TEAM ID: PNT2022TMID16743

AI-based localization and classification of skin disease with erythema

1.INTRODUCTION

1.1 Project Overview:

Now a day's people are suffering from skin diseases, More than 125 million people suffering from Psoriasis also skin cancer rate is rapidly increasing over the last few decades especially Melanoma is most diversifying skin cancer. If skin diseases are not treated at an earlier stage, then it may lead to complications in the body including spreading of the infection from one individual to the other. The skin diseases can be prevented by investigating the infected region at an early stage. The characteristic of the skin images is diversified so that it is a challenging job to devise an efficient and robust algorithm for automatic detection of skin disease and its severity. Skin tone and skin colour play an important role in skin disease detection

To overcome the above problem, we are building a model which is used for the prevention and early detection of skin cancer, psoriasis. Basically, skin disease diagnosis depends on the different characteristics like colour, shape, texture etc. Here the person can capture the images of skin and then the image will be sent the trained model. The model analyses the image and detect whether the person is having skin disease or not.

1.2 Purpose:

We classify each cluster into different common skin diseases using another neural network model. Our segmentation model achieves better performance compared to previous studies, and also achieves a near-perfect sensitivity score in unfavorable conditions. Our classification model is more accurate than a baseline model trained without segmentation, while also being able to classify diseases within a single image.

2.LITERATURE SURVEY

2.1 Existing Problem:

An inherent disadvantage of clustering a skin disease is its lack of robustness against noise. Clustering algorithms rely on the identification of a centroid that can generalize a cluster of data. Noisy data, or the presence of outliers, can significantly degrade the performance of these algorithms. Therefore, with noisy datasets, caused by images with different types of lighting, non-clustering algorithms may be preferred. Owing to the disadvantages of these traditional approaches, convolution neural networks (CNNs) have gained popularity because of their ability to extract high-level features with minimal preprocessing. By learning to accurately create a higher-resolution image, CNNs can determine the location of the targets to segment.

2.2 References:

- Doi, K. Computer-aided diagnosis in medical imaging: Historical review, current status and future potential. Compute. Med. Imaging Graph.
- Yoshida, H. & Dachman, A. H. Computer-aided diagnosis for CT colonography. Semin. Ultrasound CT MRI.
- Trabelsi, O., Tlig, L., Sayadi, M. & Fnaiech, F., Skin disease analysis and tracking based on image segmentation. 2013 International Conference on Electrical Engineering and Software Applications, Hammamet, 1–7.'
- Rajab, M. I., Woolfson, M. S. & Morgan, S. P. Application of region-based segmentation and neural network edge detection to skin lesions. Comput. Med. Imaging Graph. 28, 61–68.

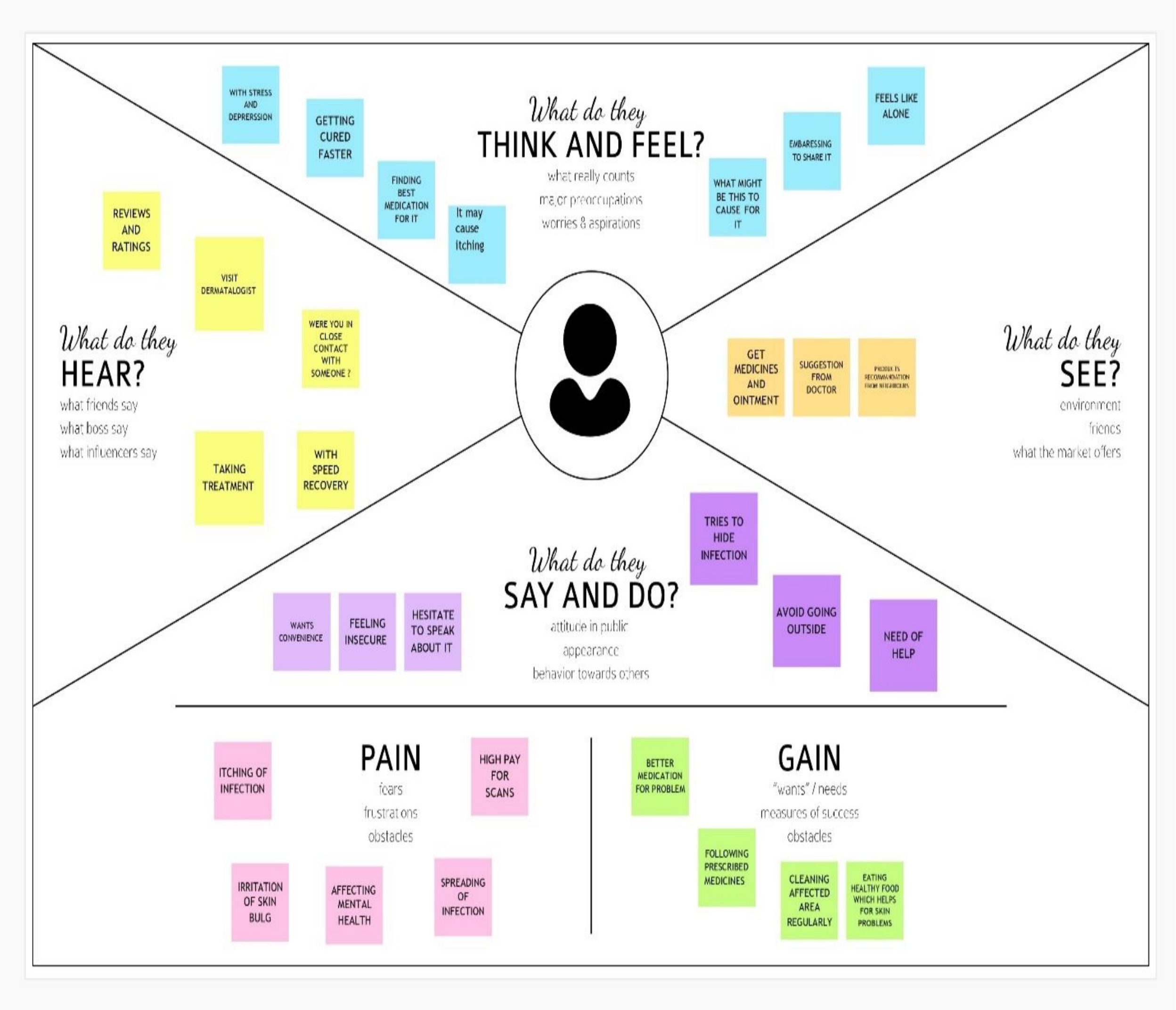
2.3 Problem Statement Definition:

Create a problem statement to understand your customer's point of view. The Customer Problem Statement template helps you focus on what matters to create experiences people will love.

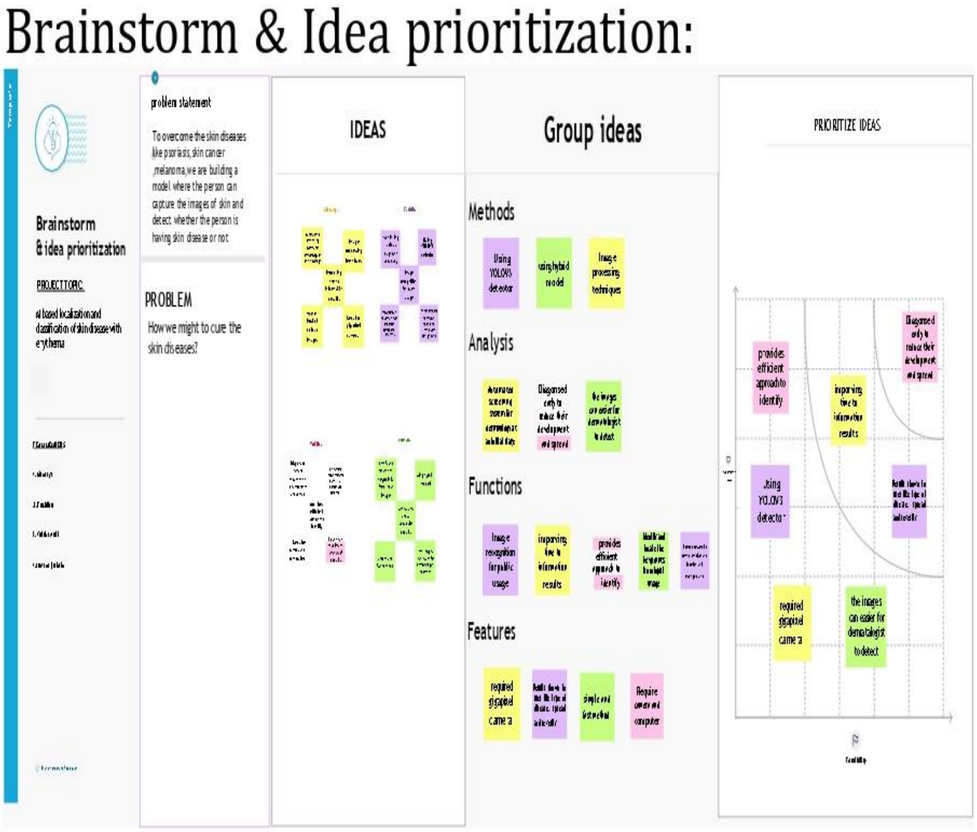
A well-articulated customer problem statement allows you and your team to find the ideal solution for the challenges your customers face. Throughout the process, you’ll also be able to empathize with your customers, which helps you better understand how they perceive your product or service.

