BME Design-Spring 2026 - Madison Michels
Complete Notebook

PDF Version generated by

Presley Hansen

on

Apr 29, 2026 @01:18 PM CDT

Table of Contents

Team contact Information

John Puccinelli - Dec 19, 2013, 11:28 AM CST

Last Name	First Name	Role	E-mail	Phone	Office Room/Building
		Advisor			
		Client			
		Leader			
		Communicator			
		BSAC			
		BWIG			
		BPAG			

Project description

John Puccinelli - Aug 14, 2013, 12:01 PM CDT

Course Number:

Project Name:

Short Name:

Project description/problem statement:

About the client:

2/10/2026 - Expenses

SADIE ROWE - Feb 10, 2026, 4:28 PM CST

Title: Expenses

Date: 2/10/2026

Content by: Sadie Rowe

Present: N/A

Goals: Document project expenses

Content:

	Description	Manufacturer	Mft Pt#	Vendor	Vendor Cat#	Date	QTY	Cost Each	Total	Link
Amazon										
Raspberry Pi Kit	Contains: Raspberry Pi 5 8GB, 27W power supply, active cooler, 64 GB SD card, card reader, 4K Mico HD out cables, and case	Vemico	B0D2WYFS23			2/8/2026	1	$173.99	$173.99	
Arducam IMX477 Pi HQ Camera	HQ Camera + CS 6mm lens	Arducam	B024002			2/8/2026	1	$67.99	$67.99	
								TOTAL:	**$241.98**	

Conclusions/action items: N/A

1/28/2025 - Raspberry Pi Camera Module Comparison

Madison Michels - Jan 28, 2026, 12:56 PM CST

Title: Raspberry Pi Camera Module Comparison

Date: 1/28/2026

Content by: Maddie

Present: Maddie

Goals: The goal of this entry is to learn about Raspberry Pi camera module options and characteristics.

Content:

- Module 3 Series
  - 12MP Sony sensor
  - Autofocus
  - HDR
  - 1080p @ 50fps video
  - Best for general use, computer vision, surveillance
- Module 2
  - 8MP Sony
  - 62.2 degree field of view
  - Fixed focus
  - 1080p @ 30 fps
  - $20
  - Best for budget projects, basic imaging
- High Quality (HQ) Camera
  - 12.3 MP Sony sensor
  - $50
  - Best for professional photography, low light, creative projects
- AI Camera
  - 12MP Sony sensor
  - On-board AI
  - Preloaded MobileNet-SSD
  - $70
  - Best for Edge AI/ML

Citation

“Raspberry Pi Camera Module Comparison: Complete 2025 Guide,” ThinkRobotics.com. Accessed: Jan. 28, 2026. [Online]. Available: https://thinkrobotics.com/blogs/learn/raspberry-pi-camera-module-comparison-complete-2025-guide

Conclusions/action items:

In conclusion, the Module 3 series camera and AI camera options are the most promising for our design because they have implemented AI capabilities or have good CV balances.

1/28/2025 - How to Use a Raspberry Pi Camera Module 3

Madison Michels - Jan 28, 2026, 1:23 PM CST

Title: How to Use a Raspberry Pi Caera Module 3

Date: 1/28/2026

Content by: Maddie

Present: Maddie

Goals: The goal of this entry is to learn about Raspberry Pi camera module options and characteristics.

Content:

- Key Specifications
  - Sensor: Sony IMX708
  - Resolution: 12 megapixels
  - Autofocus
  - FoV
    - 66 degrees horizontal
  - 1080p and fps
  - 3.3V power supply
  - Dimensions: 25x24x11.5 mm
  - 3g weight
- Pin Configurations

Pin Number	Pin Name	Description
1	GND	Ground
2	3.3V Power	Power supply for the camera module
3	I2C SDA	I2C data line for camera communication
4	I2C SCL	I2C clock line for camera communication
5	CSI Data Lane 0+	Positive differential data signal
6	CSI Data Lane 0-	Negative differential data signal
7	CSI Clock Lane+	Positive differential clock signal
8	CSI Clock Lane-	Negative differential clock signal
9	CSI Data Lane 1+	Positive differential data signal
10	CSI Data Lane 1-	Negative differential data signal

- Usage Instructions
  - Connecting the camera module
    - Power off raspberry Pi
    - Locate the CSI port
    - Insert the Ribbon cable
    - Power on the raspberry pi
  - Enabling the camera
    - Open a terminal on the Pi
    - Run: sudo raspi-config
    - Navigate to Interface Options > Camera and enable the camera
    - Reboot the Pi to apply the changes: sudo reboot
  - Capturing images and videos
    - Control using the libcamera tools
  - Capturing images

        - libcamera-still -o image.jpg
    - Recording a video
        - libcamera-vid -o video.h264 -t 10000
    - To control via Python
        - from picamera2 import Picamera2
        - import time
    - Initialize the camera: picam2 = Picamera2()
    - Configure the camera for preview: picam2.configure(picam2.preview_configuration())
    - Capture an image after 5 seconds: time.sleep(5) picam2.captu_file("image.jpg") print ("Image captured and saved as 'image.jpg'")
    - Stop the camera: picam2.stop() print("Camera stopped")

Citation

C. Design, “Cirkit Designer Tutorials.” Accessed: Jan. 28, 2026. [Online]. Available: https://docs.cirkitdesigner.com

Conclusions/action items:

In conclusion, if the group chooses to pursue the Module 3 camera, we can utilize this code to run our device.

1/28/2025 - AI Camera Raspberry Pi Documentation

Madison Michels - Jan 28, 2026, 11:28 PM CST

Title: AI Camera Raspberry Pi Documentation

Date: 1/28/2026

Content by: Maddie

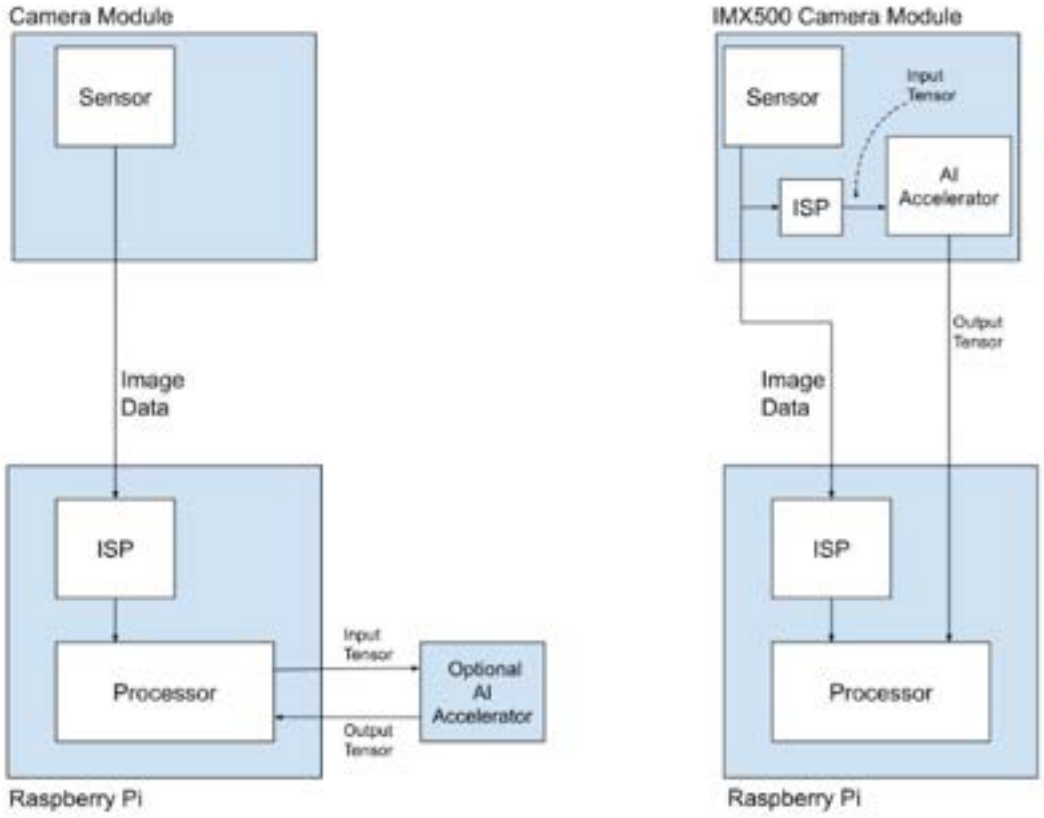
Present: Maddie

Goals: The goal of this entry is to learn about the details and steps needed to run a Raspberry Pi AI camera.

Content:

- Sony IMX500 Sensor
- Camera software stack for easy ML application
- Attach you raspberry pi 5 board to the camera
    - Raspberry Pi 4 Model B or Raspberry Pi 5 board
- Run the following code:
    - sudo apt update && sudo apt full-upgrade
    - sudo apt install imx500-all
    - sudo reboot
- MobileNet SSD NN performs object detection and provides bounding boxes and confidence values for the identifications

Raspberry Pi AI Camera System



- The left side is the traditional AI camera system in which the camera delivers images to the Pi, the Pi processes the images and then performs AI inference.

- The right side demonstrates the IMX500 system that contains a small image signal processor (ISP) which turns raw camera image data into an output tensor.
    - Tensor sent to the AI accelerator within the camera, which produces output tensors that contain inferencing units.
    - Accelerator sends these to the Pi.
- Vocab

    - Input Tensor: The part of the sensor image passed to the AI engine for inferencing. Produced by a small on-board ISP which also crops and scales the camera image to the dimensions expected by the neural network that has been loaded. The input tensor is not normally made available to applications, though it is possible to access it for debugging purposes
    - Region of Interest (ROI): Specifies exactly which part of the sensor image is cropped out before being rescaled to the size demanded by the neural network. Can be queried and set by an application. The units used are always pixels in the full resolution sensor output. The default ROI setting uses the full image received from the sensor, cropping no data.
    - Output Tensors: The results of inferencing performed by the neural network. The precise number and shape of the outputs depend on the neural network. Application code must understand how to handle the tensors.

- Pi Camera 2 functions:

Function	Description
`IMX500.get_full_sensor_resolution()`	Return the full sensor resolution of the IMX500.
`IMX500.config`	Returns a dictionary of the neural network configuration.
`IMX500.convert_inference_coords(coords, metadata, picamera2)`	Converts the coordinates *coords* from the input tensor coordinate space to the final ISP output image space. Must be passed Picamera2's image metadata for the image, and the Picamera2 object. There are a number of scaling/cropping/translation operations occurring from the original sensor image to the fully processed ISP output image. This function converts coordinates provided by the output tensor to the equivalent coordinates after performing these operations.
`IMX500.show_network_fw_progress_bar()`	Displays a progress bar on the console showing the progress of the neural network firmware upload to the IMX500.
`IMX500.get_roi_scaled(request)`	Returns the region of interest (ROI) in the ISP output image coordinate space.
`IMX500.get_isp_output_size(picamera2)`	Returns the ISP output image size.
`IMX5000.get_input_size()`	Returns the input tensor size based on the neural network model used.
`IMX500.get_outputs(metadata)`	Returns the output tensors from the Picamera2 image metadata.
`IMX500.get_output_shapes(metadata)`	Returns the shape of the output tensors from the Picamera2 image metadata for the neural network model used.
`IMX500.set_inference_roi_abs(rectangle)`	Sets the region of interest (ROI) crop rectangle which determines which part of the sensor image is converted to the input tensor that is used for inferencing on the IMX500. The region of interest

Function	Description
	should be specified in units of pixels at the full sensor resolution, as a `(x_offset, y_offset, width, height)` tuple.
`IMX500.set_inference_aspect_ratio(aspect_ratio)`	Automatically calculates region of interest (ROI) crop rectangle on the sensor image to preserve the given aspect ratio. To make the ROI aspect ratio exactly match the input tensor for this network, use `imx500.set_inference_aspect_ratio(imx500.get_input_size())`.
`IMX500.get_kpi_info(metadata)`	Returns the frame-level performance indicators logged by the IMX500 for the given image metadata.

To deploy a new neural network model to the Pi AI camera:

1. Provide a flaoting point NN model (PyTorch or TensorFlow)
2. Run the model throuhg Edge-MDT
    1. pytorch: pip install edge-mdt[pt]
    2. TensorFlow: pip install edge-mdt[tf]
        1. Convert the compressed model to IMX500 format
3. Package the model into a firmware file that can be loaded at runtime into the camera

Citation

“AI Camera - Raspberry Pi Documentation.” Accessed: Jan. 28, 2026. [Online]. Available: https://www.raspberrypi.com/documentation/accessories/ai-camera.html

Conclusions/action items:

Once the group acquires our Pi camera, we can utilize this code to upload our model onto the Raspberry Pi software.

2/19/2026 - User Interface Workflow Code Loop

Madison Michels - Feb 19, 2026, 2:00 AM CST

Title: User Interface Code Loop

Date: 12/19/2026

Content by: Maddie

Present: Maddie

Goals: The goal of this entry is to create a scaffold code for the user interface required to integrate our model into a suture training system.

Content:

Code Outline:

1. Wait for button to press
2. Turn white/yellow LED on (indicates processing)
3. Capture image from HQ camera
4. Run PyTorch model inference
5. Save image in the format: YYYYMMDD_HHMMSS_CLASSNAME.jpg
6. Turn
    1. Red LED on if positive (tight)
    2. Green LED on if negative (loose)
7. Turn white LED off
8. Reset and wait for next press

config.py:

```
# GPIO Pins (BCM mode)
BUTTON_PIN = 17

LED_RED = 27
LED_GREEN = 22
LED_WHITE = 23

# Model
MODEL_PATH = "model.pth"
CLASS_NAMES = ["loose", "tight"]  # change to yours
IMAGE_SIZE = 500

# Save folder
SAVE_DIR = "captures"
```

camera.py:

```
sudo apt install python3-picamera2

from picamera2 import Picamera2
from datetime import datetime
import os

class Camera:
    def __init__(self, save_dir):
        self.picam2 = Picamera2()
        self.picam2.configure(self.picam2.create_still_configuration())
        self.picam2.start()
        self.save_dir = save_dir
        os.makedirs(save_dir, exist_ok=True)
```

```
    def capture(self):
        timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")
        filename = f"{timestamp}.jpg"
        filepath = os.path.join(self.save_dir, filename)

        self.picam2.capture_file(filepath)
        return filepath, timestamp
```

model.py:

```
# model.py

import torch
import torch.nn as nn
from torchvision import transforms
from PIL import Image
from config import MODEL_PATH, CLASS_NAMES, IMAGE_SIZE

class Classifier:
    def __init__(self):
        self.device = torch.device("cuda" if torch.cuda.is_available() else "cpu")

        #     Replace with your actual model architecture
        self.model = torch.load(MODEL_PATH, map_location=self.device)
        self.model.eval()
        self.model.to(self.device)

        self.transform = transforms.Compose([
            transforms.Resize((IMAGE_SIZE, IMAGE_SIZE)),
            transforms.ToTensor(),
            transforms.Normalize([0.485, 0.456, 0.406],
                                 [0.229, 0.224, 0.225])
        ])

    def predict(self, image_path):
        image = Image.open(image_path).convert("RGB")
        tensor = self.transform(image).unsqueeze(0).to(self.device)

        with torch.no_grad():
            outputs = self.model(tensor)
            _, predicted = torch.max(outputs, 1)

        class_name = CLASS_NAMES[predicted.item()]
        return class_name
```

main.py:

```
import RPi.GPIO as GPIO
import time
import os

from config import *
from camera import Camera
from model import Classifier

def setup_gpio():
    GPIO.setmode(GPIO.BCM)

    GPIO.setup(BUTTON_PIN, GPIO.IN, pull_up_down=GPIO.PUD_UP)

    GPIO.setup(LED_RED, GPIO.OUT)
    GPIO.setup(LED_GREEN, GPIO.OUT)
    GPIO.setup(LED_WHITE, GPIO.OUT)
```

```
def set_led(red=False, green=False, white=False):
    GPIO.output(LED_RED, red)
    GPIO.output(LED_GREEN, green)
    GPIO.output(LED_WHITE, white)

def rename_with_class(filepath, timestamp, class_name):
    new_name = f"{timestamp}_{class_name}.jpg"
    new_path = os.path.join(SAVE_DIR, new_name)
    os.rename(filepath, new_path)
    return new_path

def main():
    setup_gpio()
    camera = Camera(SAVE_DIR)
    classifier = Classifier()

    print("System ready. Waiting for button press...")

    try:
        while True:
            if GPIO.input(BUTTON_PIN) == 0:  # button pressed

                print("Button pressed")
                set_led(white=True)

                # Capture image
                filepath, timestamp = camera.capture()
                print("Image captured")

                # Run model
                class_name = classifier.predict(filepath)
                print(f"Prediction: {class_name}")

                # Rename file
                new_path = rename_with_class(filepath, timestamp, class_name)

                # Turn on result LED
                if class_name == "positive":
                    set_led(red=True)
                else:
                    set_led(green=True)

                # Turn off white LED
                GPIO.output(LED_WHITE, False)

                # Show result for 3 seconds
                time.sleep(3)

                # Reset LEDs
                set_led(False, False, False)

                print("Ready for next capture\n")

                # debounce
                time.sleep(0.5)

            time.sleep(0.1)

    except KeyboardInterrupt:
        GPIO.cleanup()

if __name__ == "__main__":
    main()
```

Conclusions/action items:

Once we decide which model to upload to the Pi, we hope to be able to refine this code to create an entire workflow for our user system to operate. This code implements every aspect of our intended design.

2/2/2025 - Introduction to k-Fold Cross Validation

Madison Michels - Feb 02, 2026, 11:25 AM CST

Title: Introduction to k-Fold Cross Validation

Date: 2/2/2025

Content by: Maddie

Present: Maddie

Goals: The goal of this research is to learn how K-Folld operates and how to implement it into our model.

Content:

- Resampling procedure for limited sample model evaluation
- Used to evaluate multiple time to eliminate the possibility of one-time good results
- k = the number of groups that a given data sample is split into
- Evaluates model design, not model training (trained many times on different data)
- Procedure
    - Shuffle dataset randomly
    - Split dataset into k groups
    - For each unique group:
        - Take the group as a hold out or test data set
        - Take the remaining images as a training dataset
        - Fit a model on the training set and evaluate it on the test set
        - Retain the evaluation scores and discard the model
    - Summarize the skill of the model using the sample of model evaluation scores
    - Each sample is placed in the testing set 1 time and trains the model k-1 times
- Kfold scikit code:

```
# enumerate splits

for train, test in kfold.split(data):

print('train: %s, test: %s' % (train, test))
```

Citations:

J. Brownlee, “A Gentle Introduction to k-fold Cross-Validation,” MachineLearningMastery.com. Accessed: Feb. 02, 2026. [Online]. Available: https://www.machinelearningmastery.com/k-fold-cross-validation/

Conclusions/action items:

In conclusion, kfold evaluates the model's performance based on a series of trainings. We still need to determine what value of k to use for our model.

2/2/2025 - K-Fold Cross Validation in Machine Learning

Madison Michels - Feb 02, 2026, 2:03 PM CST

Title: K-Fold Cross Validation in Machine Learning

Date: 2/2/2025

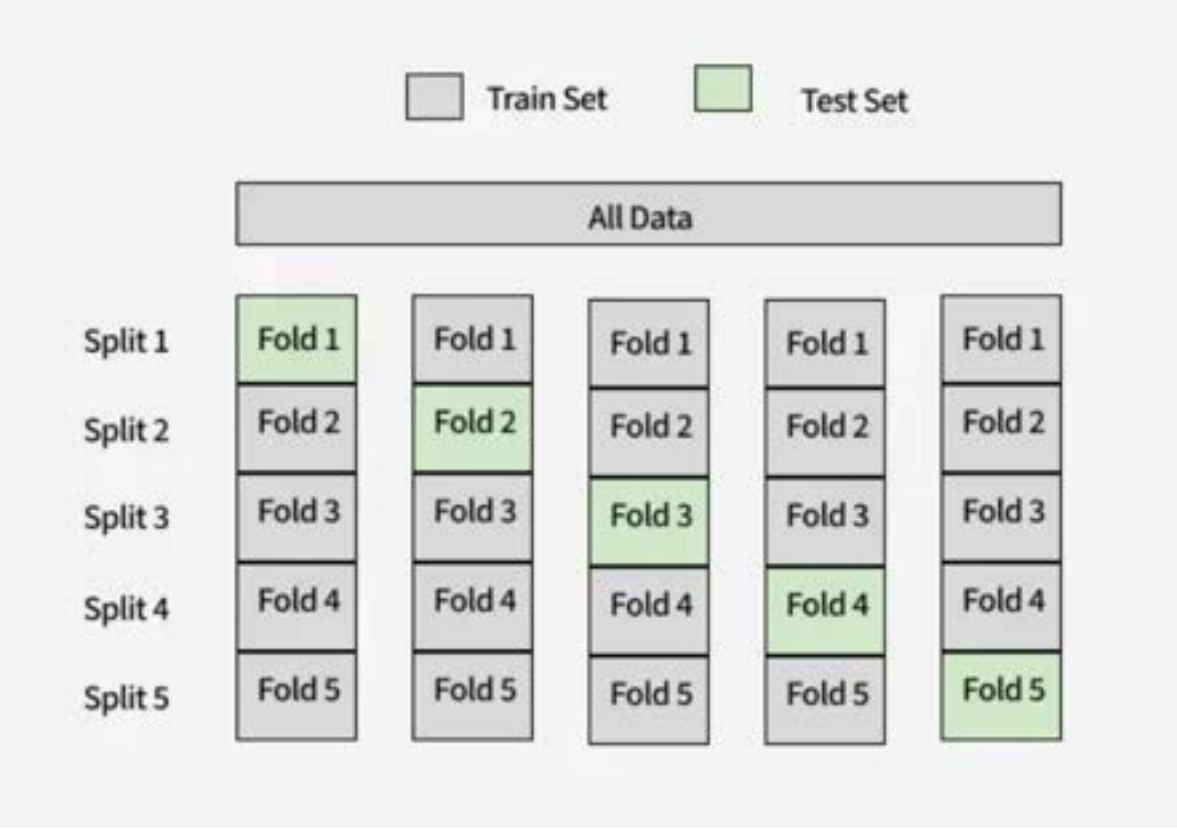
Content by: Maddie

Present: Maddie

Goals: The goal of this research is to learn how K-Fold operates and how to implement it into our model.

Content:

- Performance of the model is averaged over all K iterations to provide an estimate of its generalization ability.



- Avoids overfitting
- Maximizes utilization of dataset
- Aggregates results (accuracy, precision, recall...) for each fold and averages the results

The model's performance is computed as:

$$\text{Performance} = \frac{1}{K}\sum_{k=1}^{K} \text{Metric}(M_k, F_k)$$

- Choosing K
  - Small K (2-5): faster computation with increased variance in performance estimates
  - Large K (10-n): Lower variance but higher computation cost. K=n relates to Leave-one-out cross validation
- Variants of K-Fold cross validation

    - Stratified K-Fold cross validation: maintains the same class distribution for each fold as for the entire dataset (good for imbalanced datasets)
    - Repeated K-Fold Cross Validation: performs the K-Fold operation several times with varying splits, giving more stable performance estimates
    - Leave-one-out cross validation (LOOCV): a special case where K is equal to the number of data points, so every fold has one data point
- Implementation:

```
from sklearn.model_selection import KFold, cross_val_score
from sklearn.datasets import load_iris
from sklearn.ensemble import RandomForestClassifier
import numpy as np

data = load_iris()
X, y = data.data, data.target

# K-Fold Cross Validation
k = 5
kf = KFold(n_splits=k, shuffle=True, random_state=42)

# Initialize the RandomForestClassifier model
model = RandomForestClassifier(random_state=42)

# Perform Cross Validation
scores = cross_val_score(model, X, y, cv=kf, scoring='accuracy')

print(f"Accuracy for each fold: {scores}")

average_accuracy = np.mean(scores)
print(f"Average Accuracy: {average_accuracy:.2f}")
```

Citation:

> “K- Fold Cross Validation in Machine Learning,” GeeksforGeeks. Accessed: Feb. 02, 2026. [Online]. Available: https://www.geeksforgeeks.org/machine-learning/k-fold-cross-validation-in-machine-learning/

Conclusions/action items:

In conclusion, kfold evaluates the model's performance based on a series of trainings. The group will likely try to use a high K value, but depending on computer capabilities, might need to lower the value.

2/2/2025 - Cross-Validation Using K-Fold With Scikit-Learn

Madison Michels - Feb 02, 2026, 11:48 AM CST

Title: Cross-Validation Using K-Fold With Scikit-Learn

Date: 2/2/2025

Content by: Maddie

Present: Maddie

Goals: The goal of this research is to learn how K-Folld operates and how to implement it into our model.

Content:

*sklearn.model_selection.KFold(n_splits=5, *, shuffle=False, random_state=None)*

Parameters:

- *n_splits: number of folds, must be atleast 2*
  - *default is 5*
- *shuffle: whether to shuffle the data before splitting into batches*
  - *default = False*
  - *boolean value*
- *random_state: when shuffle is True, random_state affects the ordering of the indices, which controls the randomness of each fold*
  - *default = None*

Methods:

- get_metadata_routing
- get_n_splits(X=None, y=None, groups=None): returns the number of splitting iterations in the cross validation
- split(X, y=None, groups=None): generate indices to split data into training and testing sets

Code:

```
import numpy as np

from sklearn import datasets
from sklearn.model_selection import KFold

# synthetic regression dataset
X, y = datasets.make_regression(
  n_samples=10, n_features=1, n_informative=1,
  noise=0, random_state=0)

# KFold split
kf = KFold(n_splits=4)
for i, (train_index, test_index) in enumerate(kf.split(X)):
    print(f"Fold {i}:")
    print(f"  Training dataset index: {train_index}")
    print(f"  Test dataset index: {test_index}")
```

Check number of splits: *kf.get_n_splits(X)*

Visualization:

```
from sklearn.datasets import make_classification
# define dataset
X, y = make_classification(
  n_samples=100, n_features=20, n_informative=15, n_redundant=5)

# prepare the K-Fold cross-validation procedure
n_splits = 10
cv = KFold(n_splits=n_splits)
```

```
import matplotlib.pyplot as plt
from matplotlib.patches import Patch
import numpy as np

def plot_kfold(cv, X, y, ax, n_splits, xlim_max=100):
    """
    Plots the indices for a cross-validation object.

    Parameters:
    cv: Cross-validation object
    X: Feature set
    y: Target variable
    ax: Matplotlib axis object
    n_splits: Number of folds in the cross-validation
    xlim_max: Maximum limit for the x-axis
    """

    # Set color map for the plot
    cmap_cv = plt.cm.coolwarm
    cv_split = cv.split(X=X, y=y)

    for i_split, (train_idx, test_idx) in enumerate(cv_split):
        # Create an array of NaNs and fill in training/testing indices
        indices = np.full(len(X), np.nan)
        indices[test_idx], indices[train_idx] = 1, 0

        # Plot the training and testing indices
        ax_x = range(len(indices))
        ax_y = [i_split + 0.5] * len(indices)
        ax.scatter(ax_x, ax_y, c=indices, marker="_",
                   lw=10, cmap=cmap_cv, vmin=-0.2, vmax=1.2)

    # Set y-ticks and labels
    y_ticks = np.arange(n_splits) + 0.5
    ax.set(yticks=y_ticks, yticklabels=range(n_splits),
           xlabel="X index", ylabel="Fold",
           ylim=[n_splits, -0.2], xlim=[0, xlim_max])

    # Set plot title and create legend
    ax.set_title("KFold", fontsize=14)
    legend_patches = [Patch(color=cmap_cv(0.8), label="Testing set"),
                      Patch(color=cmap_cv(0.02), label="Training set")]
    ax.legend(handles=legend_patches, loc=(1.03, 0.8))

# Create figure and axis
fig, ax = plt.subplots(figsize=(6, 3))
plot_kfold(cv, X, y, ax, n_splits)
plt.tight_layout()
fig.subplots_adjust(right=0.6)



from numpy import mean
from numpy import std

from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import cross_val_score

# create model
log_reg = LogisticRegression()
# evaluate model
scores = cross_val_score(
    log_reg, X, y, scoring='accuracy', cv=cv, n_jobs=-1)
# accuracy
print('Accuracy: %.3f ,\nStandard Deviations :%.3f' %
      (mean(scores), std(scores)))
```

```
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import KFold
import numpy as np


X_housing = housing.data
y_housing = housing.target

# Scaling the data
scaler = StandardScaler()
X_scaler = scaler.fit_transform(X_housing)

# K-Fold split
cnt = 0
n_splits = 10
kf = KFold(n_splits=n_splits, shuffle=True, random_state=42)
for train_index, test_index in kf.split(X_scaler, y_housing):
    print(f'Fold:{cnt}, Train set: {len(train_index)}, \
    Test set:{len(test_index)}')
    cnt += 1



fig, ax = plt.subplots(figsize=(6, 3))
plot_kfold(kf, X_scaler, y_housing, ax, n_splits, xlim_max=2000)
# Make the legend fit
plt.tight_layout()
fig.subplots_adjust(right=0.7)
```

Citation:

“Cross-Validation Using K-Fold With Scikit-Learn,” GeeksforGeeks. Accessed: Feb. 02, 2026. [Online]. Available: https://www.geeksforgeeks.org/machine-learning/cross-validation-using-k-fold-with-scikit-learn/

Conclusions/action items:

In conclusion, this code will be used and modified to apply to our model.

2/13/2026 - Data Augmentation and How it Works

Madison Michels - Feb 18, 2026, 11:01 AM CST

Title: Data Augmentation and How it Works

Date: 2/13/2026

Content by: Maddie

Present: Maddie

Goals: To understand the most common types of data augmentation and how/why you use them.

Content:

- Geometric Transformations
  - Rotation
  - Flipping
  - Scaling (we don't want to do to our images, as the camera will be in a fixed location)
  - Translation (we don't want to do to our images, as the camera will be in a fixed location)
  - Shearing (we don't want to do to our images, as the camera will be in a fixed location)
- Color Space Augmentations
  - Brightness adjustments
  - Contrast adjustments
  - Saturation adjustments
  - Hue adjustments
- Kernel Filters
  - Blurring
  - Sharpening
  - Edge detection (won't do)
- Random Erasing - takes random rectangular cuts out of the image. Helps the model tune to occlusions in the images.
- Combinations of augmentations above.
- Improves model generalization
- Reduces overfitting
- Enhances robustness
- cost-effective dataset generation

Code:

```
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import matplotlib.pyplot as plt
import numpy as np

datagen = ImageDataGenerator(
    rotation_range=40,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest'
)

(imgs, labels), _ = tf.keras.datasets.cifar10.load_data()
img = imgs[0]

plt.figure(figsize=(6, 6))
plt.imshow(img.astype('uint8'))
plt.title("Original Image")
```

```
plt.axis('off')
plt.show()


i = 0
augmented_images = []
for batch in datagen.flow(img, batch_size=1):
    augmented_images.append(batch[0].astype('uint8'))
    i += 1
    if i % 4 == 0:
        break

fig, axes = plt.subplots(1, 4, figsize=(20, 5))
axes = axes.flatten()
for img, ax in zip(augmented_images, axes):
    ax.imshow(img)
    ax.axis('off')
plt.suptitle("Augmented Images")
plt.show()
```

Citation:

“What is Data Augmentation? How Does Data Augmentation Work for Images?,” GeeksforGeeks. Accessed: Feb. 13, 2026. [Online]. Available: https://www.geeksforgeeks.org/computer-vision/what-is-data-augmentation-how-does-data-augmentation-work-for-images/

Conclusions/action items:

In conclusion, we can implement rotation, flipping, brightness, contrast, saturation, and hue changes to replicate the images our model will see in practice.

2/10/2026 - ML Automated Image Cropping

Title: ML Automated Image Cropping

Date: 2/10/2026

Content by:

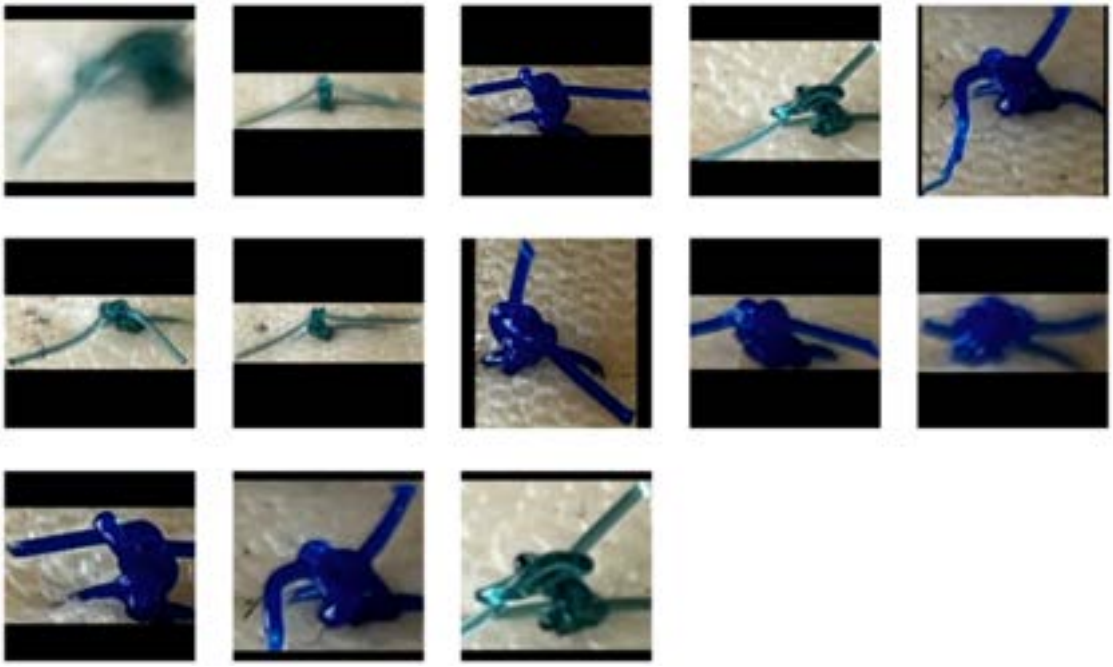
Present: Maddie

Goals: Maddie

Content:

- This code was used in an attempt to detect an object using a premade object detection algorithm and highlight that object in a box. The code will them crop the image down to the size of picture.
- I added a blur filter using the LaPlace function. This calculates the variance of each images by detecting edges in the picture and averaging their contrast.
  - A greater LaPlace value correlates to a "sharper" image
- Each knot in the image is highlighted with a red box and cropped down to the size of the box. Any remaining pixels needed to achieve the 224 x 224 box are filled in with black (as shown
- I added a threshold confidence value of 0.3 to ensure that only knots are cropped, not cropping to non-knot specimens.

The images shown below are taken and saved before the blur filter was activated:



After the blur filter was implemented, only two out of the images above passed the threshhold of 50.

```
import os
import random
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
# Load images
from PIL import Image
from skimage import img_as_float
from ultralytics import YOLO
import cv2
import matplotlib.pyplot as plt
import math
import numpy as np

from ultralytics import YOLO
import cv2
import matplotlib.pyplot as plt
import math
import numpy as np

# ---------- CONFIG ----------
IMAGE_PATH = r"C:\Users\madis\Downloads\Knot Model\sem2\lucypictures_2_9_26\IMG_9344 2.JPG"
TARGET_SIZE = (224, 224)
CONF_THRESHOLD = 0.3
BLUR_THRESHOLD = 50
# ----------------------------------------

# Load YOLO model
model = YOLO("yolov8n.pt")

image = cv2.imread(IMAGE_PATH)
results = model(image)

# Resize with padding (no distortion)
def resize_with_padding(img, target_size):
    h, w = img.shape[:2]
    scale = min(target_size[0] / w, target_size[1] / h)
    new_w, new_h = int(w * scale), int(h * scale)

    resized = cv2.resize(img, (new_w, new_h))
    padded = np.zeros((target_size[1], target_size[0], 3), dtype=np.uint8)

    x_offset = (target_size[0] - new_w) // 2
    y_offset = (target_size[1] - new_h) // 2
    padded[y_offset:y_offset + new_h, x_offset:x_offset + new_w] = resized
    return padded

# Blur detection (Laplacian variance)
def is_blurry(img, threshold):
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    variance = cv2.Laplacian(gray, cv2.CV_64F).var()
    return variance < threshold, variance

crops = []

boxes = sorted(results[0].boxes, key=lambda b: float(b.conf[0]), reverse=True)

for box in boxes:
    conf = float(box.conf[0])
    if conf < CONF_THRESHOLD:
        continue

    x1, y1, x2, y2 = map(int, box.xyxy[0])
    crop = image[y1:y2, x1:x2]

    if crop.size == 0:
        continue

    blurry, blur_score = is_blurry(crop, BLUR_THRESHOLD)
```

```
    if blurry:
        continue  # □ skip blurry crops

    crop_fixed = resize_with_padding(crop, TARGET_SIZE)
    crops.append(cv2.cvtColor(crop_fixed, cv2.COLOR_BGR2RGB))

# Display crops
n = len(crops)
if n == 0:
    print("No sharp crops passed the filters.")
else:
    cols = min(5, n)
    rows = math.ceil(n / cols)

    plt.figure(figsize=(15, 3 * rows))
    for i, crop in enumerate(crops):
        plt.subplot(rows, cols, i + 1)
        plt.imshow(crop)
        plt.axis("off")

    plt.show()
```

Conclusions/action items:

In conclusion, this code will not be used to crop all of the images entering the training set because it adds black borders where the images have been cropped to ensure they are the same size. objects as birds, which doesn't change the output of the boxes, but is not needed for the cropping step.

2/10/2026 - Automated Cropping to Resolution

Madison Michels - Feb 10, 2026, 11:48 PM CST

Title: Automated Image Cropping to Resolution

Date: 2/10/2026

Content by: Maddie

Present: Maddie

Goals: The goal of this code is to automate and expedite the cropping process before feeding the images into the ML algorithm for training and before augmentation.

Content:

- The BLUE code uses pillow_heif to convert any non JPG or JPEG files to that format. Many of the images taken for model training were taken with a phone as HEIC images.
- The RED code takes in a files folder of images. It sets the ideal target size to 500 x 500 pixels and begins at the center of the image. This crops the image from the crux of the center of the picture, where the knots centered in initial image capturing.
- Lastly, the images were saved by number.

```
import os
import random
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
# Load images
from PIL import Image
from skimage import img_as_float

from PIL import Image
import os

from PIL import Image
import pillow_heif
from pathlib import Path

pillow_heif.register_heif_opener()

input_dir = Path(r"C:\Users\madis\Downloads\Knot Model\sem2\sadieHEICuncropped")
output_dir = Path(r"C:\Users\madis\Downloads\Knot Model\sem2\sadieHEICuncropped")
output_dir.mkdir(exist_ok=True)

for heic_file in input_dir.glob("*.heic"):
    img = Image.open(heic_file)
    output_path = output_dir / (heic_file.stem + ".jpg")
    img.save(output_path, "JPEG", quality=95)

print("Done!")


INPUT_DIR = r"C:\Users\madis\Downloads\Knot Model\sem2\loose2_uncropped"
OUTPUT_DIR = r"C:\Users\madis\Downloads\Knot Model\sem2\loose2_cropped"
```

```
TARGET_WIDTH = 500
TARGET_HEIGHT = 500

os.makedirs(OUTPUT_DIR, exist_ok=True)

index = 1
skipped = 0

for filename in sorted(os.listdir(INPUT_DIR)):
    if not filename.lower().endswith((".jpg", ".jpeg", ".png")):
        continue

    input_path = os.path.join(INPUT_DIR, filename)

    with Image.open(input_path) as img:
        img = img.convert("RGB")
        w, h = img.size
        if w < TARGET_WIDTH or h < TARGET_HEIGHT:
            skipped += 1
            continue

        left = (w - TARGET_WIDTH) // 2
        top = (h - TARGET_HEIGHT) // 2
        right = left + TARGET_WIDTH
        bottom = top + TARGET_HEIGHT

        cropped = img.crop((left, top, right, bottom))
        output_name = f"{index:06d}.jpg"
        output_path = os.path.join(OUTPUT_DIR, output_name)
        cropped.save(output_path, quality=95)
        index += 1

print(f"Saved images: {index - 1}")
print(f"Skipped (too small): {skipped}")
```

End Result:

---------->

Conclusions/action items:

In conclusion, this code will be used to crop all of the images entering the training set because it can center the larger picture and cut out a consistent square from the middle.

2/18/2026 - K-Fold Cross Validation

Madison Michels - Feb 18, 2026, 11:51 PM CST

Title: K-Fold Cross Validation (for Augmented Images)

Date: 2/18/2026

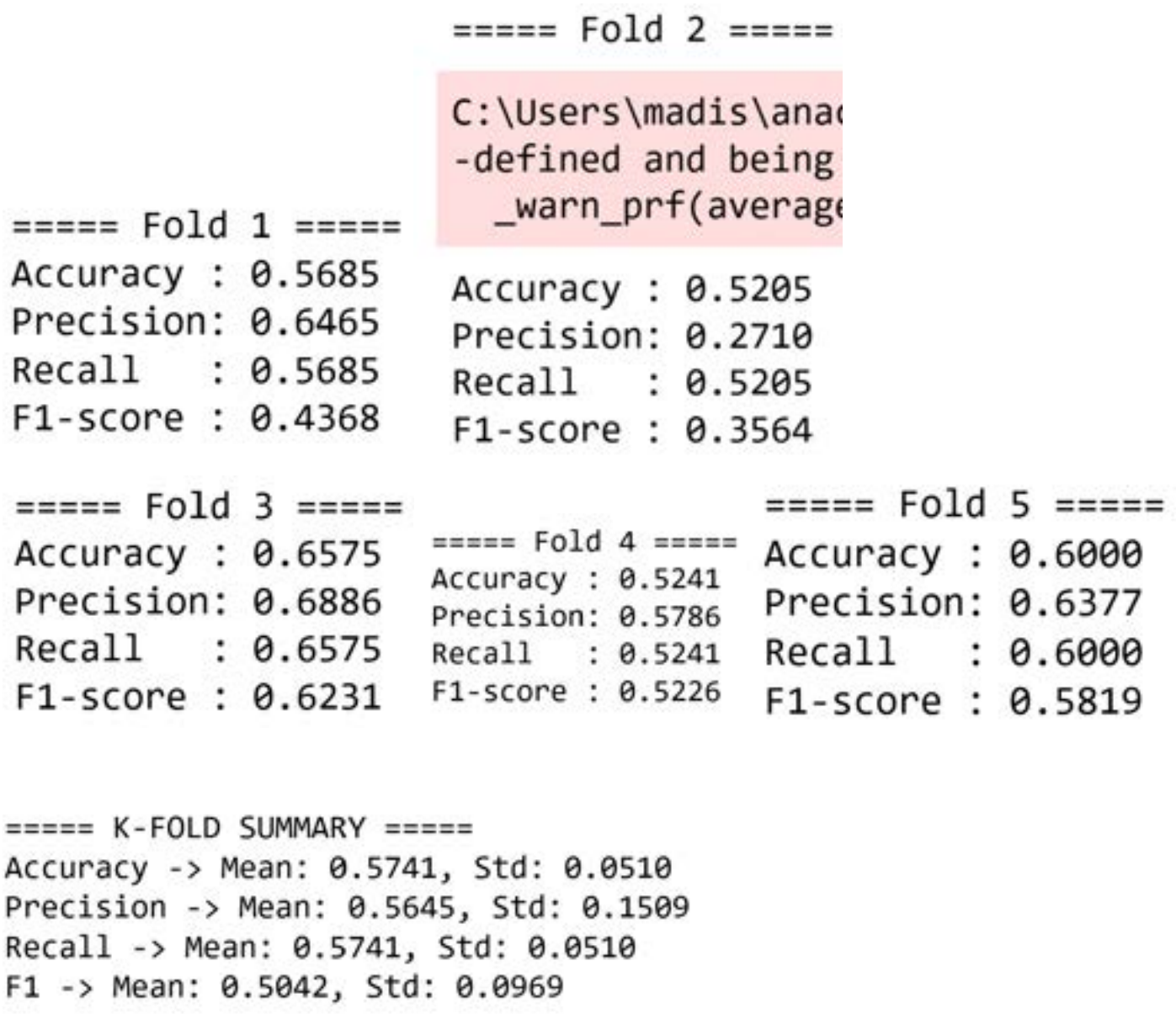
Content by: Maddie

Present: Maddie

Goals: The goal of this code is to perform k-fold cross validation with k=5 on our augmented images. We hope to obtain an accuracy, recall, precision, and F1 score from each fold and average them at the end.

Content:

RESULTS:



CODE:

import torch
from torchvision import datasets, transforms

```
from torch.utils.data import DataLoader
from sklearn.model_selection import KFold
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
from torch.utils.data import Subset, DataLoader
from torchvision.models import resnet18, ResNet18_Weights
import torch
import torch.nn as nn
import torch.optim as optim
import numpy as np


transform = transforms.Compose([
    transforms.ToTensor(),
])

dataset = datasets.ImageFolder(
    root=r"C:\Users\madis\Downloads\Knot Model\sem2\cropped_final_500_500",
    transform=transform
)
print("Classes:", dataset.classes)
print("Total images:", len(dataset))



device = torch.device("cuda" if torch.cuda.is_available() else "cpu")

kfold = KFold(n_splits=5, shuffle=True, random_state=42)

fold_metrics = {
    "accuracy": [],
    "precision": [],
    "recall": [],
    "f1": []
}

for fold, (train_ids, val_ids) in enumerate(kfold.split(dataset)):

    print(f"\n===== Fold {fold + 1} =====")

    train_subset = Subset(dataset, train_ids)
    val_subset = Subset(dataset, val_ids)

    train_loader = DataLoader(train_subset, batch_size=32, shuffle=True)
    val_loader = DataLoader(val_subset, batch_size=32, shuffle=False)

    # Model
    model = resnet18(weights=ResNet18_Weights.DEFAULT)
    model.fc = nn.Linear(model.fc.in_features, len(dataset.classes))
    model = model.to(device)

    criterion = nn.CrossEntropyLoss()
    optimizer = optim.Adam(model.parameters(), lr=0.001)

    # -------- TRAIN (1 epoch example) --------
    model.train()
    for images, labels in train_loader:
        images, labels = images.to(device), labels.to(device)

        optimizer.zero_grad()
        outputs = model(images)
        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()

    # -------- VALIDATION --------
```

```
    model.eval()
    all_preds = []
    all_labels = []

    with torch.no_grad():
        for images, labels in val_loader:
            images = images.to(device)
            outputs = model(images)
            _, preds = torch.max(outputs, 1)

            all_preds.extend(preds.cpu().numpy())
            all_labels.extend(labels.numpy())

    # Calculate metrics
    acc = accuracy_score(all_labels, all_preds)
    prec = precision_score(all_labels, all_preds, average='weighted')
    rec = recall_score(all_labels, all_preds, average='weighted')
    f1 = f1_score(all_labels, all_preds, average='weighted')

    # Store metrics
    fold_metrics["accuracy"].append(acc)
    fold_metrics["precision"].append(prec)
    fold_metrics["recall"].append(rec)
    fold_metrics["f1"].append(f1)

    print(f"Accuracy : {acc:.4f}")
    print(f"Precision: {prec:.4f}")
    print(f"Recall   : {rec:.4f}")
    print(f"F1-score : {f1:.4f}")

# -------- FINAL SUMMARY --------
print("\n===== K-FOLD SUMMARY =====")

for metric in fold_metrics:
    mean = np.mean(fold_metrics[metric])
    std = np.std(fold_metrics[metric])
    print(f"{metric.capitalize()} -> Mean: {mean:.4f}, Std: {std:.4f}")
```

Conclusions/action items:

In conclusion, these images did not perform as well as we had hoped for our new model. The team will need to look into other ways to augment our model, but will continue to train a model on these images. We are hoping for accuracy above 80% and F1 of 0.8.

2/18/2026 - K-Fold Cross Validation for Non-Augmented Images

Madison Michels - Feb 23, 2026, 11:03 AM CST

Title: K-Fold Cross Validation (for Augmented Images)

Date: 2/18/2026

Content by: Maddie

Present: Maddie

Goals: The goal of this code is to perform k-fold cross validation with k=10 on our augmented images. We hope to obtain an accuracy, recall, precision, and F1 score from each fold and average them at the end. These files are split by the folds we divided them into.

Content:

```
===== Fold 1 =====

Train files:
LNEW_001.jpg
LNEW_002.jpg
LNEW_003.jpg
LNEW_004.jpg
LNEW_005.jpg
LNEW_007.jpg
LNEW_008.jpg
LNEW_009.jpg
LNEW_010.jpg
LNEW_012.jpg
LNEW_013.jpg
LNEW_014.jpg
L_012.jpg
L_013.jpg
L_015.jpg
L_022.jpg
L_023.jpg
L_024.jpg
L_027.jpg
L_029.jpg
L_030.jpg
L_031.jpg
L_034.jpg
L_036.jpg
L_037.jpg
L_038.jpg
L_040.jpg
L_051.jpg
L_052.jpg
L_053.jpg
L_054.jpg
L_055.jpg
L_058.jpg
L_059.jpg
L_060.jpg
L_070.jpg
L_072.jpg
L_074.jpg
L_075.jpg
L_077.jpg
L_082.jpg
L_083.jpg
S_003.jpg
S_004.jpg
S_005.jpg
S_006.jpg
S_007.jpg
S_008.jpg
S_009.jpg
S_010.jpg
S_011.jpg
S_012.jpg
S_013.jpg
S_014.jpg
S_015.jpg
S_016.jpg
S_017.jpg
S_018.jpg
S_019.jpg
S_020.jpg
S_021.jpg
S_024.jpg
S_025.jpg
S_026.jpg
```

S_027.jpg
S_028.jpg
S_029.jpg
S_031.jpg
S_032.jpg
S_033.jpg
S_034.jpg
S_035.jpg
S_037.jpg
S_038.jpg
S_039.jpg
S_041.jpg
S_042.jpg
S_045.jpg
S_048.jpg
S_049.jpg
S_050.jpg
S_051.jpg
S_052.jpg
S_057.jpg
S_059.jpg
old_003.jpg
old_004.jpg
old_005.jpg
old_007.jpg
old_008.jpg
old_009.jpg
old_010.jpg
old_011.jpg
old_013.jpg
old_014.jpg
old_015.jpg
old_016.jpg
old_017.jpg
old_018.jpg
old_020.jpg
old_021.jpg
old_032.jpg
old_035.jpg
old_054.jpg
old_057.jpg
old_065.jpg
old_066.jpg
old_068.jpg
old_069.jpg
old_071.jpg
old_082.jpg
old_083.jpg
old_085.jpg
old_086.jpg
old_090.jpg
old_091.jpg
old_092.jpg
old_093.jpg
old_094.jpg
old_095.jpg
old_096.jpg
old_097.jpg
old_098.jpg
old_103.jpg
old_104.jpg
old_106.jpg
old_107.jpg
old_109.jpg
old_111.jpg
old_112.jpg
L_001.jpg
L_002.jpg

L_003.jpg
L_004.jpg
L_005.jpg
L_008.jpg
L_009.jpg
L_010.jpg
L_011.jpg
L_014.jpg
L_016.jpg
L_025.jpg
L_026.jpg
L_028.jpg
L_032.jpg
L_039.jpg
L_041.jpg
L_042.jpg
L_043.jpg
L_044.jpg
L_045.jpg
L_046.jpg
L_047.jpg
L_048.jpg
L_049.jpg
L_050.jpg
L_057.jpg
L_061.jpg
L_062.jpg
L_063.jpg
L_064.jpg
L_065.jpg
L_066.jpg
L_067.jpg
L_068.jpg
L_071.jpg
SNEW_001.jpg
SNEW_002.jpg
SNEW_003.jpg
SNEW_004.jpg
SNEW_005.jpg
SNEW_006.jpg
SNEW_007.jpg
SNEW_008.jpg
SNEW_009.jpg
SNEW_010.jpg
SNEW_011.jpg
SNEW_012.jpg
SNEW_013.jpg
SNEW_014.jpg
SNEW_015.jpg
SNEW_016.jpg
SNEW_017.jpg
SNEW_018.jpg
SNEW_019.jpg
SNEW_020.jpg
SNEW_021.jpg
SNEW_022.jpg
SNEW_023.jpg
SNEW_024.jpg
SNEW_025.jpg
SNEW_026.jpg
SNEW_027.jpg
SNEW_028.jpg
SNEW_029.jpg
SNEW_030.jpg
SNEW_031.jpg
SNEW_032.jpg
SNEW_034.jpg
SNEW_035.jpg

```
SNEW_037.jpg
SNEW_038.jpg
SNEW_039.jpg
SNEW_041.jpg
SNEW_042.jpg
SNEW_044.jpg
SNEW_045.jpg
SNEW_047.jpg
SNEW_048.jpg
SNEW_049.jpg
SNEW_050.jpg
SNEW_051.jpg
SNEW_052.jpg
SNEW_054.jpg
SNEW_055.jpg
SNEW_056.jpg
SNEW_057.jpg
SNEW_058.jpg
SNEW_059.jpg
SNEW_060.jpg
S_030.jpg
S_044.jpg
old_001.jpg
old_003.jpg
old_004.jpg
old_006.jpg
old_007.jpg
old_010.jpg
old_011.jpg
old_012.jpg
old_013.jpg
old_014.jpg
old_015.jpg
old_016.jpg
old_017.jpg
old_018.jpg
old_020.jpg
old_021.jpg
old_023.jpg
old_024.jpg
old_025.jpg
old_027.jpg
old_028.jpg
old_029.jpg
old_030.jpg
old_031.jpg
old_032.jpg
old_033.jpg
old_034.jpg
old_035.jpg
old_036.jpg
old_038.jpg
old_039.jpg

Validation files:
LNEW_011.jpg
L_056.jpg
L_081.jpg
S_001.jpg
S_002.jpg
S_036.jpg
S_058.jpg
old_001.jpg
old_002.jpg
old_019.jpg
old_089.jpg
L_006.jpg
L_007.jpg
```

```
L_021.jpg
L_069.jpg
SNEW_033.jpg
SNEW_036.jpg
SNEW_040.jpg
SNEW_043.jpg
SNEW_046.jpg
SNEW_053.jpg
old_002.jpg
old_005.jpg
old_008.jpg
old_009.jpg
old_019.jpg
old_022.jpg
old_026.jpg
old_037.jpg
```

Accuracy : 0.3793
Precision: 0.1439
Recall : 0.3793
F1-score : 0.2086

```
===== Fold 2 =====

Train files:
LNEW_001.jpg
LNEW_002.jpg
LNEW_003.jpg
LNEW_004.jpg
LNEW_005.jpg
LNEW_007.jpg
LNEW_009.jpg
LNEW_010.jpg
LNEW_011.jpg
LNEW_012.jpg
LNEW_013.jpg
LNEW_014.jpg
L_012.jpg
L_013.jpg
L_015.jpg
L_022.jpg
L_023.jpg
L_024.jpg
L_027.jpg
L_029.jpg
L_030.jpg
L_034.jpg
L_038.jpg
L_040.jpg
L_051.jpg
L_052.jpg
L_054.jpg
L_055.jpg
L_056.jpg
L_058.jpg
L_059.jpg
L_060.jpg
L_070.jpg
L_072.jpg
L_074.jpg
L_075.jpg
L_077.jpg
L_081.jpg
L_082.jpg
L_083.jpg
S_001.jpg
S_002.jpg
S_003.jpg
S_004.jpg
S_005.jpg
S_006.jpg
S_007.jpg
S_008.jpg
S_009.jpg
S_010.jpg
S_011.jpg
S_013.jpg
S_014.jpg
S_015.jpg
S_017.jpg
S_018.jpg
S_019.jpg
S_020.jpg
S_021.jpg
```

S_025.jpg
S_027.jpg
S_028.jpg
S_029.jpg
S_031.jpg
S_033.jpg
S_034.jpg
S_035.jpg
S_036.jpg
S_037.jpg
S_039.jpg
S_041.jpg
S_042.jpg
S_045.jpg
S_048.jpg
S_049.jpg
S_050.jpg
S_051.jpg
S_052.jpg
S_057.jpg
S_058.jpg
S_059.jpg
old_001.jpg
old_002.jpg
old_003.jpg
old_004.jpg
old_005.jpg
old_007.jpg
old_008.jpg
old_009.jpg
old_010.jpg
old_011.jpg
old_013.jpg
old_014.jpg
old_015.jpg
old_016.jpg
old_017.jpg
old_018.jpg
old_019.jpg
old_020.jpg
old_021.jpg
old_032.jpg
old_057.jpg
old_065.jpg
old_066.jpg
old_068.jpg
old_069.jpg
old_071.jpg
old_082.jpg
old_083.jpg
old_085.jpg
old_086.jpg
old_089.jpg
old_090.jpg
old_092.jpg
old_093.jpg
old_094.jpg
old_095.jpg
old_096.jpg
old_097.jpg
old_098.jpg
old_103.jpg
old_104.jpg
old_106.jpg
old_109.jpg
old_112.jpg
L_001.jpg
L_002.jpg

L_003.jpg
L_004.jpg
L_005.jpg
L_006.jpg
L_007.jpg
L_009.jpg
L_010.jpg
L_011.jpg
L_014.jpg
L_016.jpg
L_021.jpg
L_025.jpg
L_026.jpg
L_028.jpg
L_039.jpg
L_041.jpg
L_042.jpg
L_043.jpg
L_044.jpg
L_047.jpg
L_048.jpg
L_049.jpg
L_050.jpg
L_057.jpg
L_061.jpg
L_062.jpg
L_063.jpg
L_064.jpg
L_065.jpg
L_066.jpg
L_067.jpg
L_068.jpg
L_069.jpg
L_071.jpg
SNEW_001.jpg
SNEW_002.jpg
SNEW_003.jpg
SNEW_004.jpg
SNEW_005.jpg
SNEW_006.jpg
SNEW_007.jpg
SNEW_008.jpg
SNEW_009.jpg
SNEW_010.jpg
SNEW_011.jpg
SNEW_012.jpg
SNEW_013.jpg
SNEW_014.jpg
SNEW_015.jpg
SNEW_016.jpg
SNEW_018.jpg
SNEW_019.jpg
SNEW_020.jpg
SNEW_021.jpg
SNEW_022.jpg
SNEW_023.jpg
SNEW_024.jpg
SNEW_025.jpg
SNEW_026.jpg
SNEW_028.jpg
SNEW_029.jpg
SNEW_032.jpg
SNEW_033.jpg
SNEW_034.jpg
SNEW_035.jpg
SNEW_036.jpg
SNEW_037.jpg
SNEW_038.jpg

```
SNEW_039.jpg
SNEW_040.jpg
SNEW_042.jpg
SNEW_043.jpg
SNEW_044.jpg
SNEW_045.jpg
SNEW_046.jpg
SNEW_048.jpg
SNEW_049.jpg
SNEW_050.jpg
SNEW_051.jpg
SNEW_052.jpg
SNEW_053.jpg
SNEW_054.jpg
SNEW_055.jpg
SNEW_056.jpg
SNEW_057.jpg
SNEW_059.jpg
SNEW_060.jpg
S_044.jpg
old_001.jpg
old_002.jpg
old_003.jpg
old_004.jpg
old_005.jpg
old_006.jpg
old_007.jpg
old_008.jpg
old_009.jpg
old_010.jpg
old_011.jpg
old_012.jpg
old_013.jpg
old_014.jpg
old_015.jpg
old_016.jpg
old_017.jpg
old_018.jpg
old_019.jpg
old_020.jpg
old_021.jpg
old_022.jpg
old_023.jpg
old_024.jpg
old_025.jpg
old_026.jpg
old_027.jpg
old_028.jpg
old_030.jpg
old_031.jpg
old_032.jpg
old_033.jpg
old_034.jpg
old_035.jpg
old_036.jpg
old_037.jpg
old_038.jpg
old_039.jpg

Validation files:
LNEW_008.jpg
L_031.jpg
L_036.jpg
L_037.jpg
L_053.jpg
S_012.jpg
S_016.jpg
S_024.jpg
```

```
S_026.jpg
S_032.jpg
S_038.jpg
old_035.jpg
old_054.jpg
old_091.jpg
old_107.jpg
old_111.jpg
L_008.jpg
L_032.jpg
L_045.jpg
L_046.jpg
SNEW_017.jpg
SNEW_027.jpg
SNEW_030.jpg
SNEW_031.jpg
SNEW_041.jpg
SNEW_047.jpg
SNEW_058.jpg
S_030.jpg
old_029.jpg
Accuracy : 0.6552
Precision: 0.7878
Recall   : 0.6552
F1-score : 0.5885

===== Fold 3 =====

Train files:
LNEW_001.jpg
LNEW_002.jpg
LNEW_003.jpg
LNEW_004.jpg
LNEW_005.jpg
LNEW_008.jpg
LNEW_009.jpg
LNEW_010.jpg
LNEW_011.jpg
LNEW_013.jpg
LNEW_014.jpg
L_012.jpg
L_013.jpg
L_023.jpg
L_029.jpg
L_030.jpg
L_031.jpg
L_034.jpg
L_036.jpg
L_037.jpg
L_038.jpg
L_040.jpg
L_051.jpg
L_052.jpg
L_053.jpg
L_054.jpg
L_055.jpg
L_056.jpg
L_058.jpg
L_059.jpg
L_060.jpg
L_072.jpg
L_074.jpg
L_075.jpg
L_077.jpg
L_081.jpg
L_082.jpg
L_083.jpg
S_001.jpg
```

S_002.jpg
S_003.jpg
S_004.jpg
S_005.jpg
S_006.jpg
S_007.jpg
S_008.jpg
S_009.jpg
S_010.jpg
S_011.jpg
S_012.jpg
S_013.jpg
S_014.jpg
S_015.jpg
S_016.jpg
S_017.jpg
S_018.jpg
S_019.jpg
S_020.jpg
S_021.jpg
S_024.jpg
S_026.jpg
S_027.jpg
S_028.jpg
S_029.jpg
S_031.jpg
S_032.jpg
S_033.jpg
S_035.jpg
S_036.jpg
S_037.jpg
S_038.jpg
S_039.jpg
S_041.jpg
S_045.jpg
S_049.jpg
S_051.jpg
S_052.jpg
S_057.jpg
S_058.jpg
S_059.jpg
old_001.jpg
old_002.jpg
old_003.jpg
old_004.jpg
old_005.jpg
old_008.jpg
old_009.jpg
old_010.jpg
old_013.jpg
old_014.jpg
old_015.jpg
old_016.jpg
old_017.jpg
old_018.jpg
old_019.jpg
old_021.jpg
old_032.jpg
old_035.jpg
old_054.jpg
old_057.jpg
old_065.jpg
old_068.jpg
old_069.jpg
old_071.jpg
old_083.jpg
old_085.jpg
old_086.jpg

old_089.jpg
old_090.jpg
old_091.jpg
old_093.jpg
old_094.jpg
old_095.jpg
old_096.jpg
old_097.jpg
old_098.jpg
old_103.jpg
old_104.jpg
old_106.jpg
old_107.jpg
old_109.jpg
old_111.jpg
L_001.jpg
L_002.jpg
L_003.jpg
L_004.jpg
L_005.jpg
L_006.jpg
L_007.jpg
L_008.jpg
L_009.jpg
L_010.jpg
L_011.jpg
L_014.jpg
L_016.jpg
L_021.jpg
L_025.jpg
L_026.jpg
L_032.jpg
L_041.jpg
L_042.jpg
L_043.jpg
L_044.jpg
L_045.jpg
L_046.jpg
L_047.jpg
L_048.jpg
L_049.jpg
L_050.jpg
L_057.jpg
L_061.jpg
L_062.jpg
L_063.jpg
L_064.jpg
L_065.jpg
L_066.jpg
L_067.jpg
L_069.jpg
L_071.jpg
SNEW_001.jpg
SNEW_002.jpg
SNEW_003.jpg
SNEW_004.jpg
SNEW_005.jpg
SNEW_006.jpg
SNEW_007.jpg
SNEW_008.jpg
SNEW_009.jpg
SNEW_010.jpg
SNEW_011.jpg
SNEW_012.jpg
SNEW_014.jpg
SNEW_015.jpg
SNEW_017.jpg
SNEW_018.jpg

SNEW_019.jpg
SNEW_020.jpg
SNEW_021.jpg
SNEW_022.jpg
SNEW_023.jpg
SNEW_024.jpg
SNEW_025.jpg
SNEW_026.jpg
SNEW_027.jpg
SNEW_028.jpg
SNEW_029.jpg
SNEW_030.jpg
SNEW_031.jpg
SNEW_033.jpg
SNEW_034.jpg
SNEW_035.jpg
SNEW_036.jpg
SNEW_037.jpg
SNEW_038.jpg
SNEW_039.jpg
SNEW_040.jpg
SNEW_041.jpg
SNEW_042.jpg
SNEW_043.jpg
SNEW_044.jpg
SNEW_045.jpg
SNEW_046.jpg
SNEW_047.jpg
SNEW_048.jpg
SNEW_049.jpg
SNEW_050.jpg
SNEW_051.jpg
SNEW_052.jpg
SNEW_053.jpg
SNEW_054.jpg
SNEW_055.jpg
SNEW_056.jpg
SNEW_057.jpg
SNEW_058.jpg
SNEW_059.jpg
SNEW_060.jpg
S_030.jpg
S_044.jpg
old_001.jpg
old_002.jpg
old_003.jpg
old_004.jpg
old_005.jpg
old_006.jpg
old_007.jpg
old_008.jpg
old_009.jpg
old_010.jpg
old_011.jpg
old_012.jpg
old_013.jpg
old_014.jpg
old_015.jpg
old_016.jpg
old_018.jpg
old_019.jpg
old_020.jpg
old_022.jpg
old_024.jpg
old_025.jpg
old_026.jpg
old_027.jpg
old_028.jpg

```
old_029.jpg
old_030.jpg
old_031.jpg
old_032.jpg
old_033.jpg
old_034.jpg
old_035.jpg
old_036.jpg
old_037.jpg
old_038.jpg
old_039.jpg

Validation files:
LNEW_007.jpg
LNEW_012.jpg
L_015.jpg
L_022.jpg
L_024.jpg
L_027.jpg
L_070.jpg
S_025.jpg
S_034.jpg
S_042.jpg
S_048.jpg
S_050.jpg
old_007.jpg
old_011.jpg
old_020.jpg
old_066.jpg
old_082.jpg
old_092.jpg
old_112.jpg
L_028.jpg
L_039.jpg
L_068.jpg
SNEW_013.jpg
SNEW_016.jpg
SNEW_032.jpg
old_017.jpg
old_021.jpg
old_023.jpg
Accuracy : 0.6071
Precision: 0.5698
Recall   : 0.6071
F1-score : 0.5822

===== Fold 4 =====

Train files:
LNEW_001.jpg
LNEW_002.jpg
LNEW_004.jpg
LNEW_005.jpg
LNEW_007.jpg
LNEW_008.jpg
LNEW_009.jpg
LNEW_010.jpg
LNEW_011.jpg
LNEW_012.jpg
LNEW_013.jpg
LNEW_014.jpg
L_012.jpg
L_013.jpg
L_015.jpg
L_022.jpg
L_023.jpg
L_024.jpg
L_027.jpg
```

L_029.jpg
L_030.jpg
L_031.jpg
L_034.jpg
L_036.jpg
L_037.jpg
L_038.jpg
L_040.jpg
L_051.jpg
L_052.jpg
L_053.jpg
L_055.jpg
L_056.jpg
L_058.jpg
L_059.jpg
L_060.jpg
L_070.jpg
L_074.jpg
L_075.jpg
L_077.jpg
L_081.jpg
L_082.jpg
L_083.jpg
S_001.jpg
S_002.jpg
S_003.jpg
S_004.jpg
S_005.jpg
S_006.jpg
S_007.jpg
S_008.jpg
S_009.jpg
S_010.jpg
S_012.jpg
S_014.jpg
S_015.jpg
S_016.jpg
S_017.jpg
S_018.jpg
S_019.jpg
S_020.jpg
S_021.jpg
S_024.jpg
S_025.jpg
S_026.jpg
S_028.jpg
S_029.jpg
S_031.jpg
S_032.jpg
S_033.jpg
S_034.jpg
S_036.jpg
S_038.jpg
S_039.jpg
S_041.jpg
S_042.jpg
S_045.jpg
S_048.jpg
S_049.jpg
S_050.jpg
S_051.jpg
S_052.jpg
S_057.jpg
S_058.jpg
S_059.jpg
old_001.jpg
old_002.jpg
old_003.jpg

old_004.jpg
old_005.jpg
old_007.jpg
old_008.jpg
old_009.jpg
old_010.jpg
old_011.jpg
old_013.jpg
old_014.jpg
old_016.jpg
old_017.jpg
old_018.jpg
old_019.jpg
old_020.jpg
old_021.jpg
old_035.jpg
old_054.jpg
old_066.jpg
old_069.jpg
old_082.jpg
old_083.jpg
old_085.jpg
old_086.jpg
old_089.jpg
old_090.jpg
old_091.jpg
old_092.jpg
old_093.jpg
old_094.jpg
old_095.jpg
old_096.jpg
old_097.jpg
old_098.jpg
old_103.jpg
old_104.jpg
old_106.jpg
old_107.jpg
old_109.jpg
old_111.jpg
old_112.jpg
L_001.jpg
L_004.jpg
L_005.jpg
L_006.jpg
L_007.jpg
L_008.jpg
L_009.jpg
L_010.jpg
L_011.jpg
L_014.jpg
L_016.jpg
L_021.jpg
L_025.jpg
L_026.jpg
L_028.jpg
L_032.jpg
L_039.jpg
L_041.jpg
L_042.jpg
L_043.jpg
L_044.jpg
L_045.jpg
L_046.jpg
L_047.jpg
L_050.jpg
L_057.jpg
L_061.jpg
L_062.jpg

