# Explainable Real-Time Facial Recognition System with Web Backend

Reham Hafeez, Hiba Fatima, Kabeer Jaffri
*Department of Computer Science, Institute of Business Administration Karachi*
{r.hafeez.30574, h.fatima.29005, k.jaffri.29027}@khi.iba.edu.pk

***Abstract*—Machine learning's power is often matched by its transparency. Hence, to ensure the systematic interpretability within this paper, we have the written foundation detailing the creation of a real-time facial recognition system. To sidestep the "black box" problem, a responsive client-server architecture was engineered, where communication over WebSockets advances real-time performance whilst ensuring the system engine at its core remains transparent. We deliberately constructed the backend pipeline from foundational algorithms to demystify the format of identification. For each face, a rich "digital fingerprint" is constructed by weaving the form of the Histogram of Oriented Gradients (HOG) with the texture analysis of Local Binary Patterns (LBP). Our choice of a Decision Tree model was central, its inherent flowchart-like structure allows us to genuinely peer inside and understand its reasoning. The final system not only achieves a functional accuracy of 91.0% but, more importantly, incorporates a confidence-based mechanism to intelligently handle faces it has never seen before. The result is a system that is both effective and demonstrable while also being understandable.**

***Index Terms*—Explainable AI (XAI), Interpretability, Facial Recognition, Decision Tree, Histogram of Oriented Gradients (HOG), Local Binary Patterns (LBP), Real-Time Systems, Computational Thinking.**

## I. INTRODUCTION

Growing advancements in technology and the resulting popularity of devices created the requirement for a secure method of data accessibility and security due to the integral data stored on personal devices. Simultaneously providing convenient and incredibly secure access to personal devices led to the creation of facial recognition.

However, as machine learning models driving it have achieved accelerated growth to the point of reaching near-human levels of performance, they have also become significantly complex. Which in turn, has given rise to the challenge: lack of clarity. When high-stakes errors and unforeseen bias leave the users with no explanation, it forces us to ask if we can fully trust this powerful tool.

To confront this dilemma, we changed the question from "Can we build a system that recognizes people ?" to "Can we build a system that can also explain how it reaches its conclusions?" This, while a minor change, demanded a fundamental shift in our philosophy. We made the conscious choice to upload the level of interpretability, despite the consequences and trade-off with the raw predictive power offered by more opaque methods.

To chart this course, we returned to the foundational pillars of computer vision. We felt it was important to stand on the shoulders of giants, drawing inspiration from the elegant efficiency of the Viola-Jones detection framework and the descriptive prowess of texture and shape descriptors like Local Binary Patterns (LBP) and Histogram of Oriented Gradients (HOG). For our classifier, the very "brain" of our operation, we selected the humble yet powerful Decision Tree. Its logic isn't buried in a web of millions of weighted parameters but is instead expressed as an intuitive flowchart—a series of simple questions that a person can follow, understand, and critique.

This paper, therefore, tells the story of that undertaking. It is a narrative of deliberate design choices, of challenges uncovered and solved, and of the algorithmic thinking that shaped our final system—a system built not just to be used, but to be understood.

## II. PROBLEM UNDERSTANDING

Inorder to accurately maintain and build the underlying foundation of a system meant to work on a series of practical and real-world hurdles, required us to build a blueprint before even beginning to code.

### *A. The Chaos of Real-World Conditions:*

The difference in the unpredictability of everyday life and a laboratory setting causes it to be a poor substitute. Therefore, we had to implement changes in our systems to ensure that any live video feed—being a dynamic and imperfect source of data, subject to fluctuating ambient light, subtle but important shifts in pose, and varying distances from the camera—would be able to handle it via innate robustness without failing.

### *B. The Mandate for Data Consistency:*

Machine learning models are creatures of habit; they thrive on consistency. We quickly realized that if we just threw random images at the model - all different sizes, qualities, and orientations - it would perform terribly. It became obvious early on that we couldn't skip proper preprocessing and therefore we needed solid, automated cleanup before anything else would work.

