Indian Institute of Science Education and Research Bhopal

Computer Vision(DSE-312/EECS-320)

Assignment-2

Name: Aditya Mishra
Roll No.: 21013
Date of submission: October 16, 2024 Marks Obtained:

Please follow the instructions given in the assignment carefully.

Please provide your detailed answers and any explanations or diagrams directly below each question in the 'Answers' section.

1. Implement a face detection algorithm from scratch using Haar-like features and Integral Image computation. ***(Marks: 1+2+7)***

   - Capture an image of yourself using a webcam or upload a face image.
   - Compute the Integral Image to efficiently calculate pixel sums over rectangular regions. Use integral image to detect face using Haar features.

   Answer: We capture the image using webcam for face detection. The image is given by Figure 1.

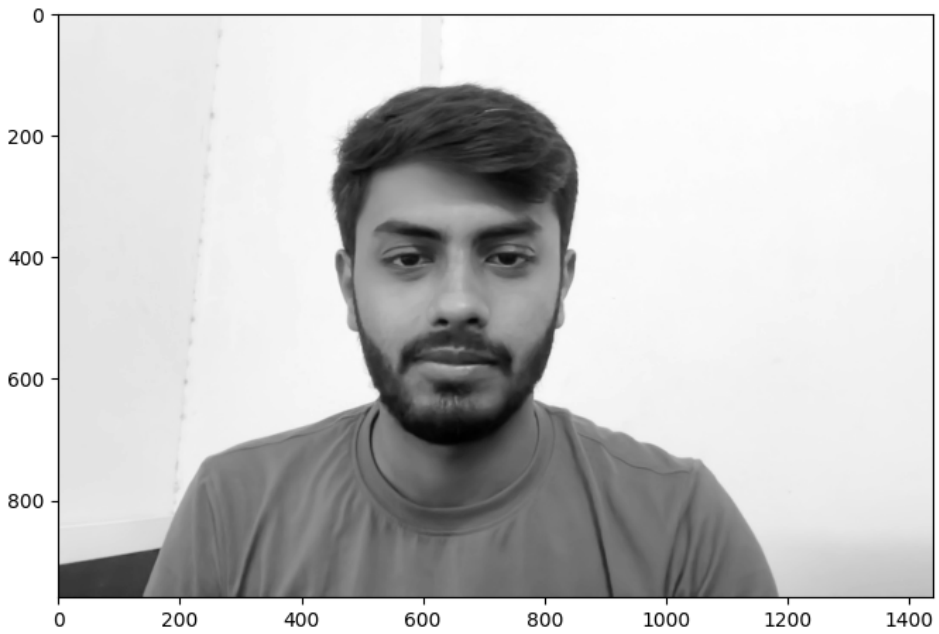

Figure 1: The image captured by laptop's webcam that will be used for question 1.

Integral image is defined as a table that holds the sum of all pixel values to the left and top of a given pixel, inclusive. Mathematically, it is given by equation 1.

$$I(x,y) = \sum_{i=0}^{x} \sum_{j=0}^{y} A(i,j) \tag{1}$$

Where:

- $I(x,y)$ is the value of the integral image at the coordinates (x,y).
- $A(i,j)$ is the pixel value in the original image at the coordinates (i,j).
- The summation includes all pixel values from the top-left corner $(0,0)$ to the position (x,y).

Figure 2 depicts the integral image along with the original image calculated using the equation 1 from scratch and not without using any existing library.

