CSCE 5222 Feature Engineering

Project Plan

1. Problem statement

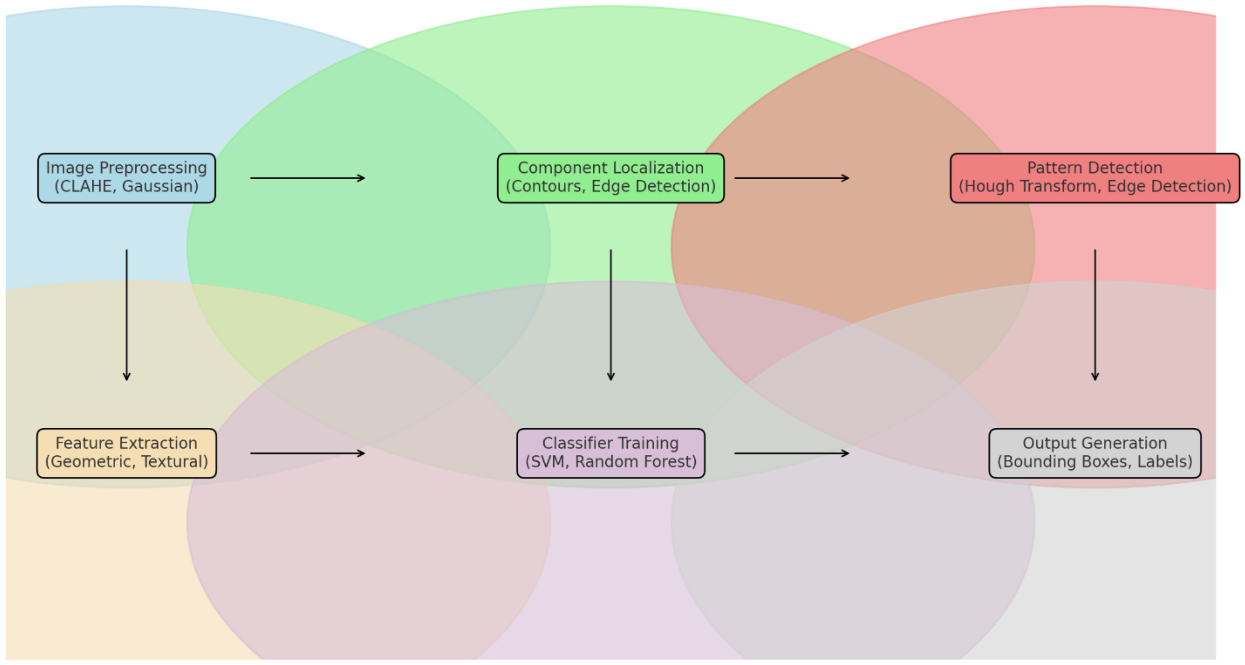
This project is designed to create an automated image processing system for detecting and locating square elements in images. These parts are stored as series of 1000x1000 pixels photos which can differ in light, surface characteristics and orientation. As a result of the changes in lighting (over-exposure or under-exposure), shapes on these components (strips, crosses) can become hidden and the components cannot be classified easily.

The intention is to find and spot the elements in the pictures and mark them as "Good" or "Bad" based on the apparent pattern. The good parts show distinct strip or cross, the bad parts do not show pattern. And to do that, we have to build a method that will be able to detect and localize them, and classify them, under harsh conditions of lighting. To localize is to identify the exact location of the objects in the image and apply bounding box.

2. Data used

The project will utilize a dataset of images captured under different lighting conditions. Each image is 1000x1000 pixels and features square components of varying clarity. The dataset includes ground truth annotations that indicate the location of the square components (bounding boxes) and their classifications as 'Good' or 'Bad'.
Example of Ground Truth is presented in Figure 1 in this report below.
The ground truth annotations will be used to train and validate the model for both component localization and classification.

3. Methodology Overview:



- **Image Preprocessing:**

  Normalization:

  - Histogram equalization (histeq) will be used to blur the images so that patterns will be visible under a range of light. CLAHE (Contrast-Limited Adaptive Histogram Equalization) will be applied to localize light enhancement for image that has inconsistent illumination, to make the components located in different areas of the image obvious.

