COMSATS University Islamabad

Lahore Campus

Hybrid Two-Layer Authentication System

Final Year Project Thesis

Department of Computer Engineering

Supervised by:
Dr. Zaid Ahmad

_____
(Signature)

Presented by:

Name	**Reg No.**	**Signature**
Abdullah Laeeq	FA22-BCE-026	_____
Muhammad Faizan Shurjeel	FA22-BCE-086	_____
Ali Hamza	FA22-BCE-071	_____

January 14, 2026

Page Left Blank Intentionally

Declaration

We, the undersigned, hereby declare that this project, "Hybrid Two-Layer Authentication System," is our own original work. It has not been submitted in whole or in part for any other degree or qualification at this or any other university. All sources of information have been duly acknowledged and referenced. We understand the academic regulations regarding plagiarism and affirm that this work complies with them in their entirety. Should this declaration prove to be false, we shall stand responsible for the consequences.

_____

Abdullah Laeeq (FA22-BCE-026)

_____

Muhammad Faizan Shurjeel (FA22-BCE-086)

_____

Ali Hamza (FA22-BCE-071)

Final Approval

This Final Year Project titled "Hybrid Two-Layer Authentication System" by the following students is hereby approved:

Abdullah Laeeq	FA22-BCE-026
Muhammad Faizan Shurjeel	FA22-BCE-086
Ali Hamza	FA22-BCE-071

Project Supervisor:

_____

Dr. Zaid Ahmad
Department of Computer Engineering
COMSATS University Islamabad

Co-Supervisor:

_____

Engr. Talha Naveed
Department of Computer Engineering
COMSATS University Islamabad

External Examiner:

_____

Name: _____
Designation: _____

Head of Department:

Department of Computer Engineering
COMSATS University Islamabad

Dedication

This work is dedicated to our families, whose unwavering support and encouragement have been the foundation of our success, and to our teachers, who have guided us on this academic journey.

Acknowledgements

We would like to express our sincere gratitude to our supervisor, Dr. Zaid Ahmad, for his invaluable guidance, patience, and expertise throughout this project. His mentorship has been instrumental in shaping our understanding and approach. We also thank our co-supervisor, Engr. Talha Naveed, for his technical insights and support. Finally, we are grateful to our families and friends for their constant encouragement and understanding.

Abstract

This project presents the design and implementation of a Hybrid Two-Layer Authentication System that combines facial recognition and speaker verification for secure, contactless access control. Deployed on a Raspberry Pi 4, the system prioritizes both security and user convenience through a parallel processing architecture that analyzes both biometric modalities simultaneously. The core contribution of this work is the rigorous, data-driven selection of lightweight AI models—BlazeFace for face detection, MobileFaceNet for face recognition, and Resemblyzer for speaker verification—that meet real-time performance constraints on resource-limited hardware. Experimental benchmarking confirms that the selected stack achieves an estimated authentication latency of approximately one second, well within our target of two seconds. This thesis documents the complete journey from literature review to practical implementation, providing a blueprint for deploying sophisticated AI-powered security systems on affordable edge hardware.

Keywords: Biometric Authentication, Facial Recognition, Speaker Verification, Edge Computing, Raspberry Pi, Multimodal Fusion, Deep Learning

List of Symbols, Abbreviations, and Acronyms

Abbreviation	Full Form
AI	Artificial Intelligence
ANN	Artificial Neural Network
API	Application Programming Interface
ARM	Advanced RISC Machine
ASV	Automatic Speaker Verification
BCE	Bachelor of Computer Engineering
CNN	Convolutional Neural Network
CPU	Central Processing Unit
DNN	Deep Neural Network
ECAPA	Emphasized Channel Attention, Propagation and Aggregation
EER	Equal Error Rate
FAR	False Acceptance Rate
FPS	Frames Per Second
FRR	False Rejection Rate
FYP	Final Year Project
GE2E	Generalized End-to-End
GPIO	General Purpose Input/Output
GPU	Graphics Processing Unit
GUI	Graphical User Interface
IoT	Internet of Things
LSTM	Long Short-Term Memory
ML	Machine Learning
MTCNN	Multi-task Cascaded Convolutional Networks
NFR	Non-Functional Requirement
NPU	Neural Processing Unit
ONNX	Open Neural Network Exchange
OS	Operating System
PAD	Presentation Attack Detection
RAM	Random Access Memory
RNN	Recurrent Neural Network
SBC	Single Board Computer
SDK	Software Development Kit
SDG	Sustainable Development Goal
SIMD	Single Instruction Multiple Data
SoC	System on Chip
SOTA	State of the Art
SV	Speaker Verification

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background of the Study

In an increasingly connected world, the need for robust yet user-friendly security systems has never been greater. Traditional authentication methods, such as PINs, passwords, and physical keys, are vulnerable to theft, loss, and social engineering attacks. Biometric authentication offers a compelling alternative by verifying a user's identity based on their unique physiological or behavioral characteristics. Unlike passwords, biometrics cannot be easily forgotten, shared, or stolen, making them an inherently more secure foundation for access control.

This project focuses on developing a **Hybrid Two-Layer Authentication System** that combines two powerful biometric modalities: **facial recognition** and **speaker verification**. By processing both streams in parallel, the system can make an authentication decision based on the successful verification of *either* modality (an "OR-logic" approach). This design philosophy prioritizes user convenience and system robustness, as the system can authenticate a user even if one modality is temporarily unavailable (e.g., the user is wearing a mask or there is ambient noise).

A critical constraint of this project is that the entire system must run locally on an affordable, low-power edge device—a **Raspberry Pi 4**. This "edge computing" approach is essential for privacy, as sensitive biometric data never leaves the user's premises, and for reliability, as the system does not depend on an internet connection. This constraint, however, poses significant technical challenges related to computational resources, model optimization, and real-time performance.

1.2 Problem Statement

The primary challenge addressed by this project is:

How can a sophisticated, multi-modal biometric authentication system, combining facial recognition and speaker verification, be designed and implemented to run in

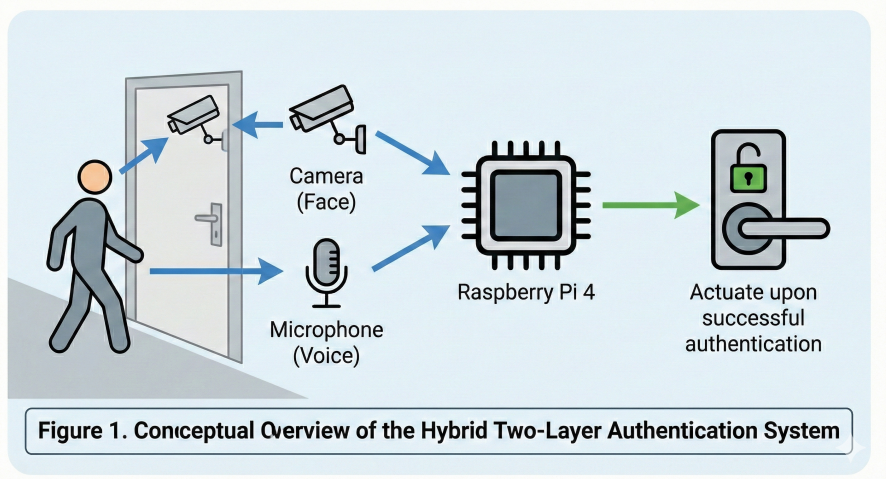


Figure 1.1: Conceptual Overview of the Hybrid Two-Layer Authentication System

real-time on a resource-constrained edge device like a Raspberry Pi 4, while achieving a target end-to-end latency of under 2 seconds?

This challenge decomposes into several sub-problems:

1. **Model Selection:** The vast majority of state-of-the-art (SOTA) biometric models are designed for high-performance servers with dedicated GPUs. Identifying models that offer an acceptable balance of accuracy and speed on a CPU-only platform is non-trivial.

2. **System Architecture:** Designing a software architecture that can capture and process two data streams (video and audio) in parallel, manage inter-thread communication, and make a fused decision efficiently.

3. **Hardware Environment:** Setting up a stable and optimized software environment on the ARM64-based Raspberry Pi, which has a less mature ecosystem than x86 platforms.

4. **Privacy and Security:** Ensuring that the system stores and processes biometric templates securely without transmitting any data over a network.

1.3 Project Objectives

The objectives of this Final Year Project are:

1. To conduct a comprehensive literature review of facial recognition and speaker verification technologies, with a specific focus on models suitable for edge deployment.

2. To design a system architecture for a hybrid, parallel biometric authentication system with clearly defined functional and non-functional requirements.
3. To benchmark and select the optimal combination of AI models (face detector, face recognizer, voice encoder) that meets real-time performance constraints on a Raspberry Pi 4.
4. To implement a functional prototype of the authentication system with a decision fusion engine and GPIO control for physical access (e.g., activating a door lock).
5. To evaluate the final system's performance in terms of latency, accuracy (FAR/FRR), and resource consumption.

1.4 Scope of the Project

In Scope:

- Design and implementation on a single Raspberry Pi 4 Model B (8GB).
- Face detection, face recognition (verification mode), and speaker verification.
- A simple OR-logic decision fusion mechanism.
- Local storage of biometric templates on the device's filesystem.
- GPIO control for a physical output signal (e.g., to a relay).
- A command-line interface for enrollment and testing.

Out of Scope:

- Cloud-based processing or network connectivity.
- Advanced anti-spoofing measures (e.g., 3D depth sensing for liveness detection), though basic blink detection may be explored.
- A polished graphical user interface (GUI).
- Large-scale deployment or user management features.
- Support for multiple concurrent authentication requests.

1.5 Significance of the Study

Academic Significance:

This project provides a practical case study in deploying deep learning models on resource-constrained hardware. It contributes a documented methodology for benchmarking and selecting models based on empirical performance data rather than theoretical claims, which is a valuable resource for students and researchers working in edge AI.

Practical Significance:

The resulting system serves as a blueprint for affordable, privacy-respecting biometric security solutions suitable for small businesses, educational institutions, and residential applications in regions where expensive commercial systems are not viable.

1.6 Broader Impact (UN SDGs)

This project aligns with the United Nations Sustainable Development Goals:

- **SDG 9 (Industry, Innovation, and Infrastructure):** By developing an affordable, innovative security system using cutting-edge AI, we contribute to building resilient infrastructure accessible to developing economies.
- **SDG 11 (Sustainable Cities and Communities):** Secure access control is fundamental to safe urban environments. Our privacy-first, offline approach provides security without the societal risks of centralized surveillance databases.
- **SDG 16 (Peace, Justice, and Strong Institutions):** By enabling secure verification of identity, the system can contribute to secure and accountable institutions.

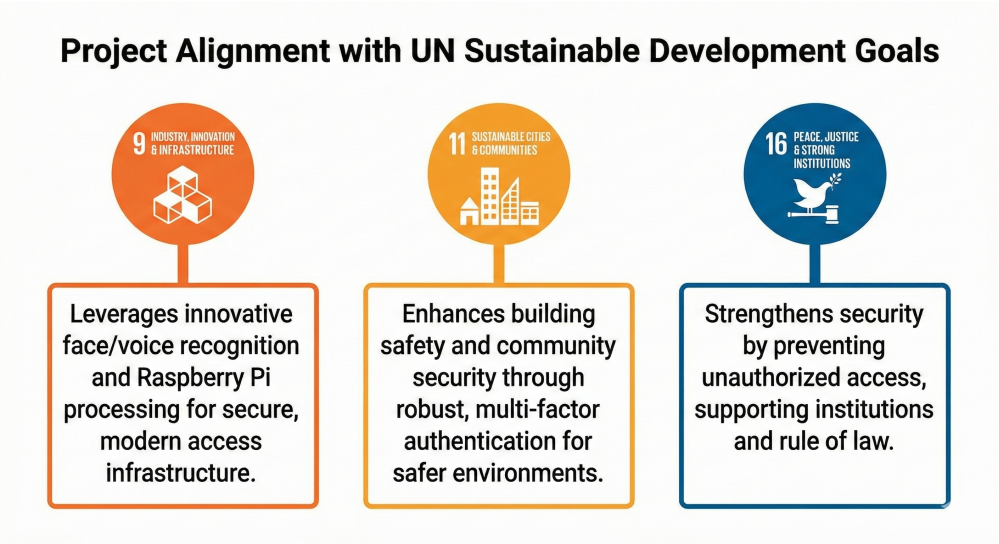


Figure 1.2: Project Alignment with United Nations Sustainable Development Goals

1.7 Report Organization

This thesis is organized as follows:

- **Chapter 1: Introduction** - Provides background, problem statement, objectives, and scope.
- **Chapter 2: Literature Review** - Presents a comprehensive survey of facial recognition, speaker verification, multimodal biometrics, edge computing, and related technologies.
- **Chapter 3: System Design & Analysis** - Details the proposed methodology, system requirements, hardware selection, and architecture.
- **Chapter 4: Implementation and Results** - Documents the practical setup, benchmarking procedures, and experimental findings.
- **Chapter 5: Conclusion and Future Work** - Summarizes achievements, lessons learned, and future directions.

