Image classification through network protocol

COMPUTER NETWORKS CS : 552

TEAM

DHEEKSHITH DEV MANOHAR MEKALA

dm1653

TEJA RANGA REO PALADUGU

tp577

INTRODUCTION

Team Members: 2

We developed a robust Machine Learning Classifier model that transfers images through **Transmission Control Protocol (TCP)** network protocol and classifies those images as either animated or real image by performing various classification operations with relative accuracies for each operation. We programmed three distinct image classifiers where one performs on byte-array and the other two classifiers use Computer Vision (CV).

IMPORTANCE

Distinguishing animated images from real-life photos is critical for many applications, including web browsers and image searches in multimedia documents. The initial technique we implemented is significant if **robustness** and **security** are a primary concern than accuracy because we operated on byte arrays rather than pixels, unlike many fundamental ML models do.

MOTIVATION FOR THE PROJECT

The project's usage of image processing on byte-arrays to classify images is a new notable technological factor. The comparison of efficacies and inefficacies of all three models based on the domain of usage is what makes this project intriguing. If the project is successful it would be deemed useful for domains where speed and security are the primary factors for image classification. This model is a good trade-off for speed vs accuracy factors.

Real-Life Example: If a system's function is to scour through millions and millions of images and classify them on the internet, the model we built would be a better fit as it takes <0.05 seconds to classify an image with a slightly less accuracy.

TECHNOLOGIES USED

We used Transmission Control Protocol (TCP) for image transfer between the client and the server.

Byte-Array Model (with One-Hot Encoding)

Computer Vision Bilateral-Filtering Process Model

Computer Vision Color Counting Model

The entire code was programmed in Python 3.10.

WORKFLOW

- ❖ Part 1: The image is randomly picked from a directory and is converted into byte stream buffer for TCP transfer on the client's side.

- ❖ Part 2: The image sent is read and written into a different directory from the stream buffer on the server's side.

- ❖ Part 3: The written image is subjected to three different classifiers and the final output of whether the sent image is real or animated is displayed.

IMPLEMENTATION LOGIC

- ❖ We used socket programming module for setting up TCP and an IPv4 version server. In Sockets, **socket.AF_INET** specifies IP version 4 and **socket.SOCK_STREAM** specifies TCP.
- ❖ The byte-array image classification is performed on three parameters:
  - ➢ The image is converted into a byte-array and its total length is calculated.
  - ➢ The byte-array is again converted into hexadecimal byte-array and its total length is calculated for further classification.
  - ➢ The image is also converted into a Base-64 string for more efficient classification.
  - ➢ All the observation metrics of these three parameters are One-Hot encoded (specifying exact ranges) for improved accuracy.
- ❖ Computer Vision Bilateral Filtering is a technique that blurs the image intentionally and compares the blurred image against original image using their respective RGB histograms.
- ❖ Color Counting is a technique that counts the total number of colors used from the majority of the image by taking the vertical 512 pixels of the image.

CHALLENGES

1. We learned that the byte-array length is quite dependent on the resolution of the image. So, we tested 83 images of equal resolution for precise comparison and we found that animated images have more byte count than real images.
2. The initial data has many outliers, so we came up with the One-Hot Encoding technique to narrow down the total range of length of bytes effectively ignoring the outliers. With this, we were able to improve the accuracy by a bit.
3. If accuracy is critical for a system then this model wouldn't be a good fit. So in order to improve the accuracy a lot more we implemented the latest and best image processing models that offers proper trade-off between speed and accuracy.
4. For CV bilateral filter, we observed that the level of blur of image had a slight impact on accuracy, but if the blur level crosses certain threshold then it negatively affects the accuracy of the model.
5. Initially, we seldom got corrupted images through TCP. Later, we figured out that it was due to the time-out limit of the TCP socket connection.

WHY TCP?

- ➔ Why tcp over udp for image transfer?

    - ◆ Although User Datagram Protocol (UDP), is quick for sending data, TCP guarantees successful image transfer whereas UDP, once a packet is lost, or corrupted it's really difficult to reform the image again.

    - ◆ Protocols cannot allow for complete transfer of a full image directly if the size is large. They can only send images in chunks. The buffer size we used for image chunks is 2048 bytes. As, TCP is a continuous stream of data, it will efficiently handle the data without any data loss.