3. IDEATION AND PROPOSED SOLUTION

3.1 Empathy Map canvas:



3.2 Ideation and Brainstorming:



Brainstorm & Idea prioritization:

Brainstorm & idea prioritization

PROJECT TOPIC:

ai based localization and classification of skin disease with erythema

problem statement

To overcome the skin diseases like psoriasis, skin cancer ,melanoma,we are building a model where the person can capture the images of skin and detect whether the person is having skin disease or not

PROBLEM

How we might to cure the skin diseases?

IDEAS

Group ideas

Methods

- Using YOLOV3 detector
- using hybrid model
- Image processing techniques

Analysis

- Diagnosed early to reduce their development and spread
- the images can easier for dermatologist to detect

Functions

- Image recognition for public usage
- improving time to information results
- provides efficient approach to identify

Features

- required gigapixel camera
- simple and fast method
- Require camera and computer

PRIORITIZE IDEAS

- provides efficient approach to identify
- improving time to information results
- Diagnosed early to reduce their development and spread
- Using YOLOV3 detector
- required gigapixel camera
- the images can easier for dermatologist to detect

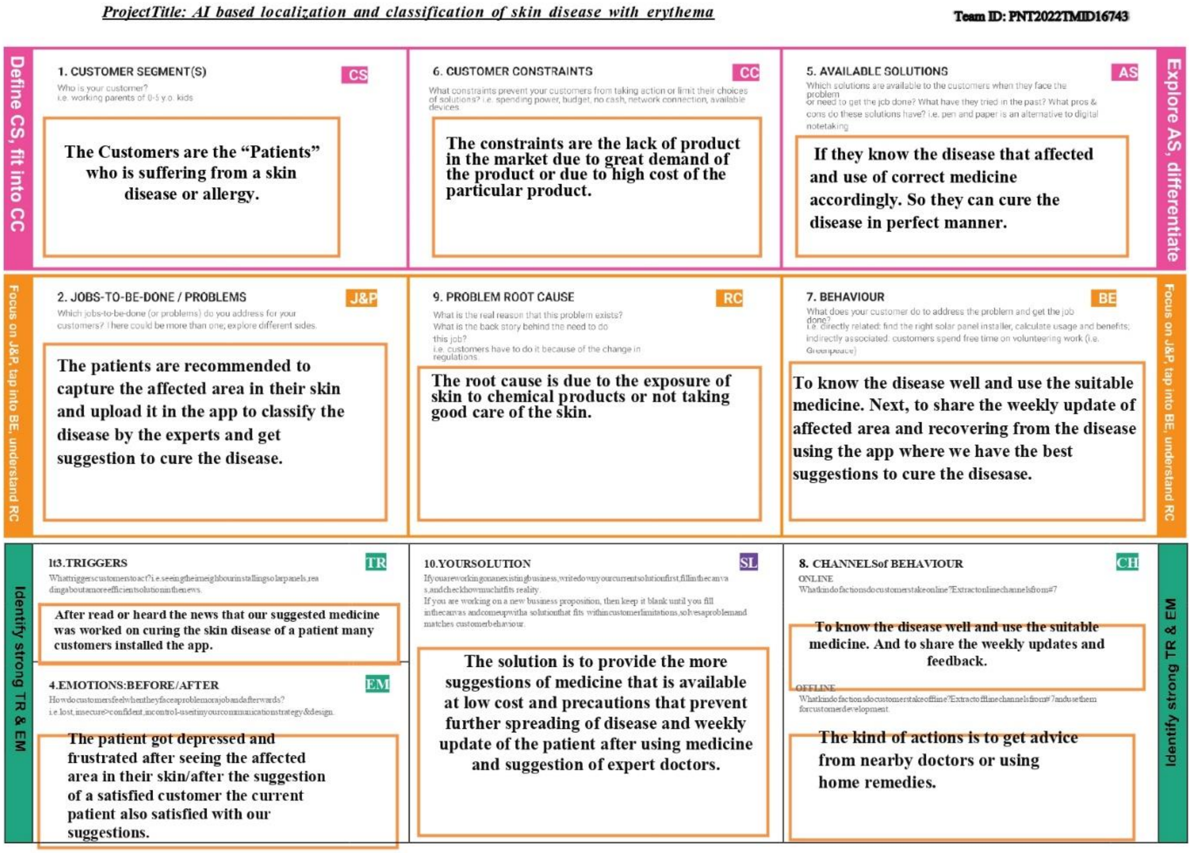
3.3 Proposed solution:

S.No.	Parameter	Description
1.	Problem Statement (Problem to besolved)	Now day's skin diseases become more commonproblem in human life. Most of these diseases are dangerous and harmful, particularly if not treated at an initial stage. People do not treat skin diseases seriously. Sometimes, most of thepeople treat these infections of the skin using their own household methods. However, if these household treatments are not suitable for that particular skin problem then it would affect the skin. Also they may not be aware of severe problem of skin diseases. Skin diseases have tendency to pass from one person to another person easily. Hence it is very important to control it at earlier stage to prevent it from spreading in people. The damage done to the skin due to skin diseases also could damage the self-confidence, mental confidence as well as wellbeing of people. Therefore, the skin diseases are become a hugeproblem among people. It has become an important thing to treat these skin diseases properly at the earlier stages itself to prevent serious damage to skin. This system would help to solve this problem to a great extent. Since it

		system would allow users to determine the skin diseases to provide treatments or advice topatient by making use of images of skin infected with the disease and by obtaining information from the patient.
2.	Idea / Solution description	Let us assume that the own monitoring system. Proposed type of operation is using camera with motion sensor which collects image of thepeople for analysis of the skin. Reliable system design. Accurate system design with classification accuracy of 100 %
3.	Novelty / Uniqueness	This System to classify skin lesions. The Unique features space consist of Grayscale, colour redness and texture of the skinn.
4.	Social Impact / Customer Satisfaction	Social impacts : Especially those whose skin disorder have a visible localization (acne patients), show lower levels of self-esteem, as well as lower levels ofperceived social support and Depression. Customer Satisfaction : To give Accuracy .
5.	Business Model (Revenue Model)	Mobile Application models , Give some Awareness about this Techniques , Recommended by Dermatologists.

6.	Scalability of the Solution	Scalability is important , because it decides thequality or value of the technique or the Result.

3.4 Proposed solution Fit:



ProjectTitle: AI based localization and classification of skin disease with erythema

Team ID: PNT2022TMID16743

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

Who is your customer?
i.e. working parents of 0-5 y.o. kids

The Customers are the "Patients" who is suffering from a skin disease or allergy.

6. CUSTOMER CONSTRAINTS CC

What constraints prevent your customers from taking action or limit their choices of solutions? i.e. spending power, budget, no cash, network connection, available devices

The constraints are the lack of product in the market due to great demand of the product or due to high cost of the particular product.

5. AVAILABLE SOLUTIONS AS

Which solutions are available to the customers when they face the problem or need to get the job done? What have they tried in the past? What pros & cons do these solutions have? i.e. pen and paper is an alternative to digital notetaking

If they know the disease that affected and use of correct medicine accordingly. So they can cure the disease in perfect manner.

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P

Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different sides.

The patients are recommended to capture the affected area in their skin and upload it in the app to classify the disease by the experts and get suggestion to cure the disease.

9. PROBLEM ROOT CAUSE RC

What is the real reason that this problem exists?
What is the back story behind the need to do this job?
i.e. customers have to do it because of the change in regulations.

The root cause is due to the exposure of skin to chemical products or not taking good care of the skin.

7. BEHAVIOUR BE

What does your customer do to address the problem and get the job done?
i.e. directly related: find the right solar panel installer, calculate usage and benefits; indirectly associated: customers spend free time on volunteering work (i.e. Greenpeace)

To know the disease well and use the suitable medicine. Next, to share the weekly update of affected area and recovering from the disease using the app where we have the best suggestions to cure the disesase.

Focus on J&P, tap into BE, understand RC

Identify strong TR & EM

lt3.TRIGGERS TR

Whattriggerscustomerstoact?i.e.seeingtheirneighbourinstallingsolarpanels,readingaboutamoreefficientsolutioninthenews.

After read or heard the news that our suggested medicine was worked on curing the skin disease of a patient many customers installed the app.

4.EMOTIONS:BEFORE/AFTER EM

Howdocustomersfeelwhentheyfaceaproblemorajobandafterwards?
i.e.lost,insecure>confident,incontrol-useitinyourcommunicationstrategy&design.

The patient got depressed and frustrated after seeing the affected area in their skin/after the suggestion of a satisfied customer the current patient also satisfied with our suggestions.