L_063.jpg
L_064.jpg
L_065.jpg
L_066.jpg
L_067.jpg
L_068.jpg
L_069.jpg
L_071.jpg
SNEW_001.jpg
SNEW_005.jpg
SNEW_006.jpg
SNEW_007.jpg
SNEW_008.jpg
SNEW_009.jpg
SNEW_010.jpg
SNEW_011.jpg
SNEW_012.jpg
SNEW_013.jpg
SNEW_014.jpg
SNEW_016.jpg
SNEW_017.jpg
SNEW_018.jpg
SNEW_019.jpg
SNEW_020.jpg
SNEW_021.jpg
SNEW_022.jpg
SNEW_023.jpg
SNEW_025.jpg
SNEW_026.jpg
SNEW_027.jpg
SNEW_028.jpg
SNEW_029.jpg
SNEW_030.jpg
SNEW_031.jpg
SNEW_032.jpg
SNEW_033.jpg
SNEW_034.jpg
SNEW_035.jpg
SNEW_036.jpg
SNEW_038.jpg
SNEW_039.jpg
SNEW_040.jpg
SNEW_041.jpg
SNEW_042.jpg
SNEW_043.jpg
SNEW_044.jpg
SNEW_045.jpg
SNEW_046.jpg
SNEW_047.jpg
SNEW_048.jpg
SNEW_049.jpg
SNEW_050.jpg
SNEW_051.jpg
SNEW_052.jpg
SNEW_053.jpg
SNEW_054.jpg
SNEW_055.jpg
SNEW_056.jpg
SNEW_057.jpg
SNEW_058.jpg
SNEW_059.jpg
SNEW_060.jpg
S_030.jpg
old_001.jpg
old_002.jpg
old_003.jpg
old_005.jpg
old_006.jpg

```
old_007.jpg
old_008.jpg
old_009.jpg
old_010.jpg
old_011.jpg
old_012.jpg
old_013.jpg
old_014.jpg
old_015.jpg
old_016.jpg
old_017.jpg
old_019.jpg
old_020.jpg
old_021.jpg
old_022.jpg
old_023.jpg
old_024.jpg
old_025.jpg
old_026.jpg
old_027.jpg
old_028.jpg
old_029.jpg
old_030.jpg
old_032.jpg
old_033.jpg
old_034.jpg
old_035.jpg
old_036.jpg
old_037.jpg
old_038.jpg
old_039.jpg

Validation files:
LNEW_003.jpg
L_054.jpg
L_072.jpg
S_011.jpg
S_013.jpg
S_027.jpg
S_035.jpg
S_037.jpg
old_015.jpg
old_032.jpg
old_057.jpg
old_065.jpg
old_068.jpg
old_071.jpg
L_002.jpg
L_003.jpg
L_048.jpg
L_049.jpg
SNEW_002.jpg
SNEW_003.jpg
SNEW_004.jpg
SNEW_015.jpg
SNEW_024.jpg
SNEW_037.jpg
S_044.jpg
old_004.jpg
old_018.jpg
old_031.jpg
```

Accuracy : 0.7857
Precision: 0.8111
Recall : 0.7857
F1-score : 0.7812

```
===== Fold 5 =====

Train files:
LNEW_001.jpg
LNEW_002.jpg
LNEW_003.jpg
LNEW_004.jpg
LNEW_005.jpg
LNEW_007.jpg
LNEW_008.jpg
LNEW_009.jpg
LNEW_010.jpg
LNEW_011.jpg
LNEW_012.jpg
LNEW_013.jpg
L_012.jpg
L_013.jpg
L_015.jpg
L_022.jpg
L_023.jpg
L_024.jpg
L_027.jpg
L_029.jpg
L_030.jpg
L_031.jpg
L_034.jpg
L_036.jpg
L_037.jpg
L_038.jpg
L_040.jpg
L_053.jpg
L_054.jpg
L_055.jpg
L_056.jpg
L_058.jpg
L_060.jpg
L_070.jpg
L_072.jpg
L_074.jpg
L_075.jpg
L_077.jpg
L_081.jpg
L_082.jpg
L_083.jpg
S_001.jpg
S_002.jpg
S_003.jpg
S_004.jpg
S_005.jpg
S_006.jpg
S_007.jpg
S_008.jpg
S_009.jpg
S_010.jpg
S_011.jpg
S_012.jpg
S_013.jpg
S_014.jpg
S_015.jpg
S_016.jpg
S_017.jpg
S_018.jpg
```

S_019.jpg
S_020.jpg
S_024.jpg
S_025.jpg
S_026.jpg
S_027.jpg
S_028.jpg
S_029.jpg
S_031.jpg
S_032.jpg
S_034.jpg
S_035.jpg
S_036.jpg
S_037.jpg
S_038.jpg
S_039.jpg
S_041.jpg
S_042.jpg
S_045.jpg
S_048.jpg
S_050.jpg
S_051.jpg
S_052.jpg
S_057.jpg
S_058.jpg
S_059.jpg
old_001.jpg
old_002.jpg
old_003.jpg
old_004.jpg
old_007.jpg
old_008.jpg
old_009.jpg
old_010.jpg
old_011.jpg
old_013.jpg
old_014.jpg
old_015.jpg
old_016.jpg
old_017.jpg
old_018.jpg
old_019.jpg
old_020.jpg
old_021.jpg
old_032.jpg
old_035.jpg
old_054.jpg
old_057.jpg
old_065.jpg
old_066.jpg
old_068.jpg
old_071.jpg
old_082.jpg
old_083.jpg
old_085.jpg
old_086.jpg
old_089.jpg
old_091.jpg
old_092.jpg
old_093.jpg
old_094.jpg
old_095.jpg
old_096.jpg
old_098.jpg
old_103.jpg
old_104.jpg
old_106.jpg
old_107.jpg

```
old_109.jpg
old_111.jpg
old_112.jpg
L_001.jpg
L_002.jpg
L_003.jpg
L_004.jpg
L_005.jpg
L_006.jpg
L_007.jpg
L_008.jpg
L_009.jpg
L_010.jpg
L_011.jpg
L_016.jpg
L_021.jpg
L_026.jpg
L_028.jpg
L_032.jpg
L_039.jpg
L_041.jpg
L_042.jpg
L_043.jpg
L_044.jpg
L_045.jpg
L_046.jpg
L_047.jpg
L_048.jpg
L_049.jpg
L_050.jpg
L_061.jpg
L_062.jpg
L_064.jpg
L_066.jpg
L_068.jpg
L_069.jpg
L_071.jpg
SNEW_001.jpg
SNEW_002.jpg
SNEW_003.jpg
SNEW_004.jpg
SNEW_007.jpg
SNEW_008.jpg
SNEW_009.jpg
SNEW_010.jpg
SNEW_011.jpg
SNEW_012.jpg
SNEW_013.jpg
SNEW_015.jpg
SNEW_016.jpg
SNEW_017.jpg
SNEW_018.jpg
SNEW_019.jpg
SNEW_020.jpg
SNEW_021.jpg
SNEW_022.jpg
SNEW_023.jpg
SNEW_024.jpg
SNEW_025.jpg
SNEW_026.jpg
SNEW_027.jpg
SNEW_028.jpg
SNEW_030.jpg
SNEW_031.jpg
SNEW_032.jpg
SNEW_033.jpg
SNEW_034.jpg
SNEW_035.jpg
```

```
SNEW_036.jpg
SNEW_037.jpg
SNEW_039.jpg
SNEW_040.jpg
SNEW_041.jpg
SNEW_042.jpg
SNEW_043.jpg
SNEW_044.jpg
SNEW_045.jpg
SNEW_046.jpg
SNEW_047.jpg
SNEW_048.jpg
SNEW_049.jpg
SNEW_050.jpg
SNEW_053.jpg
SNEW_054.jpg
SNEW_055.jpg
SNEW_056.jpg
SNEW_057.jpg
SNEW_058.jpg
SNEW_059.jpg
SNEW_060.jpg
S_030.jpg
S_044.jpg
old_001.jpg
old_002.jpg
old_003.jpg
old_004.jpg
old_005.jpg
old_006.jpg
old_007.jpg
old_008.jpg
old_009.jpg
old_010.jpg
old_012.jpg
old_013.jpg
old_014.jpg
old_015.jpg
old_016.jpg
old_017.jpg
old_018.jpg
old_019.jpg
old_020.jpg
old_021.jpg
old_022.jpg
old_023.jpg
old_024.jpg
old_025.jpg
old_026.jpg
old_027.jpg
old_028.jpg
old_029.jpg
old_030.jpg
old_031.jpg
old_033.jpg
old_034.jpg
old_036.jpg
old_037.jpg
old_039.jpg

Validation files:
LNEW_014.jpg
L_051.jpg
L_052.jpg
L_059.jpg
S_021.jpg
S_033.jpg
S_049.jpg
```

```
old_005.jpg
old_069.jpg
old_090.jpg
old_097.jpg
L_014.jpg
L_025.jpg
L_057.jpg
L_063.jpg
L_065.jpg
L_067.jpg
SNEW_005.jpg
SNEW_006.jpg
SNEW_014.jpg
SNEW_029.jpg
SNEW_038.jpg
SNEW_051.jpg
SNEW_052.jpg
old_011.jpg
old_032.jpg
old_035.jpg
old_038.jpg
Accuracy : 0.5714
Precision: 0.5442
Recall   : 0.5714
F1-score : 0.5464

===== Fold 6 =====

Train files:
LNEW_002.jpg
LNEW_003.jpg
LNEW_004.jpg
LNEW_005.jpg
LNEW_007.jpg
LNEW_008.jpg
LNEW_009.jpg
LNEW_010.jpg
LNEW_011.jpg
LNEW_012.jpg
LNEW_014.jpg
L_012.jpg
L_013.jpg
L_015.jpg
L_022.jpg
L_023.jpg
L_024.jpg
L_027.jpg
L_029.jpg
L_030.jpg
L_031.jpg
L_034.jpg
L_036.jpg
L_037.jpg
L_040.jpg
L_051.jpg
L_052.jpg
L_053.jpg
L_054.jpg
L_055.jpg
L_056.jpg
L_058.jpg
L_059.jpg
L_070.jpg
L_072.jpg
L_074.jpg
L_075.jpg
L_081.jpg
L_082.jpg
```

```
    L_083.jpg
    S_001.jpg
    S_002.jpg
    S_003.jpg
    S_004.jpg
    S_005.jpg
    S_006.jpg
    S_008.jpg
    S_009.jpg
    S_010.jpg
    S_011.jpg
    S_012.jpg
    S_013.jpg
    S_014.jpg
    S_015.jpg
    S_016.jpg
    S_018.jpg
    S_019.jpg
    S_020.jpg
    S_021.jpg
    S_024.jpg
    S_025.jpg
    S_026.jpg
    S_027.jpg
    S_028.jpg
    S_029.jpg
    S_031.jpg
    S_032.jpg
    S_033.jpg
    S_034.jpg
    S_035.jpg
    S_036.jpg
    S_037.jpg
    S_038.jpg
    S_039.jpg
    S_041.jpg
    S_042.jpg
    S_045.jpg
    S_048.jpg
    S_049.jpg
    S_050.jpg
    S_051.jpg
    S_052.jpg
    S_057.jpg
    S_058.jpg
    S_059.jpg
    old_001.jpg
    old_002.jpg
    old_003.jpg
    old_005.jpg
    old_007.jpg
    old_009.jpg
    old_011.jpg
    old_013.jpg
    old_014.jpg
    old_015.jpg
    old_016.jpg
    old_017.jpg
    old_018.jpg
    old_019.jpg
    old_020.jpg
    old_021.jpg
    old_032.jpg
    old_035.jpg
    old_054.jpg
    old_057.jpg
    old_065.jpg
    old_066.jpg
```

```
old_068.jpg
old_069.jpg
old_071.jpg
old_082.jpg
old_083.jpg
old_085.jpg
old_086.jpg
old_089.jpg
old_090.jpg
old_091.jpg
old_092.jpg
old_093.jpg
old_094.jpg
old_095.jpg
old_096.jpg
old_097.jpg
old_098.jpg
old_103.jpg
old_104.jpg
old_107.jpg
old_109.jpg
old_111.jpg
old_112.jpg
L_002.jpg
L_003.jpg
L_005.jpg
L_006.jpg
L_007.jpg
L_008.jpg
L_009.jpg
L_011.jpg
L_014.jpg
L_016.jpg
L_021.jpg
L_025.jpg
L_026.jpg
L_028.jpg
L_032.jpg
L_039.jpg
L_041.jpg
L_042.jpg
L_043.jpg
L_044.jpg
L_045.jpg
L_046.jpg
L_047.jpg
L_048.jpg
L_049.jpg
L_050.jpg
L_057.jpg
L_061.jpg
L_063.jpg
L_064.jpg
L_065.jpg
L_066.jpg
L_067.jpg
L_068.jpg
L_069.jpg
SNEW_001.jpg
SNEW_002.jpg
SNEW_003.jpg
SNEW_004.jpg
SNEW_005.jpg
SNEW_006.jpg
SNEW_007.jpg
SNEW_008.jpg
SNEW_009.jpg
SNEW_010.jpg
```

```
    SNEW_011.jpg
    SNEW_013.jpg
    SNEW_014.jpg
    SNEW_015.jpg
    SNEW_016.jpg
    SNEW_017.jpg
    SNEW_018.jpg
    SNEW_019.jpg
    SNEW_021.jpg
    SNEW_022.jpg
    SNEW_024.jpg
    SNEW_025.jpg
    SNEW_026.jpg
    SNEW_027.jpg
    SNEW_029.jpg
    SNEW_030.jpg
    SNEW_031.jpg
    SNEW_032.jpg
    SNEW_033.jpg
    SNEW_034.jpg
    SNEW_035.jpg
    SNEW_036.jpg
    SNEW_037.jpg
    SNEW_038.jpg
    SNEW_040.jpg
    SNEW_041.jpg
    SNEW_042.jpg
    SNEW_043.jpg
    SNEW_046.jpg
    SNEW_047.jpg
    SNEW_048.jpg
    SNEW_049.jpg
    SNEW_050.jpg
    SNEW_051.jpg
    SNEW_052.jpg
    SNEW_053.jpg
    SNEW_054.jpg
    SNEW_055.jpg
    SNEW_056.jpg
    SNEW_057.jpg
    SNEW_058.jpg
    SNEW_059.jpg
    SNEW_060.jpg
    S_030.jpg
    S_044.jpg
    old_001.jpg
    old_002.jpg
    old_004.jpg
    old_005.jpg
    old_006.jpg
    old_008.jpg
    old_009.jpg
    old_010.jpg
    old_011.jpg
    old_012.jpg
    old_013.jpg
    old_014.jpg
    old_015.jpg
    old_016.jpg
    old_017.jpg
    old_018.jpg
    old_019.jpg
    old_021.jpg
    old_022.jpg
    old_023.jpg
    old_024.jpg
    old_025.jpg
    old_026.jpg
```

```
old_027.jpg
old_028.jpg
old_029.jpg
old_030.jpg
old_031.jpg
old_032.jpg
old_034.jpg
old_035.jpg
old_037.jpg
old_038.jpg
old_039.jpg

Validation files:
LNEW_001.jpg
LNEW_013.jpg
L_038.jpg
L_060.jpg
L_077.jpg
S_007.jpg
S_017.jpg
old_004.jpg
old_008.jpg
old_010.jpg
old_106.jpg
L_001.jpg
L_004.jpg
L_010.jpg
L_062.jpg
L_071.jpg
SNEW_012.jpg
SNEW_020.jpg
SNEW_023.jpg
SNEW_028.jpg
SNEW_039.jpg
SNEW_044.jpg
SNEW_045.jpg
old_003.jpg
old_007.jpg
old_020.jpg
old_033.jpg
old_036.jpg
```

Accuracy : 0.3929
Precision: 0.1543
Recall : 0.3929
F1-score : 0.2216

```
===== Fold 7 =====

Train files:
LNEW_001.jpg
LNEW_002.jpg
LNEW_003.jpg
LNEW_004.jpg
LNEW_007.jpg
LNEW_008.jpg
LNEW_009.jpg
LNEW_010.jpg
LNEW_011.jpg
LNEW_012.jpg
LNEW_013.jpg
LNEW_014.jpg
L_012.jpg
L_013.jpg
L_015.jpg
L_022.jpg
L_023.jpg
L_024.jpg
L_027.jpg
L_029.jpg
L_030.jpg
L_031.jpg
L_036.jpg
L_037.jpg
L_038.jpg
L_051.jpg
L_052.jpg
L_053.jpg
L_054.jpg
L_056.jpg
L_058.jpg
L_059.jpg
L_060.jpg
L_070.jpg
L_072.jpg
L_074.jpg
L_077.jpg
L_081.jpg
L_082.jpg
S_001.jpg
S_002.jpg
S_003.jpg
S_004.jpg
S_005.jpg
S_006.jpg
S_007.jpg
S_008.jpg
S_009.jpg
S_010.jpg
S_011.jpg
S_012.jpg
S_013.jpg
S_014.jpg
S_015.jpg
S_016.jpg
S_017.jpg
S_019.jpg
S_021.jpg
S_024.jpg
```

```
S_025.jpg
S_026.jpg
S_027.jpg
S_029.jpg
S_031.jpg
S_032.jpg
S_033.jpg
S_034.jpg
S_035.jpg
S_036.jpg
S_037.jpg
S_038.jpg
S_039.jpg
S_041.jpg
S_042.jpg
S_045.jpg
S_048.jpg
S_049.jpg
S_050.jpg
S_051.jpg
S_052.jpg
S_057.jpg
S_058.jpg
S_059.jpg
old_001.jpg
old_002.jpg
old_003.jpg
old_004.jpg
old_005.jpg
old_007.jpg
old_008.jpg
old_009.jpg
old_010.jpg
old_011.jpg
old_013.jpg
old_014.jpg
old_015.jpg
old_016.jpg
old_017.jpg
old_018.jpg
old_019.jpg
old_020.jpg
old_021.jpg
old_032.jpg
old_035.jpg
old_054.jpg
old_057.jpg
old_065.jpg
old_066.jpg
old_068.jpg
old_069.jpg
old_071.jpg
old_082.jpg
old_083.jpg
old_089.jpg
old_090.jpg
old_091.jpg
old_092.jpg
old_094.jpg
old_095.jpg
old_096.jpg
old_097.jpg
old_098.jpg
old_103.jpg
old_106.jpg
old_107.jpg
old_111.jpg
old_112.jpg
```

L_001.jpg
L_002.jpg
L_003.jpg
L_004.jpg
L_005.jpg
L_006.jpg
L_007.jpg
L_008.jpg
L_009.jpg
L_010.jpg
L_011.jpg
L_014.jpg
L_021.jpg
L_025.jpg
L_028.jpg
L_032.jpg
L_039.jpg
L_041.jpg
L_042.jpg
L_044.jpg
L_045.jpg
L_046.jpg
L_047.jpg
L_048.jpg
L_049.jpg
L_050.jpg
L_057.jpg
L_061.jpg
L_062.jpg
L_063.jpg
L_064.jpg
L_065.jpg
L_067.jpg
L_068.jpg
L_069.jpg
L_071.jpg
SNEW_002.jpg
SNEW_003.jpg
SNEW_004.jpg
SNEW_005.jpg
SNEW_006.jpg
SNEW_007.jpg
SNEW_008.jpg
SNEW_009.jpg
SNEW_010.jpg
SNEW_011.jpg
SNEW_012.jpg
SNEW_013.jpg
SNEW_014.jpg
SNEW_015.jpg
SNEW_016.jpg
SNEW_017.jpg
SNEW_018.jpg
SNEW_019.jpg
SNEW_020.jpg
SNEW_021.jpg
SNEW_023.jpg
SNEW_024.jpg
SNEW_025.jpg
SNEW_026.jpg
SNEW_027.jpg
SNEW_028.jpg
SNEW_029.jpg
SNEW_030.jpg
SNEW_031.jpg
SNEW_032.jpg
SNEW_033.jpg
SNEW_034.jpg

```
SNEW_035.jpg
SNEW_036.jpg
SNEW_037.jpg
SNEW_038.jpg
SNEW_039.jpg
SNEW_040.jpg
SNEW_041.jpg
SNEW_043.jpg
SNEW_044.jpg
SNEW_045.jpg
SNEW_046.jpg
SNEW_047.jpg
SNEW_048.jpg
SNEW_051.jpg
SNEW_052.jpg
SNEW_053.jpg
SNEW_055.jpg
SNEW_057.jpg
SNEW_058.jpg
SNEW_060.jpg
S_030.jpg
S_044.jpg
old_001.jpg
old_002.jpg
old_003.jpg
old_004.jpg
old_005.jpg
old_006.jpg
old_007.jpg
old_008.jpg
old_009.jpg
old_010.jpg
old_011.jpg
old_012.jpg
old_013.jpg
old_014.jpg
old_015.jpg
old_016.jpg
old_017.jpg
old_018.jpg
old_019.jpg
old_020.jpg
old_021.jpg
old_022.jpg
old_023.jpg
old_025.jpg
old_026.jpg
old_027.jpg
old_028.jpg
old_029.jpg
old_031.jpg
old_032.jpg
old_033.jpg
old_034.jpg
old_035.jpg
old_036.jpg
old_037.jpg
old_038.jpg
old_039.jpg

Validation files:
LNEW_005.jpg
L_034.jpg
L_040.jpg
L_055.jpg
L_075.jpg
L_083.jpg
S_018.jpg
```

```
S_020.jpg
S_028.jpg
old_085.jpg
old_086.jpg
old_093.jpg
old_104.jpg
old_109.jpg
L_016.jpg
L_026.jpg
L_043.jpg
L_066.jpg
SNEW_001.jpg
SNEW_022.jpg
SNEW_042.jpg
SNEW_049.jpg
SNEW_050.jpg
SNEW_054.jpg
SNEW_056.jpg
SNEW_059.jpg
old_024.jpg
old_030.jpg
```

Accuracy : 0.5000
Precision: 0.2500
Recall : 0.5000
F1-score : 0.3333

```
===== Fold 8 =====

Train files:
LNEW_001.jpg
LNEW_003.jpg
LNEW_005.jpg
LNEW_007.jpg
LNEW_008.jpg
LNEW_009.jpg
LNEW_010.jpg
LNEW_011.jpg
LNEW_012.jpg
LNEW_013.jpg
LNEW_014.jpg
L_012.jpg
L_015.jpg
L_022.jpg
L_023.jpg
L_024.jpg
L_027.jpg
L_029.jpg
L_030.jpg
L_031.jpg
L_034.jpg
L_036.jpg
L_037.jpg
L_038.jpg
L_040.jpg
L_051.jpg
L_052.jpg
L_053.jpg
L_054.jpg
L_055.jpg
L_056.jpg
L_059.jpg
L_060.jpg
L_070.jpg
L_072.jpg
L_075.jpg
L_077.jpg
L_081.jpg
L_083.jpg
S_001.jpg
S_002.jpg
S_004.jpg
S_006.jpg
S_007.jpg
S_008.jpg
S_010.jpg
S_011.jpg
S_012.jpg
S_013.jpg
S_014.jpg
S_015.jpg
S_016.jpg
S_017.jpg
S_018.jpg
S_019.jpg
S_020.jpg
S_021.jpg
S_024.jpg
S_025.jpg
```

```
S_026.jpg
S_027.jpg
S_028.jpg
S_029.jpg
S_031.jpg
S_032.jpg
S_033.jpg
S_034.jpg
S_035.jpg
S_036.jpg
S_037.jpg
S_038.jpg
S_042.jpg
S_045.jpg
S_048.jpg
S_049.jpg
S_050.jpg
S_051.jpg
S_052.jpg
S_057.jpg
S_058.jpg
S_059.jpg
old_001.jpg
old_002.jpg
old_004.jpg
old_005.jpg
old_007.jpg
old_008.jpg
old_009.jpg
old_010.jpg
old_011.jpg
old_013.jpg
old_015.jpg
old_017.jpg
old_018.jpg
old_019.jpg
old_020.jpg
old_021.jpg
old_032.jpg
old_035.jpg
old_054.jpg
old_057.jpg
old_065.jpg
old_066.jpg
old_068.jpg
old_069.jpg
old_071.jpg
old_082.jpg
old_083.jpg
old_085.jpg
old_086.jpg
old_089.jpg
old_090.jpg
old_091.jpg
old_092.jpg
old_093.jpg
old_094.jpg
old_095.jpg
old_096.jpg
old_097.jpg
old_103.jpg
old_104.jpg
old_106.jpg
old_107.jpg
old_109.jpg
old_111.jpg
old_112.jpg
L_001.jpg
```

L_002.jpg
L_003.jpg
L_004.jpg
L_006.jpg
L_007.jpg
L_008.jpg
L_009.jpg
L_010.jpg
L_011.jpg
L_014.jpg
L_016.jpg
L_021.jpg
L_025.jpg
L_026.jpg
L_028.jpg
L_032.jpg
L_039.jpg
L_041.jpg
L_043.jpg
L_045.jpg
L_046.jpg
L_047.jpg
L_048.jpg
L_049.jpg
L_050.jpg
L_057.jpg
L_061.jpg
L_062.jpg
L_063.jpg
L_064.jpg
L_065.jpg
L_066.jpg
L_067.jpg
L_068.jpg
L_069.jpg
L_071.jpg
SNEW_001.jpg
SNEW_002.jpg
SNEW_003.jpg
SNEW_004.jpg
SNEW_005.jpg
SNEW_006.jpg
SNEW_007.jpg
SNEW_008.jpg
SNEW_009.jpg
SNEW_011.jpg
SNEW_012.jpg
SNEW_013.jpg
SNEW_014.jpg
SNEW_015.jpg
SNEW_016.jpg
SNEW_017.jpg
SNEW_018.jpg
SNEW_020.jpg
SNEW_022.jpg
SNEW_023.jpg
SNEW_024.jpg
SNEW_027.jpg
SNEW_028.jpg
SNEW_029.jpg
SNEW_030.jpg
SNEW_031.jpg
SNEW_032.jpg
SNEW_033.jpg
SNEW_034.jpg
SNEW_036.jpg
SNEW_037.jpg
SNEW_038.jpg

```
SNEW_039.jpg
SNEW_040.jpg
SNEW_041.jpg
SNEW_042.jpg
SNEW_043.jpg
SNEW_044.jpg
SNEW_045.jpg
SNEW_046.jpg
SNEW_047.jpg
SNEW_048.jpg
SNEW_049.jpg
SNEW_050.jpg
SNEW_051.jpg
SNEW_052.jpg
SNEW_053.jpg
SNEW_054.jpg
SNEW_055.jpg
SNEW_056.jpg
SNEW_058.jpg
SNEW_059.jpg
S_030.jpg
S_044.jpg
old_001.jpg
old_002.jpg
old_003.jpg
old_004.jpg
old_005.jpg
old_006.jpg
old_007.jpg
old_008.jpg
old_009.jpg
old_010.jpg
old_011.jpg
old_012.jpg
old_015.jpg
old_016.jpg
old_017.jpg
old_018.jpg
old_019.jpg
old_020.jpg
old_021.jpg
old_022.jpg
old_023.jpg
old_024.jpg
old_025.jpg
old_026.jpg
old_027.jpg
old_028.jpg
old_029.jpg
old_030.jpg
old_031.jpg
old_032.jpg
old_033.jpg
old_034.jpg
old_035.jpg
old_036.jpg
old_037.jpg
old_038.jpg
old_039.jpg

Validation files:
LNEW_002.jpg
LNEW_004.jpg
L_013.jpg
L_058.jpg
L_074.jpg
L_082.jpg
S_003.jpg
```

```
S_005.jpg
S_009.jpg
S_039.jpg
S_041.jpg
old_003.jpg
old_014.jpg
old_016.jpg
old_098.jpg
L_005.jpg
L_042.jpg
L_044.jpg
SNEW_010.jpg
SNEW_019.jpg
SNEW_021.jpg
SNEW_025.jpg
SNEW_026.jpg
SNEW_035.jpg
SNEW_057.jpg
SNEW_060.jpg
old_013.jpg
old_014.jpg
Accuracy : 0.5357
Precision: 0.7679
Recall   : 0.5357
F1-score : 0.4356

===== Fold 9 =====

Train files:
LNEW_001.jpg
LNEW_002.jpg
LNEW_003.jpg
LNEW_004.jpg
LNEW_005.jpg
LNEW_007.jpg
LNEW_008.jpg
LNEW_011.jpg
LNEW_012.jpg
LNEW_013.jpg
LNEW_014.jpg
L_013.jpg
L_015.jpg
L_022.jpg
L_024.jpg
L_027.jpg
L_029.jpg
L_030.jpg
L_031.jpg
L_034.jpg
L_036.jpg
L_037.jpg
L_038.jpg
L_040.jpg
L_051.jpg
L_052.jpg
L_053.jpg
L_054.jpg
L_055.jpg
L_056.jpg
L_058.jpg
L_059.jpg
L_060.jpg
L_070.jpg
L_072.jpg
L_074.jpg
L_075.jpg
L_077.jpg
L_081.jpg
```

```
    L_082.jpg
    L_083.jpg
    S_001.jpg
    S_002.jpg
    S_003.jpg
    S_004.jpg
    S_005.jpg
    S_006.jpg
    S_007.jpg
    S_009.jpg
    S_011.jpg
    S_012.jpg
    S_013.jpg
    S_014.jpg
    S_016.jpg
    S_017.jpg
    S_018.jpg
    S_020.jpg
    S_021.jpg
    S_024.jpg
    S_025.jpg
    S_026.jpg
    S_027.jpg
    S_028.jpg
    S_029.jpg
    S_032.jpg
    S_033.jpg
    S_034.jpg
    S_035.jpg
    S_036.jpg
    S_037.jpg
    S_038.jpg
    S_039.jpg
    S_041.jpg
    S_042.jpg
    S_048.jpg
    S_049.jpg
    S_050.jpg
    S_051.jpg
    S_058.jpg
    old_001.jpg
    old_002.jpg
    old_003.jpg
    old_004.jpg
    old_005.jpg
    old_007.jpg
    old_008.jpg
    old_009.jpg
    old_010.jpg
    old_011.jpg
    old_013.jpg
    old_014.jpg
    old_015.jpg
    old_016.jpg
    old_017.jpg
    old_019.jpg
    old_020.jpg
    old_032.jpg
    old_035.jpg
    old_054.jpg
    old_057.jpg
    old_065.jpg
    old_066.jpg
    old_068.jpg
    old_069.jpg
    old_071.jpg
    old_082.jpg
    old_083.jpg
```

```
old_085.jpg
old_086.jpg
old_089.jpg
old_090.jpg
old_091.jpg
old_092.jpg
old_093.jpg
old_095.jpg
old_097.jpg
old_098.jpg
old_104.jpg
old_106.jpg
old_107.jpg
old_109.jpg
old_111.jpg
old_112.jpg
L_001.jpg
L_002.jpg
L_003.jpg
L_004.jpg
L_005.jpg
L_006.jpg
L_007.jpg
L_008.jpg
L_009.jpg
L_010.jpg
L_011.jpg
L_014.jpg
L_016.jpg
L_021.jpg
L_025.jpg
L_026.jpg
L_028.jpg
L_032.jpg
L_039.jpg
L_041.jpg
L_042.jpg
L_043.jpg
L_044.jpg
L_045.jpg
L_046.jpg
L_048.jpg
L_049.jpg
L_050.jpg
L_057.jpg
L_062.jpg
L_063.jpg
L_064.jpg
L_065.jpg
L_066.jpg
L_067.jpg
L_068.jpg
L_069.jpg
L_071.jpg
SNEW_001.jpg
SNEW_002.jpg
SNEW_003.jpg
SNEW_004.jpg
SNEW_005.jpg
SNEW_006.jpg
SNEW_007.jpg
SNEW_008.jpg
SNEW_009.jpg
SNEW_010.jpg
SNEW_011.jpg
SNEW_012.jpg
SNEW_013.jpg
SNEW_014.jpg
```

```
    SNEW_015.jpg
    SNEW_016.jpg
    SNEW_017.jpg
    SNEW_019.jpg
    SNEW_020.jpg
    SNEW_021.jpg
    SNEW_022.jpg
    SNEW_023.jpg
    SNEW_024.jpg
    SNEW_025.jpg
    SNEW_026.jpg
    SNEW_027.jpg
    SNEW_028.jpg
    SNEW_029.jpg
    SNEW_030.jpg
    SNEW_031.jpg
    SNEW_032.jpg
    SNEW_033.jpg
    SNEW_034.jpg
    SNEW_035.jpg
    SNEW_036.jpg
    SNEW_037.jpg
    SNEW_038.jpg
    SNEW_039.jpg
    SNEW_040.jpg
    SNEW_041.jpg
    SNEW_042.jpg
    SNEW_043.jpg
    SNEW_044.jpg
    SNEW_045.jpg
    SNEW_046.jpg
    SNEW_047.jpg
    SNEW_049.jpg
    SNEW_050.jpg
    SNEW_051.jpg
    SNEW_052.jpg
    SNEW_053.jpg
    SNEW_054.jpg
    SNEW_055.jpg
    SNEW_056.jpg
    SNEW_057.jpg
    SNEW_058.jpg
    SNEW_059.jpg
    SNEW_060.jpg
    S_030.jpg
    S_044.jpg
    old_002.jpg
    old_003.jpg
    old_004.jpg
    old_005.jpg
    old_007.jpg
    old_008.jpg
    old_009.jpg
    old_010.jpg
    old_011.jpg
    old_013.jpg
    old_014.jpg
    old_015.jpg
    old_017.jpg
    old_018.jpg
    old_019.jpg
    old_020.jpg
    old_021.jpg
    old_022.jpg
    old_023.jpg
    old_024.jpg
    old_025.jpg
    old_026.jpg
```

```
old_028.jpg
old_029.jpg
old_030.jpg
old_031.jpg
old_032.jpg
old_033.jpg
old_035.jpg
old_036.jpg
old_037.jpg
old_038.jpg
old_039.jpg

Validation files:
LNEW_009.jpg
LNEW_010.jpg
L_012.jpg
L_023.jpg
S_008.jpg
S_010.jpg
S_015.jpg
S_019.jpg
S_031.jpg
S_045.jpg
S_052.jpg
S_057.jpg
S_059.jpg
old_018.jpg
old_021.jpg
old_094.jpg
old_096.jpg
old_103.jpg
L_047.jpg
L_061.jpg
SNEW_018.jpg
SNEW_048.jpg
old_001.jpg
old_006.jpg
old_012.jpg
old_016.jpg
old_027.jpg
old_034.jpg
```

Accuracy : 0.6429
Precision: 0.4133
Recall : 0.6429
F1-score : 0.5031

```
===== Fold 10 =====

Train files:
LNEW_001.jpg
LNEW_002.jpg
LNEW_003.jpg
LNEW_004.jpg
LNEW_005.jpg
LNEW_007.jpg
LNEW_008.jpg
LNEW_009.jpg
LNEW_010.jpg
LNEW_011.jpg
LNEW_012.jpg
LNEW_013.jpg
LNEW_014.jpg
L_012.jpg
L_013.jpg
L_015.jpg
L_022.jpg
L_023.jpg
L_024.jpg
L_027.jpg
L_031.jpg
L_034.jpg
L_036.jpg
L_037.jpg
L_038.jpg
L_040.jpg
L_051.jpg
L_052.jpg
L_053.jpg
L_054.jpg
L_055.jpg
L_056.jpg
L_058.jpg
L_059.jpg
L_060.jpg
L_070.jpg
L_072.jpg
L_074.jpg
L_075.jpg
L_077.jpg
L_081.jpg
L_082.jpg
L_083.jpg
S_001.jpg
S_002.jpg
S_003.jpg
S_005.jpg
S_007.jpg
S_008.jpg
S_009.jpg
S_010.jpg
S_011.jpg
S_012.jpg
S_013.jpg
S_015.jpg
S_016.jpg
S_017.jpg
S_018.jpg
S_019.jpg
```

S_020.jpg
S_021.jpg
S_024.jpg
S_025.jpg
S_026.jpg
S_027.jpg
S_028.jpg
S_031.jpg
S_032.jpg
S_033.jpg
S_034.jpg
S_035.jpg
S_036.jpg
S_037.jpg
S_038.jpg
S_039.jpg
S_041.jpg
S_042.jpg
S_045.jpg
S_048.jpg
S_049.jpg
S_050.jpg
S_052.jpg
S_057.jpg
S_058.jpg
S_059.jpg
old_001.jpg
old_002.jpg
old_003.jpg
old_004.jpg
old_005.jpg
old_007.jpg
old_008.jpg
old_010.jpg
old_011.jpg
old_014.jpg
old_015.jpg
old_016.jpg
old_018.jpg
old_019.jpg
old_020.jpg
old_021.jpg
old_032.jpg
old_035.jpg
old_054.jpg
old_057.jpg
old_065.jpg
old_066.jpg
old_068.jpg
old_069.jpg
old_071.jpg
old_082.jpg
old_085.jpg
old_086.jpg
old_089.jpg
old_090.jpg
old_091.jpg
old_092.jpg
old_093.jpg
old_094.jpg
old_096.jpg
old_097.jpg
old_098.jpg
old_103.jpg
old_104.jpg
old_106.jpg
old_107.jpg
old_109.jpg

```
old_111.jpg
old_112.jpg
L_001.jpg
L_002.jpg
L_003.jpg
L_004.jpg
L_005.jpg
L_006.jpg
L_007.jpg
L_008.jpg
L_010.jpg
L_014.jpg
L_016.jpg
L_021.jpg
L_025.jpg
L_026.jpg
L_028.jpg
L_032.jpg
L_039.jpg
L_042.jpg
L_043.jpg
L_044.jpg
L_045.jpg
L_046.jpg
L_047.jpg
L_048.jpg
L_049.jpg
L_057.jpg
L_061.jpg
L_062.jpg
L_063.jpg
L_065.jpg
L_066.jpg
L_067.jpg
L_068.jpg
L_069.jpg
L_071.jpg
SNEW_001.jpg
SNEW_002.jpg
SNEW_003.jpg
SNEW_004.jpg
SNEW_005.jpg
SNEW_006.jpg
SNEW_010.jpg
SNEW_012.jpg
SNEW_013.jpg
SNEW_014.jpg
SNEW_015.jpg
SNEW_016.jpg
SNEW_017.jpg
SNEW_018.jpg
SNEW_019.jpg
SNEW_020.jpg
SNEW_021.jpg
SNEW_022.jpg
SNEW_023.jpg
SNEW_024.jpg
SNEW_025.jpg
SNEW_026.jpg
SNEW_027.jpg
SNEW_028.jpg
SNEW_029.jpg
SNEW_030.jpg
SNEW_031.jpg
SNEW_032.jpg
SNEW_033.jpg
SNEW_035.jpg
SNEW_036.jpg
```

```
SNEW_037.jpg
SNEW_038.jpg
SNEW_039.jpg
SNEW_040.jpg
SNEW_041.jpg
SNEW_042.jpg
SNEW_043.jpg
SNEW_044.jpg
SNEW_045.jpg
SNEW_046.jpg
SNEW_047.jpg
SNEW_048.jpg
SNEW_049.jpg
SNEW_050.jpg
SNEW_051.jpg
SNEW_052.jpg
SNEW_053.jpg
SNEW_054.jpg
SNEW_056.jpg
SNEW_057.jpg
SNEW_058.jpg
SNEW_059.jpg
SNEW_060.jpg
S_030.jpg
S_044.jpg
old_001.jpg
old_002.jpg
old_003.jpg
old_004.jpg
old_005.jpg
old_006.jpg
old_007.jpg
old_008.jpg
old_009.jpg
old_011.jpg
old_012.jpg
old_013.jpg
old_014.jpg
old_016.jpg
old_017.jpg
old_018.jpg
old_019.jpg
old_020.jpg
old_021.jpg
old_022.jpg
old_023.jpg
old_024.jpg
old_026.jpg
old_027.jpg
old_029.jpg
old_030.jpg
old_031.jpg
old_032.jpg
old_033.jpg
old_034.jpg
old_035.jpg
old_036.jpg
old_037.jpg
old_038.jpg

Validation files:
L_029.jpg
L_030.jpg
S_004.jpg
S_006.jpg
S_014.jpg
S_029.jpg
S_051.jpg
```

```
old_009.jpg
old_013.jpg
old_017.jpg
old_083.jpg
old_095.jpg
L_009.jpg
L_011.jpg
L_041.jpg
L_050.jpg
L_064.jpg
SNEW_007.jpg
SNEW_008.jpg
SNEW_009.jpg
SNEW_011.jpg
SNEW_034.jpg
SNEW_055.jpg
old_010.jpg
old_015.jpg
old_025.jpg
old_028.jpg
old_039.jpg
Accuracy : 0.6786
Precision: 0.7561
Recall   : 0.6786
F1-score : 0.6699

===== K-FOLD SUMMARY =====
Accuracy -> Mean: 0.5749, Std: 0.1208
Precision -> Mean: 0.5198, Std: 0.2522
Recall -> Mean: 0.5749, Std: 0.1208
F1 -> Mean: 0.4870, Std: 0.1781
```

Conclusions/action items:

In conclusion, these images did not perform as well as we had hoped for our new model. The team will need to look into other ways to augment our model, but will continue to train a model on these images. We are hoping for accuracy above 80% and F1 of 0.8.

2/18/2026 - K-Fold Cross Validation for Original Model

Madison Michels - Feb 23, 2026, 11:00 AM CST

Title: K-Fold Cross Validation for Original Model

Date: 2/18/2026

Content by: Maddie

Present: Maddie

Goals: The goal of this code is to perform k-fold cross validation with k=5 on our augmented images. We hope to obtain an accuracy, recall, precision, and F1 score from each fold and average them at the end.

Content:

```
===== Fold 1 =====

Train files:
1.png
10.jpeg
11.jpeg
12.jpeg
13.png
15.jpeg
16.jpeg
17.jpeg
19.png
2.png
20.jpeg
22.png
23.png
25.jpeg
27.png
28.png
29.jpeg
3.png
31.png
36.png
38.png
39.png
4.png
40.png
41.png
43.png
44.png
45.png
46.png
48.png
49.jpeg
50.jpeg
51.jpeg
52.jpeg
57.jpeg
58.png
6.png
8.png
Copy of 1.png
Copy of 10.jpeg
Copy of 19.png
Copy of 20.jpeg
Copy of 35.png
Copy of 42.png
Copy of 44.png
Copy of 45.png
Copy of 47.png
Copy of 51.jpeg
Copy of 52.jpeg
Copy of 54.jpeg
Copy of 58.png
Copy of 6.png
Kate_23L.JPG
Kate_25L.JPG
Kate_34L.JPG
Kate_3_loose.JPG
Kate_43L.JPG
Kate_45L.JPG
Kate_46L.JPG
Kate_47L.JPG
Kate_48L.JPG
Kate_4_loose.JPG
Kate_53L.JPG
Kate_54L.JPG
```

```
Kate_55L.JPG
Kate_56L.JPG
Kate_57L.JPG
Kate_5L.JPG
Kate_62L.JPG
Kate_64L.JPG
Kate_7L.JPG
1.png
11.png
112.jpeg
113.jpeg
114.jpeg
115.jpeg
116.jpeg
117.jpeg
118.jpeg
119.jpeg
12.png
120.jpeg
121.jpeg
123(1).jpeg
123.jpeg
125.jpeg
126.jpeg
127.jpeg
128.jpeg
129.jpeg
130.jpeg
131.jpeg
132.jpeg
133.jpeg
134.jpeg
136.jpeg
138.jpeg
139.jpeg
142.jpeg
144.jpeg
150.jpeg
16.png
163.png
17.png
19.png
20.png
21.png
24.png
25.png
26.png
27.png
28.png
32.png
42.png
44.png
45.png
46.png
48.png
53.png
54.png
58.png
60.png
61.png
63.png
65.png
66.png
67.png
7.png
71.png
72.png
84.jpeg
```

88.jpeg
9.png
95.jpeg
Copy of 10.png
Copy of 101.jpeg
Copy of 103.jpeg
Copy of 111.jpeg
Copy of 112.jpeg
Copy of 12.png
Copy of 122.jpeg
Copy of 13.png
Copy of 135.jpeg
Copy of 137.jpeg
Copy of 14.png
Copy of 140.jpeg
Copy of 15.png
Copy of 155.png
Copy of 156.png
Copy of 157.png
Copy of 159.jpeg
Copy of 160.jpeg
Copy of 162.jpeg
Copy of 163.png
Copy of 18.png
Copy of 2.png
Copy of 22.png
Copy of 23.png
Copy of 27.png
Copy of 29.png
Copy of 3.png
Copy of 30.png
Copy of 31.png
Copy of 37.png
Copy of 4.png
Copy of 41.png
Copy of 43.png
Copy of 45.png
Copy of 46.png
Copy of 47.png
Copy of 49.png
Copy of 5.png
Copy of 50.png
Copy of 51.png
Copy of 52.png
Copy of 56.png
Copy of 57.png
Copy of 59.png
Copy of 62.png
Copy of 64.png
Copy of 65.png
Copy of 69.png
Copy of 73.png
Copy of 74.png
Copy of 75.png
Copy of 76.png
Copy of 8.png
Copy of 82.jpeg
Copy of 86.jpeg
Copy of 87.jpeg
Copy of 88.jpeg
Copy of 96.jpeg
Copy of 99.jpeg
Kate_10T.JPG
Kate_11T.JPG
Kate_12T.JPG
Kate_13T.JPG
Kate_14T.JPG
Kate_15T.JPG

```
Kate_16T.JPG
Kate_17T.JPG
Kate_19T.JPG
Kate_1T.JPG
Kate_20T.JPG
Kate_21T.JPG
Kate_22T.JPG
Kate_24T.JPG
Kate_28T.JPG
Kate_2T.JPG
Kate_30T.JPG
Kate_31T.JPG
Kate_32T.JPG
Kate_38T.JPG
Kate_39T.JPG
Kate_40T.JPG
Kate_41T.JPG
Kate_42T.JPG
Kate_43T.JPG
Kate_44T.JPG
Kate_49T.JPG
Kate_50T.JPG
Kate_58T.JPG
Kate_59T.JPG
Kate_61T.JPG
Kate_63T.JPG
Kate_6T.JPG
Kate_74T.JPG
Kate_75T.JPG
Kate_78T.JPG
Kate_79T.JPG
Kate_80T.JPG
Kate_81T.JPG
Kate_82T.JPG
Kate_83T.JPG
Kate_84T.JPG
Kate_86T.JPG
Kate_87T.JPG
Kate_8T.JPG
Screenshot 2025-11-12 124316.png
Screenshot 2025-11-12 225851.png
Screenshot 2025-11-12 225920.png
Screenshot 2025-11-12 225944.png
Screenshot 2025-11-12 225958.png
Screenshot 2025-11-12 230011.png
Screenshot 2025-11-12 230055.png
Screenshot 2025-11-12 230120.png
Screenshot 2025-11-12 230154.png
Screenshot 2025-11-12 230409.png
Screenshot 2025-11-12 at 11.59.32 AM.jpg
Screenshot 2025-11-12 at 11.59.37 AM.jpg
Screenshot 2025-11-12 at 11.59.43 AM.jpg
Screenshot 2025-11-12 at 12.00.21 PM.jpg
Screenshot 2025-11-12 at 12.00.29 PM.jpg
Screenshot 2025-11-12 at 12.00.43 PM.jpg
Screenshot 2025-11-12 at 12.09.16 PM.jpg
Screenshot 2025-11-12 at 12.09.22 PM.jpg
Screenshot 2025-11-12 at 12.09.36 PM.jpg

Validation files:
14.jpeg
18.png
5.png
Copy of 31.png
IMG_2688.PNG
Kate_60L.JPG
13.png
145.jpeg
```

```
146.jpeg
156.png
161.jpeg
68.png
74.png
76.png
85.jpeg
Copy of 102.jpeg
Copy of 104.jpeg
Copy of 110.jpeg
Copy of 6.png
Copy of 83.jpeg
Copy of 97.jpeg
Kate_18T.JPG
Kate_27T.JPG
Kate_33T.JPG
Kate_35T.JPG
Kate_76T.JPG
Kate_77T.JPG
Kate_9T.JPG
Screenshot 2025-11-12 225907.png
Accuracy : 0.2414
Precision: 0.5685
Recall   : 0.2414
F1-score : 0.1867

===== Fold 2 =====

Train files:
1.png
10.jpeg
11.jpeg
12.jpeg
13.png
14.jpeg
16.jpeg
17.jpeg
18.png
19.png
2.png
20.jpeg
22.png
23.png
25.jpeg
27.png
28.png
29.jpeg
3.png
31.png
36.png
39.png
40.png
41.png
43.png
44.png
45.png
48.png
49.jpeg
5.png
50.jpeg
51.jpeg
52.jpeg
57.jpeg
58.png
6.png
8.png
Copy of 1.png
Copy of 19.png
```

```
Copy of 20.jpeg
Copy of 31.png
Copy of 42.png
Copy of 44.png
Copy of 45.png
Copy of 47.png
Copy of 51.jpeg
Copy of 52.jpeg
Copy of 54.jpeg
Copy of 58.png
Copy of 6.png
IMG_2688.PNG
Kate_23L.JPG
Kate_25L.JPG
Kate_34L.JPG
Kate_43L.JPG
Kate_45L.JPG
Kate_47L.JPG
Kate_48L.JPG
Kate_4_loose.JPG
Kate_53L.JPG
Kate_54L.JPG
Kate_55L.JPG
Kate_56L.JPG
Kate_57L.JPG
Kate_5L.JPG
Kate_60L.JPG
Kate_62L.JPG
Kate_7L.JPG
11.png
112.jpeg
113.jpeg
114.jpeg
116.jpeg
118.jpeg
119.jpeg
12.png
120.jpeg
121.jpeg
123(1).jpeg
123.jpeg
125.jpeg
127.jpeg
128.jpeg
129.jpeg
13.png
130.jpeg
131.jpeg
132.jpeg
133.jpeg
134.jpeg
136.jpeg
138.jpeg
139.jpeg
142.jpeg
144.jpeg
145.jpeg
146.jpeg
150.jpeg
156.png
161.jpeg
163.png
17.png
19.png
20.png
24.png
25.png
26.png
```

```
27.png
28.png
32.png
42.png
44.png
45.png
46.png
48.png
53.png
54.png
58.png
60.png
61.png
63.png
65.png
66.png
67.png
68.png
7.png
71.png
72.png
74.png
76.png
84.jpeg
85.jpeg
95.jpeg
Copy of 10.png
Copy of 101.jpeg
Copy of 102.jpeg
Copy of 103.jpeg
Copy of 104.jpeg
Copy of 110.jpeg
Copy of 111.jpeg
Copy of 112.jpeg
Copy of 12.png
Copy of 122.jpeg
Copy of 13.png
Copy of 135.jpeg
Copy of 137.jpeg
Copy of 14.png
Copy of 155.png
Copy of 156.png
Copy of 157.png
Copy of 159.jpeg
Copy of 160.jpeg
Copy of 162.jpeg
Copy of 2.png
Copy of 22.png
Copy of 27.png
Copy of 3.png
Copy of 30.png
Copy of 31.png
Copy of 37.png
Copy of 4.png
Copy of 43.png
Copy of 45.png
Copy of 46.png
Copy of 47.png
Copy of 49.png
Copy of 5.png
Copy of 50.png
Copy of 51.png
Copy of 52.png
Copy of 57.png
Copy of 59.png
Copy of 6.png
Copy of 62.png
Copy of 64.png
```

Copy of 65.png
Copy of 69.png
Copy of 73.png
Copy of 74.png
Copy of 75.png
Copy of 76.png
Copy of 82.jpeg
Copy of 83.jpeg
Copy of 86.jpeg
Copy of 87.jpeg
Copy of 88.jpeg
Copy of 97.jpeg
Copy of 99.jpeg
Kate_10T.JPG
Kate_11T.JPG
Kate_12T.JPG
Kate_13T.JPG
Kate_14T.JPG
Kate_15T.JPG
Kate_16T.JPG
Kate_17T.JPG
Kate_18T.JPG
Kate_19T.JPG
Kate_1T.JPG
Kate_20T.JPG
Kate_21T.JPG
Kate_22T.JPG
Kate_24T.JPG
Kate_27T.JPG
Kate_28T.JPG
Kate_2T.JPG
Kate_30T.JPG
Kate_31T.JPG
Kate_32T.JPG
Kate_33T.JPG
Kate_35T.JPG
Kate_38T.JPG
Kate_39T.JPG
Kate_40T.JPG
Kate_41T.JPG
Kate_42T.JPG
Kate_43T.JPG
Kate_44T.JPG
Kate_49T.JPG
Kate_50T.JPG
Kate_58T.JPG
Kate_59T.JPG
Kate_61T.JPG
Kate_63T.JPG
Kate_6T.JPG
Kate_74T.JPG
Kate_76T.JPG
Kate_77T.JPG
Kate_78T.JPG
Kate_79T.JPG
Kate_80T.JPG
Kate_81T.JPG
Kate_82T.JPG
Kate_83T.JPG
Kate_84T.JPG
Kate_86T.JPG
Kate_87T.JPG
Kate_8T.JPG
Kate_9T.JPG
Screenshot 2025-11-12 124316.png
Screenshot 2025-11-12 225907.png
Screenshot 2025-11-12 225920.png
Screenshot 2025-11-12 225944.png

```
Screenshot 2025-11-12 225958.png
Screenshot 2025-11-12 230011.png
Screenshot 2025-11-12 230055.png
Screenshot 2025-11-12 230120.png
Screenshot 2025-11-12 230154.png
Screenshot 2025-11-12 230409.png
Screenshot 2025-11-12 at 11.59.32 AM.jpg
Screenshot 2025-11-12 at 11.59.37 AM.jpg
Screenshot 2025-11-12 at 11.59.43 AM.jpg
Screenshot 2025-11-12 at 12.00.21 PM.jpg
Screenshot 2025-11-12 at 12.00.29 PM.jpg
Screenshot 2025-11-12 at 12.00.43 PM.jpg
Screenshot 2025-11-12 at 12.09.16 PM.jpg
Screenshot 2025-11-12 at 12.09.22 PM.jpg
Screenshot 2025-11-12 at 12.09.36 PM.jpg

Validation files:
15.jpeg
38.png
4.png
46.png
Copy of 10.jpeg
Copy of 35.png
Kate_3_loose.JPG
Kate_46L.JPG
Kate_64L.JPG
1.png
115.jpeg
117.jpeg
126.jpeg
16.png
21.png
88.jpeg
9.png
Copy of 140.jpeg
Copy of 15.png
Copy of 163.png
Copy of 18.png
Copy of 23.png
Copy of 29.png
Copy of 41.png
Copy of 56.png
Copy of 8.png
Copy of 96.jpeg
Kate_75T.JPG
Screenshot 2025-11-12 225851.png
Accuracy : 0.7241
Precision: 0.7109
Recall   : 0.7241
F1-score : 0.6732

===== Fold 3 =====

Train files:
1.png
10.jpeg
11.jpeg
12.jpeg
13.png
14.jpeg
15.jpeg
16.jpeg
17.jpeg
18.png
2.png
20.jpeg
22.png
23.png
```

28.png
31.png
36.png
38.png
39.png
4.png
41.png
43.png
44.png
45.png
46.png
48.png
49.jpeg
5.png
50.jpeg
51.jpeg
52.jpeg
58.png
6.png
8.png
Copy of 1.png
Copy of 10.jpeg
Copy of 19.png
Copy of 20.jpeg
Copy of 31.png
Copy of 35.png
Copy of 42.png
Copy of 44.png
Copy of 45.png
Copy of 47.png
Copy of 51.jpeg
Copy of 52.jpeg
Copy of 54.jpeg
Copy of 58.png
Copy of 6.png
IMG_2688.PNG
Kate_23L.JPG
Kate_25L.JPG
Kate_34L.JPG
Kate_3_loose.JPG
Kate_43L.JPG
Kate_45L.JPG
Kate_46L.JPG
Kate_47L.JPG
Kate_48L.JPG
Kate_55L.JPG
Kate_56L.JPG
Kate_57L.JPG
Kate_5L.JPG
Kate_60L.JPG
Kate_62L.JPG
Kate_64L.JPG
Kate_7L.JPG
1.png
11.png
113.jpeg
114.jpeg
115.jpeg
116.jpeg
117.jpeg
118.jpeg
12.png
120.jpeg
121.jpeg
123.jpeg
125.jpeg
126.jpeg
127.jpeg

```
128.jpeg
129.jpeg
13.png
130.jpeg
131.jpeg
132.jpeg
134.jpeg
136.jpeg
138.jpeg
139.jpeg
142.jpeg
144.jpeg
145.jpeg
146.jpeg
150.jpeg
156.png
16.png
161.jpeg
17.png
19.png
21.png
24.png
25.png
26.png
27.png
28.png
32.png
42.png
44.png
45.png
46.png
48.png
53.png
54.png
60.png
61.png
63.png
65.png
66.png
67.png
68.png
71.png
72.png
74.png
76.png
84.jpeg
85.jpeg
88.jpeg
9.png
95.jpeg
Copy of 10.png
Copy of 101.jpeg
Copy of 102.jpeg
Copy of 103.jpeg
Copy of 104.jpeg
Copy of 110.jpeg
Copy of 111.jpeg
Copy of 112.jpeg
Copy of 12.png
Copy of 122.jpeg
Copy of 13.png
Copy of 135.jpeg
Copy of 137.jpeg
Copy of 14.png
Copy of 140.jpeg
Copy of 15.png
Copy of 155.png
Copy of 156.png
```

```
Copy of 157.png
Copy of 159.jpeg
Copy of 160.jpeg
Copy of 162.jpeg
Copy of 163.png
Copy of 18.png
Copy of 2.png
Copy of 22.png
Copy of 23.png
Copy of 27.png
Copy of 29.png
Copy of 3.png
Copy of 30.png
Copy of 31.png
Copy of 37.png
Copy of 4.png
Copy of 41.png
Copy of 43.png
Copy of 45.png
Copy of 46.png
Copy of 47.png
Copy of 49.png
Copy of 5.png
Copy of 50.png
Copy of 51.png
Copy of 52.png
Copy of 56.png
Copy of 57.png
Copy of 59.png
Copy of 6.png
Copy of 62.png
Copy of 64.png
Copy of 65.png
Copy of 69.png
Copy of 73.png
Copy of 75.png
Copy of 76.png
Copy of 8.png
Copy of 83.jpeg
Copy of 86.jpeg
Copy of 87.jpeg
Copy of 88.jpeg
Copy of 96.jpeg
Copy of 97.jpeg
Copy of 99.jpeg
Kate_11T.JPG
Kate_12T.JPG
Kate_13T.JPG
Kate_15T.JPG
Kate_16T.JPG
Kate_18T.JPG
Kate_19T.JPG
Kate_1T.JPG
Kate_20T.JPG
Kate_21T.JPG
Kate_22T.JPG
Kate_24T.JPG
Kate_27T.JPG
Kate_2T.JPG
Kate_30T.JPG
Kate_31T.JPG
Kate_32T.JPG
Kate_33T.JPG
Kate_35T.JPG
Kate_38T.JPG
Kate_39T.JPG
Kate_40T.JPG
Kate_41T.JPG
```

```
Kate_42T.JPG
Kate_43T.JPG
Kate_44T.JPG
Kate_49T.JPG
Kate_50T.JPG
Kate_58T.JPG
Kate_59T.JPG
Kate_63T.JPG
Kate_6T.JPG
Kate_74T.JPG
Kate_75T.JPG
Kate_76T.JPG
Kate_77T.JPG
Kate_78T.JPG
Kate_79T.JPG
Kate_82T.JPG
Kate_83T.JPG
Kate_84T.JPG
Kate_86T.JPG
Kate_87T.JPG
Kate_8T.JPG
Kate_9T.JPG
Screenshot 2025-11-12 124316.png
Screenshot 2025-11-12 225851.png
Screenshot 2025-11-12 225907.png
Screenshot 2025-11-12 225920.png
Screenshot 2025-11-12 225944.png
Screenshot 2025-11-12 225958.png
Screenshot 2025-11-12 230011.png
Screenshot 2025-11-12 230055.png
Screenshot 2025-11-12 230120.png
Screenshot 2025-11-12 230409.png
Screenshot 2025-11-12 at 11.59.32 AM.jpg
Screenshot 2025-11-12 at 11.59.37 AM.jpg
Screenshot 2025-11-12 at 11.59.43 AM.jpg
Screenshot 2025-11-12 at 12.00.21 PM.jpg
Screenshot 2025-11-12 at 12.00.29 PM.jpg
Screenshot 2025-11-12 at 12.00.43 PM.jpg
Screenshot 2025-11-12 at 12.09.22 PM.jpg
Screenshot 2025-11-12 at 12.09.36 PM.jpg

Validation files:
19.png
25.jpeg
27.png
29.jpeg
3.png
40.png
57.jpeg
Kate_4_loose.JPG
Kate_53L.JPG
Kate_54L.JPG
112.jpeg
119.jpeg
123(1).jpeg
133.jpeg
163.png
20.png
58.png
7.png
Copy of 74.png
Copy of 82.jpeg
Kate_10T.JPG
Kate_14T.JPG
Kate_17T.JPG
Kate_28T.JPG
Kate_61T.JPG
Kate_80T.JPG
```

```
Kate_81T.JPG
Screenshot 2025-11-12 230154.png
Screenshot 2025-11-12 at 12.09.16 PM.jpg
```

```
Accuracy : 0.7241
Precision: 0.7147
Recall   : 0.7241
F1-score : 0.7157

===== Fold 4 =====

Train files:
1.png
10.jpeg
11.jpeg
12.jpeg
13.png
14.jpeg
15.jpeg
16.jpeg
17.jpeg
18.png
19.png
2.png
20.jpeg
22.png
23.png
25.jpeg
27.png
28.png
29.jpeg
3.png
31.png
36.png
38.png
39.png
4.png
40.png
41.png
43.png
44.png
45.png
46.png
49.jpeg
5.png
50.jpeg
51.jpeg
52.jpeg
57.jpeg
6.png
8.png
Copy of 1.png
Copy of 10.jpeg
Copy of 19.png
Copy of 20.jpeg
Copy of 31.png
Copy of 35.png
Copy of 42.png
Copy of 44.png
Copy of 45.png
Copy of 47.png
Copy of 51.jpeg
Copy of 52.jpeg
Copy of 54.jpeg
Copy of 58.png
IMG_2688.PNG
Kate_25L.JPG
Kate_34L.JPG
Kate_3_loose.JPG
Kate_43L.JPG
Kate_45L.JPG
```

```
Kate_46L.JPG
Kate_47L.JPG
Kate_48L.JPG
Kate_4_loose.JPG
Kate_53L.JPG
Kate_54L.JPG
Kate_56L.JPG
Kate_57L.JPG
Kate_5L.JPG
Kate_60L.JPG
Kate_62L.JPG
Kate_64L.JPG
Kate_7L.JPG
1.png
112.jpeg
113.jpeg
114.jpeg
115.jpeg
116.jpeg
117.jpeg
118.jpeg
119.jpeg
12.png
120.jpeg
121.jpeg
123(1).jpeg
123.jpeg
126.jpeg
127.jpeg
128.jpeg
129.jpeg
13.png
130.jpeg
131.jpeg
132.jpeg
133.jpeg
134.jpeg
136.jpeg
138.jpeg
139.jpeg
142.jpeg
144.jpeg
145.jpeg
146.jpeg
150.jpeg
156.png
16.png
161.jpeg
163.png
17.png
20.png
21.png
25.png
26.png
27.png
28.png
42.png
45.png
46.png
48.png
53.png
54.png
58.png
60.png
61.png
63.png
65.png
67.png
```

```
68.png
7.png
71.png
72.png
74.png
76.png
84.jpeg
85.jpeg
88.jpeg
9.png
95.jpeg
Copy of 10.png
Copy of 101.jpeg
Copy of 102.jpeg
Copy of 103.jpeg
Copy of 104.jpeg
Copy of 110.jpeg
Copy of 111.jpeg
Copy of 112.jpeg
Copy of 12.png
Copy of 122.jpeg
Copy of 13.png
Copy of 135.jpeg
Copy of 137.jpeg
Copy of 14.png
Copy of 140.jpeg
Copy of 15.png
Copy of 155.png
Copy of 159.jpeg
Copy of 162.jpeg
Copy of 163.png
Copy of 18.png
Copy of 2.png
Copy of 23.png
Copy of 27.png
Copy of 29.png
Copy of 3.png
Copy of 31.png
Copy of 37.png
Copy of 4.png
Copy of 41.png
Copy of 45.png
Copy of 46.png
Copy of 47.png
Copy of 49.png
Copy of 5.png
Copy of 50.png
Copy of 51.png
Copy of 52.png
Copy of 56.png
Copy of 57.png
Copy of 59.png
Copy of 6.png
Copy of 62.png
Copy of 64.png
Copy of 65.png
Copy of 69.png
Copy of 74.png
Copy of 76.png
Copy of 8.png
Copy of 82.jpeg
Copy of 83.jpeg
Copy of 86.jpeg
Copy of 87.jpeg
Copy of 88.jpeg
Copy of 96.jpeg
Copy of 97.jpeg
Copy of 99.jpeg
```

```
    Kate_10T.JPG
    Kate_11T.JPG
    Kate_12T.JPG
    Kate_13T.JPG
    Kate_14T.JPG
    Kate_15T.JPG
    Kate_17T.JPG
    Kate_18T.JPG
    Kate_1T.JPG
    Kate_20T.JPG
    Kate_22T.JPG
    Kate_24T.JPG
    Kate_27T.JPG
    Kate_28T.JPG
    Kate_2T.JPG
    Kate_30T.JPG
    Kate_31T.JPG
    Kate_32T.JPG
    Kate_33T.JPG
    Kate_35T.JPG
    Kate_38T.JPG
    Kate_39T.JPG
    Kate_40T.JPG
    Kate_41T.JPG
    Kate_42T.JPG
    Kate_43T.JPG
    Kate_44T.JPG
    Kate_49T.JPG
    Kate_50T.JPG
    Kate_58T.JPG
    Kate_59T.JPG
    Kate_61T.JPG
    Kate_63T.JPG
    Kate_6T.JPG
    Kate_74T.JPG
    Kate_75T.JPG
    Kate_76T.JPG
    Kate_77T.JPG
    Kate_78T.JPG
    Kate_79T.JPG
    Kate_80T.JPG
    Kate_81T.JPG
    Kate_83T.JPG
    Kate_84T.JPG
    Kate_86T.JPG
    Kate_87T.JPG
    Kate_9T.JPG
    Screenshot 2025-11-12 225851.png
    Screenshot 2025-11-12 225907.png
    Screenshot 2025-11-12 225920.png
    Screenshot 2025-11-12 225958.png
    Screenshot 2025-11-12 230055.png
    Screenshot 2025-11-12 230120.png
    Screenshot 2025-11-12 230154.png
    Screenshot 2025-11-12 230409.png
    Screenshot 2025-11-12 at 11.59.32 AM.jpg
    Screenshot 2025-11-12 at 11.59.43 AM.jpg
    Screenshot 2025-11-12 at 12.00.21 PM.jpg
    Screenshot 2025-11-12 at 12.00.29 PM.jpg
    Screenshot 2025-11-12 at 12.00.43 PM.jpg
    Screenshot 2025-11-12 at 12.09.16 PM.jpg
    Screenshot 2025-11-12 at 12.09.22 PM.jpg
    Screenshot 2025-11-12 at 12.09.36 PM.jpg

    Validation files:
    48.png
    58.png
    Copy of 6.png
```

```
Kate_23L.JPG
Kate_55L.JPG
11.png
125.jpeg
19.png
24.png
32.png
44.png
66.png
Copy of 156.png
Copy of 157.png
Copy of 160.jpeg
Copy of 22.png
Copy of 30.png
Copy of 43.png
Copy of 73.png
Copy of 75.png
Kate_16T.JPG
Kate_19T.JPG
Kate_21T.JPG
Kate_82T.JPG
Kate_8T.JPG
Screenshot 2025-11-12 124316.png
Screenshot 2025-11-12 225944.png
Screenshot 2025-11-12 230011.png
Screenshot 2025-11-12 at 11.59.37 AM.jpg
Accuracy : 0.5517
Precision: 0.8755
Recall   : 0.5517
F1-score : 0.5952

===== Fold 5 =====

Train files:
1.png
10.jpeg
12.jpeg
13.png
14.jpeg
15.jpeg
16.jpeg
17.jpeg
18.png
19.png
2.png
22.png
23.png
25.jpeg
27.png
28.png
29.jpeg
3.png
31.png
36.png
38.png
39.png
4.png
40.png
41.png
43.png
44.png
46.png
48.png
49.jpeg
5.png
50.jpeg
51.jpeg
52.jpeg
```

```
57.jpeg
58.png
6.png
8.png
Copy of 1.png
Copy of 10.jpeg
Copy of 19.png
Copy of 20.jpeg
Copy of 31.png
Copy of 35.png
Copy of 42.png
Copy of 44.png
Copy of 45.png
Copy of 47.png
Copy of 51.jpeg
Copy of 52.jpeg
Copy of 54.jpeg
Copy of 58.png
Copy of 6.png
IMG_2688.PNG
Kate_23L.JPG
Kate_25L.JPG
Kate_34L.JPG
Kate_3_loose.JPG
Kate_43L.JPG
Kate_45L.JPG
Kate_46L.JPG
Kate_47L.JPG
Kate_4_loose.JPG
Kate_53L.JPG
Kate_54L.JPG
Kate_55L.JPG
Kate_56L.JPG
Kate_57L.JPG
Kate_5L.JPG
Kate_60L.JPG
Kate_64L.JPG
1.png
11.png
112.jpeg
113.jpeg
114.jpeg
115.jpeg
116.jpeg
117.jpeg
119.jpeg
12.png
120.jpeg
121.jpeg
123(1).jpeg
123.jpeg
125.jpeg
126.jpeg
127.jpeg
128.jpeg
13.png
130.jpeg
131.jpeg
132.jpeg
133.jpeg
134.jpeg
136.jpeg
139.jpeg
142.jpeg
145.jpeg
146.jpeg
150.jpeg
156.png
```

```
16.png
161.jpeg
163.png
19.png
20.png
21.png
24.png
26.png
27.png
28.png
32.png
44.png
46.png
48.png
53.png
54.png
58.png
60.png
61.png
63.png
65.png
66.png
67.png
68.png
7.png
71.png
72.png
74.png
76.png
84.jpeg
85.jpeg
88.jpeg
9.png
95.jpeg
Copy of 10.png
Copy of 101.jpeg
Copy of 102.jpeg
Copy of 103.jpeg
Copy of 104.jpeg
Copy of 110.jpeg
Copy of 111.jpeg
Copy of 13.png
Copy of 135.jpeg
Copy of 137.jpeg
Copy of 14.png
Copy of 140.jpeg
Copy of 15.png
Copy of 155.png
Copy of 156.png
Copy of 157.png
Copy of 159.jpeg
Copy of 160.jpeg
Copy of 162.jpeg
Copy of 163.png
Copy of 18.png
Copy of 2.png
Copy of 22.png
Copy of 23.png
Copy of 29.png
Copy of 3.png
Copy of 30.png
Copy of 31.png
Copy of 4.png
Copy of 41.png
Copy of 43.png
Copy of 46.png
Copy of 47.png
Copy of 49.png
```

```
Copy of 5.png
Copy of 50.png
Copy of 56.png
Copy of 59.png
Copy of 6.png
Copy of 62.png
Copy of 64.png
Copy of 65.png
Copy of 69.png
Copy of 73.png
Copy of 74.png
Copy of 75.png
Copy of 76.png
Copy of 8.png
Copy of 82.jpeg
Copy of 83.jpeg
Copy of 86.jpeg
Copy of 87.jpeg
Copy of 88.jpeg
Copy of 96.jpeg
Copy of 97.jpeg
Copy of 99.jpeg
Kate_10T.JPG
Kate_11T.JPG
Kate_12T.JPG
Kate_14T.JPG
Kate_15T.JPG
Kate_16T.JPG
Kate_17T.JPG
Kate_18T.JPG
Kate_19T.JPG
Kate_1T.JPG
Kate_21T.JPG
Kate_22T.JPG
Kate_24T.JPG
Kate_27T.JPG
Kate_28T.JPG
Kate_2T.JPG
Kate_30T.JPG
Kate_31T.JPG
Kate_32T.JPG
Kate_33T.JPG
Kate_35T.JPG
Kate_38T.JPG
Kate_39T.JPG
Kate_40T.JPG
Kate_41T.JPG
Kate_43T.JPG
Kate_44T.JPG
Kate_49T.JPG
Kate_50T.JPG
Kate_59T.JPG
Kate_61T.JPG
Kate_63T.JPG
Kate_6T.JPG
Kate_74T.JPG
Kate_75T.JPG
Kate_76T.JPG
Kate_77T.JPG
Kate_78T.JPG
Kate_79T.JPG
Kate_80T.JPG
Kate_81T.JPG
Kate_82T.JPG
Kate_83T.JPG
Kate_84T.JPG
Kate_86T.JPG
Kate_8T.JPG
```

```
Kate_9T.JPG
Screenshot 2025-11-12 124316.png
Screenshot 2025-11-12 225851.png
Screenshot 2025-11-12 225907.png
Screenshot 2025-11-12 225920.png
Screenshot 2025-11-12 225944.png
Screenshot 2025-11-12 225958.png
Screenshot 2025-11-12 230011.png
Screenshot 2025-11-12 230055.png
Screenshot 2025-11-12 230120.png
Screenshot 2025-11-12 230154.png
Screenshot 2025-11-12 230409.png
Screenshot 2025-11-12 at 11.59.32 AM.jpg
Screenshot 2025-11-12 at 11.59.37 AM.jpg
Screenshot 2025-11-12 at 11.59.43 AM.jpg
Screenshot 2025-11-12 at 12.00.21 PM.jpg
Screenshot 2025-11-12 at 12.00.29 PM.jpg
Screenshot 2025-11-12 at 12.00.43 PM.jpg
Screenshot 2025-11-12 at 12.09.16 PM.jpg
Screenshot 2025-11-12 at 12.09.36 PM.jpg

Validation files:
11.jpeg
20.jpeg
45.png
Kate_48L.JPG
Kate_62L.JPG
Kate_7L.JPG
118.jpeg
129.jpeg
138.jpeg
144.jpeg
17.png
25.png
42.png
45.png
Copy of 112.jpeg
Copy of 12.png
Copy of 122.jpeg
Copy of 27.png
Copy of 37.png
Copy of 45.png
Copy of 51.png
Copy of 52.png
Copy of 57.png
Kate_13T.JPG
Kate_20T.JPG
Kate_42T.JPG
Kate_58T.JPG
Kate_87T.JPG
Screenshot 2025-11-12 at 12.09.22 PM.jpg
Accuracy : 0.7931
Precision: 0.8210
Recall   : 0.7931
F1-score : 0.8032

===== Fold 6 =====

Train files:
10.jpeg
11.jpeg
12.jpeg
13.png
14.jpeg
15.jpeg
16.jpeg
17.jpeg
18.png
```

```
19.png
2.png
20.jpeg
22.png
23.png
25.jpeg
27.png
28.png
29.jpeg
3.png
31.png
36.png
38.png
39.png
4.png
40.png
43.png
45.png
46.png
48.png
49.jpeg
5.png
50.jpeg
57.jpeg
58.png
6.png
8.png
Copy of 10.jpeg
Copy of 19.png
Copy of 20.jpeg
Copy of 31.png
Copy of 35.png
Copy of 42.png
Copy of 44.png
Copy of 45.png
Copy of 47.png
Copy of 52.jpeg
Copy of 54.jpeg
Copy of 58.png
Copy of 6.png
IMG_2688.PNG
Kate_23L.JPG
Kate_25L.JPG
Kate_34L.JPG
Kate_3_loose.JPG
Kate_45L.JPG
Kate_46L.JPG
Kate_47L.JPG
Kate_48L.JPG
Kate_4_loose.JPG
Kate_53L.JPG
Kate_54L.JPG
Kate_55L.JPG
Kate_56L.JPG
Kate_57L.JPG
Kate_5L.JPG
Kate_60L.JPG
Kate_62L.JPG
Kate_64L.JPG
Kate_7L.JPG
1.png
11.png
112.jpeg
113.jpeg
114.jpeg
115.jpeg
116.jpeg
117.jpeg
```

```
118.jpeg
119.jpeg
12.png
120.jpeg
121.jpeg
123(1).jpeg
123.jpeg
125.jpeg
126.jpeg
127.jpeg
129.jpeg
13.png
131.jpeg
133.jpeg
134.jpeg
136.jpeg
138.jpeg
139.jpeg
142.jpeg
144.jpeg
145.jpeg
146.jpeg
150.jpeg
156.png
16.png
161.jpeg
163.png
17.png
19.png
20.png
21.png
24.png
25.png
26.png
27.png
28.png
32.png
42.png
44.png
45.png
46.png
48.png
53.png
54.png
58.png
60.png
61.png
63.png
66.png
67.png
68.png
7.png
74.png
76.png
84.jpeg
85.jpeg
88.jpeg
9.png
95.jpeg
Copy of 101.jpeg
Copy of 102.jpeg
Copy of 103.jpeg
Copy of 104.jpeg
Copy of 110.jpeg
Copy of 111.jpeg
Copy of 112.jpeg
Copy of 12.png
Copy of 122.jpeg
```

```
Copy of 13.png
Copy of 135.jpeg
Copy of 137.jpeg
Copy of 14.png
Copy of 140.jpeg
Copy of 15.png
Copy of 155.png
Copy of 156.png
Copy of 157.png
Copy of 159.jpeg
Copy of 160.jpeg
Copy of 162.jpeg
Copy of 163.png
Copy of 18.png
Copy of 2.png
Copy of 22.png
Copy of 23.png
Copy of 27.png
Copy of 29.png
Copy of 30.png
Copy of 37.png
Copy of 4.png
Copy of 41.png
Copy of 43.png
Copy of 45.png
Copy of 46.png
Copy of 47.png
Copy of 49.png
Copy of 5.png
Copy of 50.png
Copy of 51.png
Copy of 52.png
Copy of 56.png
Copy of 57.png
Copy of 6.png
Copy of 62.png
Copy of 64.png
Copy of 69.png
Copy of 73.png
Copy of 74.png
Copy of 75.png
Copy of 76.png
Copy of 8.png
Copy of 82.jpeg
Copy of 83.jpeg
Copy of 87.jpeg
Copy of 96.jpeg
Copy of 97.jpeg
Copy of 99.jpeg
Kate_10T.JPG
Kate_12T.JPG
Kate_13T.JPG
Kate_14T.JPG
Kate_15T.JPG
Kate_16T.JPG
Kate_17T.JPG
Kate_18T.JPG
Kate_19T.JPG
Kate_1T.JPG
Kate_20T.JPG
Kate_21T.JPG
Kate_27T.JPG
Kate_28T.JPG
Kate_2T.JPG
Kate_30T.JPG
Kate_32T.JPG
Kate_33T.JPG
Kate_35T.JPG
```

```
Kate_38T.JPG
Kate_39T.JPG
Kate_40T.JPG
Kate_41T.JPG
Kate_42T.JPG
Kate_43T.JPG
Kate_44T.JPG
Kate_50T.JPG
Kate_58T.JPG
Kate_59T.JPG
Kate_61T.JPG
Kate_63T.JPG
Kate_6T.JPG
Kate_74T.JPG
Kate_75T.JPG
Kate_76T.JPG
Kate_77T.JPG
Kate_78T.JPG
Kate_79T.JPG
Kate_80T.JPG
Kate_81T.JPG
Kate_82T.JPG
Kate_83T.JPG
Kate_84T.JPG
Kate_86T.JPG
Kate_87T.JPG
Kate_8T.JPG
Kate_9T.JPG
Screenshot 2025-11-12 124316.png
Screenshot 2025-11-12 225851.png
Screenshot 2025-11-12 225907.png
Screenshot 2025-11-12 225920.png
Screenshot 2025-11-12 225944.png
Screenshot 2025-11-12 230011.png
Screenshot 2025-11-12 230055.png
Screenshot 2025-11-12 230154.png
Screenshot 2025-11-12 230409.png
Screenshot 2025-11-12 at 11.59.32 AM.jpg
Screenshot 2025-11-12 at 11.59.37 AM.jpg
Screenshot 2025-11-12 at 12.00.21 PM.jpg
Screenshot 2025-11-12 at 12.00.29 PM.jpg
Screenshot 2025-11-12 at 12.00.43 PM.jpg
Screenshot 2025-11-12 at 12.09.16 PM.jpg
Screenshot 2025-11-12 at 12.09.22 PM.jpg
Screenshot 2025-11-12 at 12.09.36 PM.jpg

Validation files:
1.png
41.png
44.png
51.jpeg
52.jpeg
Copy of 1.png
Copy of 51.jpeg
Kate_43L.JPG
128.jpeg
130.jpeg
132.jpeg
65.png
71.png
72.png
Copy of 10.png
Copy of 3.png
Copy of 31.png
Copy of 59.png
Copy of 65.png
Copy of 86.jpeg
Copy of 88.jpeg
```

```
Kate_11T.JPG
Kate_22T.JPG
Kate_24T.JPG
Kate_31T.JPG
Kate_49T.JPG
Screenshot 2025-11-12 225958.png
Screenshot 2025-11-12 230120.png
Screenshot 2025-11-12 at 11.59.43 AM.jpg
```

```
C:\Users\madis\anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1471:
UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

