Report: Sky Pixel Identification in Images using Traditional Computer Vision Techniques

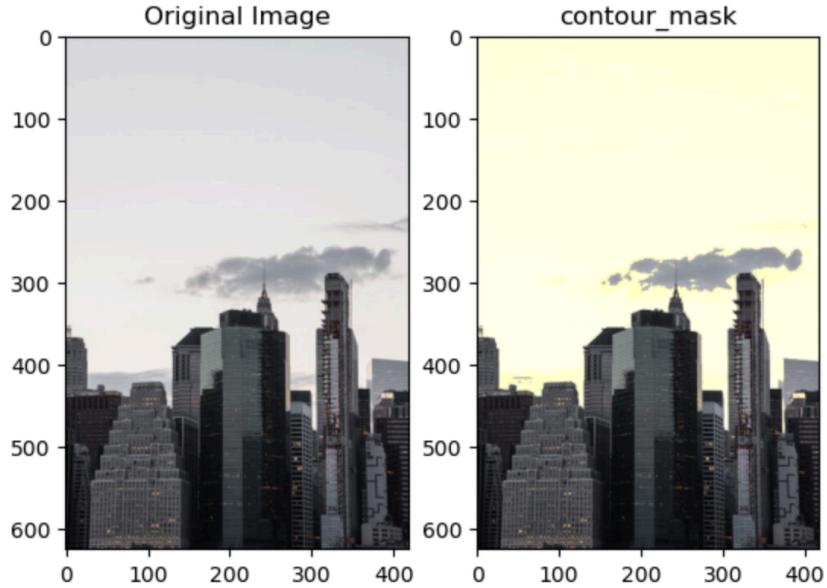
Student Name: Meng(Maxine) Tian

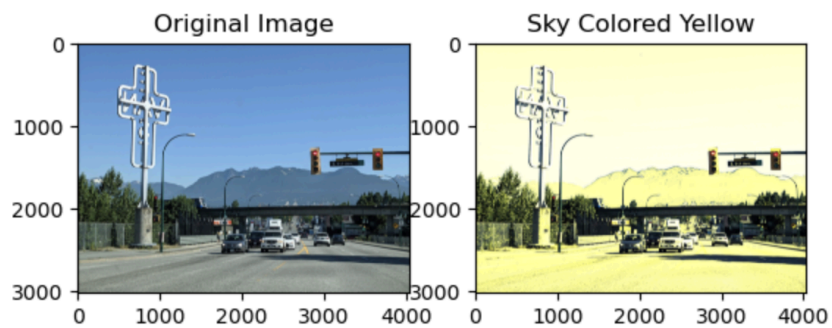
Introduction:
In this project, I embarked on a journey to understand and implement traditional computer vision techniques, specifically for the task of sky detection in images. My primary goal was to reliably identify the sky region and color it yellow, which involved a series of methodical experimentation with color spaces and thresholding techniques.

Approach and Documentation:
I began my experimentation with a straightforward approach, utilizing grayscale conversion of images with the intent of isolating lighter areas that typically represent the sky. The initial code was simple:

```python
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
_, sky_mask = cv2.threshold(gray, 100, 255, cv2.THRESH_BINARY)
```

This method, while conceptually sound, proved to be less effective in practice. The main issue was its inability to distinguish between the sky and other light regions, such as cloud details and the deeper parts of the sky, which are not uniformly bright.

