To,

IITD-AIA Foundation of Smart Manufacturing

**Subject: Weekly Progress Report for Week-5**

Respected sir,

***I have completed the following tasks in the past few days ( July /3,4,5,6,7,8/ )***

***July 3 :***

***Gained knowledge about RESNET - This approach makes it possible to train the network on thousands of layers without affecting performance.***

***Advantage - Improves the efficiency of deep neural networks with more neural layers while minimizing the percentage of errors.***

***Resnet algo -***

```
import tensorflow as tf

from tensorflow.keras.layers import Conv2D, BatchNormalization, Activation, Add, GlobalAveragePooling2D, Dense

from tensorflow.keras import Input, Model

def residual_block(input_tensor, filters, strides=(1, 1), use_projection=False):

    shortcut = input_tensor



    # First convolutional layer

    x = Conv2D(filters, kernel_size=(3, 3), strides=strides, padding='same')(input_tensor)

    x = BatchNormalization()(x)

    x = Activation('relu')(x)



    # Second convolutional layer

    x = Conv2D(filters, kernel_size=(3, 3), padding='same')(x)
```

```
    x = BatchNormalization()(x)




    if use_projection or strides != (1, 1)

        shortcut = Conv2D(filters, kernel_size=(1, 1), strides=strides,
padding='same')(shortcut)

        shortcut = BatchNormalization()(shortcut)

   x= Add()([x, shortcut])

    x = Activation('relu')(x)



    return x

def resnet(input_shape, num_classes, num_layers):

    input_tensor = Input(shape=input_shape)

   x= Conv2D(64, kernel_size=(7, 7), strides=(2, 2),
padding='same')(input_tensor)

    x = BatchNormalization()(x)

    x = Activation('relu')(x)

    x = tf.keras.layers.MaxPooling2D(pool_size=(3, 3), strides=(2, 2),
padding='same')(x)

    filters = 64

    for i in range(num_layers):

        if i == 0:

            x = residual_block(x, filters, use_projection=True)

        else:

            x = residual_block(x, filters)
```

```
        if i == num_layers // 2 - 1:
            filters *= 2
    x = GlobalAveragePooling2D()(x)
    x = Dense(num_classes, activation='softmax')(x)

    # Create the model
    model = Model(inputs=input_tensor, outputs=x)

    return model
```

*July 4:*

*Gained knowledge about different steps in Image Processing in OpenCV -*

*The process of transforming an image into a digital form and performing certain operations to get some useful information from it.*

*Changing Color spaces*

*Learn to change images between different color spaces. Plus learn to track a colored object in a video.*

*Geometric Transformations of Images*

*Learn to apply different geometric transformations to images like rotation, translation etc.*

*Image Thresholding*

*Learn to convert images to binary images using global thresholding, Adaptive thresholding, Otsu's binarization etc*

***Smoothing Images***

***Learn to blur the images, filter the images with custom kernels etc.***

***Morphological Transformations***

***Learn about morphological transformations like Erosion, Dilation, Opening, Closing etc***

***Image Gradients***

***Learn to find image gradients, edges etc.***

***Canny Edge Detection***

***Learn to find edges with Canny Edge Detection***

***Image Pyramids***

***Learn about image pyramids and how to use them for image blending***

***Contours in OpenCV***

***All about Contours in OpenCV***

***Histograms in OpenCV***

***All about histograms in OpenCV***

***Image Transforms in OpenCV***

***Meet different Image Transforms in OpenCV like Fourier Transform, Cosine Transform etc.***

***Template Matching***

***Learn to search for an object in an image using Template Matching***

***Hough Line Transform***

***Learn to detect lines in an image***

***Hough Circle Transform***

***Learn to detect circles in an image***

***Image Segmentation with Watershed Algorithm***

***Learn to segment images with watershed segmentation***

***Interactive Foreground Extraction using GrabCut Algorithm***

***Learn to extract foreground with GrabCut algorithm***

***Reference : https://youtu.be/oUJs03eZ0S8***

***https://www.simplilearn.com/image-processing-article#:~:text=Image%20processing%20is%20the%20process,certain%20predetermined%20signal%20processing%20methods.***

