Future Medical Data Analysis through Data Cleaning and Image Processing Report

Part 2: Image Processing

1. Introduction:

This report describes the implementation of various image processing techniques to improve the quality of fundus eye images for an ophthalmic clinic's autonomous decision support system. The clinic's system uses a pre-trained classifier to evaluate input images and provide a binary label indicating whether the eye is healthy or not healthy. However, the test images suffer from several issues due to a broken capture device. This report outlines the implemented techniques to address these issues and enhance image quality.

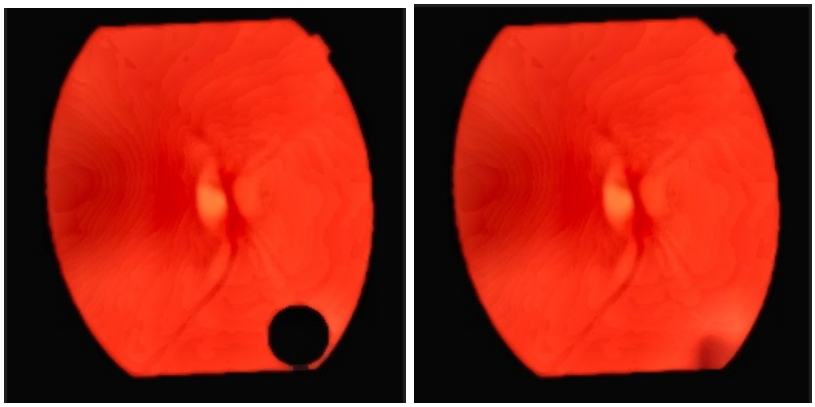
2. Image Processing Techniques:

The following image processing techniques were implemented to address the issues identified in the test images:

2.1. Inpainting Missing Regions:

A circular portion of the image at the bottom right of all images is missing. The inpaint_missing_region function was implemented to fill in the missing region using the inpainting method provided by OpenCV. Towards this goal, another python script was written called ‘cord.py’ that opened the image in a file and we could draw a circle of varying radiuses to completely capture the missing circular region in the original test_images. We managed to find the circle center at (188, 211) with a radius of 22 to fully encompass the missing region.

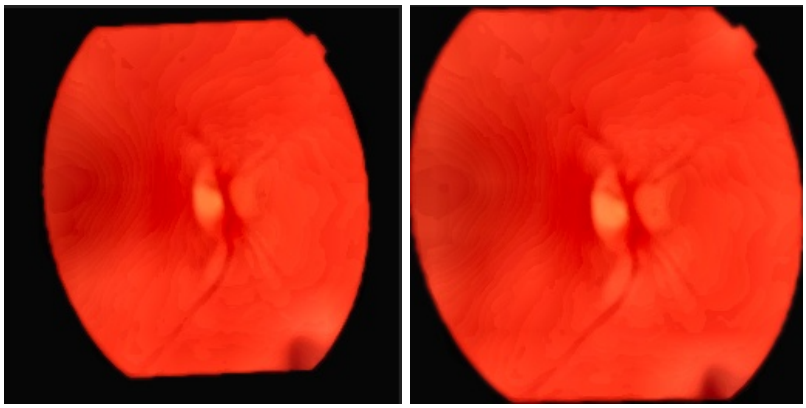
This achieved the following transformation:



2.2. Projective Transformation for Warping:

The test images are distorted, and the unwarp_image function was implemented to correct these distortions using projective transformations. The function calculates the transformation matrix based on source and destination points, and then applies the transformation using OpenCV's warpPerspective function. To find the source points of the corners, We used the ‘cord.py’ script again to manually check the corners and correctly warp the image around the corners. This resulted in a very acceptable outcome.

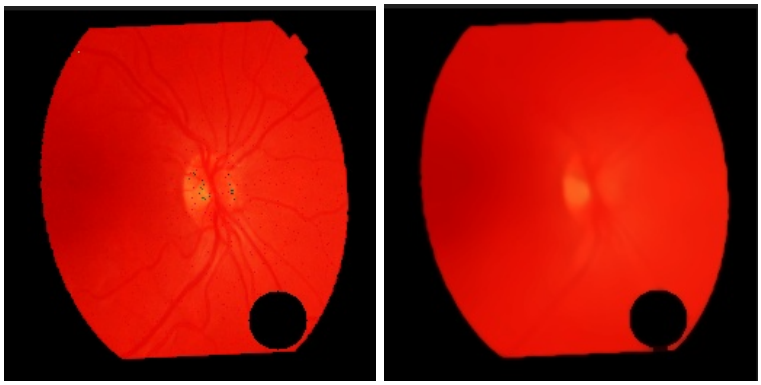
This achieved the following transformation:



2.3. Denoising

The images contain significant amounts of Gaussian and salt/pepper noise. The denoise_image function was implemented to address these issues using the Non-local Means denoising algorithm for Gaussian noise and a median filter for salt/pepper noise.

This achieved the following transformation on the original test image:



2.4. Sharpening

The sharpen_image function was implemented to improve the sharpness of the images. The function applies a Gaussian blur to the input image, calculates the difference between the original and blurred image, and adds the difference back to the original image to enhance sharpness.

2.5. Enhancing Contrast and Brightness

The contrast and brightness of the images are not properly adjusted. The enhance_contrast_and_brightness function was implemented to improve the contrast and brightness using Contrast Limited Adaptive Histogram Equalization (CLAHE). We found the optimal parameters for CLAHE as clipLimit=3.0, tileGridSize=(8, 8).

3. Processing Pipeline

The implemented image processing techniques were applied in the following order:

1. Inpaint missing regions.
2. Correct image distortion using projective transformation.
3. Denoise the image.
4. Sharpen the image.
5. Enhance the contrast and brightness.

This order was determined through trial and testing and basic intuition. As can be seen we first fix the missing regions, then correct distortion and move on to denoising the image itself.

5. Iterative Improvement and Achieving Higher Accuracy

Throughout the development of the image processing pipeline, multiple iterations were performed to improve the accuracy of the pre-trained classifier. Each addition or modification to the processing techniques was carefully assessed to ensure that it contributed positively to the overall accuracy of the classifier.

With each implemented technique, the performance of the classifier was evaluated to determine if the modifications improved the results. By iteratively adjusting the parameters and combining different techniques, the classifier's accuracy increased gradually. For example, adjusting the clip limit and tile grid size in the enhance_contrast_and_brightness function allowed us to find

the optimal settings. Additionally, the order in which the techniques were applied in the pipeline was fine-tuned to achieve the best results. This iterative process involved multiple rounds of experimentation and evaluation, ultimately leading to a significant improvement in the classifier's performance.

The final pipeline, combining iterative improvements and multiple image processing techniques, increased the pre-trained classifier's accuracy from 50% to 65%, showcasing the effectiveness of these enhancements on test image quality.

6. Conclusion

In conclusion, the development of an effective image processing pipeline for the ophthalmic clinic's autonomous decision support system was an iterative and experimental process. By continuously evaluating the performance of the pre-trained classifier with each modification, we were able to fine-tune the techniques and parameters to achieve a higher accuracy. This iterative approach, coupled with the combination of various image processing techniques, led to a significant improvement in the classifier's performance, enabling more accurate diagnoses for the ophthalmic clinic's patients.