  Noise Reduction:

  - Gaussian filtering with 5x5 pixel kernel will be used to flatten the image and remove noise. We'll look at median filtering as an option for edge retention with salt-and-pepper reduction.
  - Will median filter as a backup option to preserve edges even more, and remove salt-and-pepper noise.

- **Pattern Detection:**

  Edge Detection:

- We will use the Canny edge detection algorithm to bring up key changes in intensity, so we can see the edges of the squares.
- Need to fine tune settings like threshold values to get the best edge detection according to initial tests.

Component Localization Using Region Proposal Networks

- We have observed that the components are at the same coordinates in the image. Thus, now keeping the coordinates same we have extracted the individual components and labelled them as good or bad.

Line Detection:

- We will use the Hough Transform to detect straight lines in the binary edge-detected image. This method will help identify potential strip or cross patterns.
- Extract parameters such as angle and distance to further analyze the detected lines and determine if they correspond to the expected pattern structures.

➢ **Feature Extraction:**

Geometric Features:

- Calculate the number of detected lines and their orientations, storing these features in a structured format for analysis.
- Measure the area and dimensions of each component to ensure that they meet expected criteria for classification.

Textural Features:

- We will use Local Binary Patterns (LBP) to capture texture information from the detected patterns, which is crucial for distinguishing between good and bad components.
- Calculate texture contrast and homogeneity metrics to provide additional context for the pattern analysis.

Pattern Characteristics:

- The sharpness of the pattern will be measured by the number of edges that are found and uniformity of the pattern in the bounding box.

➢ **Classification:**

Support Vector Machine (SVM):

- We will use SVM to classify components based on the extracted features. The model handles non-linear boundaries between the classes effectively.
- We are performing stratified sampling and training an SVM. The stratified sampling enables us to randomly test the model by avoiding training on the whole data.
- Train the model on a labeled dataset, employing techniques such as grid search for hyperparameter optimization (e.g., adjusting the C and gamma parameters).

Random Forest:

- We use Random Forest as ensemble learning for stronger trainings and avoid overfitting. There will be several decision trees in the model and the classification will be done by majority of votes in the trees.
- We will use feature importance scores from the Random Forest model to identify which features contribute most significantly to classification, adding model legibility.

➢ **Output Generation:**

Performance Metrics:

- A performance report including accuracy, precision, recall, and mIoU (mean Intersection over Union) for localization will be created. mIoU will count the ratio between predicted and ground truth bounding boxes to calculate the localization accuracy.

.

➢ **Expected Output Example:**

Good Component Output:

- An image with a bounding box around a clearly defined strip or cross pattern, labeled as "Good."

Bad Component Output:

- An image with a bounding box around an unclear or indistinct pattern, labeled as "Bad."

4. Project Steps:

Ground truth:

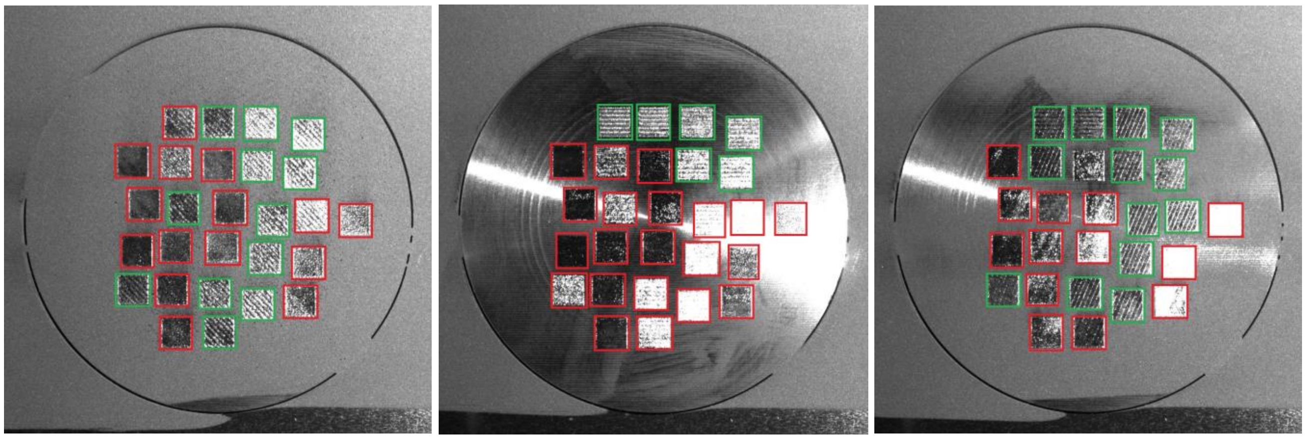


Figure 1, 2 and 3: Green boxes are the good components whereas red boxes are the bad ones.