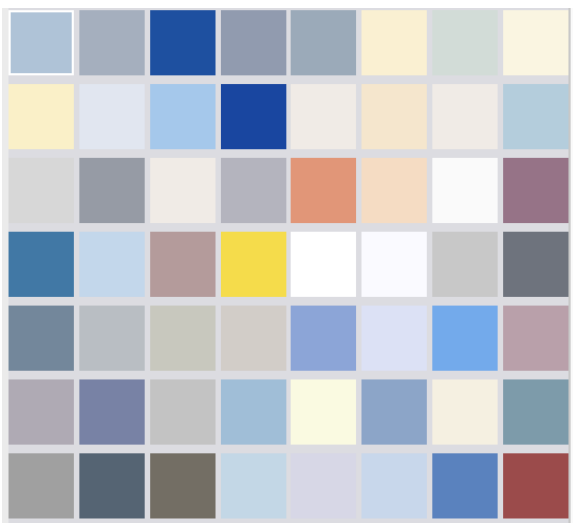


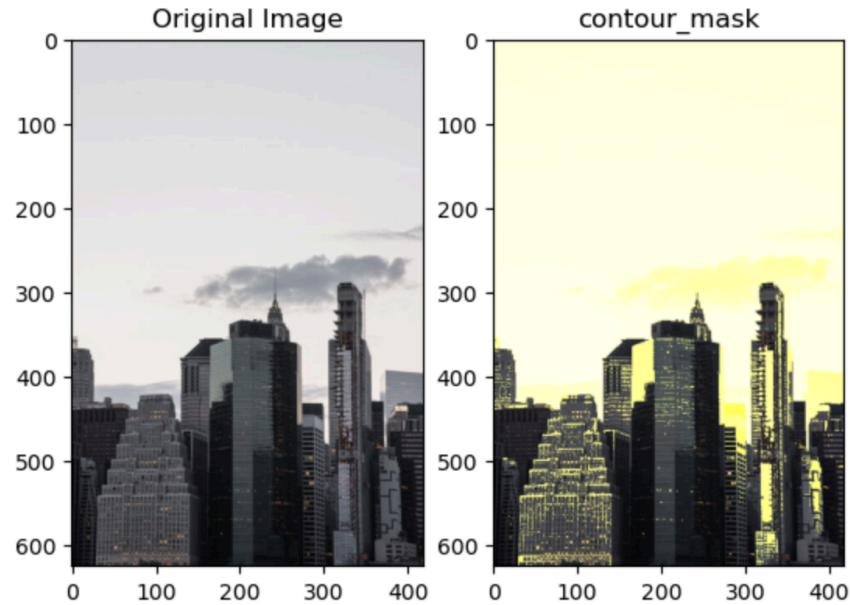


To overcome this, I pivoted to a more sophisticated method involving color analysis.
By selecting pixels from a sample of 20 images that represented the sky, I established a color range in the RGB space:

```python
lower_blue = np.array([0, 80, 120])
upper_blue = np.array([255, 255, 255])
```

These are the pixels I selected in the sky image.

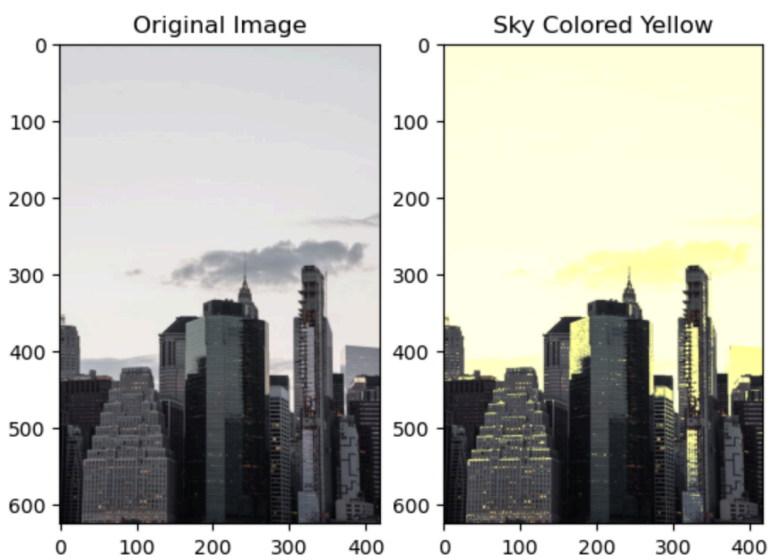




But this initial range was too broad, inadvertently including non-sky regions in the mask. Through iterative refinement and careful adjustments, I fine-tuned the range to:

```
lower_blue = np.array([80, 80, 120])
upper_blue = np.array([255, 255, 255])
```