```
Accuracy : 0.7241
Precision: 0.5244
Recall   : 0.7241
F1-score : 0.6083

===== Fold 7 =====

Train files:
1.png
10.jpeg
11.jpeg
12.jpeg
14.jpeg
15.jpeg
16.jpeg
17.jpeg
18.png
19.png
20.jpeg
22.png
23.png
25.jpeg
27.png
28.png
29.jpeg
3.png
31.png
36.png
38.png
4.png
40.png
41.png
44.png
45.png
46.png
48.png
5.png
50.jpeg
51.jpeg
52.jpeg
57.jpeg
58.png
6.png
Copy of 1.png
Copy of 10.jpeg
Copy of 19.png
Copy of 31.png
Copy of 35.png
Copy of 42.png
Copy of 44.png
Copy of 45.png
Copy of 47.png
Copy of 51.jpeg
Copy of 52.jpeg
Copy of 54.jpeg
Copy of 58.png
Copy of 6.png
IMG_2688.PNG
Kate_23L.JPG
Kate_25L.JPG
Kate_34L.JPG
Kate_3_loose.JPG
Kate_43L.JPG
Kate_46L.JPG
Kate_48L.JPG
Kate_4_loose.JPG
Kate_53L.JPG
```

```
Kate_54L.JPG
Kate_55L.JPG
Kate_57L.JPG
Kate_5L.JPG
Kate_60L.JPG
Kate_62L.JPG
Kate_64L.JPG
Kate_7L.JPG
1.png
11.png
112.jpeg
113.jpeg
114.jpeg
115.jpeg
116.jpeg
117.jpeg
118.jpeg
119.jpeg
12.png
120.jpeg
121.jpeg
123(1).jpeg
123.jpeg
125.jpeg
126.jpeg
127.jpeg
128.jpeg
129.jpeg
13.png
130.jpeg
131.jpeg
132.jpeg
133.jpeg
134.jpeg
136.jpeg
138.jpeg
139.jpeg
142.jpeg
144.jpeg
145.jpeg
146.jpeg
150.jpeg
156.png
16.png
161.jpeg
163.png
17.png
19.png
20.png
21.png
24.png
25.png
26.png
32.png
42.png
44.png
45.png
48.png
53.png
54.png
58.png
60.png
61.png
65.png
66.png
68.png
7.png
71.png
```

```
72.png
74.png
76.png
84.jpeg
85.jpeg
88.jpeg
9.png
95.jpeg
Copy of 10.png
Copy of 101.jpeg
Copy of 102.jpeg
Copy of 104.jpeg
Copy of 110.jpeg
Copy of 112.jpeg
Copy of 12.png
Copy of 122.jpeg
Copy of 13.png
Copy of 135.jpeg
Copy of 14.png
Copy of 140.jpeg
Copy of 15.png
Copy of 155.png
Copy of 156.png
Copy of 157.png
Copy of 159.jpeg
Copy of 160.jpeg
Copy of 162.jpeg
Copy of 163.png
Copy of 18.png
Copy of 2.png
Copy of 22.png
Copy of 23.png
Copy of 27.png
Copy of 29.png
Copy of 3.png
Copy of 30.png
Copy of 31.png
Copy of 37.png
Copy of 41.png
Copy of 43.png
Copy of 45.png
Copy of 46.png
Copy of 47.png
Copy of 49.png
Copy of 5.png
Copy of 50.png
Copy of 51.png
Copy of 52.png
Copy of 56.png
Copy of 57.png
Copy of 59.png
Copy of 6.png
Copy of 62.png
Copy of 64.png
Copy of 65.png
Copy of 69.png
Copy of 73.png
Copy of 74.png
Copy of 75.png
Copy of 76.png
Copy of 8.png
Copy of 82.jpeg
Copy of 83.jpeg
Copy of 86.jpeg
Copy of 88.jpeg
Copy of 96.jpeg
Copy of 97.jpeg
Copy of 99.jpeg
```

```
    Kate_10T.JPG
    Kate_11T.JPG
    Kate_12T.JPG
    Kate_13T.JPG
    Kate_14T.JPG
    Kate_15T.JPG
    Kate_16T.JPG
    Kate_17T.JPG
    Kate_18T.JPG
    Kate_19T.JPG
    Kate_1T.JPG
    Kate_20T.JPG
    Kate_21T.JPG
    Kate_22T.JPG
    Kate_24T.JPG
    Kate_27T.JPG
    Kate_28T.JPG
    Kate_30T.JPG
    Kate_31T.JPG
    Kate_32T.JPG
    Kate_33T.JPG
    Kate_35T.JPG
    Kate_40T.JPG
    Kate_41T.JPG
    Kate_42T.JPG
    Kate_43T.JPG
    Kate_44T.JPG
    Kate_49T.JPG
    Kate_58T.JPG
    Kate_59T.JPG
    Kate_61T.JPG
    Kate_6T.JPG
    Kate_74T.JPG
    Kate_75T.JPG
    Kate_76T.JPG
    Kate_77T.JPG
    Kate_79T.JPG
    Kate_80T.JPG
    Kate_81T.JPG
    Kate_82T.JPG
    Kate_83T.JPG
    Kate_84T.JPG
    Kate_86T.JPG
    Kate_87T.JPG
    Kate_8T.JPG
    Kate_9T.JPG
    Screenshot 2025-11-12 124316.png
    Screenshot 2025-11-12 225851.png
    Screenshot 2025-11-12 225907.png
    Screenshot 2025-11-12 225920.png
    Screenshot 2025-11-12 225944.png
    Screenshot 2025-11-12 225958.png
    Screenshot 2025-11-12 230011.png
    Screenshot 2025-11-12 230055.png
    Screenshot 2025-11-12 230120.png
    Screenshot 2025-11-12 230154.png
    Screenshot 2025-11-12 at 11.59.32 AM.jpg
    Screenshot 2025-11-12 at 11.59.37 AM.jpg
    Screenshot 2025-11-12 at 11.59.43 AM.jpg
    Screenshot 2025-11-12 at 12.00.21 PM.jpg
    Screenshot 2025-11-12 at 12.09.16 PM.jpg
    Screenshot 2025-11-12 at 12.09.22 PM.jpg
    Screenshot 2025-11-12 at 12.09.36 PM.jpg

    Validation files:
    13.png
    2.png
    39.png
```

```
43.png
49.jpeg
8.png
Copy of 20.jpeg
Kate_45L.JPG
Kate_47L.JPG
Kate_56L.JPG
27.png
28.png
46.png
63.png
67.png
Copy of 103.jpeg
Copy of 111.jpeg
Copy of 137.jpeg
Copy of 4.png
Copy of 87.jpeg
Kate_2T.JPG
Kate_38T.JPG
Kate_39T.JPG
Kate_50T.JPG
Kate_63T.JPG
Kate_78T.JPG
Screenshot 2025-11-12 230409.png
Screenshot 2025-11-12 at 12.00.29 PM.jpg
Screenshot 2025-11-12 at 12.00.43 PM.jpg
Accuracy : 0.7931
Precision: 0.7890
Recall   : 0.7931
F1-score : 0.7868

===== Fold 8 =====

Train files:
1.png
11.jpeg
13.png
14.jpeg
15.jpeg
16.jpeg
17.jpeg
18.png
19.png
2.png
20.jpeg
22.png
25.jpeg
27.png
28.png
29.jpeg
3.png
31.png
36.png
38.png
39.png
4.png
40.png
41.png
43.png
44.png
45.png
46.png
48.png
49.jpeg
5.png
50.jpeg
51.jpeg
52.jpeg
```

```
57.jpeg
58.png
8.png
Copy of 1.png
Copy of 10.jpeg
Copy of 20.jpeg
Copy of 31.png
Copy of 35.png
Copy of 44.png
Copy of 47.png
Copy of 51.jpeg
Copy of 52.jpeg
Copy of 58.png
Copy of 6.png
IMG_2688.PNG
Kate_23L.JPG
Kate_25L.JPG
Kate_34L.JPG
Kate_3_loose.JPG
Kate_43L.JPG
Kate_45L.JPG
Kate_46L.JPG
Kate_47L.JPG
Kate_48L.JPG
Kate_4_loose.JPG
Kate_53L.JPG
Kate_54L.JPG
Kate_55L.JPG
Kate_56L.JPG
Kate_57L.JPG
Kate_5L.JPG
Kate_60L.JPG
Kate_62L.JPG
Kate_64L.JPG
Kate_7L.JPG
1.png
11.png
112.jpeg
113.jpeg
115.jpeg
116.jpeg
117.jpeg
118.jpeg
119.jpeg
12.png
120.jpeg
121.jpeg
123(1).jpeg
123.jpeg
125.jpeg
126.jpeg
128.jpeg
129.jpeg
13.png
130.jpeg
131.jpeg
132.jpeg
133.jpeg
134.jpeg
138.jpeg
142.jpeg
144.jpeg
145.jpeg
146.jpeg
150.jpeg
156.png
16.png
161.jpeg
```

```
163.png
17.png
19.png
20.png
21.png
24.png
25.png
26.png
27.png
28.png
32.png
42.png
44.png
45.png
46.png
48.png
53.png
54.png
58.png
61.png
63.png
65.png
66.png
67.png
68.png
7.png
71.png
72.png
74.png
76.png
85.jpeg
88.jpeg
9.png
95.jpeg
Copy of 10.png
Copy of 101.jpeg
Copy of 102.jpeg
Copy of 103.jpeg
Copy of 104.jpeg
Copy of 110.jpeg
Copy of 111.jpeg
Copy of 112.jpeg
Copy of 12.png
Copy of 122.jpeg
Copy of 13.png
Copy of 137.jpeg
Copy of 14.png
Copy of 140.jpeg
Copy of 15.png
Copy of 155.png
Copy of 156.png
Copy of 157.png
Copy of 159.jpeg
Copy of 160.jpeg
Copy of 162.jpeg
Copy of 163.png
Copy of 18.png
Copy of 2.png
Copy of 22.png
Copy of 23.png
Copy of 27.png
Copy of 29.png
Copy of 3.png
Copy of 30.png
Copy of 31.png
Copy of 37.png
Copy of 4.png
Copy of 41.png
```

```
Copy of 43.png
Copy of 45.png
Copy of 46.png
Copy of 47.png
Copy of 49.png
Copy of 50.png
Copy of 51.png
Copy of 52.png
Copy of 56.png
Copy of 57.png
Copy of 59.png
Copy of 6.png
Copy of 62.png
Copy of 65.png
Copy of 73.png
Copy of 74.png
Copy of 75.png
Copy of 76.png
Copy of 8.png
Copy of 82.jpeg
Copy of 83.jpeg
Copy of 86.jpeg
Copy of 87.jpeg
Copy of 88.jpeg
Copy of 96.jpeg
Copy of 97.jpeg
Copy of 99.jpeg
Kate_10T.JPG
Kate_11T.JPG
Kate_13T.JPG
Kate_14T.JPG
Kate_16T.JPG
Kate_17T.JPG
Kate_18T.JPG
Kate_19T.JPG
Kate_20T.JPG
Kate_21T.JPG
Kate_22T.JPG
Kate_24T.JPG
Kate_27T.JPG
Kate_28T.JPG
Kate_2T.JPG
Kate_31T.JPG
Kate_32T.JPG
Kate_33T.JPG
Kate_35T.JPG
Kate_38T.JPG
Kate_39T.JPG
Kate_41T.JPG
Kate_42T.JPG
Kate_43T.JPG
Kate_49T.JPG
Kate_50T.JPG
Kate_58T.JPG
Kate_61T.JPG
Kate_63T.JPG
Kate_74T.JPG
Kate_75T.JPG
Kate_76T.JPG
Kate_77T.JPG
Kate_78T.JPG
Kate_79T.JPG
Kate_80T.JPG
Kate_81T.JPG
Kate_82T.JPG
Kate_83T.JPG
Kate_86T.JPG
Kate_87T.JPG
```

```
Kate_8T.JPG
Kate_9T.JPG
Screenshot 2025-11-12 124316.png
Screenshot 2025-11-12 225851.png
Screenshot 2025-11-12 225907.png
Screenshot 2025-11-12 225920.png
Screenshot 2025-11-12 225944.png
Screenshot 2025-11-12 225958.png
Screenshot 2025-11-12 230011.png
Screenshot 2025-11-12 230055.png
Screenshot 2025-11-12 230120.png
Screenshot 2025-11-12 230154.png
Screenshot 2025-11-12 230409.png
Screenshot 2025-11-12 at 11.59.37 AM.jpg
Screenshot 2025-11-12 at 11.59.43 AM.jpg
Screenshot 2025-11-12 at 12.00.21 PM.jpg
Screenshot 2025-11-12 at 12.00.29 PM.jpg
Screenshot 2025-11-12 at 12.00.43 PM.jpg
Screenshot 2025-11-12 at 12.09.16 PM.jpg
Screenshot 2025-11-12 at 12.09.22 PM.jpg
Screenshot 2025-11-12 at 12.09.36 PM.jpg

Validation files:
10.jpeg
12.jpeg
23.png
6.png
Copy of 19.png
Copy of 42.png
Copy of 45.png
Copy of 54.jpeg
114.jpeg
127.jpeg
136.jpeg
139.jpeg
60.png
84.jpeg
Copy of 135.jpeg
Copy of 5.png
Copy of 64.png
Copy of 69.png
Kate_12T.JPG
Kate_15T.JPG
Kate_1T.JPG
Kate_30T.JPG
Kate_40T.JPG
Kate_44T.JPG
Kate_59T.JPG
Kate_6T.JPG
Kate_84T.JPG
Screenshot 2025-11-12 at 11.59.32 AM.jpg
Accuracy : 0.6071
Precision: 0.4857
Recall   : 0.6071
F1-score : 0.5397

===== Fold 9 =====

Train files:
1.png
10.jpeg
11.jpeg
12.jpeg
13.png
14.jpeg
15.jpeg
18.png
19.png
```

```
2.png
20.jpeg
23.png
25.jpeg
27.png
29.jpeg
3.png
31.png
36.png
38.png
39.png
4.png
40.png
41.png
43.png
44.png
45.png
46.png
48.png
49.jpeg
5.png
51.jpeg
52.jpeg
57.jpeg
58.png
6.png
8.png
Copy of 1.png
Copy of 10.jpeg
Copy of 19.png
Copy of 20.jpeg
Copy of 31.png
Copy of 35.png
Copy of 42.png
Copy of 44.png
Copy of 45.png
Copy of 47.png
Copy of 51.jpeg
Copy of 54.jpeg
Copy of 58.png
Copy of 6.png
IMG_2688.PNG
Kate_23L.JPG
Kate_25L.JPG
Kate_3_loose.JPG
Kate_43L.JPG
Kate_45L.JPG
Kate_46L.JPG
Kate_47L.JPG
Kate_48L.JPG
Kate_4_loose.JPG
Kate_53L.JPG
Kate_54L.JPG
Kate_55L.JPG
Kate_56L.JPG
Kate_57L.JPG
Kate_60L.JPG
Kate_62L.JPG
Kate_64L.JPG
Kate_7L.JPG
1.png
11.png
112.jpeg
114.jpeg
115.jpeg
117.jpeg
118.jpeg
119.jpeg
```

```
12.png
123(1).jpeg
125.jpeg
126.jpeg
127.jpeg
128.jpeg
129.jpeg
13.png
130.jpeg
131.jpeg
132.jpeg
133.jpeg
134.jpeg
136.jpeg
138.jpeg
139.jpeg
142.jpeg
144.jpeg
145.jpeg
146.jpeg
156.png
16.png
161.jpeg
163.png
17.png
19.png
20.png
21.png
24.png
25.png
26.png
27.png
28.png
32.png
42.png
44.png
45.png
46.png
53.png
58.png
60.png
63.png
65.png
66.png
67.png
68.png
7.png
71.png
72.png
74.png
76.png
84.jpeg
85.jpeg
88.jpeg
9.png
95.jpeg
Copy of 10.png
Copy of 101.jpeg
Copy of 102.jpeg
Copy of 103.jpeg
Copy of 104.jpeg
Copy of 110.jpeg
Copy of 111.jpeg
Copy of 112.jpeg
Copy of 12.png
Copy of 122.jpeg
Copy of 13.png
Copy of 135.jpeg
```

```
Copy of 137.jpeg
Copy of 140.jpeg
Copy of 15.png
Copy of 156.png
Copy of 157.png
Copy of 159.jpeg
Copy of 160.jpeg
Copy of 163.png
Copy of 18.png
Copy of 2.png
Copy of 22.png
Copy of 23.png
Copy of 27.png
Copy of 29.png
Copy of 3.png
Copy of 30.png
Copy of 31.png
Copy of 37.png
Copy of 4.png
Copy of 41.png
Copy of 43.png
Copy of 45.png
Copy of 46.png
Copy of 47.png
Copy of 49.png
Copy of 5.png
Copy of 50.png
Copy of 51.png
Copy of 52.png
Copy of 56.png
Copy of 57.png
Copy of 59.png
Copy of 6.png
Copy of 64.png
Copy of 65.png
Copy of 69.png
Copy of 73.png
Copy of 74.png
Copy of 75.png
Copy of 8.png
Copy of 82.jpeg
Copy of 83.jpeg
Copy of 86.jpeg
Copy of 87.jpeg
Copy of 88.jpeg
Copy of 96.jpeg
Copy of 97.jpeg
Copy of 99.jpeg
Kate_10T.JPG
Kate_11T.JPG
Kate_12T.JPG
Kate_13T.JPG
Kate_14T.JPG
Kate_15T.JPG
Kate_16T.JPG
Kate_17T.JPG
Kate_18T.JPG
Kate_19T.JPG
Kate_1T.JPG
Kate_20T.JPG
Kate_21T.JPG
Kate_22T.JPG
Kate_24T.JPG
Kate_27T.JPG
Kate_28T.JPG
Kate_2T.JPG
Kate_30T.JPG
Kate_31T.JPG
```

```
Kate_32T.JPG
Kate_33T.JPG
Kate_35T.JPG
Kate_38T.JPG
Kate_39T.JPG
Kate_40T.JPG
Kate_42T.JPG
Kate_44T.JPG
Kate_49T.JPG
Kate_50T.JPG
Kate_58T.JPG
Kate_59T.JPG
Kate_61T.JPG
Kate_63T.JPG
Kate_6T.JPG
Kate_74T.JPG
Kate_75T.JPG
Kate_76T.JPG
Kate_77T.JPG
Kate_78T.JPG
Kate_79T.JPG
Kate_80T.JPG
Kate_81T.JPG
Kate_82T.JPG
Kate_84T.JPG
Kate_87T.JPG
Kate_8T.JPG
Kate_9T.JPG
Screenshot 2025-11-12 124316.png
Screenshot 2025-11-12 225851.png
Screenshot 2025-11-12 225907.png
Screenshot 2025-11-12 225920.png
Screenshot 2025-11-12 225944.png
Screenshot 2025-11-12 225958.png
Screenshot 2025-11-12 230011.png
Screenshot 2025-11-12 230120.png
Screenshot 2025-11-12 230154.png
Screenshot 2025-11-12 230409.png
Screenshot 2025-11-12 at 11.59.32 AM.jpg
Screenshot 2025-11-12 at 11.59.37 AM.jpg
Screenshot 2025-11-12 at 11.59.43 AM.jpg
Screenshot 2025-11-12 at 12.00.29 PM.jpg
Screenshot 2025-11-12 at 12.00.43 PM.jpg
Screenshot 2025-11-12 at 12.09.16 PM.jpg
Screenshot 2025-11-12 at 12.09.22 PM.jpg
Screenshot 2025-11-12 at 12.09.36 PM.jpg

Validation files:
16.jpeg
17.jpeg
22.png
28.png
50.jpeg
Copy of 52.jpeg
Kate_34L.JPG
Kate_5L.JPG
113.jpeg
116.jpeg
120.jpeg
121.jpeg
123.jpeg
150.jpeg
48.png
54.png
61.png
Copy of 14.png
Copy of 155.png
Copy of 162.jpeg
```

```
Copy of 62.png
Copy of 76.png
Kate_41T.JPG
Kate_43T.JPG
Kate_83T.JPG
Kate_86T.JPG
Screenshot 2025-11-12 230055.png
Screenshot 2025-11-12 at 12.00.21 PM.jpg
```

```
Accuracy : 0.6786
Precision: 0.5026
Recall   : 0.6786
F1-score : 0.5775

===== Fold 10 =====

Train files:
1.png
10.jpeg
11.jpeg
12.jpeg
13.png
14.jpeg
15.jpeg
16.jpeg
17.jpeg
18.png
19.png
2.png
20.jpeg
22.png
23.png
25.jpeg
27.png
28.png
29.jpeg
3.png
38.png
39.png
4.png
40.png
41.png
43.png
44.png
45.png
46.png
48.png
49.jpeg
5.png
50.jpeg
51.jpeg
52.jpeg
57.jpeg
58.png
6.png
8.png
Copy of 1.png
Copy of 10.jpeg
Copy of 19.png
Copy of 20.jpeg
Copy of 31.png
Copy of 35.png
Copy of 42.png
Copy of 45.png
Copy of 51.jpeg
Copy of 52.jpeg
Copy of 54.jpeg
Copy of 6.png
IMG_2688.PNG
Kate_23L.JPG
Kate_34L.JPG
Kate_3_loose.JPG
Kate_43L.JPG
Kate_45L.JPG
Kate_46L.JPG
Kate_47L.JPG
```

```
Kate_48L.JPG
Kate_4_loose.JPG
Kate_53L.JPG
Kate_54L.JPG
Kate_55L.JPG
Kate_56L.JPG
Kate_5L.JPG
Kate_60L.JPG
Kate_62L.JPG
Kate_64L.JPG
Kate_7L.JPG
1.png
11.png
112.jpeg
113.jpeg
114.jpeg
115.jpeg
116.jpeg
117.jpeg
118.jpeg
119.jpeg
120.jpeg
121.jpeg
123(1).jpeg
123.jpeg
125.jpeg
126.jpeg
127.jpeg
128.jpeg
129.jpeg
13.png
130.jpeg
132.jpeg
133.jpeg
136.jpeg
138.jpeg
139.jpeg
144.jpeg
145.jpeg
146.jpeg
150.jpeg
156.png
16.png
161.jpeg
163.png
17.png
19.png
20.png
21.png
24.png
25.png
27.png
28.png
32.png
42.png
44.png
45.png
46.png
48.png
54.png
58.png
60.png
61.png
63.png
65.png
66.png
67.png
68.png
```

```
7.png
71.png
72.png
74.png
76.png
84.jpeg
85.jpeg
88.jpeg
9.png
Copy of 10.png
Copy of 102.jpeg
Copy of 103.jpeg
Copy of 104.jpeg
Copy of 110.jpeg
Copy of 111.jpeg
Copy of 112.jpeg
Copy of 12.png
Copy of 122.jpeg
Copy of 135.jpeg
Copy of 137.jpeg
Copy of 14.png
Copy of 140.jpeg
Copy of 15.png
Copy of 155.png
Copy of 156.png
Copy of 157.png
Copy of 160.jpeg
Copy of 162.jpeg
Copy of 163.png
Copy of 18.png
Copy of 22.png
Copy of 23.png
Copy of 27.png
Copy of 29.png
Copy of 3.png
Copy of 30.png
Copy of 31.png
Copy of 37.png
Copy of 4.png
Copy of 41.png
Copy of 43.png
Copy of 45.png
Copy of 5.png
Copy of 51.png
Copy of 52.png
Copy of 56.png
Copy of 57.png
Copy of 59.png
Copy of 6.png
Copy of 62.png
Copy of 64.png
Copy of 65.png
Copy of 69.png
Copy of 73.png
Copy of 74.png
Copy of 75.png
Copy of 76.png
Copy of 8.png
Copy of 82.jpeg
Copy of 83.jpeg
Copy of 86.jpeg
Copy of 87.jpeg
Copy of 88.jpeg
Copy of 96.jpeg
Copy of 97.jpeg
Kate_10T.JPG
Kate_11T.JPG
Kate_12T.JPG
```

```
Kate_13T.JPG
Kate_14T.JPG
Kate_15T.JPG
Kate_16T.JPG
Kate_17T.JPG
Kate_18T.JPG
Kate_19T.JPG
Kate_1T.JPG
Kate_20T.JPG
Kate_21T.JPG
Kate_22T.JPG
Kate_24T.JPG
Kate_27T.JPG
Kate_28T.JPG
Kate_2T.JPG
Kate_30T.JPG
Kate_31T.JPG
Kate_33T.JPG
Kate_35T.JPG
Kate_38T.JPG
Kate_39T.JPG
Kate_40T.JPG
Kate_41T.JPG
Kate_42T.JPG
Kate_43T.JPG
Kate_44T.JPG
Kate_49T.JPG
Kate_50T.JPG
Kate_58T.JPG
Kate_59T.JPG
Kate_61T.JPG
Kate_63T.JPG
Kate_6T.JPG
Kate_75T.JPG
Kate_76T.JPG
Kate_77T.JPG
Kate_78T.JPG
Kate_80T.JPG
Kate_81T.JPG
Kate_82T.JPG
Kate_83T.JPG
Kate_84T.JPG
Kate_86T.JPG
Kate_87T.JPG
Kate_8T.JPG
Kate_9T.JPG
Screenshot 2025-11-12 124316.png
Screenshot 2025-11-12 225851.png
Screenshot 2025-11-12 225907.png
Screenshot 2025-11-12 225944.png
Screenshot 2025-11-12 225958.png
Screenshot 2025-11-12 230011.png
Screenshot 2025-11-12 230055.png
Screenshot 2025-11-12 230120.png
Screenshot 2025-11-12 230154.png
Screenshot 2025-11-12 230409.png
Screenshot 2025-11-12 at 11.59.32 AM.jpg
Screenshot 2025-11-12 at 11.59.37 AM.jpg
Screenshot 2025-11-12 at 11.59.43 AM.jpg
Screenshot 2025-11-12 at 12.00.21 PM.jpg
Screenshot 2025-11-12 at 12.00.29 PM.jpg
Screenshot 2025-11-12 at 12.00.43 PM.jpg
Screenshot 2025-11-12 at 12.09.16 PM.jpg
Screenshot 2025-11-12 at 12.09.22 PM.jpg

Validation files:
31.png
36.png
```

```
Copy of 44.png
Copy of 47.png
Copy of 58.png
Kate_25L.JPG
Kate_57L.JPG
12.png
131.jpeg
134.jpeg
142.jpeg
26.png
53.png
95.jpeg
Copy of 101.jpeg
Copy of 13.png
Copy of 159.jpeg
Copy of 2.png
Copy of 46.png
Copy of 47.png
Copy of 49.png
Copy of 50.png
Copy of 99.jpeg
Kate_32T.JPG
Kate_74T.JPG
Kate_79T.JPG
Screenshot 2025-11-12 225920.png
Screenshot 2025-11-12 at 12.09.36 PM.jpg
Accuracy : 0.2500
Precision: 0.0625
Recall   : 0.2500
F1-score : 0.1000

===== K-FOLD SUMMARY =====
Accuracy -> Mean: 0.6087, Std: 0.1949
Precision -> Mean: 0.6055, Std: 0.2244
Recall -> Mean: 0.6087, Std: 0.1949
F1 -> Mean: 0.5586, Std: 0.2242
```

Conclusions/action items:

In conclusion, these images did not perform as well as we had hoped for our new model. The team will need to look into other ways to augment our model, but will continue to train a model on these images. We are hoping for accuracy above 80% and F1 of 0.8.

2/20/2026 - New ML Model Training and Feature Map

Madison Michels - Feb 23, 2026, 7:07 PM CST

Title: New ML Model Training and Feature Map

Date: 2/20/2026

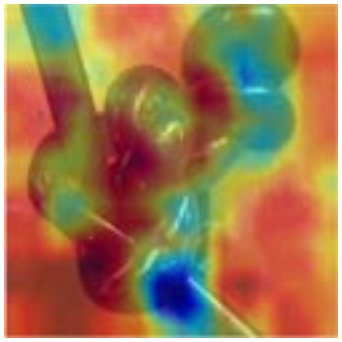
Content by: Maddie

Present: Maddie

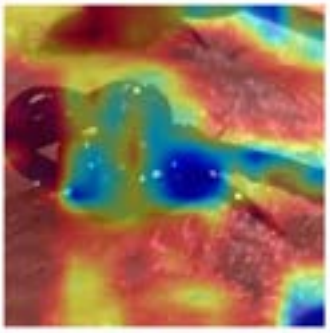
Goals: The goal of this code is to train our second model using the 500x500 resolution images and to identify where the model is selecting image features to consider during training.

Content:

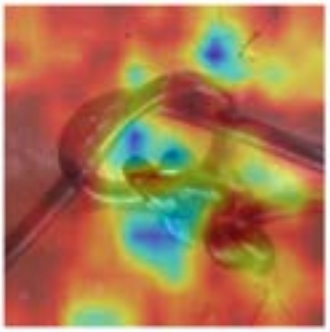
```
1 - Training Loss: 43.343..  Validation Loss: 0.808..  Validation Accuracy: 0.643
```



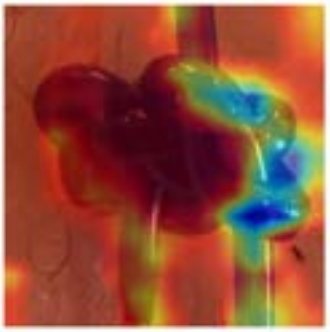
```
Epoch: 2/25..  Training Loss: 39.924..  Validation Loss: 0.724..  Validation Accuracy: 0.643
```



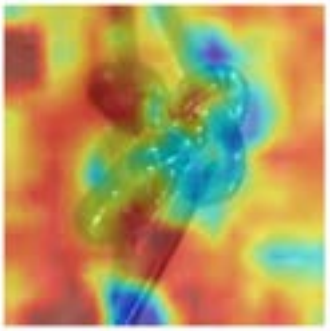
```
Epoch: 3/25..  Training Loss: 34.118..  Validation Loss: 0.638..  Validation Accuracy: 0.679
```



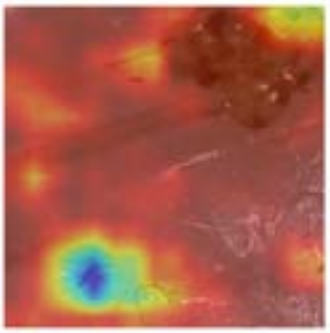
```
Epoch: 4/25..  Training Loss: 28.447..  Validation Loss: 0.692..  Validation Accuracy: 0.643
```



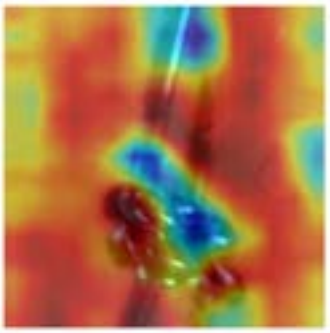
Epoch: 5/25.. Training Loss: 31.725.. Validation Loss: 0.952.. Validation Accuracy: 0.643



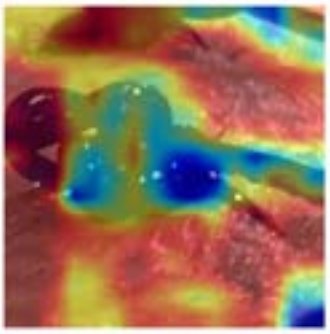
Epoch: 6/25.. Training Loss: 29.892.. Validation Loss: 0.630.. Validation Accuracy: 0.643



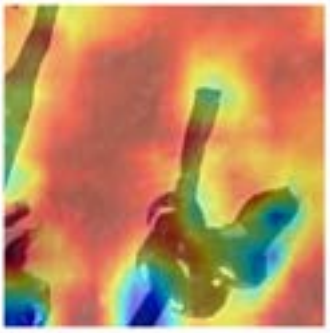
Epoch: 7/25.. Training Loss: 29.432.. Validation Loss: 0.802.. Validation Accuracy: 0.679



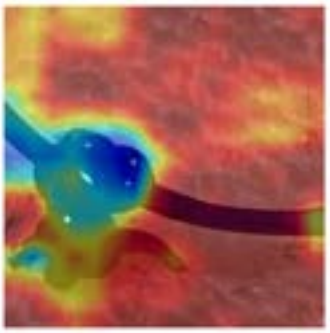
Epoch: 8/25.. Training Loss: 28.241.. Validation Loss: 0.814.. Validation Accuracy: 0.679



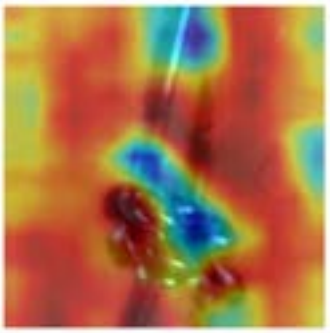
Epoch: 9/25.. Training Loss: 28.482.. Validation Loss: 0.723.. Validation Accuracy: 0.714



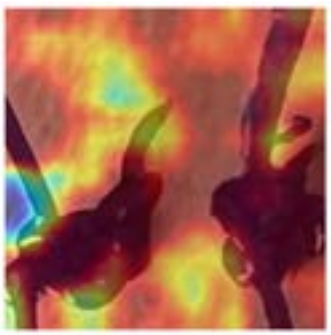
Epoch: 10/25.. Training Loss: 29.300.. Validation Loss: 0.803.. Validation Accuracy: 0.679



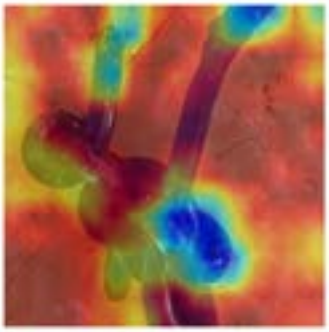
Epoch: 11/25.. Training Loss: 26.811.. Validation Loss: 0.647.. Validation Accuracy: 0.679



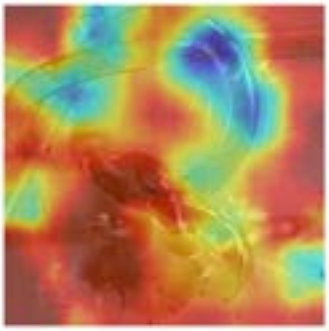
Epoch: 12/25.. Training Loss: 49.428.. Validation Loss: 1.700.. Validation Accuracy: 0.500



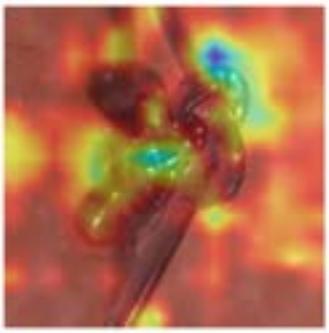
Epoch: 13/25.. Training Loss: 32.686.. Validation Loss: 0.989.. Validation Accuracy: 0.643



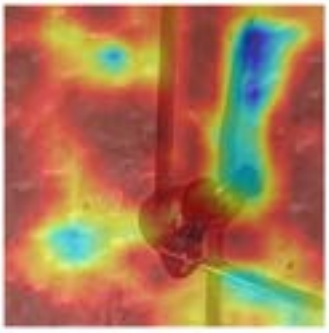
Epoch: 14/25.. Training Loss: 26.303.. Validation Loss: 0.633.. Validation Accuracy: 0.714



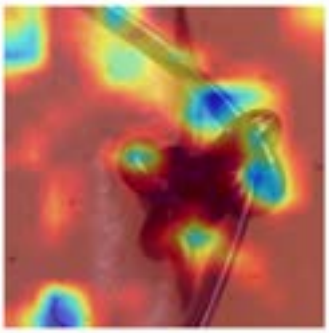
Epoch: 15/25.. Training Loss: 31.079.. Validation Loss: 0.705.. Validation Accuracy: 0.607



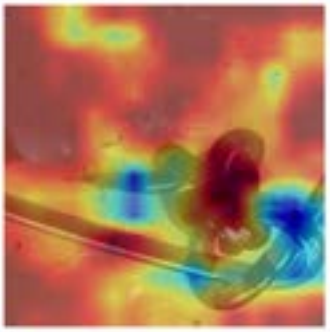
Epoch: 16/25.. Training Loss: 24.489.. Validation Loss: 0.608.. Validation Accuracy: 0.714



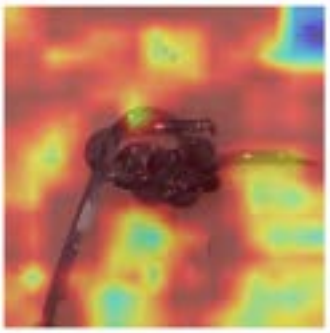
Epoch: 17/25.. Training Loss: 30.058.. Validation Loss: 0.659.. Validation Accuracy: 0.607



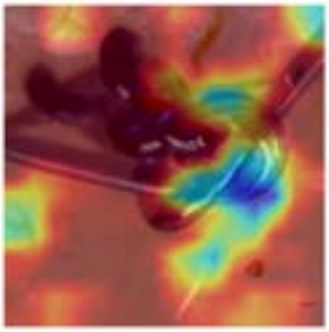
```
Epoch: 18/25..  Training Loss: 25.284..  Validation Loss: 0.652..  Validation Accuracy: 0.714
```



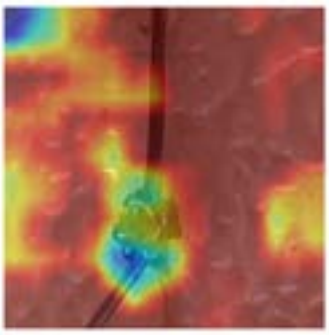
```
Epoch: 19/25..  Training Loss: 26.372..  Validation Loss: 0.646..  Validation Accuracy: 0.714
```



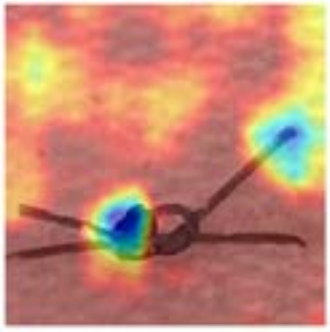
```
Epoch: 20/25..  Training Loss: 24.392..  Validation Loss: 0.599..  Validation Accuracy: 0.679
```



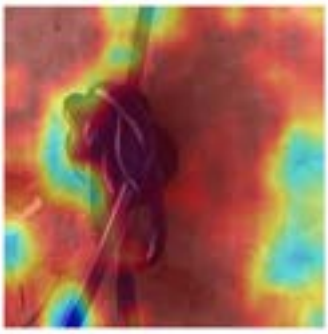
```
Epoch: 21/25..  Training Loss: 25.872..  Validation Loss: 0.647..  Validation Accuracy: 0.679
```



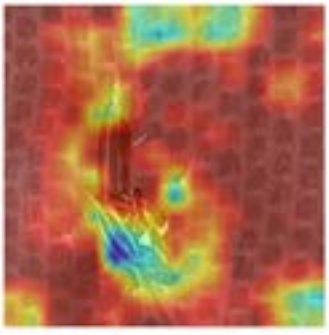
```
Epoch: 22/25..  Training Loss: 27.291..  Validation Loss: 0.796..  Validation Accuracy: 0.750
```



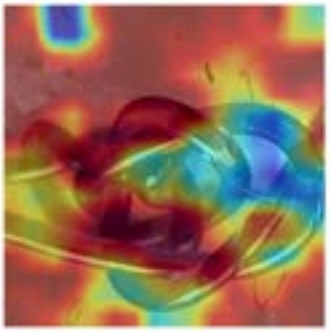
```
Epoch: 23/25..  Training Loss: 24.977..  Validation Loss: 0.718..  Validation Accuracy: 0.714
```

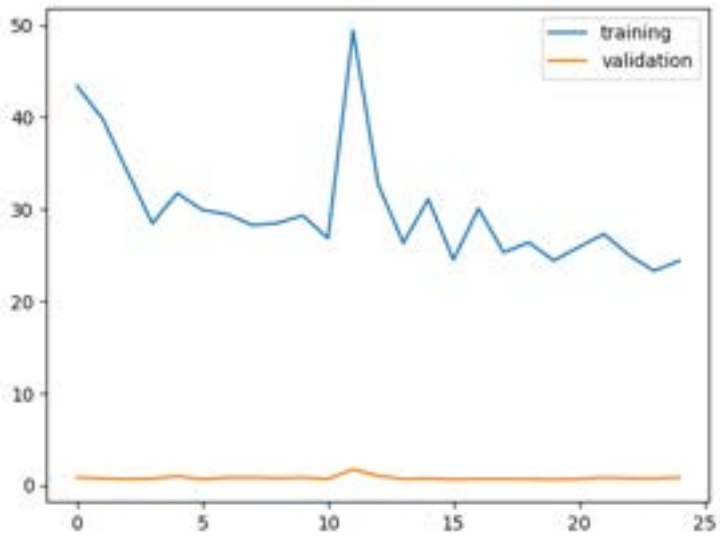


```
Epoch: 24/25..  Training Loss: 23.274..  Validation Loss: 0.696..  Validation Accuracy: 0.714
```

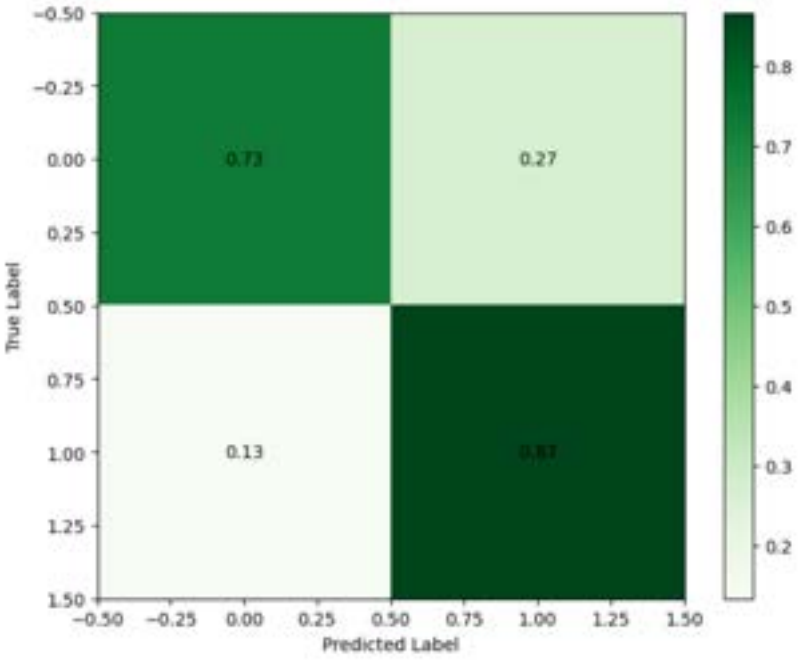


```
Epoch: 25/25..  Training Loss: 24.356..  Validation Loss: 0.822..  Validation Accuracy: 0.714
```





Confusion Matrix:



The model above has a relatively low FP rate. It also has the following statistics:

	precision	recall	f1-score	support
0	0.83	0.86	0.84	22
1	0.85	0.81	0.83	21
accuracy			0.84	43
macro avg	0.84	0.84	0.84	43
weighted avg	0.84	0.84	0.84	43

The model has achieved all of our goal setpoints for accuracy, F1 score, recall (sensitivity), and precision (PPV).

Conclusions/action items:

In conclusion, it appears that the model is primarily looking at the background skin pad and random pieces of hte knot to make its classification. Future remedies could entail blacking out the background, retaking images on the same skin pad, or using a standardized suture color.

2/23/2026 - Background Removal

Madison Michels - Feb 23, 2026, 7:33 PM CST

Title: Background Removal

Date: 02/23/2026

Content by: Maddie

Present: Maddie

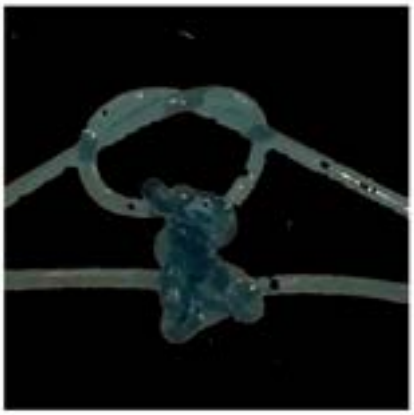
Goals: The goal of this code is to determien the most accurate and repeatable background removal strategy for the knot images.

Content:

Background removal using simple threshold: ***FASTEST***

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

# Load image
image = cv2.imread(r"C:\Users\madis\Downloads\Knot Model\sem2\resolution_500_500\Loose\old_095.jpg")
image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

# Convert to grayscale
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
plt.imshow(gray)
plt.axis("off")
plt.show()

# Threshold (invert so suture becomes white)
_, mask = cv2.threshold(gray, 100, 255, cv2.THRESH_BINARY_INV)

# Clean small noise
kernel = np.ones((3,3), np.uint8)
mask = cv2.morphologyEx(mask, cv2.MORPH_CLOSE, kernel, iterations=2)

# Apply mask
result = cv2.bitwise_and(image_rgb, image_rgb, mask=mask)

plt.imshow(result)
plt.axis("off")
plt.show()
```

------->



Background removal using Otsu: **** BEST ****

```
import cv2

import matplotlib.pyplot as plt

image = cv2.imread(r"C:\Users\madis\Downloads\Knot Model\sem2\resolution_500_500\Loose\old_095.jpg")

if image is None:
    raise ValueError("Image not found. Check file path.")

gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

plt.imshow(gray, cmap="gray")
plt.axis("off")
plt.show()


_, mask = cv2.threshold(
    gray, 0, 255,
    cv2.THRESH_BINARY_INV + cv2.THRESH_OTSU
)

kernel = np.ones((3,3), np.uint8)
mask = cv2.morphologyEx(mask, cv2.MORPH_CLOSE, kernel, iterations=2)

result = cv2.bitwise_and(image_rgb, image_rgb, mask=mask)


# Show Otsu result
plt.figure(figsize=(6,6))
plt.imshow(mask, cmap="gray")
plt.title("Otsu Threshold Mask")
plt.axis("off")
plt.show()
```

------->

Background removal, keeping only the largest contour:

```
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

plt.imshow(gray, cmap="gray")

plt.axis("off")
```

```
plt.show()

_, thresh = cv2.threshold(
    gray, 0, 255,
    cv2.THRESH_BINARY_INV + cv2.THRESH_OTSU
)
# Find contours
contours, _ = cv2.findContours(
    thresh, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE
)
# Create empty mask
mask = np.zeros_like(gray)
# Keep largest contour
largest = max(contours, key=cv2.contourArea)
cv2.drawContours(mask, [largest], -1, 255, thickness=cv2.FILLED)
# Apply mask
result = cv2.bitwise_and(image_rgb, image_rgb, mask=mask)
plt.imshow(result)
plt.axis("off")


plt.show()
```

------->

Conclusions/action items:

In conclusion, the Otsu removal looks liek the most efficient because it outlines the suture and captures the strand in only white. This eliminates the color issues and background discrepancies we might see. The largest contour option would be best, but it retains the center of the loose knot, which is not desirable, however, it might make a gap clearer to the ML algorithm. All of these moethods, however, struggle to remove the background for the clear sutures.

2/27/2026 - MATLAB code for creating squaare dataset

Madison Michels - Feb 27, 2026, 1:54 PM CST

Title: MATLAB code for creating squaare dataset

Date: 2/27

Content by: Maddie

Present: Maddie

Goals: To create a dataset for our practice ML model upload to the raspberry Pi.

Content:

Black Square Code (n=100)

```
% Number of steps
n = 100;

% Output folder
outputFolder = 'square_frames_black';
if ~exist(outputFolder, 'dir')
mkdir(outputFolder);
end

% ---- FORCE EXACT PIXEL SIZE ----
imgSize = 500; % 500x500 pixels

fig = figure('Color','black', ...
'Units','pixels', ...
'Position',[100 100 imgSize imgSize], ...
'Resize','off', ...
'MenuBar','none', ...
'ToolBar','none');

ax = axes('Position',[0 0 1 1]);
axis(ax,[0 1 0 1])
axis(ax,'equal')
axis(ax,'off')
set(ax,'Color','black')

for k = 1:n

% Capture EXACT frame pixels
frame = getframe(fig);
img = frame.cdata;

% Force exact 500x500 (safety resize if needed)
img = imresize(img,[imgSize imgSize]);

% Save image
filename = fullfile(outputFolder, sprintf('frame_%03d.png',k));
imwrite(img, filename);
```

```
end
```

White Square code

```
% Number of steps
n = 100;

% Output folder
outputFolder = 'square_frames_white';
if ~exist(outputFolder, 'dir')
mkdir(outputFolder);
end

% ---- FORCE EXACT PIXEL SIZE ----
imgSize = 500; % 500x500 pixels

fig = figure('Color','black', ...
'Units','pixels', ...
'Position',[100 100 imgSize imgSize], ...
'Resize','off', ...
'MenuBar','none', ...
'ToolBar','none');

ax = axes('Position',[0 0 1 1]);
axis(ax,[0 1 0 1])
axis(ax,'equal')
axis(ax,'off')
set(ax,'Color','black')

for k = 1:n

cla(ax)

% Square size grows
side = k / n;

x = 0.5 - side/2;
y = 0.5 - side/2;

rectangle(ax,'Position',[x y side side], ...
'FaceColor','white', ...
'EdgeColor','white');

drawnow

% Capture EXACT frame pixels
frame = getframe(fig);
img = frame.cdata;

% Force exact 500x500 (safety resize if needed)
img = imresize(img,[imgSize imgSize]);

% Save image
```

```
    filename = fullfile(outputFolder, sprintf('frame_%03d.png',k));
    imwrite(img, filename);
    end
```

Conclusions/action items:

We will use this dataset to train a simple ML model that we can practice uploading up to the pi.

03/02/2026 - Old Model Heatmap

Madison Michels - Mar 02, 2026, 11:11 PM CST

Title: Old Model Heatmap

Date: 03/02/2026

Content by: Maddie

Present: Maddie

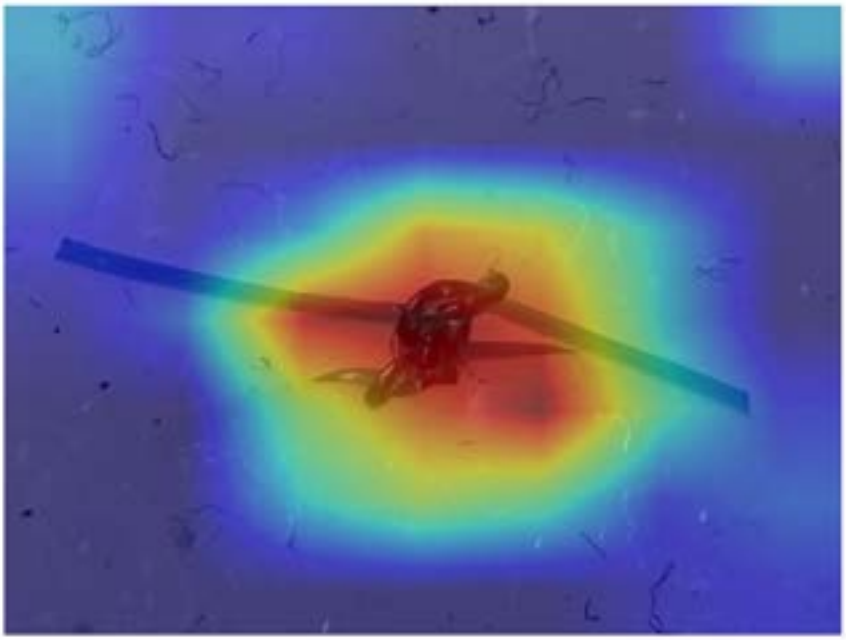
Goals: To identify where our last semester;s ML model is looking when identifying knots.

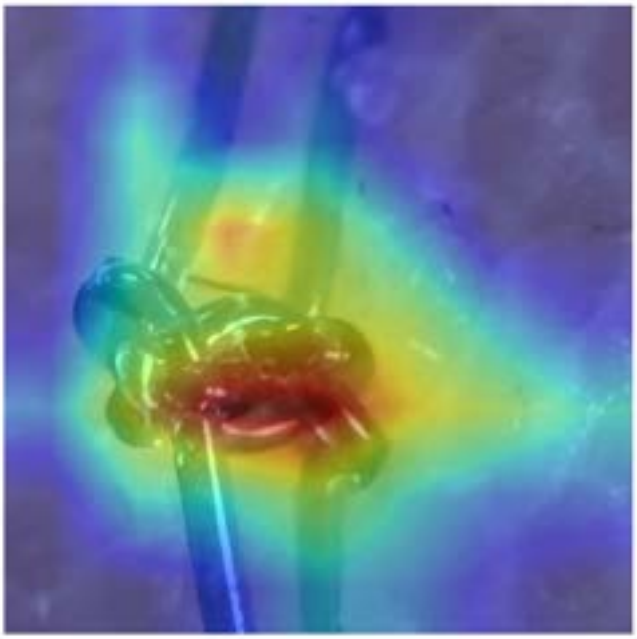
Content:

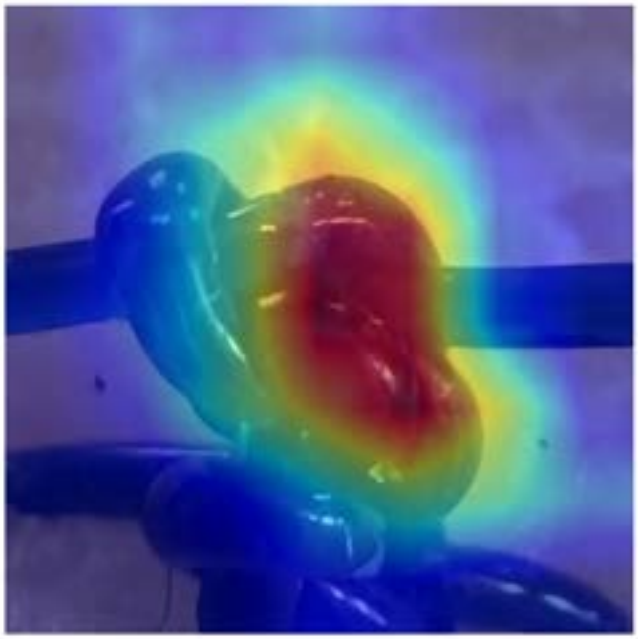
Code:

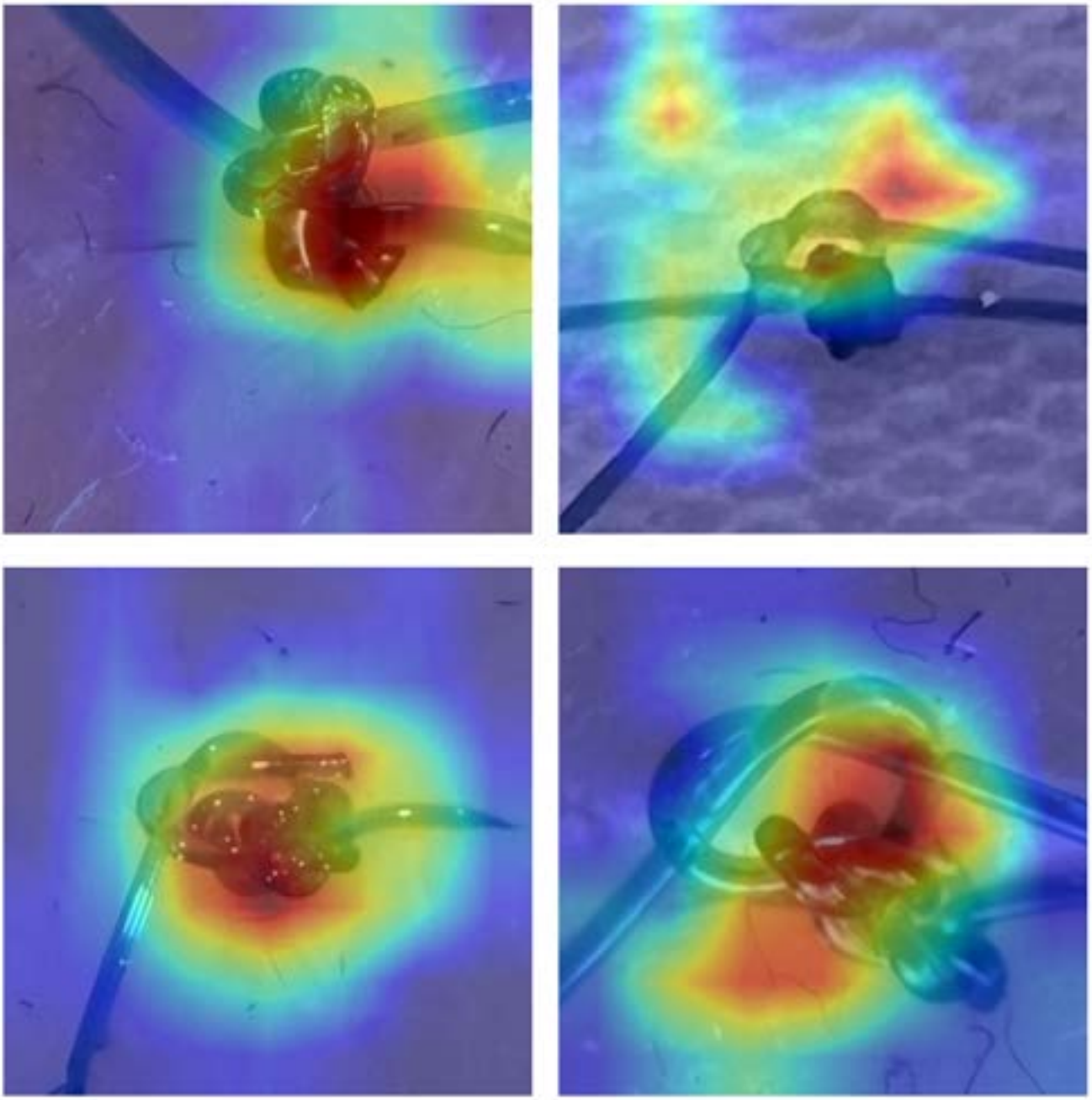
```
import torch
import numpy as np
import matplotlib.pyplot as plt
from PIL import Image
from torchvision import transforms
import cv2

# --- Use your loaded model ---
resnetmodel.eval()  # evaluation mode

# Ensure all parameters require gradients (needed for Grad-CAM)
for param in resnetmodel.parameters():
    param.requires_grad = True

# --- Pick the last convolutional layer ---
target_layer = resnetmodel.layer4[-1]  # adjust if your model differs

activations = []
gradients = []

# Hooks to capture activations and gradients
def forward_hook(module, input, output):
    activations.append(output)

def backward_hook(module, grad_input, grad_output):
    gradients.append(grad_output[0])

target_layer.register_forward_hook(forward_hook)
target_layer.register_backward_hook(backward_hook)

# --- Load and preprocess your image ---
img_path = r"C:\Users\madis\Downloads\Knot Model\testingimages\IMG_2780.jpeg"
img = Image.open(img_path).convert("RGB")
transform = transforms.Compose([
    transforms.Resize((224, 224)),
    transforms.ToTensor(),
])
input_tensor = transform(img).unsqueeze(0)
input_tensor = input_tensor.requires_grad_(True)  # allow gradients

# --- Forward pass ---
output = resnetmodel(input_tensor)
pred_class = output.argmax(dim=1)

# --- Backward pass ---
resnetmodel.zero_grad()
output[0, pred_class].backward()  # compute gradients
```

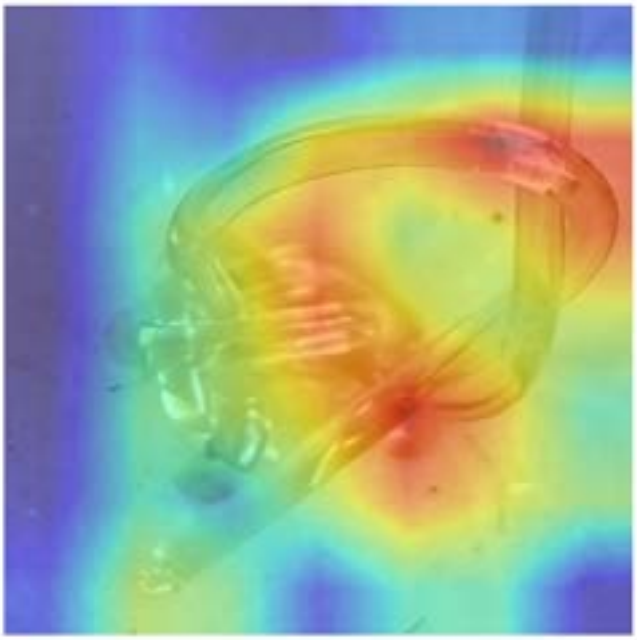
```
# --- Generate Grad-CAM heatmap ---
pooled_grads = torch.mean(gradients[0], dim=[0, 2, 3])
activation = activations[0].detach()

for i in range(pooled_grads.size(0)):
    activation[:, i, :, :] *= pooled_grads[i]

heatmap = torch.mean(activation, dim=1).squeeze()
heatmap = torch.relu(heatmap)
heatmap /= heatmap.max()  # normalize to 0-1

# --- Resize heatmap to original image size ---
heatmap = cv2.resize(heatmap.numpy(), (img.width, img.height))

# --- Overlay heatmap on image ---
plt.imshow(img)
plt.imshow(heatmap, cmap='jet', alpha=0.5)  # alpha=0.5 for transparency
plt.axis('off')
plt.show()
```

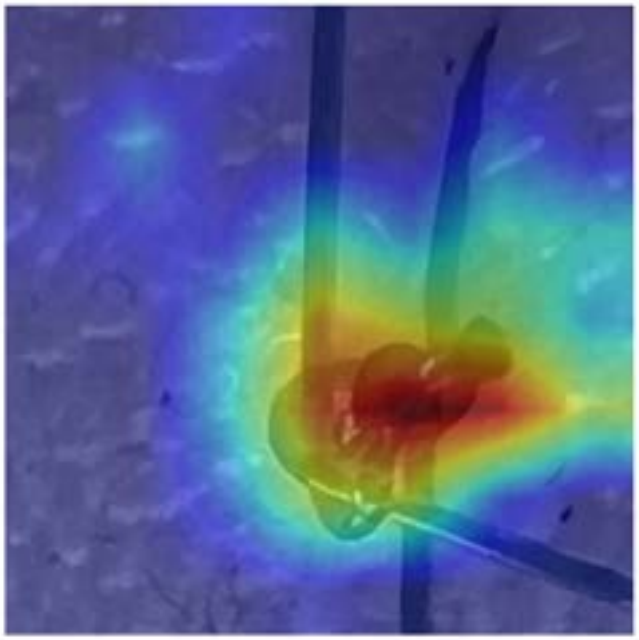












Conclusions/action items:

These are the images produced by the heatmap code from our old model. These all clearly identify the knots in the image, and focus less on the background.