    - ◆ The handling of packets in UDP is also very inefficient (*like resending the entire image data for a single lost/corrupted packet*) compared to TCP.

Future Scope

Proposed Model vs Available Models

Our Model

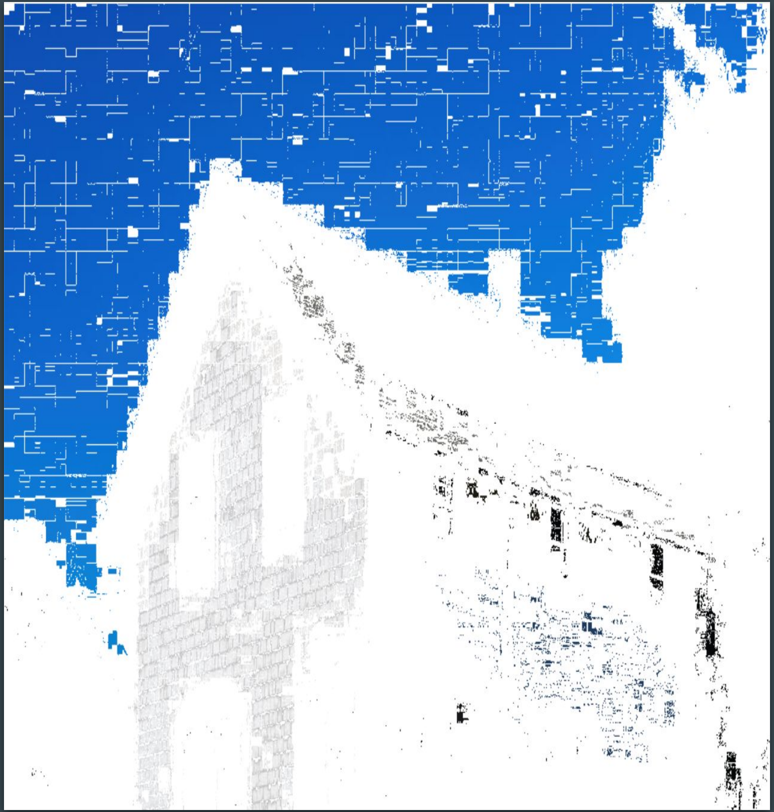
The model we built would serve as a good fundamental starting point for other future ML models;. if tweaked properly, a well developed and trained pattern recognition ML model performing its operations on byte streams rather than intensive image processing can effectively replace the existing image ML models.

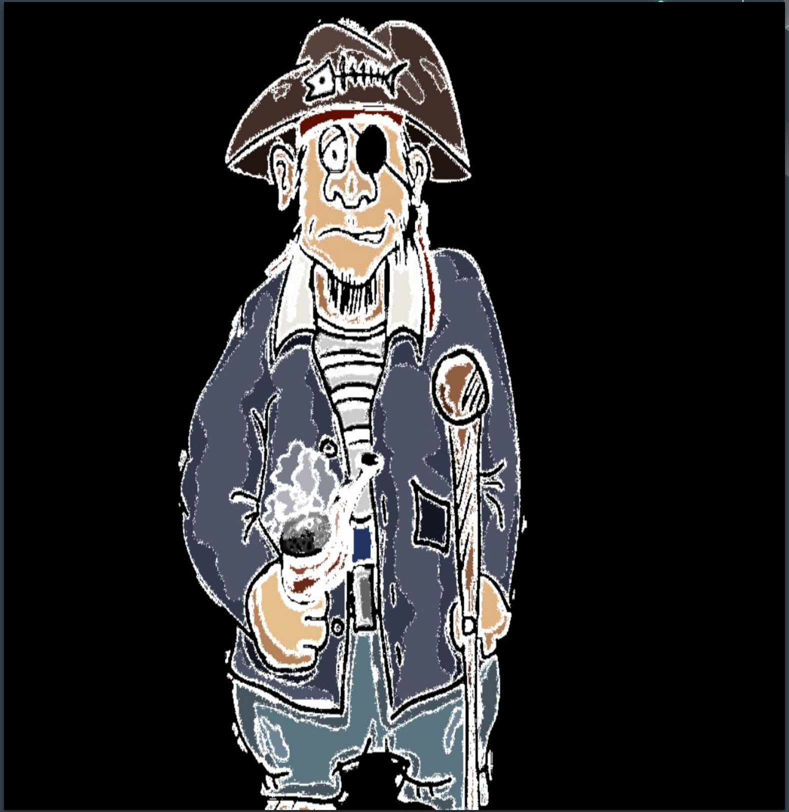
Existing Models

All the existing models till now use tensorflow, keras ML libraries to conduct image classifications. These said models are very slow and computationally intensive when used on bulk data.

