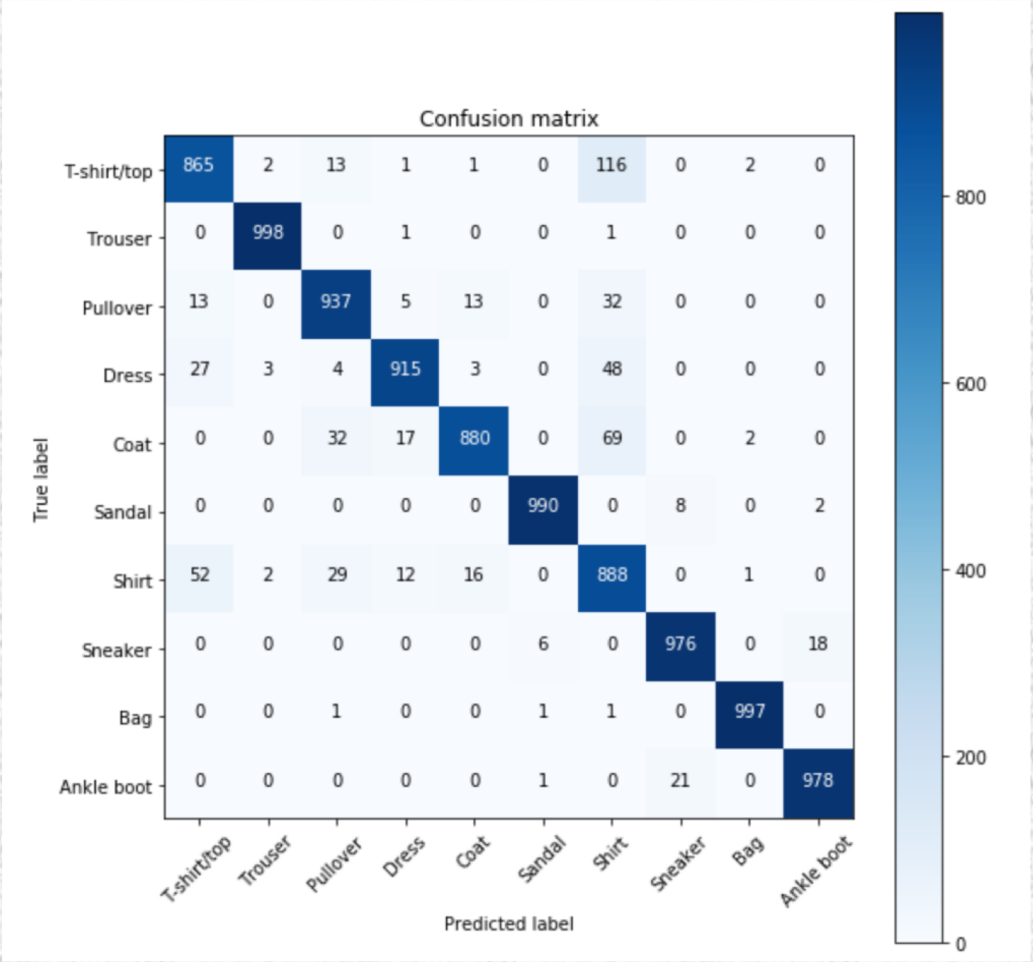
- Start with a basic 1-layer CNN .
- Add more number of Layers and Units getting the best validation performance possible.
- Also, parallelly preventing the overfitting issue by working on different regularization techniques in the process .
- Finally our main goal is to get the best test accuracy without overfitting on the training data by tuning & testing various hyper parameters.

CNN'S & ACCURACIES

CNN	Classifier Accuracy
1-Layer	91.25
2-Layer	91.73
3-Layer	92.55
4-Layer with Padding	93.52
5-Layer with Padding	93.69
5-Layer with Padding & Batch Normalization	93.89
Image Augmentation on 5-Layer	94.30

CONFUSION MATRIX

- We can observe the misclassifications in the confusion matrix plotted below.



CONCLUSION

- Finally, we observe in the confusion matrix that most of the misclassifications are happening between the classes Shirt, T-shirt/top, Pullover and Coat which are majorly impacting the performance of the classifier.
- As Deep Networks require large amount of training data to achieve good performance which we observed in our analysis how image augmentation boosted the classifier performance.
- To further improve the classifier performance, we should collect more samples and give more images from these 4 classes to the model so the classifier can learn more features or patterns even better.