4/7/2026 - K-Fold Cross Validation (x4)

Madison Michels - Apr 08, 2026, 7:50 PM CDT

Title: K-Fold Cross Validation (Binary NB, Colored NB, Contrasted, Original)

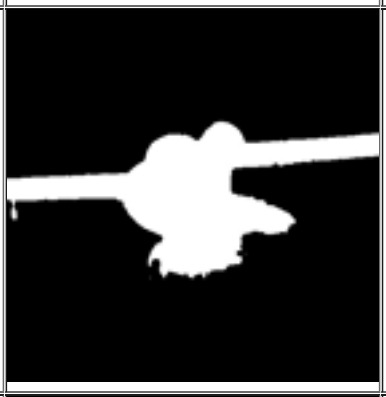
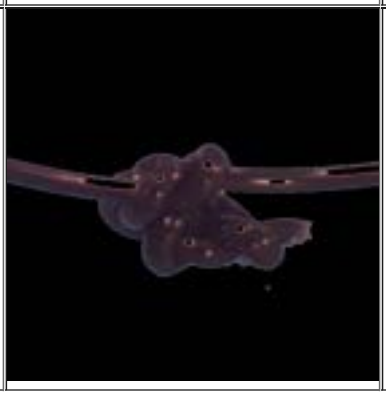
Date: 04/07/2026

Content by: Maddie

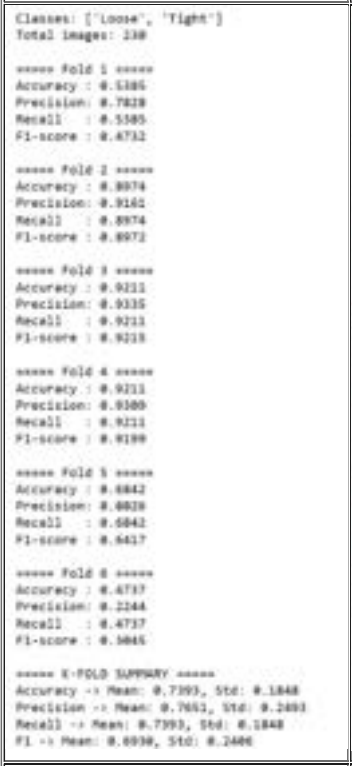
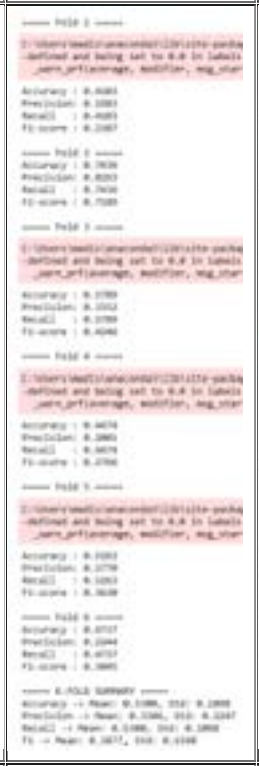
Present: Maddie

Goals: The goal of this entry is to document the process, inputs, and results of each dataset when run through K-Fold cross validation. At the end, I hope to identify which type image pre-processing will produce the best model.

Content:

Dataset Type	Example Image	Average Accuracy	Average Recall	Average F1	Average Precision	Engineering Applicability
Original		73.93%	0.7393	0.6930	0.7651	Suture pad does not appear to affect the model's decision much.
Binary with no background		53%	0.53	0.3877	0.3386	Edges only, blurry edges, some random holes, cannot use depth perception
Colored with no background		62.34%	0.6234	0.5328	0.5422	Pre processing step does not outline the color well, model cannot read image well

Contrasted (x1.5)		61.58%	0.6158	0.5166	0.5216	Most of the background is cut out, shadows impact image and light washes out suture color and edges

Original	Binary with no Background	Colored with no Background	Contrasted (x1.5)
			
Spread: 45%	Spread: 33%	Spread: 43%	Spread: 40%

Conclusions/action items:

In conclusion, the normal, unedited images received the highest averages in the k-fold cross validation. Next, I will train a ML model on this image set and evaluate its performance with heat maps and model statistics.

4/7/2026 - New Model Training Comparisons

Madison Michels - Apr 08, 2026, 12:14 PM CDT

Title: New Model Training Comparisons

Date: 4/7/2026

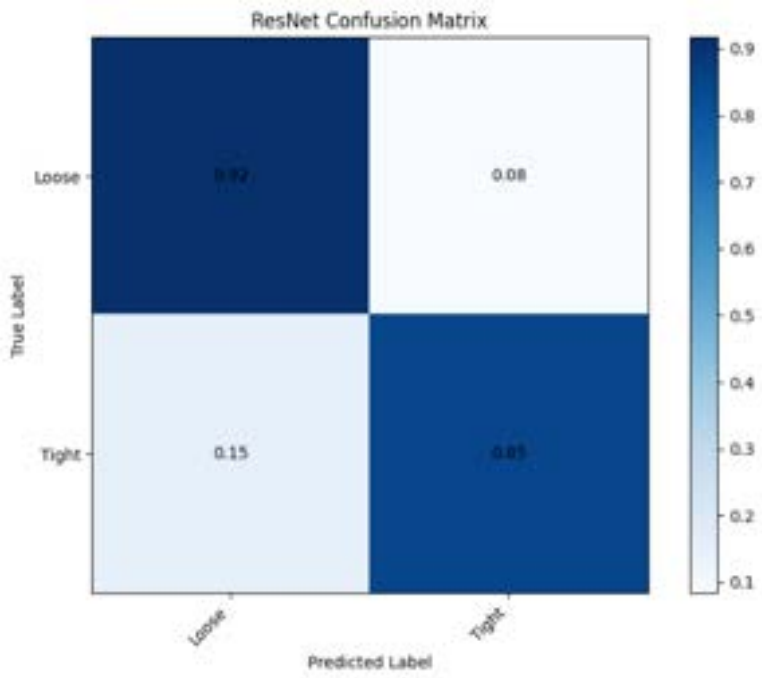
Content by: Maddie

Present: Maddie

Goals: The goal of this training was to examine the performance of the two models: binary with no background and original images.

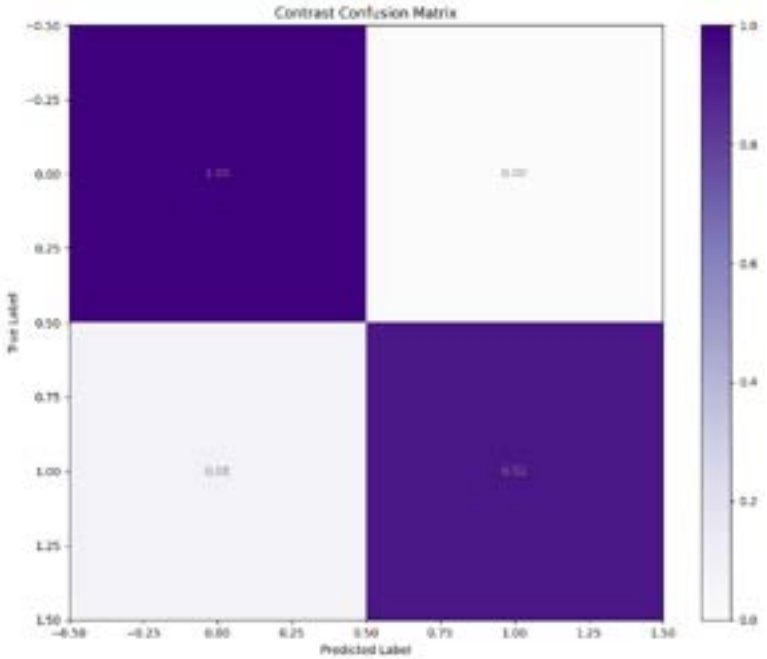
Content:

BINARY WITH NO BACKGROUND:



This confusion matrix reflects a more balanced classifier with moderate performance across both classes, achieving 88.5% overall accuracy and an F1-score of approximately 88%. The model demonstrates 85% recall (sensitivity), meaning it correctly identifies 85% of actual positive cases but misses 15%, and 92% specificity, correctly classifying 92% of negatives while generating an 8% false positive rate. With precision at roughly 91%, the model is fairly reliable when it predicts positive, though not perfect. This balanced profile suggests a practical middle-ground approach—neither overly conservative nor overly aggressive—making it suitable for applications where both false positives and false negatives carry moderate costs. The 15% false negative rate is higher than the original model's 8%, which could be concerning in critical detection scenarios, while the 8% false positive rate is manageable but still represents some wasted effort on false alarms. This model would work well in contexts where you need reasonable performance on both dimensions without extreme optimization for either precision or recall.

ORIGINAL IMAGES:



This confusion matrix reveals a highly conservative classifier with strong overall performance, achieving 96% accuracy and an F1-score of approximately 96%. The model demonstrates perfect precision (100%), meaning every positive prediction it makes is correct, with zero false positives. However, it achieves this by being selective, resulting in 92% recall—missing about 8% of actual positive cases. With 100% specificity, the model never misclassifies negatives, making it ideal for scenarios where false alarms are particularly costly or disruptive, such as situations requiring human intervention for every flagged case. The 8% false negative rate represents the trade-off for this high-confidence approach, which could be acceptable in contexts where missed positives have manageable consequences, but would be problematic in high-stakes domains like disease screening or safety monitoring where catching every positive case is critical.

Conclusions/action items:

Overall, the original image model performs the best. It has a low false positive rate, which ensures that our knots will be overly-thoroughly judged.

4/7/2026 - Filtered K-Fold Cross Validation (x4)

SADIE ROWE - Apr 17, 2026, 1:44 PM CDT

Title: K-Fold Cross Validation Filtered Images (Binary NB, Colored NB, Contrasted, Original)

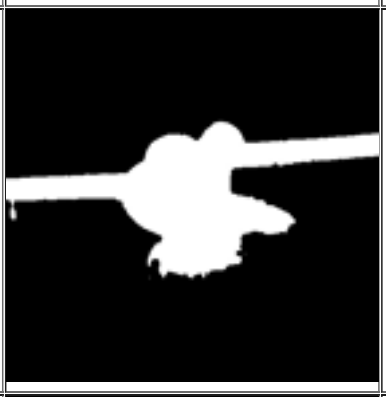
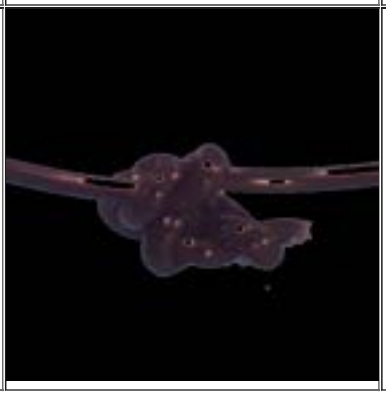
Date: 04/08/2026

Content by: Maddie

Present: Maddie

Goals: The goal of this entry is to document the process, inputs, and results of each dataset when run through K-Fold cross validation. At the end, I hope to identify which type image pre-processing will produce the best model.

Content:

Dataset Type	Example Image	Average Accuracy	Average Precision	Average Recall	Average F1	Engineering Applicability	Number of Images
Original		75.56%	77.34%	75.56%	70.25%	Suture pad does not appear to affect the model's decision much.	200
Binary with no background		67.20%	57.29%	67.20%	57.76%	Edges only, blurry edges, some random holes, cannot use depth perception	186
Colored with no background		59.09%	49.46%	59.09%	48.86%	Pre processing step does not outline the color well, model cannot read image well	198

Contrasted (x1.5)		82.05%	78.64%	82.05%	79.13%	Most of the background is cut out, shadows impact image and light washes out suture color and edges	196

Original	Binary with no Background	Colored with no Background	Contrasted (x1.5)
===== Fold 1 ===== Accuracy : 0.8529 Precision: 0.8824 Recall : 0.8529 F1-score : 0.8451 ===== Fold 2 ===== Accuracy : 0.5294 Precision: 0.2803 Recall : 0.5294 F1-score : 0.3665 ===== Fold 3 ===== Accuracy : 0.7273 Precision: 0.7798 Recall : 0.7273 F1-score : 0.7564 ===== Fold 4 ===== Accuracy : 0.4848 Precision: 0.7585 Recall : 0.4848 F1-score : 0.3476 ===== Fold 5 ===== Accuracy : 0.9394 Precision: 0.9394 Recall : 0.9394 F1-score : 0.9394 ===== Fold 6 ===== Accuracy : 1.0000 Precision: 1.0000 Recall : 1.0000 F1-score : 1.0000 ===== K-FOLD SUMMARY ===== Accuracy -> Mean: 0.7556, Std: 0.1950 Precision -> Mean: 0.7734, Std: 0.2360 Recall -> Mean: 0.7556, Std: 0.1950 F1 -> Mean: 0.7025, Std: 0.2595	===== Fold 1 ===== Accuracy : 0.9355 Precision: 0.9355 Recall : 0.9355 F1-score : 0.9355 ===== Fold 2 ===== Accuracy : 0.5806 Precision: 0.3371 Recall : 0.5806 F1-score : 0.4266 ===== Fold 3 ===== Accuracy : 0.4839 Precision: 0.2342 Recall : 0.4839 F1-score : 0.3156 ===== Fold 4 ===== Accuracy : 0.9355 Precision: 0.9423 Recall : 0.9355 F1-score : 0.9348 ===== Fold 5 ===== Accuracy : 0.6452 Precision: 0.7046 Recall : 0.6452 F1-score : 0.5737 ===== Fold 6 ===== Accuracy : 0.4516 Precision: 0.2040 Recall : 0.4516 F1-score : 0.2810 ===== K-FOLD SUMMARY ===== Accuracy -> Mean: 0.6720, Std: 0.1966 Precision -> Mean: 0.5729, Std: 0.3213 Recall -> Mean: 0.6720, Std: 0.1966 F1 -> Mean: 0.5779, Std: 0.2693	===== Fold 1 ===== Accuracy : 0.5455 Precision: 0.2975 Recall : 0.5455 F1-score : 0.3850 ===== Fold 2 ===== Accuracy : 0.5758 Precision: 0.5496 Recall : 0.5758 F1-score : 0.5021 ===== Fold 3 ===== Accuracy : 0.6970 Precision: 0.8135 Recall : 0.6970 F1-score : 0.6699 ===== Fold 4 ===== Accuracy : 0.7576 Precision: 0.8352 Recall : 0.7576 F1-score : 0.7403 ===== Fold 5 ===== Accuracy : 0.5152 Precision: 0.2654 Recall : 0.5152 F1-score : 0.3503 ===== Fold 6 ===== Accuracy : 0.4545 Precision: 0.2066 Recall : 0.4545 F1-score : 0.2841 ===== K-FOLD SUMMARY ===== Accuracy -> Mean: 0.5909, Std: 0.1046 Precision -> Mean: 0.4946, Std: 0.2566 Recall -> Mean: 0.5909, Std: 0.1046 F1 -> Mean: 0.4886, Std: 0.1674	===== Fold 1 ===== Accuracy : 0.9394 Precision: 0.9481 Recall : 0.9394 F1-score : 0.9393 ===== Fold 2 ===== Accuracy : 0.9091 Precision: 0.9259 Recall : 0.9091 F1-score : 0.9096 ===== Fold 3 ===== Accuracy : 0.9394 Precision: 0.9470 Recall : 0.9394 F1-score : 0.9397 ===== Fold 4 ===== Accuracy : 0.6667 Precision: 0.6739 Recall : 0.6667 F1-score : 0.6537 ===== Fold 5 ===== Accuracy : 0.5312 Precision: 0.2822 Recall : 0.5312 F1-score : 0.3686 ===== Fold 6 ===== Accuracy : 0.9375 Precision: 0.9437 Recall : 0.9375 F1-score : 0.9367 ===== K-FOLD SUMMARY ===== Accuracy -> Mean: 0.8205, Std: 0.1618 Precision -> Mean: 0.7864, Std: 0.2057 Recall -> Mean: 0.8205, Std: 0.1618 F1 -> Mean: 0.7913, Std: 0.2147
Spread: 52% STD: 0.1950	Spread: 48% STD: 0.1966	Spread: 30% STD: 0.1046	Spread: 40% STD: 0.1618

Original Images

The model performs reasonably well on the original images, suggesting that it is effectively leveraging a combination of **color, texture, and contextual cues**. From an engineering standpoint, this indicates that the model is not solely relying on the suture itself but also on **surrounding spatial context** (e.g., pad texture, lighting gradients, and subtle shadows). While this improves robustness, it also introduces a risk: the model may learn **spurious correlations** (like consistent background patterns) rather than purely focusing on suture tightness. This explains why performance is decent but not optimal—there is useful signal, but also noise and potential overfitting to irrelevant features.

Binary (No Background)

Binary images strip the data down to edge-level information, removing all grayscale and color gradients. In theory, this should emphasize shape and structure, which are important for distinguishing tight vs. loose sutures. However, the drop in performance suggests that the binarization process introduces information loss and artifacts:

- Blurry or broken edges reduce geometric clarity
- Holes and noise disrupt continuity of the suture line
- Loss of intensity gradients removes depth cues (e.g., how tightly the suture presses into the pad)

From an engineering perspective, this tells us that edge information alone is insufficient—the model depends heavily on continuous spatial and intensity variations, not just outlines.

Colored (No Background)

Removing the background while preserving color should, in theory, isolate the region of interest. However, the poor performance indicates that the preprocessing pipeline likely introduces color distortion or segmentation errors:

- Inconsistent masking may remove parts of the suture
- Color boundaries may become unnatural or noisy
- The model loses contextual contrast between suture and background

Engineering-wise, this highlights that segmentation quality is critical. A poorly executed preprocessing step can degrade the signal more than it helps. It also suggests that the model benefits from relative color contrast, not just absolute color values—removing the background eliminates that reference.

Contrasted (×1.5)

This configuration yields the best performance, which is a strong indication that feature amplification is beneficial. Increasing contrast enhances:

- Edge sharpness → clearer structural definition
- Local gradients → better perception of tightness and indentation
- Foreground-background separation → reduces irrelevant noise

However, the noted downsides (shadow amplification and highlight washout) point to a key tradeoff:

- Over-contrast can clip important details, especially subtle color differences in sutures
- Lighting inconsistencies become more pronounced, potentially introducing bias

From an engineering standpoint, this suggests the model is highly sensitive to mid-level features (edges + gradients), and contrast enhancement improves their visibility. It also implies that a more controlled approach (e.g., adaptive contrast or histogram equalization) could further improve performance without introducing artifacts.

Conclusions/action items:

In conclusion, the normal, unedited images received the highest averages in the k-fold cross validation. Next, I will train a ML model on this image set and evaluate its performance with heat maps and model statistics.

4/7/2026 - New Model Training Comparisons

Madison Michels - Apr 14, 2026, 8:36 PM CDT

Title: New Model Training Comparisons

Date: 4/7/2026

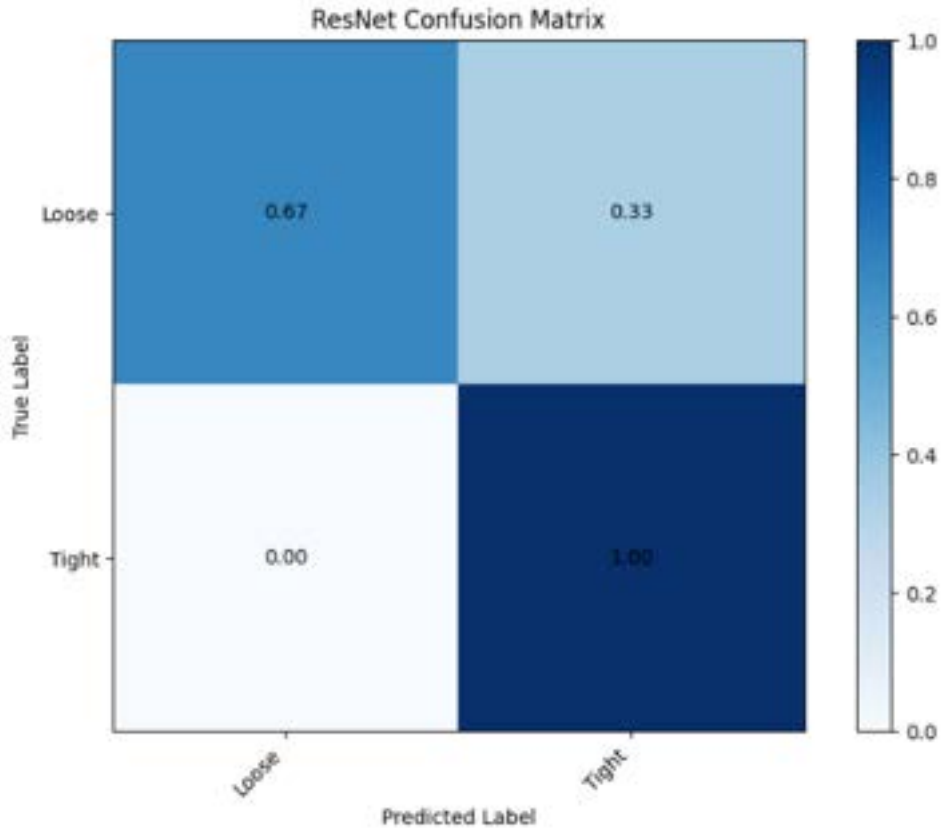
Content by: Maddie

Present: Maddie

Goals: The goal of this training was to examine the performance of the two models: binary with no background and original images.

Content:

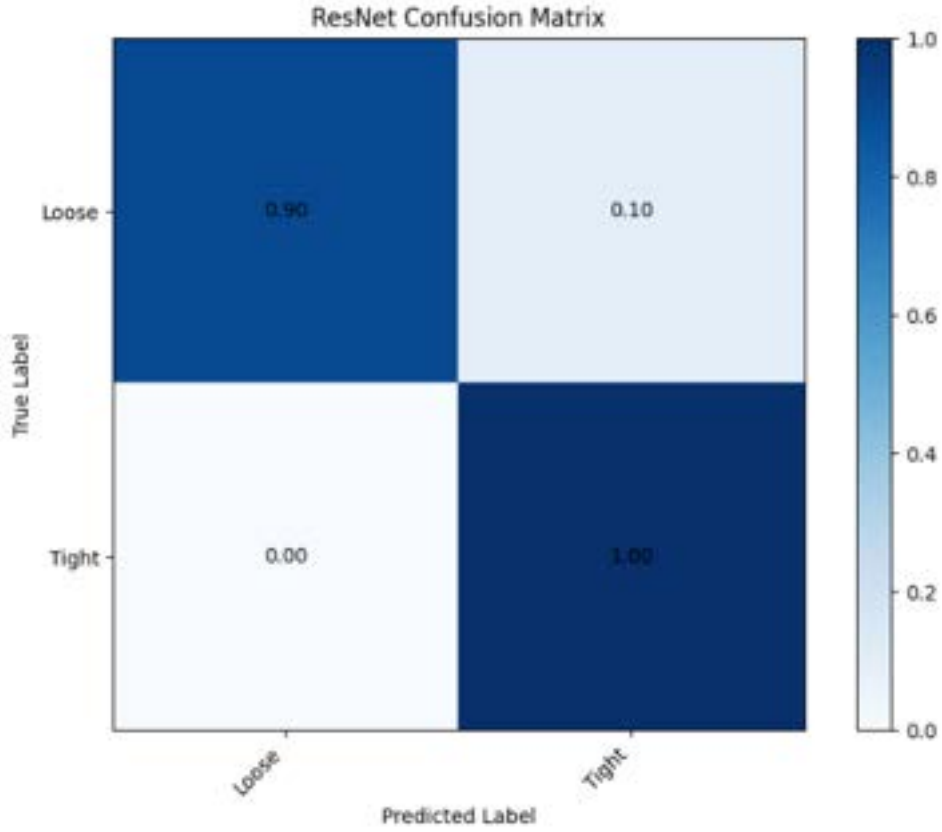
BINARY WITH NO BACKGROUND: 83.5% Accuracy



This confusion matrix reveals that the ResNet model performs asymmetrically across the two classes. It achieves perfect classification of "Tight" samples (1.00), meaning it never misidentifies a tight instance — a strong result. However, it struggles with "Loose" samples, correctly identifying only 67% of them while misclassifying 33% as "Tight." This pattern suggests the model has developed a bias toward predicting "Tight," possibly due to class imbalance in the training data or because the visual features distinguishing loose cases are more ambiguous and harder for the network to learn. In practical terms, the cost of this bias depends heavily on the application: if "Loose" cases represent a critical outcome (e.g., a loose mechanical component or a medical condition), a one-in-three miss rate could be consequential. Overall, while the model shows promise — particularly its flawless tight-class recall — it would benefit from techniques like oversampling the loose class, adjusting class weights during training, or tuning the classification threshold to reduce false negatives on the "Loose" category.

ORIGINAL IMAGES: 95% Accuracy

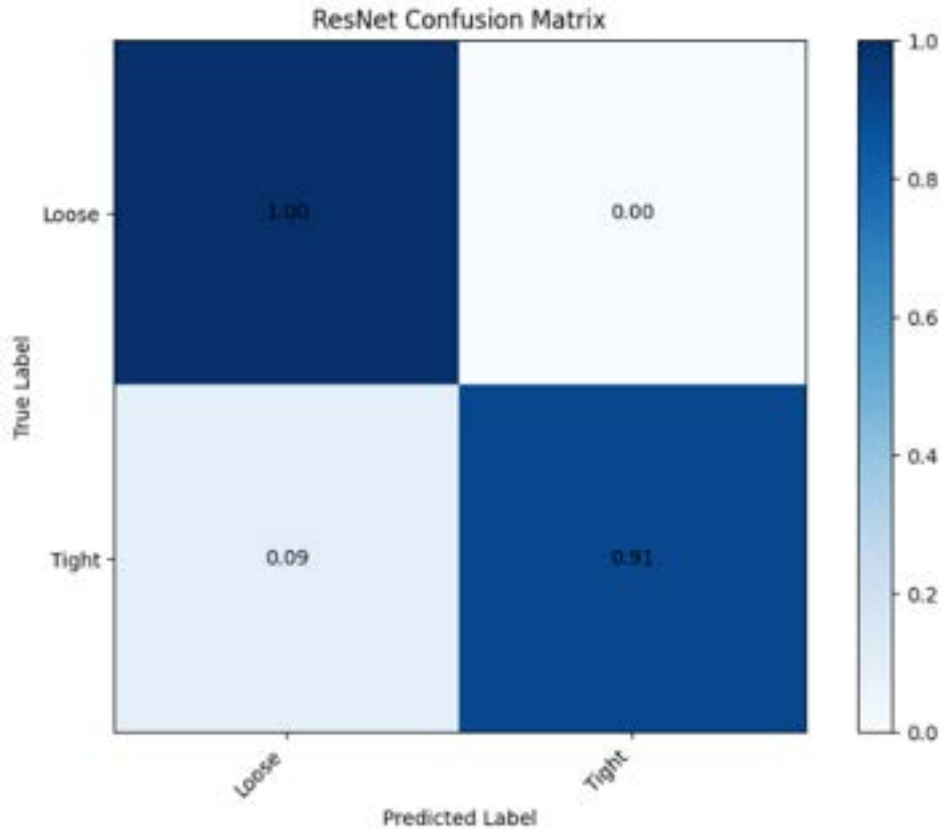
	precision	recall	f1-score	support
0	1.00	0.90	0.95	10
1	0.91	1.00	0.95	10
accuracy			0.95	20
macro avg	0.95	0.95	0.95	20
weighted avg	0.95	0.95	0.95	20



The model again shows strong and balanced performance, achieving an overall accuracy of 95% on the dataset. For class 0, it attains perfect precision (1.00), meaning all predictions labeled as class 0 are correct, though its recall (0.90) indicates it misses a small portion of actual class 0 instances. In contrast, for class 1, the model achieves perfect recall (1.00), successfully identifying all true instances, while its precision (0.91) suggests a few false positives when predicting this class. The F1-scores for both classes are identical at 0.95, reflecting an effective balance between precision and recall despite these small trade-offs. Additionally, the macro and weighted averages are the same, indicating consistent performance across both classes and no noticeable class bias, which aligns with the equal support of 10 samples per class. Overall, the model performs reliably, with only minor differences in how it handles false positives versus false negatives between the two classes.

CINTRASTED IMAGES: 95% Accuracy

	precision	recall	f1-score	support
0	0.92	1.00	0.96	11
1	1.00	0.91	0.95	11
accuracy			0.95	22
macro avg	0.96	0.95	0.95	22
weighted avg	0.96	0.95	0.95	22



The model demonstrates strong and well-balanced classification performance across both classes (“Loose” and “Tight”), achieving an overall accuracy of 95% on the test set. It shows perfect precision (1.00) for class 1, meaning that whenever it predicts “Tight,” it is always correct, though its slightly lower recall (0.91) indicates it misses a small number of true “Tight” instances. Conversely, for class 0, the model achieves perfect recall (1.00), successfully identifying all “Loose” samples, with a slightly lower precision (0.92), implying a few false positives. The F1-scores for both classes (0.95–0.96) are high and nearly identical, reflecting a good balance between precision and recall. Additionally, the macro and weighted averages are nearly the same, suggesting the model performs consistently across classes without bias, which is expected given the balanced dataset (equal support of 11 samples per class). Overall, the model is reliable, with only minor trade-offs between precision and recall depending on the class.

Overall Engineering Insight

Across all variations, the model appears to rely on a combination of:

- Edge definition (structure)
- Intensity gradients (depth/tightness cues)
- Contextual contrast (foreground vs. background separation)

The key takeaway is that removing information is more harmful than refining it. Methods that preserve and enhance natural image statistics (like contrast adjustment) outperform those that aggressively simplify the image (binary or segmentation).

In practical terms, the best-performing pipeline should:

- Retain grayscale and color gradients
- Enhance edge clarity without distortion
- Avoid introducing artifacts through aggressive preprocessing

Conclusions/action items:

Overall, the normal model and contrasted modelss performed the best. These likely did well because the black and white blurred too much of the image out an obscured key features to train the model.

3/17/2026 - Full Integration Code

Madison Michels - Mar 17, 2026, 7:17 PM CDT

Title: Full Integration Code

Date: 3/17/2026

Content by: Maddie

Present: Maddie

Goals: The goal of this code is to create the full user workflow in the system.

Content:

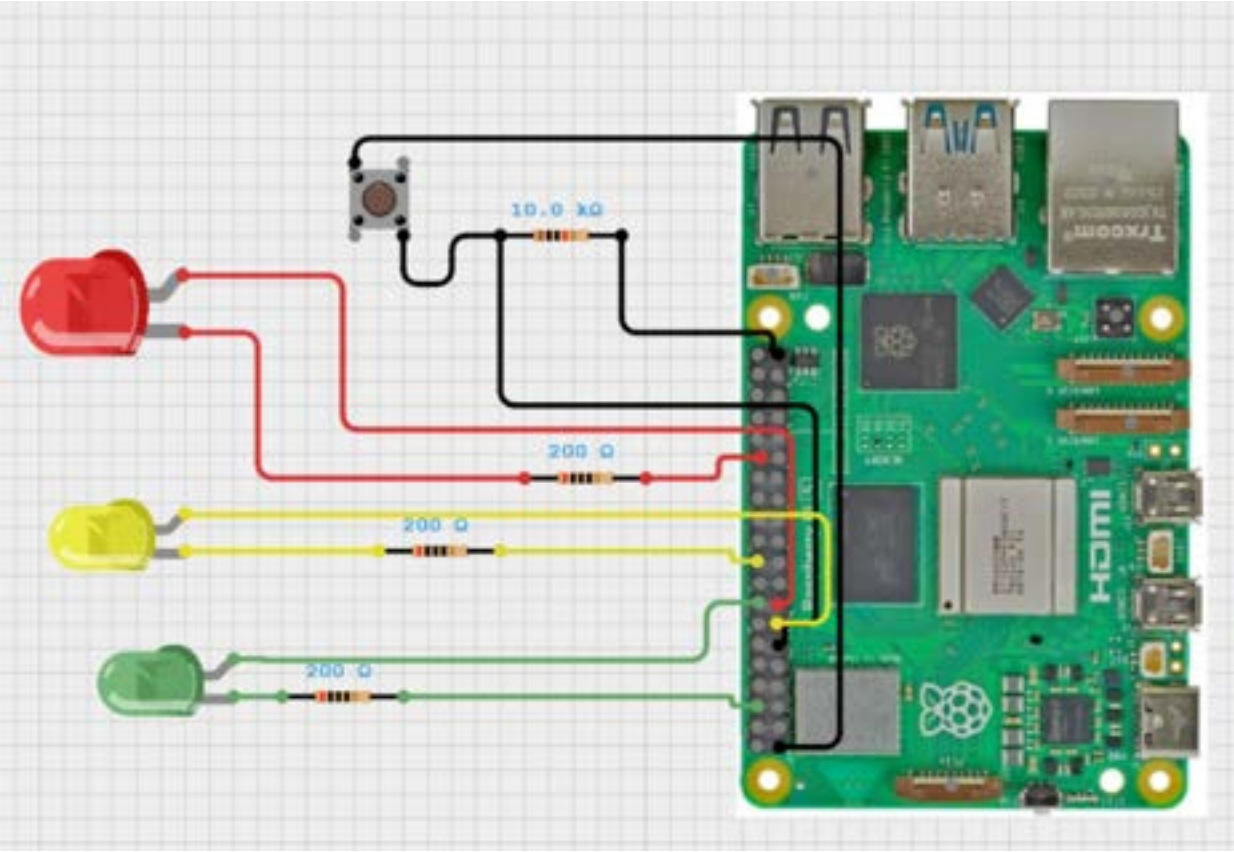
```
#!/usr/bin/env python3
"""Capture an image via button press, run model, signal result via LEDs,
and save to ./TIGHT/ or ./LOOSE/ based on prediction."""
import sys
import subprocess
import time
from pathlib import Path

import torch
from torchvision import transforms
from PIL import Image
import RPi.GPIO as GPIO

# --- Config ---
MODEL_PATH = "/home/knotorious-five/squaredetect/top_and_side_resnet_full.pth"
FOLDERS = {
    "tight": Path(__file__).parent / "TIGHT",
    "loose": Path(__file__).parent / "LOOSE",
}
CLASS_MAPPING = {"Loose": 0, "Tight": 1}
DATA_TRANSFORMS = transforms.Compose([
    transforms.Resize((256, 256)),
    transforms.ToTensor(),
])
LOG_FILE = Path(__file__).parent / "results.csv"

# --- GPIO Pin Config ---
PIN_BUTTON = 17
PIN_LED_PROCESSING = 27   # LED1: on while model runs
PIN_LED_TIGHT = 22        # LED2: on if prediction is Tight
PIN_LED_LOOSE = 23        # LED3: on if prediction is Loose


def setup_gpio():
    """Configure GPIO pins."""
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(PIN_BUTTON, GPIO.IN, pull_up_down=GPIO.PUD_UP)
    GPIO.setup(PIN_LED_PROCESSING, GPIO.OUT, initial=GPIO.LOW)
    GPIO.setup(PIN_LED_TIGHT, GPIO.OUT, initial=GPIO.LOW)
    GPIO.setup(PIN_LED_LOOSE, GPIO.OUT, initial=GPIO.LOW)


def all_leds_off():
    """Turn all LEDs off."""
    GPIO.output(PIN_LED_PROCESSING, GPIO.LOW)
    GPIO.output(PIN_LED_TIGHT, GPIO.LOW)
    GPIO.output(PIN_LED_LOOSE, GPIO.LOW)
```

```
def capture_image(output_path: str) -> str:
    """Capture an image using the RPi camera.
    Tries picamera2 first, falls back to libcamera-still CLI.
    """
    try:
        from picamera2 import Picamera2
        cam = Picamera2()
        config = cam.create_still_configuration()
        cam.configure(config)
        cam.start()
        cam.capture_file(output_path)
        cam.stop()
        cam.close()
        print(f"Captured image (picamera2): {output_path}")
        return output_path
    except ImportError:
        print("picamera2 not available, falling back to libcamera-still...")

    result = subprocess.run(
        ["libcamera-still", "-o", output_path, "--nopreview", "-t", "1000"],
        capture_output=True, text=True,
    )
    if result.returncode != 0:
        print(f"libcamera-still failed:\n{result.stderr}", file=sys.stderr)
        sys.exit(1)
    print(f"Captured image (libcamera-still): {output_path}")
    return output_path


def load_image(image_path: str) -> torch.Tensor:
    """Load and preprocess an image for the model."""
    img = Image.open(image_path).convert("RGB")
    img = DATA_TRANSFORMS(img)
    img = img.unsqueeze(0)
    return img


def predict(image_path: str, model: torch.nn.Module) -> tuple[str, float]:
    """Run model inference on an image and return (class_name, confidence)."""
    img = load_image(image_path)
    with torch.no_grad():
        output = model(img)
    probabilities = torch.exp(output)
    prediction = probabilities.max(dim=1)[1].item()
    confidence = probabilities.max(dim=1)[0].item()
    class_name = next(k for k, v in CLASS_MAPPING.items() if v == prediction)
    return class_name, confidence


def get_next_index(images_dir: Path) -> int:
    """Find highest existing image index to avoid collisions."""
    existing = list(images_dir.glob("*.jpg"))
    if existing:
        return max(int(f.stem.split("_")[0]) for f in existing) + 1
    return 1


def log_result(image_path: str, class_name: str, confidence: float):
    """Append prediction result to the CSV log file."""
    if not LOG_FILE.exists():
        LOG_FILE.write_text("image_path,prediction,confidence\n")
    with LOG_FILE.open("a") as f:
        f.write(f"{image_path},{class_name},{confidence:.4f}\n")
```

```
def run_capture_and_predict(model: torch.nn.Module):
    """Full pipeline: capture, predict, save, and signal via LEDs."""
    all_leds_off()

    # Capture to a temporary path first
    tmp_path = str(Path(__file__).parent / "tmp_capture.jpg")
    capture_image(tmp_path)

    # Turn on processing LED while model runs
    GPIO.output(PIN_LED_PROCESSING, GPIO.HIGH)
    print("Running model...")
    class_name, confidence = predict(tmp_path, model)
    print(f"Result: {class_name} (confidence: {confidence:.4f})")

    # Save to correct folder
    category = class_name.lower()
    images_dir = FOLDERS[category]
    images_dir.mkdir(exist_ok=True)
    next_index = get_next_index(images_dir)
    final_path = str(images_dir / f"{next_index}_{category}.jpg")
    Path(tmp_path).rename(final_path)
    print(f"Image saved: {final_path}")

    # Log result
    log_result(final_path, class_name, confidence)

    # Turn off processing LED, turn on result LED
    GPIO.output(PIN_LED_PROCESSING, GPIO.LOW)
    if category == "tight":
        GPIO.output(PIN_LED_TIGHT, GPIO.HIGH)
    else:
        GPIO.output(PIN_LED_LOOSE, GPIO.HIGH)


def main():
    setup_gpio()

    # Load model once at startup
    print("Loading model...")
    model = torch.load(MODEL_PATH, weights_only=False, map_location="cpu")
    model.eval()
    print("Model loaded. Waiting for button press...\n")

    try:
        while True:
            # Wait for button press (active LOW with pull-up)
            GPIO.wait_for_edge(PIN_BUTTON, GPIO.FALLING)
            time.sleep(0.05)  # Debounce
            if GPIO.input(PIN_BUTTON) == GPIO.LOW:
                print("Button pressed!")
                run_capture_and_predict(model)
                print("\nWaiting for next button press...")

    except KeyboardInterrupt:
        print("\nExiting.")
    finally:
        all_leds_off()
        GPIO.cleanup()


if __name__ == "__main__":
    main()
```

Key notes:

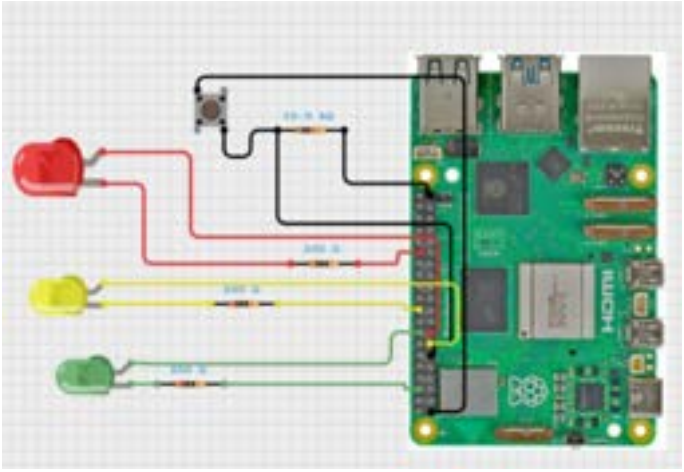
- **GPIO setup**
  - Button on pin 17 (pull-up, active LOW)
  - LED1 (white) on 27
  - LED2 (green-tight) on 22
  - LED3 (red-loose) on 23
- **Button loop** — `GPIO.wait_for_edge` blocks until a press, with a 50ms debounce check to filter noise.
- **LED logic** — LED1 turns on when the model starts running and off when done. LED2 or LED3 then turns on based on the result, staying on until the next button press.
- **Model determines folder** — the user prompt is gone; the model's prediction decides whether the image goes to `TIGHT/` or `LOOSE/`, with the same auto-incrementing filename logic.
- **CSV logging** — results are appended to `results.csv` alongside the image path and confidence score.
- **Model loads once** at startup rather than on every capture, so button presses are fast.



Conclusions/action items:

We will run this from the pi's terminal to allow the user to interact with the pi and training system.

Madison Michels - Apr 08, 2026, 2:00 PM CDT



Download

1773792959925.png (354 kB)

3/17/2026 - Image Capture and Save Code

Madison Michels - Mar 17, 2026, 7:23 PM CDT

Title: Image Capture and Save Code

Date: 3/17/2026

Content by: Maddie

Present: Maddie

Goals: The goal of this code is to be able to take new images to retrain our model on higher quality, consistent images.

Content:

```
#!/usr/bin/env python3
"""Capture an image from RPi camera and save to ./TIGHT/ or ./LOOSE/."""
import sys
import subprocess
from pathlib import Path

# --- Config ---
FOLDERS = {
    "tight": Path(__file__).parent / "TIGHT",
    "loose": Path(__file__).parent / "LOOSE",
}


def prompt_category() -> str:
    """Prompt the user to select a category."""
    while True:
        choice = input("Save to TIGHT or LOOSE? [t/l]: ").strip().lower()
        if choice in ("t", "tight"):
            return "tight"
        elif choice in ("l", "loose"):
            return "loose"
        print("Invalid input. Please enter 't' for TIGHT or 'l' for LOOSE.")


def capture_image(output_path: str) -> str:
    """Capture an image using the RPi camera.
    Tries picamera2 first, falls back to libcamera-still CLI.
    """
    try:
        from picamera2 import Picamera2
        cam = Picamera2()
        config = cam.create_still_configuration()
        cam.configure(config)
        cam.start()
        cam.capture_file(output_path)
        cam.stop()
        cam.close()
        print(f"Captured image (picamera2): {output_path}")
        return output_path
    except ImportError:
        print("picamera2 not available, falling back to libcamera-still...")

    result = subprocess.run(
        ["libcamera-still", "-o", output_path, "--nopreview", "-t", "1000"],
        capture_output=True, text=True,
    )
    if result.returncode != 0:
        print(f"libcamera-still failed:\n{result.stderr}", file=sys.stderr)
```

```
        sys.exit(1)
    print(f"Captured image (libcamera-still): {output_path}")
    return output_path


def main():
    # Prompt for category
    category = prompt_category()
    images_dir = FOLDERS[category]

    # Ensure directory exists
    images_dir.mkdir(exist_ok=True)

    # Find highest existing index to avoid collisions
    existing = list(images_dir.glob("*.jpg"))
    if existing:
        highest = max(int(f.stem.split("_")[0]) for f in existing)
    else:
        highest = 0
    image_path = str(images_dir / f"{highest + 1}_{category}.jpg")

    # Capture
    capture_image(image_path)

    print(f"\nImage saved: {image_path}")

if __name__ == "__main__":
    main()
```

Conclusions/action items:

We can utilize this code by running the code within the pi terminal. The code saves images to tight or loose once a picture is taken. It will prompt the user to select whether this knot image should be classified as tight or loose and name/number it accordingly.

3/17/2026 - Capture and Predict

Madison Michels - Mar 17, 2026, 7:34 PM CDT

Title: Capture and Predict

Date: 3/17/2026

Content by: Maddie

Present: Maddie

Goals: The goal of this code is to take an image with the Pi camera and then run that image through the model. It will then save the image, prediction, confidence level, date, and time in a csv file on the pi for traceability.

Content:

```
#!/usr/bin/env python3
"""Capture an image from RPi camera, save to ./images/, and run the model."""

import os
import sys
import subprocess
from datetime import datetime
from pathlib import Path

import torch
import torchvision.models as models
from torchvision import transforms
from PIL import Image

# --- Config ---
MODEL_PATH = "/home/knotorious-five/squaredetect/top_and_side_resnet_full.pth"
IMAGES_DIR = Path(__file__).parent / "images"
CLASS_MAPPING = {"Loose": 0, "Tight": 1}

DATA_TRANSFORMS = transforms.Compose([
    transforms.Resize((256, 256)),
    transforms.ToTensor(),
])


def capture_image(output_path: str) -> str:
    """Capture an image using the RPi camera.

    Tries picamera2 first, falls back to libcamera-still CLI.
    """
    try:
        from picamera2 import Picamera2

        cam = Picamera2()
        config = cam.create_still_configuration()
        cam.configure(config)
        cam.start()
        cam.capture_file(output_path)
        cam.stop()
        cam.close()
        print(f"Captured image (picamera2): {output_path}")
        return output_path

    except ImportError:
        print("picamera2 not available, falling back to libcamera-still...")

    result = subprocess.run(
        ["libcamera-still", "-o", output_path, "--nopreview", "-t", "1000"],
```

```
        capture_output=True, text=True,
    )
    if result.returncode != 0:
        print(f"libcamera-still failed:\n{result.stderr}", file=sys.stderr)
        sys.exit(1)

    print(f"Captured image (libcamera-still): {output_path}")
    return output_path


def load_image(image_path: str) -> torch.Tensor:
    """Load and preprocess an image for the model."""
    img = Image.open(image_path).convert("RGB")
    img = DATA_TRANSFORMS(img)
    img = img.unsqueeze(0)
    return img


def predict(image_path: str, model: torch.nn.Module) -> tuple[str, float]:
    """Run model inference on an image and return (class_name, confidence)."""
    img = load_image(image_path)
    with torch.no_grad():
        output = model(img)
    probabilities = torch.exp(output)
    prediction = probabilities.max(dim=1)[1].item()
    confidence = probabilities.max(dim=1)[0].item()

    class_name = next(k for k, v in CLASS_MAPPING.items() if v == prediction)
    return class_name, confidence


def main():
    # Ensure images directory exists
    IMAGES_DIR.mkdir(exist_ok=True)

    # Generate timestamped filename
    timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")
    image_path = str(IMAGES_DIR / f"capture_{timestamp}.jpg")

    # Capture
    capture_image(image_path)

    # Load model
    print("Loading model...")
    model = torch.load(MODEL_PATH, weights_only=False, map_location="cpu")
    model.eval()

    # Predict
    class_name, confidence = predict(image_path, model)
    print(f"\nResult: {class_name} (confidence: {confidence:.4f})")
    print(f"Image saved: {image_path}")


if __name__ == "__main__":
    main()
```

Conclusions/action items:

This code will be run from the pi's terminal and will be setup to execute with the button push.

04/14/2026 - Model Testing Protocol

Madison Michels - Apr 15, 2026, 12:42 PM CDT

Title: Model Testing Protocol

Date: 4/14/2026

Content by: Maddie

Present: Maddie

Goals: This entry aims to outline a testing procedure for evaluating ech model's performance. We also will record the results here.

Protocol:

1. Place 224 (112 tight, 112 loose) images in a consistent testing folder
2. Load in the models
    - Black and white no background
    - Original images
    - Contrasted
3. Perform required imag preprocessing for each model individually
4. Run the 224 images through each model
5. Save the predicted label, true label, path file, and confidence level to a csv

Results:

	BW	Contrast	Normal	Original
Total Correct	123	105	116	141
Accuracy Percentage	54.91%	46.88%	51.79%	62.95%
Average Confidence	0.8466958	0.8862217	0.9004925	0.7848186
Tight Correct	49	26	103	101
Loose Correct	74	79	13	40

Num Total: 224
Num Tight: 112
Num Loose: 112

Conclusions/action items:

In conclusion, all of the models struggled significantly on the loose image classifications. The black and white performed the best on loose because the edge detection highlights the holes in the suture knots, but cannot evaluate a tight knot because all of the features are ambiguous. The normal images trained the best model because they retained image features and knot features that are key to the model's decision making.

4/17/2026 - Full Workflow Normal Model Testing (OG)

Madison Michels - Apr 17, 2026, 1:41 PM CDT

Title: Full Workflow Normal Model Testing

Date: 4/17/2026

Content by: Maddie

Present: Maddie and Kate

Goals: The goal of this entry is to evaluate the performance of our model by running it through the pi workflow and recording

Content:

Image	Prediction	Confidence
Tight 1	Tight	95.69%
Tight 2	Tight	92.39%
Tight 3	Tight	87.32%
Tight 4	Tight	81.25%
Tight 5	Tight	89.9%
Tight 6	Tight	93.26%
Tigh 7	Tight	94.56%
Tight 8	Tight	96.67%
Tight 9	Tight	96.98%
Tight 10	Tight	97.82%
Loose 1	Tight (Loose)	60.55%
Loose 2	Tight (loose)	79.40%
Loose 3	Tight	89.22%
Loose 4	Tight	96.14%
Loose 5	Tight (Loose)	81.63%
Loose 6	Tight (Loose)	84.80%
Loose 7	Tight	94.59%
Loose 8	Tight (Loose)	74.67%
Loose 9	Tight	89.58%
Loose 10	Tight	97.29%

Conclusions/action items:

The model's overall accuracy is 75%.

3/15/2026 - Design Innovation Lab Trainings Completed

Madison Michels - Mar 15, 2026, 6:05 PM CDT

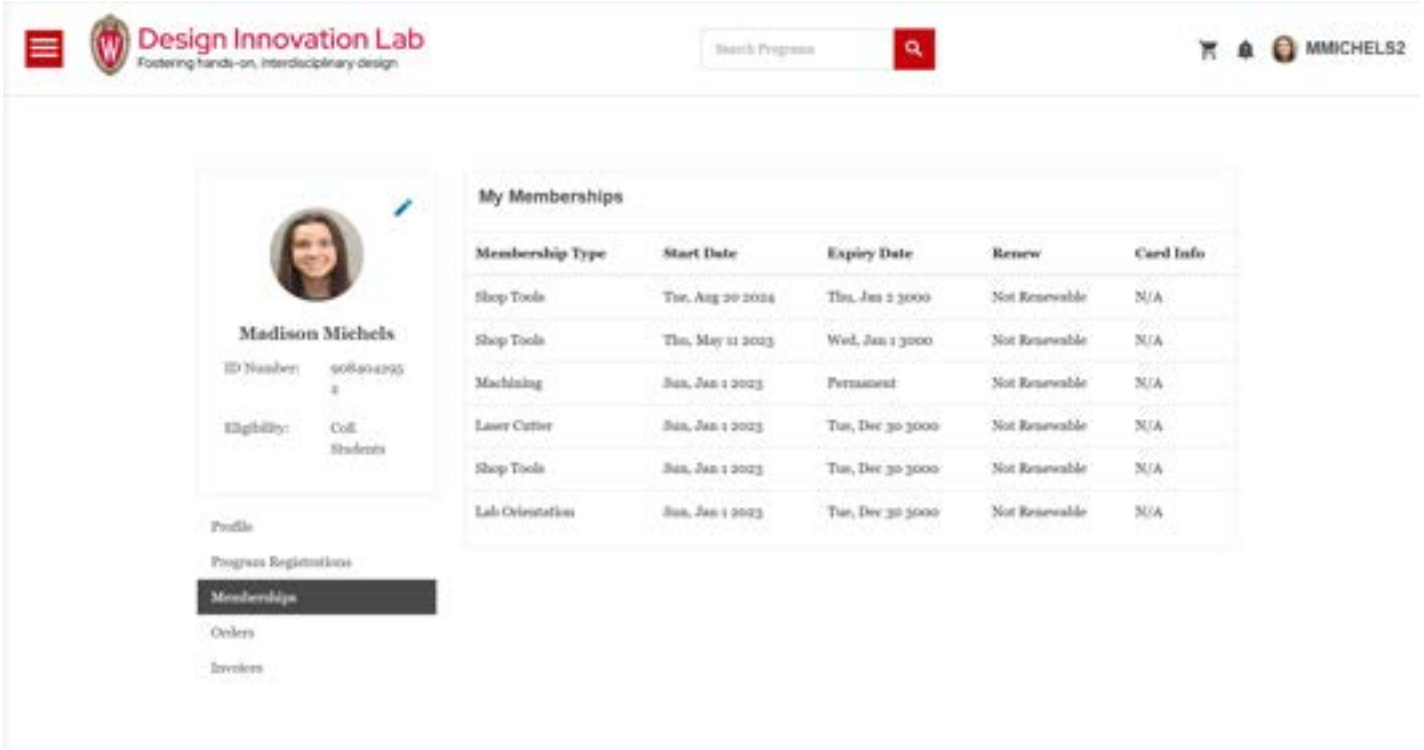
Title: Design Innovation Lab Training

Date: 3/15/2026

Content by: Maddie

Present: N/A

Content:



3/15/2026 - UW Safety Trainings

Madison Michels - Mar 15, 2026, 6:04 PM CDT

Title: Design Innovation Lab Training

Date: 3/15/2026

Content by: Maddie

Present: N/A

Content:

VCRGE Training Information Lookup Tool — University of Wisconsin-Madison

WISCONSIN
UNIVERSITY OF WISCONSIN-MADISON

This certifies that Madison Michels has completed training for the following course(s):

Course	Assignment	Completion	Expiration
Biosafety Required Training	Biosafety Required Training Quiz 2024	1/25/2024	1/25/2029
Chemical Safety: The OSHA Lab Standard	Final Quiz	1/24/2024	
UW Human Subjects Protections Course	Basic/Refresher Course - Human Subjects Research	1/26/2025	1/26/2028

Data Last Imported: 09/08/2025 12:45 PM

3/15/2026 - Research Animal Resources and Compliance (RARC)

Madison Michels - Mar 15, 2026, 6:04 PM CDT

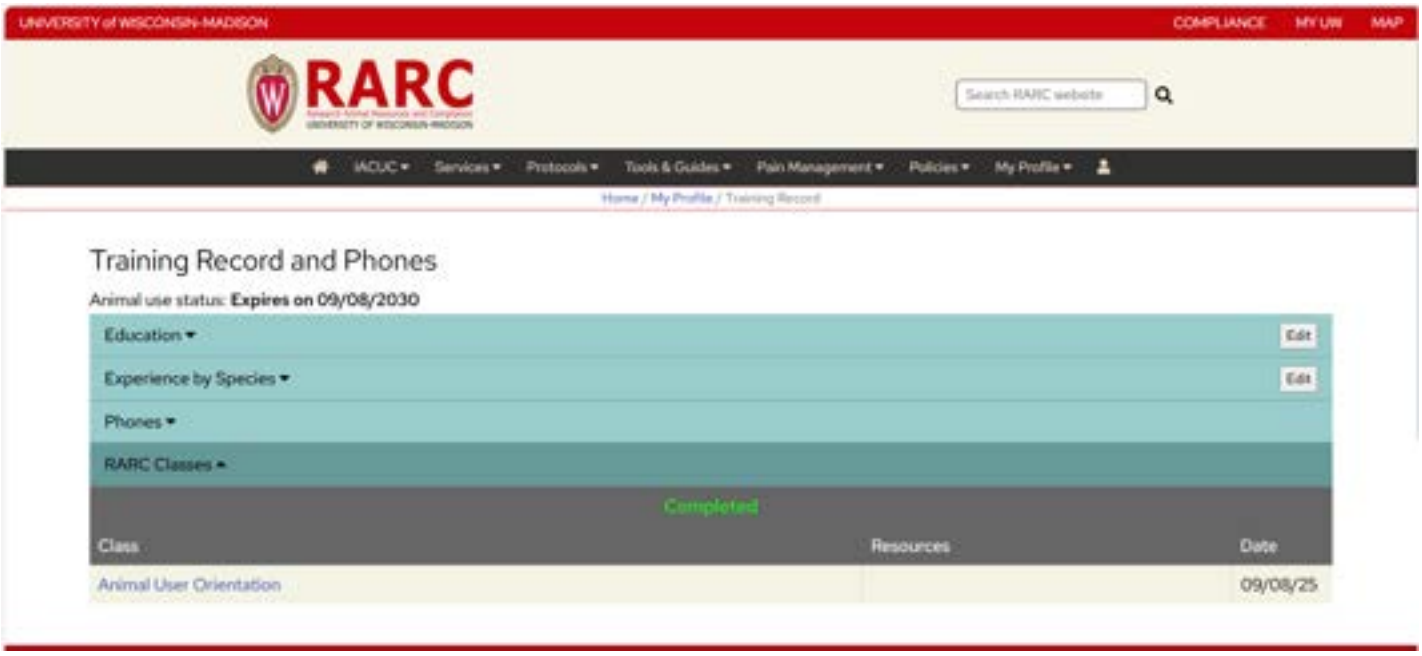
Title: Research Animal Resources and Compliance (RARC)

Date: 3/15/2026

Content by: Maddie

Present: N/A

Content:



3/15/2026 - Programming Foundations: Artificial Intelligence

Madison Michels - Mar 15, 2026, 6:43 PM CDT

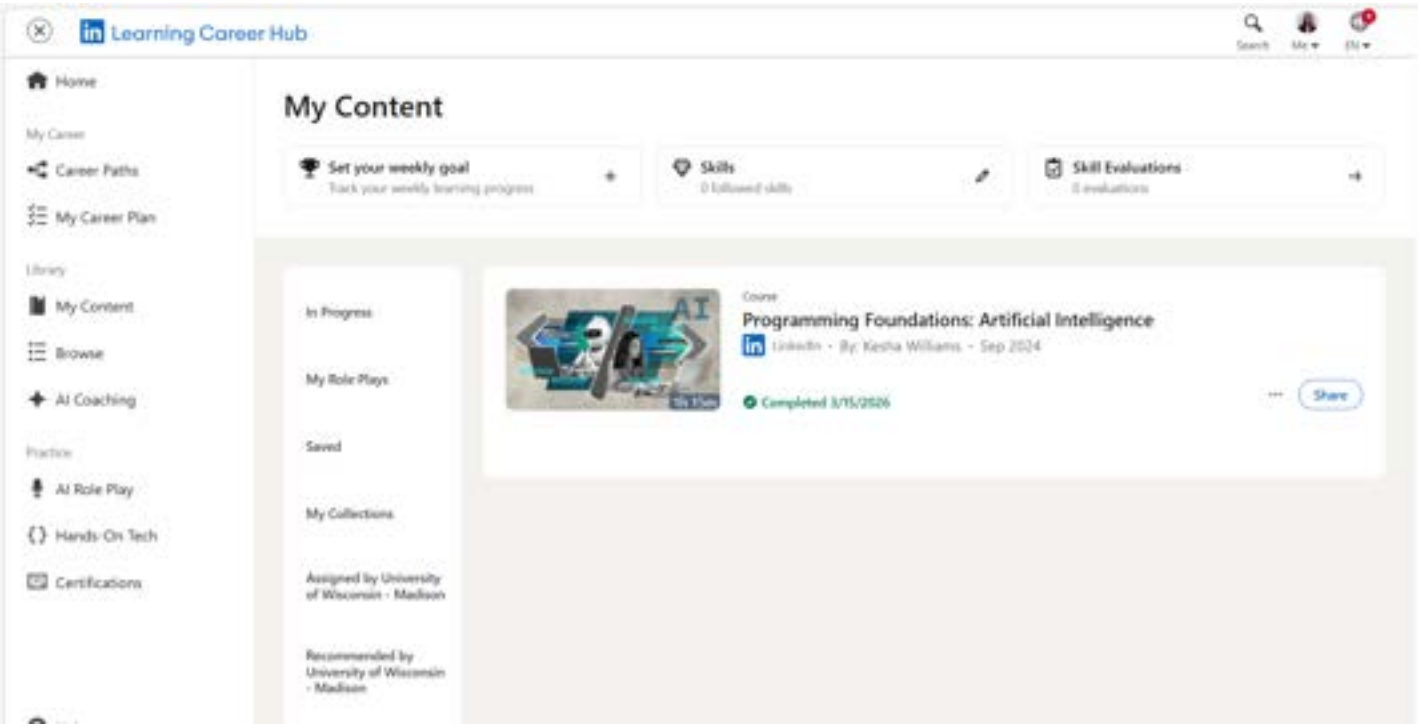
Title: Programming Foundations: Artificial Intelligence (LinkedIN)

Date: 3/15/2026

Content by: Maddie

Present: N/A

Content:



03/06/2026 - TONG Lecture

Madison Michels - Mar 06, 2026, 2:41 PM CST

Title: TONG Lecture - Justin Williams

Date: 03/06/2026

Content by: Maddie

Present: Maddie

Goals: The goal of this entry is to learn about Justin Williams' experience and accomplishments.

Content:

- Grew up in a rural town on a farm.
- Earned a track scholarship to the University of South Dakota and worked at Daktronics.
- A close friend died from a traumatic brain injury, which inspired him to pursue graduate school in biomedical engineering at Arizona State University.
- Worked in a rat brain laboratory in Michigan.
- Later worked on deep brain stimulation (DBS) technologies at the University of Wisconsin–Madison.

Neural Integration Technologies

- The company was acquired by Gore, which ultimately shelved the technology in order to develop its own competing products.

NeuroNexus

- Developed neural electrode arrays used for brain research.
- The company was acquired by Greatbatch but later bought back its independence.
- Today, NeuroNexus supplies a large portion of electrophysiology equipment used in neural research

NeuroOne

- Key idea: sometimes less is more; small improvements can be powerful
- Important to design with the end user in mind and consider whether clients are ready for advanced tech
- Spent a lot of time in the operating room learning from neurosurgeons and helping implement DBS tech
- Observed many epilepsy surgeries using multi-contact electrodes placed through a large horseshoe-shaped incision in the skull
- A surgeon challenged him to create a better epilepsy treatment → flexible electronics
- Reduced the amount of material in implanted electronics
- More flexibility → less foreign material in the brain and reduced immune response
- Developed electrodes that unfurl after insertion, allowing implantation through a much smaller incision
- Company eventually went public and captured ~12% of the epilepsy market

BrainSync

- Major lesson: market size and timing matter a lot
- Electrodes placed over the motor cortex
- Allowed patients to control an arm with brain signals without moving their arm
- Initially targeted ALS patients

- Enabled the first tweet sent purely by thinking using a non-invasive EEG
- Named one of the top 10 inventions of 2009
- ALS market is small (~5,000 new patients/year) → not a strong business case
- Technology was ultimately given away for free
- Later adapted for stroke rehabilitation
- Records brain signals when a patient thinks about moving their arm
- CyberGlove stimulates muscles to move the arm, helping retrain the brain

NeuraWorx

- Focus on progressive supranuclear palsy (PSP)
- Disease has similarities to Alzheimer's and Parkinson's
- Caused by buildup of tau protein that damages brain cells
- Work centers on the glymphatic system (brain's waste clearance system)
- Cerebrospinal fluid clears waste through tiny capillaries
- Most waste removal happens during sleep
- Aging and brain injuries make vessels stiffer and reduce waste clearance
- Initially used vagus nerve stimulation
- Too much/too long stimulation can cause the vagus nerve to shut down
- Mapped the pathways of waste removal in the brain
- Developed a mouthguard that stimulates the trigeminal nerve in the mouth
- Also uses a headband to stimulate nerves in the forehead
- Working toward a future micro-implant version of the technology

Takeaways

- Incremental innovation can still be very impactful
- Learning directly from surgeons and the clinical environment is critical
- Market timing and market size can determine whether a technology succeeds commercially

Conclusions/action items:

These experiences show how many breakthroughs in neurotechnology come from small, practical improvements rather than entirely new inventions. Spending time with surgeons and understanding real clinical needs was critical for developing technologies that actually work in practice. At the same time, the success of a medical technology often depends not just on the science, but also on market timing and the size of the patient population.

1/29/2026 - Model Evaluation with k-fold cross validation

SADIE ROWE - Jan 29, 2026, 2:44 PM CST

Title: Model Evaluation with k-fold cross validation

Date: 1/29/2026

Content by: Sadie Rowe

Present: N/A

Goals: Learn more about k-fold cross validation and how it can be used to evaluate and improve our existing model

Content:

Sources:

https://www.geeksforgeeks.org/machine-learning/k-fold-cross-validation-in-machine-learning/

What is k-fold cross validation? a method for checking how well a machine learning model will generalize to new, unseen data

- rather than training and testing the model just once, you train and test it multiple times on different splits of the data
- validation/evaluation technique used during model development

Steps:

1. split dataset into k equal parts (aka: folds)

2. run k training/testing rounds (in each round, 1 fold = test set and the other k-1 folds = training set)

- for each fold:
  - use k-1 folds for training the model
  - use the remaining fold as the test set to evaluate the model

3. each fold gets used once as the test set

4. performance is averaged across all k runs

- performance metrics (accuracy, precision, recall, etc.) are calculated for each fold and averaged

Choosing the value of K: affects the trade-off between bias and variance

- Small K (2-5): Faster computation with increased variance in performance estimates
- Large K (K=10, K = n where n is the dataset size): lower variance but higher computational cost

Why?