**July 5:**

***Today I get to know about how to work on data set and tried to implement on that .***

***Steps :***

*1 )Understand the dataset's structure and variables.*

*2 )Clean and preprocess the data.*

*3 )Perform exploratory data analysis (EDA).*

*4 )Select and engineer relevant features.*

*5 )Choose a suitable model and train it.*

*6 )Evaluate the model's performance.*

*7 )Deploy the model for predictions.*

*8) Monitor and iterate for improvement*

**Code :**

```
import numpy as np

import pandas as pd

import os

 import matplotlib.pyplot as plt

%matplotlib inline

import seaborn as sns

for dirname,_, filenames in os.walk('/kaggle/input'):

for filename in filenames:

print(os.path.join(dirname, filename))

 data = pd.read_csv(r"C:\Users\Nandan\Desktop\fsm\AC-Piston-Checks-main\data\defect_log.csv")

data.head()

data.shape

 print(data['image_path'].unique())

 print(data['target'].unique()) ##check missing or null values
```

**data.isnull().sum()**

**data.describe()**

**data-pd.get_dummies (data,drop_first=True)**

**data.head()**

**print(data['image_path_defect_1/defect_1.jpeg'].unique())**

**data=pd.get_dummies (data,drop_first=True)**

**data.head()**

**sns.pairplot(data)**

**corrmat-data.corr()**

**top_corr_features-corrmat.index**

**plt.figure(figsize=(20,20))**

**# plot heat map**

**g=sns.heatmap(data[top_corr_features].corr(),annot=True, cmap="RdYlgn")**

**July 6:**

*Today I explored on library LazyPredict .*

*Lazy Predict is a simple Python library that automatically tests and evaluates different machine learning models on our data, giving us a report on their performance without us having to write a lot of code or make complex decisions. It saves lot of time and makes it easy to see which models are promising for our task .*

*Code:*

*# fitting data in LazyRegressor because*

*# here we are solving Regression use case.*

*reg = LazyRegressor(verbose=0,*

*ignore_warnings=False,*

*custom_metric=None)*

*# fitting data in LazyClassifier*

*models, predictions = reg.fit(X_train, X_test,*

*y_train, y_test)*

*# lets check which model did better*

*# on Breast Cancer Dataset*

*print(models)*

*References :*

*https://www.geeksforgeeks.org/lazy-predict-library-in-python-for-machine-learning/*

*https://www.youtube.com/watch?v=IUUpPL5JRUU&pp=ygUMbGF6eSBwcmVkaWN0*

***July 7 :***

***Today I explored on data cleaning .***

***Data cleaning in machine learning involves identifying and addressing issues like missing values, duplicates, outliers, inconsistencies, and noise in a dataset. It involves various techniques such as handling missing data, handling outliers, data transformation, data integration, data validation and verification, and data formatting. The goal of data cleaning is to prepare the data for analysis and ensure that the insights derived from it are accurate and reliable.***

**Data cleansing tools:**

- **OpenRefine**
- **Trifacta Wrangler**
- **TIBCO Clarity**
- **Cloudingo**
- **IBM Infosphere Quality Stage**

**Steps:**

***1) Take a first look at the data.***

***2) See how many missing data points we have.***

***3) Figure out why the data is missing.***

*4) Drop missing values.*

*5) Filling in missing values.*

*Reference:*

*https://www.kaggle.com/code/rtatman/data-cleaning-challenge-handling-missing-values*

*https://www.geeksforgeeks.org/data-cleansing-introduction/*

*July8:*

*Today I  explored on feature selection in machine learning .*

**The goal of feature selection is to improve the model's performance by reducing the dimensionality of the data and removing irrelevant, redundant, or noisy features.**

**Feature selection can be categorized into three types .**

**Filter methods:**
**Assess feature relevance independently of any specific machine learning algorithm.**
**Rank features based on statistical measures or heuristics.**

**Wrapper methods:**
**Evaluate feature subsets by training and testing the model with different combinations of features.**
**Use a specific machine learning algorithm as a black box to determine performance**

**Embedded methods:**
**Incorporate feature selection within the model training process itself.**
**Optimize the model's objective function to implicitly select relevant features .**

- **The choice of technique depends on factors such as dataset size, computational resources, and the specific requirements**
- `import pandas as pd`
- `import numpy as np`
- `from sklearn.feature_selection import SelectKBest`
- `from sklearn.feature_selection import chi2`
-
- `mobile_data = pd.read_csv("../input/mobile-price-classification/train.csv")`
-

- ```
  X = mobile_data.iloc[:,0:20]  #independent variables
  y = mobile_data.iloc[:,-1]    #target variable i.e price range
  ```

***Reference:***

***https://www.kaggle.com/code/piyushagni5/feature-selection-techniques-in-machine-learning***

***https://www.youtube.com/watch?v=vZDDmULsCUU&pp=ygUnZmVhdHVyZSBzZWxlY3Rpb24gaW4gbWFjaGluZSBsZWFybmluZyAg***

-
  ```