- **Preprocessing**:

  - Image normalization via **Histogram Equalization** has been performed to handle lighting issues.
  - **CLAHE** has been applied for local contrast enhancement in unevenly lit areas.
  - **Gamma Correction** has been used to brighten images, significantly reducing noise.
  - **Edge detection** is set up for the images, preparing them for pattern analysis.

- **Image Annotation**:

  - 15 images have been annotated with **green bounding boxes** around the good components and **red bounding boxes** around the bad ones to establish ground truth as shown above in Figure 1, 2 and 3.
  - 5 additional images are held aside for testing, ensuring the model's ability to generalize.

- **Labelling for the Images and their Components**:

- The Coordinates_list of the components is like given below:
  [(504, 795, 98, 91), (397, 794, 93, 93), ……. (505, 253, 90, 96)]
- We used the annotated images and ran a python script to extract the coordinates of the components. Now using these coordinates we extracted the components and label them as 1 -> Good and 0 -> Bad.

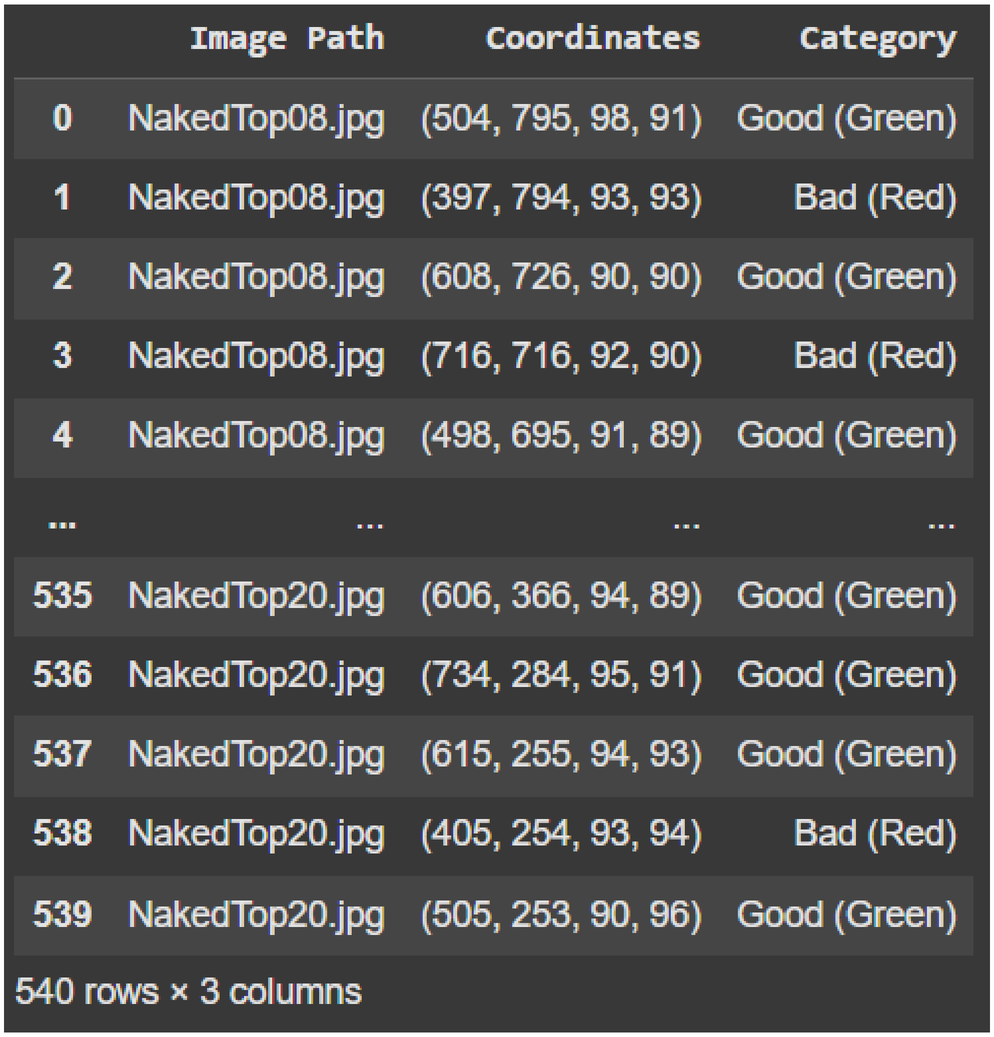