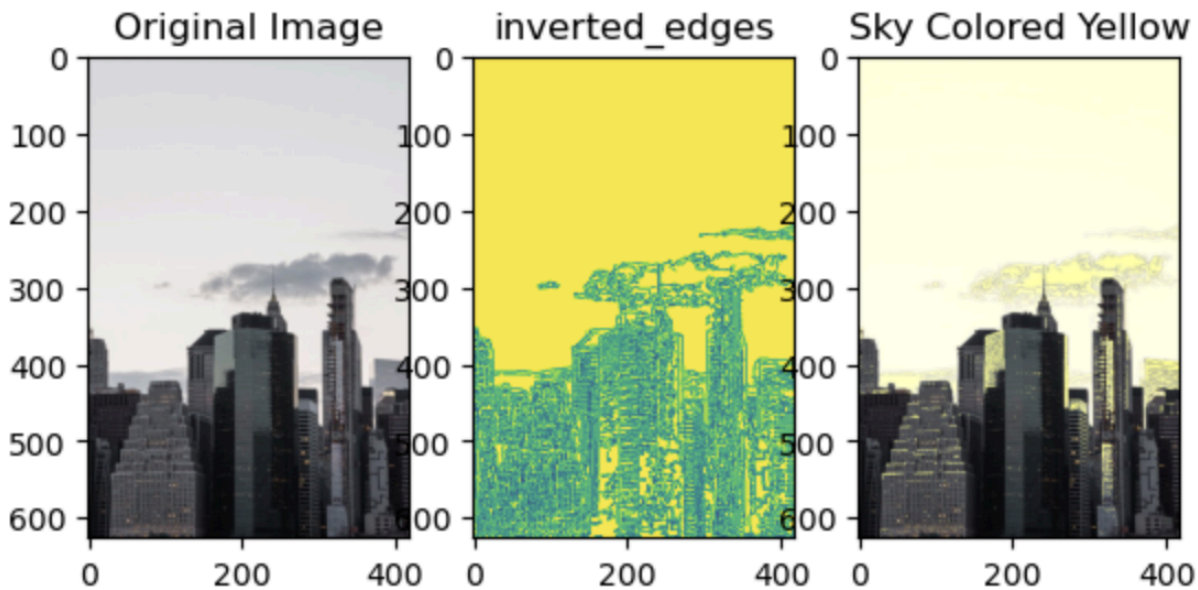
This new range significantly improved the detection of overcast skies, a common and challenging scenario in sky detection tasks.



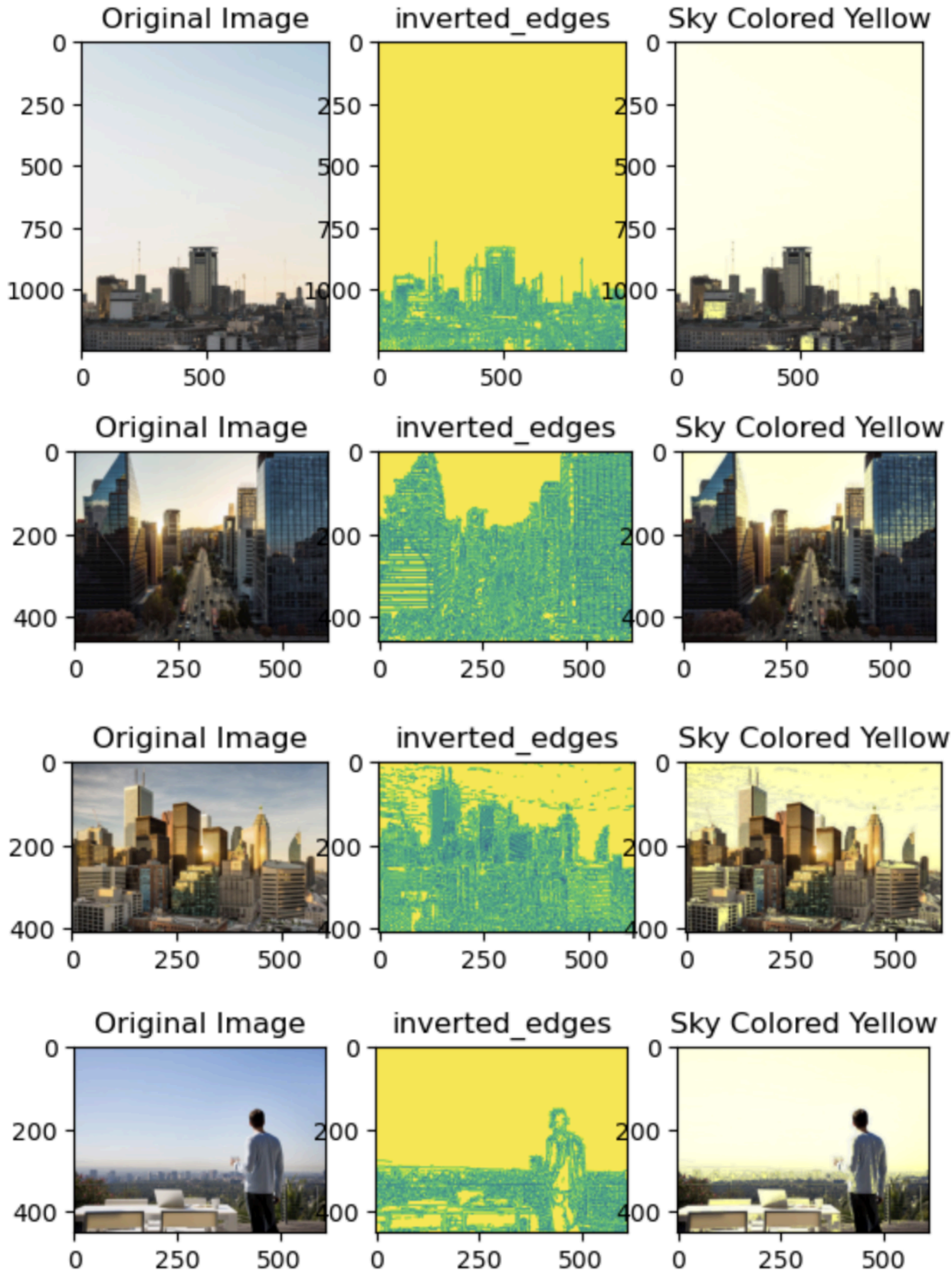
After refining the range for detecting the sky, the mask still includes some non-sky areas. However, the accuracy has improved significantly compared to before.