- Uses all data efficiently (good for small datasets)
  - maximizes the utilization of the dataset
- reduces dependence on one lucky/unlucky train-test split
- gives a more reliable estimate of real-world performance
  - more robust evaluation: averages results over multiple iterations, reducing bias and variance in performance metrics
- helps compare models and tune hyperparameters
- avoids overfitting ensures that the model doesn't perform well only on the training data but generalizes to unseen data

Limitations:

- Computational Cost: Re-training the model k times can be time-consuming
- Data leakage risk: care must be taken to ensure no data preprocessing leaks information from the test set into the training set

Implementation in Python using skikit-learn library:

- cross_val_score: returns an array of scores (one per fold)
  - Code:
    - from sklearn.model_selection import cross_val_score

      from sklearn.linear_model import LogisticRegression

      model = LogisticRegression()

      scores = cross_val_score(model, X, y, cv=5)

      print(scores)

      print("Average score:", scores.mean())
  - scikit-learn: splits the data, trains the model K times, evaluates it, and returns a score
- Explicit Kfold object: use if needed to control shuffling or random state
  - Code:
    - from sklearn.model_selection import KFold, cross_val_score

      kf = KFold(n_splits=5, shuffle=True, random_state=42)

      scores = cross_val_score(model, X, y, cv=kf)
  - have control whether data is shuffled and reproducibility (everything else is the same)
- StratufieldKfold: used for classification problems

    - Code:
        - from sklearn.model_selection import StratifiedKFold

          skf = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)

          scores = cross_val_score(model, X, y, cv=skf)
    - each fold has the same class proportions as the full dataset
    - best to use when classes are imbalanced (the case in our project)
- Hyperparameter tuning (cross-validation & model selection):
    - Code
        - from sklearn.model_selection import GridSearchCV

          param_grid = {'C': [0.01, 0.1, 1, 10]}

          grid = GridSearchCV(LogisticRegression(), param_grid, cv=5)

          grid.fit(X, y)

          print(grid.best_params_)

          print(grid.best_score_)

Conclusions/action items: Compile research with teammates to decide if k-fold cross validation is something we want to implement in our project. I think this is a logical next step to gain a better understanding of how our model works and if we need to continue making improvements.

2/2/2026 - K fold cross validation

SADIE ROWE - Feb 02, 2026, 12:30 PM CST

Title: K fold cross validation

Date: 2/2/2026

Content by: Sadie Rowe

Present: N/A

Goals: Continue learning about k fold cross validation in preparation to implement in our project

Sources:

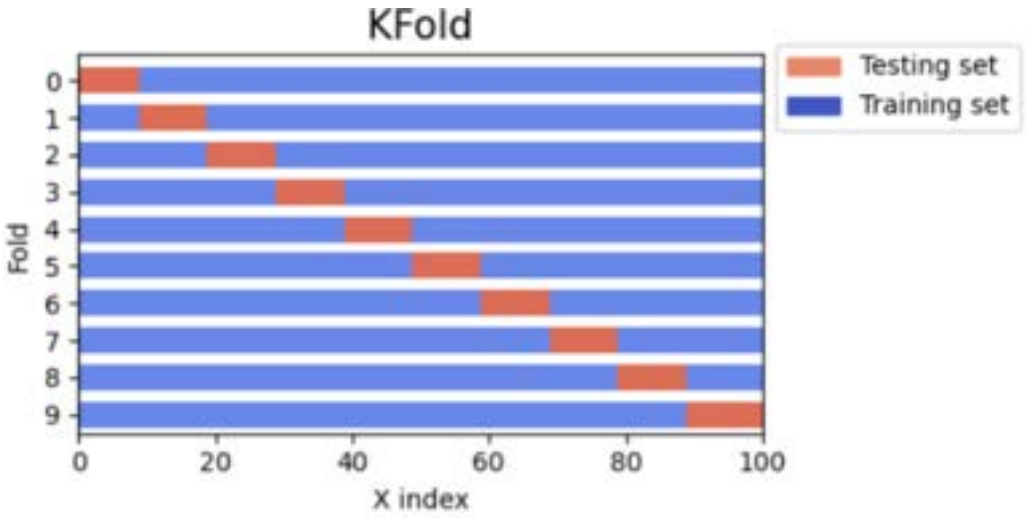
https://www.bing.com/videos/riverview/relatedvideo?q=k+fold+cross+validation&mid=EB41D143FB7FFECF495EEB41D143FB7FFECF495E&FORM=VIRE

Cross-Validation Using K-Fold With Scikit-Learn - GeeksforGeeks

Content:

Why to use k fold cross validation?

A model that is trained on a different subset set of images in a dataset can have different performance than a model trained on a slightly different grouping of images in that same set. You create bias by selecting a designated cohort of images to train a model. This can also be used for hyper-parameter selection



Step 1: divide data into K folds

Step 2: use each fold in the testing data (called as validation set) for each model (Models 1-5 in our casw)

Step 3: select the model with the lowest cross validation error (this is the model which performed best)

Number of experiments:

If choosing between m models, using k-folds: the number of experiments required = m*k

K = 10 is the default number of folds used in practice

Variations of K-fold cross validation

- Repeated K-Fold: can be used when one requires to run K-fold n times, producing different splits in each repetition
- Stratified K-fold: variation of K-Fold which returns stratified sample
- Group K-Fold: variation of K-fold which ensures that the same group is not represented in both testing and training sets
- Stratified Group K-Fold: cross validation scheme that combines both Stratified K fold and Group k fold

Conclusions/action items: Next steps include implementing k-fold cross validation to increase model performance by avoiding overfitting and underfitting.

2/13/2025 - Image augmentation techniques

SADIE ROWE - Feb 15, 2026, 12:03 PM CST

Title: Image augmentation techniques

Date: 2/13/2026

Content by: Sadie Rowe

Present: N/A

Goals: Understand typical image augmentation techniques and the percentage of images that each technique will be applied to

Link: A Comprehensive Survey of Image Augmentation Techniques for Deep Learning - ScienceDirect

Citation: M. Xu, S. Yoon, A. Fuentes, and D. S. Park, “A Comprehensive Survey of Image Augmentation Techniques for Deep Learning,” *Pattern Recognition*, vol. 137, p. 109347, May 2023, doi: 10.1016/j.patcog.2023.109347.

Search Term: 'Image augmentation techniques'

Content:

Paper	Image augmentation method
AlexNet [11]	Translate, Flip, Intensity Changing
ResNet [12]	Crop, Flip
DenseNet [13]	Flip, Crop, Translate
MobileNet [14]	Crop, Elastic distortion
NasNet [15]	Cutout, Crop, Flip
ResNeSt [16]	AutoAugment, Mixup, Crop
DeiT [17]	AutoAugmentat, RandAugment, Random Erasing, Mixup, CutMix
Swin Transformer [18]	RandAugment, Mixup, CutMix, Random Erasieg
Faster R-CNN [19]	Flip
YOLO [20]	Scale, Translate, Color space
SSD [21]	Crop, Resize, Flip, Color Space, Distortion
YOLOv4 [22]	Mosaic, Distortion, Scale, Color space, Crop, Flip, Rotate, Random erase, Cutout, Hide-and-Seek, GridMask, Mixup, CutMix, StyleGAN

Motivation to perform image augmentation:

- Class imbalance: When trained with an imbalanced dataset, a model assigns a higher probability to the normal case
- Domain shift: training & testing exhibit different distributions
- Image variations: variations in illumination, deformation, occlusion, background, viewpoint, and multiscale
- Data remembering: larger models with many learnable parameters tend to remember specific datapoints, which may result in overfitting

Types of augmentation:

- Geometric transformation: translation, rotation, flip
- Color image processing: yield superior performance but are rarely used because color variations between training and testing datasets are small

Simple transformations:

- Resize: all images same size
- Grayscale
- Normalize
- Random Rotation: 50, 100, 150
- Center Crop
- Random Crop
- Gaussian Blur

More advanced techniques:

- Gaussian Noise
- Random Blocks
- Central Region

Conclusions/action items: Apply image augmentation techniques to dataset

2/15/2026 - Augmentation Details

SADIE ROWE - Feb 20, 2026, 12:50 PM CST

Title: Augmentation Details

Date: 2/15/2026

Content by: Sadie Rowe

Present: N/A

Goals: Document the augmentation process through Roboflow

Content:

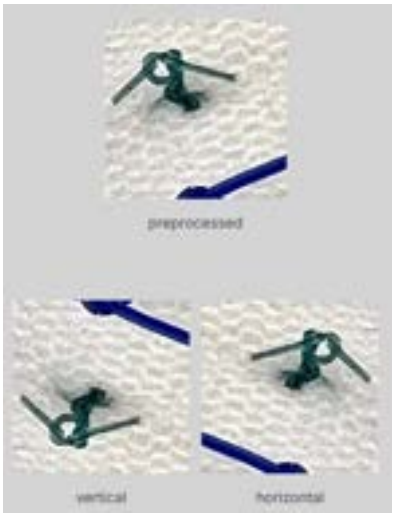
Platform used: Roboflow

Source Images: 276 total

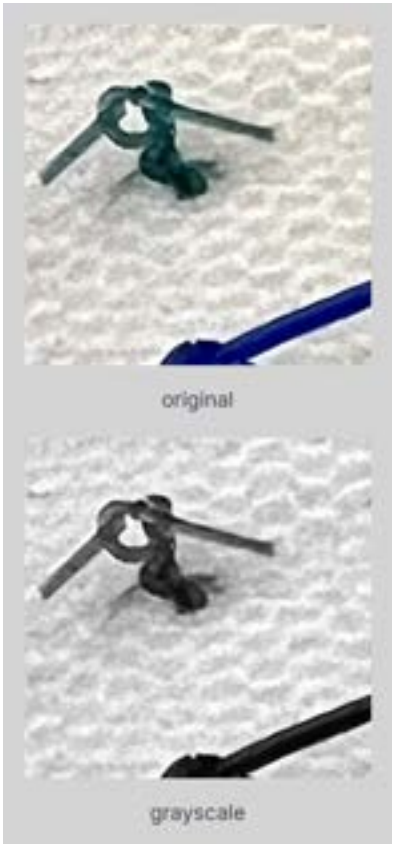
- 138 tight images
- 138 loose images
- All images have been cropped to 500x500 pixels

Preprocessing: None (already completed with each image individually)

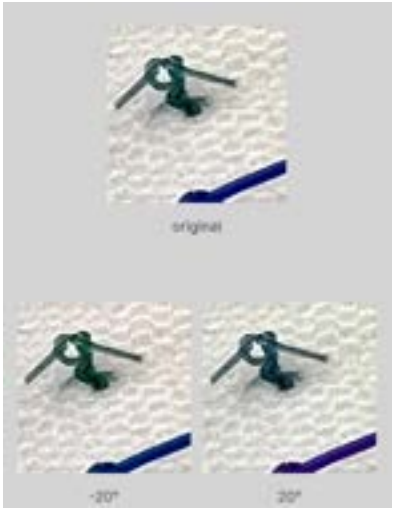
Augmentation:

- Flip: horizontal, vertical
  - 

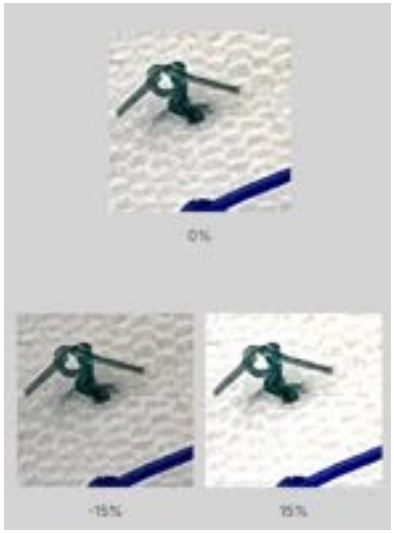
- Grayscale: 20% of outputted images will be changed to grayscale
  - reduces over reliance on color cues since different suture colors were photographed



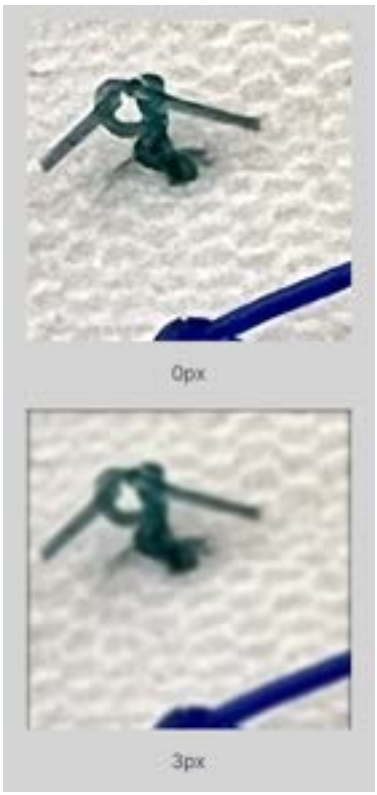
- Hue: +/- 20 degrees
  - randomly adjust the colors in the image by rotating the hues around the color wheel
  - used due to the presence of different suture colors and slightly different backgrounds (skin pads)
  - equivalent to ColorJitter hue = 0.055 in PyTorch (using a normalized 0-1 scale)



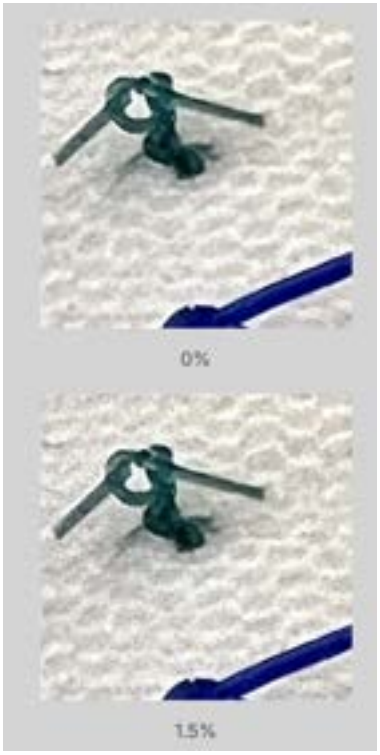
- Brightness: 15% brighten & darken



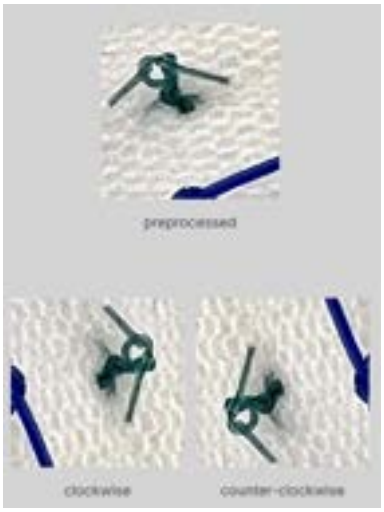
- Blur: 3px
  - adds random gaussian blue to help the model be more resilient to camera focus



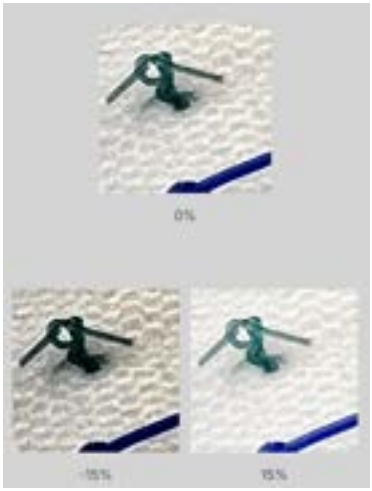
- Noise: 1.5%
  - adds noise to help the model be more resilient to camera artifacts



- 90 degree rotate: clockwise and counterclockwise
  - helps the model be more insensitive to camera orientation as it will depend which orientation users tie a knot with the fixed camera



- Exposure: 15%
  - adds variability to image brightness to help the model be more resilient to lighting & camera setting changes

- 

Version size: 828 images (3x dataset)

Conclusions/action items: Save augmented images for use in model training.

2/10/2026 - Clarified Terminology

SADIE ROWE - Feb 10, 2026, 3:24 PM CST

Title: Clarified Terminology

Date: 2/10/2026

Content by: Sadie Rowe

Present: N/A

Goals: Clarify performance metric terms and describe how they can be understood a medical context (ex: translate into BME terms)

Content:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

	Actually Positive	Actually Negative
Predicted Positive	True Positive (TP)	False Positive (FP)
Predicted Negative	False Negative (FN)	True Negative (TN)

	Actually Loose	Actually Tight
Predicted Loose	True Positive (TP)	False Positive (FP)
Predicted Tight	False Negative (FN)	True Negative (TN)

Accuracy: proportion of *all predictions* the model got correct

Precision: of all the **predicted** positive cases, how many were **actually** positive

- AKA: positive predictive value (PPV)
- “When the model says positive, can I trust it? Are we falsely predicting something positive?”

Recall: of all the **actual** positive cases, how many were **predicted** correctly

- AKA: Sensitivity
- “How many of the positive cases did we actually catch?”

F1 Score: A single metric that balances precision and recall.

- If either precision OR recall is bad, f1 will drop
- F1 is only high when the model both FINDS the positive and is usually CORRECT when it does

Specificity: of all the actual negative cases, how many were predicted correctly

- Specificity = TN/(FP+TN)

Conclusions/action items: N/A

2/1/2026 - Hardware Overview

SADIE ROWE - Feb 01, 2026, 9:26 PM CST

Title: Hardware Overview

Date: 2/1/2026

Content by: Sadie Rowe

Present: N/A

Goals: Better understand how the hardware could be setup in order to create a training module

Source: https://www.raspberrypi.com/documentation/accessories/camera.html#install-a-raspberry-pi-camera

Advice from a Comp Sci/Electrical Engineering Student (Nico Grafe)

Content:

Hardware:

- Raspberry pi AI camera: https://www.raspberrypi.com/documentation/accessories/
- Raspberry pi 5: https://www.raspberrypi.com/products/raspberry-pi-5/
  - with use of Raspberry Pi OS
  - 16GB RAM
  - $145
- Raspberry pi active cooler: https://www.raspberrypi.com/products/active-cooler/
  - makes sure the microcontroller runs smoothly while working its hardest
- Raspberry pi 27W USB-C power supply: https://www.raspberrypi.com/products/27w-power-supply/
  - robust Pi running at peak performance
- Micro HDMI to HDMI cord to connect Raspberry Pi to a monitor
- Micro SD card for storage (32G)

1. Initial hardware setup

- Remove Raspberry Pi from packaging
- Insert a microSD card into the Raspberry Pi
- Plug into power source
- Connect the Raspberry Pi to a monitor using a micro-HDMI to HDMI cable
- Plug in a USB keyboard and mouse
- Connect the power supply to the raspberry pi and plug it into the wall

2. Booting and operating system

- power on the raspberry pi
- boot using raspberry Pi OS (recommended for easiest peripheral and camera support)
  - Linux OS could also be used for requires more configuration
- Complete initial setup (language, Wi-Fi, updates)

3. AI Camera connection

- Locate the camera connector on the Raspberry Pi board
    - the two camera and display connectors are by the edge closest to you between the micro HDMI connector and the Ethernet port. They’re labelled CAM/DISP0 and CAM/DISP1. You can use either of these connectors for your camera.
- open the flap connector
- insert the end of your camera cable with the metallic contacts facing away from the flap
    - ensure the camera cable is inserted firmly into the connector and is seated straight to correctly align all contacts
- close the connector flap by tilting it back towards the cable and push it down into the connector until it clicks in place
- connect the cable to the camera

Ideal flow of events:

take picture of knot --> run image through the model --> display output (red/green light) --> do not store image

Conclusions/action items: To do:

- Research if a Raspberry Pi can be hooked up to a laptop
- Research Raspberry pi AI hat (a module that is good for running machine learning models)
    - explore whether or not this would be a better us than the raspberry pi AI camera

2/9/2026 - Additional Camera Options

SADIE ROWE - Feb 09, 2026, 7:32 PM CST

Title: Additional Raspberry Pi Camera Options

Date: 2/9/2026

Content by: Sadie Rowe

Present: N/A

Goals: Document additional camera that could be used with the Raspberry Pi 5. Given that we model will process relatively low-resolution images, it's not necessary to purchase such an advanced camera (like the Raspberry Pi HQ camera), so I've explored other potential options. These will need to be reviewed with the team and then an order will be placed this week via the client

Content:

NOTE: both cameras are compatible with a raspberry pi 5

Raspberry Pi Camera Module 2: Amazon Link

- Can be used for still photographs or videos
- 8 megapixels
- Sony IMX219 8-megapixel
- Attaches via a 15cm ribbon cable to the CSI port on the Raspberry Pi.
- $29.95
- Second Generation Raspberry Pi Camera Module with Fixed Focus Lens
- Sony Exmor IMX219 Sensor Capable of 4K30 1080P60 720P180 8MP Still
- 3280 (H) x 2464 (V) Active Pixel Count
- Maximum of 1080P30 and 8MP Stills in Raspberry Pi Board
- 2A Power Supply Highly Recommended

FREENOVE 5MP Camera for Raspberry Pi

- Not raspberry pi brand, so might not work as well
- $11.95
- Compatible Models: Raspberry Pi 5 / 4B / 3B+ / 3B / 3A+ / 2B / 1B+ / 1A+ / Zero 2 W / Zero W / Zero
- Camera Parameters: 5 megapixels (2592 x 1944 pixels), 62° field of view, 1080p 30fps, 720p 60fps video modes
- Adjustable Holder: Allows you to easily adjust the camera to the most suitable angle

Conclusions/action items: Discuss camera options with team and add final camera to the expense sheet.

1/26/26 - Raspberry Pi Camera Research

SADIE ROWE - Jan 26, 2026, 7:56 PM CST

Title: Raspberry Pi Camera Research

Date: 1/26/26

Content by: Sadie Rowe

Present: N/A

Goals: Better understand if a raspberry pi camera could be used for implementation of our model.

Link: https://www.raspberrypi.com/documentation/accessories/ai-camera.html#ai-camera

Content:

Raspberry Pi camera modules:

- Camera Module 2: A 8-megapixel camera available in standard (visible light) and NoIR (visible light + infrared) versions with a standard FoV
- Camera Module 3: A 12-megapixel camera available in standard & NoIR versions (both with standard and wide field of view (FoV) for a total of 4 variants)
- High quality camera: 12-megapixel camera that comes with CS- or M12-mount variants for use with external lenses
- AI Camera: 12-megapixal camera that provides low-latency and high-performance AI capabilities to any camera application
    - tight integration with raspberry pi's camera software stack enables users to deploy own NN models easily
- Global shutter camera: 1.5-megapixel camera using a global shutter mechanisms. Captures light from every pixel in the scene at once and is ideal for fast-motion photography

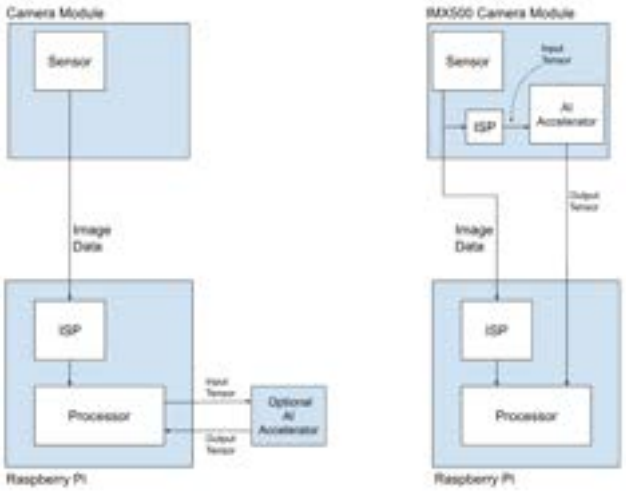
The following table compares the features and capabilities of the Raspberry Pi camera hardware.

	Camera Module 1	**Camera Module 2**	**Camera Module 3**	**Camera Module 3 Wide**	**HQ Camera**	**AI Camera**	**GS Camera**
Size	Around 25 × 24 × 9 mm	Around 25 × 24 × 9 mm	Around 25 × 24 × 11.5 mm	Around 25 × 24 × 12.4 mm	38 × 38 × 18.4 mm (excluding lens)	25 × 24 × 11.9 mm	38 × 38 × 19.8 mm (29.5 mm with adaptor and dust cap)
Weight	3 g	3 g	4 g	4 g	30.4 g	6 g	34 g (41 g with adaptor and dust cap)
Still resolution	5 megapixels	8 megapixels	11.9 megapixels	11.9 megapixels	12.3 megapixels	12.3 megapixels	1.58 megapixels

	Camera Module 1	Camera Module 2	Camera Module 3	Camera Module 3 Wide	HQ Camera	AI Camera	GS Camera
Video modes	1080p30, 720p60 and 640 × 480p60/90	1080p47, 1640 × 1232p41 and 640 × 480p206	2304 × 1296p56, 2304 × 1296p30 HDR, 1536 × 864p120	2304 × 1296p56, 2304 × 1296p30 HDR, 1536 × 864p120	2028 × 1080p50, 2028 × 1520p40 and 1332 × 990p120	2028 × 1520p30, 4056 × 3040p10	1456 × 1088p60
Sensor	OmniVision OV5647	Sony IMX219	Sony IMX708	Sony IMX708	Sony IMX477	Sony IMX500	Sony IMX296
Sensor resolution	2592 × 1944 pixels	3280 × 2464 pixels	4608 × 2592 pixels	4608 × 2592 pixels	4056 × 3040 pixels	4056 × 3040 pixels	1456 × 1088 pixels
Sensor image area	3.76 × 2.74 mm	3.68 × 2.76 mm (4.6 mm diagonal)	6.45 × 3.63 mm (7.4 mm diagonal)	6.45 × 3.63 mm (7.4 mm diagonal)	6.287 × 4.712 mm (7.9 mm diagonal)	6.287 × 4.712 mm (7.9 mm diagonal)	6.3 mm diagonal
Pixel size	1.4 µm × 1.4 µm	1.12 µm × 1.12 µm	1.4 µm × 1.4 µm	1.4 µm × 1.4 µm	1.55 µm × 1.55 µm	1.55 µm × 1.55 µm	3.45 µm × 3.45 µm
Optical size	1/4"	1/4"	1/2.43"	1/2.43"	1/2.3"	1/2.3"	1/2.9"
Focus	Fixed	Adjustable	Motorised	Motorised	Adjustable	Adjustable	Adjustable
Depth of field	Approx 1 m to ∞	Approx 10 cm to ∞	Approx 10 cm to ∞	Approx 5 cm to ∞	N/A	Approx 20 cm to ∞	N/A
Focal length	3.60 mm ± 0.01	3.04 mm	4.74 mm	2.75 mm	Depends on lens	4.74 mm	Depends on lens
Horizontal Field of View (FoV)	53.50 ± 0.13 degrees	62.2 degrees	66 degrees	102 degrees	Depends on lens	66 ± 3 degrees	Depends on lens
Vertical Field of View (FoV)	41.41 ± 0.11 degrees	48.8 degrees	41 degrees	67 degrees	Depends on lens	52.3 ± 3 degrees	Depends on lens
Focal ratio (F-Stop)	F2.9	F2.0	F1.8	F2.2	Depends on lens	F1.79	Depends on lens
Maximum exposure time (seconds)	3.28	11.76	112	112	670.74	112	15.5
Lens Mount	N/A	N/A	N/A	N/A	C/CS- or M12-mount	N/A	C/CS
NoIR version available?	Yes	Yes	Yes	Yes	No	No	No

Raspberry Pi AI Camera:

- Uses Sony IMX500 imaging sensor
- low-latency, high-performance AI capabilities
- able to deploy own neural network model with minimal effort



  - Left: architecture of a traditional AI camera system (camera delivers images to raspberry pi --> raspberry pi processes the images then performs an AI interference)
    - may use external AI accelerator OR rely exclusively on the CPU
  - Right: architecture of a system using IMX500. Camera module contains a small image signal processor (ISP) which turns the raw camera image data into an input tensor --> camera module sends tensor directly to the AI accelerator within the camera --> output tensors produced (containing the inferencing results) --> sent to the raspberry pi
    - no need for an external accelerator
    - no need for raspberry pi to run neural network software on the CPU
- To deploy a new neural network model to the raspberry pi AI camera:
  - 1. provide a floating-point neural network model (PyTorch or TensorFlow)
  - 2. run the model through Edge-MDT (edge AI model development toolkit)
  - 3. package the model into a firmware that can be loaded at runtime onto the camera

Concepts:

Input Tensor
: The part of the sensor image passed to the AI engine for inferencing. Produced by a small on-board ISP which also crops and scales the camera image to the dimensions expected by the neural network that has been loaded. The input tensor is not normally made available to applications, though it is possible to access it for debugging purposes.

Region of Interest (ROI)
: Specifies exactly which part of the sensor image is cropped out before being rescaled to the size demanded by the neural network. Can be queried and set by an application. The units used are always pixels in the full resolution sensor output. The default ROI setting uses the full image received from the sensor, cropping no data.

Output Tensors

The results of inferencing performed by the neural network. The precise number and shape of the outputs depend on the neural network. Application code must understand how to handle the tensors.

Conclusions/action items: Based on this initial research, it seems that the Rapsberry Pi AI camera would be the best fit for use in our design. I plan to discuss this information with the team and determine whether further research is needed/if this is something we want to pursue within the semester.

2/2/2026 - Laptop connection to Raspberry Pi

SADIE ROWE - Feb 02, 2026, 10:37 AM CST

Title: Laptop connection to Raspberry Pi

Date: 2/2/2026

Content by: Sadie Rowe

Present: N/A

Goals: Learn about how/if a Raspberry Pi can be connected to control the training module

Sources:

How Can I Connect My Raspberry Pi to a Laptop?

How to Connect Raspberry Pi to Laptop: Complete Step-by-Step Guide (2025) - Emerging Technologies

Content:

Hardware needed:

- A Raspberry Pi
- Laptop w/ internet
- MicoSD card
- Power supply for Pi
- ethernet cable (optional)

Software needed:

- Raspberry Pi OS installed on the SD card
- SSH client (PuTTY on Windows)
- A VNC viewer (RealVNC Viewer)
- SD card formatting tool if needed

Basic Knowledge:

- Use of command line tools
- understanding of IP addresses

TIP: get Raspberry Pi imager from official site to load the OS onto SD card

Connecting Raspberry Pi to a Laptop via SSH (Secure Shell)

- SSH (Secure Shell): one of the most common methods to establish a secure and efficient connection
- enables remove command-line access (allowing control of Raspberry Pi from the laptop w/out a separate monitor, keyboard, or mouse)
- How to connect via SSH:

    - enable SSH on raspberry pi: done by placing an empty file named 'ssh' in the boot partition of the Raspberry Pi's SD card before the first boot
    - power on raspberry pi & find IP address
    - Open a terminal on laptop (Linux or macOS) or use an SSH like PuTTY (Windows)
    - connect using the following command:
        - “`bash
          ssh pi@
          “`

Using VNC for Graphical Interface Access

- Use if full graphical access to Raspberry Pi desktop environment is needed
- VNC = virtual network computing
- VNC transmits the graphical screen of Raspberry Pi to laptop (allowing interaction as if directly connected)
- Setup Steps for VNC:
    - Enable VNC on the Raspberry Pi via `raspi-config` or the Raspberry Pi Configuration tool.
    - Install a VNC viewer on your laptop. RealVNC Viewer is the official client and supports multiple platforms.
    - Ensure both devices are on the same network.
    - Launch the VNC viewer on your laptop and enter the Raspberry Pi’s IP address followed by the display number (usually `:1`).
    - Authenticate with your Raspberry Pi credentials.

Direct USB connection between Raspberry Pi and Laptop

- Connect Raspberry Pi directly to laptop with a USB cable
- bypasses need for a network
- particularly useful method for Raspberry Pi Zero models (support USB gadget mode
- requires a more advanced configuration but is beneficial when no other network is available
- Setup steps:
    - Use a USB data cable to connect the Raspberry Pi Zero to your laptop.
    - Configure the Raspberry Pi to act as a USB Ethernet device by editing the `config.txt` and `cmdline.txt` files on the boot partition.
    - On your laptop, the Raspberry Pi will appear as a network device, allowing you to SSH into it using a designated IP address.

Summary:

Connection Method	Use Case	Setup Complexity	Performance	Requires Network
SSH	Remote command-line access	Low	High (low bandwidth)	Yes
VNC	Graphical desktop access	Medium	Moderate (higher bandwidth)	Yes
Direct USB Connection	No network available; USB gadget mode	High	Moderate	No

Connecting using USB Ethernet Gadget Mode:

- **Preparation:** Enable USB gadget mode by modifying the boot configuration files on the Pi's SD card.
- **Steps:**
  - Edit `config.txt` to add `dtoverlay=dwc2`.
  - Edit `cmdline.txt` and add `modules-load=dwc2,g_ether` immediately after `rootwait`.
  - Create a new empty file named `ssh` in the boot partition to enable SSH.
  - Connect the Pi to the laptop via a USB data cable.
  - The laptop will recognize a new network device; assign static IP addresses as with Ethernet connection or use DHCP.

Conclusions/action items: Based on this research, we need to decide on a method to build our training module, compile materials, and place and order to begin prototyping.

2/18/2026 - Raspberry Pi Fabrication Protocol

SADIE ROWE - Mar 19, 2026, 6:02 PM CDT

Title: Raspberry Pi Fabrication Protocol

Date: 3/19/2026

Content by: Sadie Rowe

Present: N/A

Goals: Document fabrication of the RasTech Case and Active Cooler Installation Method

Content:

1. Install a small black power button into the RasTech case, insert it into the circular hole on the bottom case from the inside of the shell. The small end of the button is outside the case, and the large end is inside the case.
2. Put the board into the bottom case and secure it with four screws. Test again whether the black power button can press the power switch of the board normally.
3. Attach thermal pads to top of board for heat transfer
4. Install the active cooler on the board by connecting to the same four-pin JST connector as the case fan
5. Secure the board via two new mounting holes, and spring-loaded push pins for mounting onto board
6. Fix the middle frame and upper cover, and then insert the white four wire port of the active cooler PWM fan into the PWM fan port in the upper right corner of the board, as shown in the figure.
7. Fix the bottom cover, middle frame, and top cover. Insert the SD card of burned Raspberry Pi system into the SD card slot on the board, connect the 5V/3.1A USB-C power supply, and use the USB port to connect the keyboard and mouse. Use a Micro HDMI cable to connect the board and display screen, then you can starting use your Raspberry Pi 5 board.

Conclusions/action items: N/A

2/19/2026 - How to install TensorFlow model on Raspberry Pi 5

SADIE ROWE - Feb 19, 2026, 2:57 PM CST

Title: How to install TensorFlow model on a Raspberry Pi 5 Board

Date: 2/19/2026

Content by: Sadie Rowe

Present: N/A

Goals: Learn how to implement our TensorFlow model onto the Pi board in order to test whether this will be a viable method of deploying the model onto a set training module

Sources:

TensorFlow and AI on Raspberry Pi: A Beginner's Guide – RaspberryTips

Real-World Example of Deploying A Machine Learning Model on a Raspberry Pi | Codez Up

Content:

Before upload: convert to TensorFlow Lite:

```
import tensorflow as tf

converter = tf.lite.TFLiteConverter.from_saved_model("my_model")
converter.optimizations = [tf.lite.Optimize.DEFAULT]  # Recommended
tflite_model = converter.convert()

with open("model.tflite", "wb") as f:
 f.write(tflite_model)
```

Tensorflow can be installed on a Raspberry Pi with the pip command and then used with Python IDE

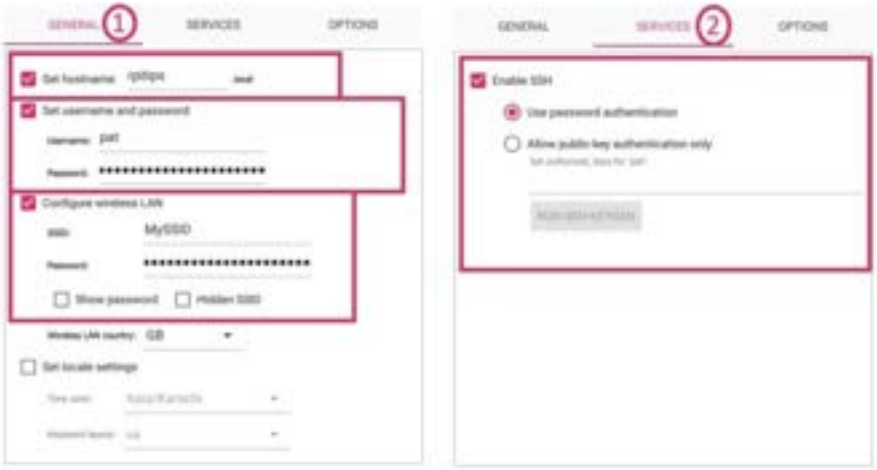
Dependencies need to be configured/installed first:

1. Install Raspberry Pi OS (Raspberry Pi OS) on Pi board using Raspberry Pi Imager

- What is Raspberry Pi OS: official operating system - most kits include it by default but might need to be installed.
- Steps to install OS:
  - Download raspberry pi images from the website: Raspberry Pi software – Raspberry Pi
  - install on computer: keep default values if given a choice
  - open imager --> choose Raspberry Pi OS version to install
    - version: Raspberry Pi OS Trixie (64-bit)
    - no customization
  - Insert SD card and write it
    - choose the SD card to put it on

        - click "WRITE" button to begin flashing process
    - Insert the SD card into Raspberry Pi
    - Start Raspberry Pi with SD card and complete configuration (screen & keyboard plugged in)

2. Ensure internet connectivity is established and SSH enabled during the installation using the configuration settings below



3. Make sure device is up to date with the following command: `sudo apt update && sudo apt full-upgrade`

4. Make sure python version is between 3.9 and 3.12 using the following command: `python --version`

5. Set up a virtual environment (venv) using the command: `python3 -m venv tf`

6. Activate the virtual environment using the command: `source tf/bin/activate`

7. Update pip version to 19.0 or higher with the command: `pip install --upgrade pip`

8. Raspberry Pi is now ready to install TensorFlow, but to have better visualization aids, we should downloard the following dependencies:

9. Pydot - run the following command inside virtual environment of tensorflow setup: `pip install pydot`

10. GraphViz - run the following command inside standard shell: `sudo apt install graphviz`

11. SSH into Raspberry Pi with: `ssh <username>@<hostname>`

12. Activate the virtual environment: `source tf/bin/activate`

13. Install TensorFlow: `pip install tensorflow`

- Once finished, will be returned to SSH prompt

14. To verify that tensorflow has been successfully installed: `pip show tensorflow`

Conclusions/action items: Implement steps on Raspberry Pi next week

2/25/2026 - Raspberry Pi OS Setup

SADIE ROWE - Feb 25, 2026, 9:47 PM CST

Title: Raspberry Pi Setup Protocol

Date: 2/25/2026

Content by: Sadie Rowe

Present: Entire Team

Goals: Document the initial setup of Raspberry Pi and identify areas to prepare for next meeting

Source Used to guide setup: Getting started - Raspberry Pi Documentation

Content:

NOTES
** OS came with python 3.13 already installed (use pip3 to initialize downloads)

Setup Steps

1. Install Raspberry Pi 64 bit OS from Raspberry Pi Imager
2. Add customizations
    a. Hostname: knotorious-5
    b. Localization: city, timezone, keyboard layout
    c. User:
        i. Username: knotorious-five
        ii. Password: Badger2026
    d. WiFi: skipped
    e. Remote access: did not establish SSH (
3. Write Raspberry Pi 64 bit OS onto micro SD card
4. Eject microSD card from computer
5. Insert microSD card into Raspberry Pi
6. Add peripherals to Raspberry Pi
    a. Keyboard
    b. Mouse
    c. Ethernet cable
    d. Monitor
7. Once on the monitor, open raspberry pi terminal
8. Install python 3.11 alongside python 3.13
    a. troubleshooting this code right now
    b. Start of CODE: Sudo apt install python3.11 python3.11-venv
9. Open virtual environment: lets you create an isolated workspace for a specific project. This isolation gives you control over dependencies and avoids conflicts between projects.
    a. CODE: python3 -m venv knotvenv #knotvenv is the virtual environment name
    source knotvenv/bin/activate
    pip3 install pandas

Next steps:

Install of tensorflow Lite: pip3 install tflite-runtime

If this does not work, you can install full TensorFlow version: pip3 install tensorflow

Transfer the trained model: scp model.tflite pi@<pi-ip-address>:/home/pi/

Write inference script to run the model:

- Example Script: import tflite_runtime.interpreter as tfliteimport numpy as np

  ```
  # Load model
  interpreter = tflite.Interpreter(model_path="model.tflite")
  interpreter.allocate_tensors()

  # Get input/output details
  input_details = interpreter.get_input_details()
  output_details = interpreter.get_output_details()

  # Prepare dummy input (adjust shape!)
  input_shape = input_details[0]['shape']
  input_data = np.random.random_sample(input_shape).astype(np.float32)

  # Run inference
  interpreter.set_tensor(input_details[0]['index'], input_data)
  interpreter.invoke()

  # Get output
  output = interpreter.get_tensor(output_details[0]['index'])
  print(output)
  ```

Conclusions/action items: Experiment with the following code to install model - troubleshoot as necessary

2/27/2026 - Raspberry Pi Connect & Virtual Environment Setup

SADIE ROWE - Mar 19, 2026, 6:06 PM CDT

Title: Raspberry Pi Connect & Virtual Environment Setup

Date: 3/19/2026

Content by: Sadie Rowe & Lucy Hockerman

Present: N/A

Goals: Document setup of Raspberry Pi Connect & Virtual Environment with the help of Makerspace staff (Norbu & Dani)

Content:

Connecting to Raspberry Pi Connect:

1. Connect device to wifi
    1. The Pi and your computer need to be on the same Wifi
2. Turn on Raspberry Pi Connect and login
3. Create a name for the device
    1. KnotPi
4. Open Raspberry Pi connect on personal laptop

Downgrading Python - version 3.9:

1. Open virtual environment

CODE USED:

python3 -m venv knotvenv

source knotvenv/bin/activate

python --version #checking which version is currently downloaded

pyenv install 3.9 #Installing pyenv: copied code from duckduckgo → installed version 3.9

→ next step: how to make 3.9 the working version of python

Install python version 3.9.13:

→ pyenv install 3.9.13

Do this in the (knotnewyay) environment

How to Enter and Exit virtual environment

Exit: deactivate

Enter: source knotvenv/bin/activate

To check what python version: python --version

How to shut off the pi:

- Raspberry Pi icon
- Shut-down

Virtual environment name: knotnewyay

Code used to make virtual environment with Python 3.9

Python3.9 -m venv knotnewyay

Source Kootenay/bin/activate

Code to install applications:

pip install --update pip

Reboot?

pip install pytorch

Conclusions/action items: Next steps include uploading the model.

3/4/2026 - Pi HQ Camera Setup Methods

SADIE ROWE - Mar 19, 2026, 6:11 PM CDT

Title: Pi HQ Camera Setup Methods

Date: 3/4/2026

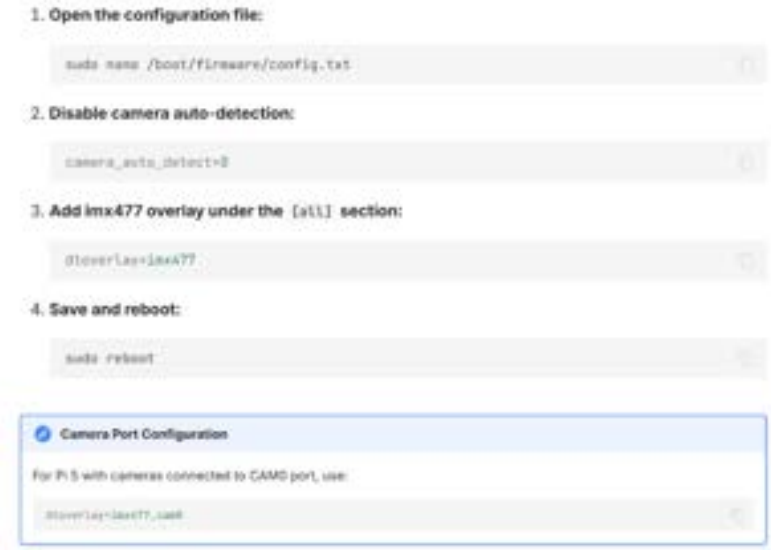
Content by: Sadie Rowe, Kate Hiller, Maddie Michels, Presley Hansen

Present: N/A

Goals: Document setup of HQ camera and code used to capture images

Content:

Setup:



Source: [12MP IMX477 - Arducam Wiki]

CODE:

sudo nano /boot/firmware/config.txt # open configuration file

camera_auto_detect=0 #disable camera autodetection

dtoverlay=imx477,cam0   # Add imx477 overlay under the [all] section

sudo reboot

Camera usage:

rpicam-still --list-cameras # list available cameras

rpicam-still -t 0 # live preview (use Ctrl+C to exit)

rpicam-still -t 5000 -o test.jpg # Capture image with 5-second preview:

Conclusion: Next steps include using the camera to capture new dataset images

3/19/2026 - Dataset Rebuild

SADIE ROWE - Mar 19, 2026, 6:15 PM CDT

Title: Dataset Rebuild

Date: 3/19/2026

Content by: Sadie Rowe, Lucy Heckerman, Presley Hansen

Present: Sadie Rowe, Lucy Heckerman, Presley Hansen, Maddie Michels, Kate Hiller

Goals: Document process of rebuilding dataset with consistent sutures and use of the Pi HQ Camera

Content:

CODE Used:

rpicam-still -t 0 # live preview (use Ctrl+C to exit)

- Mark center of screen

rpicam-still -t 8000 -o test.jpg

- File name was changed according to Tight/Loose and number of knot (EX: tight1)

3/18: Team took photos of a total of 38 Loose and 31 Tight

3/19: Team finished taking photos, with the following final counts

- Loose: 115
- Tight: 122

Example Image:

Conclusions/action items: Next steps include image processing (cropping & making images binary)

4/9/2026 - Raspberry Pi Workflow

SADIE ROWE - Apr 09, 2026, 10:00 AM CDT

Title: Raspberry Pi Workflow

Date: 4/9/2026

Content by: Sadie Rowe, Maddie Michels, Kate Hiller

Present: Sadie Rowe, Maddie Michels, Kate Hiller

Goals: Document set-up of Raspberry Pi Workflow

Content:

General Process Flow:

1. Press button (1): opens display of camera to position knot in focus (camera preview)

2. Press button again (2): picture captured

3. Image preprocessed

4. Image sent through model

5. Model output saved in CSV file

6. LED indication of progress:

- Yellow LED = processing
- Green LED = adequate knot (tight)
- Red LED = too loose

Necessary Package Downloads:

RPI.GPIO

- Install method: sudo apt install python3-rpi.gpio

Import time

Current Code:

```
#!/usr/bin/env python3
"""Capture an image via button press, run model, signal result via LEDs,
and save to CSV with timestamp."""
import sys
import subprocess
import time
import csv
```

```
from datetime import datetime
from pathlib import Path
import torch
from torchvision import transforms
from PIL import Image
import RPi.GPIO as GPIO
# --- GPIO Pin Config ---
PIN_BUTTON = 17
PIN_LED_PROCESSING = 27   # LED1: on while model runs
PIN_LED_TIGHT = 22        # LED2: on if prediction is Tight
PIN_LED_LOOSE = 23        # LED3: on if prediction is Loose



# --- Paths and Config ---
MODEL_PATH = "/home/knotorious-five/squaredetect/top_and_side_resnet_full.pth"
IMAGES_DIR = Path(__file__).parent / "images"
RESULTS_CSV = Path(__file__).parent / "results.csv"
CLASS_MAPPING = {"Loose": 0, "Tight": 1}

# Create images directory
IMAGES_DIR.mkdir(exist_ok=True)



# --- Image Transforms ---
DATA_TRANSFORMS = transforms.Compose([
    transforms.Resize((500, 500)),
    transforms.ToTensor(),
])




def setup_gpio():
    """Configure GPIO pins."""
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(PIN_BUTTON, GPIO.IN, pull_up_down=GPIO.PUD_UP)
    GPIO.setup(PIN_LED_PROCESSING, GPIO.OUT, initial=GPIO.LOW)
    GPIO.setup(PIN_LED_TIGHT, GPIO.OUT, initial=GPIO.LOW)
    GPIO.setup(PIN_LED_LOOSE, GPIO.OUT, initial=GPIO.LOW)



def all_leds_off():
    """Turn all LEDs off."""
    GPIO.output(PIN_LED_PROCESSING, GPIO.LOW)
    GPIO.output(PIN_LED_TIGHT, GPIO.LOW)
    GPIO.output(PIN_LED_LOOSE, GPIO.LOW)



def show_preview_and_wait_for_capture():
```

```
    """Show camera preview until button is pressed again."""
    print("Starting preview... Press button to capture.")

    try:
        from picamera2 import Picamera2

        cam = Picamera2()
        config = cam.create_preview_configuration()
        cam.configure(config)
        cam.start()

        # Wait for second button press
        GPIO.wait_for_edge(PIN_BUTTON, GPIO.FALLING)
        time.sleep(0.05)  # Debounce

        return cam

    except ImportError:
        print("picamera2 not available - no preview mode")
        print("Waiting 2 seconds for second button press...")
        time.sleep(2)  # Give user time to press again
        return None


def capture_image(output_path: str, cam=None) -> str:
    """Capture an image using the RPi camera."""
    try:
        if cam is not None:
            # Use existing camera instance from preview
            cam.capture_file(output_path)
            cam.stop()
            cam.close()
            print(f"Captured image (picamera2): {output_path}")
            return output_path
        else:
            # Try to create new camera instance
            from picamera2 import Picamera2
            cam = Picamera2()
            config = cam.create_still_configuration()
            cam.configure(config)
            cam.start()
            cam.capture_file(output_path)
            cam.stop()
            cam.close()
            print(f"Captured image (picamera2): {output_path}")
            return output_path



    except ImportError:
        print("picamera2 not available, falling back to rpicam-still...")
        result = subprocess.run(
```

```
            ["rpicam-still", "-o", output_path, "--nopreview", "-t", "1000"],
            capture_output=True, text=True,
        )
        if result.returncode != 0:
            print(f"rpicam-still failed:\n{result.stderr}", file=sys.stderr)
            sys.exit(1)
        print(f"Captured image (rpicam-still): {output_path}")
        return output_path



def load_image(image_path: str) -> torch.Tensor:
    """Load and preprocess an image for the model."""
    img = Image.open(image_path).convert("RGB")
    img = DATA_TRANSFORMS(img)
    img = img.unsqueeze(0)
    return img



def predict(image_path: str, model: torch.nn.Module) -> tuple[str, float]:
    """Run model inference on an image and return (class_name, confidence)."""
    img = load_image(image_path)
    with torch.no_grad():
        output = model(img)
    probabilities = torch.exp(output)
    prediction = probabilities.max(dim=1)[1].item()
    confidence = probabilities.max(dim=1)[0].item()



    class_name = next(k for k, v in CLASS_MAPPING.items() if v == prediction)
    return class_name, confidence



def get_next_index(images_dir: Path) -> int:
    """Find highest existing image index to avoid collisions."""
    existing = list(images_dir.glob("*.jpg"))
    if not existing:
        return 1
    return max(int(f.stem.split("_")[0]) for f in existing) + 1



def log_result(timestamp: str, image_path: str, class_name: str, confidence: float):
    """Append prediction result to the CSV log file."""
    write_header = not RESULTS_CSV.exists()

    with open(RESULTS_CSV, "a", newline="") as f:
        writer = csv.writer(f)
        if write_header:
            writer.writerow(["timestamp", "image_filename", "prediction", "confidence"])
        writer.writerow([timestamp, Path(image_path).name, class_name, f"{confidence:.4f}"])
```

```
    print(f"Result logged to {RESULTS_CSV}")



def run_capture_and_predict(model: torch.nn.Module):
    """Full pipeline: preview, capture on 2nd press, predict, save, and signal via LEDs."""
    all_leds_off()

    # Generate timestamp and filename
    timestamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    next_index = get_next_index(IMAGES_DIR)
    final_path = IMAGES_DIR / f"{next_index}_capture.jpg"

    # Show preview and wait for second button press
    cam = show_preview_and_wait_for_capture()

    # Capture image
    print("Capturing image...")
    capture_image(str(final_path), cam)

    # Turn on processing LED while model runs
    GPIO.output(PIN_LED_PROCESSING, GPIO.HIGH)
    print("Running model...")

    class_name, confidence = predict(str(final_path), model)
    print(f"Result: {class_name} (confidence: {confidence:.4f})")

    # Rename file to include prediction
    new_path = IMAGES_DIR / f"{next_index}_{class_name.lower()}.jpg"
    final_path.rename(new_path)
    print(f"Image saved: {new_path}")

    # Log result to CSV
    log_result(timestamp, str(new_path), class_name, confidence)

    # Turn off processing LED, turn on result LED
    GPIO.output(PIN_LED_PROCESSING, GPIO.LOW)
    if class_name.lower() == "tight":
        GPIO.output(PIN_LED_TIGHT, GPIO.HIGH)
    else:
        GPIO.output(PIN_LED_LOOSE, GPIO.HIGH)

    # Keep result LED on for 3 seconds
    time.sleep(3)
    all_leds_off(



def main():
    setup_gpio()

    # Load model once at startup
```

```
    print("Loading model...")
    model = torch.load(MODEL_PATH, weights_only=False, map_location="cpu")
    model.eval()
    print("Model loaded. Ready!\n")
    print("Press button to start preview, press again to capture.\n")

    try:
        while True:
            # Wait for first button press (active LOW with pull-up)
            GPIO.wait_for_edge(PIN_BUTTON, GPIO.FALLING)
            time.sleep(0.05)  # Debounce

            if GPIO.input(PIN_BUTTON) == GPIO.LOW:
                print("\n=== Button pressed! Starting capture sequence ===")
                run_capture_and_predict(model)
                print("\n=== Ready for next capture ===\n")

    except KeyboardInterrupt:
        print("\nExiting.")
    finally:
        all_leds_off()
        GPIO.cleanup()



if __name__ == "__main__":
    main()
```

Conclusions/action items: This code will be used in place of the current model code in order to run the entire training workflow on the Pi. Once the new model is updated, it will be swapped in place of the current model (using the same code).

SADIE ROWE - Apr 12, 2026, 4:35 PM CDT

Updated code to fix errors, correct button usage, and include a dot in the center of the preview to center knot:

```
#!/usr/bin/env python3
"""Capture an image via button press, run model, signal result via LEDs,
and save to CSV with timestamp."""



import os
os.environ["GPIOZERO_PIN_FACTORY"] = "lgpio"



import sys
import subprocess
import time
import csv
from datetime import datetime
```

```
from pathlib import Path


import torch
from torchvision import transforms
from PIL import Image
from gpiozero import Button, LED


# --- GPIO Pin Config ---
PIN_BUTTON = 17
PIN_LED_PROCESSING = 27   # LED1: on while model runs
PIN_LED_TIGHT = 23        # LED2: on if prediction is Tight
PIN_LED_LOOSE = 22        # LED3: on if prediction is Loose


# --- Paths and Config ---
MODEL_PATH = "/home/knotorious-five/squaredetect/top_and_side_resnet_full.pth"
IMAGES_DIR = Path(__file__).parent / "images"
RESULTS_CSV = Path(__file__).parent / "results.csv"
CLASS_MAPPING = {"Loose": 0, "Tight": 1}


# Create images directory
IMAGES_DIR.mkdir(exist_ok=True)


# --- Image Transforms ---
DATA_TRANSFORMS = transforms.Compose([
    transforms.Resize((500, 500)),
    transforms.ToTensor(),
])


# --- GPIO Setup (FIXED) ---
button = Button(PIN_BUTTON, pull_up=True, bounce_time=0.1)
led_processing = LED(PIN_LED_PROCESSING)
led_tight = LED(PIN_LED_TIGHT)
led_loose = LED(PIN_LED_LOOSE)



def all_leds_off():
    led_processing.off()
    led_tight.off()
    led_loose.off()
```

```
def wait_for_clean_press():
    """Wait for a clean button press (press + release)."""
    while not button.is_pressed:
        time.sleep(0.01)
    while button.is_pressed:
        time.sleep(0.01)
```

```
import subprocess
import time
```

```
import cv2
from picamera2 import Picamera2
import time

def start_preview_with_dot():
    cam = Picamera2()
    config = cam.create_preview_configuration()
    cam.configure(config)
    cam.start()

    print("Preview running (press 'q' to exit preview loop)...")

    while True:
        frame = cam.capture_array()

        h, w, _ = frame.shape
        cx, cy = w // 2, h // 2

        # draw center dot
        cv2.circle(frame, (cx, cy), 6, (0, 0, 255), -1)

        cv2.imshow("Camera Preview", frame)

        if cv2.waitKey(1) & 0xFF == ord('q'):
```

```
            break


        # optional: break on button press
        if button.is_pressed:
            wait_for_clean_press()
            break


    cv2.destroyAllWindows()
    return cam




import cv2
from picamera2 import Picamera2


def start_preview_with_dot():
    cam = Picamera2()
    config = cam.create_preview_configuration()
    cam.configure(config)
    cam.start()


    print("Preview running (press button to capture, or 'q' to quit)...")


    while True:
        frame = cam.capture_array()


        h, w, _ = frame.shape
        cx, cy = w // 2, h // 2


        cv2.circle(frame, (cx, cy), 6, (0, 0, 255), -1)


        cv2.imshow("Camera Preview", frame)


        key = cv2.waitKey(1) & 0xFF
```

```
        if key == ord('q'):
            cam.stop()
            cam.close()
            cv2.destroyAllWindows()
            return None



        if button.is_pressed:
            wait_for_clean_press()
            cv2.destroyAllWindows()
            return cam




def capture_image(output_path: str, cam=None) -> str:
    try:
        # stop preview safely
        if cam is not None:
            cam.stop()
            cam.close()
            print("Preview stopped.")



        result = subprocess.run([
            "rpicam-still",
            "-o", output_path,
            "-t", "1",
            "--nopreview"
        ], capture_output=True, text=True)



        if result.returncode != 0:
            print(result.stderr, file=sys.stderr)
            sys.exit(1)



        print(f"Captured image: {output_path}")
        return output_path



    except Exception as e:
        print(f"Capture failed: {e}")
        sys.exit(1)



def load_image(image_path: str) -> torch.Tensor:
    img = Image.open(image_path).convert("RGB")
```

```
    img = DATA_TRANSFORMS(img)
    img = img.unsqueeze(0)
    return img




def predict(image_path: str, model: torch.nn.Module) -> tuple[str, float]:
    img = load_image(image_path)
    with torch.no_grad():
        output = model(img)



    probabilities = torch.exp(output)
    prediction = probabilities.max(dim=1)[1].item()
    confidence = probabilities.max(dim=1)[0].item()



    class_name = next(k for k, v in CLASS_MAPPING.items() if v == prediction)
    return class_name, confidence




def get_next_index(images_dir: Path) -> int:
    existing = list(images_dir.glob("*.jpg"))
    if not existing:
        return 1
    return max(int(f.stem.split("_")[0]) for f in existing) + 1




def log_result(timestamp: str, image_path: str, class_name: str, confidence: float):
    write_header = not RESULTS_CSV.exists()



    with open(RESULTS_CSV, "a", newline="") as f:
        writer = csv.writer(f)
        if write_header:
            writer.writerow(["timestamp", "image_filename", "prediction", "confidence"])
        writer.writerow([timestamp, Path(image_path).name, class_name, f"{confidence:.4f}"])



    print(f"Result logged to {RESULTS_CSV}")
```

```
def run_capture_and_predict(model: torch.nn.Module):
    all_leds_off()



    timestamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    next_index = get_next_index(IMAGES_DIR)
    final_path = IMAGES_DIR / f"{next_index}_capture.jpg"



    cam = start_preview_with_dot()
    print("Capturing image...")
    capture_image(str(final_path), cam)



    led_processing.on()
    print("Running model...")



    class_name, confidence = predict(str(final_path), model)
    print(f"Result: {class_name} (confidence: {confidence:.4f})")



    new_path = IMAGES_DIR / f"{next_index}_{class_name.lower()}.jpg"
    final_path.rename(new_path)
    print(f"Image saved: {new_path}")



    log_result(timestamp, str(new_path), class_name, confidence)



    led_processing.off()
    if class_name.lower() == "tight":
        led_tight.on()
    else:
        led_loose.on()



    time.sleep(3)
    all_leds_off()




def main():
    print("Loading model...")
    model = torch.load(MODEL_PATH, weights_only=False, map_location="cpu")
    model.eval()
```

```
    print("Model loaded. Ready!\n")
    print("Press button to start preview, press again to capture.\n")



    try:
        while True:
            wait_for_clean_press()



            print("\n=== Button pressed! Starting capture sequence ===")
            run_capture_and_predict(model)
            print("\n=== Ready for next capture ===\n")



    except KeyboardInterrupt:
        print("\nExiting.")
    finally:
        all_leds_off()




if __name__ == "__main__":
    main()
```

4/26/2026 - Dataset Creation Summary

SADIE ROWE - Apr 26, 2026, 4:47 PM CDT

Title: Dataset Creation Summary

Date: 4/26/2026

Content by: Sadie Rowe

Present: N/A

Goals: Document dataset creation and comparison of preprocessing techniques to summarize in report

Content:

Summarized K-fold results: Unedited images achieved with highest average accuracy --> used to train ResNet50 model.

Dataset Type	Average Accuracy	Average Recall	Average Precision	Average F1 Score
Binary Images	53.00	0.5300	0.3386	0.3877
Contrasted Images	61.58	0.6158	0.5216	0.5166
Unedited Images	73.93	0.7393	0.7651	0.6930

- Created a dataset to train a ResNet50 CNN for classifying suture knots
- Two classes: tight (adequate) and loose (inadequate)
- Used black 2-0 monofilament sutures on a light silicone skin pad for consistency
- Each knot: square knot with four throws; final throw determines class
- Sutures placed at 0.5-inch intervals; 100 samples per class (200 total)
- Images captured under consistent conditions:
  - Diffuse daylight
  - ~45° camera angle
  - Fixed focus using Arducam IMX477 Pi HQ Camera
- Images saved as JPEGs and transferred from Raspberry Pi for processing
- Preprocessing steps:
  - Center-cropped to 500 × 500 pixels

    - Removed background (skin pad edges, adjacent knots)
    - Standardized framing with knot centered
- Dataset labeled and validated as tight or loose
- To improve performance with limited data, multiple preprocessing methods were tested:
    - Binarization: converted images to black-and-white to emphasize edges and knot structure
    - Contrast scaling (1.5×): enhanced differences between suture and background
- Compared three datasets:
    - Binarized images
    - Contrast-enhanced images
    - Unedited images
- Used 6-fold cross-validation to evaluate preprocessing methods
- Goal: identify the preprocessing approach that yields the most accurate and generalizable model before final training

Conclusions/action items: Summarize information into report.

4/26/2026 - Workflow implementation

SADIE ROWE - Apr 26, 2026, 4:52 PM CDT

Title: Workflow Implementation

Date: 4/26/2026

Content by: Sadie Rowe

Present: N/A

Goals: Document workflow of full training system to document in the report

Content:

- Deployed trained ResNet model on a **Raspberry Pi 5** for a physical training module
- Platform chosen for **compact size** and ability to perform **on-device inference** in a portable setting
- Integrated hardware components:
  - Arducam IMX477 Pi HQ Camera (same as dataset collection)
  - Custom circuit board with **3 LEDs** and a **user input button**
- Designed a **custom 3D-printed mount**to:
  - Fix camera position and angle
  - Ensure consistency with training image conditions
- Final prototype includes:
  - Raspberry Pi 5
  - Camera + LED/button circuit
  - Wireless connection to external display for live preview
- System workflow:
  - Button press → activates live camera preview
  - On-screen reference marker helps center the knot
  - Second button press → captures image and runs model pipeline
    - Includes center-cropping and classification
- User feedback via LEDs:
  - **Yellow LED** → processing in progress
  - **Green LED** → tight (adequate) knot
  - **Red LED** → loose (inadequate) knot

Conclusions/action items: Compile information into the report.

4/29/2026 - Image Preprocessing Comparison

SADIE ROWE - Apr 29, 2026, 1:01 PM CDT

Title: Image preprocessing analysis

Date: 2/29/2026

Content by: Sadie Rowe

Present: N/A

Goals: Address pros and cons of each preprocessing technique compare

Content:

- Both techniques were applied and compared using K-fold Cross Validation
- Binarization had issues primarily with depth (if there was overlap between the loose ends, it created the appearance of another hole)
  - poor at detecting tight knots
- Contrast highlighted shadows and made the sutures blurred in some areas due to overexposure
  - shadows could have been mistaken as suture material
  - poor at detecting loose knots

	Binarization	Contrast
Pros	Clearly defines edge boundaries Negates lighting inconsistencies	Reduces background noise while preserving structure and depth
Cons	Oversimplifies knot features Reduces depth cues	Exaggerates shadows Overexposes suture material

Conclusions/action items: Document in report and final deliverables

3/16/2026 - Programming Foundations: Artificial Intelligence Training

SADIE ROWE - Mar 16, 2026, 6:20 PM CDT

Title: Programming Foundations: Artificial Intelligence Training

Date: 3/16/2026

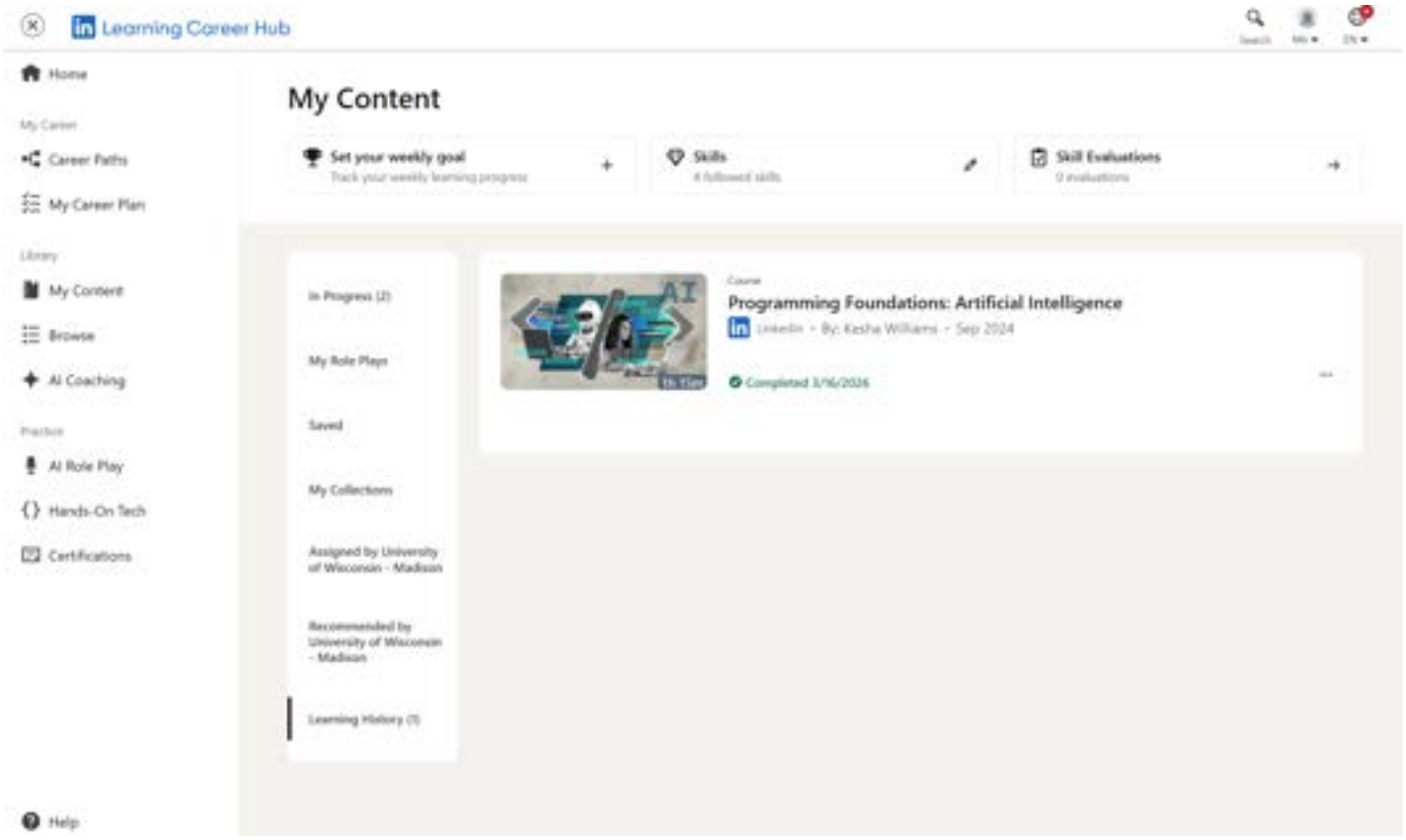
Content by: Sadie Rowe

Present: N/A

Goals: Document Artificial Intelligence Training through LinkedIn learning. This training is intended to satisfy the BME Design: training throughout the curriculum assignment

Content:

LinkedIn Learning Completed Content Dashboard:



Conclusions/action items: This training was very helpful and applicable to our project. It helped me fill in some of the gaps in my understanding, specifically with background regarding what each of the Python packages do.

