Checkpoint 3

Rida Tirmizi Fatima Raza Ushna Jalil

Abstract— This project presents a facial recognition system developed using interpretable machine learning techniques. Three feature extraction techniques were used to build the facial recognition system. Those techniques were Scale-Invariant Feature Transform (SIFT), Local Binary Patterns (LBP), and Histogram of Oriented Gradients (HoG). Moreover, we also used Decision Trees, and Logistic Regression to achieve more accuracy. We used Tkinter-based GUI was used to take input via webcam, which makes it practically applicable.

Special importance was given to interpretability during system design, particularly with employing decision trees and feature importance visualization.

This results in a clear, modular approach that provides accurate, precise, and comprehensible results for any number of people.

I. Introduction

Facial recognition has become extremely important in many fields, such as, security, identity verification, surveillance, etc. As the systems are growing, they have become more complex, leading to a better trade-off between the role of accuracy and interpretability. Building interpretable models has become an integral part as the need of transparency grows while dealing with sensitive domains like security and privacy. The main objective of this project was to create a facial recognition system that prioritizes transparency in decision making using interpretable machine learning models.

II. System Design

We used a dataset containing images of 11 people, each stored in different sub folders. Images were extracted from video frames, which were then greyscaled. OpenCV's Haar Cascade classifier was used to detect and crop facial regions, to retain images with containing exactly one face. Then these images are cropped and resized to 128x128 pixels to make the data consistent.

Feature extraction was done using three methods, i.e. Histogram of Gradients (HoG), Local Binary Patterns (LBP), Scale-Invariant Feature Transform (SIFT). HoG captures edge and gradient information effectively identifying facial features. LBP compares each pixel of the image with its eight neighbors within a radius of one to extract texture patterns. SIFT highlights keypoints invariant to scale and rotation. Using these features, we are able to balance robustness and interpretability.

Classification is done by using Logistic Regression and Decision Trees. The system also uses a Tkinter-based which enables users to interact with the facial recognition system, and give the input using a webcam. The whole pipelining of the project was done in python using relevant libraries.

III. Computational Thinking Application

Principles of computational thinking were applied while building this system. This was to make sure a structural and logic-based approach.

The first principle, "decomposition", was implemented during face detection, preprocessing, model training, feature extraction, classification, and interpretability analysis. All of the mentioned modules were handled separately before integrating in the main pipeline. Apart from this, pattern recognition was also utilized for the purpose of understanding and utilizing the facial features in the dataset.

HoG was used to identify gradient patterns representing facial contours and LBP was used for capturing local texture patterns. SIFT was a very useful tool in identifying keypoints of the scale and rotation. This contributed to a good feature set.

Abstraction also played an essential role in simplifying the dataset by removing the noise from the data, such as background. It also helped us focus the facial regions that were essential in identifying the person. This helped us in training our model effectively by abstracting raw images into meaningful descriptors using all these tools, while retaining critical facial structure information.

Algorithmic thinking was used in the process of developing procedures for detecting faces, extracting necessary features, and training classifiers. The implementation of all this includes looping through each folder in the dataset that represents an individual while extracting the features and labeling them. This process helped us in the automation of pipelines and prepared it for scaling up for larger datasets.

IV. Implementation Details

Python and other libraries of python were used to develop this system. OpenCV was used to detect faces, and for the preprocessing of the images. The libraries, for example NumPy, Matplotlib, and Seaborn, were used for numerical computations and visualizations. On the other hand, Scikit-learn is used for logistic regression, and decision trees, and sikit-image is used for HoG, and LBP feature extraction. Haar Cascade was used for face detection, and it also helped to locate and crop the facial regions. In order to maintain the integrity, the images that had multiple faces or no face were eliminated. All the valid images from the dataset were then converted to grayscale.

The dataset is split into 80% training and 20% testing data. Label encoding assigned each individual a numerical value. Moreover, a tkinter-based GUI is developed that captures live video input, and uses the trained model to do the face recognition.

V. Results and Analysis

Logistic Regression and Decision Trees are crucial to locate individuals in datasets with varying number of persons. Logistic Regression provides high accuracy, and its coefficients help us understand which features, HoG, LBP, or SIFT, have an impact on the predictions. Moreover, decision trees improved interpretability. Although they are prone to overfitting in high dimensional spaces without pruning, they enhance interpretability by providing visual paths of classification decisions.

EDA helped us in evaluating feature extraction with consistent patterns in HoG and LBP histograms and ensured a balanced class distribution. Corrupted images were also identified and removed. All these features, HoG, LBP, and SIFT contribute to accurate identification. Although feature extraction takes a lot of time to process, the approach is computationally efficient.

VI. Conclusion and Future Work

The person identification system successfully identifies individuals from the dataset. We combined HoG, LBP, and SIFT into a unified feature representation and we evaluated multiple classifiers. This enabled us to achieve a balance between accuracy and interpretability.

Logistic Regression and Decision Trees offer transparent and efficient classification. While, decision trees use simple rules to explain prediction logistic regression provides consistent performance with clear feature contributions.

Finally, we've used tkinter-based GUI that supports real time testing, making the system practically applicable.

VII. References

[1] Dalal, N., & Triggs, B. (2005). Histograms of Oriented Gradients for Human Detection. In CVPR.

[2] Ojala, T., Pietikainen, M., & Harwood, D. (1996). A comparative study of texture measures with classification based on feature distributions. Pattern Recognition.

[3] Lowe, D.G. (2004). Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision.

[4] Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In CVPR.

[5] Schroff, F., Kalenichenko, D., & Philbin, J. (2015). FaceNet: A Unified Embedding for Face Recognition and Clustering. In IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).

[6] Molnar, C. Interpretable Machine Learning. Retrieved from https://christophm.github.io/interpretable-ml-book/

VIII. Contribution

Rida was responsible for writing and testing the core code which includes model training of the recognition system, data preprocessing, feature extraction, and visualization. Rida also worked on enhancing the GUI and EDA. Fatima did implemented the EDA, and working on the GUI. Ushna worked on research, testing, and writing detailed reports. Initially, each of us worked independently to gain a thorough understanding of the concepts needed for this project.