However, the detection method often misidentified buildings as part of the sky. We know that the key difference between the sky and buildings lies in their boundaries; the sky is characterized by its smoothness. Therefore, edge detection can be employed to distinguish these features. By implementing Canny edge detection, we can effectively remove edges from the identified sky regions. This addition helps in excluding building outlines, enhancing the accuracy of sky detection.

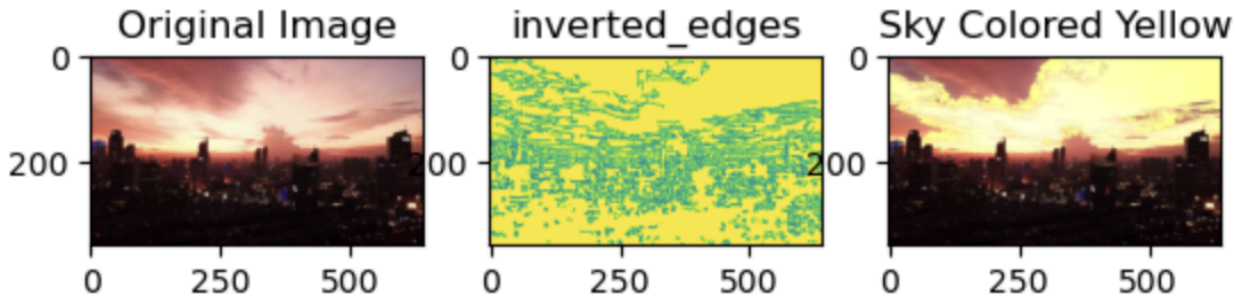
```
# Find the edge and remove the edge from the sky
edges = cv2.Canny(image_rgb, 10, 50)
inverted_edges = cv2.bitwise_not(edges)
combined_mask = cv2.bitwise_and(sky_mask, inverted_edges)
```



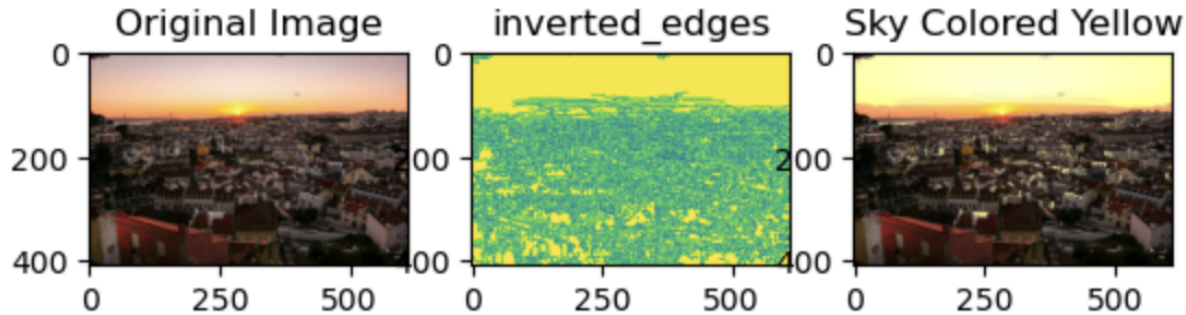
However, a challenge arises with this approach: the clouds are also removed from the sky in the process. Consequently, we are only able to detect parts of the sky that are devoid of clouds. The following is the performance of the image on a sunny day, which is good.