10.YOURSOLUTION SL

Ifyouareworkingonanexistingbusiness,writedownyourcurrentsolutionfirst,fillinthecanvas,andcheckhowmuchitfits reality.
If you are working on a new business proposition, then keep it blank until you fill inthecanvas andcomeupwitha solutionthat fits withincustomerlimitations,solvesaproblemand matches customerbehaviour.

The solution is to provide the more suggestions of medicine that is available at low cost and precautions that prevent further spreading of disease and weekly update of the patient after using medicine and suggestion of expert doctors.

8. CHANNELSof BEHAVIOUR CH

ONLINE
Whatkindofactionsdocustomerstakeonline?Extractonlinechannelsfrom#7

To know the disease well and use the suitable medicine. And to share the weekly updates and feedback.

OFFLINE
Whatkindofactionsdocustomerstakeoffline?Extractofflinechannelsfrom#7andusethem forcustomerdevelopment.

The kind of actions is to get advice from nearby doctors or using home remedies.

Identify strong TR & EM

4.REQUIREMENTS ANALYSIS

4.1 Functional requirements:

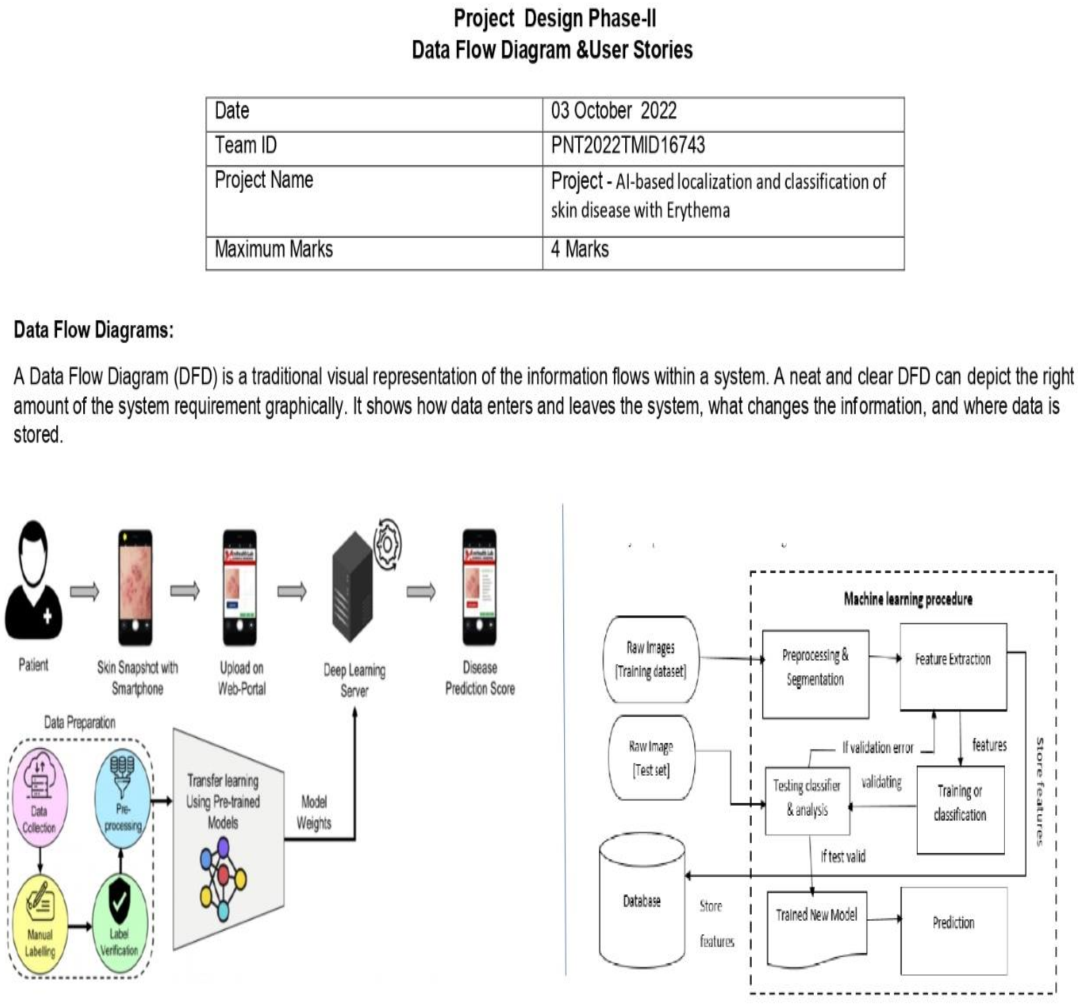
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Mobile Number Registration through Google AccountRegistration through Facebook
FR-2	User Confirmation	Confirmation via Email Confirmation via Call Confirmation via OTP
FR-3	Patient Image Capturing Process	Provide Access to Capture Image Through Camera Provide Access to Upload Image Through Gallery
FR-4	Patient Medicine Reminder	Remind the Patients to take their Medicines/ointmentsAt right time through remaindering alarm.
FR-5	Suggestion Box	Patients can take suggestions from the Doctors throughChats.
FR-6	Flareup Cycles	Patients can know their medicine level from doctorsThrough message.

4.2 Non-Functional Requirements:

FR No.	Non-FunctionalRequirement	Description
NFR-1	**Usability**	Our Mobile phone application designed to improve the quality of patient-heldphotos, and was developed to generate and hold their own skin images to helpguide their skin care.
NFR-2	**Security**	Data privacy and security practices may vary based on users and their age
NFR-3	**Reliability**	Easy to use app to get personalized answers to your skin conditionsquestions.
NFR-4	**Performance**	Good treatments are available for a variety of skin conditions includingrash, itchy skin, skin fungus etc.
NFR-5	**Availability**	Our app helps you to screen your skin symptoms and prepare for yourpractitioner visit.
NFR-6	**Scalability**	The app gives users evidence-based dermatologist approved health information insights on diseases affecting various parts of our body.

5.PROJECT DESIGN

5.1 Data flow diagrams:



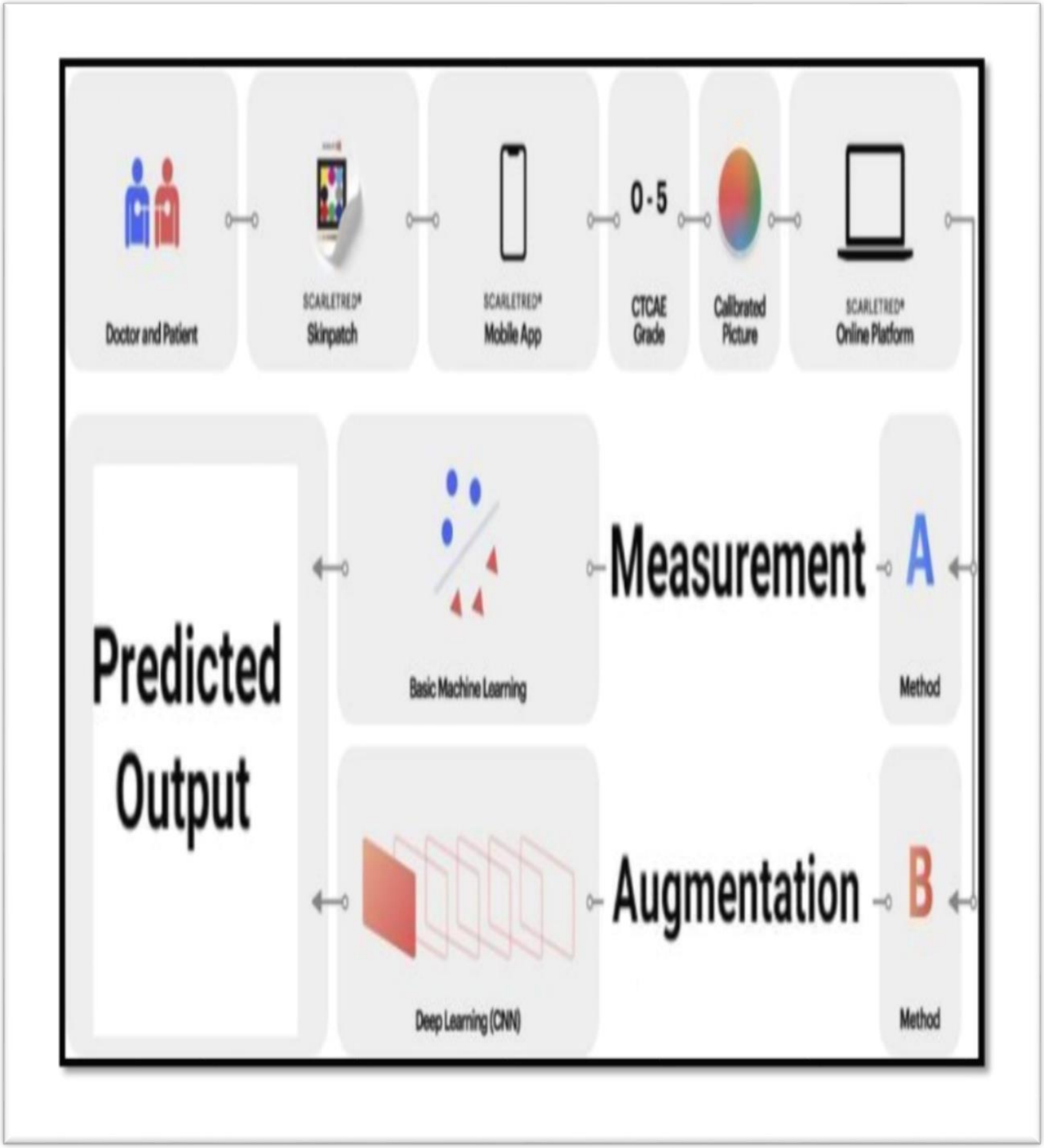
Project Design Phase-II
Data Flow Diagram &User Stories