### *C. The Need for an Instantaneous Response:*

For any system that works directly with people, speed is everything. That tiny delay between when someone's face shows up on screen and when their name pops up? It had to feel instant - like magic, not technology. This constraint immediately ruled out any naive approach involving the transfer

of raw, high-resolution video streams over the network. This very challenge was the primary catalyst for our split client-server architecture, a design purposefully crafted to minimise network load and centralise the heavy computational work on a dedicated backend.

## III. LITERATURE REVIEW

The methodology in this case is defined by a deliberate selection of well-established, comprehensible techniques, representing a conscious step away from the current ”black box” deep learning trend and a return to a classical, explainable pipeline, encouraging understanding which aligns with the pedagogical goals of the project.

### *A. LBP Histograms: Reading the Texture of a Face*

The Local Binary Patterns (LBP) algorithm introduced by Ojala et al. is an efficient method to analyse the description of texture. Its functioning relies on utilising every pixel in an image, analysing its immediate neighbours, and constructing a binary number based on the varying brightness of its neighbours. A histogram of these binary codes is then tallied across the entire face, creating a compact yet descriptive ”texture fingerprint.” Given our project's constraints on computational resources, LBP's speed and effectiveness made it an ideal candidate for one-half of the feature descriptor.

### *B. The Viola-Jones Algorithm: Finding a Face in the Crowd*

The Viola-Jones framework is a moment in the history of real-time computer vision. Its genius lies not in its complexity, but in its clever efficiency. Instead of exhaustively analysing every pixel of an image, it employs several increasingly complex classifiers. This structure enables the algorithm to quickly disregard regions of an image that are obviously not a face, ensuring that it only has to spend its computational budget on the most relevant regions. This structure allows classifications which enable it to disregard regions of an image that are evidently not a face. Understanding this logic was vital to increasing the efficiency during the initial face detection stage.

### *C. HOG: Describing a Face by its Shape*

The Histogram of Oriented Gradients (HOG) descriptor excels at capturing shape. Popularised by Dalal and Triggs for pedestrian detection, the technique is aptly suited for faces as well. It functions by analysing the orientation of edges and intensity gradients. Practically, this means it learns to describe the structural lines of a face—the curve of the jaw, the sharp angle of the eyebrows, the contour of the nose. By creating a hybrid feature vector that combines the structural information from HOG with the textural information from LBP, we are able to provide the classifier with a description of each face that is far more detailed, nuanced and robust than either technique can offered independently.

### *D. The Road Not Taken: A Deliberate Choice Against Deep Learning*

For this project, explicit consideration over using state-of-the-art deep learning models like FaceNet or VGG-Face took place and the idea was rejected. Though their accuracy on large-scale benchmarks is undeniable since these models learn to project faces into a complex mathematical ”embedding” space, where the distance between two points directly corresponds to facial similarity, this path came with two elements that are at odds with our project's core mission. Firstly, the computational overhead for real-time inference is immense, typically requiring specialised GPU hardware that is beyond our scope. Secondly, and more importantly, the process is opaque due to the fact that an embedding is an abstract vector of numbers which offers no clear or intuitive explanation as to the specific facial features that lead the model to its final decision. This inherent lack of transparency was in direct opposition to the philosophy of explainability driving our work.

## IV. DATASET DESCRIPTION AND EDA

To ensure that the model is functioning at its optimum capacity, the data it learns from needed to be evaluated, which is why the first practical step was to collect, inspect, and deeply understand the raw material that forms the foundation of our system.

### *A. Custom Dataset Collection*

We created a dataset composed of images from the ten members of our group, with each person assigned a label from ’01’ to ’10’. Each subject provided 50 photographs, in order to form a focused dataset of 500 images total. This process was conducted with explicit consent from all participants, adhering to the ethical responsibilities of handling personal biometric data.

### *B. Exploratory Data Analysis (EDA)*

Our EDA went surpasses simple file checks to also investigate the nuanced characteristics of our data.

