**Problem Statement**

Given a fired cartridge case image, we have to automatically mask the image into various portions, including breech-face impression, aperture shear, firing pin impression, firing pin drag, and the direction of the firing pin drag by creating an algorithm.

**Programming Environment:** Python 3.11.0, OpenCV-python 4.9.0, TensorFlow==2.15.0, matplotlib==3.8.2, NumPy==1.23.5, pillow==9.5.0, Streamlit==1.30.0

**To Run the Code**

1. Install the packages from the requirements.txt file. From the command prompt run

**python -m pip install --user -r requirements.txt**

command from the same folder where the requirements.txt file is located.

1. After installing the packages, you can use IDEs, such as Visual Studio Code, Jupyter Notebook or JupyterLab to run the cells in the cartridge-segmentation.ipynb notebook.
2. You can also use the Streamlit web application to test the masking algorithm by selecting cartridge images from the interface. To run the app, open the command prompt, change the directory to the project folder, and run the batch script file, start.cmd by typing in **start.cmd** command.
3. The script will check whether all the packages are installed from the requirements.txt file and install the packages if required. Then it will run the command to start the Streamlit web app on the local server. To close the app, press **cltr** and **c** buttons simultaneously from the command prompt.

**My Approach**

I have used a deep learning model to segment/mask the cartridge images. As only one example was provided, I created more training images by segmenting collected cartridge images. Later, I applied data augmentation to increase the training set samples and trained the model. I applied some post-processing methods to the generated images to reduce noise.

1. **Dataset and Preprocessing**

The cartridge images were collected from [1]. The train set contained 20 cartridge images and the test set had 10 images fired from a range of firearms, including Ruger SR9, Taurus 24/7 G2 pistol, SCCY CPX-2, Smith & Wesson SD9VE, and Springfield XD9. All the cartridges were 9 mm cartridge cases.

Target Image

Input Image

Figure 1: Some examples from the training set. On the left, we have the input images. The corresponding masked images are on the right. The first cartridge was fired from Ruger SR9, and the second one was from SCCY CPX-2.

I manually masked the collected images using the Paint 3D application. All the training images can be found in the train-images folder. The images were first loaded from their directory, resized to (256x256x3) to match the input layer of the model, and then normalized to keep the pixel values in between the range 0 and 1 before passing them into the model.

1. **Data Augmentation**

As there were only 20 training images, I performed data augmentation to increase the training images. I performed seven augmentation techniques, such as rotation, shifting images horizontally and vertically, random shearing, zooming, and flipping images both horizontally and vertically. For one input image, there were seven augmented images. As a result, there was a total 160 images for training the model.



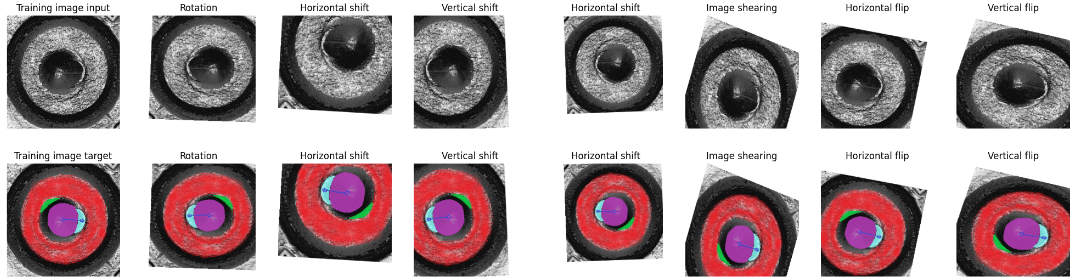


Figure 2: Some examples of image augmentation from the training set.

1. **Model Architecture and Parameter Tuning**

I used the UNet model [2] to mask a cartridge case image. The UNet architecture has two parts, one is the down-sampling portion or encoder, and the second one is the up-sampling portion or the decoder. In the encoder part, the resolution of the image gets reduced by half at each stage, and in the decoder, the resolution increases. In the original UNet architecture, the shape of the input layer is 572 x 572. Then there are two 3x3 convolutional layers each followed by two activation layers.