Date	03 October 2022
Team ID	PNT2022TMID16743
Project Name	Project - AI-based localization and classification of skin disease with Erythema
Maximum Marks	4 Marks

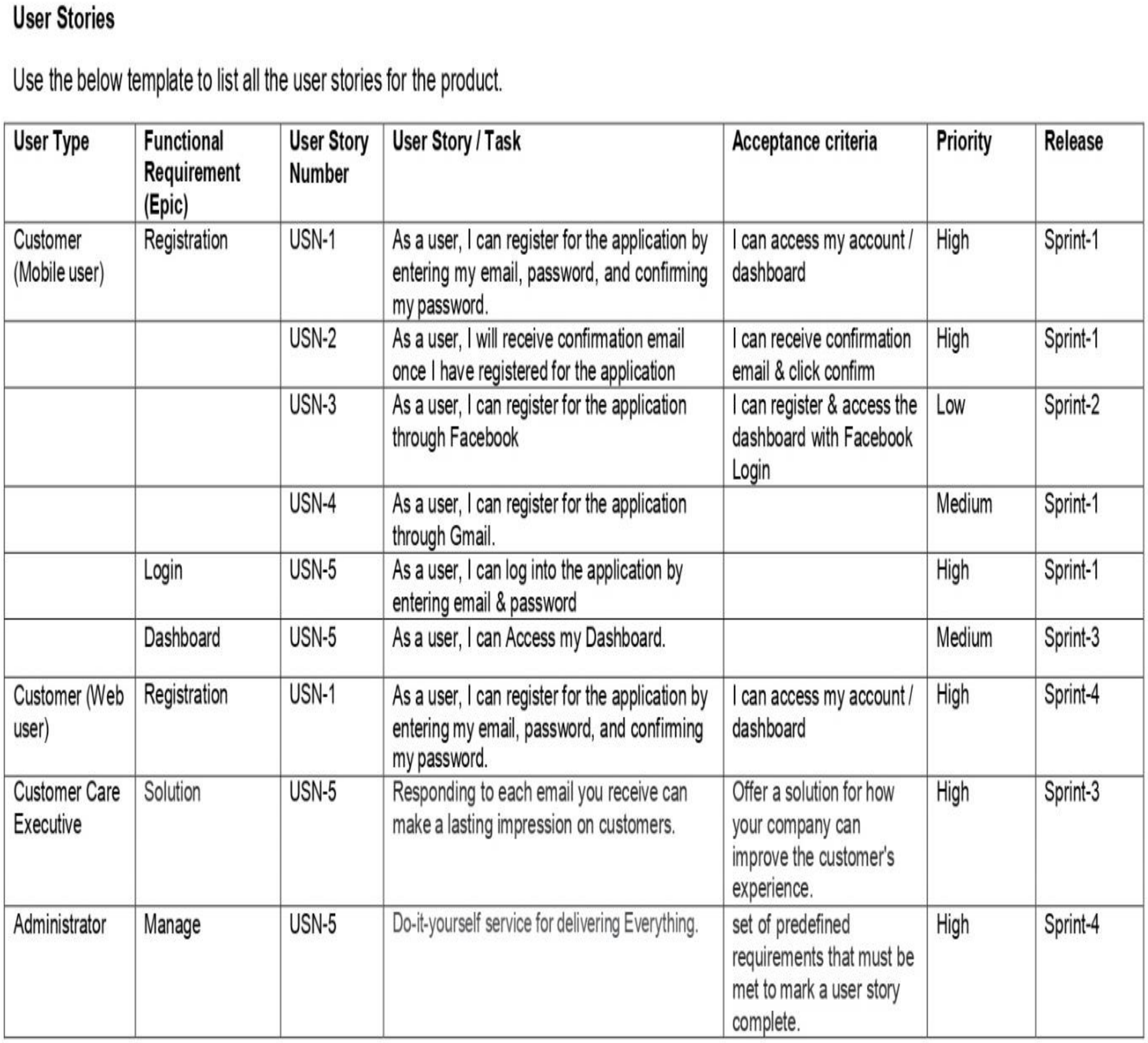
Data Flow Diagrams:

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

5.2 Solution and Technical Architecture:



5.3 User Stories:



User Stories

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail.		Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password		High	Sprint-1
	Dashboard	USN-5	As a user, I can Access my Dashboard.		Medium	Sprint-3
Customer (Web user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-4
Customer Care Executive	Solution	USN-5	Responding to each email you receive can make a lasting impression on customers.	Offer a solution for how your company can improve the customer's experience.	High	Sprint-3
Administrator	Manage	USN-5	Do-it-yourself service for delivering Everything.	set of predefined requirements that must be met to mark a user story complete.	High	Sprint-4

6.PROJECT PLANNING AND SCHEDULING

6.1 Sprint Planning and Estimation:

Sprints are the backbone of any good Agile development team. And the better prepared you are before a sprint, the more likely you are to hit your goals. You and your team requires communication and clarity and make sure that your expectations are understood and can be done by your team is key to keeping everyone motivated and productive.

Step 1: Review your product roadmap

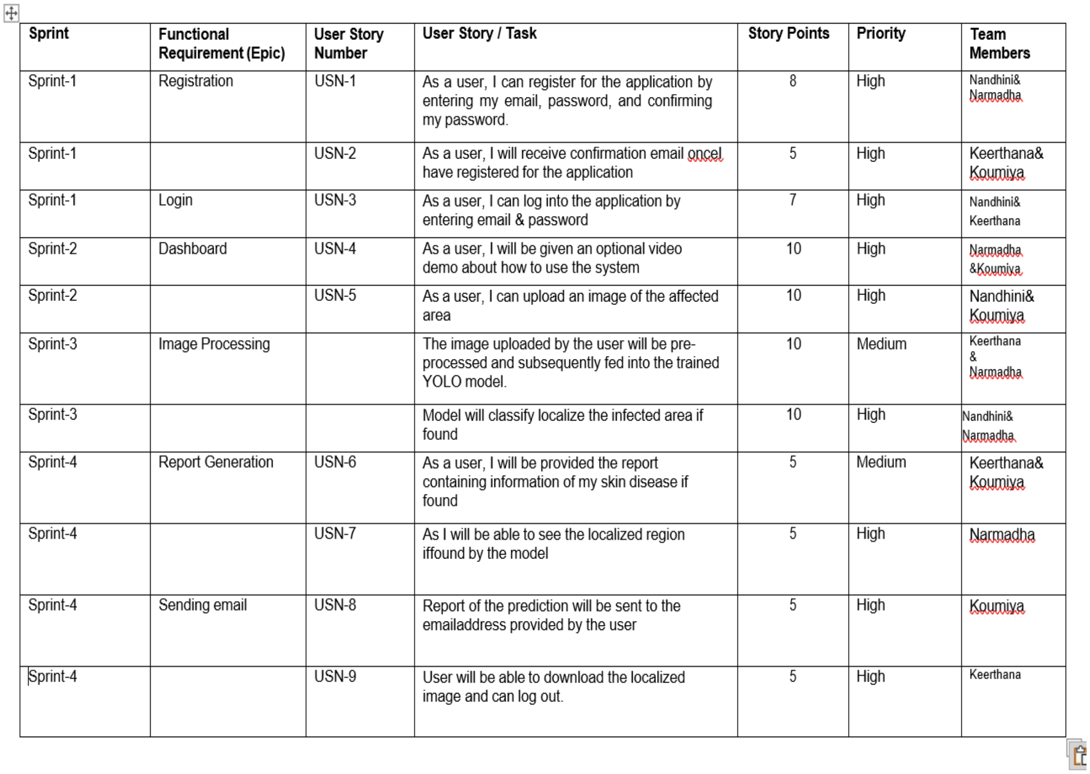
Step 2: Groom your product backlog and update user stories