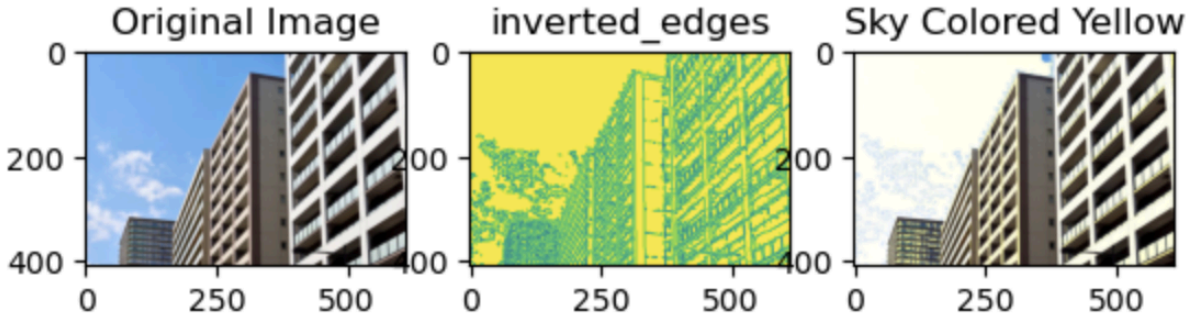

Original Image
inverted_edges
Sky Colored Yellow
0
250
500
750
1000
0
500
Original Image
inverted_edges
Sky Colored Yellow
0
200
400
0
250
500
Original Image
inverted_edges
Sky Colored Yellow
0
200
400
0
250
500
Original Image
inverted_edges
Sky Colored Yellow
0
200
400
0
250
500

However, this adjustment was still insufficient for accurately capturing the red and orange tones typical in sunset skies.

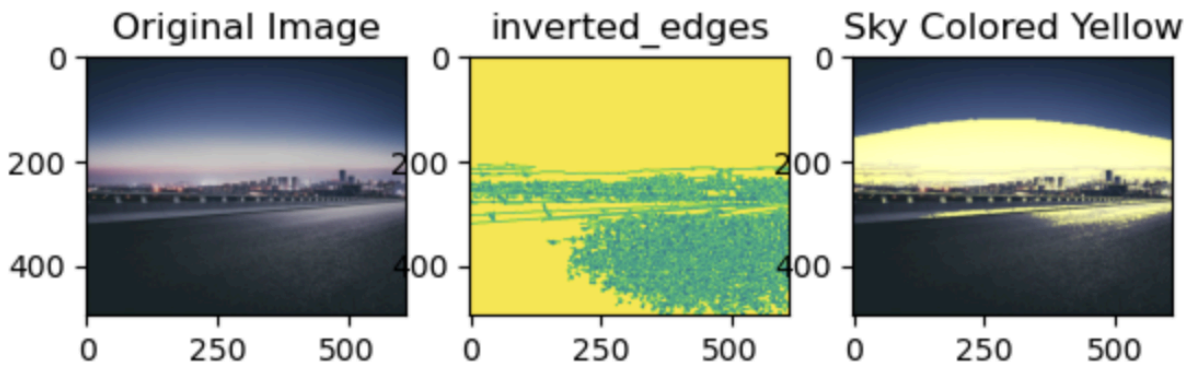


But it performs well in pictures without red cumulus clouds.





As to the partially dark sky pic, the ability to recognize the sky is very poor.



Challenges and Limitations:
Throughout the process, several challenges emerged. One notable difficulty was differentiating between the sky and reflective surfaces on buildings, which share a similar color and brightness. This limitation became apparent when parts of buildings with reflective glass were mistakenly identified as the sky.

Another challenge was the variability of the sky's appearance. Factors such as weather conditions, time of day, and camera exposure can dramatically alter the color and brightness of the sky, requiring a dynamic or adaptable threshold that can cater to a wide range of scenarios.

Traditional computer vision techniques typically rely on color differences to segment images. However, in scenarios where the sky and ground share similar hues, these methods struggle. Realizing the need for a more sophisticated approach, I explored the idea of using edge detection as potential discriminators. The rationale was that even if the colors are similar, the texture of the sky (which is generally smoother) and the ground (which might have more varied textures) could provide a clue.

Reflection:
Reflecting on this experience, I realized the importance of choosing the right color space for segmentation tasks. While the grayscale method is computationally simpler, it lacks the nuance needed for complex scenes. RGB color space, on the other hand, offers more discriminative power but requires careful tuning to avoid overgeneralization.

The project underscored the inherent trade-offs in traditional computer vision techniques. It was a balancing act between specificity and generalization, precision, and robustness. Despite the challenges, the journey was highly educational, providing me with invaluable insights into the mechanics of image processing and the subtleties of visual perception.

In conclusion, this project has deepened my understanding of traditional computer vision techniques and their application to real-world problems. It has also opened my eyes to the limitless possibilities that lie ahead with the advent of more advanced machine learning algorithms.