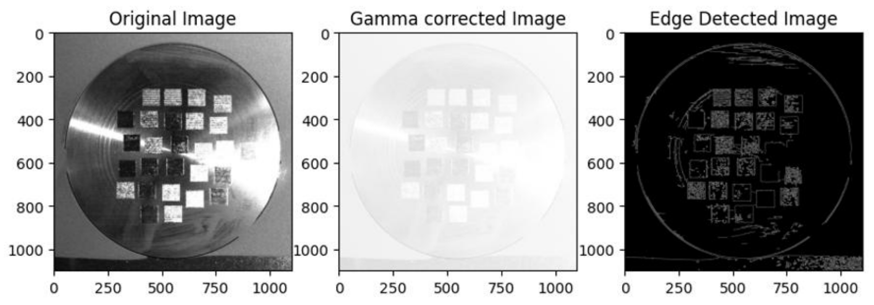
	Image Path	Coordinates	Category
0	NakedTop08.jpg	(504, 795, 98, 91)	Good (Green)
1	NakedTop08.jpg	(397, 794, 93, 93)	Bad (Red)
2	NakedTop08.jpg	(608, 726, 90, 90)	Good (Green)
3	NakedTop08.jpg	(716, 716, 92, 90)	Bad (Red)
4	NakedTop08.jpg	(498, 695, 91, 89)	Good (Green)
...	...	...	...
535	NakedTop20.jpg	(606, 366, 94, 89)	Good (Green)
536	NakedTop20.jpg	(734, 284, 95, 91)	Good (Green)
537	NakedTop20.jpg	(615, 255, 94, 93)	Good (Green)
538	NakedTop20.jpg	(405, 254, 93, 94)	Bad (Red)
539	NakedTop20.jpg	(505, 253, 90, 96)	Good (Green)

540 rows × 3 columns

- **Component Localization**:

- We extract both **geometric** and **textural features**. Here we calculate the **number of lines** detected and **average line orientation**, along with **Local Binary Patterns (LBP)** for texture.
- We are using set coordinates to get the components from. Since the data set has the same image but with varying lighting.

Results after performing Gamma correction and edge detection



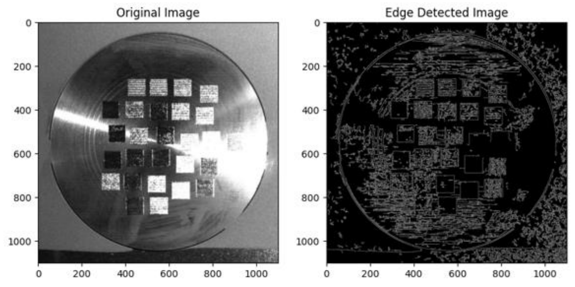
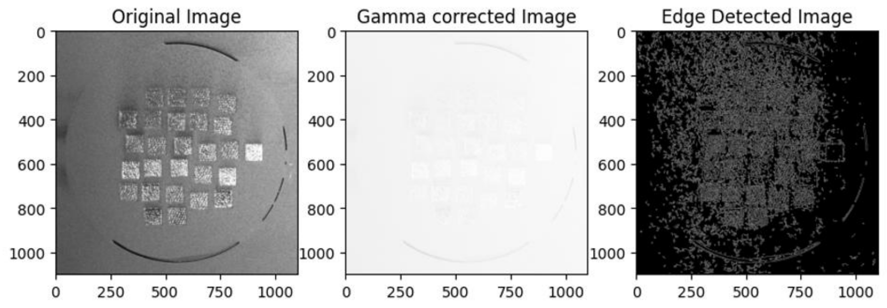