Step 3: Propose a sprint goal and backlog before the sprint planning meeting

Step 4: Use data and experience to supercharge your Sprint planning meeting

Step 5: Walk through each user story and describe what tasks need to be done

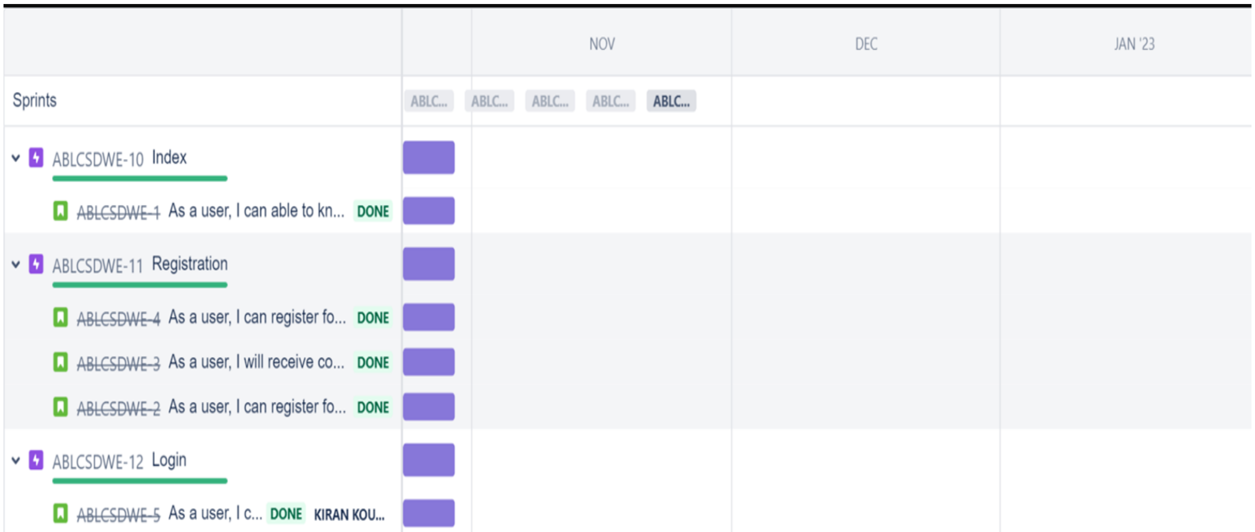
6.2 Sprint Delivery Schedule:



Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	8	High	Nandhini& Narmadha
Sprint-1		USN-2	As a user, I will receive confirmation email oncel have registered for the application	5	High	Keerthana& Koumiya
Sprint-1	Login	USN-3	As a user, I can log into the application by entering email & password	7	High	Nandhini& Keerthana
Sprint-2	Dashboard	USN-4	As a user, I will be given an optional video demo about how to use the system	10	High	Narmadha &Koumiya
Sprint-2		USN-5	As a user, I can upload an image of the affected area	10	High	Nandhini& Koumiya
Sprint-3	Image Processing		The image uploaded by the user will be pre-processed and subsequently fed into the trained YOLO model.	10	Medium	Keerthana & Narmadha
Sprint-3			Model will classify localize the infected area if found	10	High	Nandhini& Narmadha
Sprint-4	Report Generation	USN-6	As a user, I will be provided the report containing information of my skin disease if found	5	Medium	Keerthana& Koumiya
Sprint-4		USN-7	As I will be able to see the localized region iffound by the model	5	High	Narmadha
Sprint-4	Sending email	USN-8	Report of the prediction will be sent to the emailaddress provided by the user	5	High	Koumiya
Sprint-4		USN-9	User will be able to download the localized image and can log out.	5	High	Keerthana

6.3 Reports from JIRA:

Sprint 1:



Sprints

ABLCSDWE-10 Index

~~ABLCSDWE-1~~ As a user, I can able to kn... DONE

ABLCSDWE-11 Registration

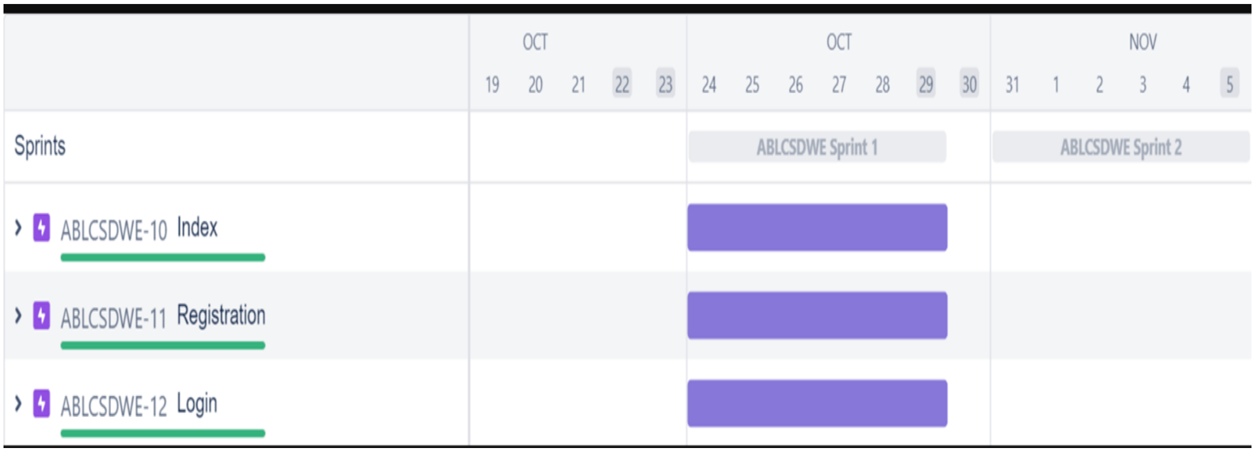
~~ABLCSDWE-4~~ As a user, I can register fo... DONE

~~ABLCSDWE-3~~ As a user, I will receive co... DONE

~~ABLCSDWE-2~~ As a user, I can register fo... DONE

ABLCSDWE-12 Login

~~ABLCSDWE-5~~ As a user, I c... DONE KIRAN KOU...



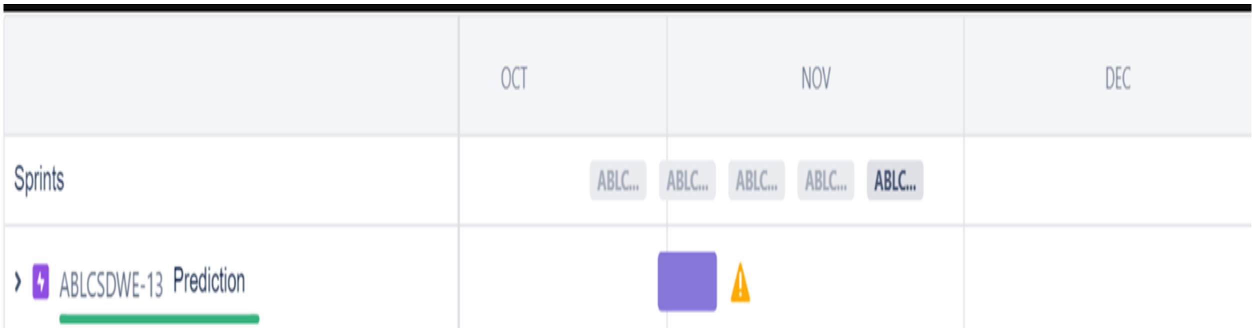
Sprints

ABLCSDWE-10 Index

ABLCSDWE-11 Registration

ABLCSDWE-12 Login

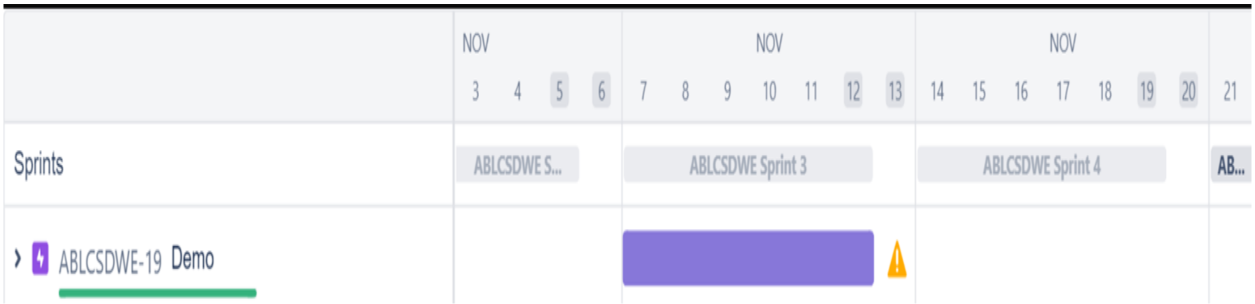
Sprint 2:



Sprints

ABLCSDWE-13 Prediction

Sprint 3:

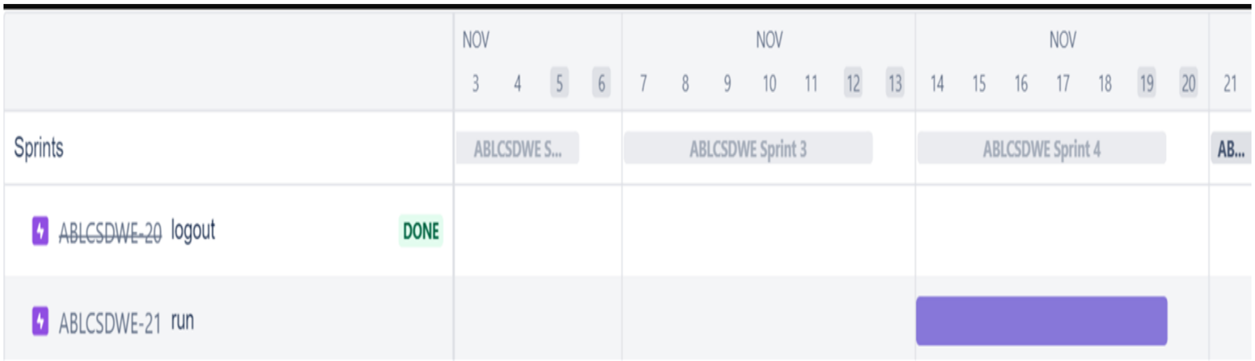


NOV 3 4 5 6 | NOV 7 8 9 10 11 12 13 | NOV 14 15 16 17 18 19 20 | 21

Sprints: ABLCSDWE S... | ABLCSDWE Sprint 3 | ABLCSDWE Sprint 4 | AB...