2025/10/27 - Animal User Training

SADIE ROWE - Oct 27, 2025, 8:56 PM CDT

Title: Training throughout the Curriculum Documentation

Date: 10/27/2025

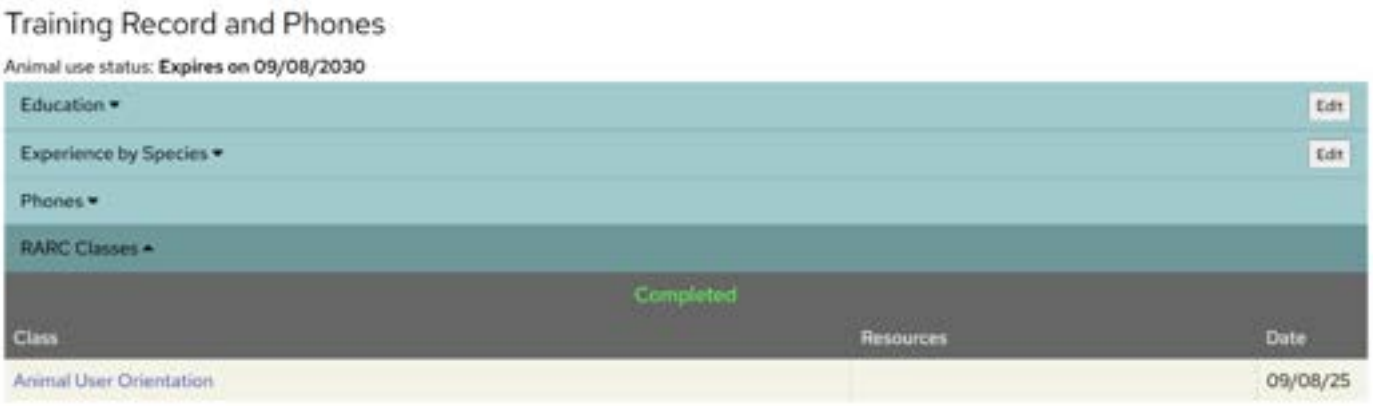
Content by: Sadie Rowe

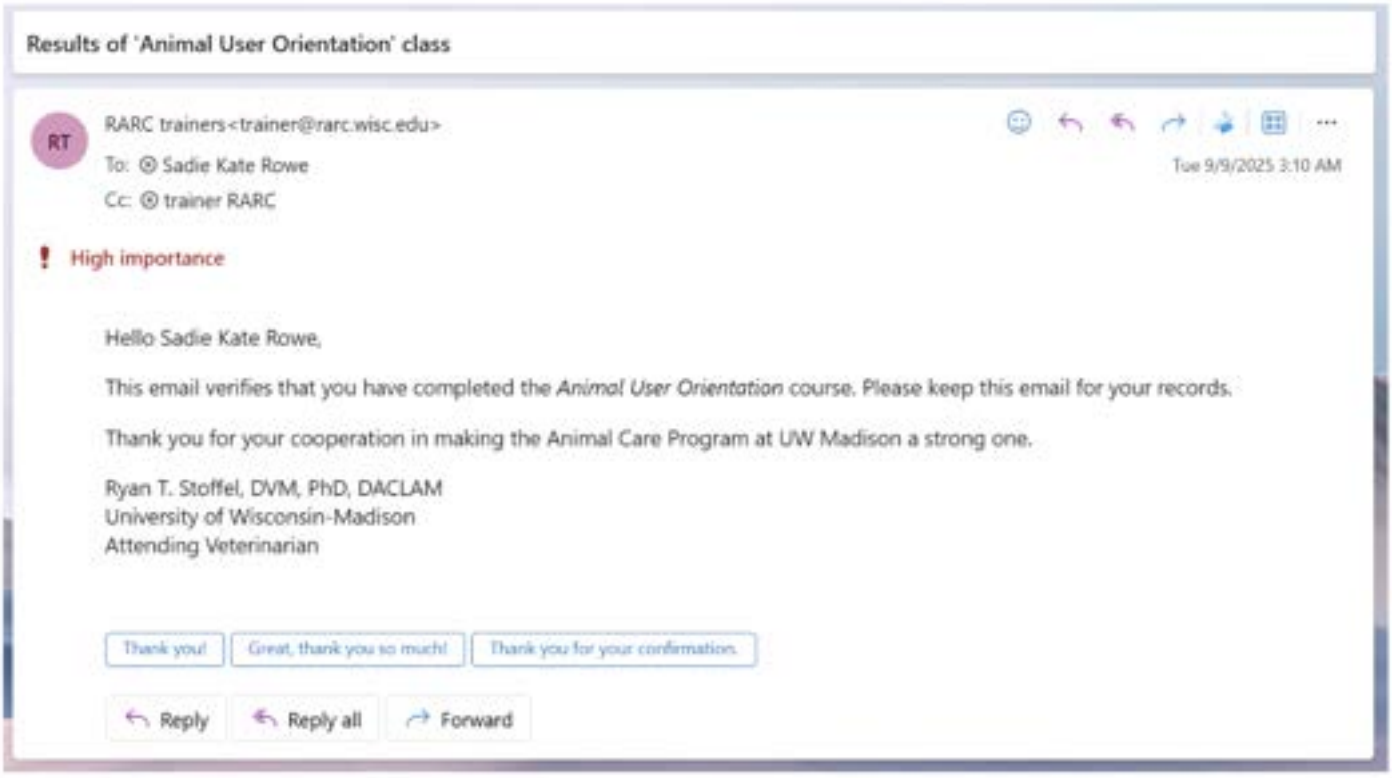
Present: N/A

Goals: Document Training

Content:

I completed the animal user orientation course as a precautionary step for my design project involving the development of a real-time suture tensioning device. While our team is not conducting research on animals with this device, I wanted to gain a clear understanding of the guidelines surrounding animal use in research to be better prepared in the case of future applications. Since our project involves direct collaboration with a veterinarian, this training ensures I am informed and aligned with the proper standards.





Conclusions/action items: N/A

2024/03/04 - Intro to Machining TEAMLab Permit

SADIE ROWE - Mar 04, 2024, 9:42 PM CST

Title: Intro to Machining TEAM Lab Permit

Date: 04/03/2024

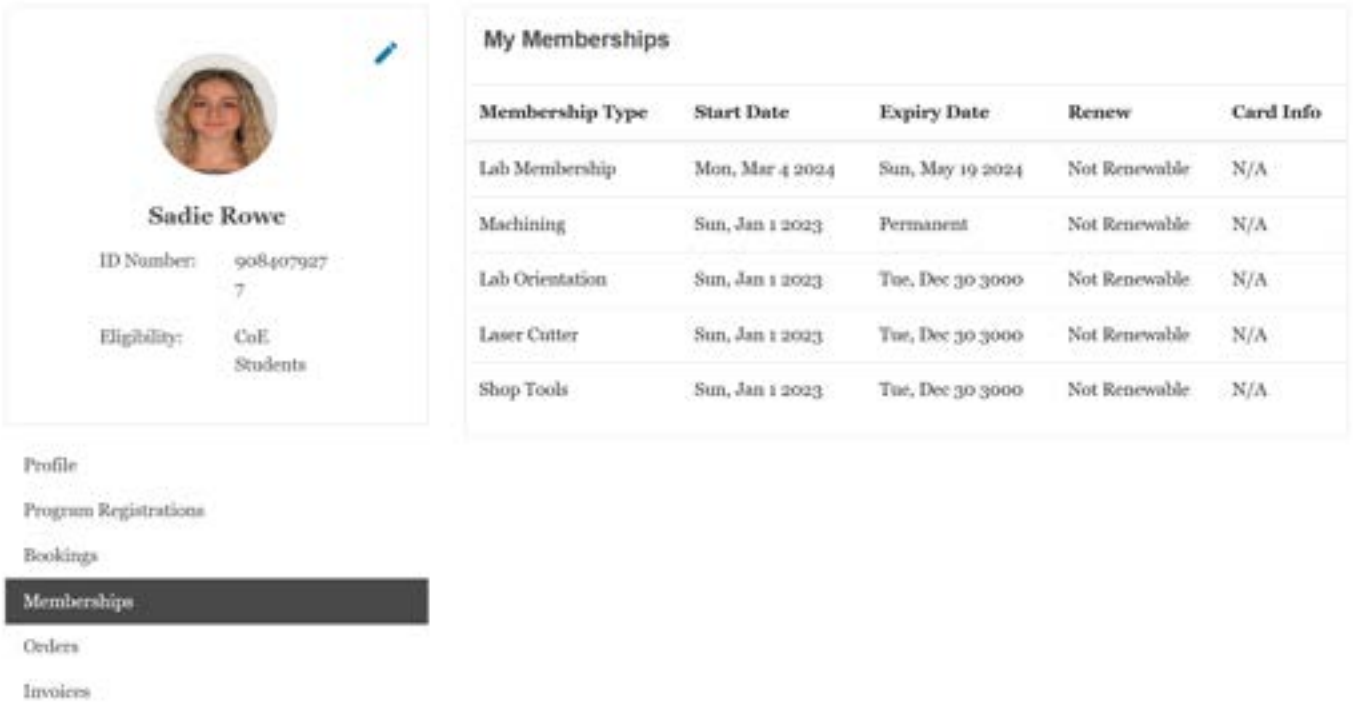
Content by: Sadie Rowe

Present: Sadie Rowe

Goals: Document Completion for Intro to Machining Permit

Content:

Date of Completion: 04/03/2024



Conclusions/action items: N/A

2024/02/17 - Online Training Completion

SADIE ROWE - Feb 17, 2024, 2:50 PM CST

Title: Into to Machining online training completion

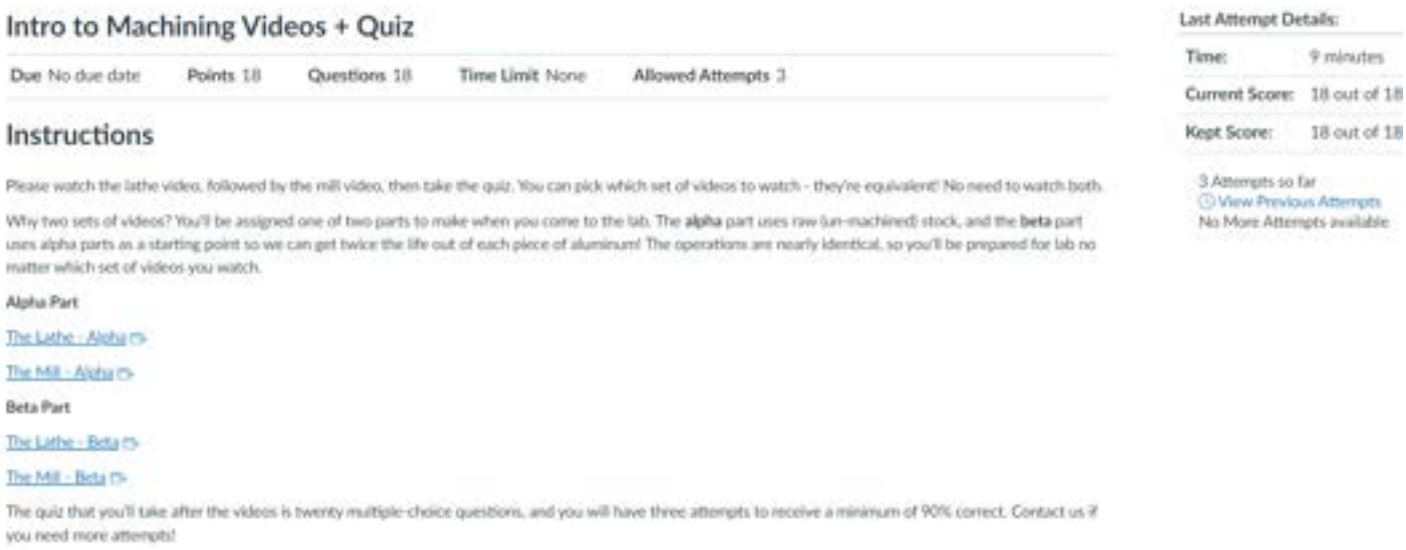
Date: Date of Completion: 01/28/2024

Content by: Sadie Rowe

Present: Sadie Rowe

Goals: Complete online training course for Green permit

Content:



Intro to Machining Videos + Quiz

Due No due date Points 18 Questions 18 Time Limit None Allowed Attempts 3

Instructions

Please watch the lathe video, followed by the mill video, then take the quiz. You can pick which set of videos to watch - they're equivalent! No need to watch both.

Why two sets of videos? You'll be assigned one of two parts to make when you come to the lab. The **alpha** part uses raw (un-machined) stock, and the **beta** part uses alpha parts as a starting point so we can get twice the life out of each piece of aluminum! The operations are nearly identical, so you'll be prepared for lab no matter which set of videos you watch.

Alpha Part

The Lathe - Alpha

The Mill - Alpha

Beta Part

The Lathe - Beta

The Mill - Beta

The quiz that you'll take after the videos is twenty multiple-choice questions, and you will have three attempts to receive a minimum of 90% correct. Contact us if you need more attempts!

Last Attempt Details:

Time:	9 minutes
Current Score:	18 out of 18
Kept Score:	18 out of 18

3 Attempts so far
View Previous Attempts
No More Attempts available

Conclusions/action items:

- Complete training in lab on lathe and mill

2024/02/17 - Lathe Training

SADIE ROWE - Feb 17, 2024, 2:53 PM CST

Title: Intro to Machining proof of Lathe Training

Date: Date of Completion: 02/13/2024

Content by: Sadie Rowe

Present: Sadie Rowe

Goals: Create Beta Part on the Lathe

Content: Completed Beta Part:

Conclusions/action items:

- Complete Beta Part training on Mill

2024/01/25 - Biosafety Required Training and Chemical Safety Training

SADIE ROWE - Jan 25, 2024, 6:55 PM CST

Download

Biosafety_and_Chem_Safety_Training.pdf (47.9 kB)

2022/9/22 - Red Permit

SADIE ROWE - Dec 07, 2023, 2:44 PM CST

Title: Red Permit

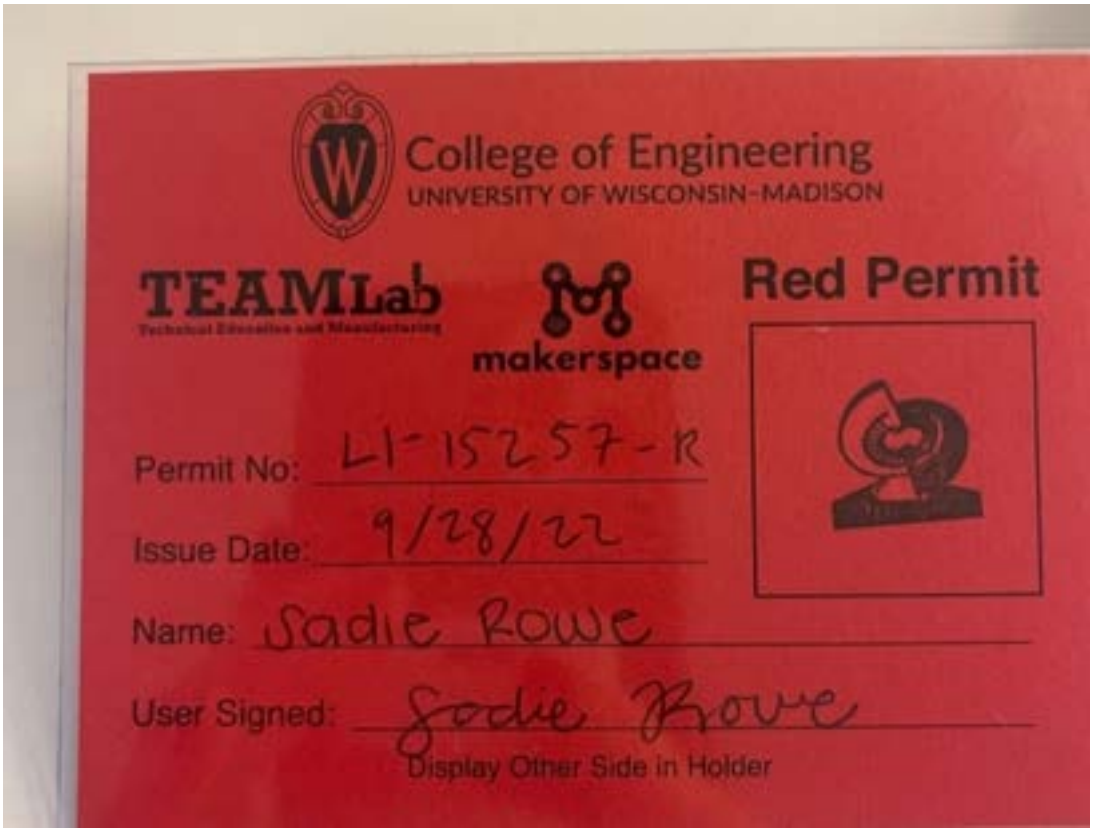
Date: 2022/9/22

Content by: Sadie Rowe

Present: N/A

Goals: Show Proof or Red Permit

Content:



Conclusions/action items: N/A

3/6/2026 - Tong Lecture

SADIE ROWE - Mar 06, 2026, 4:49 PM CST

Title: Tong Lecture

Date: 3/6/2026

Content by: Prof. Justin Williams, PhD

Present: Sadie Rowe

Goals: Document notes from Tong lecture as well as main takeaways

Content:

Background

- Grew up on a farm in a small rural town
- Attended the University of South Dakota on a track scholarship while working at Daktronics
- A close friend’s death from a traumatic brain injury motivated him to pursue graduate school in biomedical engineering at Arizona State University
- Later worked in a rat brain research lab in Michigan
- Eventually became involved in developing deep brain stimulation (DBS) technology at the University of Wisconsin–Madison

Neural Integration Technologies

- The company was later acquired by Gore
- Gore ultimately discontinued the technology to focus on its own products

NeuroNexus

- Developed neural electrode arrays used in neuroscience research
- Was acquired by Greatbatch but later regained independence
- Now supplies a large amount of electrophysiology equipment used in neural research

NeuroOne

- Focused on the idea that small design improvements can still have a big impact
- Emphasized designing technology with the end user in mind
- Spent time in operating rooms learning from neurosurgeons and helping implement DBS technology
- After seeing epilepsy surgeries using large skull openings, a surgeon challenged him to create a better solution

- This led to flexible electronics that require less implanted material and smaller incisions
- The company later went public and captured about 12% of the epilepsy market

BrainSync

- Highlighted how market size and timing affect whether a technology succeeds
- Used electrodes over the motor cortex so patients could control an arm using brain signals
- Originally targeted ALS patients
- Enabled the first tweet sent using only brain signals with a non-invasive EEG
- The ALS market was small, so the technology was eventually made freely available
- Later adapted for stroke rehabilitation using brain signals and muscle stimulation

NeuraWorx

- Focuses on progressive supranuclear palsy (PSP), a disease related to tau buildup in the brain
- Research centers on the glymphatic system, which removes waste from the brain during sleep
- Aging and brain injuries can reduce this waste clearance
- Developed devices that stimulate cranial nerves using a mouthguard and a headband
- Long-term goal is a micro-implantable version of the technology

Key Takeaways

- Small innovations can still be impactful
- Learning from surgeons and clinical settings is important
- Market timing and patient population size strongly influence commercial success

Conclusions/action items: Dr. Williams work demonstrates how impactful biomedical technologies often come from small but thoughtful improvements from tools. He was able to absorb the information around him and learn about real problems before taking action. However, I also liked his very realistic perspective regarding the dependency of commercial success on market demand.

02/01: Raspberry Pie Basics

Lucia Hockerman - Feb 01, 2026, 2:13 PM CST

Title: Getting started with your Raspberry Pi

Date: 02/01/2026

Content by: Lucy Hockerman

Goals: To begin research on machine learning model integration into a user-friendly system, I need to gather beginner basics on Raspberry Pi and set-up because I have no experience using this technology

Content:

Raspberry Pi overview: a single-board computer used for learning programing, electronics, and general computing.

Boot: short for bootstrapping and its the process a computer goes through when it is powered on to start up and load the system

Required hardware:

- Core: Raspberry Pi board, model-specific power supply
- Storage: microSD card (≥8 GB, 32 GB recommended), or USB/NVMe boot options
- Monitor, keyboard, mouse
- Networking: Wi-Fi (built-in on most boards)

Tip: Use a quality power supply to avoid instability.

Boot Media and operating system:

- Raspberry Pi doesn't include onboard storage.
- Recommended method: Raspberry Pi Imager
    - Runs on Windows/macOS/Linux
    - Select Pi model and OS (Raspberry Pi OS recommended)
    - Write image to microSD or USB

Model differences in display output and power supply

Model	Display outputs
Raspberry Pi 5, Raspberry Pi 400, Raspberry Pi 500, and Raspberry Pi 500+	2× micro HDMI
Raspberry Pi 4 Model B	2× micro HDMI, audio and composite out via 3.5 mm TRRS jack
Raspberry Pi 3 (all models)	HDMI, audio and composite out via 3.5 mm TRRS jack
Raspberry Pi 2 Model B	HDMI, audio and composite out via 3.5 mm TRRS jack
Raspberry Pi 1 Model A+ and Model B+	HDMI, audio and composite out via 3.5 mm TRRS jack
Raspberry Pi 1 Model A and Model B	HDMI, audio out via 3.5 mm TRRS jack, composite out via phono jack
Raspberry Pi Zero (all models)	mini HDMI, composite out via solder pads

Model	Recommended power supply (voltage/current)	Raspberry Pi power supply
Raspberry Pi 500+	5V/5A	27 W USB-C power supply
Raspberry Pi 5 and Raspberry Pi 500	5V/5A, 5V/3A limits peripherals to 600mA	27 W USB-C power supply
Raspberry Pi 4 Model B and Raspberry Pi 400	5V/3A	15 W USB-C power supply
Raspberry Pi 3 (all models)	5V/2.5A	12.5 W Micro USB power supply
Raspberry Pi 2 Model B	5V/2.5A	12.5 W Micro USB power supply
Raspberry Pi 1 (all models)	5V/2.5A	12.5 W Micro USB power supply
Raspberry Pi Zero (all models)	5V/2.5A	12.5 W Micro USB power supply

First Boot & Configuration

1. Insert boot media and peripherals
2. Connect power → Pi boots
3. Configuration wizard prompts:
    - Locale (country, language, keyboard layout)
    - Wi-Fi network setup
    - Create user & password
    - Update software

Desktop & Software

- Raspberry Pi OS provides a GUI desktop (if installed)
- Recommended Software app: install utilities, productivity, and programming tools
- Terminal: access Linux shell for advanced operations
- Python and GPIO: pre-installed for learning coding and controlling electronics

Next Steps

- Explore tutorials for programming, electronics projects, and learning Linux
- Update software regularly
- Connect peripherals (camera, sensors) to explore Raspberry Pi's hardware capabilities
- Join the Raspberry Pi community forums for support

Conclusions/action items:

I have a better understanding of Raspberry Pi basics and can refer to this entry for set-up steps. Refer to the link for helpful instruction photos.

Link: https://www.raspberrypi.com/documentation/computers/getting-started.html

02/01: Deploying RoboFlow on Raspberry Pi 5

Lucia Hockerman - Feb 01, 2026, 2:35 PM CST

Title: RoboFlow on Raspberry Pi 5

Date: 02/01/2026

Content by: Lucy Hockerman

Goals: Provide step-by-step instructions on how to use RoboFlow with Raspberry Pi in preparation if we were to chose the RoboFlow model in our training system.

Content:

Step 1: Install Docker

Step 2: Pull the Inference Server

Pull the Roboflow inference server image from Docker Hub:
This will automatically detect your Raspberry Pi's architecture and download the appropriate version.

```
sudo docker pull roboflow/inference-server:cpu
```

- **For Raspberry Pi (ARM CPU)**:
  If you are using a Raspberry Pi or another ARM-based device, you can pull and run the optimized Docker container with the following command:

  ```
  sudo docker run -it --rm -p 9001:9001 roboflow/roboflow-inference-server-arm-cpu
  ```

Step 3: Run the Inference Server

Run the inference server, passing through your network card:

```
sudo docker run --net=host roboflow/inference-server:cpu
```

Or

```
sudo docker run -it --rm -p 9001:9001 roboflow/roboflow-inference-server-arm-cpu
```

Verify this works:

After starting the server, verify it is running by navigating to `http://localhost:9001` in your browser on the Raspberry Pi (or using `curl` in headless mode). A welcome message indicates the server is operational.

```
curl http://localhost:9001
{
    "server": {
        "package": "roboflow-inference-server-cpu",
        "version": "1.4.0"
    },
    "roboflow": {
        "package": "roboflow-node",
        "version": "0.2.25"
```

```
    }
}
```

Step

Set

```
python -m venv roboflow
source roboflow/bin/activate
```

Insta

```
pip install roboflow
```

Step 5: Test Your Setup with a Python Script

Create a Python script (`infer.py`) to test the Roboflow module. Use the following example:

```
import json

# import the Roboflow Python package
from roboflow import Roboflow

VERSION_NUMBER = 1
# instantiate the Roboflow object and authenticate with your credentials
rf = Roboflow(api_key="YOUR_PRIVATE_API_KEY")
# load/connect to your project
#project = rf.workspace("YOUR_WORKSPACE").project("YOUR_PROJECT")
project = rf.workspace().project("YOUR_PROJECT")

# load/connect to your trained model
model = project.version(VERSION_NUMBER, local="http://localhost:9001").model

# perform inference on an image file
prediction = version.model.predict("YOUR_IMAGE.jpg")
# print prediction results in JSON
print(prediction.json())

json_string = prediction.json()
formatted_json = json.dumps(json_string), indent=4)
print(formatted_json)

# predict on a local image
prediction = model.predict("YOUR_IMAGE.jpg")

# Predict on a hosted image via file name
prediction = model.predict("YOUR_IMAGE.jpg", hosted=True)

# Predict on a hosted image via URL
prediction = model.predict("https://...", hosted=True)

# save inference image
prediction.save("result.jpg")

# Plot the prediction in an interactive environment
```

```
prediction.plot()

# Convert predictions to JSON
print(prediction.json())
```

- Repl
  your
- Save

```
python3 infer.py
```

Step 6: Enable

```
model = project.version(VERSION_NUMBER, local="http://<Raspberry_Pi_IP>:9001/").model
```

Conclusions/a

I can now refer

02/01: Machine Learning Integration Example

Lucia Hockerman - Feb 01, 2026, 8:41 PM CST

Title: PCB Defect Detection with Computer Vision - Raspberry Pi

Date: 02/01/2026

Content by: Lucy Hockerman

Goals: Through this research, I wanted to explore the existing systems that include a machine learning software, a Raspberry Pi and a camera in hope it will help in our own knot-detection system integration.

Content:

Key definitions:
- FOMO is a specific object-detection model architecture optimized for tiny/embedded devices

- PCB stands for Printed Circuit Board. A PCB is the flat board that mechanically supports and electrically connects electronic components

Project/system overview: The idea for this project is to reduce the number of defects on PCBs by creating a machine learning system to identify three common defects on a PCB (missing holes, open and short circuits). The project was completed using FOMO (rather than TensorFlow or RoboFlow utilized in our project).

Can run this project here: https://studio.edgeimpulse.com/public/135820/latest

The dataset had a total of 1366 images: 887 for training, 253 for testing (380 for each defect in training and testing)

Uses edge impulse studio (Impulse Design) for the machine learning pipeline that is packaged in one of the deployments.

Impulse design looks like it can be a helpful program for our system, providing helpful metrics (F1 score, inference time, peak RAM, accuracy, etc). It also claims to make it easier to integrate into Raspberry Pi.

More on impulse design: https://docs.edgeimpulse.com/studio/projects/impulse-design

More on deployment: https://docs.edgeimpulse.com/studio/projects/deployment

Deploying to a Raspberry Pi 4: Flash the raspberry Pi OS image to an SD card, install Edge Impulse dependencies on the Raspberry Pi and connect a camera to the Pi (used the 8 megapixel V2 camera)

More specific instructions found here: https://docs.edgeimpulse.com/hardware/boards/raspberry-pi-4

Link: https://docs.edgeimpulse.com/projects/expert-network/pcb-defect-detection-with-computer-vision-raspberry-pi

Conclusions/action items: Based on this project, I want to discuss with the group about using edge impulse studio as this could be an easier way to implement our current model into a training system using Raspberry Pi. The developer plan is free.

02/19: How to Autostart on Raspberry Pi

Lucia Hockerman - Feb 20, 2026, 9:46 AM CST

Title: Run Script on start-up with your Raspberry Pi

Date: February 19, 2026

Content by: Lucy

Goals: The team met with Dr. Adamczyk to discuss his advice on integrating our model onto a fully functioning training system involving Raspberry Pi. He discussed the option for autostart, so the code is functional when a student potentially walks up to our training system, no other action would be needed. The goal of this research is to dive into the resources he mentioned in order to apply this function to our prototype.

Content:

This source provides a guide on how to automatically execute scripts or programs when a Raspberry Pi boots up, which is particularly useful for **headless setups** where no monitor is connected. It details two primary methods for achieving this: using `rc.local` for system-wide startup and `.bashrc` for user-specific terminal sessions.

Method 1: Using `rc.local`

The `rc.local` file is recommended for running commands during the boot sequence.

- **Setup:** Edit the file as root using `sudo nano /etc/rc.local` and add your command before the `exit 0` line.
- **Best Practices:**
  - Always use **absolute filenames** (e.g., `/home/pi/myscript.py`) instead of relative paths.
  - Ensure the file is executable by running `sudo chmod +x /etc/rc.local`.
- **Non-Blocking Execution:** To prevent a long-running script from stalling the boot process, add an **ampersand (`&`)** to the end of the command to fork it into a separate process.
- **Network and Logging:**
  - If the script requires internet, add `sleep 5` to allow time for a connection to be established.
  - To capture output for debugging, you can redirect the script's output to a **logfile** using a bash command.

Method 2: Using `.bashrc`

Alternatively, you can use the `.bashrc` file if you want a script to run every time a **terminal window is opened**, including during boot or when establishing a new SSH connection.

- **Setup:** Edit the file via `sudo nano /home/pi/.bashrc` and add your commands to the very last line.
- **Control:** Unlike `rc.local` processes, programs started this way can be manually aborted using `ctrl-c` while they are running

Link: https://raspberrypi-guide.github.io/programming/run-script-on-boot

Conclusions/action items: Write this code into our model to apply the autostart function.

02/19: Five Ways to Run a Program on Raspberry Pi

Lucia Hockerman - Feb 20, 2026, 9:59 AM CST

Title: Five Ways to Run a Program on Raspberry Pi at Startup

Date: February 19, 2026

Content by: Lucy

Goals: The team recently received our Raspberry Pi and I wanted to research different methods to run a program on a Raspberry Pi so the team has options and we can pick the best method that fits our materials and project.

Content:

This article outlines five distinct methods for automatically running programs on a Raspberry Pi at startup *(options 1 and 2 were mentioned in my "02.19: How to Autostart" notebook entry)*:

1. rc.local

This method is ideal for **headless setups** where a program needs to run automatically without manual configuration.

- **Implementation:** Commands are added to `/etc/rc.local` before the `exit 0` line.
- **Key Requirement:** Use **absolute file paths** (e.g., `/home/pi/sample.py`).
- **Process Forking:** If a script runs in an infinite loop, you must add an **ampersand (`&`)** to the end of the command; otherwise, the Pi will fail to complete its boot sequence.

2. .bashrc

The `.bashrc` file executes commands every time a user logs in, opens a new terminal window, or connects via SSH.

- **Usage:** Add the command to the bottom of the `/home/pi/.bashrc` file.
- **Control:** Unlike background processes, programs launched here can be manually aborted using **Ctrl+C** while running.

3. init.d Directory

This directory contains scripts that are triggered during the boot, reboot, or shutdown processes.

- **Setup:** The script must be moved to `/etc/init.d/` and formatted as a **Linux Standard Base (LSB)** init script with specific header information (dependencies and run levels).
- **Activation:** The script must be made executable (`chmod +x`) and registered using the `update-rc.d` command.

4. systemd

Standard on Raspbian Jessie and newer, **systemd** provides the most robust control over program execution.

- **Unit Files:** Users create a service file (e.g., `sample.service`) in `/lib/systemd/system/`.
- **Sequencing:** It allows for precise timing; for example, setting `Type=idle` ensures the script only runs after all other services have loaded.
- **Management:** Requires setting file permissions to 644, reloading the system daemon, and enabling the service via `systemctl`.

5. crontab

While not detailed in depth, the source lists **crontab** as a valid fifth method for scheduling programs to run at boot.

**** Choosing the Right Method**

- **Simple/Headless:** Use `rc.local`.
- **Terminal-based:** Use `.bashrc`.

- **Dependency-heavy:** If your script relies on the **network being connected**, the system time being updated via NTP, or specific directories being mounted, **systemd** or **init.d** are the preferred methods because they allow you to control exactly when in the startup sequence the script executes.

Link: https://www.dexterindustries.com/howto/run-a-program-on-your-raspberry-pi-at-startup/

Conclusions/action items: Based on this article, it seems rc.local will be the best approach for running our model on Raspberry Pi but will discuss with the team to confirm.

02/20: Raspberry Pi GPIO Pinout Guide

Lucia Hockerman - Feb 20, 2026, 11:30 AM CST

Title: Raspberry Pi GPIO Pinout Guide

Date: February 20, 2026

Content by: Lucy

Goals: As discussed in the meeting the Dr. Adamcyzk, Raspberry Pi 5 has a new and updated GPIO pinout that differs from the previous Raspberry Pi models. The purpose of this content is to gain a basic understanding of the GPIO pinout of Pi 5.

Content:

The Raspberry Pi 5 features a **40-pin GPIO (General Purpose Input/Output) header** that serves as the interface for connecting external components such as sensors, motors, and displays. While the physical layout remains consistent with the Raspberry Pi 4, **the internal architecture and software handling have been updated.**

Pin Numbering Systems

There are two primary ways to identify pins, and consistency is required when coding:

- **Physical Numbering:** Pins 1 through 40, following their physical location on the board.
- **BCM Numbering:** Refers to the Broadcom chip's internal GPIO numbers (e.g., Physical Pin 11 is GPIO 17).

Pin Categories and Functions

- **Power Pins:** Provides **5V** (Pins 2, 4) for high-power components and **3.3V** (Pins 1, 17) for low-voltage sensors.
- **Ground (GND):** Multiple pins (6, 9, 14, 20, 25, 30, 34, 39) that act as the circuit's return path.
- **Standard GPIO:** General-purpose pins that can be individually programmed as **input** (reading data) or **output** (controlling devices).
- **Special Function Pins:**Dedicated pins for specific communication protocols:
  - **I²C:** Pins 3 (SDA) and 5 (SCL).
  - **SPI:** Pins 19, 21, 23, 24, and 26.
  - **UART:** Pins 8 (TX) and 10 (RX).
  - **PCM/I2S:** Used for digital audio projects.

Key Updates in Raspberry Pi 5

- **RP1 I/O Controller:** GPIO functions are now managed by a dedicated RP1 chip instead of the main CPU, providing better performance and more consistent timing for I²C and SPI.
- **Software Paths:** For developers using direct library access, the GPIO pins are now located under `/dev/gpiochip4` instead of the older `gpiochip0`.

Safety and Compatibility

- **3.3V Logic Limit:** GPIO pins operate at 3.3V. Connecting **5V directly to a GPIO pin** will likely damage the board; a logic level shifter should be used for 5V components.
- **Function Overlap:** Many special function pins overlap with standard GPIOs. You can use a pin for its special protocol or as a regular GPIO, but **not both simultaneously**.
- **Hardware Compatibility:** Most HATs and accessories designed for the Raspberry Pi 4 will work on the Pi 5 because the physical pinout has not changed.

Conclusions/action items: The article is a good resource to reference as we are building our prototype.

02/26: Temporary Model for Raspberry Pi Implementation

Lucia Hockerman - Feb 26, 2026, 3:00 PM CST

Title: Image Classification on TensorFlow Light

Date: Feb. 26, 2026

Content by: Lucy

Goals: Find a pre-trained image classification model to pilot uploading and running a model on our Raspberry Pi. Ideally the source has simple step-by-step instructions for easy implementation process.

Content:

This source contains three different versions of a model that is trained on 1000+ images to identify various objects and animals (ranging from cats all the way to objects like toilet paper). With this training set, the user can download the code onto Jupyter notebook or python. Once downloaded, the user can upload images already included in the website or upload their own photos to verify the model runs correctly.

The source include folders: sample_pictures, model_files, requirements.sh, classify.py

Sample_pictures: contains 8 pictures of different objects/animals to test the script

Model_files: contains 3 different pre-trained models. All performing the image classification using a common label file that contains 1000 objects. The script uses one model at a time to perform inference. Model is changeable in the script.

Requirements.sh: to install pre-requisites (like Tensorflow Light on Raspberry Pi). Use command 'sudo sh requirements.sh' in Terminal to install

Classify.py: the python script with image classification code

- **Uncomment line 12 or 13 to use a different Model**
- **Use a different image by changing image name at line 44.**

Link to access GitHub: https://github.com/jiteshsaini/Image-Classification-TensorFlowLite/blob/master/sample_pictures/5.jpg

Conclusions/action items: This source includes good information and code to run this model.The next steps include getting this code to run on a laptop and then implement onto Raspberry Pi.

3/2: Setting up Raspberry Pi

Lucia Hockerman - Mar 20, 2026, 9:37 AM CDT

Title: Setting up Raspberry Pi

Date: 3/2

Content by: Lucy

Present: Lucy, Sadie, Kate, Presley, Maddie

Goals: The team just received the raspberry pi. After assembling the hardware (fan, housing, etc), we went to the Makerspace to ask questions and get assistance on uploading our model onto the pi system.

Content:

The main accomplishments of the day was (1) getting the Pi set up to wifi by adding as a device onto wisc net and (2) creating a virtual environment with the correct python version

Advantages of setting up the pi on wifi allowed us to navigate the Pi through an interface on our personal laptops instead of having to externally access the server on a monitor.

The biggest struggle was downgrading the python version. This step is necessary because the Pi automatically sets the python version to 3.13, which is too advanced to run Pytorch (the location of our model). In order to fix this, the team needed to match the version of python on the Pi to the version the model was trained on (3.9).

Initial setup code:

** already had python, pip 3
** Current python version: 3.13

1. Install Raspberry Pi 64 bit OS from Raspberry Pi Images
2. Add customizations
    a. Hostname: knotorious-5
    b. Localization: city, timezone, keyboard layout
    c. User:
        i. Username: knotorious-five
        ii. Password: Badger2026
    d. WiFi: skipped
    e. Remote access: did not establish SSH
3. Write Raspberry Pi 64 bit OS onto micro SD card
4. Eject microSD card from computer
5. Insert microSD card into Raspberry Pi
6. Add peripherals to Raspberry Pi
    a. Keyboard
    b. Mouse
    c. Ethernet cable
    d. Monitor
7. Open terminal
8. Install python 3.11 alongside python 3.13
9. Open virtual environment

Sudo apt install python3.11 python3.11-venv
Current python version: 3.13

Installed:

- Pytorch (had to reboot for successful download)

- Pandas
- Numpy

CODE:

python3 -m venv knotvenv #knotvenv is the virtual environment name

source knotvenv/bin/activate

pip3 install pandas

NORBU (makerspace worker) SETUP
Connecting to Raspberry Pi Connect:

1. Connect device to wifi
    a. The Pi and your computer need to be on the same Wifi
2. Turn on Raspberry Pi Connect and login
3. Create a name for the device
    a. KnotPi
4. Open Raspberry Pi connect on personal laptop

Downgrading Python - version 3.9:

CODE USED:

python3 -m venv knotvenv
source knotvenv/bin/activate
python --version #checking which version is currently downloaded
pyenv install 3.9 #Installing pyenv: copied code from duckduckgo → installed version 3.9

→ next step: how to make 3.9 the working version of python

Install python version 3.9.13:

→ pyenv install 3.9.13
Do this in the (knotnewyay) environment

How to Enter and Exit virtual environment
Exit: deactivate
Enter: source knotvenv/bin/activate
To check what python version: python --version

How to shut off the pi:

- Raspberry Pi icon
- Shut-down

Virtual environment name: knotnewyay

Code used to make virtual environment with Python 3.9
Python3.9 -m venv knotnewyay
Source knotnewyay/bin/activate

Code to install applications:

pip install --update pip
Reboot?
pip install pytorch

To upload Temporary PyTorch Model on Raspberry Pi:

1. USB name: Memorex USB
2. Open python
3. Temporary model path: square_model_weights.pth
4. Physically moved square model into knotnewyay environment folder > lib > python3 >site-packages > torch>serialization
    a. Follow location given in path

CODE:

```
pip3 install torch torchvison
Python
import torch
import torchvision.models as models
squaremodel = models.resnet50()
squaremodel.load_state_dict(torch.load(“square_model_weights.pth”) # path name
torch.load(“square_model_weights.pth”, map_location=torch.device(‘cpu’))
Import djando storages
```

Conclusions/action items: The team successfully set up the Pi but now more work is needed to upload the model correctly

3/16: New knots for model refinement

Lucia Hockerman - Mar 20, 2026, 12:43 PM CDT

Title: New knots for model refinement

Date: 3/16

Content by: Lucy

Goals: Tie new suture knots with consistent suture size and color for model retraining.

Content:

Upon discussion with the team, we decided to retrain the model on darker sutures with a consistent background. This decision was based on the heat map Maddie created that showed the model classifying knots based on the incorrect location on the skin pad. We noticed the most issues arose from the clear/light colored sutures.

I emailed the client to ask for more dark sutured and picked them up from the Vet school. Afterwards, I split up the materials to Kate, Sadie and myself and we created suture knots. We made sure our knots were consistent among the three of us by following the rules below:

1.) Follow the suture technique learned last semester

2.) Tie 4 knots total, leaving the top knot loose. **Once photos are taken of the loose condition, the team will tighten the knot to take tight knot photos (saving suture material and time)**

3.) Create about a finger distance between each surrounding knot

4.) Only tie sutures on the light colored skin pad to remain consistent

5.) Cut the excess suture material after the knot is tied to avoid interference with other knot photos

Suture materials used:

Wego-nylon, black, 2-0 size (80 mm diameter), non-absorbable surgical suture, not for human use

Photo example of loose suture knots:

Photo example of tight suture knots:

Conclusions/action items: I tied 50+ knots and between the three of us, we tied a total of 121 tight knots and 115 loose knots. Next steps include using the HQ camera to take photos of knots and retraining the model.

3/18 and 3/20: Taking Pi photos for model training

Lucia Hockerman - Mar 20, 2026, 10:05 AM CDT

Title: Taking Pi photos for model training

Date: 3/18-3/19

Content by: Lucy

Goals: Now that we have tied new suture knots, the team wants to use or knew raspberry pi camera set up to take consistent photos and retrain the model.

Content:

The team created code that provided HQ camera visual feedback, took a photo when the user indicated (10 second delay), asked user to identify tight or loose knot, and saved in the corresponding folder.

1. Download “capture_and_save.py” code
   (Created by Maddie)

2. Plug in Pi camera, adjust camera settings (turning barrel of camera) on camera so knots are in focus
3. Drag downloaded code into known folder
4. Open new terminal in Pi system:

   Code:
   python3 capture_and_save.py
   * user identifies knot of tight or loose (t or l)

Current size image: 4056x3040 pixels → team is concerned about this size

3/18 code changes:

- Programmed the camera to take 500x500 pixels
- Added a zoom of x3
  - Added a 10 sec delay between running the code and pressing the camera. Will give the user 10 seconds to adjust the view on the knot

1. Run: python3 capture_and_save.py
2. User adjusts camera (10 seconds)
3. Ensure nothing shakes the camera !

* Added a grid on laptop screen to ensure knot was in the center of the view for future cropping

4. Code take photo, sorts as tight or loose, creates file name, adjusts 500x500 pixels and zooms
5. Final image is saved into “tight” and “loose” folders within the “squaredetect” folder

3/18: team took photos of a total of 38 Loose and 31 Tight

FINAL COUNT:

Loose: 115

Tight: 121

Conclusions/action items: The team now needs to train the model on these new images and decide if more images are needed to improve model outcome.

3/20: Raspberry Pi + HQ Camera setup diagram

Lucia Hockerman - Mar 20, 2026, 10:17 AM CDT

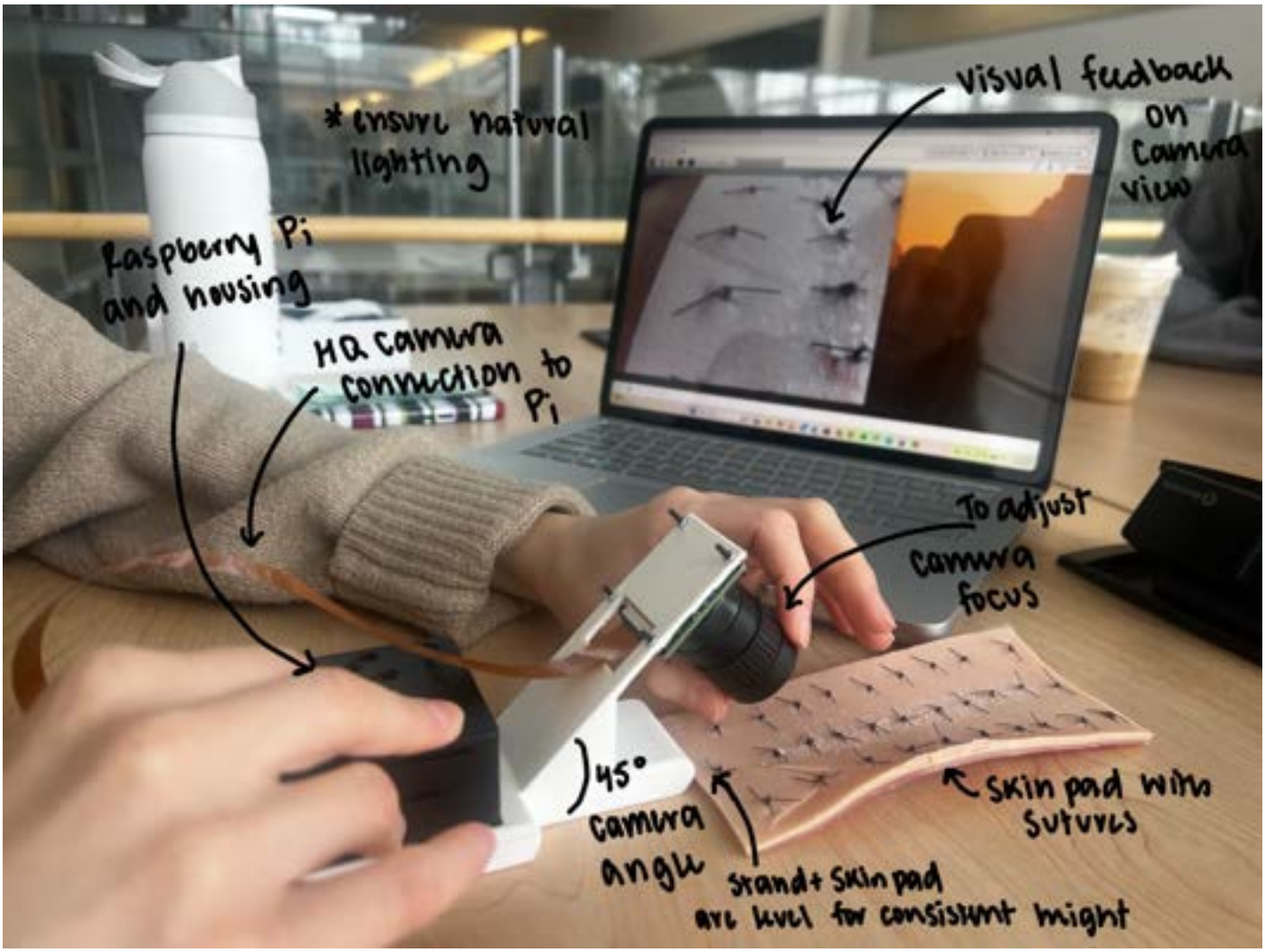
Title: Raspberry Pi + HQ Camera Setup Diagram

Date: 3/20

Content by: Lucy

Goals: For documentation purposes, include a diagram of our HQ camera system that was used for taking photos to train the model so these conditions can be replicated if needed.

Content:



3/20/2026: New Trainings Completed

Lucia Hockerman - Mar 20, 2026, 11:24 AM CDT

Title: UW Safety Trainings - NEW in 2026

Date: 03/20/2026

Content:

The new trainings completed Spring 2026 are highlighted in my official training record:

WISCONSIN
UNIVERSITY OF WISCONSIN-MADISON

This certifies that Lucia Hockerman has completed training for the following course(s):

Expand All | Collapse All

Course	Assignment	Completion	Expiration
2023-24 HIPAA Privacy & Security Training	HIPAA Attestation	2/21/2024	
2024-2025 HIPAA Privacy & Security Training	2024-2025 HIPAA Privacy & Security Training	12/6/2024	
2025-2026 HIPAA Privacy & Security Training	2025-2026 HIPAA Privacy & Security Training	2/3/2026	
Biosafety 102: Bloodborne Pathogens for Laboratory and Research	Biosafety 102: Bloodborne Pathogens Safety in Research Quiz 2024	2/9/2024	2/9/2025
Biosafety 105: Biosafety Cabinet Use	Biosafety 105: Biosafety Cabinet Use Quiz	2/13/2024	No Expiration
Biosafety 107: Centrifuge Safety	Biosafety 107: Centrifuge Safety Verification Quiz	2/13/2024	No Expiration
Biosafety Required Training	Biosafety Required Training Quiz 2023	11/1/2023	11/1/2028
Biosafety Required Training	Biosafety Required Training Quiz 2024-2025	1/20/2026	1/20/2029
Chemical Safety: Personal Protective Equipment	PPE Final Quiz	2/13/2024	2/13/2029
Chemical Safety: The OSHA Lab Standard	Final Quiz	11/1/2023	
Conflict of Interest Training	Conflict of Interest Quiz	2/5/2025	2/5/2029
EXPIRED Radiation Safety 107: Laser Safety 2023-2024	Laser Safety 2023-2024	2/11/2024	
Good Clinical Practice for Drug/Device Researchers	Good Clinical Practice	10/29/2024	10/29/2027
Responsible and Ethical Conduct of Research (RECR)	RCR Certification	2/10/2024	No Expiration
UW Human Subjects Protections Course	Basic/Refresher Course - Human Subjects Research	1/10/2024	1/10/2027

Data Last Imported: 03/20/2026 10:56 AM

10/26/2025: UW Safety Trainings

Lucia Hockerman - Oct 26, 2025, 10:06 PM CDT

Title: UW Safety Trainings

Date: 10/26/2025

Content:

This certifies that Lucia Hockerman has completed training for the following course(s):

Expand All Collapse All

Course	Assignment	Completion	Expiration
2023-24 HIPAA Privacy & Security Training	HIPAA Attestation	2/21/2024	
2024-2025 HIPAA Privacy & Security Training	2024-2025 HIPAA Privacy & Security Training	12/6/2024	
Biosafety 102: Bloodborne Pathogens for Laboratory and Research	Biosafety 102: Bloodborne Pathogens Safety in Research Quiz 2024	2/9/2024	2/9/2025
Biosafety 105: Biosafety Cabinet Use	Biosafety 105: Biosafety Cabinet Use Quiz	2/13/2024	No Expiration
Biosafety 107: Centrifuge Safety	Biosafety 107: Centrifuge Safety Verification Quiz	2/13/2024	No Expiration
Biosafety Required Training	Biosafety Required Training Quiz 2023	11/1/2023	11/1/2028
Chemical Safety: Personal Protective Equipment	PPE Final Quiz	2/13/2024	2/13/2029
Chemical Safety: The OSHA Lab Standard	Final Quiz	11/1/2023	
Conflict of Interest Training	Conflict of Interest Quiz	2/5/2025	2/5/2029
EXPIRED Radiation Safety 107: Laser Safety 2023-2024	Laser Safety 2023-2024	2/11/2024	
Good Clinical Practice for Drug/Device Researchers	Good Clinical Practice	10/29/2024	10/29/2027
Responsible and Ethical Conduct of Research (RECR)	RCR Certification	2/10/2024	No Expiration
UW Human Subjects Protections Course	Basic/Refresher Course - Human Subjects Research	1/10/2024	1/10/2027

Data Last Imported: 10/26/2025 09:53 PM

10/26/2025: COE Design Innovation Lab Trainings

Lucia Hockerman - Oct 26, 2025, 10:07 PM CDT

Title: COE Design Innovation Lab Trainings

Date: 10/26/2025

Content:

Lucia Hockerman

ID Number: 908428314 3

Eligibility: CoE Students

My Memberships

Membership Type	Start Date	Expiry Date	Renew	Card Info
Machining	Sun, Jan 1 2023	Permanent	Not Renewable	N/A
Laser Cutter	Sun, Jan 1 2023	Tue, Dec 30 3000	Not Renewable	N/A
Shop Tools	Sun, Jan 1 2023	Tue, Dec 30 3000	Not Renewable	N/A
Lab Orientation	Sun, Jan 1 2023	Tue, Dec 30 3000	Not Renewable	N/A
Shop Tools - Training Eligible	Sun, Jan 1 2023	Tue, Dec 30 3000	Not Renewable	N/A

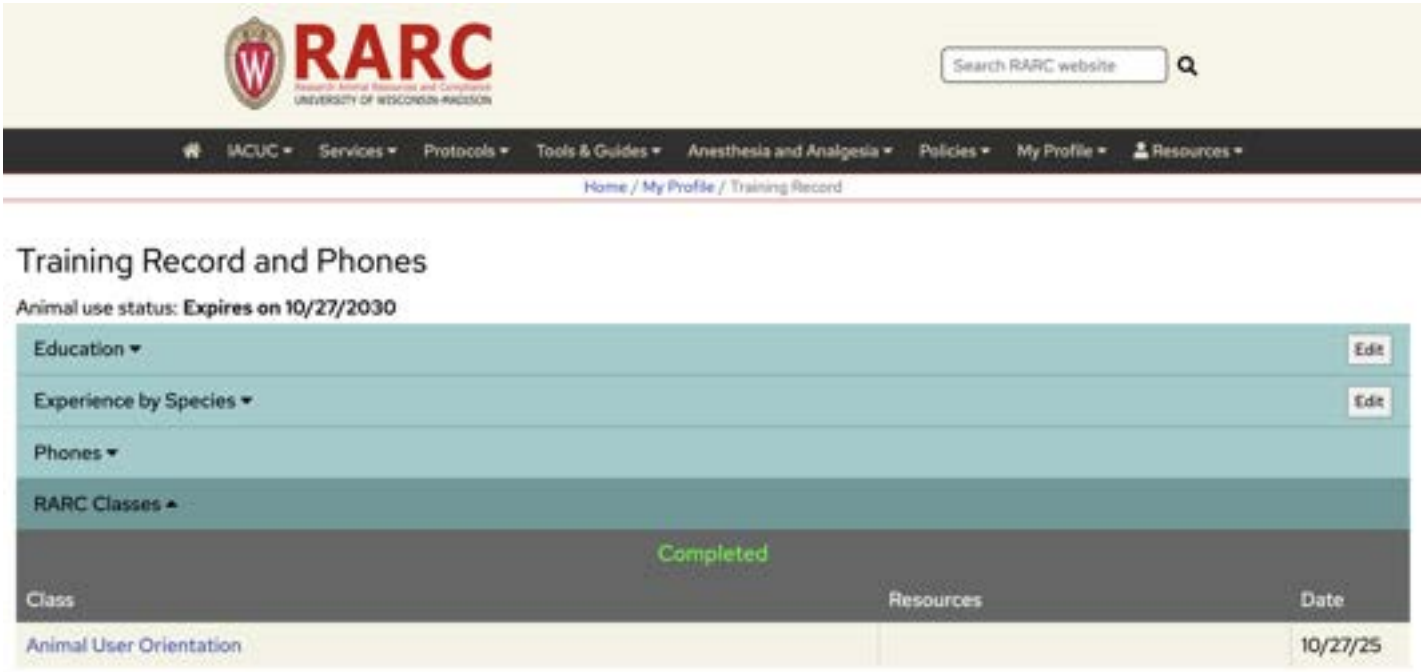
10/26/2025: Research Animal Resources and Compliance Training

Lucia Hockerman - Oct 26, 2025, 10:08 PM CDT

Title: Research Animal Resources and Compliance Training (New Fall 2026 Training)

Date: 10/26/2025

Content:



4/15: Title: Latency Testing of the Full Workflow

Lucia Hockerman - Apr 15, 2026, 11:24 AM CDT

Title: Latency Testing of the Full Workflow

Date: April 15, 2026

Content by: Lucy

Present: Maddie, Kate, Sadie, Presley

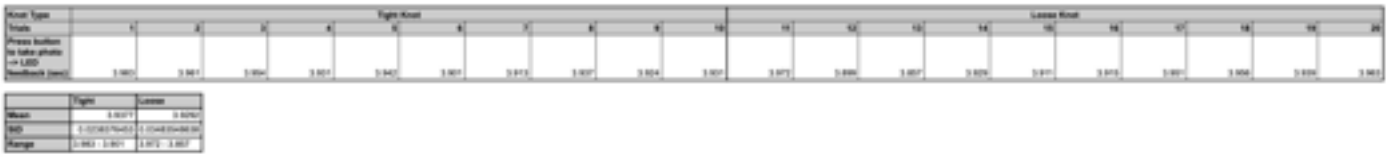
Goals: Run latency testing to assess the efficiency of the full user-interactive workflow.

Content:

Maddie and Kate added time stamps to the time code. The time stamps that will be collected in this testing are (1) and (2).
The team set up the Pi and chose one knot on the skin pad to run all analysis on.

We excluded the time in the total workflow where the user initially presses the button and presses the button a second time to take the photo. Between these two button presses, the user is given a preview and as much time as needed to adjust the knot within the camera view. This is excluded because it is entirely user-dependent and does not asses the efficiency of the model.

Spreadsheet with latency results, including mean and standard deviation:



Conclusions/action items: Latency test results were positive with under the desired maximum running time of 5 seconds. For tight knots the mean is 3.937 seconds and for loose knots the mean is 3.929 seconds.

1/26/26 AI Camera RasberryPi

Kate Hiller - Jan 26, 2026, 1:14 PM CST

Title: AI Camera ResberryPi

Date: 1/26/26

Content by: Kate Hiller

Present:

Goals: To find a camera for the training system to process the machine learning model

Content:

The team was advised to use a Raspberry Pi camera

link: https://www.raspberrypi.com/documentation/accessories/ai-camera.html

- uses PyTorch and TensorFlow

IMX500 does the AI inside the camera instead of needing a computer to process information. The AI camera just sends the tensor output to the computer

- does object detection - unsure if it would be able to do tight vs. loose

For the model development, if the team were to move forward with this camera, here are the steps outlined on the page:

"To deploy a new neural network model to the Raspberry Pi AI Camera, complete the following steps:

1. Provide a floating-point neural network model (PyTorch or TensorFlow).

2. Run the model through Edge-MDT (Edge AI Model Development Toolkit).

a. ***Quantise*** *and compress the model so that it can run using the resources available on the IMX500 camera module.*

b. ***Convert*** *the compressed model to IMX500 format.*

3. Package the model into a firmware file that can be loaded at runtime onto the camera.

The first two steps will normally be performed on a more powerful computer such as a desktop or server. You must run the final packaging step on a Raspberry Pi."

We would upload TensorFlow or Roboflow to the camera

Conclusions/action items:

This is a potential camera to order for us to use, will continue to look into other options.

1/28/26 K-Fold Cross Validation

Kate Hiller - Jan 28, 2026, 5:27 PM CST

Title: K-Fold Cross Validation

Date: 1/28/26

Content by: Kate Hiller

Present: n/a

Goals: To understand what K-fold is, as it was mentioned to us last semester at the poster presentation.

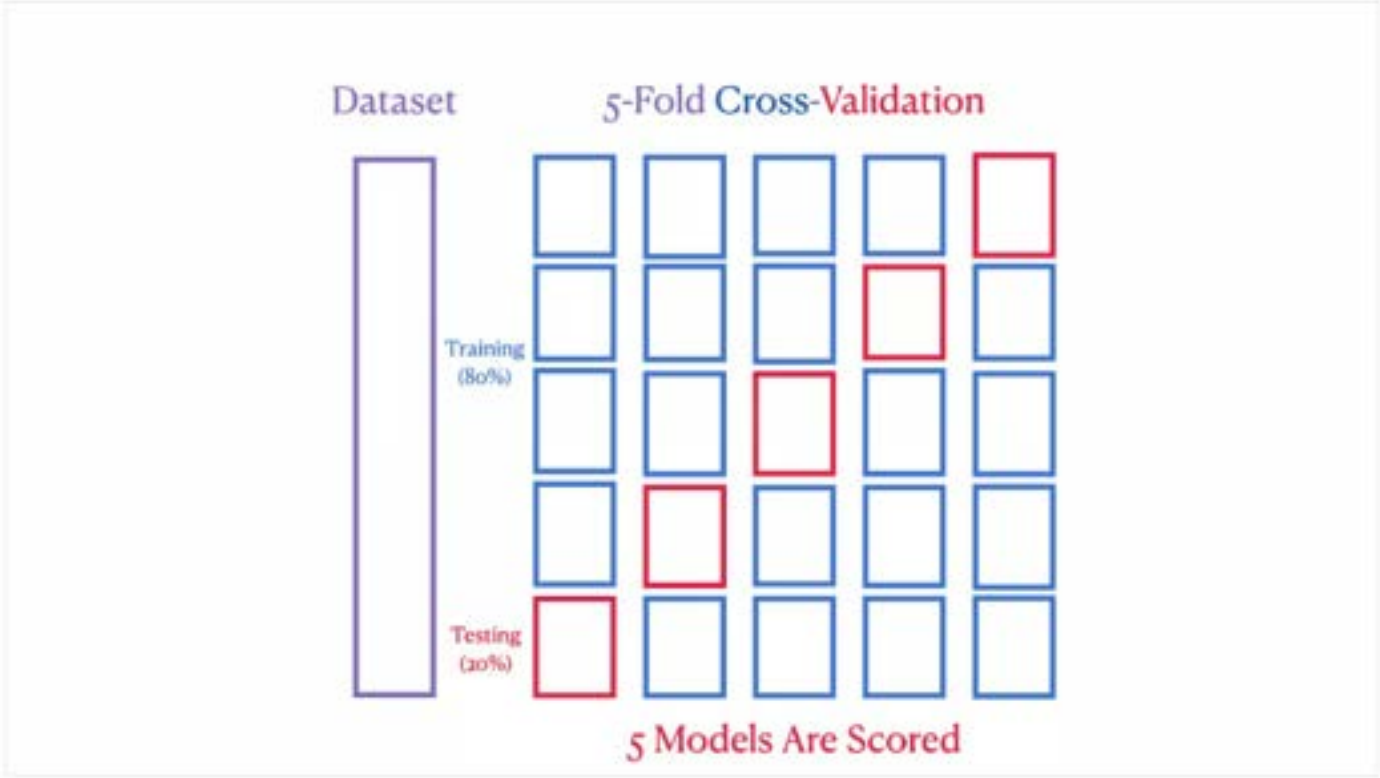
Link: https://www.datacamp.com/tutorial/k-fold-cross-validation

Content:

- used to evaluate the performance of machine learning models

- ensures that the model generalizes well to unseen data by using different portions of the dataset for training and testing in multiple iterations

- k-fold CV portions the dataset into multiple subsets to iteratively train and test the model



5-Fold Cross-Validation Scheme. Image by Author

k-fold = 5-fold the dataset is split into 5 parts, each part serving as a testing set in one of the five iterations, ensuring each segment is used for both training and testing.

- by using multiple training and testing cycles it minimizes the risk of over fitting to a particular data split.

- this ensures that every data point is used for both training and validation, which results in a more comprehensive evaluation of the models performance.

-U can do this with python code - code for running the k-fold is at this link

Conclusions/action items:

K-fold CV provides a more robust and reliable performance estimate becuase it reduces the impact of data variability. Next steps is implimenting this for our model

1/31/26 Research Into Implementing the ML Model Into Raspberry Pi

Kate Hiller - Jan 31, 2026, 12:48 PM CST

Title: Research Into Implementing the ML Model Into Raspberry Pi

Date: 1/31/26

Content by: Kate Hiller

Present: n/a

Goals: To understand/make a plan on how to put the ML model on a Raspberry Pi microcontroller

link: https://www.raspberrypi.com/documentation/computers/camera_software.html#use-a-usb-webcam

Content:

Raspberry Pi model b (4 GB)

- first train the model on the computer

- optimize the model

- then need to compact the model into a smaller file size depending on the neural network

Camera and Raspberry Pi

USB Webcam possible

- All Raspberry Pi cams can record high-resolution photographs and HD 1080p or better with the software tools

cameras:

12-megapixel AI cam using Sony IMX500 imaging sensor to provide low latency, high performance, AI capabilities to any camera application - $75

12.3 MP Sony IMX500 Intelligent Vision Sensor with a powerful neural network accelerator

Framerates:

2×2 binned: 2028×1520 10-bit 30fps

Full resolution: 4056×3040 10-bit 10fps

7.857 mm sensor size

1.55 µm × 1.55 µm pixel size

78.3 (±3) degree FoV with manual/mechanical adjustable focus

F1.79 focal ratio

25 × 24 × 11.9 mm module dimensions

Integrated RP2040 for neural network firmware management

Works with all Raspberry Pi models, using our standard camera connector cable

Video mode: 1080p30

12-megapixel High Quality Camera with CS and M12 mount variants for use with external lenses -$63.75

Sony IMX477R stacked, back-illuminated sensor, 12.3 megapixels, 7.9 mm sensor diagonal, 1.55 µm × 1.55 µm pixel size
Output: RAW12/10/8
Back focus length of lens: 2.6mm–11.8mm (M12 Mount variant), 12.5mm–22.4mm (CS Mount variant)
Lens sensor format: 1/2.3" (7.9mm) or larger
IR cut filter: Integrated
Ribbon cable length: 200 mm
Tripod mount: 1/4"-20
Video modes: 1080p50, 720p120

Camera Model 3 - cheapest option $25

rpicam-jpeg: helps you capture images on Raspberry Pi devices

timeout options to alter the display time of the preview window. width and height to change the resolution of the saved images.

ex: *rpicam-jpeg --output test.jpg --timeout 2000 --width 640 --height 480*

ex: rpicam-still --output test.jpg $saves full resolution jpeg and saves it to the file name test.jpeg

use encoding to change to png

- can tweak camera behavior using *libcamera* library

- has an option for multiple cameras

- lots of info about the coding for the camera is on the link above

Conclusions/action items:

Bring this information to the team meeting on Monday to decide on a camera and microcontroller to order. Work on the prelim presentation using Dr. Block's advice. We will focus on K-fold cross-validation, putting the machine learning model on the microprocessor, and then on the integration of the camera.

2/9/2026 Raspberry Pi for Beginners

Kate Hiller - Feb 12, 2026, 9:50 AM CST

Title: Raspberry Pi for Beginners

Date: 2/9/26

Content by: Kate Hiller

Present: N/a

Goals: To learn the basics of the raspberry pi

Citation:

Raspberry Pi, “Raspberry Pi Documentation - Getting Started,” *www.raspberrypi.com*. https://www.raspberrypi.com/documentation/computers/getting-started.html

Content:

Need to start Raspberry Pi

- power supply (27W for raspberry pi 5)

- boot media (microSD card)

You can set up the raspberry pi "headless" computer accessible only over the network, or you can set it up as an interactive computer with a desktop

- need to install an operating system

- We will want to use our Raspberry Pi directly, so we will need a display, HDMI cable, a keyboard, and mouse
- Raspberry Pi models lack onboard storage so we have to supply it. Boot the Pi from an operating system image installed on a microSD card, but USB storage, network storage, and storage connected via PCIe HAT are also available.

- The Raspberry Pi automatically boots form microSD slot when there is a card in the slot

To install an OS on a storage device for the raspberry pi:

- computer you can use to image the storage device into a boot device

- ability to plug storage device into the computer

- there is also an option to install over the internet, but the team will have access to monitors.

**Install using Raspberry Pi Imager, how to is linked here: https://www.raspberrypi.com/documentation/computers/getting-started.html#raspberry-pi-imager

Conclusions/action items:

Setting up a Raspberry Pi requires a few components, and very minimal components depending on the method of setting up the operating system. We have ordered an HDMI to microHDMI cord to connect the Pi to a monitor, and will

need a mouse, keyboard, and monitor to program. Setting up the OS should be a relatively simple process. We need to order the hardware to arrive and then install the OS system after it arrives. We will need to figure out where we are buying our hardware from.

2/10/26 Raspberry Pi at the MakerSpace

Kate Hiller - Feb 12, 2026, 10:03 AM CST

Title: Raspberry Pi at the MakerSpace

Date: 2/10./26

Content by: Kate Hiller

Present: Kate and Presley

Goals: I found online that the MakerSpace has some Raspberry Pis and the opportunity to check out a board. Presley and I went to investigate what they had.

Content:

- The maker space has Raspberry Pi 4 Model B for $40 and Raspberry Pi 5 8GB (which we are looking for) for $50.

- They have microSD cards for sale as well and power supplies. But the power supplies they have are not the 27W that we are looking for

- We found how and where you can check out the Raspberry Pi boards, but this is not helpful for the project because we want to permanently install our ML model on the microprocessor

- We found that there is a free electronics section where people leave their extra supplies for others to use. We took two fans that could replace the fan cooling system we are going to order. They also have various buttons that we could use for the button that activates the camera to snap a picture.

Conclusions/action items:

We now know of the option to get a Raspberry Pi 5 right away at the makerspace, and this would be ideal because we will not have to wait for it to arrive in the mail. However, they dont have all the components we need, and it may be more cost-effective to order them as a bundle. We can also look to the makerspace for help with programming the Raspberry Pi as they have "office hours" to help people with electronics. Next, I want to figure out the circuit diagram for the entire device.