Image = NakedTop01.jpg
Above: Gamma corrected image (factor =20)

Below: Normal image

We can see Significantly less noise in the above image



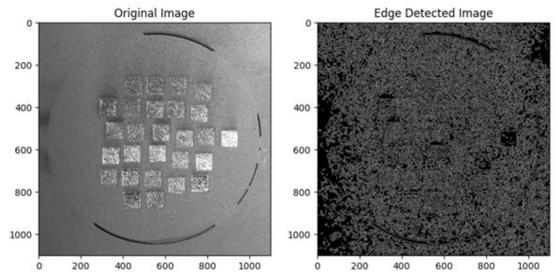


Image = NakedTop10.jpg
Above: Gamma corrected image (factor =20)

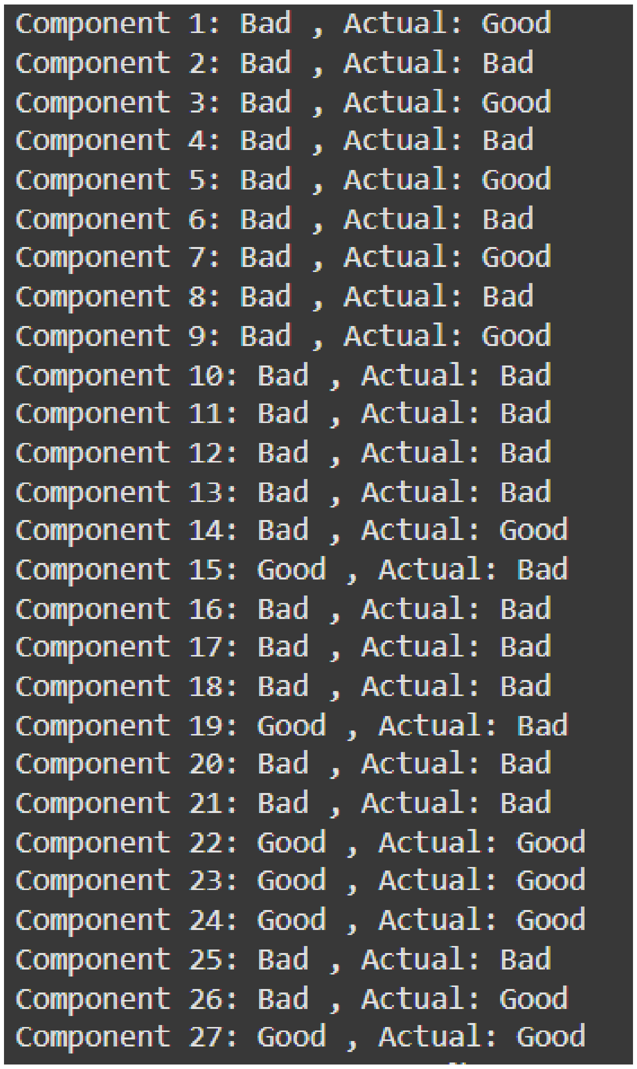
Below: Normal image

We can see Significantly less noise in the above image

- **Model Development:**

- We have reconsidered our approach to avoid the use of pre-trained RPN's.
- Instead we have observed that the components are at the same coordinates in the image. Thus, now keeping the coordinates same we have extracted the individual components and labelled them as good or bad.
- Now using this labelled dataset of the components, we are performing stratified sampling and training an SVM. The stratified sampling enables us to randomly test the model by avoiding training on the whole data.
- We have also followed the same methodology using Random Forests as well.