- **Data Integrity and Consistency Checks:** We began with programmatic audits of image formats and dimensions, ensuring a uniform starting point for the data processing pipeline.
- **EXIF Orientation Correction:** Investigation revealed a critical data integrity issue. The images for subject ’10’ were consistently being loaded with an incorrect 90-degree rotation, leading to the realisation that an EXIF orientation tag set by the camera, a piece of metadata that the standard OpenCV loading function was simply ignoring. To correct this, we created a precise solution so that the data loading script now checks if the subject being processed is ’10’ and, upon determination, uses the Python Imaging Library (PIL) to correctly interpret the EXIF tag and apply the necessary rotation. This hands-on debugging is an invaluable lesson in being meticulous and attentive during these processes.

- **Noise Estimation and Smoothing:** Using functions defined in utilities.py, minor variations in digital noise across the dataset were observed. To mitigate this, a noise estimation function was applied, followed by a thresholded smoothing filter. This pre-processing step ensured that our feature extraction algorithms received clean, normalised input, improving their reliability.

## V. METHODOLOGY

The methodology details the structured, step-by-step process we followed to build and train our system.

### A. System Architecture and Real-Time Processing

This system is founded on a client-server model engineered for real-time dialogue (see Fig. 1). The client, a Python application, is responsible for accessing the webcam via OpenCV. It is designed to be lightweight and its responsibility is to capture frames, encode them into an efficient JPEG format, convert them to a base64 string, and transmit them over a WebSocket to the server. The backend, built asynchronously with Python's websockets library, is the computational crux of the system, designed to efficiently manage the entire recognition pipeline.

### B. Data Loading and Preparation Job

The first step in our training pipeline, as defined in training.py, is the meticulous loading and preparation of data. This stage is responsible for iterating through the dataset folders, loading each image, and applying necessary corrections. As discovered during the EDA, this includes the crucial conditional logic to handle EXIF-based rotation for subject '10'. Additionally, it applies noise reduction and smoothing filters to all images, ensuring that only clean, consistently oriented data is passed to the following stages.

### C. Feature Extraction Techniques

The true heart of our recognition process resides within the FeatureExtractor class. After an image has been prepared and a face detected, this class is responsible for creating its unique digital fingerprint. It performs two distinct analyses in parallel:

- **Histogram of Oriented Gradients (HOG):** This technique maps the structure and shape of the face through analysis of the orientation of intensity gradients. Moreover, it is particularly proficient at capturing the contours of key facial features including the jawline, nose, and eyebrows.
- **Local Binary Patterns (LBP):** This method excels at describing texture by creating a compact summary of the face's micro-textures by comparing the intensity of each pixel to its neighbours.

The calculate() method within our extractor intelligently concatenates these two different descriptions into a single, comprehensive feature vector, providing our classifier with a rich, multi-faceted view of the individual.

### D. Model Training and Selection

The choice of a Decision Tree Classifier was a direct commitment to interpretability. However, it was not trained blindly. The training process was methodical and data-driven:

- **Hyperparameter Tuning:** A key part of the methodology was a systematic search for the model's "sweet spot." We developed and ran a dedicated script to train and test the model across a wide range of max_depth values (from 1 to 20). By plotting the resulting test accuracy, the optimal depth, a value of 5, could be visually identified, providing the best generalisation without overfitting.
- **Fitting the Model:** Using the optimised hyperparameters, we trained the Decision Tree on our full training set of 400 feature vectors.
- **Model Serialisation:** The final, tuned classifier object was saved to a file (decision_tree_model.pkl) using joblib, making it a persistent, and reusable asset for the main server application.

### E. Real-Time Integration and Confidence Thresholding

With a trained model in hand, the final piece was integrating it into the live server (main.py) and addressing the critical "unknown person" problem. To prevent the model from assigning a known label to an unknown face, a confidence threshold was implemented. Now, when the server receives a frame and generates a feature vector, it passes it to the model's predict_proba() method, which returns a probability for every person the model knows, then the server performs one final check by identifying the highest probability in this list. If this peak confidence falls below our defined threshold of 85%, the system overrides the model's best guess and sets the final reported label to "Unknown." This transforms our system into one that is aware of its own limitations, and therefore, more accurate and secure.