› ABLCSDWE-19 Demo

Sprint 4:



NOV 3 4 5 6 | NOV 7 8 9 10 11 12 13 | NOV 14 15 16 17 18 19 20 | 21

Sprints: ABLCSDWE S... | ABLCSDWE Sprint 3 | ABLCSDWE Sprint 4 | AB...

~~ABLCSDWE-20~~ logout DONE

ABLCSDWE-21 run

7. CODING & SOLUTIONING

7.1 Microsoft's Visual Object Tagging Tool (VoTT):

It is an open source annotation and labeling tool for image and video assets.

VoTT is a React + Redux Web application, written in TypeScript.

Features include:

- The ability to label images or video frames
- Extensible model for importing data from local or cloud storage providers
- Extensible model for exporting labeled data to local or cloud storage providers

Using VoTT:

- Creating Connections
- Creating a New Project
    - Project Settings
    - Security Tokens
- Labeling an Image
- Labeling a Video
- Exporting Labels

7.2 YOLO Project Structure:

It was proposed by Joseph Redmond et al. in 2015. It was proposed to deal with the problems faced by the object recognition models at that time, Fast R-CNN is one of the state-of-the-art models at that time but it has its own challenges such as this network cannot be used in real-time, because it takes 2-3 seconds to predicts an image and therefore cannot be used in real-time.

Whereas, in YOLO we have to look only once in the network i.e. only one forward pass is required through the network to make the final predictions.

Code:

```
from PIL import Image
from os import path, makedirs
import os
import re
import pandas as pd
import sys
import argparse



from Convert_Format import convert_vott_csv_to_yolo



def get_parent_dir(n=1):
    """ returns the n-th parent dicrectory of the current
    working directory """
    current_path = os.path.dirname(os.path.abspath(__file__))
    for k in range(n):
        current_path = os.path.dirname(current_path)
    return current_path



sys.path.append(os.path.join(get_parent_dir(1), "Utils"))



Data_Folder = os.path.join(get_parent_dir(1), "Data")
VoTT_Folder = os.path.join(
    Data_Folder, "Source_Images", "Training_Images", "vott-csv-export"
```

```
)
VoTT_csv = os.path.join(VoTT_Folder, "Annotations-export.csv")
YOLO_filename = os.path.join(VoTT_Folder, "data_train.txt")

model_folder = os.path.join(Data_Folder, "Model_Weights")
classes_filename = os.path.join(model_folder, "data_classes.txt")

if __name__ == "__main__":
    # surpress any inhereted default values
    parser = argparse.ArgumentParser(argument_default=argparse.SUPPRESS)
    """
    Command line options
    """
    parser.add_argument(
        "--VoTT_Folder",
        type=str,
        default=VoTT_Folder,
        help="Absolute path to the exported files from the image tagging step with VoTT. Default
is "
        + VoTT_Folder,
    )

    parser.add_argument(
        "--VoTT_csv",
        type=str,
        default=VoTT_csv,
        help="Absolute path to the *.csv file exported from VoTT. Default is "
        + VoTT_csv,
    )
    parser.add_argument(
        "--YOLO_filename",
        type=str,
        default=YOLO_filename,
        help="Absolute path to the file where the annotations in YOLO format should be saved.
Default is "
        + YOLO_filename,
    )

    FLAGS = parser.parse_args()

    # Prepare the dataset for YOLO
    multi_df = pd.read_csv(FLAGS.VoTT_csv)
    labels = multi_df["label"].unique()
    labeldict = dict(zip(labels, range(len(labels))))
    multi_df.drop_duplicates(subset=None, keep="first", inplace=True)
    train_path = FLAGS.VoTT_Folder
    convert_vott_csv_to_yolo(
        multi_df, labeldict, path=train_path, target_name=FLAGS.YOLO_filename
    )

    # Make classes file
    file = open(classes_filename, "w")

    # Sort Dict by Values
    SortedLabelDict = sorted(labeldict.items(), key=lambda x: x[1])
    for elem in SortedLabelDict:
        file.write(elem[0] + "\n")
    file.close()
```

7.3 Database Schema:

A database schema defines how data is organized within a relational database; this is inclusive of logical constraints such as, table names, fields, data types, and the relationships between these entities. Schemas commonly use visual representations to communicate the architecture of the database, becoming the foundation for an organization’s data management discipline.

A database schema is considered the “blueprint” of a database which describes how the data may relate to other tables or other data models. However, the schema does not actually contain data.

key benefits of database schemas include:

- **Access and security**: Database schema design helps organize data into separate entities, making it easier to share a single schema within another database.
- **Organization and communication:** Documentation of database schemas allow for more organization and better communication among internal stakeholders.
- **Integrity**: This organization and communication also helps to ensure data validity.

8. TESTING

8.1 User Acceptance Testing:

User acceptance testing, a testing methodology where the clients/end users involved in testing the product to validate the product against their requirements. It is performed at client location at developer's site.

For industry such as medicine or aviation industry, contract and regulatory compliance testing and operational acceptance testing is also carried out as part of user acceptance testing.

UAT is context dependent and the UAT plans are prepared based on the requirements and NOT mandatory to execute all kinds of user acceptance tests and even coordinated and contributed by testing team.

Acceptance criteria are defined on:

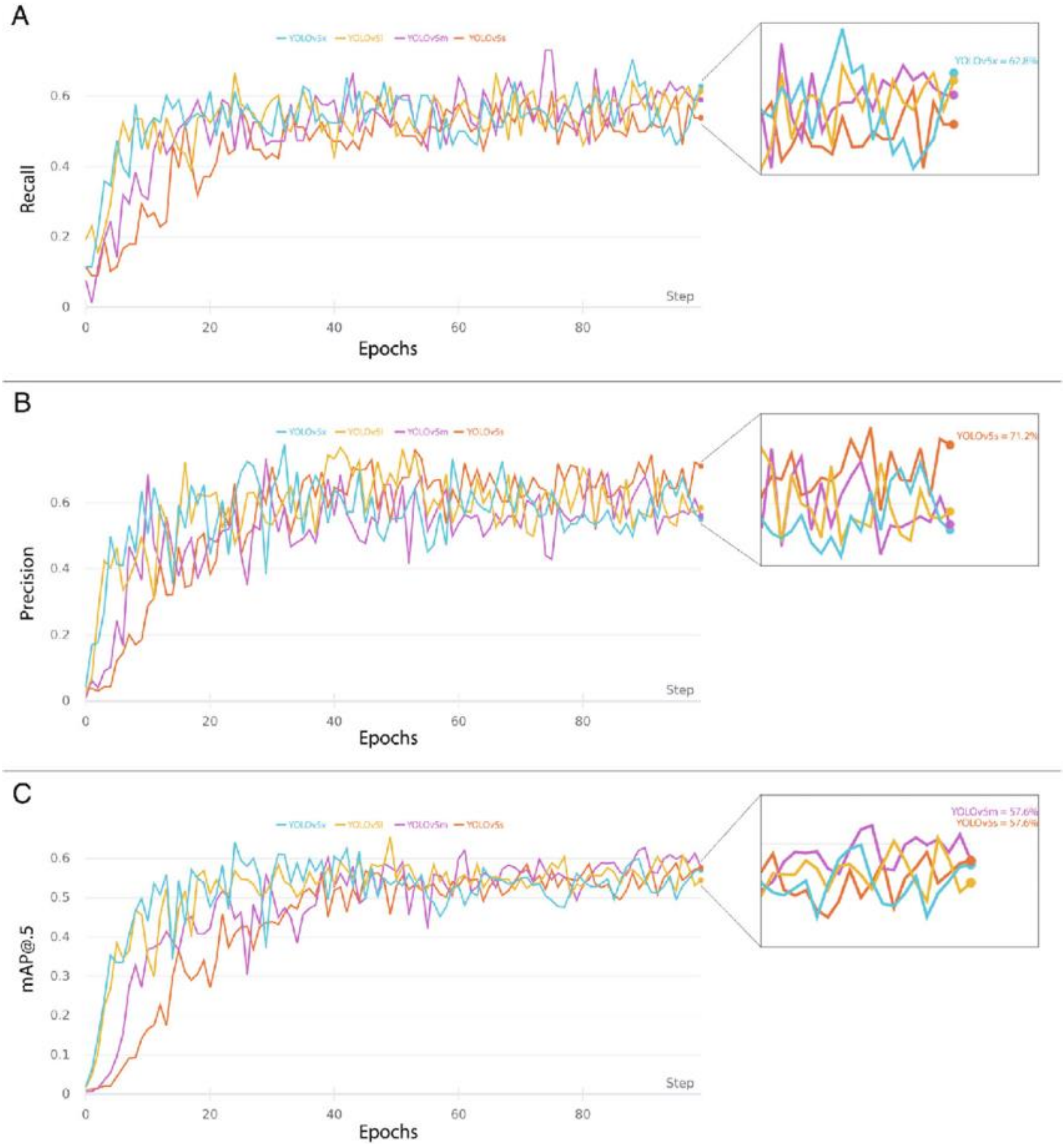
- Functional Correctness and Completeness
- Data Integrity
- Data Conversion
- Usability
- Performance
- Timeliness
- Confidentiality and Availability
- Installability and Upgradability
- Scalability
- Documentation