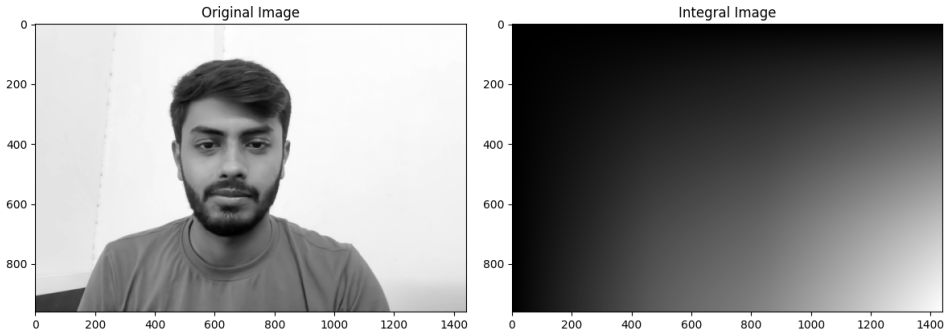

Figure 2: Integral Image

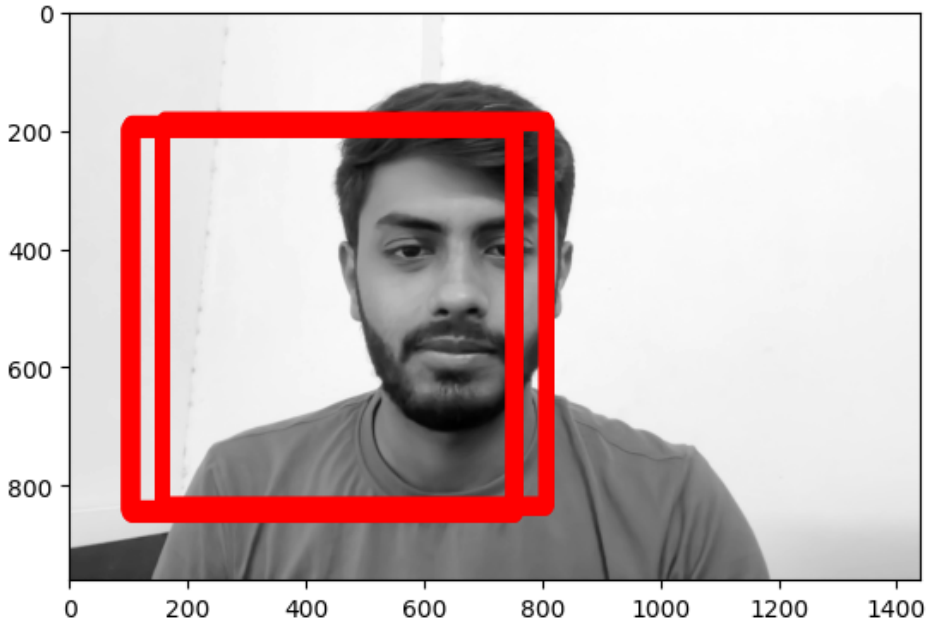

Figure 3: Face detection using integral image by Haar features.

We use Haar features for the face detection. The functions *haar_vertical_feature*, *haar_feature* and *composite_haar_feature* are used to detect the face. The *composite_haar_feature* creates a face like structure of Haar filters. The *detect_faces* function uses the above Haar features to detect the face and finally the function

visualize_detections plots bounding boxes in the original image. The code can be found on the **GitHub**. Figure 3 depicts the face detection using integral image by Haar features.

2. Using dataset **link**, implement a face anti-spoofing model that performs classification based on the below different feature extraction methods to identify fake (spoof) and real image. Compare and analyze your results using metrics accuracy, f1-score, precision, recall and confusion matrix. Document the findings and discuss the failure and success of each method. (*Note: You have to use only 1000 images and not the whole dataset*) ***(Marks: 3+3+4)***
   (a) Using the raw pixel values of the face images as features. Train a Support Vector Machine (SVM) classifier on these raw pixel features to perform face recognition. Evaluate and analyze the performance of the model on the dataset.
   (b) Extract Local Binary Patterns (LBP) features from the face images for feature extraction. Train an SVM classifier using the LBP features to perform face recognition.
   (c) Compute edge images using any two edge detectors (canny, sobel, prewitt, etc.), then use them as input features independently to train an SVM classifier and perform classification.

Answer: We use the given dataset to implement a face anti-spoofing model that identifies fake and real images. We resize every image to the dimension of 64×64 in order to have uniformity across the dataset. We also convert the color images to gray scale. We split the data in 80:20 ratio for training and testing.

(a) A Support Vector Machine (SVM) is a supervised machine learning algorithm that works by finding the optimal hyperplane to separates data points of different classes in a high-dimensional space, maximizing the margin between the closest points (support vectors) of each class. The margin is given by equation 2:

$$\gamma = \frac{2}{\|w\|} \tag{2}$$

Where:

- γ is the margin.
- w is the weight vector that defines the orientation of the hyperplane.
- $\|w\|$ is the Euclidean norm (magnitude) of the weight vector w.

Raw pixel values refer to the individual intensity values of pixels in an image, typically represented in grayscale or color formats. In grayscale images, each pixel is represented by a single value indicating its brightness (0 for black and 255 for white). In color images, pixels are often represented by three values corresponding to the intensities of red, green, and blue (RGB). These raw values can be directly used as features for image processing tasks, such as classification and recognition.