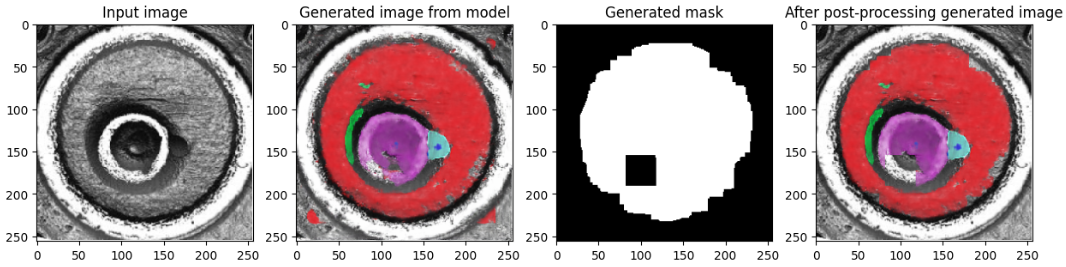


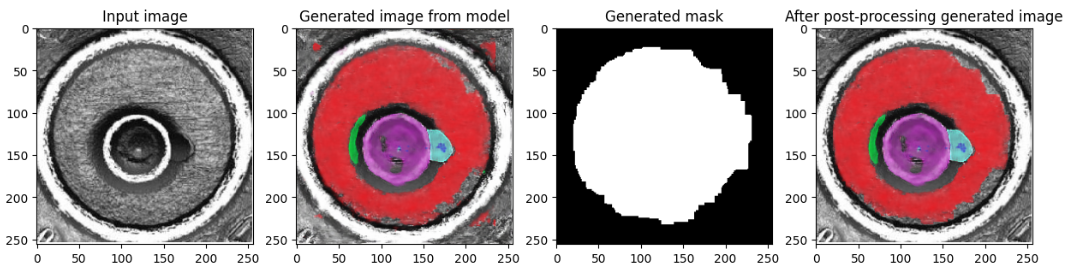
Figure 3: UNet model architecture.

I trained the model for 1000 epochs with a batch size of one. I used the Adam optimization function with a learning rate of 0.0001 and the mean squared error (MSE) loss function to train the model. The trained model was used at the inference stage to test the images from the test set. The test images and the generated segmentation masks can be found in the test-samples folder. I used all 9 mm cartridges to test the model. I used a total of 10 images to test the trained model.

1. **Post-processing the Generated Images to Reduce Noise**

There were noises in the generated images from the model. I tried to remove these noises by first creating a binary mask by performing dilation and erosion morphological operations. I used the binary mask with the input image and the generated image to create a new image with less noise.





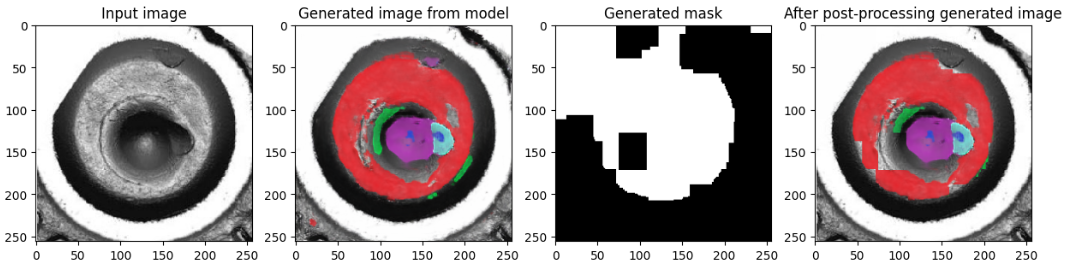


Figure 4: From the left, we have the input test image, the generated image from the model, a binary mask from the input image and the generated image, and the image with less noise after applying the binary mask.

1. **Limitations and Potential Solutions**

The model was introduced to a small variation because there were only 20 images to train the model before augmentation. Also, there was an issue with the segmentation quality of the cartridge images in the training set. I used Paint 3D software to create the segmentation masks to train the model. If the model was trained by images segmented by subject matter experts then the segmentation quality of the input images would have been better, resulting in better training of the model. Because of these issues, the generated images from the model were not segmented properly, and the direction of the firing pin drag was not properly visible. The model can be improved by adding more training images segmented by specialists.

1. **Deploying the Model**

After training, I saved the model and deployed it on my local server. I created a Streamlit application where the user can insert a cartridge image, and the app will segment the image using the saved model and display it back to the user.