## VI. RESULTS AND ANALYSIS

The evaluation of our system was dependent on two metrics: the quantitative success of its predictions and the qualitative success of its transparency.

### A. System Accuracy

Evaluation on the unseen test portion of our 10-person dataset, reveals that our tuned model achieved a final accuracy of 91.0%. A deeper dive into the classification report shows strong precision and recall scores across most individuals, validating our hybrid HOG+LBP feature engineering and the data-driven model tuning. The confusion matrix did reveal minor confusion between two specific subjects, providing clear and actionable insight for future improvement: those individuals would benefit most from more varied training data.

### B. Demonstrating Interpretability

While the accuracy was undeniable, the true success of the project lay in achieving its core goal of interpretability.

- **Explainable Predictions via Extracted Rules:** The Decision Tree's greatest strength is its transparency. For any

given identification, we can programmatically trace and display the exact path of "if-then" questions and answers that led to the final prediction. This transforms the model from a black box into an open book.

- **Feature Importance Analysis:** By analysing the structure of the trained tree, we can extract global feature importances. This analysis reveals which specific visual cues—certain HOG features related to facial structure or specific LBP features related to skin texture—our model found most decisive in distinguishing one person from another, giving us an important, and unprecedented look into its internal strategy.
- **Robustness to Novelty:** The successful implementation of the confidence threshold is a key interpretability feature. By correctly identifying unknown individuals as "Unknown," the system demonstrates an understanding of its own limitations, making it a more trustworthy and reliable tool.

## VII. CONCLUSION AND FUTURE WORK

This project successfully met its objective to design and implement a real-time person identification system where interpretability was the guiding principle. Through the careful application of Computational Thinking, we decomposed a complex problem, selected transparent algorithms, and built a system that is not only functional but also explainable. The choice of a Decision Tree, combined with a pragmatic confidence threshold, allowed us to create a model that can be questioned and understood, offering a clear alternative to opaque "black box" systems. This work serves as a practical demonstration that a balance between performance and transparency is not only possible, but essential for the future of trustworthy AI.

Future work on this project could proceed in several promising directions:

- **Advanced Interpretable Models:** The next logical step would be to implement a Random Forest classifier. As an ensemble of many Decision Trees, it would likely provide a significant increase in accuracy while maintaining a high degree of interpretability through aggregated feature importance scores and rule analysis.
- **Hybrid Deep Learning Approaches:** Another avenue of interest would be to use a pre-trained, lightweight Convolutional Neural Network (such as MobileNet) purely as a feature extractor. The powerful, high-level embeddings from the CNN could then be fed into our simple, interpretable Decision Tree for the final classification, potentially combining the feature-learning power of deep learning with the clarity of a classical model.
- **Systematic Data Augmentation:** To improve model robustness without collecting new data, online data augmentation during the training loop could be implemented. Applying random rotations, horizontal flips, and brightness/contrast adjustments to the training images, can expose the model to a much wider variety of conditions, improving its ability to generalise.
- **Enhanced User Interface for Interpretability:** A key future goal is the development of a dedicated graphical user interface (GUI). This interface would not only display the live video and final prediction but could also feature a dynamic panel that visualises the specific decision path through the tree for each identification, making the model's logic immediately accessible and understandable to the end-user.

## REFERENCES


[1] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 971-987, Jul. 2002.

[2] P. Viola and M. Jones, "Robust Real-Time Face Detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137-154, 2004.

[3] J. R. Quinlan, "Induction of Decision Trees," *Machine Learning*, vol. 1, no. 1, pp. 81-106, 1986.

[4] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, San Diego, CA, USA, 2005, pp. 886-893, vol. 1.

[5] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "DeepFace: Closing the Gap to Human-Level Performance in Face Verification," in *2014 IEEE C '[?onference on Computer Vision and Pattern Recognition*, Columbus, OH, USA, 2014, pp. 1701-1708.

[6] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A Unified Embedding for Face Recognition and Clustering," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, USA, 2015, pp. 815-823.