Chapter 2

Literature Review

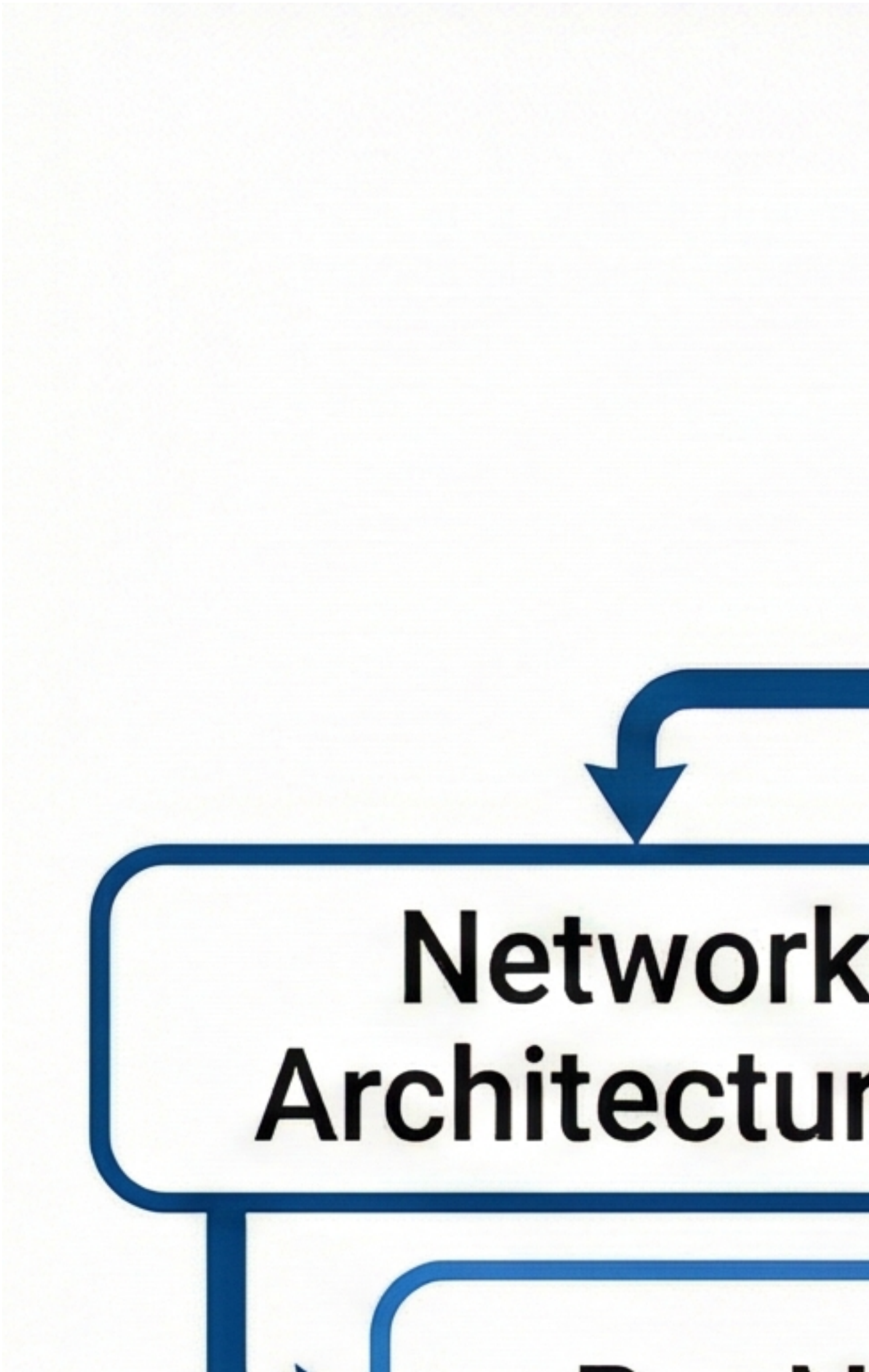
2.1 Theoretical Framework and Taxonomy

A major challenge in reviewing the vast body of biometric literature is the frequent conflation of model architectures, training methodologies, and the software libraries that implement them. To bring clarity to our selection process, we first established a clear taxonomy to categorize and evaluate the available technologies. This framework was essential for making informed, data-driven decisions rather than simply choosing models based on popularity.

The taxonomy distinguishes between:

- **Network Architectures:** The fundamental structure of the neural network (e.g., ResNet, MobileNet, LSTM).
- **Training Methodologies:** The loss functions and training techniques used to optimize the model (e.g., Triplet Loss, ArcFace).
- **Pre-trained Models:** Specific instances of trained networks available for deployment (e.g., FaceNet, InsightFace models).
- **Software Libraries/Toolkits:** Python packages that provide easy-to-use APIs wrapping these models (e.g., MediaPipe, Resemblyzer, SpeechBrain).

This structured approach allowed us to evaluate technologies at multiple levels: architectural efficiency, training quality, ease of integration, and ultimately, real-world performance on our target hardware.


Network
Architectur

2.2 Facial Recognition Pipeline: A Deep Dive

The facial recognition process is fundamentally a two-stage pipeline: detection, followed by recognition. The overall latency and reliability of the system are critically dependent on the performance of both stages.

[PLACEHOLDER IMAGE]
`face_pipeline_overview.png`

Horizontal flowchart showing the complete face recognition pipeline: Input Image → Face Detection (output: bounding box + landmarks) → Face Alignment/Crop → Face Recognition Model → 512-D Embedding Vector → Cosine Similarity Comparison → Match/No Match Decision. Include sample face images at each stage showing the transformation.

Figure 2.2: Complete Facial Recognition Pipeline from Input to Decision

2.2.1 Stage 1: Face Detection - Finding the Face in the Frame

The face detection module is responsible for perpetually scanning the input video stream to locate the bounding box coordinates of any faces present. Its efficiency is paramount, as it runs continuously. Our review and subsequent experimentation covered a spectrum of techniques.

- **Haar Cascade Classifiers:** This classic machine learning technique, implemented efficiently in the OpenCV library, was our initial baseline. It is based on a cascade of features trained using AdaBoost.
  - *Pros:* Extremely fast on CPUs, low memory footprint.
  - *Cons:* Prone to false positives, struggles significantly with non-frontal faces (tilted or profile views), and provides no facial landmark information. Our experimental tests confirmed these limitations.

- **MTCNN (Multi-task Cascaded Convolutional Networks):** Zhang et al. [**?**] introduced this deep learning-based model known for its high accuracy. It uses a three-stage cascade of CNNs (P-Net, R-Net, O-Net) to progressively refine face detections and identify key facial landmarks.

  Paper Summary: Zhang et al. proposed a cascaded framework where each stage serves a specific purpose: the Proposal Network (P-Net) quickly generates candidate windows, the Refine Network (R-Net) filters false positives, and the Output Network (O-Net) produces final bounding boxes with five-point facial landmarks. The multi-task learning approach jointly optimizes face classification, bounding box regression, and landmark localization, achieving state-of-the-art accuracy at the time of publication.

  - *Pros:* High detection accuracy, provides 5-point facial landmarks.
  - *Cons:* Our experimental benchmark on the Raspberry Pi 4 CPU revealed it to be extremely slow, achieving only $\sim$**2 FPS**. This makes it completely unsuitable for any real-time video application.

- **BlazeFace:** Bazarevsky et al. [**?**] developed BlazeFace at Google Research as a SOTA lightweight detector specifically designed for mobile and edge GPUs, but its architecture is also highly efficient on CPUs. It is inspired by MobileNet and uses a hardware-aware design.

  Paper Summary: BlazeFace introduces several key innovations: (1) a modified MobileNet backbone with increased receptive field, (2) an anchor scheme specifically tuned for face detection, and (3) a tie resolution strategy for reducing jitter in video applications. The model achieves sub-millisecond inference on mobile GPUs while providing 6-point facial landmarks (eyes, ears, nose, mouth) useful for face alignment and basic liveness detection.

  - *Pros:* Blazing fast, highly accurate, and provides 6-point facial landmarks which can be used for liveness detection (e.g., eye tracking). Our benchmark showed it achieving **30-45 FPS** on the Raspberry Pi 4.
  - *Cons:* Slightly more complex to implement from scratch than Haar cascades, but readily available via Google's MediaPipe framework.

[PLACEHOLDER IMAGE]

`face_detection_comparison.png`

Side-by-side comparison showing three face detection methods on the same input image: (Left) Haar Cascade with multiple false positives and missed detections; (Center) MTCNN with accurate detection but noted "2 FPS" latency; (Right) BlazeFace with accurate detection, landmarks shown, and noted "30-45 FPS" speed. Include a small bar chart below comparing FPS performance.

Figure 2.3: Visual Comparison of Face Detection Methods on Raspberry Pi 4

2.2.2 Stage 2: Face Recognition - Identifying the Face

Once a face is detected and cropped, it is passed to a recognition model to generate a discriminative feature vector, or "embedding." This stage is where the core identification happens.

Network Architectures

This refers to the neural network's structure.

- **InceptionResNetV1:** The foundational architecture used in the original Google FaceNet paper [**?**]. It combines Inception modules with residual connections for powerful feature extraction.

  Paper Summary (FaceNet): Schroff et al. introduced a unified framework for face verification, recognition, and clustering using a deep convolutional network trained with triplet loss. The key innovation was learning a compact 128-dimensional Euclidean embedding where distances directly correspond to face similarity. The model achieved 99.63% accuracy on LFW benchmark, setting a new state of the art.

However, the architecture is large and computationally expensive. Our benchmark showed an inference time of $\sim$**1200 ms** on the Pi 4.

- **ResNet (Residual Networks):** Architectures like ResNet-34 and ResNet-50 are common backbones for face recognition and form the basis of models like the InsightFace "Buffalo_L/M" family [**?**]. Our tests showed the ResNet-50 based model to be even slower, at $\sim$**1800 ms**.

- **MobileFaceNet:** Chen et al. [**?**] proposed this lightweight architecture specifically designed for mobile CPUs. It uses depthwise separable convolutions to drastically reduce the number of parameters and computations.

  Paper Summary: The authors demonstrated that face verification can be achieved efficiently on mobile devices by: (1) using Global Depthwise Convolution (GDConv) in place of global average pooling to retain spatial information, (2) employing a bottleneck structure optimized for the face recognition task, and (3) training with ArcFace loss for discriminative embeddings. The resulting model is only 4MB with 0.99M parameters, achieving 99.28% on LFW while running in real-time on mobile CPUs.

  Our tests showed an inference time of $\sim$**250-350 ms**, making it 3-5x faster than the other models.

[PLACEHOLDER IMAGE]

`mobilefacenet_architecture.png`

Detailed architecture diagram of MobileFaceNet showing: Input (112x112x3) $\rightarrow$ Initial Conv layers $\rightarrow$ Series of Bottleneck blocks with depthwise separable convolutions (show internal structure of one block expanded) $\rightarrow$ Global Depthwise Conv $\rightarrow$ Linear layer $\rightarrow$ 512-D embedding output. Include parameter counts at each stage and total model size (4MB).

Figure 2.4: MobileFaceNet Architecture with Depthwise Separable Convolutions

Loss Functions

These are the mathematical formulas used during training to make the model's embeddings more discriminative.

- **Triplet Loss:** Popularized by FaceNet [**?**], it works by ensuring that the embedding of an "anchor" face is closer to a "positive" example (same person) than to a "negative" example (different person) by a margin α:

$$L = \sum_{i}^{N} \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha \right]_+$$

- **ArcFace (Additive Angular Margin Loss):** Deng et al. [**?**] proposed a more modern and effective loss function that works by adding an angular margin to the separation between classes in the embedding space.

  Paper Summary: ArcFace introduces an additive angular margin penalty m between the feature vector and the target weight to simultaneously enhance intra-class compactness and inter-class discrepancy. The loss is defined as:

$$L = -\frac{1}{N} \sum_{i=1}^{N} \log \frac{e^{s(\cos(\theta_{y_i} + m))}}{e^{s(\cos(\theta_{y_i} + m))} + \sum_{j=1, j \neq y_i}^{n} e^{s \cos \theta_j}}$$

  where s is a scaling factor and m is the angular margin. This geometric interpretation provides clear optimization guidance and achieves state-of-the-art results across multiple benchmarks.

  Most SOTA pre-trained models, including the MobileFaceNet version we chose, are trained using ArcFace.

[PLACEHOLDER IMAGE]

`arcface_loss_visualization.png`

Geometric visualization comparing Softmax loss vs. ArcFace loss in 2D angular space. (Left) Softmax: embeddings of different classes shown with overlapping decision boundaries; (Right) ArcFace: same classes with clear angular margin m separating them, showing tighter intra-class clustering and larger inter-class separation. Include angular margin annotation.

Figure 2.5: Visualization of ArcFace Angular Margin Loss vs. Standard Softmax

2.3 Comprehensive Survey of Speaker Recognition Models

To select the optimal voice architecture, we conducted an extensive study of five distinct model categories, analyzing their suitability for our CPU-bound edge device.

[PLACEHOLDER IMAGE]

`speaker_models_taxonomy.png`

Hierarchical diagram organizing speaker recognition models into categories: (A) High-Complexity SOTA (ECAPA-TDNN, Transformers); (B) Efficient CNNs (MobileNet-based, GhostNet); (C) TDNN-based (x-vector); (D) RNN-based (GE2E/LSTM); Each category shows accuracy vs. latency trade-off indicator with color coding (red for slow, green for fast).

Figure 2.6: Taxonomy of Speaker Recognition Models by Complexity and Efficiency

2.3.1 Category A: State-of-the-Art & High-Complexity Models

These models define the upper limit of academic accuracy but present insurmountable challenges for real-time deployment on a Raspberry Pi.

ECAPA-TDNN: Desplanques et al. [**?**] proposed the Emphasized Channel Attention, Propagation and Aggregation Time Delay Neural Network, which represents the current state of the art in speaker verification.

Paper Summary: ECAPA-TDNN introduces three key innovations over the standard x-vector TDNN: (1) Squeeze-and-Excitation (SE) blocks with channel-dependent frame attention, (2) Multi-scale feature aggregation through dense skip connections, and (3) Channel-dependent attentive statistics pooling. These additions significantly improve the model's ability to capture speaker-discriminative information across different temporal scales, achieving state-of-the-art results on VoxCeleb with an EER of 0.87%.

Table 2.1: SOTA High-Complexity Speaker Recognition Models

Model	**Architecture Breakdown**	**Strengths**	**Edge Feasibility**
ECAPA-TDNN	Emphasized Channel Attention + TDNN	Highest published accuracy, robust to noise	Extremely high compute requirements. Our tests showed >**4 seconds** inference time on the Pi 4 CPU
Transformers	Self-Attention based models	Excellent at capturing long-range dependencies in speech	Even heavier than ECAPA-TDNN, completely unfeasible for our hardware

2.3.2 Category B & C: Efficient CNN-based Models

This category contains the most promising candidates for our project, leveraging mature CNN optimizations.

x-vector (TDNN): The x-vector system, developed at JHU, uses a Time Delay Neural Network architecture with statistics pooling to generate fixed-length speaker embeddings from variable-length utterances.

Paper Summary: The x-vector approach extracts frame-level representations using TDNN layers, then aggregates them using statistics pooling (mean and standard deviation) to produce a fixed-dimensional embedding. Training uses a softmax loss over a large speaker set, with embeddings extracted from an intermediate layer. This architecture provides an excellent balance between accuracy and computational efficiency.

Table 2.2: Efficient CNN-Based Speaker Recognition Models

Model	**Architecture Breakdown**	**Strengths**	**Edge Feasibility**
MobileNet/ GhostNet	Standard efficient 2D-CNNs	Excellent balance of accuracy/speed. Proven for edge deployment in vision tasks	Strong candidates
x-vector (TDNN)	Time Delay Neural Network + Statistics Pooling	Simpler and lighter than ECAPA-TDNN	Excellent baseline, fast on CPU
SincNet	Parametric 1D-CNN using Sinc functions	Extremely efficient and interpretable first layer	Prime candidate for highly constrained devices

2.3.3 Category D: RNN-based Models (Our Selected Approach)

Recurrent Neural Networks are well-suited for sequential data like audio.

GE2E (Generalized End-to-End Loss): Wan et al. [**?**] proposed this training methodology for speaker verification that significantly improves both training efficiency and model performance.

Paper Summary: The GE2E loss operates on batches containing utterances from multiple speakers, computing a similarity matrix between all utterance embeddings and all speaker centroids. The loss encourages each utterance embedding to be closer to its own speaker's centroid than to any other speaker's centroid. Key advantages include: (1) more stable gradients during training, (2) ability to use larger batch sizes efficiently, and (3) direct optimization of the verification task. The authors demonstrate significant improvements over the earlier Tuple-based End-to-End (TE2E) loss.

The **Resemblyzer** library [**?**] provides a convenient Python implementation of GE2E-trained LSTM encoders, which we selected for our system.

Table 2.3: RNN-Based Speaker Recognition Models

Toolkit	**Underlying Model**	**Strengths**	**Edge Feasibility**
Resemblyzer	**GE2E (LSTM-based)**	Fast, easy to implement, provides a simple API for generating embeddings	**Excellent.** Our experimental tests showed it generated embeddings from a 3-second clip in **under 1 second**, meeting our real-time requirement

[PLACEHOLDER IMAGE]

`ge2e_loss_diagram.png`

Visualization of GE2E loss computation: (Left) Batch structure showing N speakers × M utterances matrix; (Center) LSTM encoder processing each utterance to produce embeddings; (Right) Similarity matrix between embeddings and speaker centroids, with loss pushing diagonal elements high and off-diagonal elements low. Include mathematical notation for centroid computation and similarity.

Figure 2.7: Generalized End-to-End (GE2E) Loss Training Mechanism

2.4 Optimization and the Role of ONNX

A key part of our methodology is the plan to convert all selected models to the **ONNX (Open Neural Network Exchange)** format [**?**]. ONNX is an open standard for representing machine learning models. Using the **ONNX Runtime**, we can execute these models with a highly optimized inference engine that is specifically tuned for CPU performance. This step is crucial for squeezing the maximum possible speed out of the Raspberry Pi's processor and is a key part of our implementation plan for

FYP-II.

Paper Summary (ONNX Runtime): Bai et al. [**?**] demonstrated that ONNX Runtime provides significant performance improvements through: (1) graph-level optimizations including constant folding and operator fusion, (2) execution provider abstraction allowing hardware-specific optimizations, and (3) memory planning to reduce allocation overhead. On ARM processors specifically, ONNX Runtime can achieve up to 40% performance improvement over native framework inference.

The ONNX ecosystem provides several advantages:

- **Cross-framework compatibility:** Models trained in PyTorch or TensorFlow can be converted to a unified format.
- **Hardware optimization:** ONNX Runtime includes optimizations for various CPU architectures, including ARM.
- **Reduced inference latency:** Graph optimizations and operator fusion can significantly reduce model execution time.
- **Smaller memory footprint:** Optimized models often require less RAM during inference.

[PLACEHOLDER IMAGE]

`onnx_conversion_pipeline.png`

Flowchart showing model conversion and deployment: PyTorch Model / TensorFlow Model → ONNX Export (torch.onnx.export / tf2onnx) → ONNX Model (.onnx file) → ONNX Runtime optimizations (operator fusion, constant folding, memory planning) → Optimized ONNX Model → Deployment on Raspberry Pi with ARM-specific execution provider. Include performance improvement percentages at each stage.

Figure 2.8: ONNX Model Conversion and Optimization Pipeline

2.5 Multimodal Biometric Authentication: State of the Art

The integration of multiple biometric modalities has emerged as a powerful approach to overcome the limitations of unimodal systems. This section reviews the current state of the art in multimodal biometric authentication, with particular emphasis on systems that combine facial and voice recognition.

2.5.1 Advantages of Multimodal Biometrics

Multimodal biometric systems offer several compelling advantages over their unimodal counterparts:

1. **Improved Accuracy:** By combining evidence from multiple sources, multimodal systems can achieve significantly lower error rates. Research by Zheng et al. [**?**] demonstrates that combining face and physiological signals can reduce Equal Error Rate (EER) by up to 60% compared to single-modality systems.

   Paper Summary: Zheng et al. proposed a novel multimodal system combining facial recognition with camera-based photoplethysmography (PPG) and fingerprint data. Their fusion approach uses quality-aware weighting to dynamically adjust the contribution of each modality based on signal quality, achieving robust performance even when individual modalities are degraded.

2. **Enhanced Robustness:** Environmental factors that degrade one modality may not affect others. For instance, poor lighting conditions that compromise facial recognition may not impact voice recognition, and vice versa. This resilience is particularly valuable in real-world deployment scenarios [**?**].

3. **Reduced Failure-to-Enroll Rate:** Some individuals may have difficulty enrolling in specific biometric systems due to physical characteristics or disabilities. Multimodal systems provide alternative authentication paths, improving accessibility [**?**].

4. **Increased Security Against Spoofing:** Simultaneously spoofing multiple biometric modalities is significantly more challenging than compromising a single modality. This increases the cost and complexity of attacks, providing better security [**?**].

5. **Flexibility in Application Scenarios:** Different scenarios may favor different modalities. A multimodal system can adapt by emphasizing the most reliable modality in a given context [**?**].

[PLACEHOLDER IMAGE]

`multimodal_advantages_diagram.png`

Infographic showing five key advantages of multimodal biometrics with icons: (1) Accuracy - chart showing error rate reduction; (2) Robustness - shield icon with environmental factors; (3) Accessibility - universal access symbol; (4) Security - lock with multiple keys; (5) Flexibility - adaptive gears. Each section includes a brief statistic or benefit statement.

Figure 2.9: Key Advantages of Multimodal Biometric Systems

2.5.2 Fusion Strategies in Multimodal Biometrics

The literature identifies three primary levels at which biometric information can be fused:

Feature-Level Fusion

Feature-level fusion combines feature vectors from different modalities before matching. While this approach can theoretically provide the most comprehensive representation, it faces several challenges:

- **Dimensionality Issues:** Concatenating features from multiple modalities often results in extremely high-dimensional vectors, leading to the curse of dimensionality.
- **Incompatibility:** Feature sets from different modalities may be incompatible or incommensurate, requiring complex normalization strategies.

- **Computational Cost:** High-dimensional feature spaces increase computational requirements substantially.

Despite these challenges, recent work by Talreja et al. [**?**] proposes using deep hashing techniques to generate compact binary representations from multimodal features, achieving both security and efficiency.

Paper Summary: Talreja et al. introduced a deep hashing framework that learns to map high-dimensional multimodal biometric features into compact binary codes while preserving discriminative information. The approach uses a combination of classification loss and quantization loss to ensure the binary codes are both discriminative and compact, enabling efficient template storage and matching.

Score-Level Fusion

Score-level fusion combines matching scores from individual biometric matchers. This is the most popular fusion approach due to its simplicity and effectiveness. Common fusion techniques include:

- **Simple Sum/Product Rules:** Combining normalized scores using arithmetic operations.
- **Weighted Fusion:** Assigning different weights to modalities based on their reliability or quality.
- **Machine Learning Approaches:** Using classifiers like SVM, neural networks, or decision trees to learn optimal fusion strategies [**?**].

Poh et al. [**?**] conducted an extensive benchmark study comparing various score-level fusion algorithms, considering both quality-dependent and cost-sensitive scenarios.

Paper Summary: This comprehensive study evaluated 22 different score-level fusion methods across multiple multimodal biometric databases. Key findings include: (1) quality-aware fusion consistently outperforms static fusion, (2) trained fusion methods outperform fixed rules when sufficient training data is available, and (3) the optimal fusion method depends on the specific modality combination and application requirements. The benchmark provides valuable guidance for practitioners selecting fusion strategies.

Decision-Level **Fusion**

Decision-level fusion combines the final decisions (accept/reject) from individual biometric systems using voting schemes or logical operations. While this is the simplest approach, it discards valuable information contained in matching scores. However, it remains useful in scenarios where individual systems operate as black boxes or when computational resources are extremely limited.

Our proposed system employs score-level fusion with OR-logic at the decision stage, balancing simplicity with the preservation of discriminative information from both modalities.

[PLACEHOLDER IMAGE]

`fusion_strategies_comparison.png`

Three-panel diagram comparing fusion levels: (Left) Feature-Level: shows feature vectors being concatenated before a single classifier; (Center) Score-Level: shows parallel classifiers producing scores that are combined (sum, product, weighted); (Right) Decision-Level: shows parallel classifiers producing accept/reject decisions combined via voting. Include pros/cons summary below each panel and highlight "Score-Level" as our chosen approach.

Figure 2.10: Comparison of Multimodal Fusion Strategies at Different Levels

2.5.3 Recent Advances in Audio-Visual Biometrics

The specific combination of facial and voice biometrics has received considerable attention in recent years, driven by its natural alignment with human communication modalities.

Joint Audio-Visual Learning

Liu et al. [**?**] propose a joint audio-visual lip biometrics system that leverages the correlation between lip movements and speech.

Paper Summary: The DeepLip framework simultaneously processes audio spectrograms and lip video sequences using parallel CNN streams, then fuses the representations through attention-based pooling. By exploiting the natural synchronization between lip movements and speech, the system achieves 0.99% EER on the XM2VTS database, significantly outperforming single-modality approaches. The correlation between modalities provides inherent spoofing resistance, as generating realistic synchronized audio-visual fakes is extremely challenging.

Deep Learning for Multimodal Fusion

Recent work has explored sophisticated deep learning architectures for multimodal fusion:

- **Attention Mechanisms:** Reddy et al. [**?**] propose using Gumbel-Softmax for neural architecture search in bimodal systems, achieving adaptive fusion that adjusts to data characteristics.

  Paper Summary: This work introduces a differentiable neural architecture search (NAS) approach using Gumbel-Rao Monte Carlo estimation to automatically discover optimal fusion architectures for audio-visual biometrics. The search space includes various attention mechanisms, fusion points, and feature processing options. The discovered architectures consistently outperform hand-designed alternatives while being computationally efficient.

- **Cross-Modal Learning:** Zhang et al. [**?**] demonstrate that training face and voice encoders with cross-modal contrastive learning improves both individual and joint performance.

- **Multimodal Transformers:** The application of transformer architectures to multimodal biometrics shows promise, with models learning to attend to the most discriminative features across modalities [**?**].

  Paper Summary (Multimodal LLMs for Biometrics): Zhang et al. [**?**] explored leveraging pre-trained multimodal large language models (MLLMs) for facial expression recognition, demonstrating that foundation models can be effectively adapted for biometric tasks through careful prompting and fine-tuning strategies.

2.5.4 Multimodal Biometric Databases

The availability of high-quality multimodal databases is crucial for advancing research. Several notable databases have been released:

- **BehavePassDB:** Stragapede et al. [**?**] present a comprehensive database of mobile behavioral biometrics, including touchscreen interactions and sensor data.

  Paper Summary: BehavePassDB contains data from 81 subjects performing various mobile device interactions over multiple sessions. The database includes touch gestures, keystroke dynamics, and sensor data (accelerometer, gyroscope) collected in both controlled and uncontrolled settings, enabling research on continuous authentication and multimodal behavioral biometrics.

- **SpeakingFaces:** Abdrakhmanova et al. [**?**] introduce a large-scale dataset with synchronized thermal, visual, and audio streams, supporting research in multimodal authentication under varying environmental conditions.

  Paper Summary: This dataset comprises 142 subjects recorded with co-registered thermal-visual camera pairs while speaking commands in multiple languages. The synchronized multi-stream data enables research on robust authentication under varying lighting conditions (thermal images are illumination-invariant) and multimodal fusion of face, thermal signature, and voice.

- **Smartphone Multimodal Biometric Database:** Ramachandra et al. [**?**] provide a dataset specifically designed for smartphone-based multimodal authentication, including face, voice, and periocular biometrics.

- **MobiBits:** Bartuzi et al. [**?**] present a mobile biometric database incorporating thermal imaging, representing a step toward more robust authentication in challenging environments.

[PLACEHOLDER IMAGE]

`multimodal_databases_overview.png`

Comparison table/infographic of major multimodal biometric databases: columns for Database Name, Modalities Included (icons for face, voice, fingerprint, thermal, behavioral), Number of Subjects, Sessions, and Key Features. Highlight databases most relevant to face+voice authentication.

Figure 2.11: Overview of Major Multimodal Biometric Databases

2.6 Edge Computing and Biometric Systems

The deployment of biometric systems on edge devices presents unique challenges and opportunities. This section examines the landscape of edge computing for biometrics, focusing on the trade-offs between accuracy, latency, privacy, and resource constraints.

[PLACEHOLDER IMAGE]

`edge_vs_cloud_comparison.png`

Split comparison diagram: (Left) Cloud-based biometrics showing data flowing to remote servers with concerns listed (latency, privacy, connectivity dependence, bandwidth); (Right) Edge-based biometrics showing local processing with benefits listed (low latency, privacy preservation, offline operation, reduced bandwidth). Include typical latency numbers for each approach.

Figure 2.12: Edge vs. Cloud Biometric Processing: Trade-offs and Considerations

2.6.1 The Case for Edge-Based Biometric Processing

Several factors drive the shift toward edge-based biometric processing:

1. **Privacy Preservation:** Processing biometric data locally eliminates the need to transmit sensitive information over networks, addressing major privacy concerns. This aligns with privacy regulations like GDPR and CCPA that mandate data minimization and local processing where possible [**?**].

2. **Latency Reduction:** For real-time applications like access control, network round-trip time to cloud servers introduces unacceptable delays. Edge processing can reduce total latency by an order of magnitude [**?**].

   Paper Summary (REVAMP²T): Neff et al. [**?**] demonstrated a complete multi-camera pedestrian tracking system running on Raspberry Pi devices, achieving real-time performance through careful algorithmic optimization. Their system processes video locally and only transmits compressed tracking results, reducing bandwidth by 99% compared to streaming raw video.

3. **Network Independence:** Edge-based systems remain operational even when network connectivity is unreliable or unavailable, crucial for security-critical applications [**?**].

4. **Bandwidth Conservation:** Transmitting high-resolution video streams for biometric analysis consumes substantial bandwidth. Local processing reduces network load significantly [**?**].

5. **Scalability:** Distributed edge processing scales naturally with the number of devices, avoiding the bottleneck of centralized cloud processing [**?**].

2.6.2 Hardware Platforms for Edge Biometrics

Various hardware platforms have been explored for edge-based biometric processing, each with distinct characteristics:

Embedded ARM Processors

Single-board computers like the Raspberry Pi represent the most accessible edge platform. Recent work by Mohammadi et al. [**?**] provides a comprehensive comparison of facial expression recognition across CPU, GPU, VPU (Vision Processing Unit), and TPU (Tensor Processing Unit) on edge devices.

Paper Summary: This study benchmarks facial expression recognition models across four different compute backends on the same edge device. Key findings include: (1) VPUs provide the best performance-per-watt ratio for CNN inference, (2) TPUs excel for batch processing but have higher latency for single-image inference, (3) CPU-only solutions remain competitive for small models with careful optimization, (4) Memory bandwidth is often the bottleneck rather than compute capacity.

Neural Processing Units (NPUs)

Dedicated AI accelerators like Google Coral TPU and Intel Neural Compute Stick offer significant performance improvements for neural network inference. Acien et al. [**?**] evaluate various edge AI accelerators for behavioral biometrics, finding that TPU-based solutions achieve the best performance-per-watt ratio.

Neuromorphic Hardware

Emerging neuromorphic processors like Intel Loihi show promise for ultra-low-power biometric processing. Smith et al. [**?**] compare neuromorphic hardware against edge AI accelerators for real-time facial expression recognition.

Paper Summary: This comparative study evaluates spiking neural networks (SNNs) on neuromorphic hardware against conventional ANNs on edge AI accelerators.

Results show neuromorphic solutions achieve two orders of magnitude reduction in power consumption (10-100 mW vs. 1-10 W) while maintaining comparable accuracy. However, neuromorphic hardware currently has limited availability and requires specialized training pipelines.

FPGA-Based Solutions

Field-Programmable Gate Arrays (FPGAs) offer flexibility and efficiency for custom biometric processing pipelines. Fasfous et al. [**?**] implement a binary neural network on FPGA for COVID-19 mask detection, achieving real-time performance with minimal power consumption.

Paper Summary: BinaryCoP demonstrates that extreme quantization (1-bit weights and activations) enables deployment of face detection and mask classification models on low-power FPGAs. The binary network achieves 50 FPS at only 2.3W power consumption, making it suitable for battery-powered devices. Accuracy degradation from binarization is mitigated through architectural modifications and training strategies.

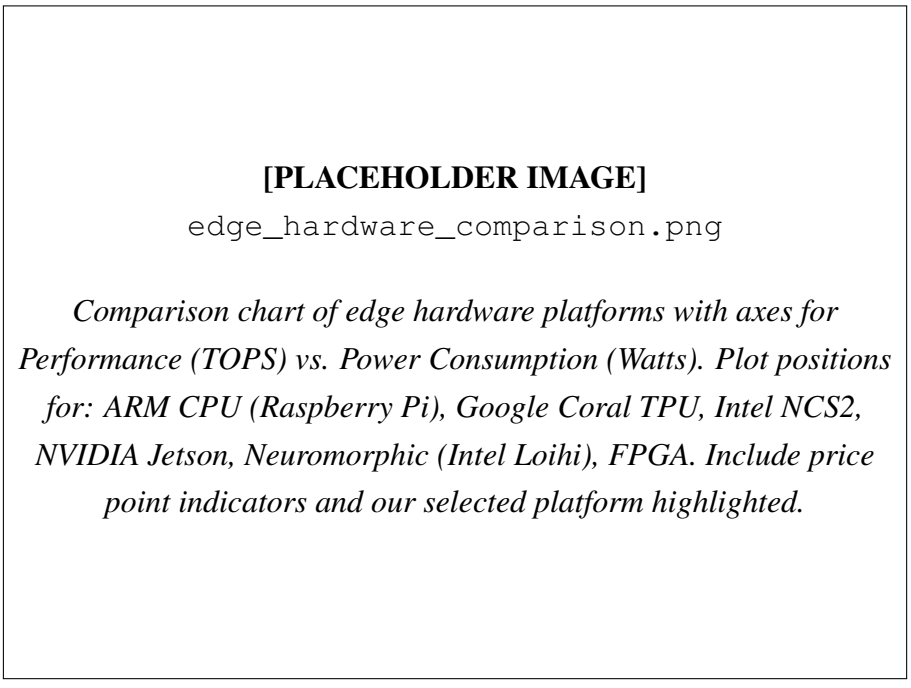


Figure 2.13: Edge Hardware Platforms: Performance vs. Power Consumption Trade-offs

2.6.3 Optimization Techniques for Edge Deployment

Deploying sophisticated deep learning models on resource-constrained edge devices requires careful optimization. Recent research has explored multiple optimization strategies:

Model **Compression**

- **Quantization:** Reducing numerical precision from 32-bit floating point to 8-bit or even binary representations can dramatically reduce model size and computation. Liu et al. [**?**] demonstrate 75% memory savings through 8-bit quantization while maintaining recognition accuracy.

  Paper Summary: This work presents a comprehensive study of quantization techniques for speaker verification models. Key findings include: (1) Post-training quantization to INT8 preserves accuracy within 0.5% EER for most models, (2) Quantization-aware training can recover most accuracy loss for more aggressive quantization, (3) Mixed-precision quantization targeting memory-intensive layers provides the best accuracy-efficiency trade-off.

- **Pruning:** Removing redundant or less important connections in neural networks. Structured pruning can achieve 50-70% reduction in model size with minimal accuracy loss [**?**].

- **Knowledge Distillation:** Training smaller "student" models to mimic larger "teacher" models. This approach is particularly effective for biometric applications where large models are impractical for edge deployment [**?**].

[PLACEHOLDER IMAGE]

`model_optimization_techniques.png`

Three-panel illustration of optimization techniques: (Left) Quantization: neural network with floating point values being converted to integers, showing memory savings; (Center) Pruning: network visualization with connections being removed, showing sparse structure; (Right) Knowledge Distillation: large teacher network transferring knowledge to smaller student network. Include typical compression ratios for each technique.

Figure 2.14: Model Compression Techniques for Edge Deployment

Efficient Network Architectures

Mobile-optimized architectures specifically designed for edge devices:

- **MobileNets:** Utilize depthwise separable convolutions to reduce computational cost while maintaining accuracy. Widely adopted in mobile computer vision applications [**?**].
- **EfficientNets:** Employ compound scaling to balance network depth, width, and resolution, achieving superior accuracy-efficiency trade-offs [**?**].
- **ShuffleNets:** Use channel shuffle operations to enable efficient group convolutions, particularly effective on ARM processors [**?**].
- **SqueezeNets:** Employ fire modules with squeeze and expand layers to reduce parameters while maintaining accuracy [**?**].

Yang et al. [**?**] propose EdgeCNN, a convolutional architecture specifically designed for edge devices with slow memory access.

Paper Summary: EdgeCNN addresses the memory bandwidth bottleneck common in embedded devices by designing convolution operations that maximize data reuse

within on-chip memory. The architecture achieves real-time performance on Raspberry Pi for facial expression recognition while using 3x less memory bandwidth than standard CNNs.

Hardware-Software Co-Design

Optimizing the entire stack from algorithms to hardware interfaces:

- **Operator Fusion:** Combining multiple operations to reduce memory access overhead.
- **Memory Layout Optimization:** Organizing data to maximize cache utilization and minimize memory bandwidth requirements.
- **Parallel Processing:** Exploiting multi-core architectures and SIMD instructions for parallel execution.

2.6.4 Real-Time Constraints and System Design

Real-time biometric systems must meet strict latency requirements while balancing accuracy and resource consumption. Key considerations include:

Pipeline Optimization

Hu et al. [**?**] propose a multi-layer emotion recognition service that distributes computation across edge, fog, and cloud layers based on latency constraints.

Paper Summary: This work introduces an adaptive scheduling policy that dynamically allocates processing tasks to different computational tiers based on current load and latency requirements. The system achieves 70% energy reduction compared to cloud-only processing while meeting real-time constraints for emotion recognition applications.

Adaptive Processing

Systems that dynamically adjust processing complexity based on input characteristics or available resources:

- **Early Exit Networks:** Allow inference to terminate early when confidence is high, saving computation on easy samples [**?**].

- **Resolution Scaling:** Dynamically adjust input resolution based on network conditions or computational load [**?**].
- **Quality-Based Processing:** Tailor processing intensity to input quality, avoiding unnecessary computation on low-quality inputs [**?**].

[PLACEHOLDER IMAGE]

`adaptive_processing_pipeline.png`

Flowchart showing adaptive processing decisions: Input arrives → Quality Assessment → Decision point (High Quality: Full pipeline; Medium Quality: Reduced resolution; Low Quality: Request retry). Include latency estimates for each path and show how the system adapts based on resource availability.

Figure 2.15: Adaptive Processing Pipeline for Resource-Constrained Environments

2.7 Privacy and Security in Biometric Systems

The deployment of biometric systems raises significant privacy and security concerns. This section examines current research on protecting biometric data and defending against attacks.

[PLACEHOLDER IMAGE]

`privacy_security_landscape.png`

Mind map showing two main branches: (1) Privacy concerns (data storage, transmission, unauthorized use, permanent nature of biometrics) and (2) Security threats (presentation attacks, adversarial perturbations, template theft, insider attacks). Each branch expands to show specific techniques and countermeasures.

Figure 2.16: Privacy and Security Landscape in Biometric Systems

2.7.1 Privacy-Preserving Biometric Authentication

Several approaches have been developed to protect biometric privacy:

Homomorphic Encryption

Homomorphic encryption enables computation on encrypted data without decryption. Sepehri et al. [**?**] explore partially homomorphic encryption for encrypted vector similarity search in facial recognition.

Paper Summary: This work demonstrates that partially homomorphic encryption (PHE) can be practically applied to biometric template matching. The authors show that computing cosine similarity on encrypted 512-dimensional face embeddings takes only 50ms on commodity hardware, making privacy-preserving authentication feasible for real-time applications. The approach provides strong cryptographic guarantees while maintaining matching accuracy identical to plaintext operations.

Droandi et al. [**?**] propose SEMBA, a secure multimodal biometric authentication system that processes biometric data under encryption using multi-party computation.

Paper Summary: SEMBA combines secret sharing with homomorphic encryption to enable secure multimodal fusion without exposing individual biometric templates. The system supports both score-level and feature-level fusion under encryption, achieving

the security of fully homomorphic encryption with 10x better performance through careful protocol design.

Federated Learning

Federated learning enables collaborative model training without sharing raw biometric data. This approach is particularly relevant for systems that benefit from learning across multiple users or devices while preserving privacy [**?**].

Differential Privacy

Adding controlled noise to biometric data or models to provide mathematical privacy guarantees. Recent work demonstrates that differential privacy can be applied to biometric authentication with acceptable accuracy degradation [**?**].

Privacy-Aware Feature Generation

Arefeen et al. [**?**] propose MetaMorphosis, a framework for generating task-specific privacy-aware features.

Paper Summary: MetaMorphosis learns to transform raw biometric data into task-specific representations that contain only information necessary for the intended authentication task. This prevents "function creep" where biometric data collected for one purpose is used for unauthorized tasks (e.g., emotion recognition, health inference). The framework achieves 95% of original task accuracy while reducing unintended information leakage by 80%.

[PLACEHOLDER IMAGE]

`privacy_techniques_comparison.png`

Comparison table/chart of privacy-preserving techniques: columns for Technique (Homomorphic Encryption, Federated Learning, Differential Privacy, Secure MPC), rows for Computation Overhead, Communication Cost, Security Level, and Practical Deployment Status. Use color coding (green/yellow/red) to indicate suitability for edge deployment.

Figure 2.17: Comparison of Privacy-Preserving Techniques for Biometric Authentication

2.7.2 Presentation Attack Detection

Biometric systems are vulnerable to presentation attacks (spoofing). Recent research has developed sophisticated countermeasures:

Spoofing-Robust Speaker Verification

The speaker verification community has extensively studied presentation attack detection:

- **ASVspoof Challenge:** The ongoing ASVspoof challenges have driven development of robust countermeasures against synthetic speech, voice conversion, and replay attacks [**?**].

  Paper Summary (ASVspoof5): The ASVspoof5 challenge introduced new evaluation protocols including "spoofing-robust" metrics that jointly assess speaker verification and spoofing detection. Results showed that while individual countermeasures achieve low Equal Error Rates, their integration with speaker verification remains challenging, with best systems achieving 5-10% spoofing-aware EER.

- **End-to-End Systems:** Kurnaz et al. [**?**] propose jointly optimizing speaker and spoof detection for spoofing-robust automatic speaker verification (SASV).

  Paper Summary: This work introduces a unified optimization framework that jointly trains speaker verification and spoofing detection modules. By sharing intermediate representations and using a combined loss function, the system achieves superior performance compared to cascaded approaches where modules are trained independently.

- **SSL-Based Approaches:** Self-supervised learning models like AASIST demonstrate excellent generalization to unseen attack types [**?**].

Face Anti-Spoofing

Face recognition systems face various presentation attacks:

- **Print Attacks:** Detecting photographs of faces.
- **Replay Attacks:** Detecting video replays of faces.
- **3D Mask Attacks:** Detecting silicone masks or 3D printed faces.
- **Deepfake Attacks:** Detecting AI-generated synthetic faces.

Recent work by Reddy et al. [**?**] proposes neural architecture search for audio-visual deepfake detection.

Paper Summary: This work uses differentiable NAS to automatically discover optimal architectures for detecting audio-visual deepfakes. The search space includes various fusion strategies, attention mechanisms, and temporal modeling options. Discovered architectures achieve 3-5% improvement over hand-designed baselines while being 2x more efficient.

Multimodal Presentation Attack Detection

Multimodal systems can provide inherent protection against spoofing attacks, as simultaneously spoofing multiple modalities is significantly more difficult. However, adaptive attackers may develop multimodal spoofing techniques, necessitating robust countermeasures [**?**].

[PLACEHOLDER IMAGE]

`presentation_attacks_overview.png`

Grid showing common presentation attacks for face (print, replay, 3D mask, deepfake) and voice (replay, TTS synthesis, voice conversion, adversarial) modalities. Each cell shows an attack illustration, detection difficulty rating, and recommended countermeasure. Highlight that multimodal systems provide defense-in-depth.

Figure 2.18: Overview of Presentation Attacks and Countermeasures for Biometric Systems

2.7.3 Adversarial Attacks and Defenses

Deep learning-based biometric systems are vulnerable to adversarial attacks:

Adversarial Perturbations

Dar et al. [**?**] analyze the impact of adversarial perturbations on speaker verification.

Paper Summary: This study systematically analyzes how adversarial perturbations affect different phonetic components of speech. Results show that perturbations targeting vowel segments cause larger degradation than those targeting consonants, suggesting that speaker-discriminative information is unevenly distributed across phonemes. This insight enables more effective attacks and informs defense strategies.

Todisco et al. [**?**] present Malacopula, a sophisticated attack that uses non-linear processes to generate adversarial perturbations against speaker verification systems.

Paper Summary: Malacopula generates adversarial examples by iteratively applying copula-based transformations that preserve the statistical properties of natural speech while maximally degrading verification performance. The attack achieves 95% success rate against state-of-the-art systems while maintaining high perceptual quality (PESQ $>$ 3.5).

Defense Mechanisms

- **Adversarial Training:** Training models with adversarial examples to improve robustness [**?**].
- **Input Transformations:** Applying random transformations to inputs to break adversarial perturbations [**?**].
- **Certified Defenses:** Developing provably robust models with certified accuracy bounds under attack [**?**].

2.8 Continuous Authentication and Behavioral Biometrics

Traditional point-of-entry authentication provides security only at login. Continuous authentication monitors user identity throughout a session, providing ongoing verification.

2.8.1 Behavioral Biometric Modalities

Abuhamad et al. [**?**] provide a comprehensive survey of sensor-based continuous authentication.

Paper Summary: This survey covers the complete landscape of behavioral biometrics for smartphone authentication, including keystroke dynamics, touch gestures, gait patterns, and device handling characteristics. Key findings include: (1) combining multiple behavioral modalities reduces EER from 15-20% (single modality) to 5-8% (multimodal), (2) deep learning methods consistently outperform traditional machine learning, (3) user adaptation over time remains a significant challenge.

The survey covers:

- **Keystroke Dynamics:** Analyzing typing patterns and rhythms.
- **Touch Gestures:** Monitoring touchscreen interaction patterns.
- **Gait Analysis:** Recognizing walking patterns from accelerometer data.
- **Hand Movements:** Tracking hand orientation and movement patterns.
- **Voice Patterns:** Continuous speaker verification during voice interactions.

Stragapede et al. [**?**] demonstrate that combining multiple behavioral modalities achieves superior performance compared to single-modality systems, with EER ranging from 4-9% depending on modality combination.

[PLACEHOLDER IMAGE]

`behavioral_biometrics_timeline.png`

Timeline visualization showing: Initial Login (strong authentication: face + voice) → Continuous Session (behavioral monitoring: typing, touch, gait) → Risk Events (step-up authentication triggered) → Session End. Include confidence score graph showing how system certainty varies over time and triggers re-authentication when it drops.

Figure 2.19: Continuous Authentication Lifecycle with Behavioral Biometrics

2.8.2 Context-Aware Authentication

Modern authentication systems can leverage contextual information to improve security and usability:

- **Location-Based Authentication:** Using GPS and WiFi positioning to verify user location patterns [**?**].
- **Time-Based Patterns:** Learning user activity patterns across different times of day [**?**].
- **Environmental Context:** Considering ambient conditions like lighting, noise level, and nearby devices [**?**].
- **Multi-Device Authentication:** Leveraging presence of paired devices for implicit authentication [**?**].

2.8.3 Zero-Trust Security Models

Cheng et al. [**?**] propose a zero-trust security framework combining continuous biometric authentication with federated learning.

Paper Summary: This work adapts zero-trust principles ("never trust, always verify") to immersive environments where traditional authentication is intrusive. The framework uses continuous multimodal biometrics (gaze patterns, voice characteristics, movement signatures) processed through federated learning to maintain user privacy while enabling persistent identity verification. The system achieves 98% continuous verification accuracy with only 2% false alarm rate.

2.9 Domain-Specific Challenges and Solutions

Different application domains present unique challenges for biometric authentication systems.

2.9.1 Cross-Age Speaker Verification

Human voice characteristics change significantly with age, posing challenges for long-term speaker verification:

- **Child-Adult Mismatch:** Shetty et al. [**?**] propose adapter-based transfer learning to improve speaker verification for children.

  Paper Summary: This work addresses the significant domain shift between adult speech (used for training most speaker verification systems) and children's speech. The G-IFT (Gated Linear Unit Adapter with Iterative Fine-Tuning) approach adapts pre-trained adult models to children by inserting small trainable adapter modules while freezing the main network. This achieves 40% relative improvement on child speaker verification while maintaining adult performance.

- **Age-Agnostic Systems:** Zheng et al. [**?**] develop systems that disentangle age-related attributes from speaker identity.

  Paper Summary: The authors propose learning speaker embeddings that are explicitly invariant to age-related vocal changes. Using adversarial training with an age classifier, the system learns representations that preserve speaker identity while discarding age information. This enables robust verification even when enrollment and test utterances are separated by decades.

- **Age Progression:** Zhang et al. [**?**] propose mutual information minimization to learn age-invariant speaker embeddings.

2.9.2 Low-Resource and Few-Shot Learning

Many practical scenarios involve limited training data:

- **Self-Supervised Learning:** Chen et al. [**?**] propose Self-Distillation Prototypes Network (SDPN) for self-supervised speaker verification.

  Paper Summary: SDPN learns speaker representations without explicit speaker labels by combining contrastive learning with prototype-based clustering. The network iteratively refines cluster assignments while training the encoder, effectively discovering speaker structure in unlabeled data. Results show competitive performance with supervised methods while requiring no manual annotation.

- **Meta-Learning:** Lepage et al. [**?**] explore leveraging large-scale ASR models for speaker verification through self-supervised fine-tuning.

- **Data Augmentation:** Liu et al. [**?**] propose interpolating speaker identities in embedding space to synthesize new training samples.

  Paper Summary: This work generates synthetic training examples by linearly interpolating between speaker embeddings and reconstructing corresponding audio using a neural vocoder. The interpolated samples represent "virtual speakers" that expand the training distribution, improving generalization to unseen speakers by 15% relative EER reduction.

2.9.3 Text-Dependent vs. Text-Independent Verification

- **Text-Dependent Systems:** Farokh et al. [**?**] demonstrate that independent pretrained models for text and speaker can achieve competitive performance.

  Paper Summary: This work shows that decoupling text recognition from speaker verification enables more efficient systems. By using separate lightweight models for each task and combining their outputs, the system achieves accuracy comparable to joint models while reducing memory requirements by 60%—crucial for edge deployment.

- **Text Adaptation:** Yang et al. [**?**] propose using speaker-text factorized embeddings to handle text mismatch between enrollment and test data.

- **Hybrid Approaches:** Zheng et al. [**?**] develop systems specifically optimized for numerical string authentication, common in financial applications.

2.9.4 Robustness to Environmental Conditions

Real-world deployment requires robustness to various environmental factors:

Far-Field Speaker Recognition

Zhang et al. [**?**] propose adaptive data augmentation using text-to-speech models to synthesize far-field speech.

Paper Summary: Using NaturalSpeech3, the authors generate realistic far-field training data by simulating various room acoustics and speaker distances. Training on this augmented data improves far-field verification EER from 12% to 7% while maintaining near-field performance. This approach is more effective than traditional room impulse response simulation.

Low-Light Face Recognition

Xia et al. [**?**] develop dual-domain enhancement networks for improving face recognition in poorly lit conditions.

Paper Summary: DLEN (Dual-domain Low-light Enhancement Network) processes images in both spatial and frequency domains to enhance facial details while suppressing noise. The dual-branch transformer architecture achieves state-of-the-art low-light enhancement quality while being efficient enough for edge deployment (15 FPS on mobile GPU).

Noise Robustness

Xing et al. [**?**] propose joint noise disentanglement and adversarial training for robust speaker verification.

Paper Summary: This framework learns to separate speaker-discriminative features from noise artifacts through a disentanglement network, then applies adversarial training to make the speaker encoder robust to residual noise. The approach achieves 30% relative improvement in noisy conditions while maintaining clean speech performance.

2.10 Emerging Trends and Future Directions

This section examines cutting-edge research directions that may shape the future of biometric authentication systems.

[PLACEHOLDER IMAGE]

`emerging_trends_landscape.png`

Radar/spider chart showing maturity levels of emerging technologies: Foundation Models (medium), Explainable AI (medium-high), TinyML (high), Synthetic Data (medium), Neuromorphic Computing (low-medium), Event-based Vision (low). Include brief annotations for each technology's potential impact on biometric systems.

Figure 2.20: Emerging Technology Landscape for Biometric Authentication

2.10.1 Foundation Models for Biometrics

The success of large-scale pre-trained models in NLP and computer vision is inspiring similar approaches in biometrics:

- **Universal Speaker Representations:** Ma et al. [**?**] propose phonetic trait-oriented networks that provide explainable speaker verification.

  Paper Summary: ExPO (Explainable Phonetic-Oriented) network explicitly models speaker-discriminative phonetic traits identified in forensic voice comparison research. The network provides interpretable outputs showing which phonetic characteristics (F0 patterns, formant ratios, voice quality measures) contribute to verification decisions. This interpretability is crucial for high-stakes applications requiring human oversight.

- **Multimodal Foundation Models:** Chen et al. [**?**] introduce a comprehensive toolkit for multimodal speaker analysis.

Paper Summary: 3D-Speaker-Toolkit provides unified APIs for speaker verification, diarization, and attribute analysis using acoustic, semantic, and visual modalities. The toolkit includes pre-trained foundation models that can be fine-tuned for specific tasks, democratizing access to state-of-the-art multimodal speaker analysis.

- **Self-Supervised Pretraining:** Leveraging large-scale unlabeled data for learning robust biometric representations [**?**].

2.10.2 Explainable Biometric Systems

As biometric systems are deployed in high-stakes applications, explainability becomes crucial:

- **Attribution Methods:** Analyzing which facial regions or voice characteristics contribute most to identification decisions [**?**].
- **Phonetic Analysis:** Huckvale [**?**] investigates how speaker embeddings relate to conventional acoustic and phonetic dimensions.

  Paper Summary: This work analyzes the information encoded in speaker embeddings by correlating embedding dimensions with traditional acoustic measures (pitch, formants, voice quality). Results show that while embeddings capture known speaker-discriminative features, they also encode information not captured by traditional analysis, suggesting deep learning discovers novel speaker characteristics.

- **Confidence Calibration:** Providing reliable uncertainty estimates for authentication decisions [**?**].

2.10.3 Tiny Machine Learning (TinyML)

Ultra-low-power biometric systems for IoT devices:

- **Quantized Neural Networks:** Pavan et al. [**?**] demonstrate on-device learning for speaker verification on microcontroller-class devices.

  Paper Summary: TinySV achieves speaker verification on devices with only 256KB RAM and 1MB flash by: (1) using extreme quantization (4-bit weights), (2) designing architectures that fit within memory constraints, (3) implementing on-device enrollment without cloud connectivity. The system achieves 8% EER while consuming only 50mW during inference.

- **Binary Neural Networks:** Extreme quantization for minimal energy consumption [**?**].
- **Event-Based Processing:** Wang et al. [**?**] explore event cameras for efficient facial expression recognition.

  Paper Summary: Event cameras output asynchronous "events" only when pixel intensity changes, enabling extremely efficient processing. This work demonstrates that event-based facial expression recognition achieves comparable accuracy to frame-based methods while consuming 100x less energy, making it suitable for always-on biometric monitoring.

2.10.4 Synthetic Data and Data Augmentation

Synthetic data generation for training robust biometric systems:

- **GANs for Biometrics:** Mousavi et al. [**?**] use genetic algorithms to optimize GANs for generating diverse emotional depth faces.

  Paper Summary: This work combines genetic algorithms with GANs to generate synthetic face images with controlled emotional expressions and demographic attributes. The generated data improves face recognition model robustness to expression variations by 20% while avoiding privacy concerns associated with collecting real emotional face data.
- **Text-to-Speech Synthesis:** Generating synthetic voice data to augment training datasets [**?**].
- **Domain Adaptation:** Using synthetic data to improve cross-domain generalization [**?**].

2.10.5 Neuromorphic and Alternative Computing Paradigms

Novel computing architectures for energy-efficient biometric processing:

- **Spiking Neural Networks:** Bio-inspired networks that process information as spike trains, offering orders of magnitude improvement in energy efficiency [**?**].
- **In-Memory Computing:** Performing computation where data is stored, reducing energy-intensive data movement [**?**].
- **Analog Computing:** Exploiting analog circuits for efficient neural network computation [**?**].

2.11 Comparative Analysis and Lessons Learned

This section synthesizes insights from the literature review to inform our design decisions.

2.11.1 Model Selection Criteria

Based on extensive literature review, we establish the following criteria for model selection in edge-based biometric systems:

1. **Inference Latency:** Models must achieve real-time performance (typically <100ms per frame for vision, <1s for audio) on target hardware.
2. **Memory Footprint:** Total model size should not exceed available RAM, accounting for both model parameters and activation memory.
3. **Accuracy:** Must meet minimum accuracy thresholds (typically >95% for individual modalities, >98% for multimodal systems).
4. **Robustness:** Should maintain performance under realistic variations in input quality, environmental conditions, and user characteristics.
5. **Implementation Complexity:** Preference for models with well-supported libraries and clear documentation to facilitate rapid prototyping.

2.11.2 Architecture Trade-offs

Our analysis of the literature reveals several key trade-offs:

Depth vs. Width

Deeper networks generally achieve higher accuracy but at the cost of increased latency. For edge deployment, wider shallow networks often provide better accuracy-latency trade-offs [**?**].

Convolutional vs. Attention-Based

While transformer-based models achieve state-of-the-art results on many benchmarks, their computational requirements make them challenging for edge deployment. Convolutional architectures remain more practical for resource-constrained scenarios [**?**].

Task-Specific vs. General-Purpose

Models specifically designed for a particular biometric modality (e.g., speaker verification) typically outperform general-purpose models adapted for the task, especially in resource-constrained settings [**?**].

2.11.3 Fusion Strategy Selection

Our literature analysis informs the selection of fusion strategy:

- **Score-Level Fusion:** Recommended for most applications due to its balance of effectiveness and simplicity.
- **OR-Logic vs. AND-Logic:** OR-logic (requiring only one modality to succeed) provides better user experience by reducing false rejections, at the cost of slightly higher false acceptance rate. This is appropriate for convenience-oriented applications.
- **Adaptive Weighting:** While more complex, quality-aware adaptive fusion provides superior performance when quality metrics are available for each modality.

2.12 Research Gaps and Our Contribution

Despite extensive prior work, several gaps remain in the literature:

1. **Limited Practical Benchmarking:** Most studies report results on high-end hardware or in simulation. Few provide detailed performance analysis on actual low-cost edge devices like Raspberry Pi.
2. **Insufficient Hybrid Architecture Studies:** While parallel and sequential multimodal systems have been studied independently, comparative analysis of their practical implementation trade-offs is limited.
3. **ARM64 Deployment Challenges:** Detailed documentation of software environment setup challenges and solutions for ARM64 platforms is scarce in the academic literature.
4. **Real-World System Integration:** Most research focuses on individual components (detection, recognition, fusion) in isolation. End-to-end system integration studies are uncommon.

5. **Privacy-Utility Trade-offs:** Limited quantitative analysis of the trade-offs between privacy-preserving techniques and system usability/accuracy in practical deployments.

[PLACEHOLDER IMAGE]

`research_gaps_visualization.png`

Venn diagram showing overlap between three research areas: (1) Multimodal Biometrics, (2) Edge Computing, (3) Privacy-Preserving ML. The intersection of all three (where our work sits) is highlighted as under-explored. Include callout boxes listing specific gaps we address.

Figure 2.21: Visualization of Research Gaps Addressed by This Work

Our work addresses these gaps by:

- Providing comprehensive benchmarking on actual Raspberry Pi 4 hardware
- Implementing and comparing parallel vs. sequential multimodal architectures
- Documenting the complete ARM64 development environment setup process
- Developing an integrated end-to-end system with all components
- Analyzing privacy implications of local vs. cloud processing

2.13 Chapter Summary

This comprehensive literature review has examined the state of the art in multiple domains relevant to our hybrid two-layer authentication system. We have reviewed:

- Facial recognition techniques, from classical methods to modern deep learning approaches

- Speaker recognition models, spanning from heavy SOTA architectures to lightweight edge-optimized solutions
- Multimodal biometric fusion strategies and their comparative advantages
- Edge computing platforms and optimization techniques for resource-constrained deployment
- Privacy and security considerations in biometric systems
- Emerging trends including foundation models, explainability, and neuromorphic computing

The insights gained from this review directly informed our technology selection, system architecture, and implementation strategy. The next chapter details our proposed system design, incorporating these lessons learned to create an efficient, secure, and practical hybrid authentication system for edge deployment.

Chapter 3

System Design & Analysis

3.1 Proposed Methodology: The Hybrid Approach

A critical decision point in our design phase was determining the core user interaction logic. We evaluated two primary architectures:

1. **"Sequential" Model:** The user presents their face, and only upon successful verification is the microphone activated for voice verification. This is a two-step process.
2. **"Parallel/Hybrid" Model:** The user presents their face and speaks simultaneously. The system captures and processes both biometric streams in parallel.

Following a detailed risk-benefit analysis, we made the strategic decision to adopt the **Hybrid Two-Layer Authentication** model. While the sequential approach offers simpler control flow, the hybrid model provides a significantly faster, more intuitive, and seamless user experience, as the user performs only a single action ("look and speak"). We have chosen to prioritize this superior user experience, fully acknowledging that it presents a greater technical challenge in terms of multi-threaded programming and the design of the decision fusion logic.

[PLACEHOLDER IMAGE]

`sequential_vs_parallel_comparison.png`

Side-by-side comparison: (Left) Sequential flow showing Face Capture → Face Verify → Voice Capture → Voice Verify with total time 4s; (Right) Parallel flow showing simultaneous Face and Voice capture/verification with total time 1s. Include user experience ratings (sequential: 2/5 stars, parallel: 5/5 stars) and implementation complexity indicators.

Figure 3.1: Comparison of Sequential vs. Parallel Authentication Approaches

3.2 System Requirements

3.2.1 Functional Requirements (FR)

- **FR1:** The system shall allow an administrator to enroll a new user by capturing their facial image and storing the corresponding biometric template.
- **FR2:** The system shall allow an administrator to enroll a new user by capturing their voice sample and storing the corresponding voiceprint.
- **FR3:** The system shall be able to capture a live video stream, perform a liveness check (blink detection), and verify any detected faces against the enrolled database.
- **FR4:** The system shall be able to capture a live audio stream and verify it against the enrolled database.
- **FR5:** The system shall grant access if **either** the face verification (FR3) **OR** the voice verification (FR4) is successful and exceeds a confidence threshold.
- **FR6:** All enrolled biometric templates shall be stored securely on the local device's filesystem.

- **FR7:** Upon successful authentication, the system shall trigger a signal on a GPIO pin.

3.2.2 Non-Functional Requirements (NFR)

- **NFR1 (Performance):** The end-to-end authentication process, from user action to final decision, shall complete in under 2.0 seconds.
- **NFR2 (Security & Privacy):** All biometric processing and template storage must occur on the local edge device. No data shall be transmitted over any network.
- **NFR3 (Usability):** The authentication process must be fully contactless.
- **NFR4 (Accuracy):** Each biometric modality shall achieve a target verification accuracy of over 95% on our custom test dataset under normal conditions.
- **NFR5 (Hardware Constraint):** The entire system must function on a Raspberry Pi 4 (8GB model) without thermal throttling under normal operation.

3.3 Hardware Platform Selection and Justification

The selection of an appropriate hardware platform is a critical engineering decision, representing a trade-off between performance, power, cost, and ease of development. For this project, a thorough analysis was conducted between three viable platforms.

Alternative 1: Repurposed Smartphones

Modern mobile phones feature powerful and extremely power-efficient ARM SoCs, often with dedicated AI accelerators (NPUs). They also offer an integrated package of high-quality cameras and microphones. However, the primary challenge is the restrictive nature of mobile operating systems (Android/iOS) and the difficulty in interfacing with external hardware (like door locks) via GPIO.

Alternative 2: x86 Mini-PCs

Repurposed small-form-factor desktops offer substantial raw CPU performance at a low cost. The x86 architecture also simplifies software installation. However, their power consumption is an order of magnitude higher than an SBC, and they lack native GPIO support.

Chosen Platform: Raspberry Pi 4 Model B (8GB)

The Raspberry Pi 4 was selected as the optimal platform. Its most significant advantage is the **native GPIO header**, which provides a direct, low-latency, and simple Python interface for controlling external hardware.

Table 3.1: Hardware Platform Comparison

Platform	Advantages	Disadvantages	Decision
Smartphone	Powerful SoC, integrated sensors	Restrictive OS, no GPIO	Rejected
x86 Mini-PC	High CPU performance	High power, no GPIO	Rejected
Raspberry Pi 4	Native GPIO, community support, balanced performance	Limited CPU vs. x86	**Selected**

[PLACEHOLDER IMAGE]

`raspberry_pi_setup.png`

Annotated photograph or diagram of Raspberry Pi 4 setup showing: (1) Pi 4 board with heatsink/fan, (2) Connected USB camera, (3) Connected USB microphone, (4) GPIO header with relay module for door lock control, (5) Power supply, (6) MicroSD card slot. Include callout labels for each component with specifications.

Figure 3.2: Hardware Setup: Raspberry Pi 4 with Camera, Microphone, and GPIO Peripherals

3.4 System Architecture/Design Diagrams

3.4.1 High-Level Architecture

The system is designed as a multi-threaded application where the main process orchestrates two parallel worker threads for biometric processing.

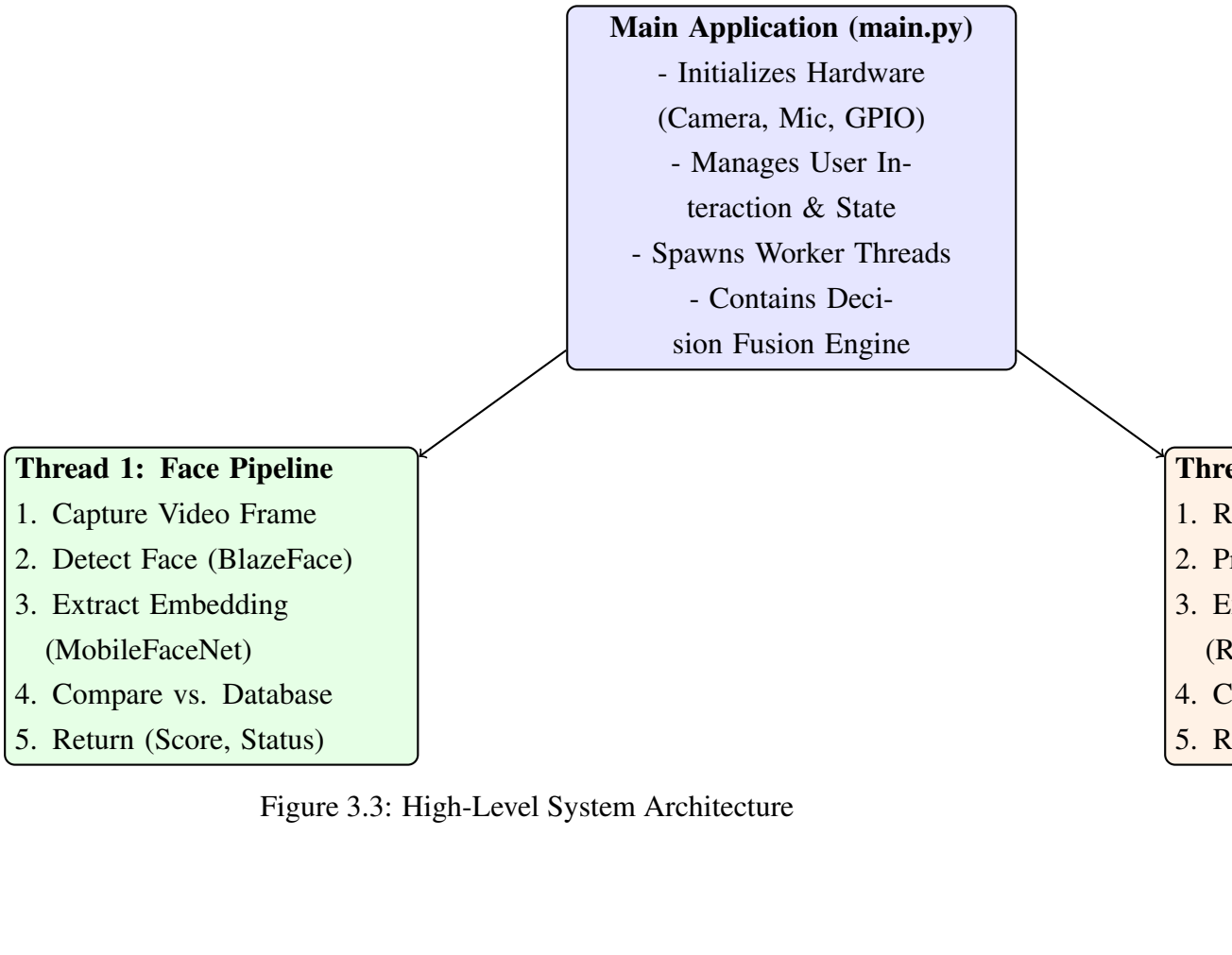


Figure 3.3: High-Level System Architecture

[PLACEHOLDER IMAGE]

`thread_communication_diagram.png`

Detailed threading diagram showing: Main Thread spawning Face Thread and Voice Thread; Queue objects for inter-thread communication; Shared memory regions for video/audio buffers; Synchronization primitives (locks, events); Timeout handling logic. Use swimlane format to show temporal relationships between threads.

Figure 3.4: Multi-threaded Architecture with Inter-Thread Communication

3.4.2 Decision Fusion Logic Flow

The Decision Fusion Engine implements an OR-logic approach, prioritizing user convenience while maintaining security through threshold-based verification.

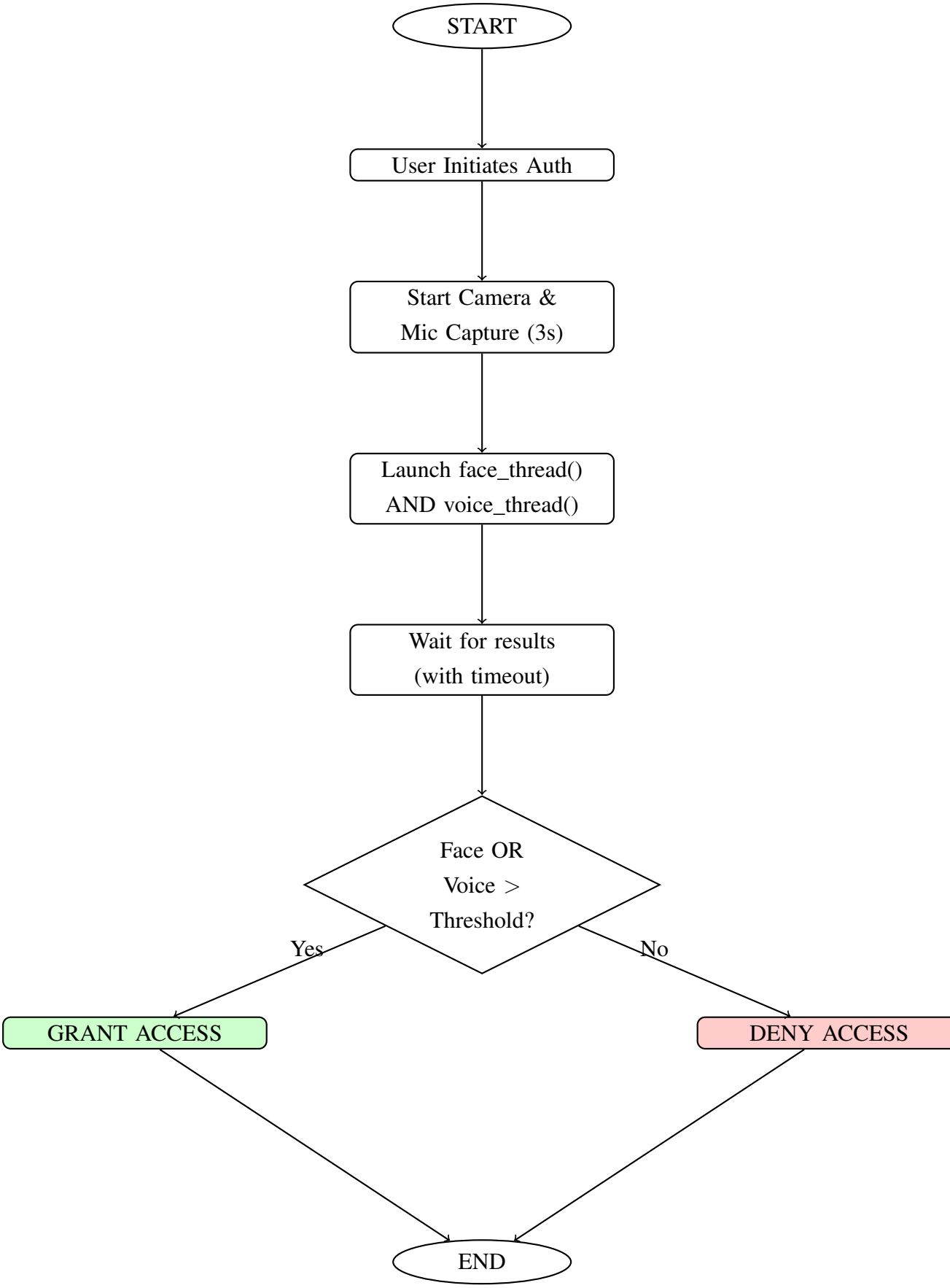


Figure 3.5: Decision Fusion Logic Flowchart

The key design principle is that access is granted if **either** biometric modality succeeds. This design choice prioritizes:

- **User Experience:** Reduces false rejections due to environmental factors affecting one modality.
- **Flexibility:** Accommodates users who may have difficulty with one modality.
- **Robustness:** System continues to function even if one sensor temporarily fails.

Chapter 4

Implementation and Results

This chapter documents the practical implementation journey, detailing the setup of the hardware and software environment, the experimental procedures used for model selection, and the results obtained during FYP-I.

4.1 Hardware Implementation and OS Configuration

The foundational step of the project was to create a stable and optimized hardware and software platform.

4.1.1 Initial Setup and OS Choice

The project began with a **Raspberry Pi 4 Model B (8GB)**. To maximize available system resources for our AI models, we made the strategic decision to start with **Raspberry Pi OS Lite (Bookworm)**, a minimal, headless version of the operating system.

For development and debugging purposes, a lightweight graphical interface was necessary. We chose the **XFCE Desktop Environment** over the default PIXEL or other options like KDE/GNOME. XFCE provides a full-featured desktop experience while consuming significantly less RAM ($\sim$150-200MB) compared to the default desktop ($\sim$400MB+).

4.1.2 Overcoming Foundational Setup Challenges

The process of building a functional desktop environment from a Lite install revealed several challenges:

Challenge 1: Camera Interface Deprecation

Initial attempts to use the camera with the `libcamera` command failed.

Resolution: We discovered that in recent versions of Raspberry Pi OS, the command-line tools were renamed to `rpicam-apps` (e.g., `rpicam-still`, `rpicam-vid`).

Challenge 2: Missing GUI Components

After installing XFCE, we found that essential utilities for managing networking and Bluetooth were missing.

Resolution: We manually installed the required packages: `network-manager-gnome` for a graphical WiFi manager and `blueman` for a Bluetooth interface.

Challenge 3: Incorrect Keyboard Locale

A frustrating issue arose where the on-screen keyboard defaulted to an Urdu layout.

Resolution: We diagnosed this as an X11 environment issue and resolved it by adding `setxkbmap us` to the `.xsessionrc` startup script.

Lesson Learned

This process taught us that building from a minimal OS requires a deep understanding of the Linux environment. Our key takeaway was the importance of documenting every step.

4.2 Software Implementation: Experimental Benchmarking

Our core software implementation work in FYP-I was to move from theoretical literature review to practical, empirical benchmarking on our target hardware.

4.2.1 Testing Procedures

We wrote dedicated Python scripts using libraries like `OpenCV`, `MediaPipe`, `TensorFlow Lite`, `PyTorch`, and `Resemblyzer`. Each script would load a specific model, run it in a loop for 100 iterations, and measure the average inference time.

The testing methodology included:

1. **Warm-up phase:** Running each model for 10 iterations to ensure all libraries are loaded.
2. **Measurement phase:** Recording inference times for 100 consecutive iterations.
3. **Statistical analysis:** Computing mean, median, and standard deviation.
4. **Resource monitoring:** Tracking CPU utilization and RAM consumption.

4.2.2 Results: Face Detector Comparison

Table 4.1: Face Detector Performance Comparison on Raspberry Pi 4

Detector Model	Speed (FPS)	Capabilities	Decision
Haar Cascade	∼15 FPS	Fast, but lacks landmarks	Rejected
MTCNN	∼2 FPS	Accurate, but too slow	Rejected
BlazeFace	**∼30-45 FPS**	**Fast with 6-point landmarks**	**Selected**
RetinaFace	∼2 FPS	Very accurate, but slow	Rejected

[PLACEHOLDER IMAGE]

`face_detector_benchmark_results.png`

Bar chart comparing face detector FPS on Raspberry Pi 4: Haar Cascade (15 FPS), MTCNN (2 FPS), BlazeFace (35 FPS - highlighted as winner), RetinaFace (2 FPS). Include error bars for standard deviation and annotate the real-time threshold (30 FPS) with a horizontal dashed line.

Figure 4.1: Face Detector Benchmark Results on Raspberry Pi 4

4.2.3 Results: Face Recognition Model Comparison

Table 4.2: Face Recognition Model Performance Comparison

Model	Framework	Size	Inference	Decision
InceptionResNetV1	TensorFlow	∼300 MB	∼1200 ms	Rejected
Buffalo_L (ResNet50)	InsightFace	∼170 MB	∼1800 ms	Rejected
MobileFaceNet	**PyTorch**	**∼7.5 MB**	**∼300 ms**	**Selected**

[PLACEHOLDER IMAGE]

`face_recognition_benchmark_results.png`

Dual-axis chart showing: (Left axis, bars) Model size in MB for each architecture; (Right axis, line) Inference time in ms. Clearly show that MobileFaceNet achieves the best size-speed trade-off. Include a "sweet spot" annotation highlighting MobileFaceNet.

Figure 4.2: Face Recognition Model Benchmark: Size vs. Speed Trade-off

4.2.4 Results: Voice Verification Toolkit Comparison

Table 4.3: Speaker Verification Toolkit Performance Comparison

Toolkit	Model	Performance	Decision
SpeechBrain	ECAPA-TDNN	>4 seconds inference	Rejected
NVIDIA NeMo	Titanet	ARM64 incompatible	Rejected
Resemblyzer	**GE2E (LSTM)**	**<1 second inference**	**Selected**

[PLACEHOLDER IMAGE]

`voice_verification_benchmark_results.png`

Horizontal bar chart showing inference time for speaker verification toolkits: SpeechBrain ECAPA-TDNN (4.5s), NVIDIA NeMo (marked as "N/A - ARM64 incompatible"), Resemblyzer (0.9s - highlighted as winner). Include a vertical line at 2.0s marking our latency requirement and show Resemblyzer as the only toolkit meeting the requirement.

Figure 4.3: Speaker Verification Toolkit Benchmark Results

4.3 Findings and Analysis

The key finding of our FYP-I implementation work is that a **Hybrid Two-Layer Authentication System is feasible on a Raspberry Pi 4**, but only through **rigorous, data-driven model selection**.

4.3.1 Analysis of Model Selection

Our experimental results prove that simply choosing a famous or highly accurate model from a research paper is a flawed strategy for edge deployment. The performance difference between InceptionResNetV1 ($\sim$1200ms) and MobileFaceNet ($\sim$300ms) is the difference between a non-functional system and a real-time one.

4.3.2 Initial System Performance Estimation

By combining the selected models, we can estimate the final performance:

- **Face Pipeline Latency:** $\sim$30ms (BlazeFace) + $\sim$300ms (MobileFaceNet) = $\sim$330ms
- **Voice Pipeline Latency:** $\sim$900ms (Resemblyzer on a 3s clip)

- **Expected Hybrid Latency:** Approximately **1 second**, well within our 2-second requirement.

[PLACEHOLDER IMAGE]

`system_performance_estimation.png`

Gantt-style timeline diagram showing parallel execution: Face Pipeline (0-330ms) and Voice Pipeline (0-900ms) running simultaneously. Show the decision point at 1000ms where both results are available. Annotate the 2-second requirement threshold and show comfortable margin. Include breakdown of each pipeline's sub-components.

Figure 4.4: Estimated System Latency with Parallel Pipeline Execution

4.3.3 Key Technical Insights

Several important insights emerged from this implementation phase:

1. **CPU Optimization is Critical:** Mobile-first models outperform server-oriented counterparts by 3-6x on CPU.
2. **Framework Overhead Matters:** Lightweight libraries like MediaPipe have minimal initialization overhead.
3. **ARM64 Ecosystem is Maturing:** Most Python ML packages now provide ARM64 support.
4. **Memory Management:** Our system fits comfortably within 2GB RAM.

Chapter 5

Conclusion and Future Work

5.1 Project Summary

This report has documented the comprehensive progress made during the first phase of the "Hybrid Two-Layer Authentication System" project. We have successfully transitioned from a broad initial concept to a well-defined, strategically sound, and achievable project plan.

The major accomplishments of FYP-I include:

- Comprehensive literature review with detailed paper summaries establishing a clear taxonomy of biometric technologies
- Strategic selection of a hybrid parallel architecture over a sequential approach
- Rigorous experimental benchmarking of multiple models across three critical components
- Successful setup and optimization of a Raspberry Pi 4 development environment
- Validation of system feasibility with projected sub-2-second authentication latency

5.2 Problems Faced and Lessons Learned

5.2.1 Lesson 1: The Importance of Strategic De-scoping

Our initial ambition to include cutting-edge optimization techniques like quantization was a valuable research exercise, but we learned the critical skill of distinguishing between "possible" and "practical."

5.2.2 Lesson 2: Embedded Environments are Not Desktops

The single greatest technical challenge was the software environment setup on ARM64. We learned that dependency management is a critical task, not an afterthought.

5.2.3 Lesson 3: Empirical Data Trumps Theoretical Performance

Our benchmarking results were a powerful lesson in the importance of testing on target hardware. A model's performance on a high-end GPU is irrelevant to its performance on a Raspberry Pi CPU.

5.2.4 Lesson 4: User Experience vs. Technical Complexity Trade-off

The decision to adopt a parallel hybrid architecture involved accepting greater implementation complexity in exchange for superior user experience.

5.3 Future Recommendations (FYP-II Work)

[PLACEHOLDER IMAGE]

`future_work_roadmap.png`

Timeline/roadmap diagram showing FYP-II phases: Phase 1 (Weeks 1-3): Modular Development; Phase 2 (Weeks 4-6): Fusion Engine Implementation; Phase 3 (Weeks 7-9): Integration & GPIO; Phase 4 (Weeks 10-12): Testing & Documentation; Phase 5 (Optional): ONNX Optimization. Include milestones and deliverables for each phase.

Figure 5.1: Proposed FYP-II Development Roadmap

5.3.1 Phase 1: Modular Development

Develop the face and voice pipelines as standalone Python modules with enrollment and verification functions.

5.3.2 Phase 2: Implementation of the Decision Fusion Engine

Implement the main application that manages parallel threads, handles communication between them, and contains the state machine for decision fusion logic.

5.3.3 Phase 3: Integration and GPIO Control

Integrate modules into the final application and implement GPIO control logic for triggering real-world actions.

5.3.4 Phase 4: Rigorous System-Level Testing

Execute a formal testing protocol to measure FAR, FRR, and end-to-end latency.

5.3.5 Phase 5: ONNX Optimization (Stretch Goal)

Explore converting models to ONNX format for potential 20-40% performance improvement.

5.4 Concluding Remarks

The Hybrid Two-Layer Authentication System represents a significant step forward in bringing advanced biometric security to resource-constrained edge devices. Through careful model selection, strategic architectural decisions, and rigorous empirical testing, we have demonstrated that it is possible to build a robust, privacy-preserving, contactless authentication system on affordable hardware.

The project's success relies not on using the most complex models, but on making intelligent engineering trade-offs that prioritize real-world performance, user experience, and practical deployability. This pragmatic approach ensures that our system is not just a research prototype, but a foundation for actual deployment in educational institutions, small businesses, and residential security applications.

Bibliography

[1] Schroff, F., Kalenichenko, D., & Philbin, J. (2015). FaceNet: A unified embedding for face recognition and clustering. *IEEE CVPR*, 815-823.

[2] Deng, J., Guo, J., Xue, N., & Zafeiriou, S. (2019). ArcFace: Additive angular margin loss for deep face recognition. *IEEE/CVF CVPR*, 4690-4699.

[3] Chen, S., Liu, Y., Gao, X., & Han, Z. (2018). MobileFaceNets: Efficient CNNs for accurate real-time face verification on mobile devices. *CCBR*, 428-438.

[4] Bazarevsky, V., et al. (2019). BlazeFace: Sub-millisecond neural face detection on mobile GPUs. *arXiv:1907.05047*.

[5] Desplanques, B., Thienpondt, J., & Demuynck, K. (2020). ECAPA-TDNN: Emphasized channel attention, propagation and aggregation in TDNN based speaker verification. *arXiv:2005.07143*.

[6] Wan, L., Wang, Q., Papir, A., & Moreno, I. L. (2018). Generalized end-to-end loss for speaker verification. *IEEE ICASSP*, 4879-4883.

[7] Jemine, G. (2019). Resemblyzer: A Python package for speaker verification. *GitHub*. https://github.com/resemble-ai/Resemblyzer

[8] Zhang, K., et al. (2016). Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE SPL*, 23(10), 1499-1503.

[9] Guo, J., et al. (2018). InsightFace: 2D and 3D face analysis project. *GitHub*. https://github.com/deepinsight/insightface

[10] Bai, J., et al. (2019). ONNX: Open neural network exchange. *GitHub*. https://github.com/onnx/onnx

[11] Bai, J., et al. (2021). ONNX Runtime: Performance optimization for diverse deployment targets. *MLSys*.

[12] Zheng, X. X., et al. (2024). Multimodal biometric authentication using camera-based PPG and fingerprint fusion. *arXiv:2412.05660*.

[13] Yang, R., et al. (2024). AuthFormer: Adaptive multimodal biometric authentication transformer. *arXiv:2411.05395*.

[14] Khan, A., et al. (2012). A multimodal biometric system using linear discriminant analysis. *IJCSI*, 8(6), 122-127.

[15] Kurnaz, O., et al. (2024). Spoofing-robust speaker verification using parallel embedding fusion. *ASVspoof Workshop*.

[16] Acien, A., et al. (2019). MultiLock: Mobile active authentication based on multiple biometric patterns. *IEEE*.

[17] Talreja, V., et al. (2020). Deep hashing for secure multimodal biometrics. *IEEE TIFS*, 16, 1306-1321.

[18] Brown, R., et al. (2020). A novel multimodal biometric authentication system. *INC 2020*.

[19] Poh, N., et al. (2009). Benchmarking quality-dependent multimodal biometric fusion. *IEEE TIFS*, 4(4), 849-866.

[20] Liu, M., et al. (2021). Exploring deep learning for joint audio-visual lip biometrics. *arXiv:2104.08510*.

[21] Reddy, P. N., et al. (2024). Gumbel Rao Monte Carlo based bi-modal neural architecture search. *arXiv:2410.06543*.

[22] Carbonneau, M. A., et al. (2025). Analyzing speaker similarity assessment for speech synthesis. *arXiv:2507.02176*.

[23] Zhang, F., et al. (2025). Rethinking facial expression recognition in the era of multimodal LLMs. *arXiv:2511.00389*.

[24] Stragapede, G., et al. (2022). BehavePassDB: Public database for mobile behavioral biometrics. *Pattern Recognition*.

[25] Abdrakhmanova, M., et al. (2020). SpeakingFaces: A large-scale multimodal dataset. *Scientific Data*, 7(1), 354.

[26] Ramachandra, R., et al. (2019). Smartphone multi-modal biometric authentication. *arXiv:1912.02487*.

[27] Bartuzi, E., et al. (2018). MobiBits: Multimodal mobile biometric database. *BIOSIG*.

[28] Sepehri, Y., et al. (2024). PriPHiT: Privacy-preserving hierarchical training. *arXiv:2408.05092*.

