Report on Custom Dataset

Under the Guidance of :- ***Prof. Bhupesh Daheria***

Date: 12/05/2025

1. Introduction

This report outlines the creation and characteristics of a custom image datasets consisting combined images and also single image dataset which is labeled using LabelImg,online platforms like roboflow , cropping methods using haar cascade ,cropping via MTCNN and also used .txt yolo annotations. The report aims to find how different datasets were created along with what preprocessing is done for images

2. Dataset Description

2.**1. Dataset Creation Steps**

Images were taken from cellphone camera from different angles on different backgrounds and with various angles .Also previous images from the test subjects is taken to improve dataset quality and no. of images.

- **Total Images Collected:**
  - **First Attempt:** 537 images
  - **Second Attempt:** 614 images (added 77 new images)
  - **Third Dataset:** 842 images (for 5 test subjects combined)
  - Fourth Dataset : 105 images (separate for each person)

2.2. Image Preprocessing:

Filtering Bad Images:-

We removed:

- Blurry images (e.g., camera shake).
- Images with no faces detected.
- Poor lighting/angled faces.

Our next step was to remove all the blurred images , repetitive images ,the images where detection or labeling can be hard .We used coding methods for preprocessing the code and also checked manually. In case of resolution size we know that yolo have built in cropping methods too but it takes 640*640 image size ,so if we have 1280*1080 image size then it will be converting to 640*480 and then gray background texture will be added to make it 640*640.

3. Methods used

3.1. Using Manual Labelling:

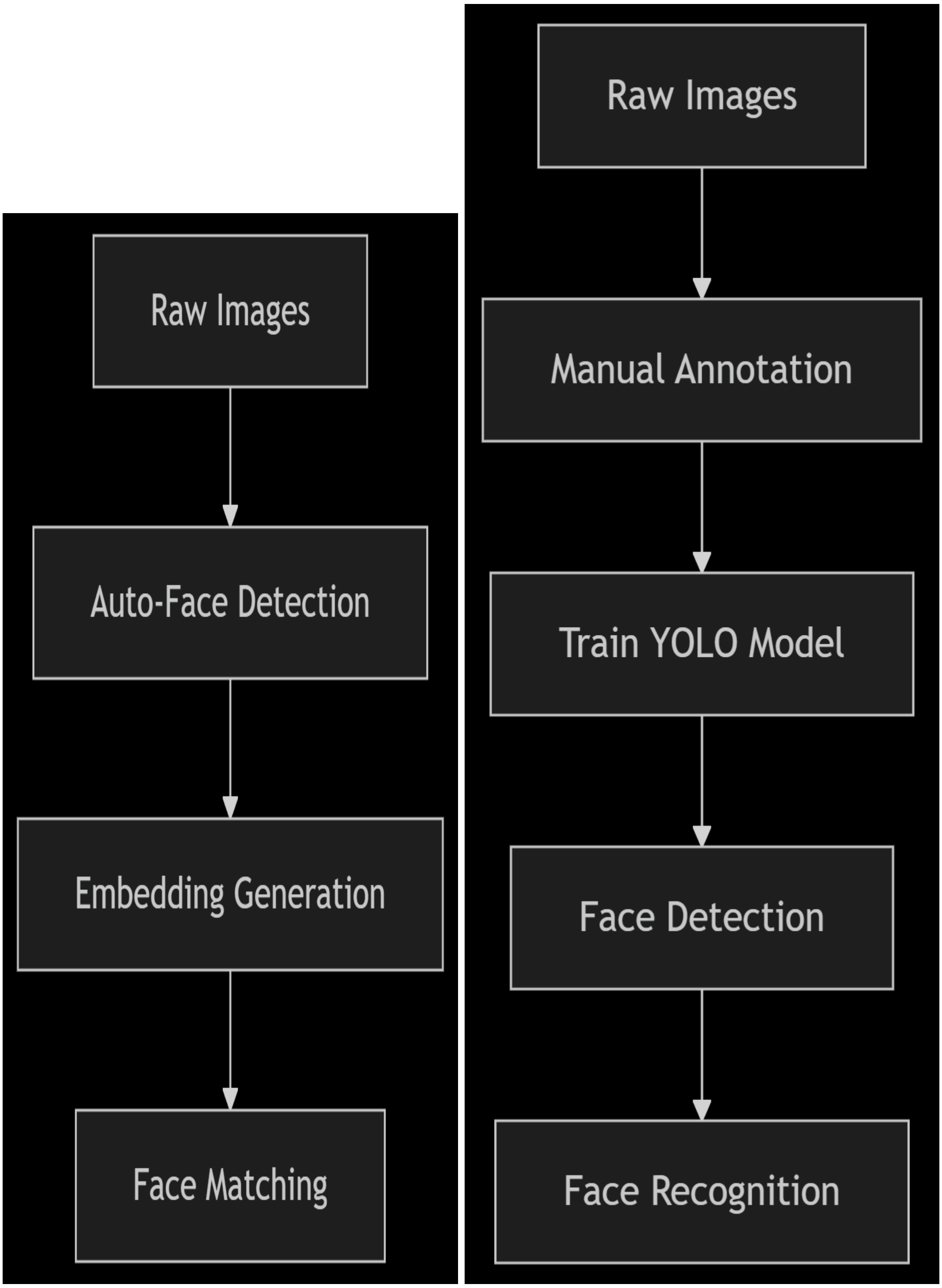
We   used Labelimg for manual labelling .We tried to install directly through the terminal but it was crashing again and again ,then finally we went into the github repository and downloaded it manually to use labelimg.