RESULTS

Color Counting Algorithm
Real vs Animated Image Preview

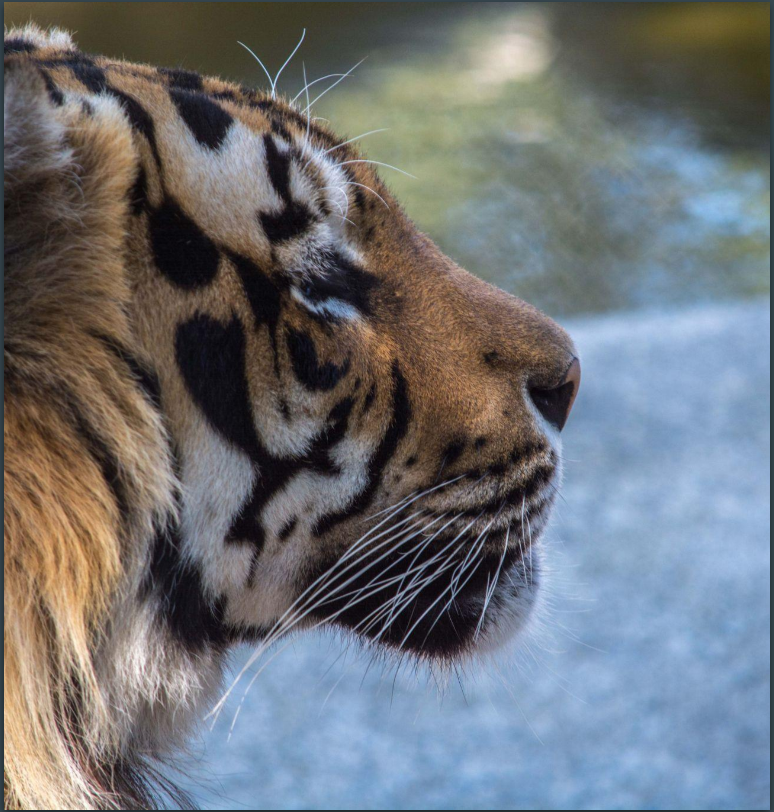


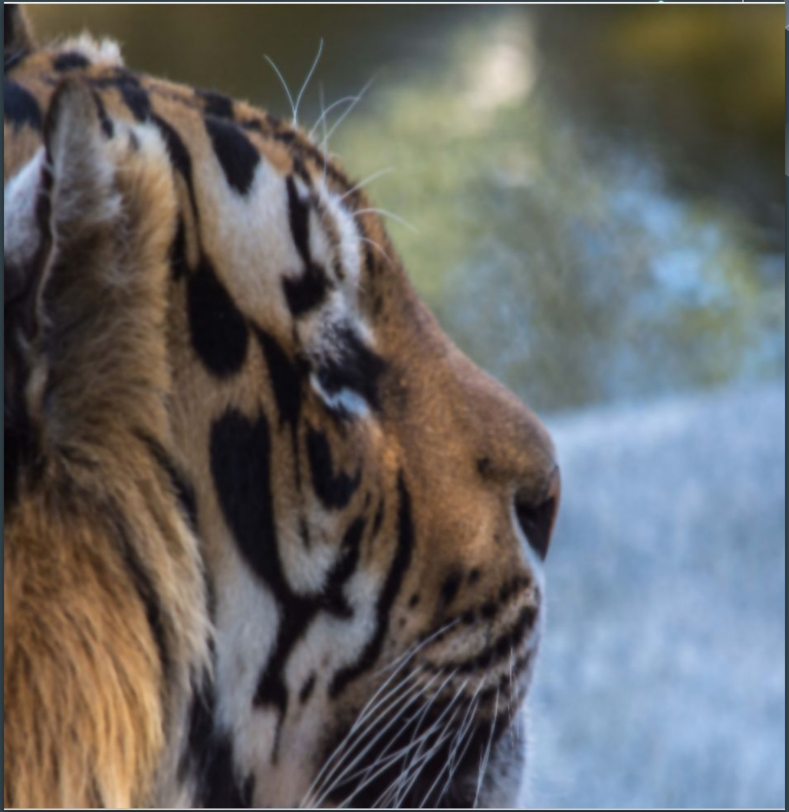


RESULTS

Bilateral Filter

Image Before and After Filter Preview

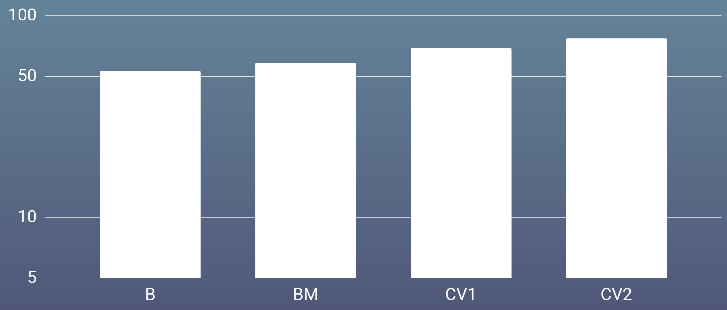




METRICS