9. RESULTS

9.1 Performance Metrics:

The performance metrics used for evaluating a classification model:

- Accuracy - The overall accuracy of a model is simply the number of correct predictions divided by the total number of predictions.
- Precision and Recall - Precision measures how good the model is at correctly identifying the positive class. Recall tell us how good the model is at correctly predicting all the positive observations in the dataset.
- F1-score - The F1 score is the harmonic mean of precision and recall. The F1 score will give a number between 0 and 1.
- AUC-ROC - The AUC is the measurement of the entire two-dimensional area under the curve and The ROC (Receiver Operating Characteristics) curve is a plot of the performance of the model


A
YOLOv5x
YOLOv5l
YOLOv5m
YOLOv5s
Recall
0
0.2
0.4
0.6
Step
0
20
40
60
80
Epochs
YOLOv5x = 62.8%
B
YOLOv5x
YOLOv5l
YOLOv5m
YOLOv5s
Precision
0
0.2
0.4
0.6
Step
0
20
40
60
80
Epochs
YOLOv5s = 71.2%
C
YOLOv5x
YOLOv5l
YOLOv5m
YOLOv5s
mAP@.5
0
0.1
0.2
0.3
0.4
0.5
0.6
Step
0
20
40
60
80
Epochs
YOLOv5m = 57.6%
YOLOv5s = 57.6%

10. ADVANTAGES & DISADVANTAGES

Merits:

- In dermatology, although skin disease is a common disease, one in which early detection and classification is crucial for the successful treatment and recovery of patients, dermatologists perform most noninvasive screening tests only with the naked eye.
- This may result in avoidable diagnostic inaccuracies as a result of human error, as the detection of the disease can be easily overlooked.
- Therefore, it would be beneficial to exploit the strengths of CAD using artificial intelligence techniques, in order to improve the accuracy of dermatology diagnosis.

Demerits:

- An inherent disadvantage of clustering a skin disease is its lack of robustness against noise.
- Centroid that can generalize a cluster of data can significantly degrade the performance of these algorithms.
- the degradation problem that occurs when CNN models become too large and complex.
- Hence,We implement skip-connections in both segmentation and classification models.

11. CONCLUSION

The Project AI-Based Localization of Skin Disease With Erythema is used to find whether the person is having erythema or not. And our project helps lots of people to find whether their skin disease is erythema or not. Our website shows the accurate result so it helps the user to check their skin Disease. It is User Friendly Website.

12. FUTURE SCOPE

Future Scope of Our Project AI - Based Localization Of Skin Disease With Erythema is to try new algorithms and improve the accuracy of the result. And also developing a mobile application is our scope of the project

13. APPENDIX

Source Code:

Convert_csv_to_YOLO.py:

```
import os
import re
from os import makedirs, path

import numpy as np
import pandas as pd
from PIL import Image

from Get_File_Paths import ChangeToOtherMachine, GetFileList


def convert_vott_csv_to_yolo(
    vott_df,
    labeldict=dict(zip(["Cat_Face"], [0,])),
    path="",
    target_name="data_train.txt",
    abs_path=False,
):

    # Encode labels according to labeldict if code's don't exist
    if not "code" in vott_df.columns:
        vott_df["code"] = vott_df["label"].apply(lambda x: labeldict[x])
    # Round float to ints
    for col in vott_df[["xmin", "ymin", "xmax", "ymax"]]:
        vott_df[col] = (vott_df[col]).apply(lambda x: round(x))

    # Create Yolo Text file
    last_image = ""
    txt_file = ""

    for index, row in vott_df.iterrows():
        if not last_image == row["image"]:
            if abs_path:
                txt_file += "\n" + row["image_path"] + " "
            else:
                txt_file += "\n" + os.path.join(path, row["image"]) + " "
            txt_file += ",".join(
                [
                    str(x)
                    for x in (row[["xmin", "ymin", "xmax", "ymax", "code"]].tolist())
                ]
            )
        else:
            txt_file += " "
            txt_file += ",".join(
                [
                    str(x)
                    for x in (row[["xmin", "ymin", "xmax", "ymax", "code"]].tolist())
                ]
            )
        last_image = row["image"]
    file = open(target_name, "w")
    file.write(txt_file[1:])
    file.close()
    return True



def csv_from_xml(directory, path_name=""):
```

```
# First get all images and xml files from path and its subfolders
image_paths = GetFileList(directory, ".jpg")
xml_paths = GetFileList(directory, ".xml")
result_df = pd.DataFrame()
if not len(image_paths) == len(xml_paths):
    print("number of annotations doesnt match number of images")
    return False
for image in image_paths:
    target_filename = os.path.join(path_name, image) if path_name else image
    source_filename = os.path.join(directory, image)
    y_size, x_size, _ = np.array(Image.open(source_filename)).shape
    source_xml = image.replace(".jpg", ".xml")
    txt = open(source_xml, "r").read()
    y_vals = re.findall(r"(?:x>\n)(.*)(?:\n</)", txt)
    ymin_vals = y_vals[::2]
    ymax_vals = y_vals[1::2]
    x_vals = re.findall(r"(?:y>\n)(.*)(?:\n</)", txt)
    xmin_vals = x_vals[::2]
    xmax_vals = x_vals[1::2]
    label_vals = re.findall(r"(?:label>\n)(.*)(?:\n</)", txt)
    label_name_vals = re.findall(r"(?:labelname>\n)(.*)(?:\n</)", txt)
    df = pd.DataFrame()
    df["xmin"] = xmin_vals
    df["xmin"] = df["xmin"].astype(float) * x_size
    df["ymin"] = ymin_vals
    df["ymin"] = df["ymin"].astype(float) * y_size
    df["xmax"] = xmax_vals
    df["xmax"] = df["xmax"].astype(float) * x_size
    df["ymax"] = ymax_vals
    df["ymax"] = df["ymax"].astype(float) * y_size
    df["label"] = label_name_vals
```

```
        df["code"] = label_vals
        df["image_path"] = target_filename
        df["image"] = os.path.basename(target_filename)
        result_df = result_df.append(df)
    #     Bring image column first
    cols = list(df.columns)
    cols = [cols[-1]] + cols[:-1]
    result_df = result_df[cols]
    return result_df


def crop_and_save(
    image_df,
    target_path,
    target_file,
    one=True,
    label_dict={0: "house"},
    postfix="cropped",
):
    """Takes a vott_csv file with image names, labels and crop_boxes
    and crops the images accordingly

    Input csv file format:

    image   xmin ymin xmax ymax label
    im.jpg  0    10   100  500  house



    Parameters
    ----------
    df : pd.Dataframe
    The input dataframe with file_names, bounding box info
    and label
source_path : str
    Path of source images
target_path : str, optional
    Path to save cropped images
one : boolean, optional
    if True, only the most central house will be returned

Returns
-------
True if completed succesfully
"""
if not path.isdir(target_path):
    makedirs(target_path)

previous_name = ""
counter = 0
image_df.dropna(inplace=True)
image_df["image_path"] = ChangeToOtherMachine(image_df["image_path"].values)

def find_rel_position(row):
    current_name = row["image_path"]
    x_size, _ = Image.open(current_name).size
    x_centrality = abs((row["xmin"] + row["xmax"]) / 2 / x_size - 0.5)
    return x_centrality

if one:
    centrality = []
    for index, row in image_df.iterrows():
        centrality.append(find_rel_position(row))
        image_df["x_centrality"] = pd.Series(centrality)
        image_df.sort_values(["image", "x_centrality"], inplace=True)
        image_df.drop_duplicates(subset="image", keep="first", inplace=True)
    new_paths = []
    for index, row in image_df.iterrows():
        current_name = row["image_path"]
        if current_name == previous_name:
            counter += 1
        else:
            counter = 0
        imageObject = Image.open(current_name)
        cropped = imageObject.crop((row["xmin"], row["ymin"], row["xmax"], row["ymax"]))
        label = row["label"]
        if type(label) == int:
            label = label_dict[label]
        image_name_cropped = (
            "_".join([row["image"][:-4], postfix, label, str(counter)]) + ".jpg"
        )
        new_path = os.path.join(target_path, image_name_cropped)
        cropped.save(new_path)
        new_paths.append(new_path.replace("\\", "/"))
        previous_name = current_name
    pd.DataFrame(new_paths, columns=["image_path"]).to_csv(target_file)
    return True


if __name__ == "__main__":
    # Prepare the houses dataset for YOLO
    labeldict = dict(zip(["house"], [0,]))
```

```
multi_df                                                                                     =
  r"C:\Users\Admin\Desktop\yolo_structure\Data\Source_Images\Training_Images\vott-csv-
  export\Annotations-export.csv"

convert_vott_csv_to_yolo(
    multi_df,
    labeldict,
    path=r"C:\Users\Admin\Desktop\data\skin",
    target_name= "data_train.txt"
)

# Prepare the windows dataset for YOLO
path = r"C:\Users\Admin\Desktop\yolo_structure\Data\Source_Images\base"
csv_from_xml(path,
  r"C:\Users\Admin\Desktop\data\windows").to_csv(r"C:\Users\Admin\Desktop\yolo_structur
  e\Data\Source_Images\base/annotations.csv")

label_names = [
    "Erythema multiforme (EM)",
    "Erythema chronicum migrans",
    "Erythema migrans",
    "Erythema marginatum",
    "Erythema infectiosum",
    "Erythema nodosum"
]
labeldict = dict(zip(label_names, list(range(6))))
convert_vott_csv_to_yolo(
    csv_from_xml(path, r"C:\Users\Admin\Desktop\data\windows"), labeldict
)
```