➢ We have trained the model and the results are discussed below in the Evaluation section. Example of the prediction results below.



```
Component 1: Bad , Actual: Good
Component 2: Bad , Actual: Bad
Component 3: Bad , Actual: Good
Component 4: Bad , Actual: Bad
Component 5: Bad , Actual: Good
Component 6: Bad , Actual: Bad
Component 7: Bad , Actual: Good
Component 8: Bad , Actual: Bad
Component 9: Bad , Actual: Good
Component 10: Bad , Actual: Bad
Component 11: Bad , Actual: Bad
Component 12: Bad , Actual: Bad
Component 13: Bad , Actual: Bad
Component 14: Bad , Actual: Good
Component 15: Good , Actual: Bad
Component 16: Bad , Actual: Bad
Component 17: Bad , Actual: Bad
Component 18: Bad , Actual: Bad
Component 19: Good , Actual: Bad
Component 20: Bad , Actual: Bad
Component 21: Bad , Actual: Bad
Component 22: Good , Actual: Good
Component 23: Good , Actual: Good
Component 24: Good , Actual: Good
Component 25: Bad , Actual: Bad
Component 26: Bad , Actual: Good
Component 27: Good , Actual: Good
```

➢ We then assessed the predicted bounding boxes localization accuracy in relation to the ground truth using Intersection over Union (IoU) and mean Intersection over Union (mIoU). By dividing the area of two bounding boxes' intersection by the size of their union, the IoU metric determines how much overlap there is between them.

- The estimated bounding box's alignment with the ground truth improves with increasing IoU. The average of the IoU scores for each dataset component is known as mIoU.
- It gives a general indication of how well the predicted bounding boxes locate the components. These measurements are essential for assessing how well the system locates and categorizes components, especially in the presence of changing illumination conditions.

5. Challenges Faced

- **Noise Reduction**:
  - Although **gamma correction** helped, some noise still remains in the images, making it difficult to detect patterns and localize components accurately.
- **Component Detection**:
  - We were initially using **Region Proposal Networks (RPN)** to detect the components; they were detecting the image as a whole instead of individual components.
  - We have reconsidered our approach to avoid the use of pre-trained RPN's.
  - Instead, we have observed that the components are at the same coordinates in the image. Thus, now keeping the coordinates same we have extracted the individual components and labelled them as good or bad.
  - Now using this labelled dataset of the components, we are performing stratified sampling and training an SVM. The stratified sampling enables us to randomly test the model by avoiding training on the whole data
- **Edge Detection Fine-Tuning**:
  - The **Canny edge detection** parameters need further refinement to improve the accuracy of detecting the edges of the components, particularly in images with challenging lighting.
- **Model Development:**

- We are encountering issues when we try to use SVM to do Bounding Box Prediction.
- We don't have the computing resources now to do that for the whole dataset. It took 28 mins for us to run it on 3 images.
- We are trying to figure out ways to speedup the training or to use a better method to plot the Bounding Box so we can include the mIoU metric.
- We have also trained a Random Forest similarly.

➢ **Mean Intersection over Union (Mean IoU)**

- **IoU Formula:**
    1. The **IoU** between two bounding boxes AAA (predicted) and BBB (ground truth) is calculated as:

       IoU(A,B) = Area of Overlap/Area of Union

       Where:
        1. **Area of Overlap**: The area where both bounding boxes intersect.
        2. **Area of Union**: The total area covered by both bounding boxes.

- **mIoU (Mean IoU) Formula:**

    1. The **mean IoU (mIoU)** is the average IoU score across all bounding boxes in the image averaged for the whole dataset:

       mIoU = (1/N) * ∑ IoU(Ai,Bi)

       Where NNN is the total number of bounding boxes, and Ai and Bi are the predicted and ground truth bounding boxes, respectively.