TEST DATA: We ran the tests on 83 images of equal resolution for comparison.

Accuracy of Models



B - Byte-Array Model (Base)
BM - Byte-Array Model (with Modifiers)
CV1 - CV Bilateral Filter
CV2 - CV Color Count

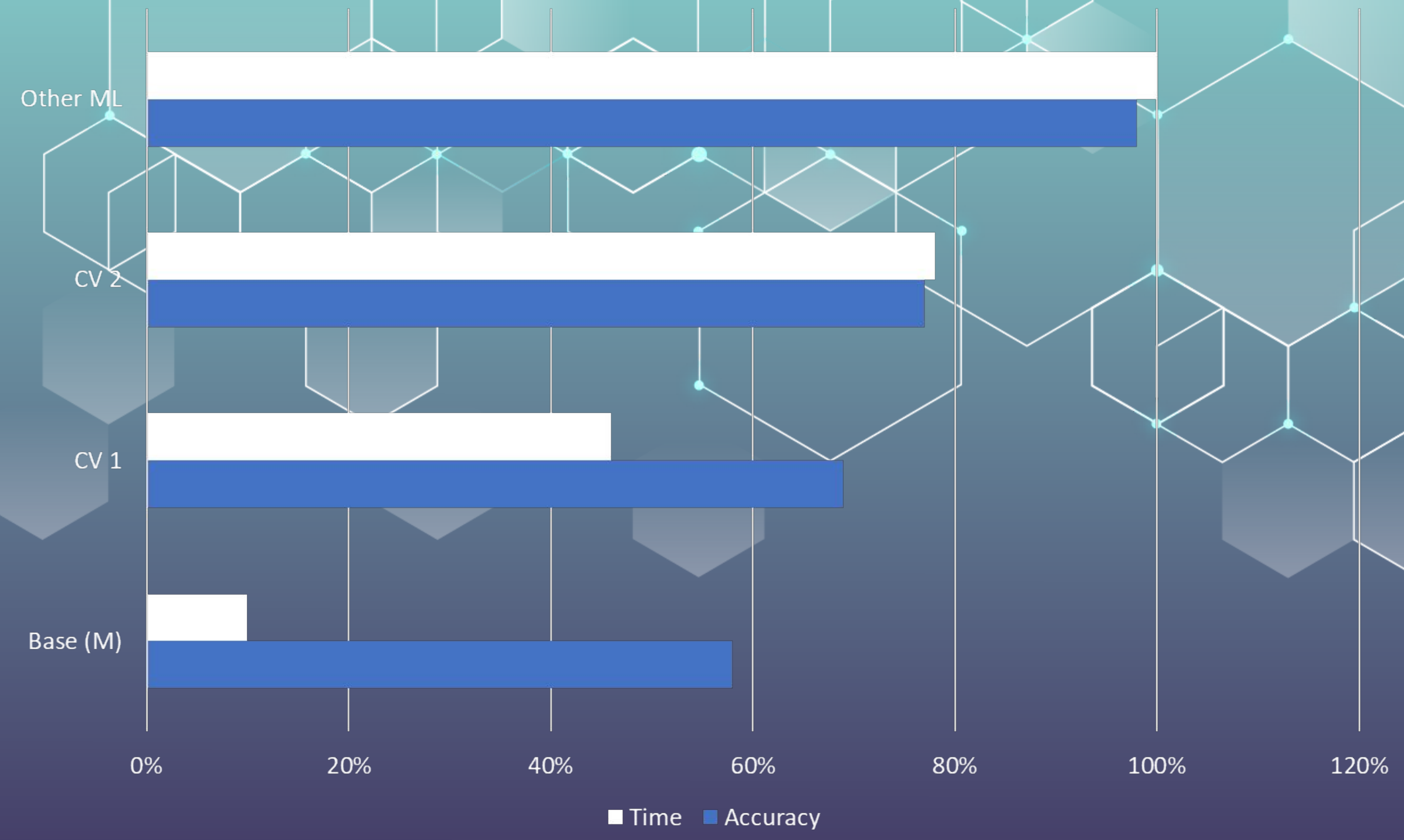
53%
Byte-Array Model (Base)