2/14/26 Circuit Schematic

Kate Hiller - Feb 14, 2026, 1:23 PM CST

Title: Circuit Schematic

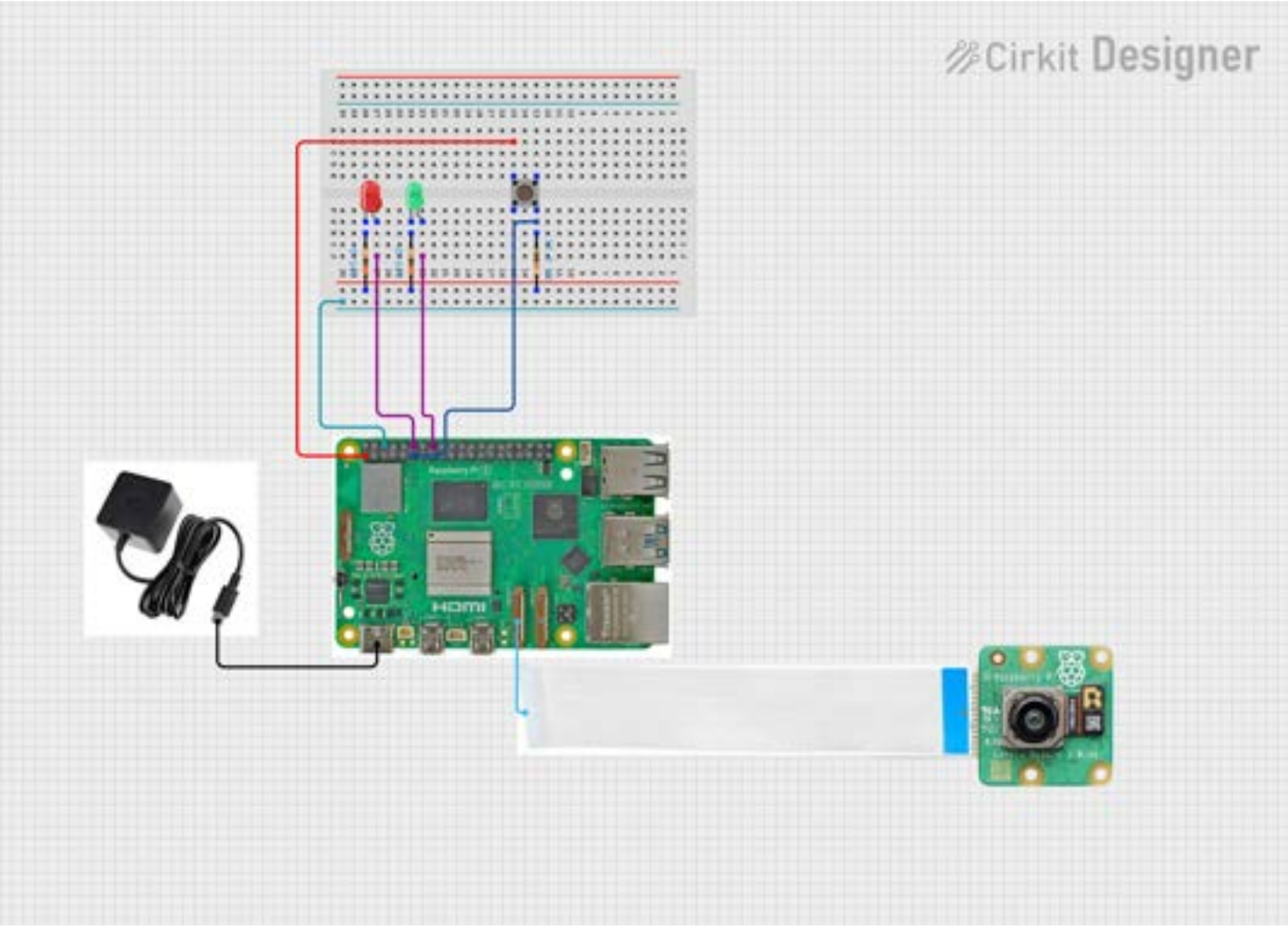
Date: 2/14/26

Content by: Kate Hiller

Present: n/a

Goals: Develop a hardware schematic for the Raspberry Pi 5 and all peripheral components

Content:



Circuit schematic of the Raspberry Pi 5 microcontroller.

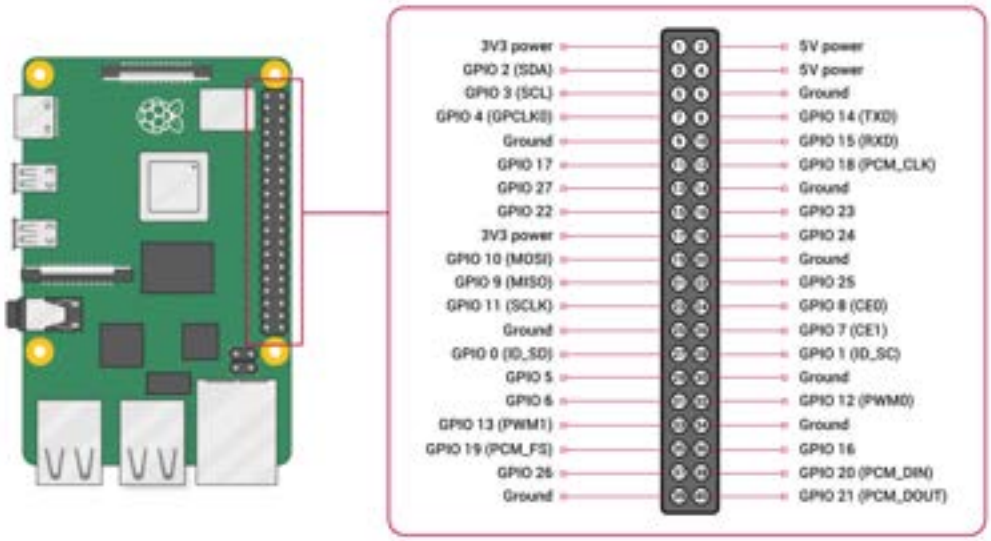


Image of the layout of the pins on the Raspberry Pi 5.

- Button is powered by 3.3V, potentially 5V

- The button has 10 kOhm resistor to draw current out of the button

- Green LED is connected analog Pin 23

- Red LED is conncted to analog Pin 18

- Button is connected to analog pin 17 and ground

- blue terminal on the breadboard is the ground connection

Conclusions/action items:

This schematic should further be updated to use HQ camera - need to take a picture of it when it arrives, and I should be able to upload it. I will potentially make a diagram with its connection to the monitor, keyboard, and mouse.

2/16/26 Updated Circuit Schematic

Kate Hiller - Feb 16, 2026, 1:57 PM CST

Title: Circuit Schematic

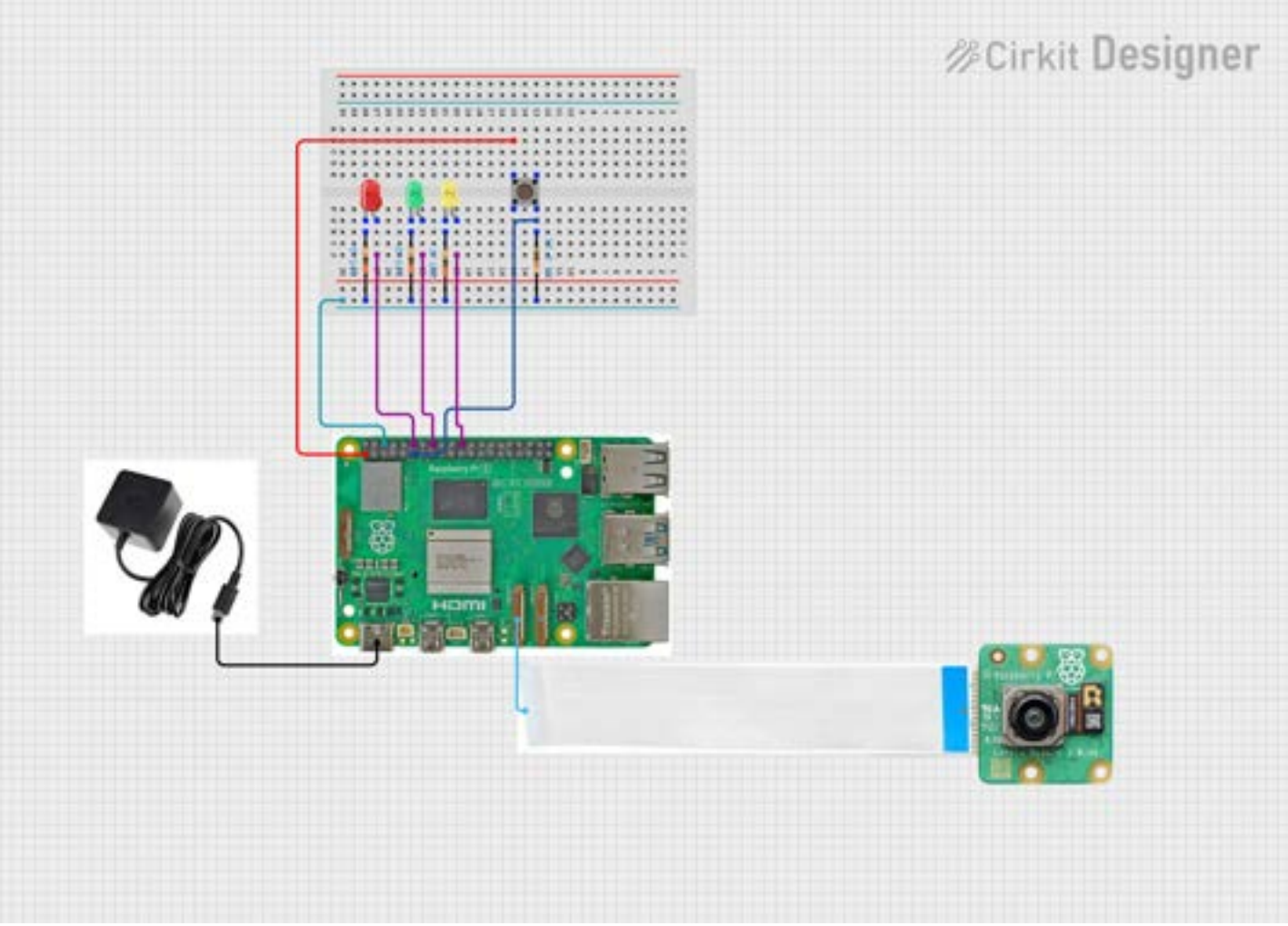
Date: 2/16/26

Content by: Kate Hiller

Present: n/a

Goals: Develop a hardware schematic for the Raspberry Pi 5 and all peripheral components, adding a yellow LED and fixing incorrect circuitry.

Content:



Circuit schematic of the Raspberry Pi 5 microcontroller.

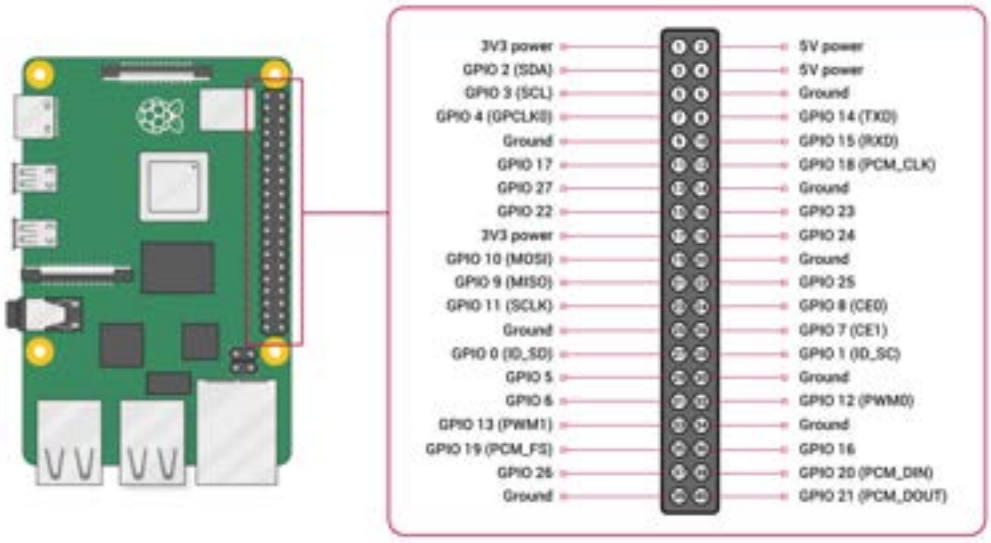


Image of the layout of the pins on the Raspberry Pi 5.

- Button is powered by 3.3V, potentially 5V

- The button has 10 kOhm resistor to draw current out of the button

- Green LED is connected digital Pin 23

- Red LED is conncted to digital Pin 18

- Yellow LED is connected to digital pin 25

- Button is connected to digital pin 17 and ground

- blue terminal on the breadboard is the ground connection

Conclusions/action items:

This schematic should further be updated to use HQ camera - need to take a picture of it when it arrives, and I should be able to upload it. I will potentially make a diagram with its connection to the monitor, keyboard, and mouse.

2/25/26 Built Circuit

Kate Hiller - Feb 26, 2026, 10:05 AM CST

Title: Built Circuit

Date: 2/26/26

Content by: Kate Hiller

Present: n/a

Goals: To build the button/LED circuit based on the schematic I developed.

Content:

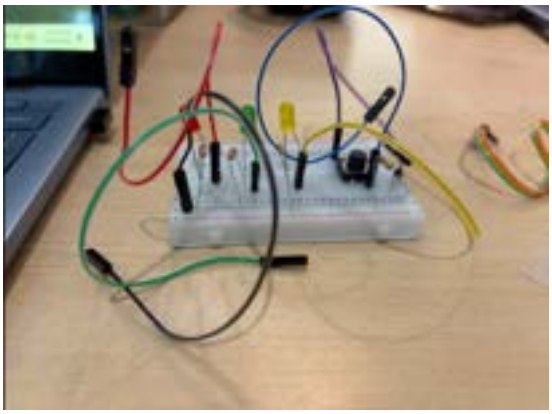


Figure 1: side view of circuit

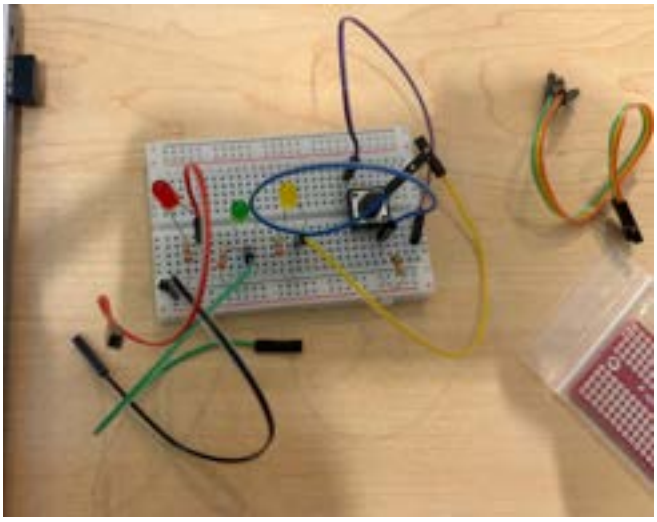


Figure 2: top view of circuit

Conclusions/action items:

We will connnect this circuit to the raspberry PI once we upload the machine learning model onto the PI. The team is currently working on trying to set up the PI. We have got it hooked up to a monitor with peripherals, but are working on

figuring out how to download a later version of Python onto the Pi.

3/4/26 Raspberry Pi Implementation

Kate Hiller - Mar 04, 2026, 11:12 AM CST

Title: Raspberry Pi Implementation

Date: 3/4/26

Content by: Kate

Present: N/a

Goals: To provide an update on the status of the Pi

Content:

Team met with Norbu in MakerSpace on 2/29 and Dani in the MakerSpace on 3/2.

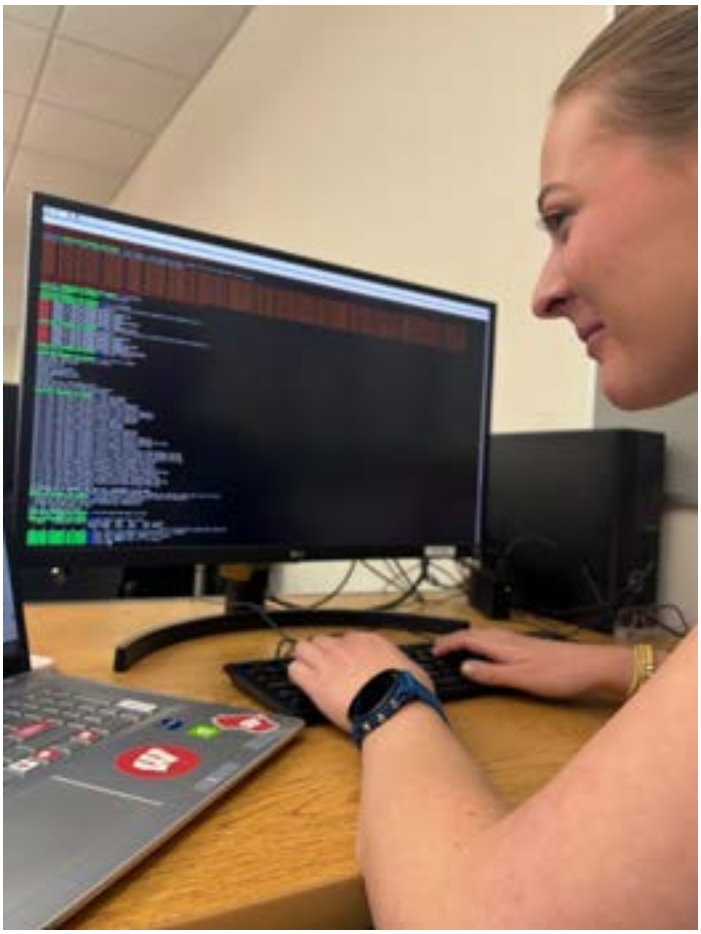
- We learned that it might not be plausible to upload the ML model on the Pi, however, I am skeptical of this because I have seen it online

- Dani recommended using Claude AI to figure out our errors with the code. You can load Claude into the terminal and it can attempt to fix the errors and run the code. However you need a premium subscription which we would have to pay for and I am unsure if you have to continuously pay for the Claude to be inside the Pi or if it is a one-time fee.

Current Accomplishments:

- we loaded OS onto SIM card

- figured out how to connect Pi to peripherals (mouse, keyboard, monitor) to run the Pi and open a terminal. We made a virtual environment called "knotnewyay"



- at this point we couldn't figure out how to download an older version of Python to match the Python version Maddie has been using, then we went to the MakerSpace.

- 2/28, MakerSpace helped us set up Raspberry Pi Connect and how to get the Pi to connect to the UW Madision wifi.

- We figured out how to install old versions of Python, and then were able to install Torch

- Put Torch is having errors with weights, and no one at the MakerSpace can figure out the problem and said the Pi wouldn't work

- They recommended just using the camera aspect for now, unless we can find a way to implement the model on the Pi. Dani was surprised we were even able to get a functioning image classification model.

Code:

```
python3 -m venv knotvenv
source knotvenv/bin/activate
python --version #checking which version is currently downloaded
pyenv install 3.9 #Installing pyenv: copied code from duckduckgo → installed version 3.9
```

Conclusions/action items:

I am struggling with figuring out how the Pi pulls and stores information form its files. We got a ML .py file on the Pi, but we can't get the Pi to access the file. I think we should try the Claude thing. Maddie made a website that can run the model and display the output of the model. Right now, we are pivoting and trying to set up the camera.

3/13/26 Meeting With CS Student

Kate Hiller - Mar 18, 2026, 7:46 PM CDT

Title: Meeting With CS Student

Date: 3/18/26

Content by: Kate

Present: Kate, Presley, Maddie

Goals: To get help running the model on the Pi

Content:

Notes from the meeting:

- Make vi script.py with model loading code (same code as jupyter notebook)
  - Vi script capture_and_detect
- Add print statement at the end
- :wq to leave script file
  - Use nano?????
- WORKED

Make python script for FULL

1. Pip install picamera2
   a. If it fails: Sudo apt-get install -y libcap-dev
2. Cd to the folder square detect
3. python3 capture_and_predict.py

SSH Into Pi

1. Sudo sys… enable SSH (within pi)
2. On personal computer: ssh knotorious-five@140

Conclusions/action items:

We were able to get the model running on the Pi, have a camera snap a picture, and send it to the model. The student also helped us with saving the picture and the output of the model into a csv file. We have made significant progress, as we were only able to open jupyter notebook before meeting with the student.

3/18/26 Creating new data set

Kate Hiller - Mar 18, 2026, 7:24 PM CDT

Title: Creating New Dataset

Date: 3/18/26

Content by: Kate

Present: Kate, Lucy, Maddie, Sadie Presley

Goals: Create new dataset for training model

Content:

- The team met to take pictures with the new Raspberry Pi set up. The camera stand was printed at the makerspace and the camera was screwed into the stand. The Pi was programmed to have a delay, then snap a picture, and sort the picture to tight or loose.

- We were able to figure out how to alter the pixel size for future reference

- In preparation, Sadie, Lucy, and Kate tied more suture knots with materials ordered by the client. The new knots are more uniform and evenly spaced on the skin pad to improve our dataset. Additionally, the photos will be taken by the Pi camera in order to establish consistency through training and implementation of the model.

Conclusions/action items:

The pictures taken will be moved off the Pi with a hard drive and used to train a new binary model on a computer. That trained model will then be uploaded to the Pi for implementation. Maddie wrote code for the LED and button component. After the creation of the dataset, we will upload the button and LED code and run the whole model.

4/8/2026 Implementation of the Button and LED

Kate Hiller - Apr 09, 2026, 11:01 AM CDT

Title: Implementation of the Button and LED

Date: 4/8/2026

Content by: Kate Hiller

Present: Entire Team

Goals: To get the buttons and LEDs working with the Pi

Content:

We hooked up the button and LEDs to the Raspberry Pi

Conclusions/action items:

We are still trying to get the GPIU pins software downloaded onto the Pi. Maddie has written code for the buttons and LEDs that is ready to be implemented. On friday the team is meeting to figure this out to finish the entire workflow of the prototype.

4/10/26 Entire Workflow Implementation

Kate Hiller - Apr 16, 2026, 11:51 AM CDT

Title: Entire Workflow Implementation

Date: 4/10/2026

Content by: Kate

Present: Kate, Maddie, Sadie, Lucy, Presley

Goals: To get the entire workflow with the button working.

Content:

The team met after the hardware of the LEDs and buttons was fabricated. The team figured out how the GPIO pins and the extra software that needed to be installed for them to work, and updated the code for pin numbers.

Workflow:

- Press the button to initiate the software

- A preview screen pops up and allows the user to position the knot in the center of the screen. We wrote code to add a red dot on the screen as a target for positioning.

-Press the button once the knot is centered. This prompts the software to capture the image and automatically crop it to a 500x500 image.

- The image is sent through the ML model and the yellow LED is turned off

- Once the model has made a decision, an LED is illuminated based on the decision. Green lights up for tight and red lights up for loose.

- The model logs the image used and the decision with a confidence score

Hardware validation was completed today, ensuring all the hardware components work and work in the workflow.

Conclusions/action items:

Next, we will meet with the team to do latency testing of the workflow. We also need to update the ML model on the Pi.

4/17/2026 Manual ML Testing

Kate Hiller - Apr 20, 2026, 11:15 AM CDT

Title: Manual ML Testing

Date: 4/17/2026

Content by: Kate Hiller

Present: Whole Team

Goals: To conduct testing to collect accuracy data for the model

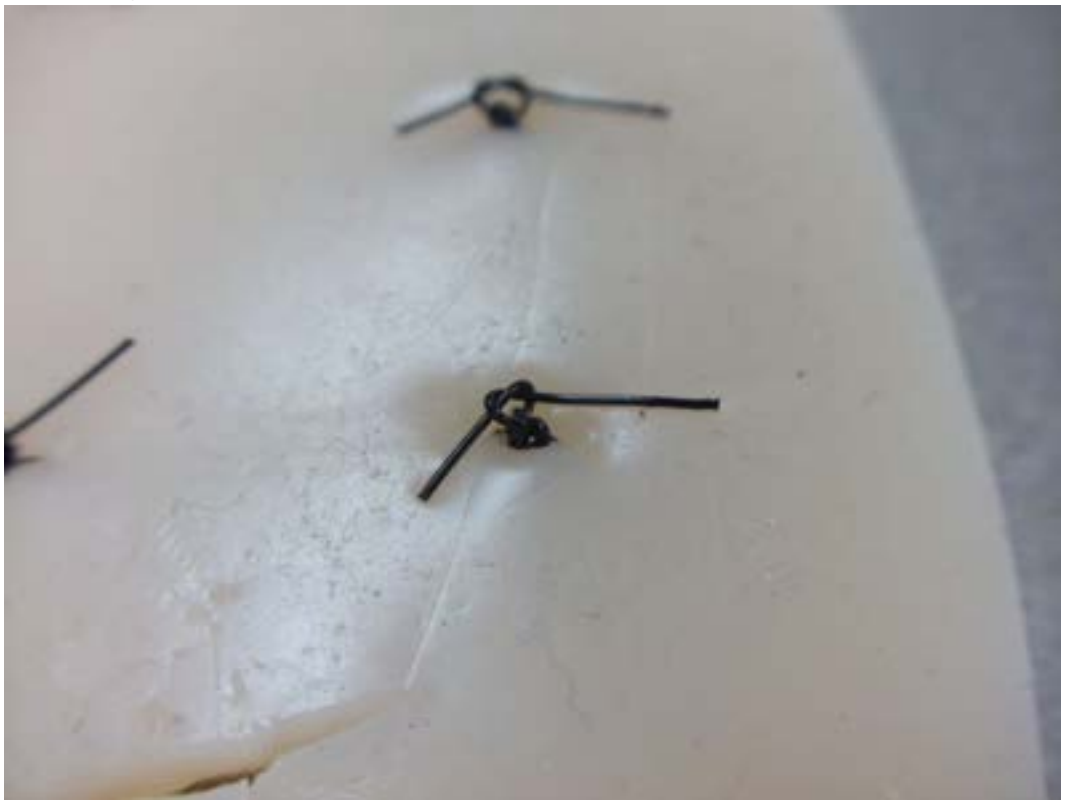
Content:

MODEL_PATH = "/home/knotorious-five/squaredetect/top_and_side_resnet_full.pth"

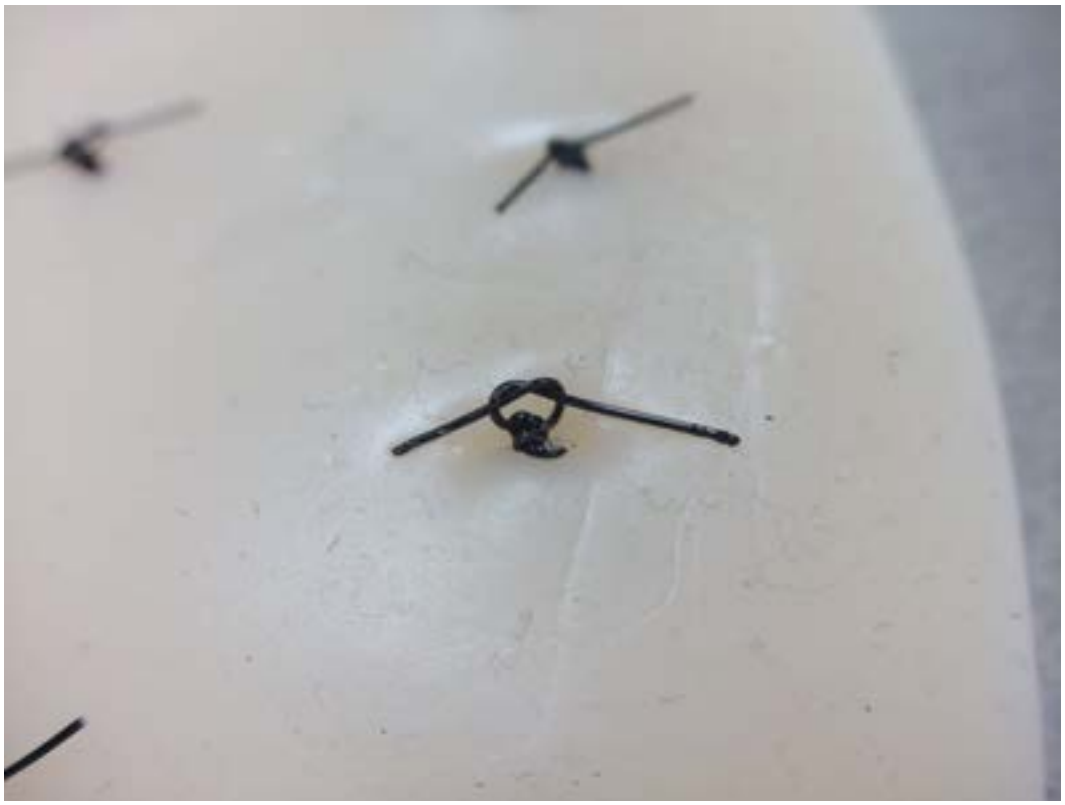
Loose 1: tight .6521

Loose 2 - 8397 tight (loose)

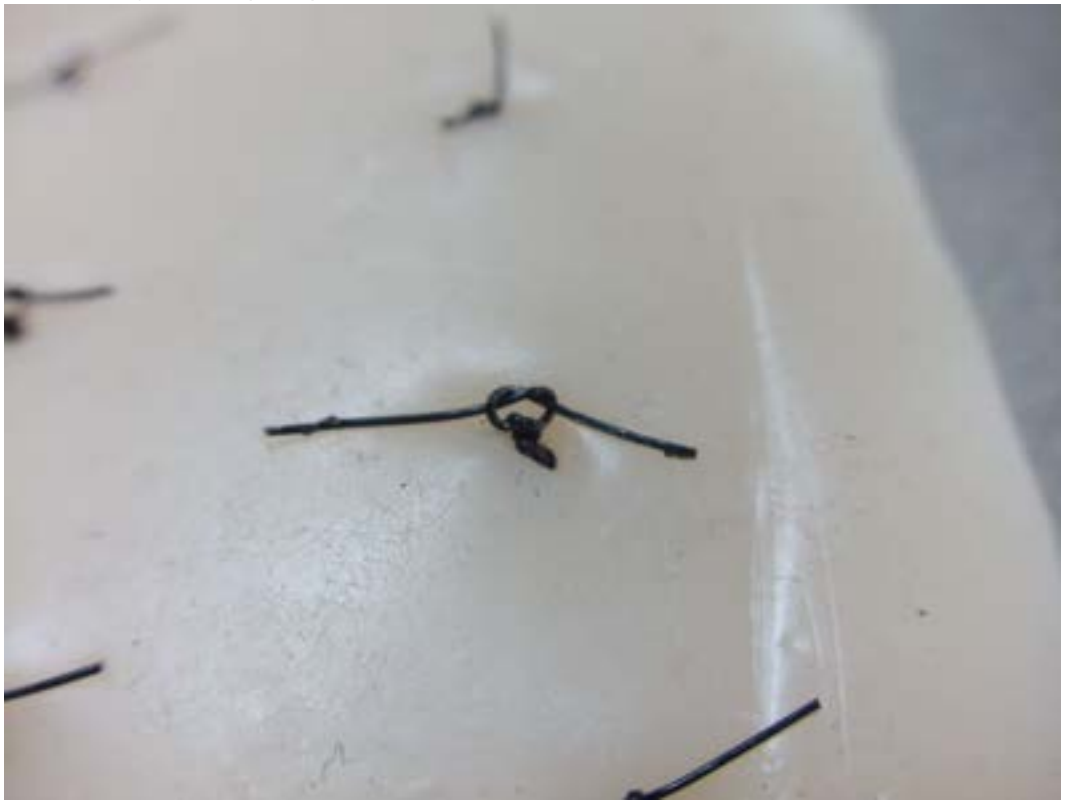
Loose 3 - tight 86.49



Loose 4 - tight 58.19

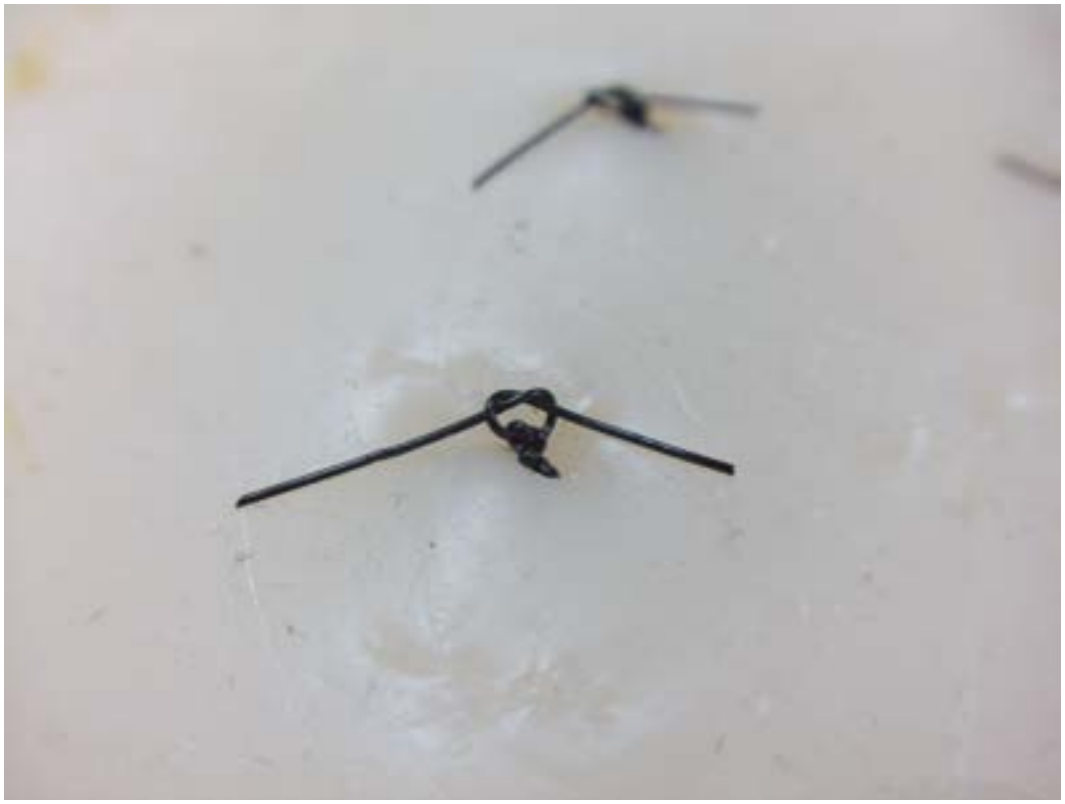


Loose 5 - Tight 78.66 (loose)



Loose 6 -84.77 tight (loose)

Loose 7 - tight 93.63



Loose 8- 89.26

Loose 9 - Tight 94.00%

Loose 10 - Tight 84.32

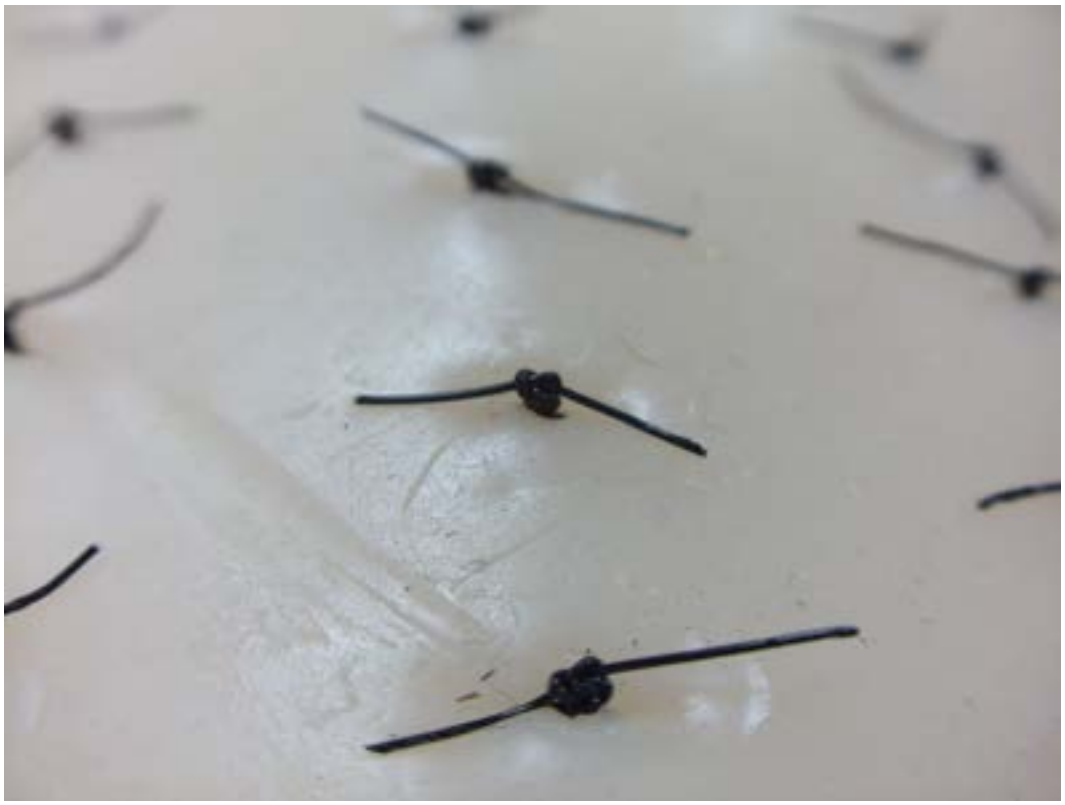
Interpreted Loose values (%)

65.21, 83.97, 86.49, 58.19, 78.66, 84.77, 93.63, 89.26, 94.00, 84.32

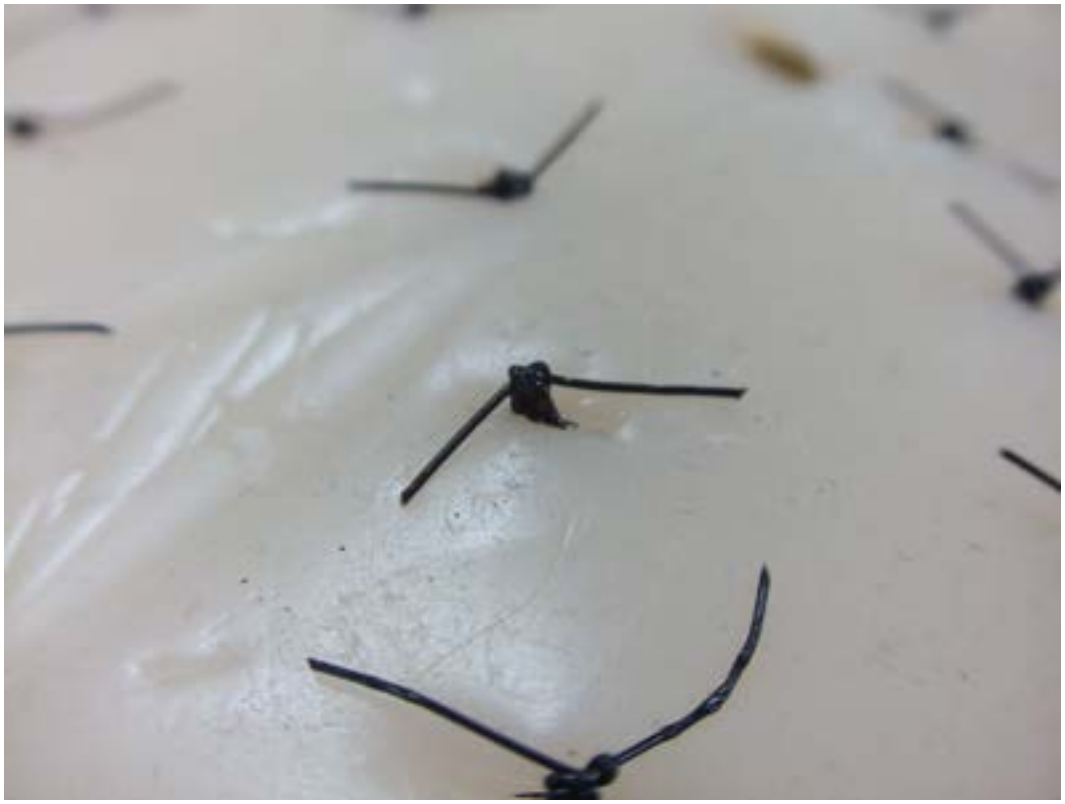
Predicted 70% (7/10) correct based on the 85% threshold

Average (Loose) = 81.85%
Decimal = 0.8185

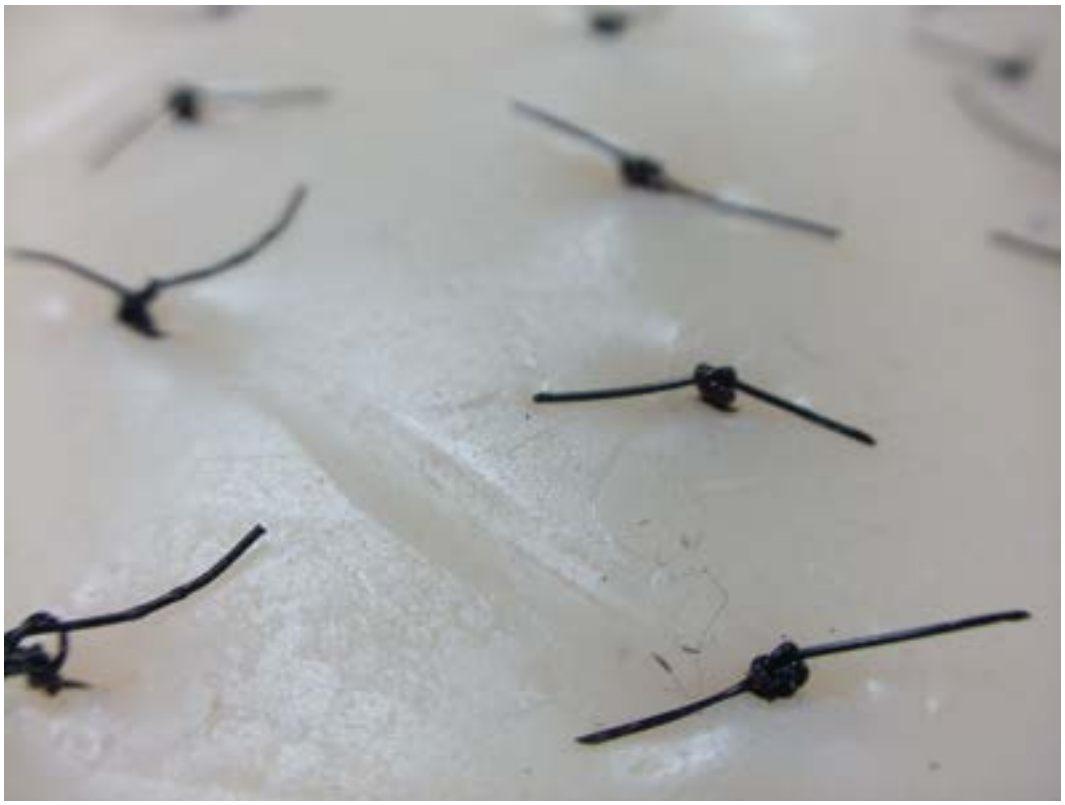
Tight 1 - 9498 tight



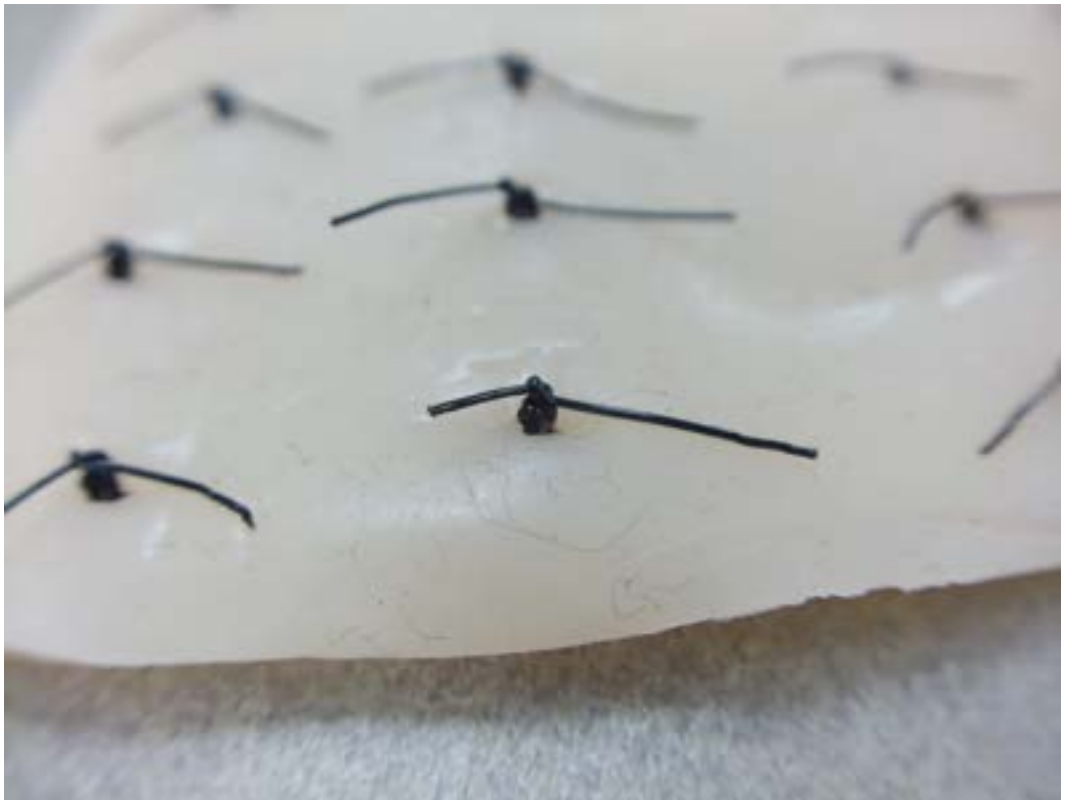
Tight 2 - 95.80 tight



Tight 3 - 97.16 tight

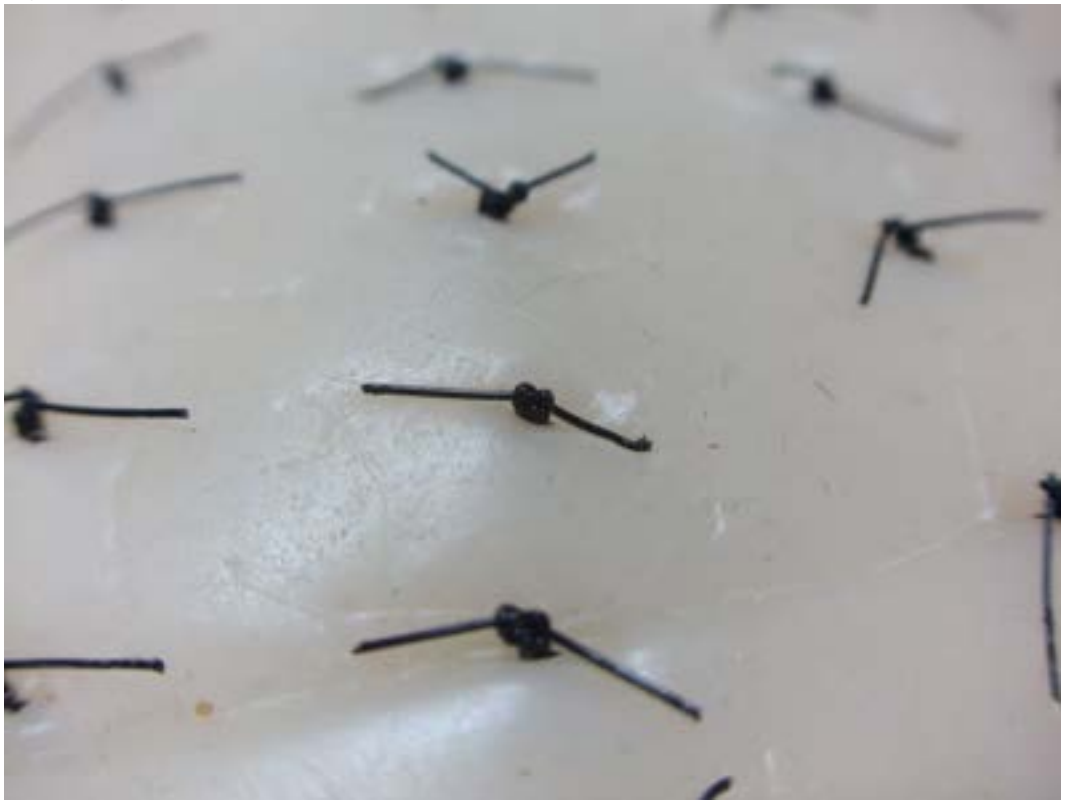


Tight 4 - tight 97.17

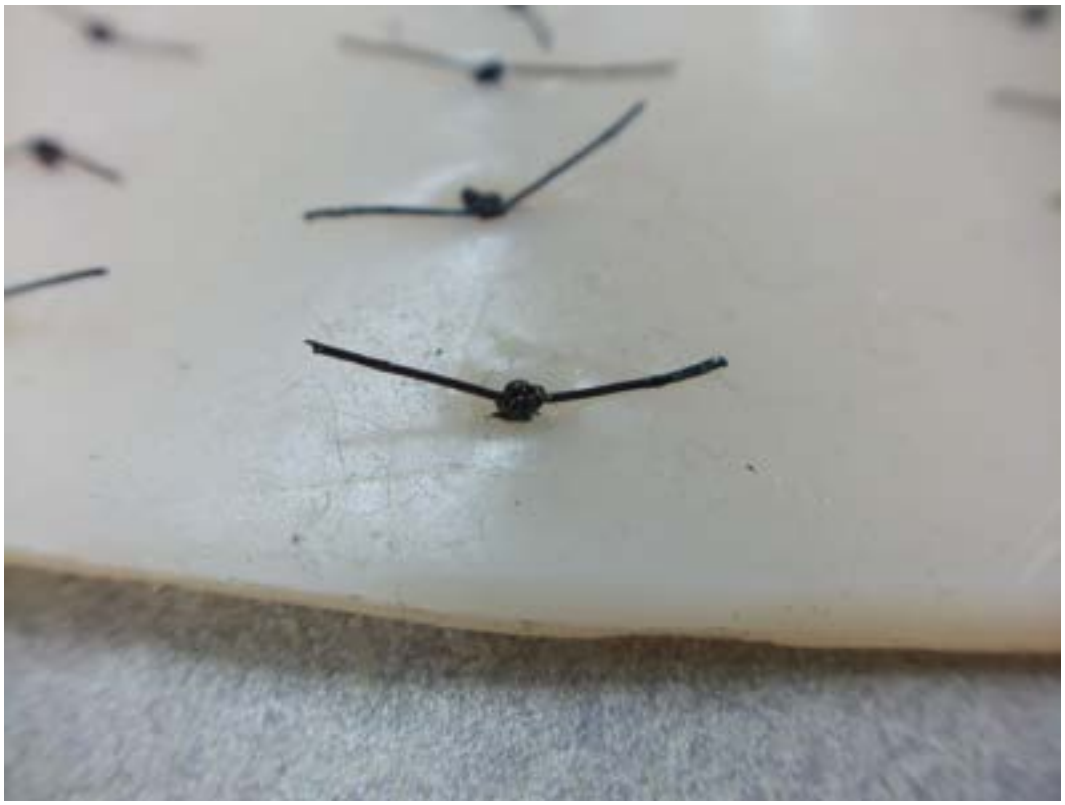


Tight 5 - tight 96.85

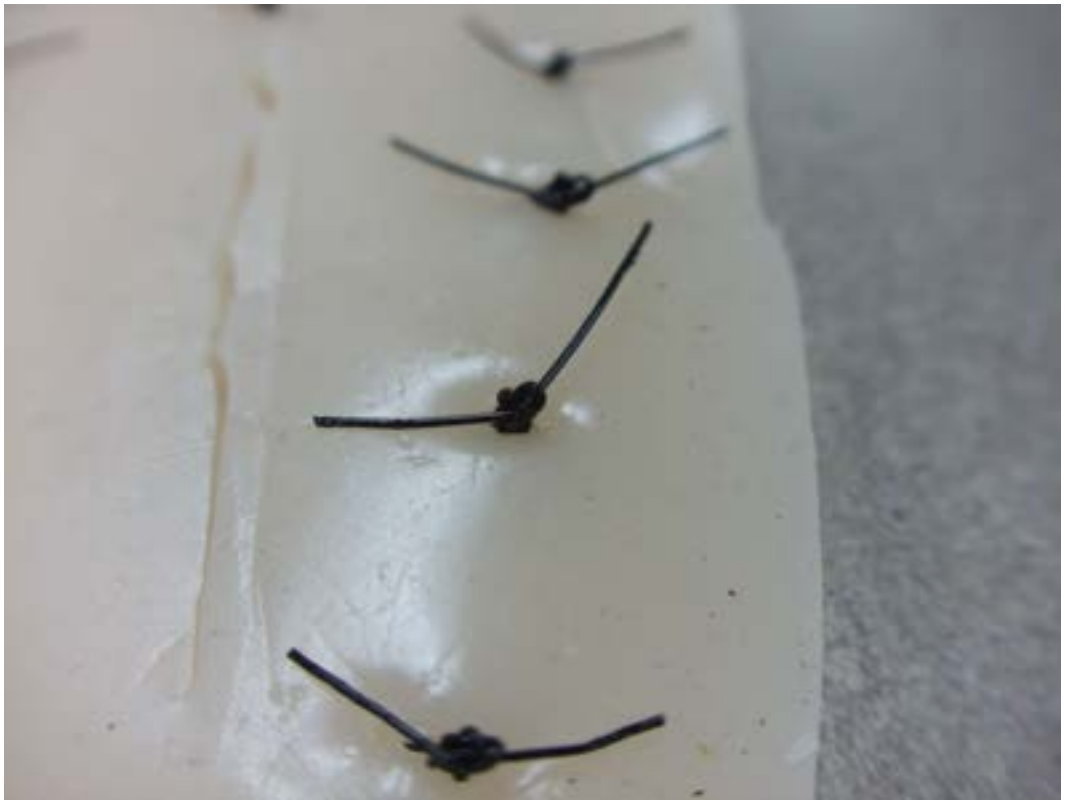
TIght 6 - tight 87.01



Tight 7 - tight 93.30

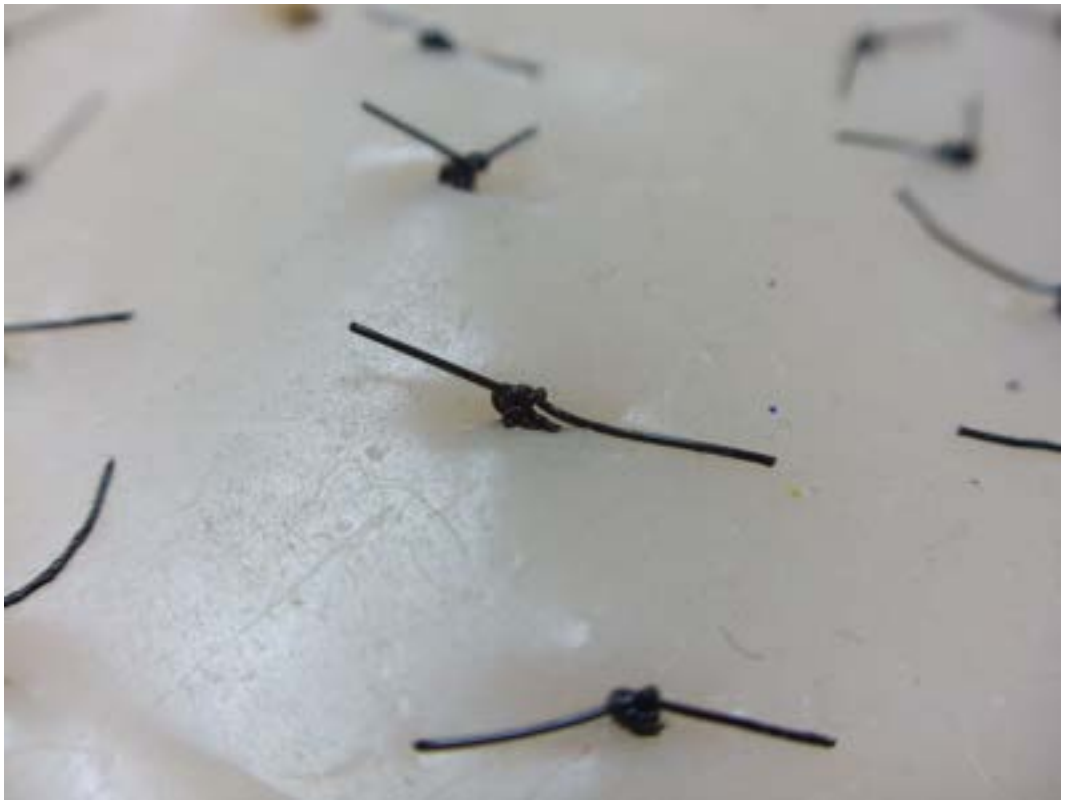


Tight 8 - 92.17 tight



Tight 9 - tight 92.62

Tight 10 - 98.23 tight



Loose Knots			Tight Knots		
Confidence (%)	**Prediction**	**Output**	**Confidence (%)**	**Prediction**	**Output**
65.21	Tight	Loose	94.98	Tight	Tight

83.97	Tight	Loose	95.80	Tight	Tight
86.49	Tight	Tight	97.16	Tight	Tight
58.19	Tight	Loose	97.17	Tight	Tight
78.66	Tight	Loose	96.85	Tight	Tight
84.77	Tight	Loose	87.01	Tight	Tight
93.63	Tight	Tight	93.30	Tight	Tight
89.26	Tight	Tight	92.17	Tight	Tight
94.00	Tight	Tight	92.62	Tight	Tight
84.32	Tight	Loose	98.23	Tight	Tight

***Values predicted under an 85% threshold is predicted loose

Tight values (%)

94.98, 95.80, 97.16, 97.17, 96.85, 87.01, 93.30, 92.17, 92.62, 98.23

Average (Tight) = 94.13%
Decimal = 0.9413

Interpreted Loose values (%)

65.21, 83.97, 86.49, 58.19, 78.66, 84.77, 93.63, 89.26, 94.00, 84.32

Predicted 60% (6/10) correct loose knots based on the 85% threshold

Average (Loose) = 81.85%
Decimal = 0.8185

Conclusions/action items:

Next, I will need to compile all the testing data and results and create that section on the poster.

4/20/2026 Continued Manual ML Testing

Kate Hiller - Apr 20, 2026, 11:19 AM CDT

Title: Manual ML Testing

Date: 4/17/2026

Content by: Kate Hiller

Present: N/a

Goals: To conduct testing to collect accuracy data for the model

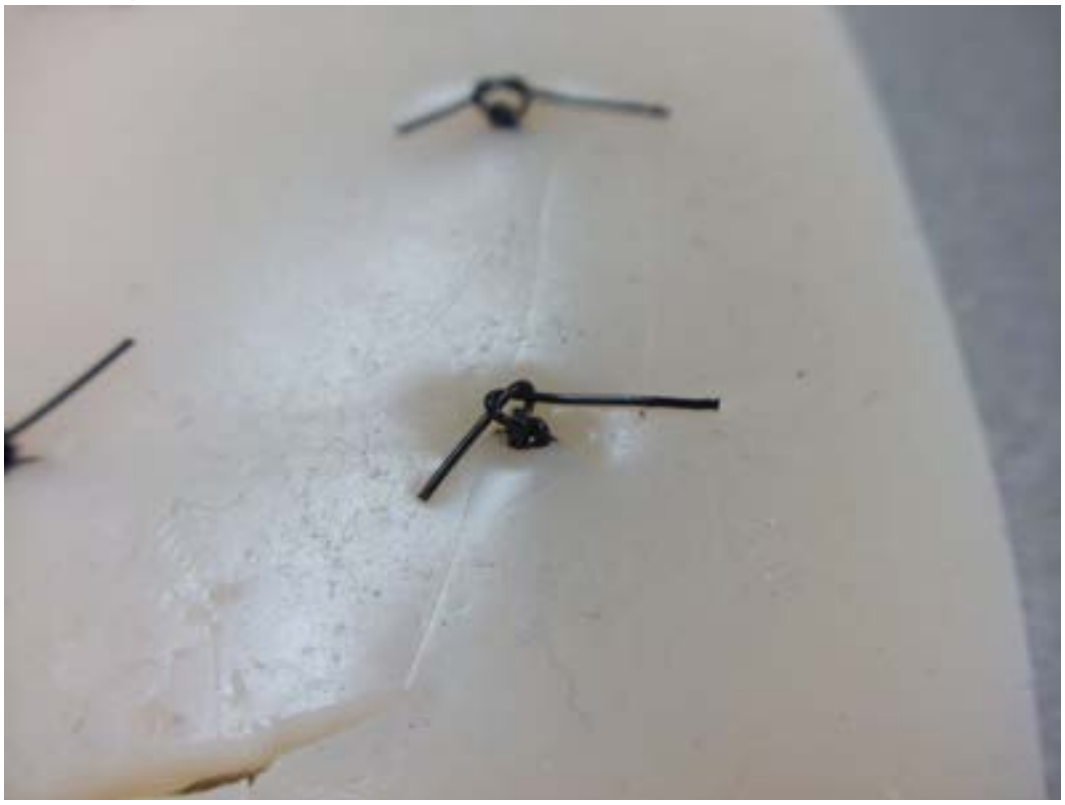
Content:

MODEL_PATH = "/home/knotorious-five/squaredetect/top_and_side_resnet_full.pth"

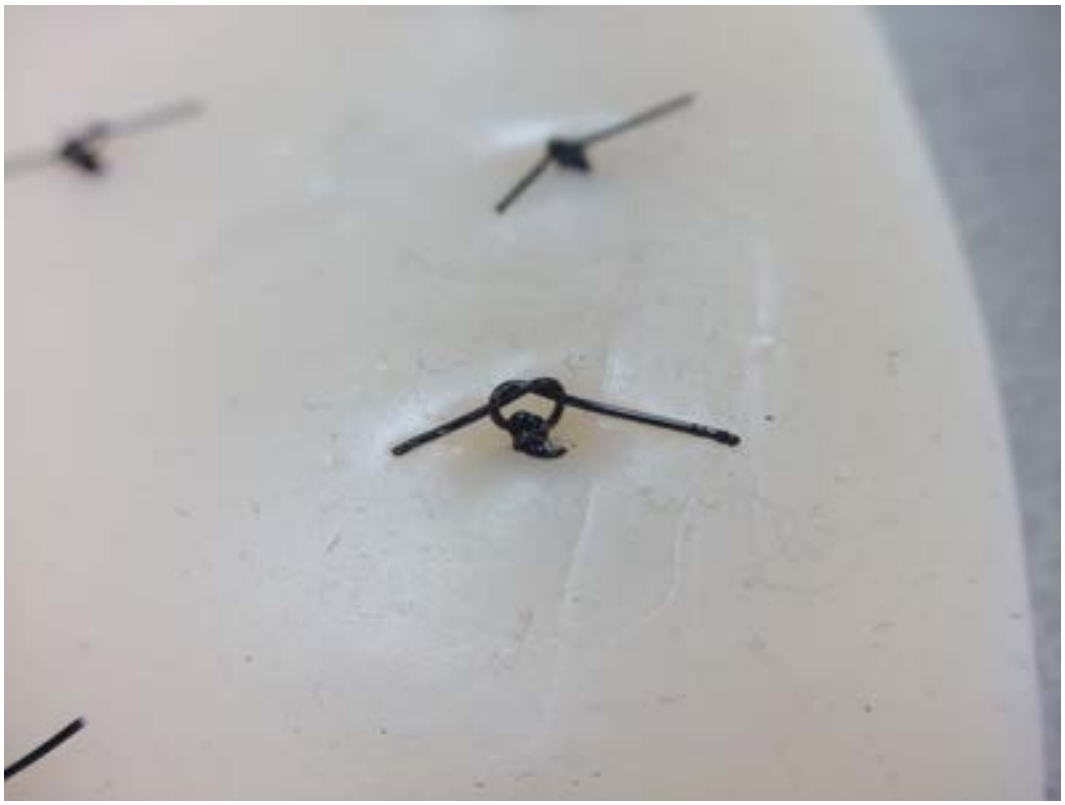
Loose 1: tight .6521

Loose 2 - 8397 tight (loose)

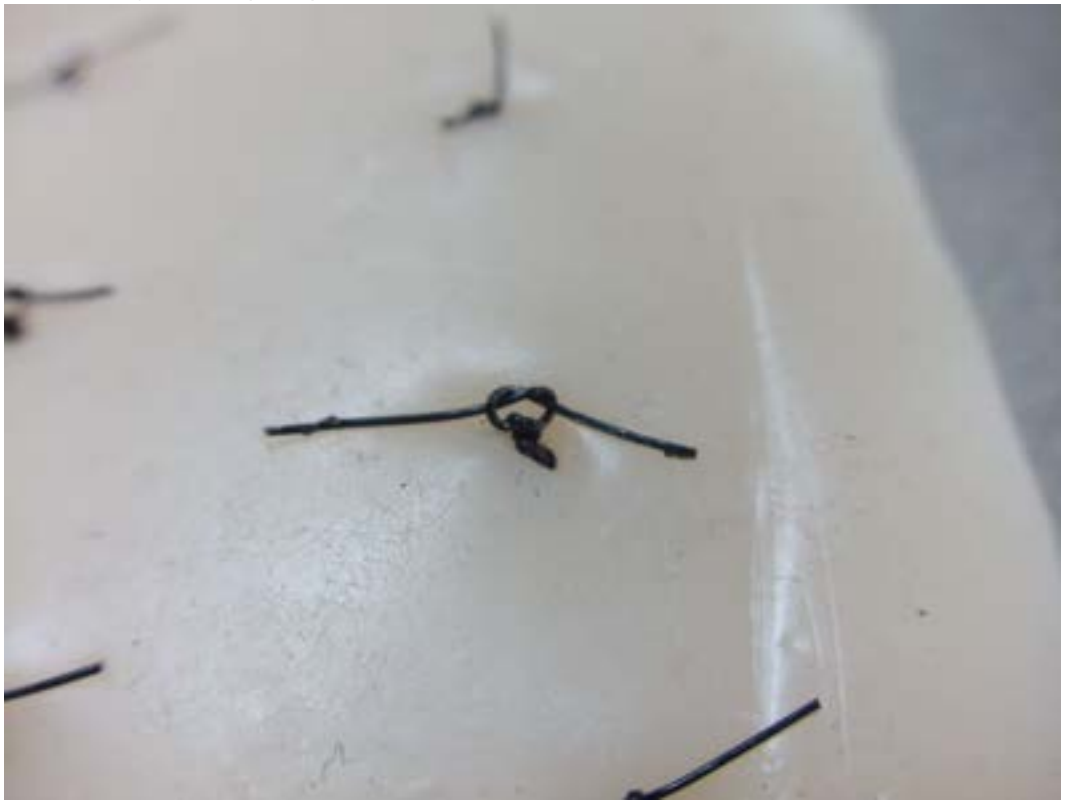
Loose 3 - tight 86.49



Loose 4 - tight 58.19

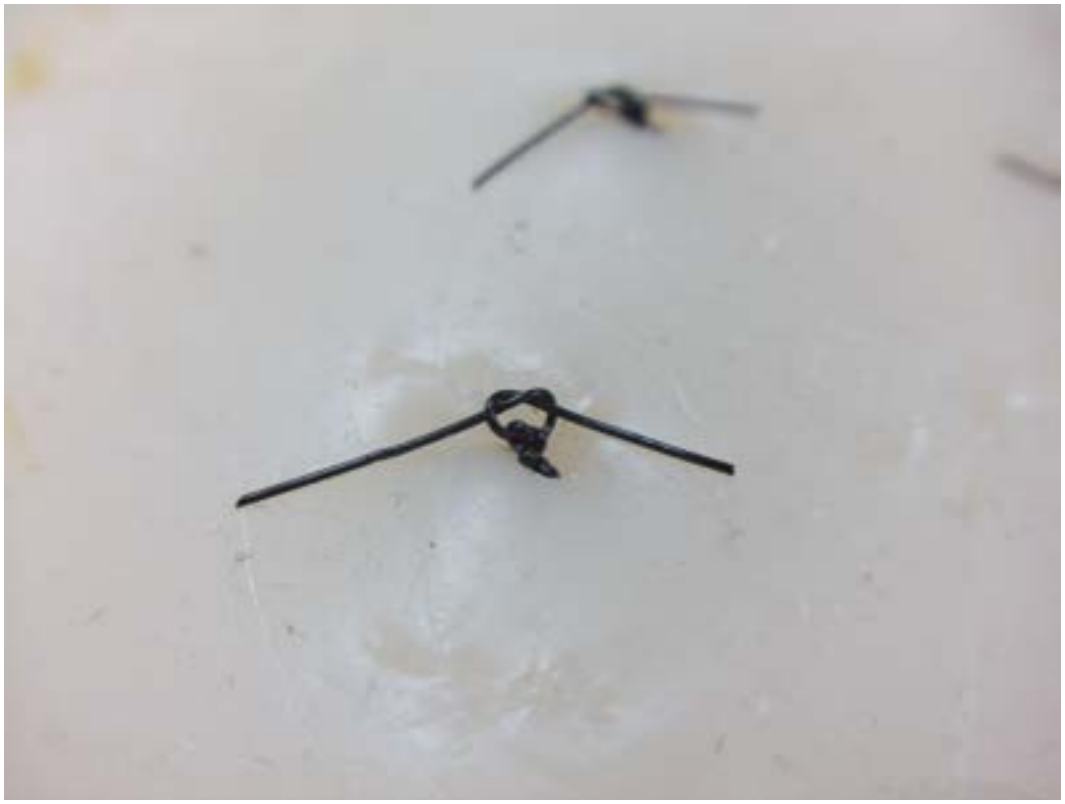


Loose 5 - Tight 78.66 (loose)



Loose 6 -84.77 tight (loose)