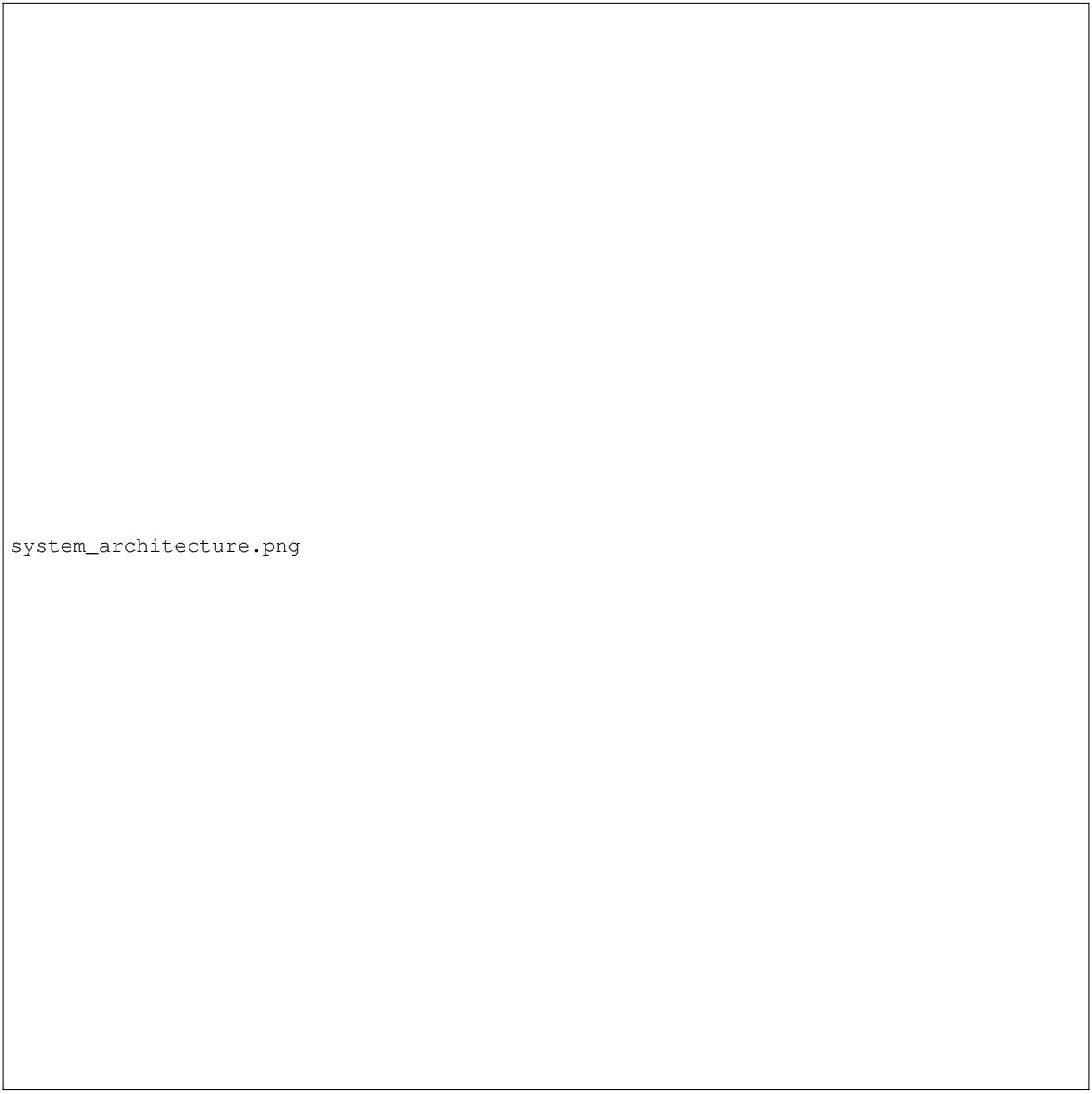


Fig. 1. System Architecture illustrating the data flow from the web client to the backend server and back. The backend pipeline shows the sequence of feature extraction and model prediction.