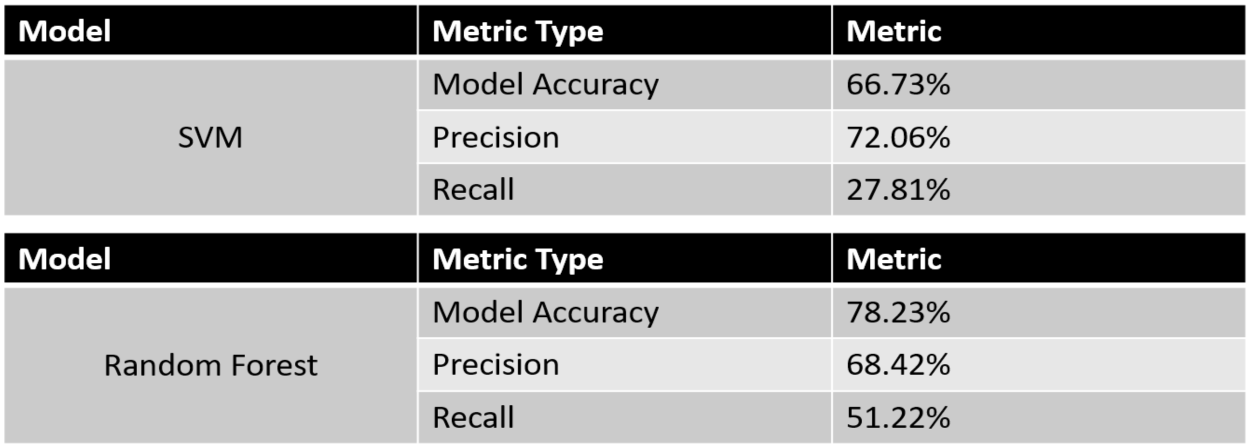
- **Steps for mIoU Calculation:**

    1. **Extract bounding boxes** from the predicted components (bounding boxes generated by contours or another localization technique).
    2. **Get ground truth bounding boxes** (from the dataset).
    3. **Calculate IoU** for each predicted bounding box with its corresponding ground truth.
    4. **Calculate mIoU** by averaging the IoU scores for all components.

6. Evaluation

The evaluation of the system will be based on several metrics, including:

- **Accuracy:** The proportion of correctly classified components.
- **Precision:** The ratio of true positive classifications to the total positive classifications predicted.
- **Recall:** The ratio of true positive classifications to the total actual positives.
- **mIoU (mean Intersection over Union):** This is the value to consider component localization quality. mIoU is the sum of the bounding boxes predicted from the ground truth.



Model	Metric Type	Metric
SVM	Model Accuracy	66.73%
	Precision	72.06%
	Recall	27.81%

Model	Metric Type	Metric
Random Forest	Model Accuracy	78.23%
	Precision	68.42%
	Recall	51.22%

Model	Mean IoU
SVN	0.6807
Random Forest	0.7636

7. Results Discussion and Conclusion:

- Both classifiers performed well, with Random Forest achieving a slightly better mIoU score, indicating improved localization of components. The mIoU score of 0.7636 for Random Forest reflects good localization accuracy, especially in challenging lighting conditions.
- **This means that the Random Forest model effectively segments and make the predicts more in line with the ground truth.**
- The SVM model, while still effective, showed a slightly lower mIoU of 0.6807. This difference in performance is mainly attributed to the Random Forest's ability to handle more complex feature relationships due to its ensemble learning approach.
- We have developed a very basic and rudimentary method to detect and classify components in images under varying lighting conditions, using **image preprocessing** and **traditional machine learning methods.**
- We have performed and understood that Preprocessing is Crucial, Component Localization, trained and evaluated SVM and Random Forest.

- **Future Improvements would Involve:**
    - Further noise reduction.
    - Improving the model architecture or increasing the model training speed.
    - Develop alternative means to integrate Deep Learning methods to better generalize the component detection to a wide range of real word images/ use cases.

8. Timeline

Milestone	**Responsible Member**	**Due Date**	**Status**
Annotate dataset with ground truth	Sai Krishna Meduri	11/10/2024	Completed
Implement image preprocessing and noise reduction	Sai Ram Amaraneni	11/10/2024	Completed
Conduct component localization using RPNs	Ram Gopal Anne	11/13/2024	Completed
Conduct pattern detection using edge detection	Ram Gopal Anne	11/13/2024	Completed
Extract features from detected patterns	Sai Krishna Meduri	11/16/2024	Completed
Train classifiers (SVM, Random Forest) on features	Sai Ram Amaraneni	11/19/2024	Completed
Evaluate classifiers using cross-validation methods	Ram Gopal/ Sai Krishna	11/23/2024	Completed
Evaluate localization using mIoU metric	Nagasai Mandalapu	11/23/2024	Completed
Analyze results and prepare outputs (visuals and metrics)	Nagasai Mandalapu	11/28/2024	Completed
Compile the final report and presentation preparation	Team	12/02/2024	Completed

9. References

1. Gonzalez, R. C., & Woods, R. E. (2018). *Digital Image Processing* (4th ed.). Pearson.
2. Duda, R. O., Hart, P. E., & Stork, D. G. (2001). *Pattern Classification* (2nd ed.). Wiley-Interscience.