58%
Byte-Array Model (with Modifiers)

69%
CV Bilateral Filtering

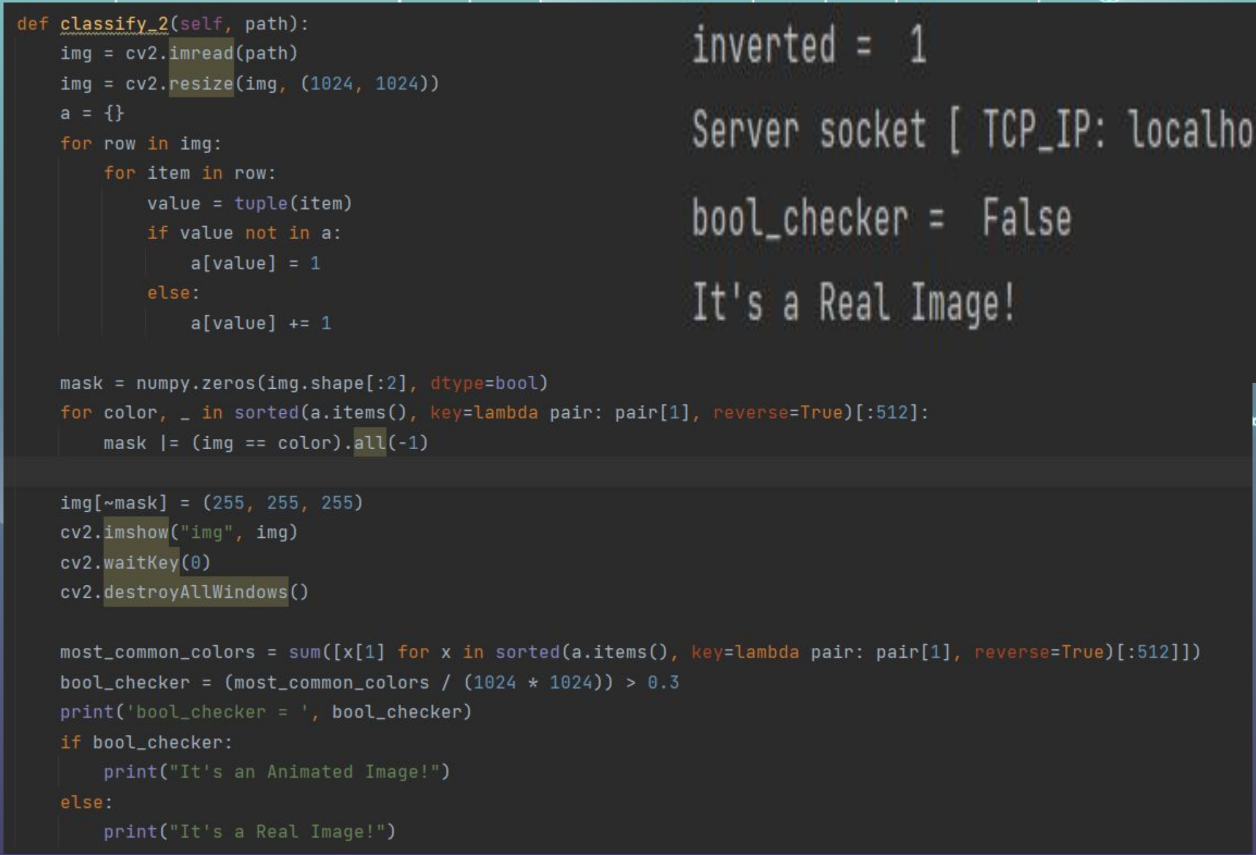
77%
CV Color Count

Models Speed and Accuracy



We can notice how the time it takes to classify images increases as the accuracy increases.

Sample Code & Output

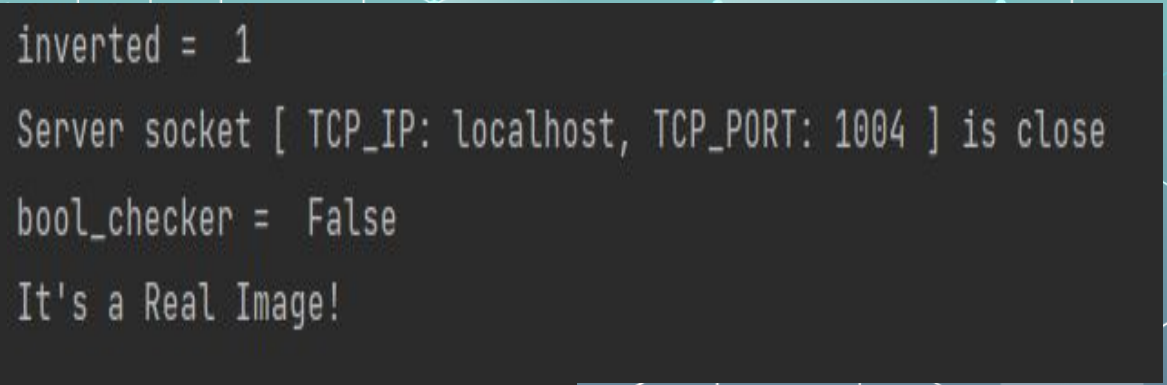


```python
def classify_2(self, path):
    img = cv2.imread(path)
    img = cv2.resize(img, (1024, 1024))
    a = {}
    for row in img:
        for item in row:
            value = tuple(item)
            if value not in a:
                a[value] = 1
            else:
                a[value] += 1

    mask = numpy.zeros(img.shape[:2], dtype=bool)
    for color, _ in sorted(a.items(), key=lambda pair: pair[1], reverse=True)[:512]:
        mask |= (img == color).all(-1)


    img[~mask] = (255, 255, 255)
    cv2.imshow("img", img)
    cv2.waitKey(0)
    cv2.destroyAllWindows()

    most_common_colors = sum([x[1] for x in sorted(a.items(), key=lambda pair: pair[1], reverse=True)[:512]])
    bool_checker = (most_common_colors / (1024 * 1024)) > 0.3
    print('bool_checker = ', bool_checker)
    if bool_checker:
        print("It's an Animated Image!")
    else:
        print("It's a Real Image!")
```



```
inverted =  1

Server socket [ TCP_IP: localhost, TCP_PORT: 1004 ] is close

bool_checker =  False

It's a Real Image!
```

SUMMARY

We think the model we came up with might provide a fundamental skeleton for future ML Image Classification models that do not necessarily use intensive image processing techniques but rather use other ways to classify images like the new methodology we implemented. The default TCP for image transfer can also be replaced with other protocols for an even faster file transfer rate in the future. For the ML approach - We observed a trade-off between speed and accuracy. So, we suggest that exploring more pattern recognition ML models is wise than image processing models, which would increase the overall accuracy. The world of AI is still uncharted, so there could be many possibilities of an even better model out there for image classification, and we tried to explore that a bit with our project.

THANKS