Table 1: Classification Report

Class	Precision	Recall	F1-Score	Support
0	0.56	0.53	0.55	218
1	0.50	0.53	0.51	191
Accuracy		0.53		
Macro Avg	0.53	0.53	0.53	409
Weighted Avg	0.53	0.53	0.53	409

Table 2: Confusion Matrix

Actual / Predicted	0	1
0	116	102
1	90	101

Table 1 and 2 demonstrates the classification report and confusion matrix, respectively, using raw pixel values.

Analysis: The model using raw pixel values produced moderate performance metrics with an overall accuracy of 53%. The precision and recall values indicate that while the classifier could somewhat differentiate between real and spoof images, it struggled to achieve high accuracy, leading to a high false positive rate (102 misclassifications of real images as spoof).

Reasons for Success: Direct Feature Use - Utilizing raw pixel values allows the model to leverage all available image data, potentially capturing subtle variations in real and spoof images.

Reasons for Failure: High Dimensionality - The model may have suffered from the *curse of dimensionality*, where the high-dimensional feature space could lead to overfitting, particularly with limited training data.

Lack of Contextual Information - Raw pixel values do not inherently capture texture or structural features that may distinguish real from spoof images, leading to suboptimal classification performance.

(b) Local Binary Patterns (LBP) is a texture descriptor that captures the local spatial patterns and texture information of an image. Mathematically:

$$LBP(x, y) = \sum_{p=0}^{P-1} s(I_p - I_c) \cdot 2^p \tag{3}$$

where:

- I_c is the intensity of the center pixel,
- I_p are the intensities of the neighboring pixels,
- P is the number of neighboring pixels,

- s is the step function defined as:

$$s(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}$$

Table 3 and 4 shows the classification report and confusion matrix, respectively, for the face detection using trained SVM on LBP features.

Table 3: Classification Report

Class	**Precision**	**Recall**	**F1-Score**	**Support**
0	0.53	1.0	0.70	218
1	0.0	0.0	0.0	191
Accuracy			0.53	
Macro Avg	0.27	0.50	0.35	409
Weighted Avg	0.28	0.53	0.37	409

Table 4: Confusion Matrix

Actual / Predicted	**0**	**1**
0	218	0
1	191	0

Analysis: The LBP approach demonstrated a stark contrast in performance, achieving perfect recall for the real class but failing to detect any spoof images (precision of 0.00). The overall accuracy remained at 53%, primarily driven by the accurate classification of real images.

Reasons for Success: Texture Sensitivity - LBP is effective for capturing local texture information, which may be more consistent in real images compared to spoof images.

Reasons for Failure: Poor Detection of Spoof Images - The method likely lacks the necessary discriminative power to differentiate between spoof and real images, leading to a complete failure in classifying any spoof images correctly.

(c) We use Sobel and Prewitt filters to compute the edge images and then use them as independent input features to train the SVM classifier. We directly use the function of Sobel filter from Assignment-1 and create the function for Prewitt filter from scratch. The results are shown in Table 5 and 6.

Table 5: Classification Report

Class	Precision	Recall	F1-Score	Support
0	0.57	0.62	0.59	218
1	0.51	0.46	0.48	191
Accuracy		0.55		
Macro Avg	0.54	0.54	0.54	409
Weighted Avg	0.54	0.55	0.54	409

Table 6: Confusion Matrix

Actual / Predicted	0	1
0	136	82
1	104	87

Analysis: The edge detection methods, using Sobel and Prewitt filters, produced an accuracy of 55%. This indicates a modest improvement over previous methods. The confusion matrix indicates that the model identified 136 real images correctly but misclassified 82 real images as spoof. Similarly, it correctly identified 87 spoof images while misclassifying 104 spoof images as real.

Reasons for Success: Utilization of Structural Features - Edge detection methods, such as Sobel and Prewitt, effectively highlight the edges and contours in images. This characteristic is useful for differentiating between the structural elements of real and spoof images, leading to better recognition capabilities compared to raw pixel values and LBP.

Reasons for Failure: High False Positive Rate for Spoof Images - The model's failure to correctly classify 104 spoof images as real indicates a need for improved detection capabilities. The edge-based features may not fully capture the nuances that differentiate spoof images (such as printed photos or digital screens) from real ones.

Limited Features for Spoof Detection - While edge detectors can highlight shapes and transitions, they may not adequately represent other important characteristics specific to face images. This limitation could lead to confusion in the classifier, resulting in misclassifications.