3.2 Cropping Faces:-

We have tried running cropping methods even on labeled images which were labelled by Labelimg as it will save time but it was not a good approach .

1. **Haar Cascade** (basic face detection)
2. **MTCNN** (advanced face detection)

Image Format: .jpg / .png / .jpeg
Annotation Format: YOLO TXT
Image Sources: 36% combined team members, 64% single image of a person.
Image Resolution: Ranges from 640x480 to 1920×1080


Raw Images
Auto-Face Detection
Embedding Generation
Face Matching
Raw Images
Manual Annotation
Train YOLO Model
Face Detection
Face Recognition

4. Labelimg v/s Cropping:-

The choice between YOLO-style annotations and direct image feeding with pre-trained embeddings depends on your specific use case, resources, and performance requirements. Here's a detailed comparison:

4.1. YOLO-Style Annotations (Manual Labeling)

What It Means:

- Draw bounding boxes around faces in images
- Label each face with person's identity
- Generate annotation files (e.g., .txt or .xml)

4.11When to Use:

✅ Multi-Face Images:

- If your training images contain multiple people
- Need to distinguish target faces from others

✅ Complex Backgrounds:

- Images with cluttered backgrounds
- Requires precise face localization

✅ Custom Detection Needs:

- If you need to train a specialized face detector
- Unusual face angles/lighting conditions

Pros:

- Full control over detection parameters
- Better handling of complex scenarios
- Enables custom model training

Cons:

- Time-consuming annotation process (~5 mins/image)
- Requires annotation tools (LabelImg, CVAT)
- Needs retraining when adding new people

4.2. Direct Image Feeding with Pre-Trained Embeddings

What It Means:

- Use pre-cropped face images
- Let models like Dlib/FaceNet handle detection & embedding
- No manual bounding box labeling

4.21 When to Use:

✅ Single-Face Images:

- Each image contains one centered face
- Clean backgrounds (e.g., ID photos)

✅ Rapid Deployment:

- Quick setup without annotation
- Easy to add new people

✅ Limited Resources:

- No budget/time for manual labeling
- Using existing face recognition APIs

Pros:

- No annotation effort required
- Faster implementation
- Leverages state-of-the-art pre-trained models

Cons:

- Limited control over detection
- May struggle with complex images
- Dependent on model's generalization

5. Problems We Faced

5.1 LabelImg Issues

- Tool Crashes: LabelImg crashed frequently when labeling 100+ images.
- Time-Consuming: Manually drawing boxes took ~2 minutes per image.

5.2 Why We Tried Cropping

- Faster: Auto-cropping processed 50 images/minute vs. manual labeling (5 images/minute).
- Fewer Images Needed: Worked with just 5–10 images per person.

6. Final Results

➢ Successful Cropped Faces: 780/800 images (97.5% accuracy).

➢ Best Tool: MTCNN detected faces better than Haar Cascade.

➢ Dataset Used: Cropped faces + YOLO annotations for training.

5. Recommendation for Attendance Systems

For employee attendance tracking, the direct image feeding approach is generally better because:

➢ Practical Considerations:

➢ Employee photos are usually single-face and well-cropped

➢ Pre-trained models achieve 99.38% accuracy on LFW dataset

All datasets can be found here :- [Google Drive link]

Note: The highlighted words above opens up the link to the code used.