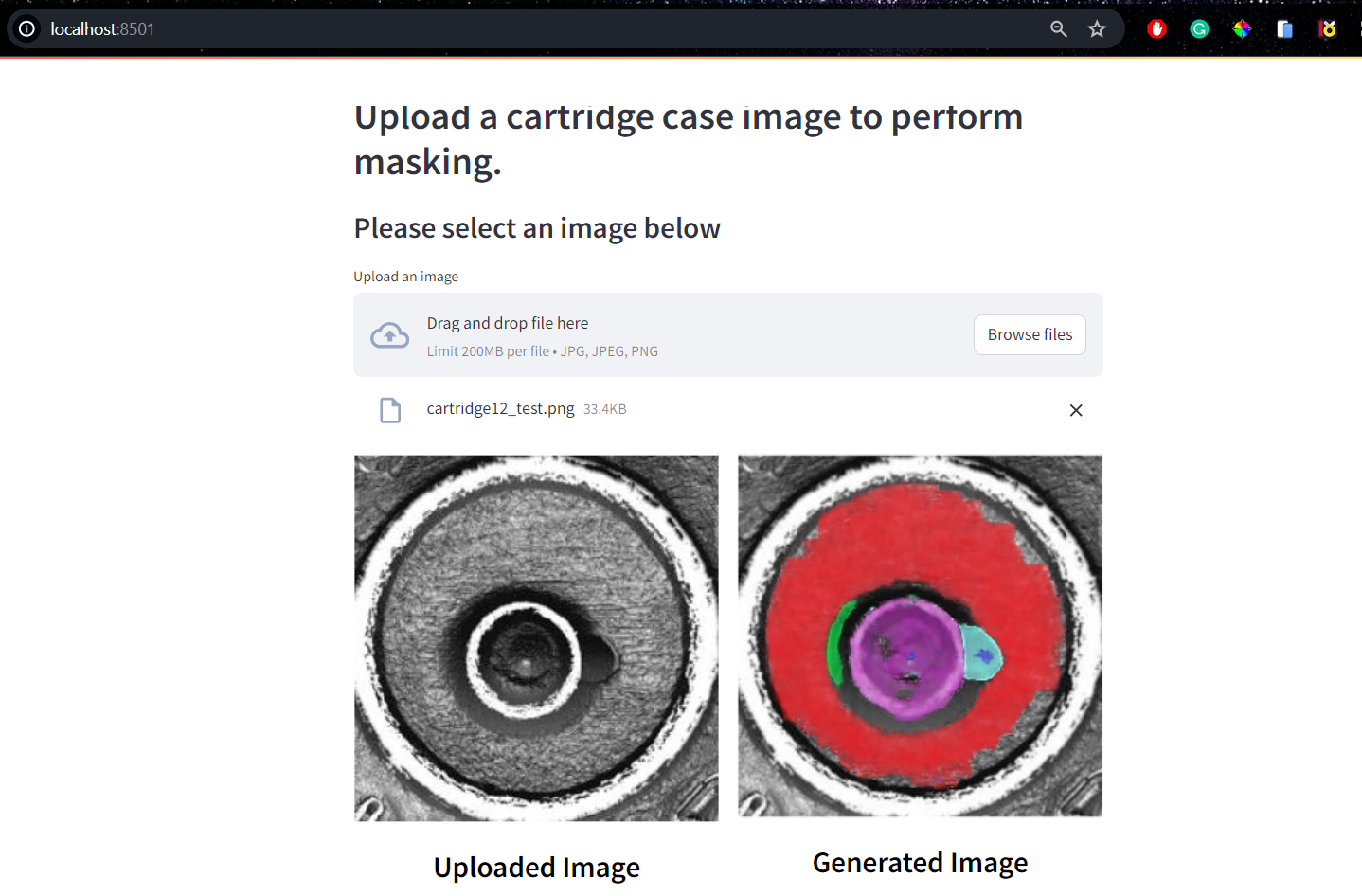


Figure 5: The user-interface of the Streamlit app using the deployed UNet model to segment cartridge case images.

**References**

[1] Law, Eric Freeman. "Evaluating the Accuracy of Firearm Examiner Conclusions using Cartridge Case Reproductions." (2020).

[2] Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18.* Springer International Publishing, 2015.

**Pushing all the Project Files to GitHub and Downloading the Model File**

The code, data, and the model have been pushed to the GitHub repository:

<https://github.com/ShawkhIbneRashid/cartridge-image-masking>. You can git clone all the files from this source. I could not add the model file in the saved-model folder because of size issue. Please download it from the GitHub repository by running the command

**git clone** <https://github.com/ShawkhIbneRashid/cartridge-image-masking>