Train YOLOv3 Detector:

```
import os
import sys


def get_parent_dir(n=1):
    """ returns the n-th parent dicrectory of the current
    working directory """
    current_path = os.path.dirname(os.path.abspath(__file__))
    for k in range(n):
        current_path = os.path.dirname(current_path)
    return current_path


src_path = os.path.join(get_parent_dir(1), "2_Training", "src")
utils_path = os.path.join(get_parent_dir(1), "Utils")

sys.path.append(src_path)
sys.path.append(utils_path)

import argparse
from keras_yolo3.yolo import YOLO, detect_video
from PIL import Image
from timeit import default_timer as timer
from utils import load_extractor_model, load_features, parse_input, detect_object
import test
import utils
import pandas as pd
import numpy as np
from Get_File_Paths import GetFileList
import random

os.environ["TF_CPP_MIN_LOG_LEVEL"] = "3"
```

```
# Set up folder names for default values
data_folder = os.path.join(get_parent_dir(n=1), "Data")

image_folder = os.path.join(data_folder, "Source_Images")

image_test_folder = os.path.join(image_folder, "Test_Images")

detection_results_folder = os.path.join(image_folder, "Test_Image_Detection_Results")
detection_results_file = os.path.join(detection_results_folder, "Detection_Results.csv")

model_folder = os.path.join(data_folder, "Model_Weights")

model_weights = os.path.join(model_folder, "trained_weights_final.h5")
model_classes = os.path.join(model_folder, "data_classes.txt")

anchors_path = os.path.join(src_path, "keras_yolo3", "model_data", "yolo_anchors.txt")

FLAGS = None

if __name__ == "__main__":
    # Delete all default flags
    parser = argparse.ArgumentParser(argument_default=argparse.SUPPRESS)
    """
    Command line options
    """

    parser.add_argument(
        "--input_path",
        type=str,
        default=image_test_folder,
        help="Path to image/video directory. All subdirectories will be included. Default is "
        + image_test_folder,
    )

    parser.add_argument(
        "--output",
        type=str,
        default=detection_results_folder,
        help="Output path for detection results. Default is "
        + detection_results_folder,
    )

    parser.add_argument(
        "--no_save_img",
        default=False,
        action="store_true",
        help="Only save bounding box coordinates but do not save output images with annotated
      boxes. Default is False.",
    )

    parser.add_argument(
        "--file_types",
        "--names-list",
        nargs="*",
        default=[],
        help="Specify list of file types to include. Default is --file_types .jpg .jpeg .png .mp4",
    )

    parser.add_argument(
        "--yolo_model",
        type=str,
        dest="model_path",
        default=model_weights,
        help="Path to pre-trained weight files. Default is " + model_weights,
    )

    parser.add_argument(
        "--anchors",
        type=str,
        dest="anchors_path",
        default=anchors_path,
        help="Path to YOLO anchors. Default is " + anchors_path,
    )

    parser.add_argument(
        "--classes",
        type=str,
        dest="classes_path",
        default=model_classes,
        help="Path to YOLO class specifications. Default is " + model_classes,
    )

    parser.add_argument(
        "--gpu_num", type=int, default=1, help="Number of GPU to use. Default is 1"
    )

    parser.add_argument(
        "--confidence",
        type=float,
        dest="score",
        default=0.25,
        help="Threshold for YOLO object confidence score to show predictions. Default is 0.25.",
    )

    parser.add_argument(
        "--box_file",
        type=str,
        dest="box",
        default=detection_results_file,
        help="File to save bounding box results to. Default is "
        + detection_results_file,
    )

    parser.add_argument(
        "--postfix",
        type=str,
        dest="postfix",
        default="_disease",
        help='Specify the postfix for images with bounding boxes. Default is "_disease"',
    )

    FLAGS = parser.parse_args()

    save_img = not FLAGS.no_save_img

    file_types = FLAGS.file_types

    if file_types:
        input_paths = GetFileList(FLAGS.input_path, endings=file_types)
    else:
        input_paths = GetFileList(FLAGS.input_path)

    # Split images and videos
```

```
img_endings = (".jpg", ".jpeg", ".png")
vid_endings = (".mp4", ".mpeg", ".mpg", ".avi")

input_image_paths = []
input_video_paths = []
for item in input_paths:
    if item.endswith(img_endings):
        input_image_paths.append(item)
    elif item.endswith(vid_endings):
        input_video_paths.append(item)

output_path = FLAGS.output
if not os.path.exists(output_path):
    os.makedirs(output_path)

# define YOLO detector
yolo = YOLO(
    **{
        "model_path": FLAGS.model_path,
        "anchors_path": FLAGS.anchors_path,
        "classes_path": FLAGS.classes_path,
        "score": FLAGS.score,
        "gpu_num": FLAGS.gpu_num,
        "model_image_size": (416, 416),
    }
)

# Make a dataframe for the prediction outputs
out_df = pd.DataFrame(
    columns=[
        "image",
        "image_path",
        "xmin",
        "ymin",
        "xmax",
        "ymax",
        "label",
        "confidence",
        "x_size",
        "y_size",
    ]
)

# labels to draw on images
class_file = open(FLAGS.classes_path, "r")
input_labels = [line.rstrip("\n") for line in class_file.readlines()]
print("Found {} input labels: {} ...".format(len(input_labels), input_labels))

if input_image_paths:
    print(
        "Found {} input images: {} ...".format(
            len(input_image_paths),
            [os.path.basename(f) for f in input_image_paths[:5]],
        )
    )
    start = timer()
    text_out = ""

    # This is for images
    for i, img_path in enumerate(input_image_paths):
        print(img_path)
        prediction, image,lat,lon= detect_object(
            yolo,
            img_path,
            save_img=save_img,
            save_img_path=FLAGS.output,
            postfix=FLAGS.postfix,
        )
        print(lat,lon)
        y_size, x_size, _ = np.array(image).shape
        for single_prediction in prediction:
            out_df = out_df.append(
                pd.DataFrame(
                    [
                        [
                            os.path.basename(img_path.rstrip("\n")),
                            img_path.rstrip("\n"),
                        ]
                        + single_prediction
                        + [x_size, y_size]
                    ],
                    columns=[
                        "image",
                        "image_path",
                        "xmin",
                        "ymin",
                        "xmax",
                        "ymax",
                        "label",
                        "confidence",
                        "x_size",
                        "y_size",
                    ],
                        )
                    )
                end = timer()
                print(
                    "Processed {} images in {:.1f}sec - {:.1f}FPS".format(
                        len(input_image_paths),
                        end - start,
                        len(input_image_paths) / (end - start),
                    )
                )
                out_df.to_csv(FLAGS.box, index=False)

            # This is for videos
            if input_video_paths:
                print(
                    "Found {} input videos: {} ...".format(
                        len(input_video_paths),
                        [os.path.basename(f) for f in input_video_paths[:5]],
                    )
                )
                start = timer()
                for i, vid_path in enumerate(input_video_paths):
                    output_path = os.path.join(
                        FLAGS.output,
                        os.path.basename(vid_path).replace(".", FLAGS.postfix + "."),
                    )
                    detect_video(yolo, vid_path, output_path=output_path)

                end = timer()
                print(
                    "Processed {} videos in {:.1f}sec".format(
                        len(input_video_paths), end - start
                    )
                )
            # Close the current yolo session
            yolo.close_session()
```

GitHub:

gh repo clone IBM-EPBL/IBM-Project-5510-1658769141

Project Demo Link:

https://drive.google.com/file/d/1-D08VKkBEN4U0HSD3wD0j406GURgmOEV/view?usp=sharing