Loose 7 - tight 93.63



Loose 8- 89.26

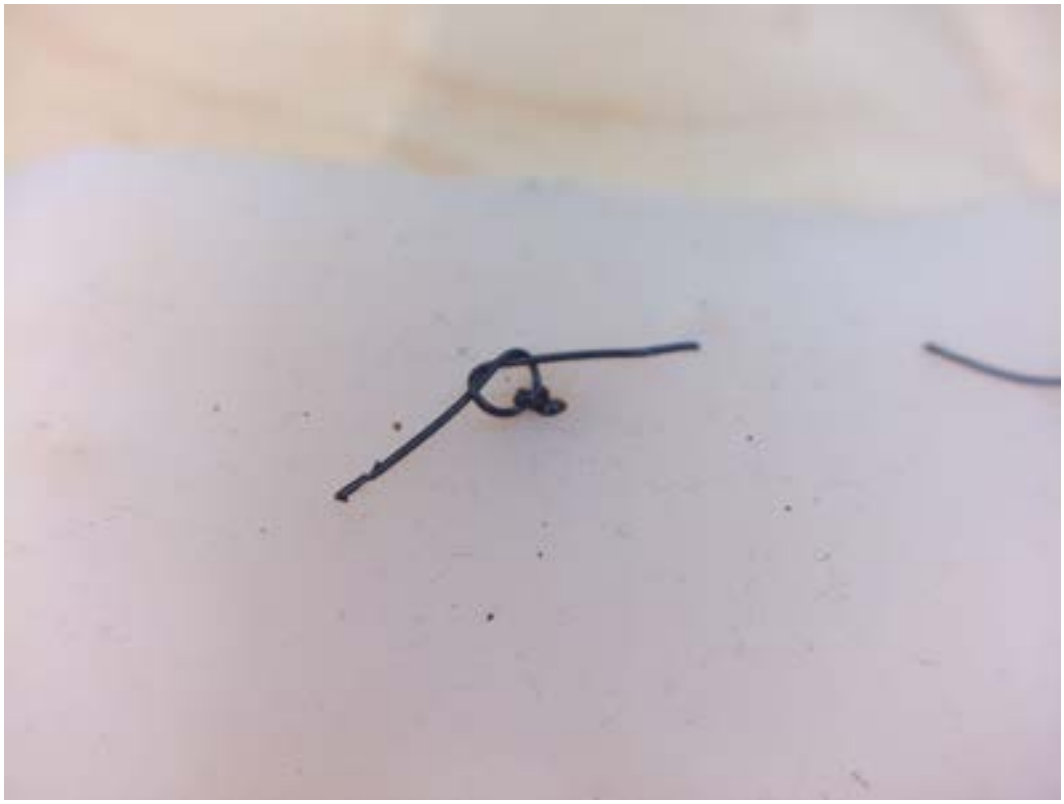
Loose 9 - Tight 94.00%

Loose 10 - Tight 84.32

Loose 11:

Tight 96.6

Loose 12: 81.56 Loose

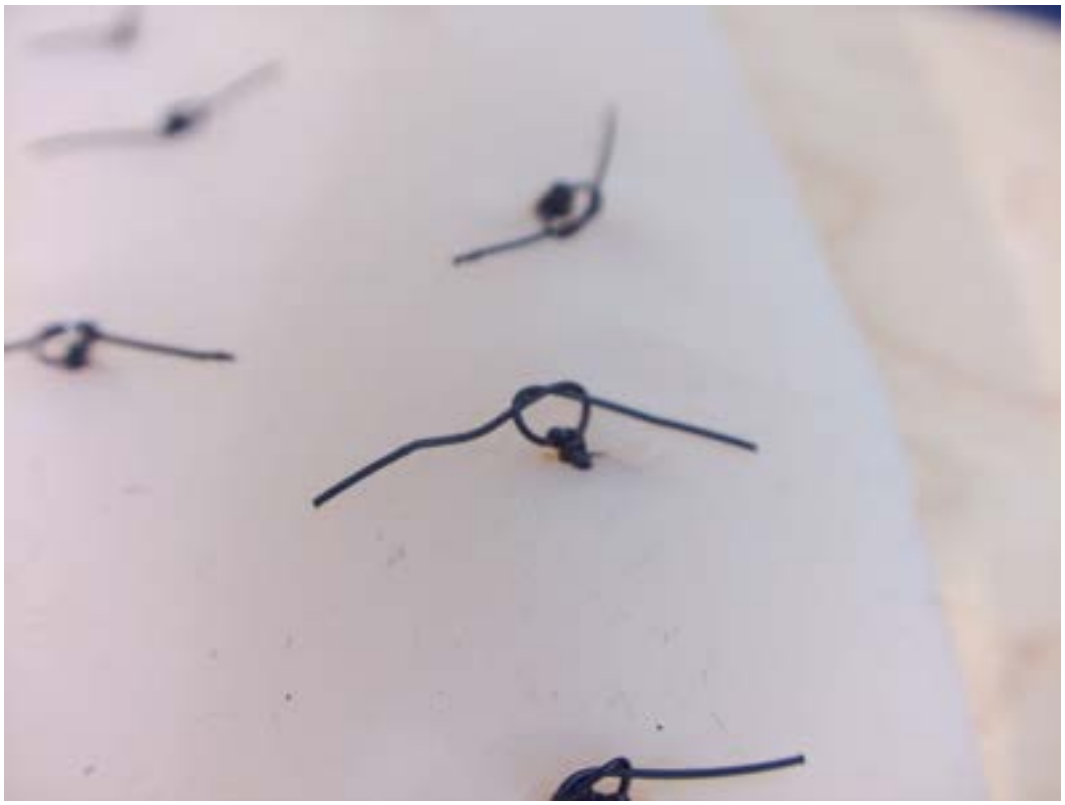


Loose 13: 81.92

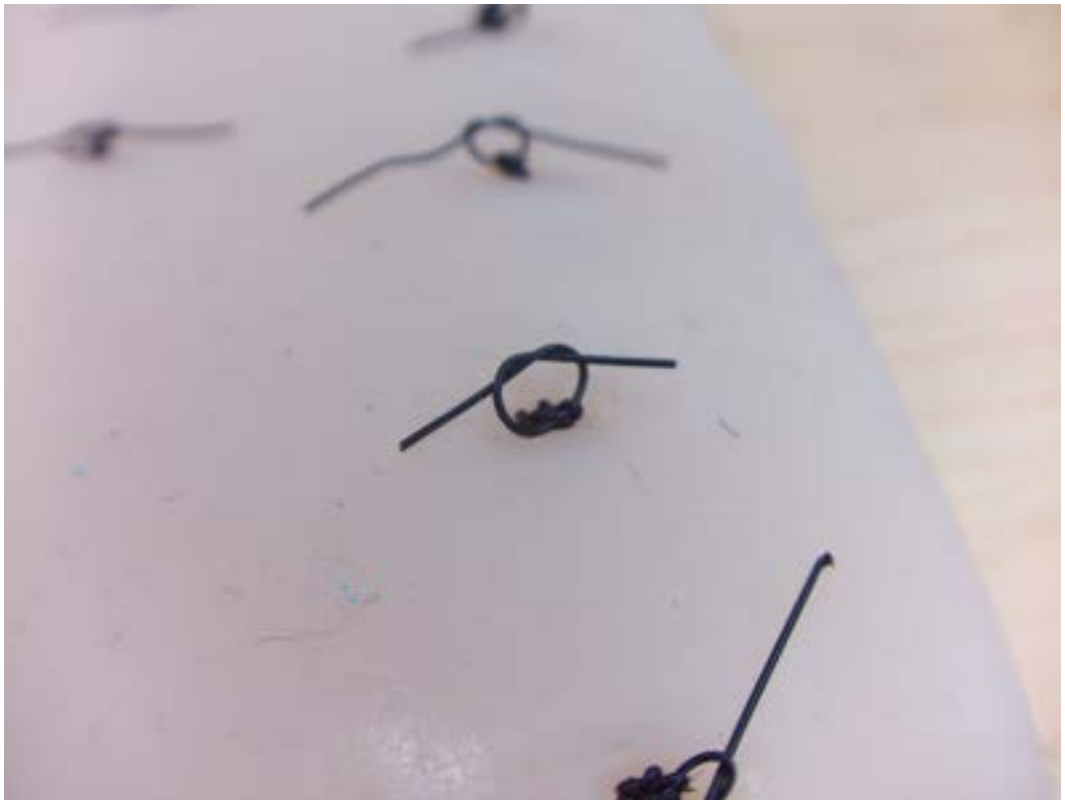
Loose 14: 67.71

Loose 15: 83.37\

Loose 16: 84.24

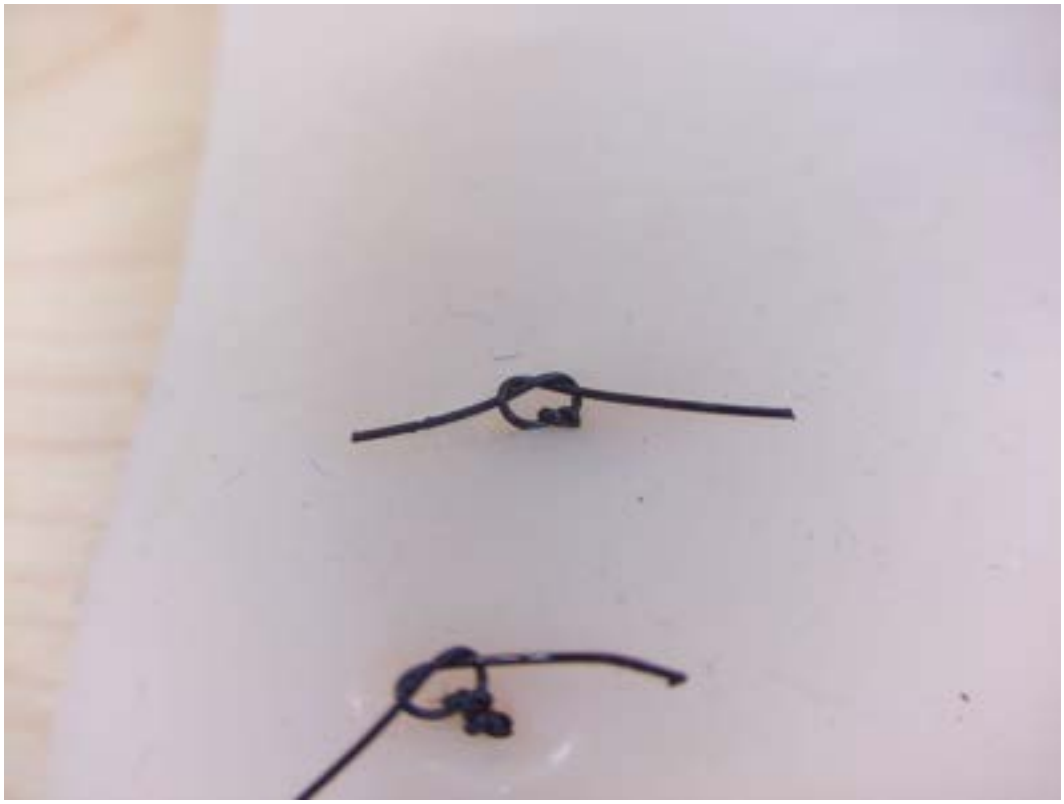


Loose 17: 80.79

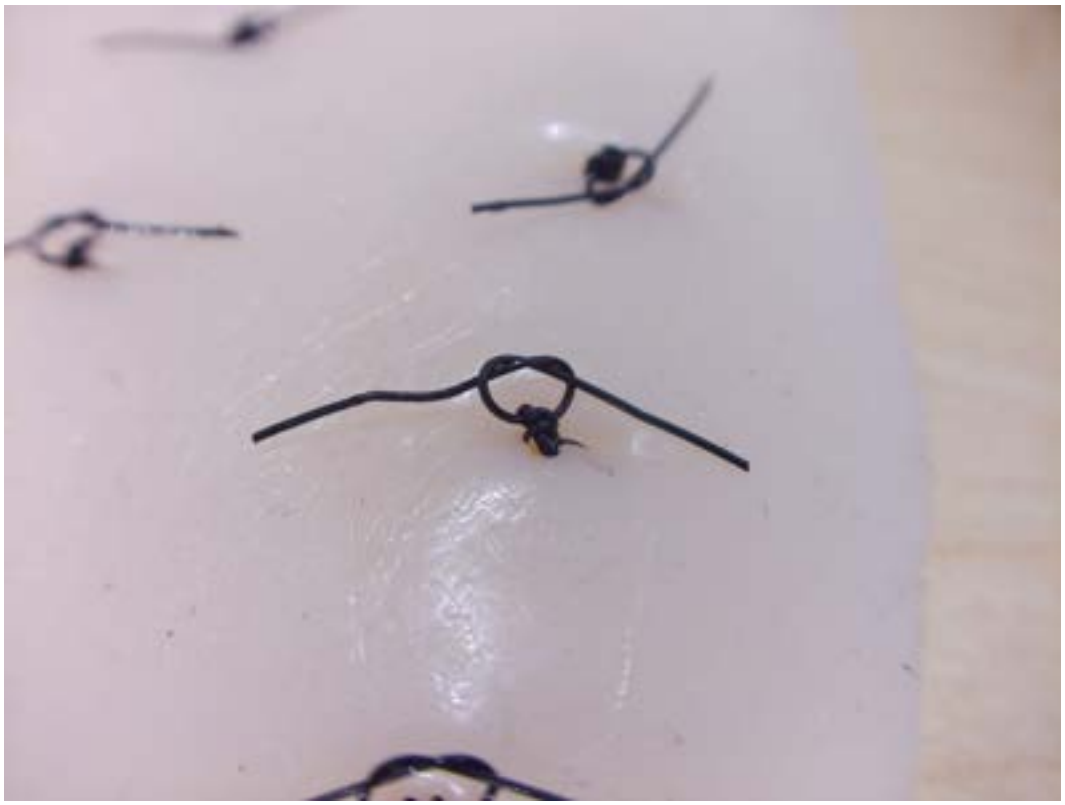


Loose 18: 61.30

Loose 19: 86.62



Loose 20: 56.05



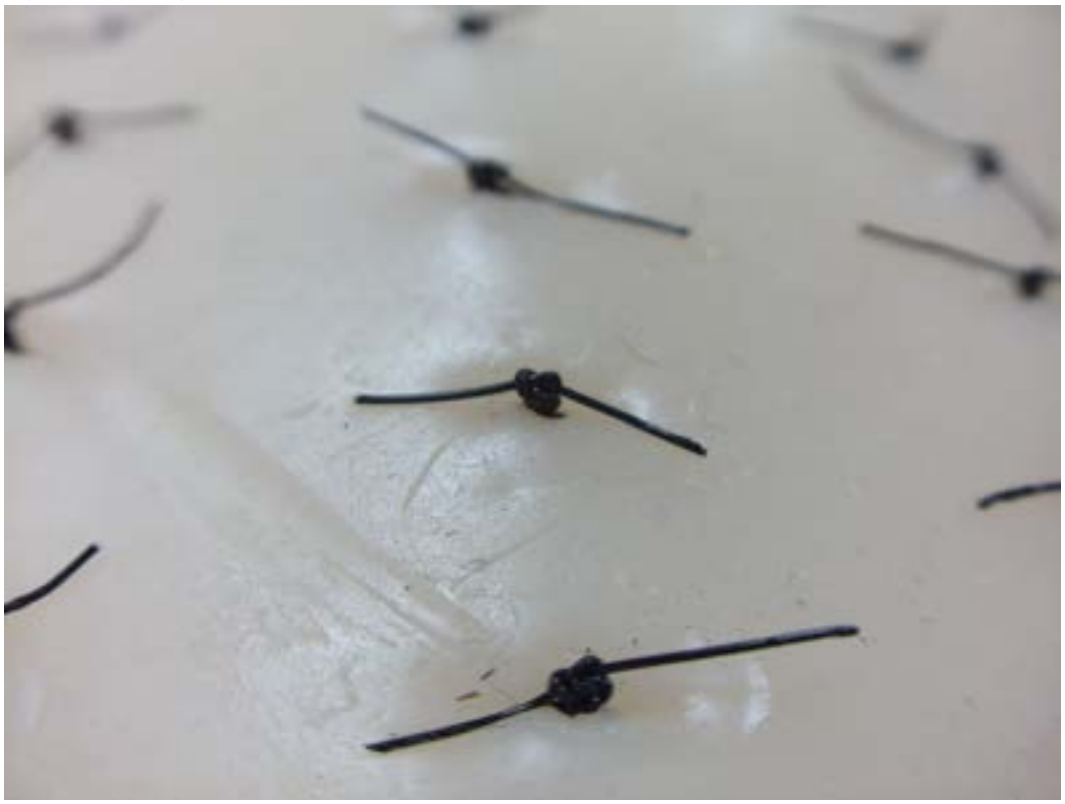
Interpreted Loose values (%)

65.21, 83.97, 86.49, 58.19, 78.66, 84.77, 93.63, 89.26, 94.00, 84.32

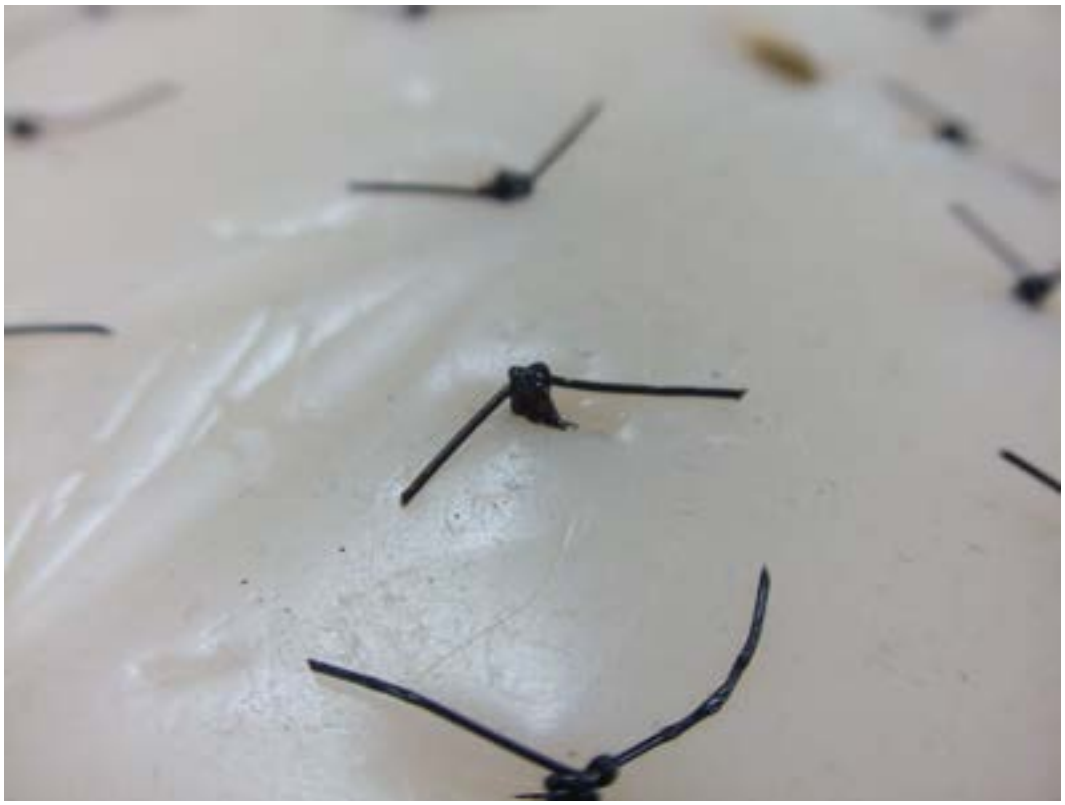
Predicted 70% (7/10) correct based on the 85% threshold

Average (Loose) = 81.85%
Decimal = 0.8185

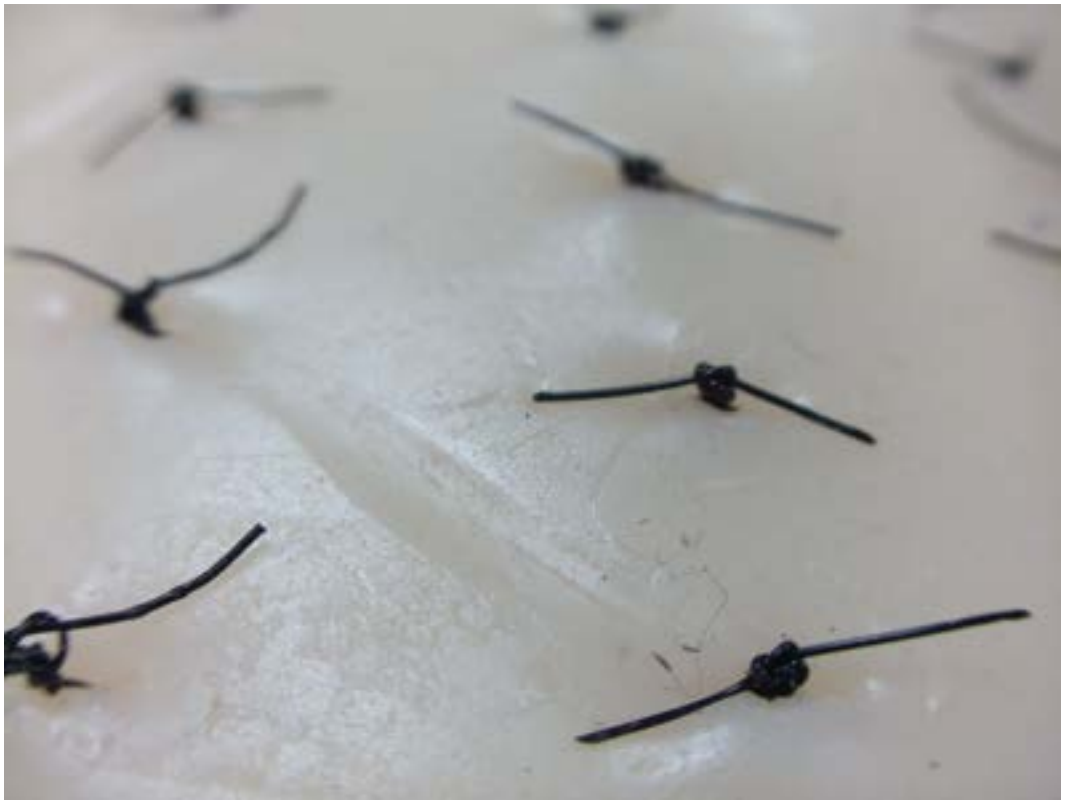
Tight 1 - 9498 tight



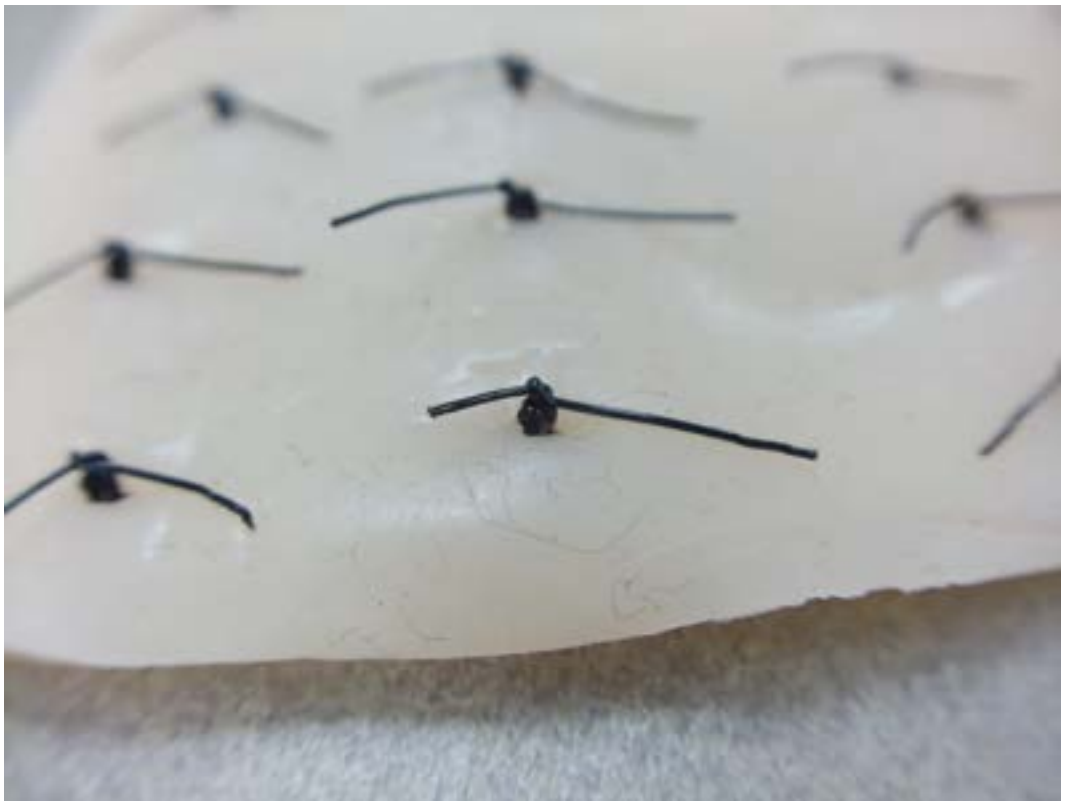
Tight 2 - 95.80 tight



Tight 3 - 97.16 tight

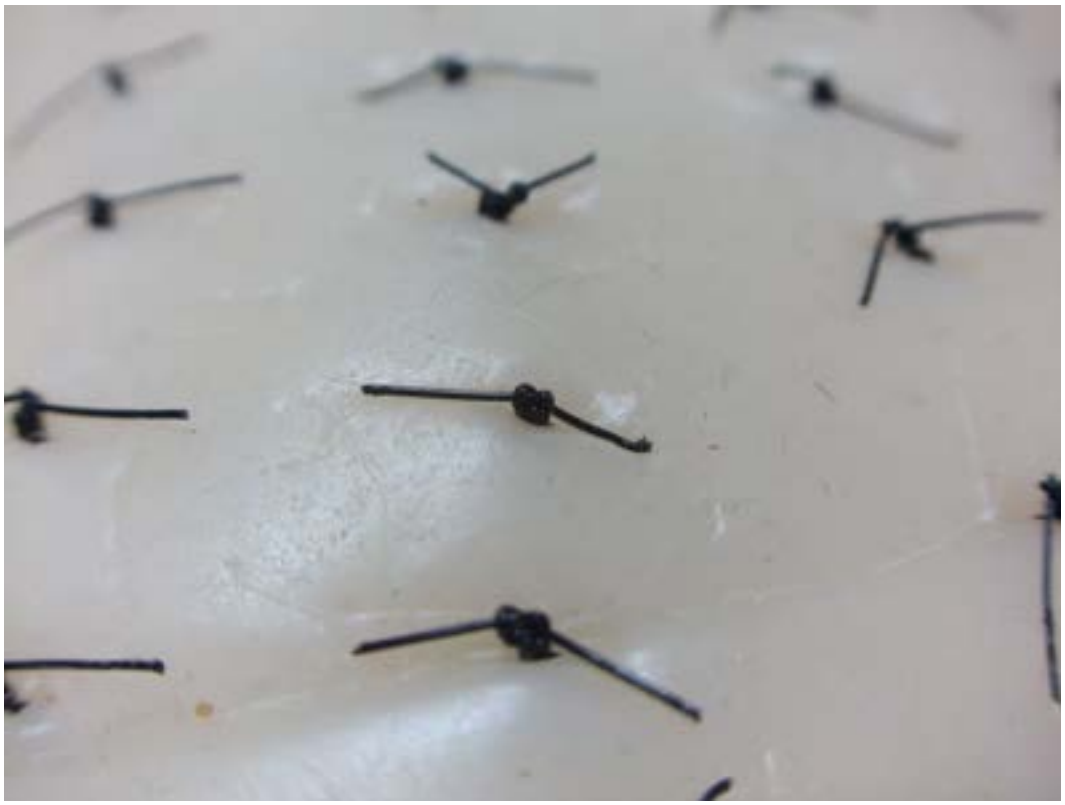


Tight 4 - tight 97.17

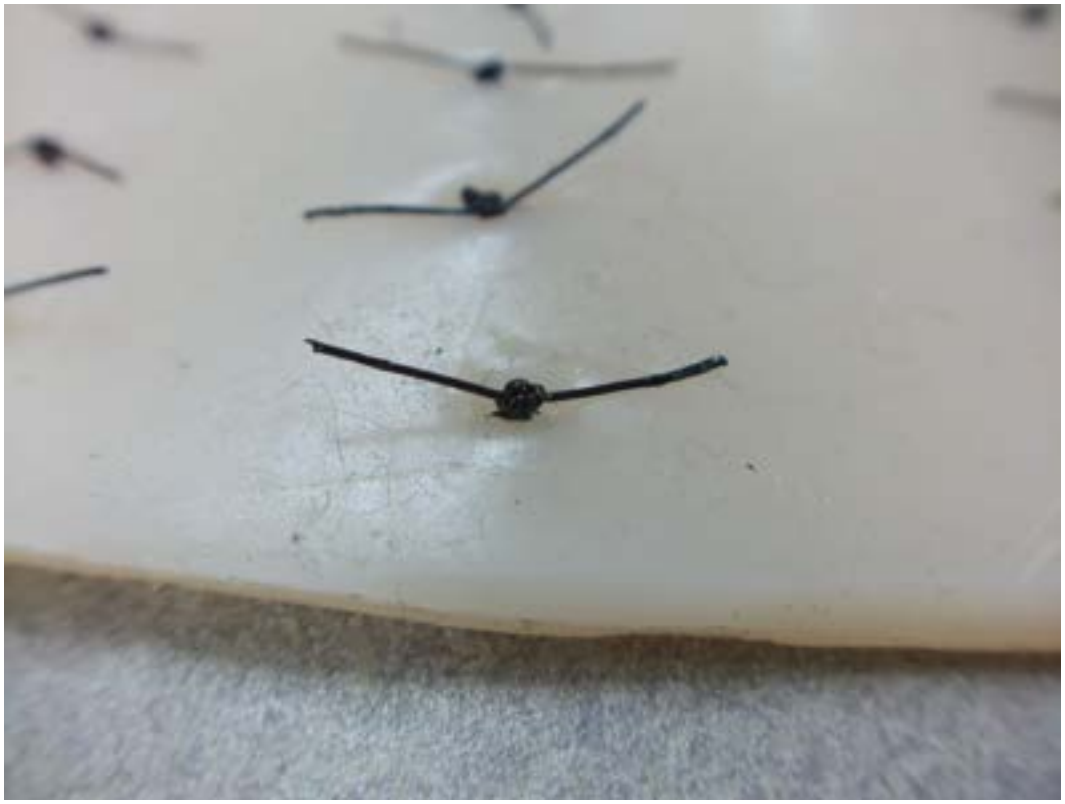


Tight 5 - tight 96.85

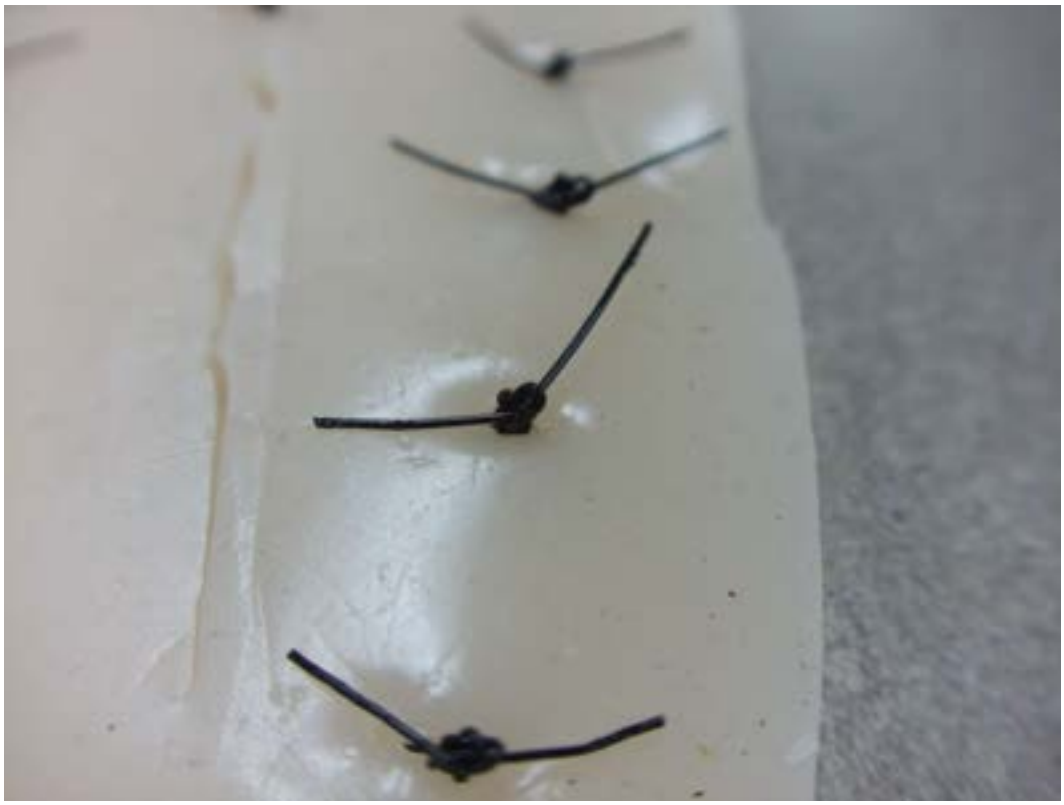
TIght 6 - tight 87.01



Tight 7 - tight 93.30

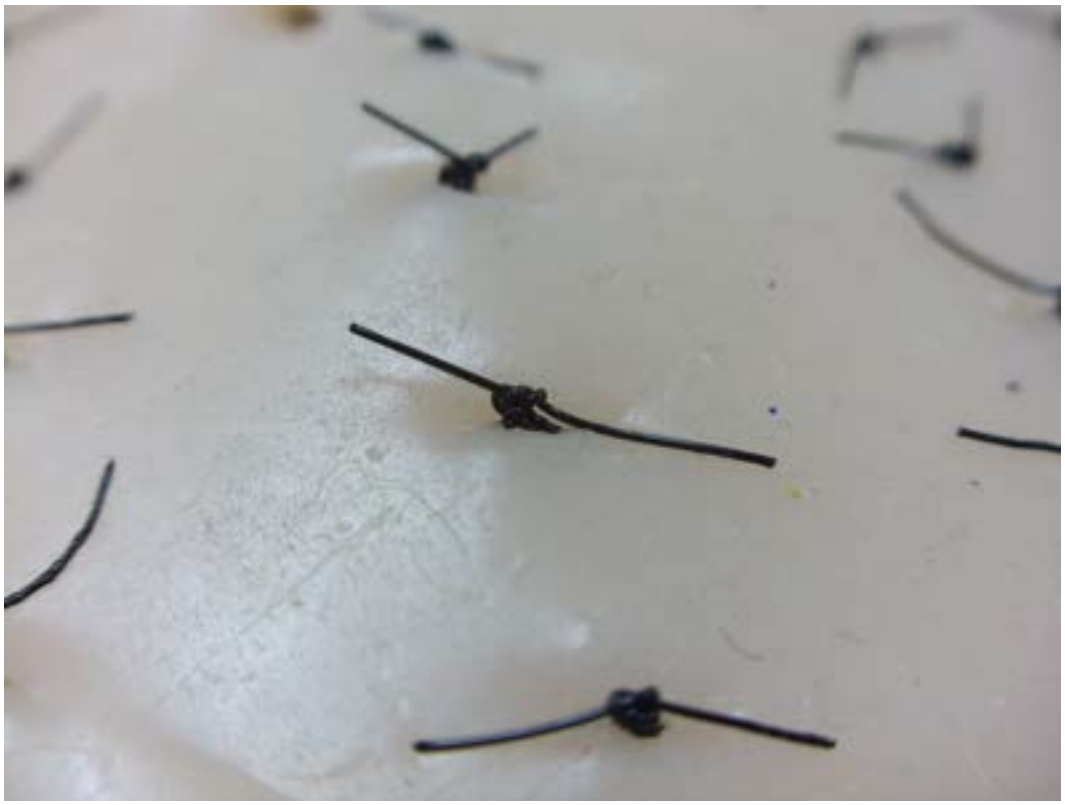


Tight 8 - 92.17 tight



Tight 9 - tight 92.62

Tight 10 - 98.23 tight



Tight 11: 92.96

Tight 12: 94.65

TIght 13: 97.08

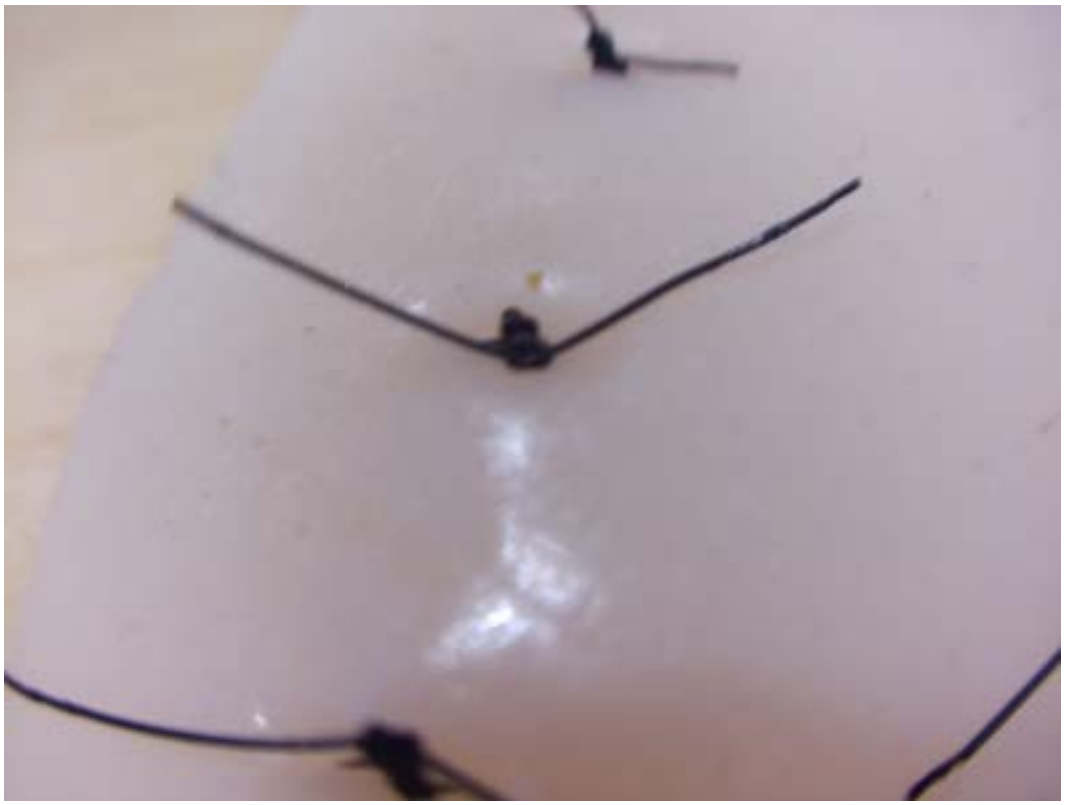
Tight 14: 93.43

Tight 15: 97.39

Tight 16:98.00

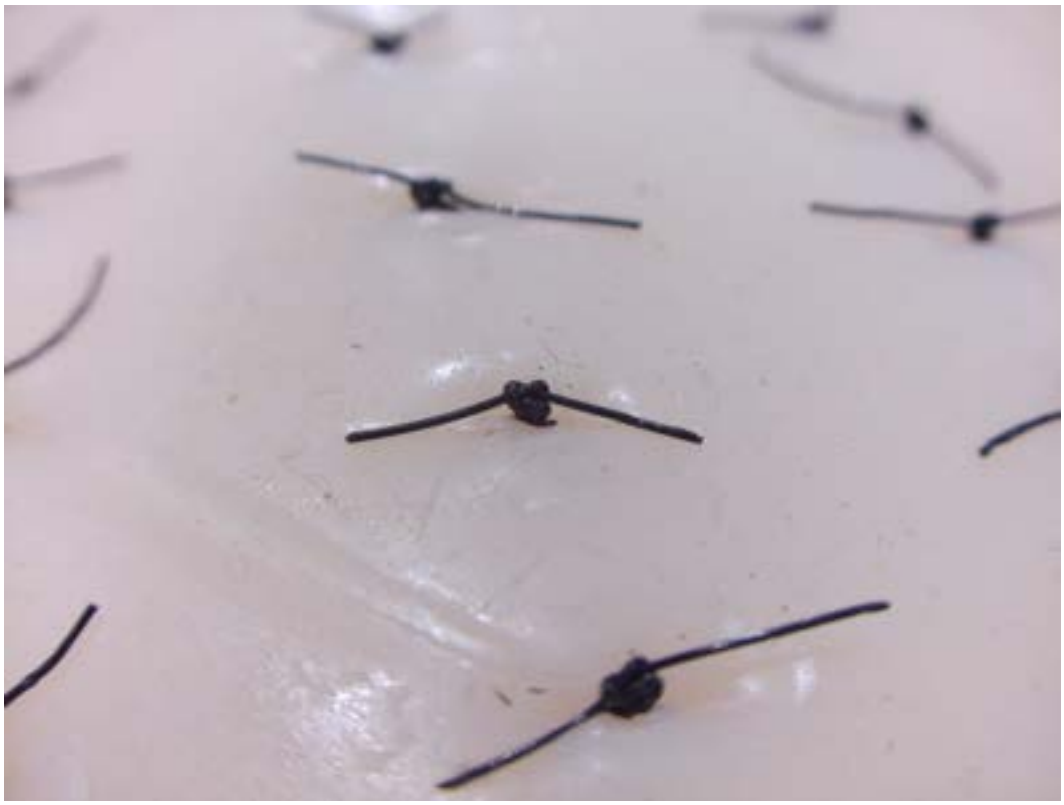
TIgh 17: 79.82

Tight: 18: 97.96



Tight 19: 97.74

Tight 20: 93.97



Loose Knots			Tight Knots		
Confidence (%)	**Prediction**	**Output**	**Confidence (%)**	**Prediction**	**Output**
65.21	Tight	Loose	94.98	Tight	Tight
83.97	Tight	Loose	95.80	Tight	Tight
86.49	Tight	Tight	97.16	Tight	Tight
58.19	Tight	Loose	97.17	Tight	Tight
78.66	Tight	Loose	96.85	Tight	Tight
84.77	Tight	Loose	87.01	Tight	Tight
93.63	Tight	Tight	93.30	Tight	Tight
89.26	Tight	Tight	92.17	Tight	Tight
94.00	Tight	Tight	92.62	Tight	Tight
84.32	Tight	Loose	98.23	Tight	Tight

Tight values (%)

94.98, 95.80, 97.16, 97.17, 96.85, 87.01, 93.30, 92.17, 92.62, 98.23

Average (Tight) = 94.13%
Decimal = 0.9413

Interpreted Loose values (%)

65.21, 83.97, 86.49, 58.19, 78.66, 84.77, 93.63, 89.26, 94.00, 84.32

Predicted 60% (6/10) correct loose knots based on the 85% threshold

Average (Loose) = 81.85%
Decimal = 0.8185

	Actually Tight	Actually Loose
Predicted Tight	True Positive (TP)	False Positive (FP)
Predicted Loose	False Negative (FN)	True Negative (TN)

Model Performance

38	12
2	28

Table X. Confusion matrix for model performance. A true positive (TP) is defined as a tight knot correctly classified as tight, and a true negative (TN) as a loose knot correctly classified as loose (n = 80).

Conclusions/action items:

Next, I will need to compile all the testing data and results and create that section on the poster.

4/19/2026 Latency Testing Analysis

Kate Hiller - Apr 20, 2026, 11:18 AM CDT

Title: Latency Testing Analysis

Date: 4/20/26

Content by: Kate Hiller

Present:

Goals: To create a way to present latency data

Content:

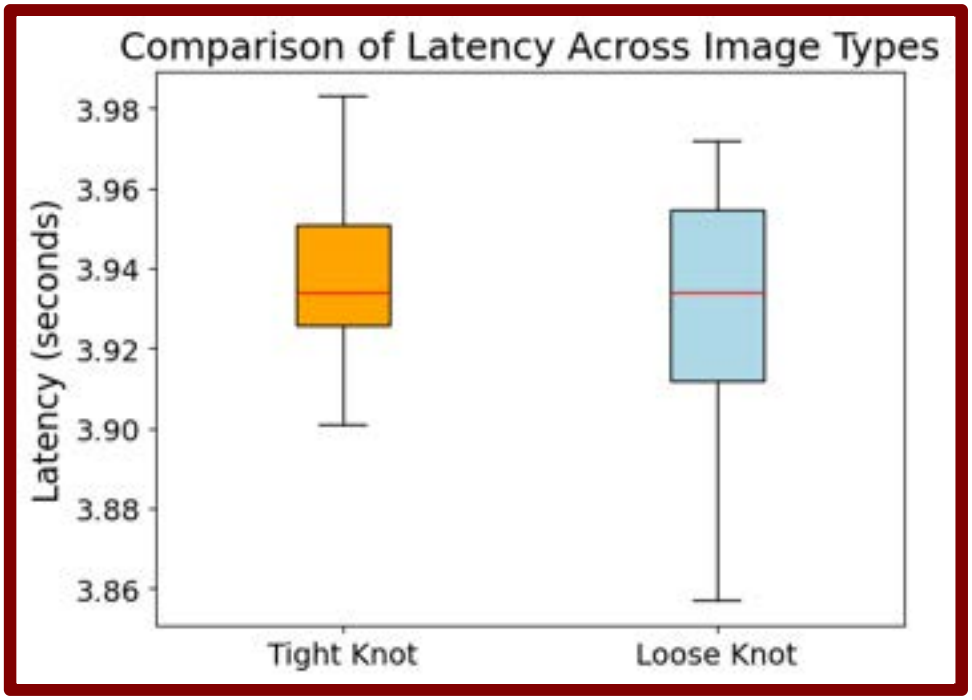
Knot Type	**Tight Knot**										**Loose Knot**									
Trials	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
Press button to take photo --> LED feedback (sec)	3.983	3.961	3.954	3.931	3.942	3.901	3.913	3.937	3.924	3.931	3.972	3.899	3.857	3.929	3.911	3.915	3.951	3.956	3.939	3.963

	Tight	**Loose**		**Mean all:**	3.93345															
Mean	3.9377	3.9292																		

Python code:

```
import matplotlib.pyplot as plt


tight = [3.983, 3.961, 3.954, 3.931, 3.942, 3.901, 3.913, 3.937, 3.924, 3.931]

loose = [3.972, 3.899, 3.857, 3.929, 3.911, 3.915, 3.951, 3.956, 3.939, 3.963]


data = [tight, loose]


box = plt.boxplot(
    data,
    positions=[1, 2],
    widths=0.25,
    patch_artist=True,
```

```
    medianprops=dict(color="red")
)



colors = ["orange", "lightblue"]

for patch, color in zip(box["boxes"], colors):
    patch.set_facecolor(color)

plt.xticks([1, 2], ["Tight Knot", "Loose Knot"], fontsize=14)
plt.ylabel("Latency (seconds)", fontsize=16)
plt.title("Comparison of Latency Across Image Types", fontsize=18)
plt.yticks(fontsize=14)
```



Conclusions/action items:

Latency (seconds) from button press to LED feedback for tight and loose knot conditions. This is for the poster presentation to display data.

3/4/24 Intro to Machining Certification

Kate Hiller - Mar 09, 2024, 11:31 AM CST

Title: Intro to Machining Certification

Date: 3/4/24

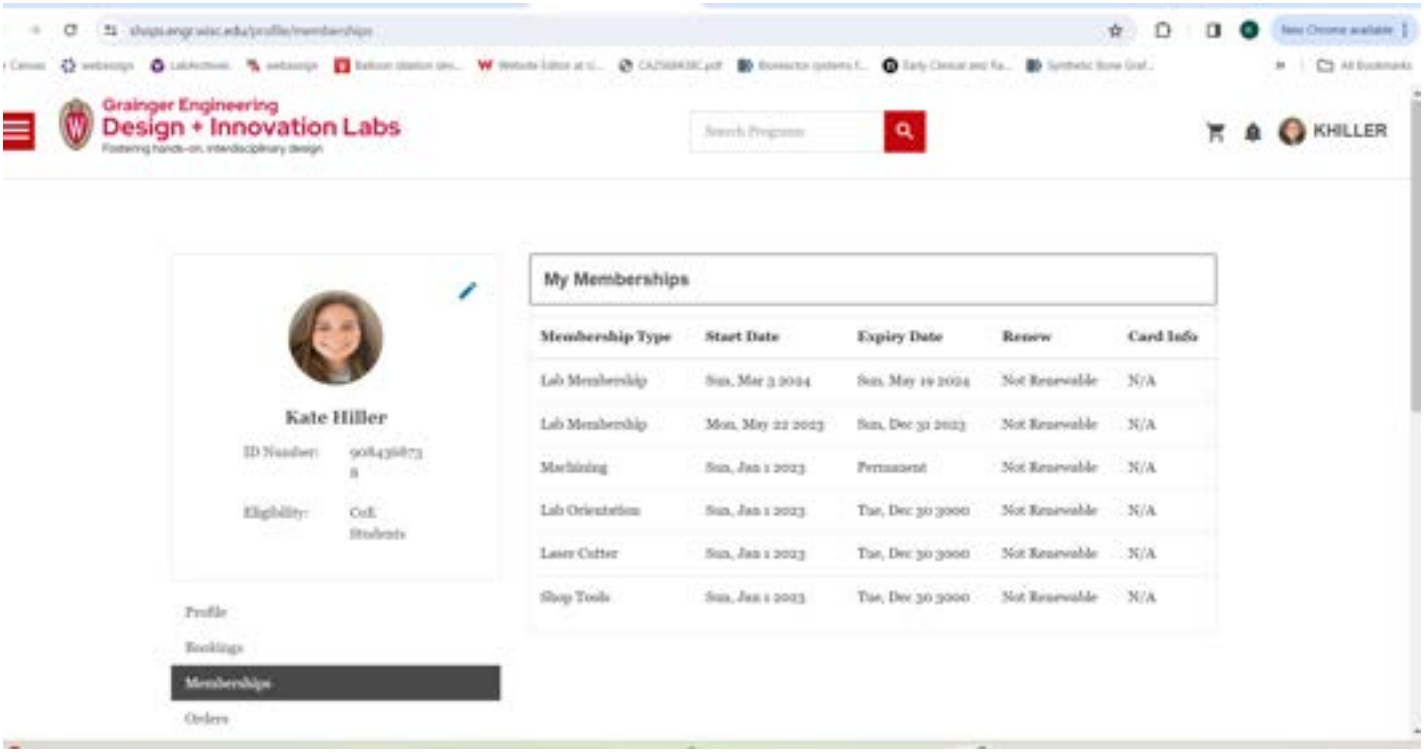
Content by: Kate Hiller

Present: N/A

Goals: To complete the intro to machining on the mill and lathe

Content:

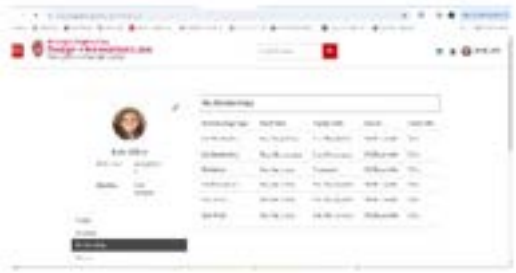
See the attached document. I was not given a physical permit.



Conclusions/action items:

This certification will be used in the continued fabrication of the project.

Kate Hiller - Mar 04, 2024, 1:52 PM CST



Download

Green_Card.pdf (201 kB)

3/9/24 Biosafety Certification Completion

Kate Hiller - Mar 09, 2024, 1:21 PM CST

Title: Biosafety Certification Completion

Date: 3/9/24

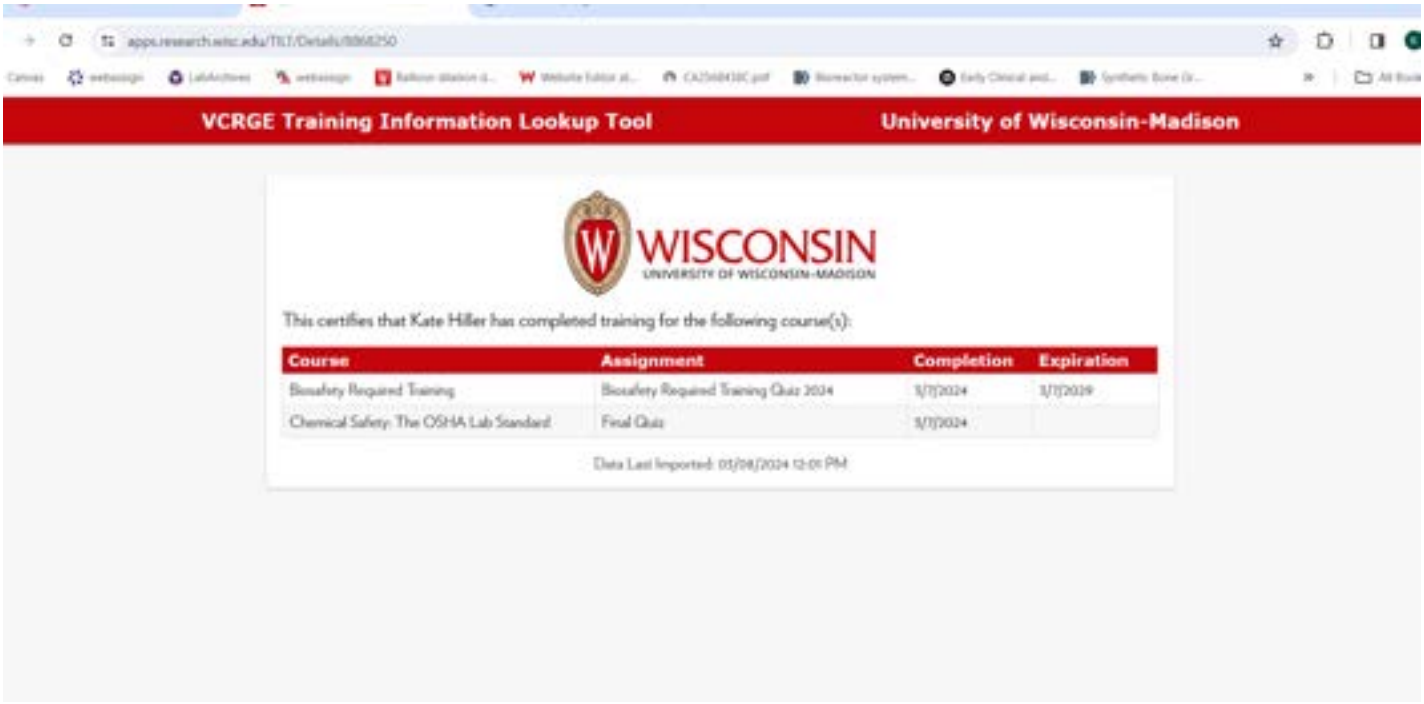
Content by: Kate Hiller

Present: N/A

Goals: Complete the two chemical/biosafety certifications

Content:

This is an image of the proof of completion:



Conclusions/action items:

This certification will be used so the team can go to the labs and fabricate the biomaterial and buffer solution for the bioreactor.

11/16/25 Ethics in Research

Presley Hansen - Nov 21, 2025, 1:34 PM CST

Title: Ethics in Research

Date: 11/16/25

Content by: Kate

Present: Kate

Goals: To complete new training for assignment

Content:

I trained for responsible and ethical conduct of research (RECR)

OVCR Training Information Lookup Tool — University of Wisconsin-Madison

WISCONSIN
UNIVERSITY OF WISCONSIN-MADISON

This certifies that Kate Hiller has completed training for the following course(s):

Course	Assignment	Completion	Expiration
Biosafety Required Training	Biosafety Required Training Quiz 2024	3/7/2024	3/7/2029
Chemical Safety: The OSHA Lab Standard	Final Quiz	3/7/2024	
Responsible and Ethical Conduct of Research (RECR)	RCR Certification	10/22/2025	No Expiration

Data Last Imported: 10/22/2025 07:25 PM

Conclusions/action items:

I have completed this training and submitted it before the deadline.

3/16/26 HIPAA Privacy & Security Training

Kate Hiller - Mar 16, 2026, 2:15 PM CDT

Title: HIPAA Privacy and Security Training

Date: 3/16/2026

Content by: Kate

Present: Kate

Goals: To complete new training

Content:

I trained for HIPAA Privacy and Security Training

OVCR Training Information Lookup Tool — University of Wisconsin-Madison

WISCONSIN
UNIVERSITY OF WISCONSIN-MADISON

This certifies that Kate Hiller has completed training for the following course(s):

Course	Assignment	Completion	Expiration
2025-2026 HIPAA Privacy & Security Training	2025-2026 HIPAA Privacy & Security Training	3/16/2026	
Biosafety Required Training	Biosafety Required Training Quiz 2024	3/7/2024	3/7/2029
Chemical Safety: The OSHA Lab Standard	Final Quiz	3/7/2024	
Responsible and Ethical Conduct of Research (RECR)	RCR Certification	10/22/2025	No Expiration

Data Last Imported: 05/16/2026 01:54 PM

Conclusions/action items:

I have completed this training and submitted it before the deadline.

3/6/2026 Tong Lecture

Kate Hiller - Mar 07, 2026, 5:01 PM CST

Title: Tong Lecture

Date: 3/7/2026

Content by: Kate Hiller

Present: n/a

Goals: To listen to the Tong Distinguished Lecturer - Professor Williams

Content:

- Justin Williams is a professor at UW-Madison who shared his career developing neurotechnology and bioinstrumentation

- Losing his father to a neurological disease and a friend to a brain injury influenced him to become interested in neuroscience

- his technologies focused on helping patients with neurological conditions

- One lesson I took from the lecture is the importance of working with good people, since PD is a long-term effort

- I thought it was interesting that the idea of his product was in Grey's Anatomy

- He has been involved with many neurotechnology companies:

    - NeuroNexus - develops neural electrode arrays widely used in brain research

    - NeuroOne Medical Technologies - flexible electrodes for epilepsy treatment

    - He founded NeuraWorx which is a company that focuses on therapies for progressive supranuclear palsy by improving the brain's waste-clearing system

    - BrainSync

- He emphasized that failure is a part of innovation processes, and it is important to design with the end user in mind

Conclusions/action items:

Professor Williams' lecture was very insightful, and it was interesting to hear about all the work he has completed throughout his career. I learned that successful biomedical innovation requires more than strong technical skills. His career of working and founding different companies shows the possibilities and how dynamic your career can be, which is inspiring.

01/27/26 - Camera types

Presley Hansen - Jan 27, 2026, 7:19 PM CST

Title: Camera Types

Date: 01/27/26

Content by: Presley Hansen

Present: N/A

Goals: Determine possible camera options to use with our machine learning model

Content:

USB Machine Vision Camera:

- pros: high image quality, low latency
- cons: more expensive
- typically fixed lens but industrial USB cameras typically have interchangeable lenses
- 12 MP (4608x2592)

Raspberry Pi Camera Module v3:

- pros: compact, inexpensive
- cons: fixed lens
- autofocused which can be good and bad
- 12 MP (4608x2592)

Raspberry Pi HQ Camera*:

- pros: interchangeable lens, macro capable, focus lock, excellent detail, stable image
- cons: higher cost
- manual focus only
- 12.3 MP (4056x3040)

Raspberry Pi AI Camera:

- pros: low latency,
- cons: higher cost, fixed lens, not great macro capability, not long focus distance
- 12.3 MP (4056x3040)

Global Shutter Raspberry Pi Cameras:

- pros: macro-capable, no motion distortion, industrial-grade imaging
- cons: lower resolution, more expensive, less texture detail
- 1.6 MP (1456x1088)

- iphone 15 resolution of photos are often 12 MP (4032x3024) --> this is close to the Raspberry Pi camera resolutions

References:

[1] R. P. Ltd, “Raspberry Pi,” *Raspberry Pi*, 2024. https://www.raspberrypi.com/

[2] Aleksandar Anastasov, “iPhone 15 camera: All upgrades and new features,” *PhoneArena*, Sep. 13, 2023. https://www.phonearena.com/news/iphone-15-camera_id148922?utm_source=chatgpt.com (accessed Jan. 28, 2026).

Conclusions/action items: The Raspberry Pi HQ Camera seems to best option because it provides high-resolution images (~12.3 MP) with interchangeable macro-capable lenses and stable manual focus, ensuring clear, repeatable views of small suturing knots. Its resolution matches our iPhone-trained ML model, and it integrates easily with the Raspberry Pi for real-time LED feedback.

01/27/26 - Image Augmentation Strategies

Presley Hansen - Jan 27, 2026, 7:45 PM CST

Title: Image Augmentation Strategies

Date: 01/27/26

Content by: Presley Hansen

Present: N/A

Goals: To determine some image augmentation strategies so we can add more images to our ML data set.

Content:

Geometric transformations:

- rotate images 5-15 degrees
- flip/mirror
- shift knot 10-20 pixels in x or y direction
- zoom in/out 10-20%

Color and lighting adjustments:

- raise/lower brightness/contrast 10-20%
- raise/lower saturation by 10%

- can try segmenting the knot and placing on different backgrounds as well

Conclusions/action items: should try applying 2-3 augmentations at once to increase differences between photos

References:

[1] Vamsikd, “Data Augmentation Techniques in Computer Vision and NLP: A Complete Guide,” *Medium*, Mar. 26, 2025. https://medium.com/%40vamsikd219/data-augmentation-techniques-in-computer-vision-and-nlp-a-complete-guide-6cbbeabc3a53 (accessed Jan. 28, 2026).

01/27/26 - K-fold Cross-Validation

Presley Hansen - Jan 27, 2026, 8:06 PM CST

Title: K-fold Cross-Validation

Date: 01/27/26

Content by: Presley Hansen

Present: N/A

Goals: To learn about the method of k-fold cross-validation

Content:

- method to assess how well your model generalizes to unseen data

- split your dataset into k roughly equal folds (folds are the k equal-sized and non-overlapping subsets of data)

- train the model k times, each time using a different fold as the validation set and the remaining k-1 folds as the training set (training set fits the model's parameters and the validation set tunes the model's hyperparameters)

- average the performance metrics across the k runs to get at better estimate of model performance

- example: for 5 folds, the first iteration would be using F2, F3, F4, and F4 to train and then F1 to validate ; switch validation for the other four iterations --> compute average metrics for accuracy, F1-score, etc.

- this method maximizes the use of limited images, reduces bias, reduces overfitting risk, and ensures all images are used in both training and validation at least once

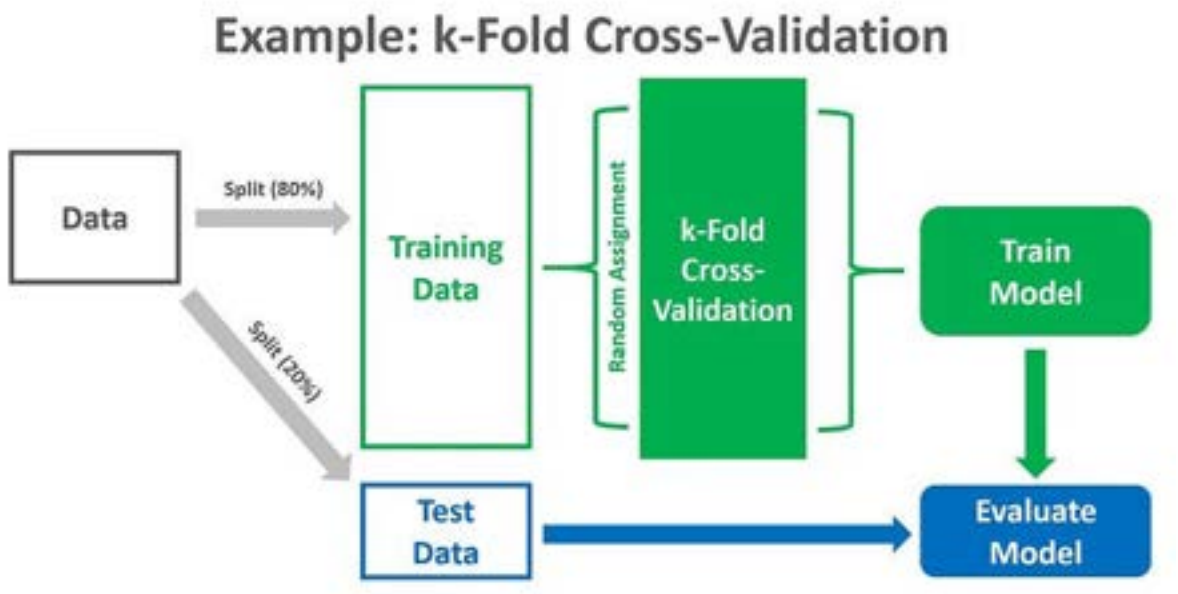


Figure 1: K-Fold Cross-Validation Process [2]

Conclusions/action items: K-Fold Cross-Validation is a method for evaluating machine learning models by splitting the dataset into K folds and training K times, each time using a different fold for validation. It ensures that every image is used for both training and validation, providing a more reliable estimate of model performance. This is especially useful for small datasets to reduce bias and improve generalization.

References:

[1] J. Brownlee, “A Gentle Introduction to k-fold Cross-Validation,” *Machine Learning Mastery*, Oct. 04, 2023. https://machinelearningmastery.com/k-fold-cross-validation/

[2] M. Hassanzadeh, “Why we should use K-fold Cross Validation?,” *Medium*, Mar. 22, 2023. https://medium.com/@mehrdadhz.75/why-we-should-use-k-fold-cross-validation-63209c8046f9

2026/02/09 - Model Implementation

Presley Hansen - Feb 09, 2026, 12:45 PM CST

Title: Model Implementation into Raspberry Pi

Date: 2026/02/09

Content by: Presley Hansen

Present: N/A

Goals: To get more familiar with how to implement our machine learning model with a raspberry pi 5 board

Content:

- general overview is to: set up the raspberry pi OS --> install ML libraries (Tensorflow, pytorch, etc) --> transfer model to pi

1. setting up the raspberry pi OS:

- use the official Raspberry Pi Imager tool on your computer to write the OS image to a microSD card

- download the Imager from the Raspberry Pi website (https://www.raspberrypi.com/software/)

- select your Pi model and OS, choose the SD card as storage, customize settings

- write the image and insert the card into your Pi and power it on to finish setup.

2. install ML libraries:

- install python and pip: `sudo apt-get install python3-pip`

- **Virtualenv (r**un TensorFlow inside a virtual environment to avoid messing with your system's global packages): `sudo pip3 install virtualenv`

- TensorFlow requires **NumPy:** `pip3 install numpy`

- need openCV for image processing:

```
sudo apt-get install libopencv-dev
pip3 install opencv-python
```

- create and activate virtual environment for tensorflow download:

```
 sudo pip3 install virtualenv
virtualenv --system-site-packages -p python3 tensorflow_env
source tensorflow_env/bin/activate
```

- install tensorflow: `pip3 install tensorflow`

- test setup by running this script:

```
import tensorflow as tf
print(f"TensorFlow version: {tf.__version__}")
```

3. transfer model to pi:

- copy model onto USB drive

- Insert the USB drive into one of the Pi 5's USB ports (rasp pi 5 board will auto-mount)

- Open Terminal on the Pi and run code that will locate the usb drive on the rasperry pi and then copy the model to the raspberry pi home directory (run code to verify model works)

4. optimize model for raspberry pi (optional): Raspberry Pi has limited CPU/GPU, so optimization helps to maximize performance. This could include things like converting to tensorflow lite, reducing precision (shortening float lengths), and/or pruning (removes unnecessary weights/parameters)

5. connect and enable camera (steps depend on which camera will be used)

6. write a python script to run the model (i.e. with tensorflow and camera)

Conclusions/action items:

References:

[1] “How to Deploy a Deep Learning Model on Raspberry Pi,” *Raspberry Pi 5*, 2023. https://www.raspberrypi-5.com/questions/how-to-deploy-deep-learning-model-in-raspberry-pi/ (accessed Feb. 09, 2026). [2] H. Amit, “TensorFlow on Raspberry Pi - Data Scientist’s Diary - Medium,” *Medium*, Nov. 30, 2024. https://medium.com/data-scientists-diary/tensorflow-on-raspberry-pi-9297185fec3b

03/16/2026 - AI training documentation

Presley Hansen - Mar 16, 2026, 2:21 PM CDT

Download

CertificateOfCompletion_Programming_Foundations_Artificial_Intelligence.pdf (66 kB)

BME Deign - Fall 2025 - Notebook

Madison Michels - Jan 26, 2026, 12:48 PM CST

Table of Contents

Download

2026_01_25_notebook_74172.pdf (229 MB)

2014/11/03-Entry guidelines

John Puccinelli - Sep 05, 2016, 1:18 PM CDT

Use this as a guide for every entry

- Every text entry of your notebook should have the **bold titles** below.
- Every page/entry should be **named starting with the date** of the entry's first creation/activity, subsequent material from future dates can be added later.

You can create a copy of the blank template by first opening the desired folder, clicking on "New", selecting "Copy Existing Page...", and then select "2014/11/03-Template")

Title: Descriptive title (i.e. Client Meeting)

Date: 9/5/2016

Content by: The one person who wrote the content

Present: Names of those present if more than just you (not necessary for individual work)

Goals: Establish clear goals for all text entries (meetings, individual work, etc.).

Content:

Contains clear and organized notes (also includes any references used)

Conclusions/action items:

Recap only the most significant findings and/or action items resulting from the entry.

2014/11/03-Template

John Puccinelli - Nov 03, 2014, 3:20 PM CST

Title:

Date:

Content by:

Present:

Goals:

Content:

Conclusions/action items: