**Optical Character Recognition (OCR) Dataset: Various Fonts and Styles**

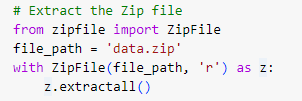
## **1. Data Introduction**

### **1.1 Overview**

Optical Character Recognition (OCR) is a technology used to convert different types of documents, such as scanned paper documents, PDFs, or images taken by a digital camera, into editable and searchable data. The dataset used in this project contains images of characters in various fonts and styles.

### **1.2 Exploratory Data Analysis (EDA)**

#### **1.2.1 Extracting the Data**

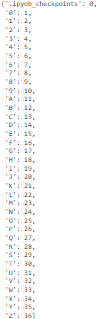


#### **1.2.2 Generating Train and Validation Data**

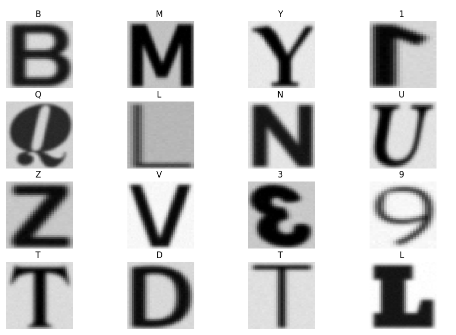
We use ImageDataGenerator from TensorFlow Keras to generate training and validation data with real-time data augmentation:



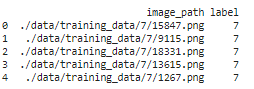
**1.2.3 Checking Class Distribution**



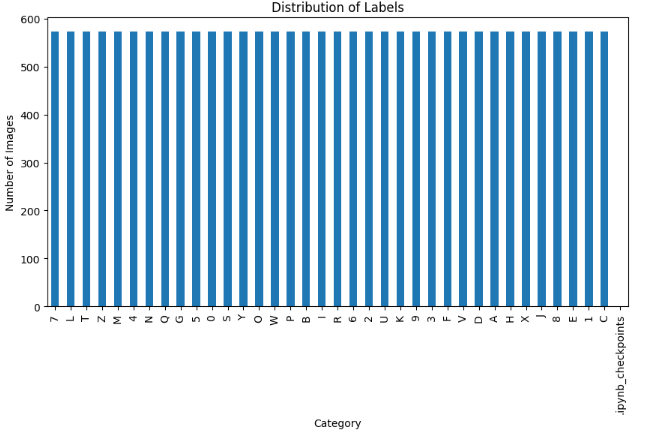
**1.2.4 Visualizing Sample Images**



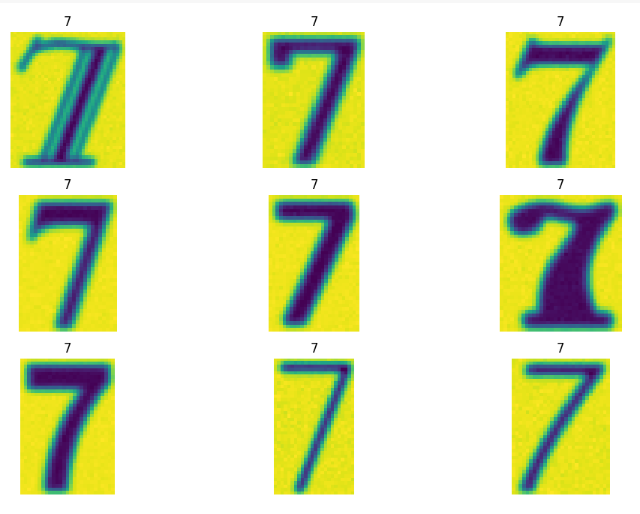
**1.2.5 Creating DataFrame for Image Paths and Labels**



**1.2.6 Visualizing Label Distribution**

All alphabets have the same number of images.

#### **1.2.7 Visualizing Sample Images with Labels**



**1.2.8 Analyzing Image Shapes**

The below snippet shows the height, width and its count in the dataset for some of the images.



**1.2.9 Calculating Image Dimensions' Statistics**



**2. Literature Review**

### **2.1 Extension RDA (Research Domain Analysis)**

Optical Character Recognition (OCR) has evolved significantly over the years, integrating various machine learning and deep learning techniques to improve accuracy and versatility. Modern OCR systems are designed to handle a variety of fonts and styles, making them robust for different applications such as digitizing books, automating data entry, and aiding visually impaired individuals.

### **2.2 Problem Statement**

The goal of this project is to develop an OCR model capable of recognizing characters from a dataset containing various fonts and styles. The model should be able to accurately classify characters in a diverse set of images, providing reliable text recognition for real-world applications.

**3. Methodology**

### **3.1 Data Preparation**

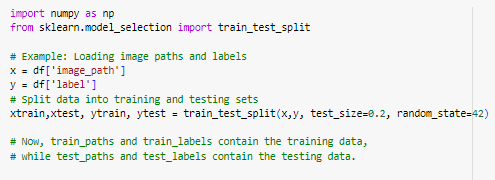
* **Data Extraction**: The dataset is extracted from a ZIP file.
* **Data Augmentation**: Using ImageDataGenerator to perform real-time data augmentation for the training and validation sets.

### **3.2 Model Building**

The model is built using Artificial Neural Networks (ANN) with TensorFlow Keras. We experimented with different activation functions and optimizers:

* **SGD Optimizer**
* **Adam Optimizer**
* **Different Activation Functions (ReLU, Sigmoid)**

**3.3 Train-Test Split**



### **3.4 ANN Model Implementation**

**Using SGD optimizer**



**Using ADAM optimizer and changing the learning rate**



## **4. Results**

### **4.1 Evaluation**

* **SGD Optimizer**: The model achieved a validation accuracy of approximately 22%.
* **Adam Optimizer**: The model achieved a validation accuracy of approximately 80%.
* **Sigmoid Activation Function**: The model's performance with sigmoid activation was comparable to ReLU but slightly less effective.

## **5. Conclusion**

### **5.1 What Worked**

* The use of the Adam optimizer with a learning rate of 0.001 provided the best results, achieving a validation accuracy of around 80%.
* Data augmentation techniques such as rescaling, validation split, and horizontal flipping helped improve model performance by providing more varied training data.

### **5.2 What Didn't Work**

* The SGD optimizer with a learning rate of 0.01 did not perform as well as the Adam optimizer.
* The sigmoid activation function, while effective, was not as robust as ReLU for this dataset.

### **5.3 Future Work**

* Experimenting with deeper neural networks and different architectures such as Convolutional Neural Networks (CNNs) could potentially improve accuracy further.
* Implementing advanced data augmentation techniques and preprocessing steps could help in handling more complex datasets with greater variety in fonts and styles.