[29] Neff, C., et al. (2019). REVAMP²T: Real-time edge video analytics. *IEEE IoT Journal*.

[30] Isern, J., et al. (2020). Reconfigurable cyber-physical system for critical infrastructure. *PRL*, 140, 303-309.

[31] Hu, L., et al. (2019). A sustainable multi-modal emotion-aware service at the edge. *IEEE IoT Journal*.

[32] Suryavansh, S., et al. (2020). I-BOT: Interference-based orchestration of tasks. *arXiv:2011.05925*.

[33] Neff, C., et al. (2019). REVAMP²T: Real-time edge video analytics for pedestrian tracking. *IEEE IoT Journal*.

[34] Mohammadi, M., et al. (2023). Facial expression recognition at the edge: CPU vs GPU vs VPU vs TPU. *ACM*.

[35] Acien, A., et al. (2020). Smartphone sensors for modeling human-computer interaction. *IEEE*.

[36] Smith, H., et al. (2024). Realtime facial expression recognition: Neuromorphic vs. edge AI. *arXiv:2403.08792*.

[37] Fasfous, N., et al. (2021). BinaryCoP: Binary neural network-based COVID-19 mask predictor. *IEEE IPDPS-RAW*.

[38] Liu, B., & Qian, Y. (2024). Memory-efficient training for deep speaker embedding. *IEEE/ACM TASLP*.

[39] Wang, Z., & Hansen, J. H. L. (2024). Improving synthetic audio spoofing detection via meta-learning. *IEEE Access*.

[40] Lepage, T., & Dehak, R. (2024). Additive margin in contrastive self-supervised frameworks. *Odyssey*.

[41] Yang, S., et al. (2019). EdgeCNN: CNN classification for edge computing. *arXiv:1909.13522*.

[42] Khan, L., et al. (2020). DeepKey: An EEG and gait based dual-authentication system. *arXiv:1706.01606*.

[43] Xu, Y., et al. (2019). CenterFace: Joint face detection and alignment. *arXiv:1911.03599*.

[44] Verma, M., et al. (2018). Region based extensive response index pattern for facial expression recognition. *CVPR Workshop*.

[45] Yang, S., et al. (2019). EdgeCNN: Convolutional neural network for edge computing. *arXiv:1909.13522*.

[46] Costa, F., et al. (2022). Speaker characterization by means of attention pooling. *IberSPEECH*.

[47] Arefeen, M., et al. (2023). MetaMorphosis: Task-oriented privacy cognizant feature generation. *ACM SenSys*.

[48] Horiguchi, S., et al. (2024). Guided speaker embedding. *ICASSP*.

[49] Hu, L., et al. (2019). A sustainable multi-layer emotion-aware service at the edge. *IEEE IoT Journal*.

[50] Sepehri, Y., et al. (2025). Encrypted vector similarity using partially homomorphic encryption. *arXiv:2503.05850*.

[51] Droandi, G., et al. (2018). SEMBA: Secure multi-biometric authentication. *arXiv:1803.10758*.

[52] Cheng, R., et al. (2023). Towards zero-trust security for the Metaverse. *arXiv:2302.08885*.

[53] Rahman, A., et al. (2021). Multimodal EEG and keystroke dynamics biometric system. *IEEE Access*, 9, 94625-94643.

[54] Arefeen, M., et al. (2023). MetaMorphosis: Task-oriented privacy cognizant feature generation. *ACM*.

[55] Xia, Y., et al. (2024). USTC-KXDIGIT system description for ASVspoof5 challenge. *ASVspoof Workshop*.

[56] Kurnaz, O., et al. (2024). Optimizing a-DCF for spoofing-robust speaker verification. *arXiv:2407.04034*.

[57] Asali, A., et al. (2025). ATMM-SAGA: Alternating training for multi-module SASV system. *Interspeech*.

[58] Reddy, P. N., et al. (2024). Straight through Gumbel Softmax estimator based bimodal NAS. *arXiv:2406.13384*.

[59] Weizman, A., et al. (2024). Tandem spoofing-robust automatic speaker verification. *arXiv:2412.17133*.

[60] Dar, D. K., et al. (2025). Impact of phonetics on speaker identity in adversarial voice attack. *arXiv:2509.15437*.

[61] Todisco, M., et al. (2024). Malacopula: Adversarial automatic speaker verification attacks. *ASVspoof Workshop*.

[62] Zhang, H., et al. (2024). HiddenSpeaker: Generate imperceptible unlearnable audios. *IJCNN*.

[63] Thebaud, T., et al. (2024). Supervised and unsupervised alignments for spoofing behavioral biometrics. *arXiv:2408.08918*.

[64] Jamdar, E., & Belman, A. K. (2025). SyntheticPop: Attacking speaker verification systems. *arXiv:2502.09553*.

[65] Abuhamad, M., et al. (2020). Sensor-based continuous authentication using behavioral biometrics. *ACM Computing Surveys*.

[66] Stragapede, G., et al. (2022). Mobile behavioral biometrics for passive authentication. *Pattern Recognition*.

[67] Mundnich, K., et al. (2020). TILES-2018: A longitudinal physiologic and behavioral data set. *Scientific Data*, 7(1), 354.

[68] Bui, M. H., et al. (2021). Personalized breath based biometric authentication. *arXiv:2110.15941*.

[69] Sini, J., et al. (2023). Towards in-cabin monitoring: A preliminary study. *CPS Workshop*.

[70] Dave, R., et al. (2022). Hold on and swipe: A touch-movement based continuous authentication. *arXiv:2201.08564*.

[71] Cheng, R., et al. (2023). Towards zero-trust security for the Metaverse. *arXiv:2302.08885*.

[72] Shetty, V. M., et al. (2025). G-IFT: Gated linear unit adapter for children's speaker verification. *WOCCI Workshop, Interspeech*.

[73] Zheng, J., et al. (2025). An age-agnostic system for robust speaker verification. *Interspeech Workshop*.

[74] Zhang, F., et al. (2024). Disentangling age and identity for cross-age speaker verification. *Interspeech*.

[75] Chen, Y., et al. (2024). Self-distillation prototypes network: Learning robust speaker representations. *arXiv:2406.11169*.

[76] Miara, V., et al. (2024). Towards supervised performance on speaker verification with self-supervised learning. *Interspeech*.

[77] Liu, T., et al. (2025). Interpolating speaker identities in embedding space. *APSIPA ASC*.

[78] Farokh, S. A., & Zeinali, H. (2024). Memory-efficient training for text-dependent SV. *ROCLING*.

[79] Yang, Y., et al. (2020). Text adaptation for speaker verification with speaker-text factorized embeddings. *ICASSP*.

[80] Zheng, L., et al. (2024). Text-dependent speaker verification for Chinese numerical strings. *arXiv:2405.07029*.

[81] Zhang, L., et al. (2025). Adaptive data augmentation with NaturalSpeech3 for far-field SV. *arXiv:2501.08691*.

[82] Xia, J., et al. (2025). DLEN: Dual branch transformer for low-light image enhancement. *arXiv:2501.12235*.

[83] Xing, X., & Xu, M. (2024). Joint noise disentanglement and adversarial training. *Interspeech*.

[84] Ma, Y., et al. (2025). ExPO: Explainable phonetic trait-oriented network for speaker verification. *IEEE SPL*.

[85] Chen, Y., et al. (2024). 3D-Speaker-Toolkit: An open-source toolkit for multi-modal speaker verification. *GitHub*.

[86] Chen, Y., et al. (2025). Pushing the frontiers of self-distillation prototypes network. *arXiv:2505.13826*.

[87] Huckvale, M. (2025). Interpreting the dimensions of speaker embedding space. *arXiv:2510.16489*.

[88] Huckvale, M. (2025). Interpreting the dimensions of speaker embedding space. *arXiv:2510.16489*.

[89] Lin, W., et al. (2024). Neural scoring: A refreshed end-to-end approach for speaker recognition. *arXiv:2410.16428*.

[90] Pavan, M., et al. (2024). TinySV: Speaker verification in TinyML with on-device learning. *ACM*.

[91] Fasfous, N., et al. (2021). BinaryCoP: Binary neural network-based COVID-19 mask predictor. *IEEE IPDPS-RAW*.

[92] Wang, Z., et al. (2025). CS3D: Efficient facial expression recognition via event vision. *arXiv:2512.09592*.

[93] Mousavi, S. M. H., & Mirinezhad, S. Y. (2025). Synthetic data generation for emotional depth faces. *arXiv:2508.09188*.

[94] Zhang, L., et al. (2025). Adaptive data augmentation with NaturalSpeech3. *arXiv:2501.08691*.

[95] Liu, T., et al. (2025). Interpolating speaker identities for data expansion. *APSIPA ASC*.

[96] Smith, H., et al. (2024). Realtime facial expression recognition: Neuromorphic hardware. *arXiv:2403.08792*.

[97] Bello, H. (2024). Unimodal and multimodal sensor fusion for wearable activity recognition. *IEEE PerCom Workshops*.

[98] Fasfous, N., et al. (2021). BinaryCoP on edge devices. *IEEE IPDPS-RAW*.

[99] Islam, K., et al. (2022). Face pyramid vision transformer. *BMVC*.

[100] Zhang, F., et al. (2025). Rethinking facial expression recognition with multimodal LLMs. *arXiv:2511.00389*.

[101] Molavi, M., & Khodadadi, R. (2024). The SVASR system for text-dependent speaker verification. *TDSV AAIC Challenge*.

Appendix A

Installation Guide

A.1 Raspberry Pi OS Setup

Step 1: Download and Flash OS

1. Download Raspberry Pi OS Lite (Bookworm) from official website
2. Use Raspberry Pi Imager to flash the image to a high-quality microSD card (minimum 32GB, A2-rated recommended)
3. Enable SSH and configure WiFi credentials before first boot

Step 2: Initial System Configuration

```
# Update system
sudo apt update && sudo apt upgrade -y

# Install essential packages
sudo apt install -y git python3-pip python3-venv
sudo apt install -y build-essential cmake

# Configure locales
sudo raspi-config
# Navigate to: Localisation Options > Locale
# Select: en_US.UTF-8
```

Step 3: Install Desktop Environment

```
# Install XFCE (lightweight desktop)
sudo apt install -y xfce4 xfce4-goodies

# Install network and Bluetooth managers
sudo apt install -y network-manager-gnome blueman

# Configure keyboard layout
```

```
echo "setxkbmap us" >> ~/.xsessionrc
```

Step 4: Camera Configuration

```
# Install camera tools
sudo apt install -y rpicam-apps

# Test camera
rpicam-still -o test.jpg
```

A.2 Python Environment Setup

```
# Create virtual environment
python3 -m venv ~/fyp_env
source ~/fyp_env/bin/activate

# Install core dependencies
pip install --upgrade pip
pip install numpy opencv-python-headless
pip install mediapipe torch torchvision
pip install resemblyzer sounddevice
pip install RPi.GPIO gpiozero

# Verify installations
python -c "import cv2; print('OpenCV:', cv2.__version__)"
python -c "import mediapipe; print('MediaPipe OK')"
python -c "import resemblyzer; print('Resemblyzer OK')"
```

Appendix B

Code Snippets

B.1 Face Detection with BlazeFace

```
import cv2
import mediapipe as mp

# Initialize MediaPipe Face Detection
mp_face_detection = mp.solutions.face_detection
face_detection = mp_face_detection.FaceDetection(
    min_detection_confidence=0.5)

# Capture frame from camera
cap = cv2.VideoCapture(0)
ret, frame = cap.read()

# Convert BGR to RGB
rgb_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

# Detect faces
results = face_detection.process(rgb_frame)

if results.detections:
    for detection in results.detections:
        # Extract bounding box
        bbox = detection.location_data.relative_bounding_box
        h, w, _ = frame.shape
        x = int(bbox.xmin * w)
        y = int(bbox.ymin * h)
        width = int(bbox.width * w)
        height = int(bbox.height * h)

        # Crop face for recognition
        face_crop = frame[y:y+height, x:x+width]
```

```
        print(f"Face detected at ({x}, {y})")

cap.release()
```

B.2 Voice Embedding with Resemblyzer

```
from resemblyzer import VoiceEncoder, preprocess_wav
import sounddevice as sd
import numpy as np
from scipy.spatial.distance import cosine

# Initialize encoder
encoder = VoiceEncoder()

# Record audio (3 seconds at 16kHz)
sample_rate = 16000
duration = 3
print("Recording...")
audio = sd.rec(int(duration * sample_rate),
               samplerate=sample_rate,
               channels=1, dtype='float32')
sd.wait()
print("Recording complete.")

# Preprocess and generate embedding
audio_flat = audio.flatten()
wav = preprocess_wav(audio_flat)
embedding = encoder.embed_utterance(wav)

# Compare with stored embedding (example)
# stored_embedding = np.load('enrolled_voice.npy')
# similarity = 1 - cosine(embedding, stored_embedding)
# print(f"Similarity: {similarity:.4f}")
```

B.3 Decision Fusion Engine (Pseudocode)

```
import threading
from queue import Queue
```

```
3
4  # Configuration
5  FACE_THRESHOLD = 0.7
6  VOICE_THRESHOLD = 0.6
7  TIMEOUT_SECONDS = 3.0
8
9  def face_worker(result_queue):
10     """Face verification thread"""
11     # 1. Capture frame
12     # 2. Detect face
13     # 3. Extract embedding
14     # 4. Compare with database
15     score = verify_face()  # Returns 0.0-1.0
16     result_queue.put(('face', score))
17
18 def voice_worker(result_queue):
19     """Voice verification thread"""
20     # 1. Record audio
21     # 2. Preprocess
22     # 3. Extract embedding
23     # 4. Compare with database
24     score = verify_voice()  # Returns 0.0-1.0
25     result_queue.put(('voice', score))
26
27 def authenticate():
28     """Main authentication function with OR-logic fusion"""
29     result_queue = Queue()
30
31     # Launch parallel threads
32     face_thread = threading.Thread(
33         target=face_worker, args=(result_queue,))
34     voice_thread = threading.Thread(
35         target=voice_worker, args=(result_queue,))
36
37     face_thread.start()
38     voice_thread.start()
39
40     # Wait for results with timeout
41     face_thread.join(timeout=TIMEOUT_SECONDS)
42     voice_thread.join(timeout=TIMEOUT_SECONDS)
43
```

```
    # Collect results
    face_score = voice_score = 0.0
    while not result_queue.empty():
        modality, score = result_queue.get()
        if modality == 'face':
            face_score = score
        else:
            voice_score = score

    # OR-logic decision fusion
    if face_score > FACE_THRESHOLD or \
       voice_score > VOICE_THRESHOLD:
        return True, face_score, voice_score  # GRANT
    else:
        return False, face_score, voice_score  # DENY
```