SILHOUETTE-BASED GAIT RECOGNITION USING LOCAL TEMPORAL AGGREGATION

A PROJECT REPORT

Submitted by

ADARSH ACHUTHAN

(2023178015)

A report of the project
submitted to the Faculty of

INFORMATION AND COMMUNICATION ENGINEERING

in partial fulfillment
for the award of the degree
of

MASTER OF COMPUTER APPLICATIONS

DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY
COLLEGE OF ENGINEERING, GUINDY
ANNA UNIVERSITY
CHENNAI 600 025

APRIL 2025

ANNA UNIVERSITY

CHENNAI - 600 025

BONAFIDE CERTIFICATE

Certified that this project report titled **SILHOUETTE-BASED GAIT RECOGNITION USING LOCAL TEMPORAL AGGREGATION** is the bonafide work of **ADARSH ACHUTHAN(2023178015)** who carried out project work under my supervision. Certified further that to the best of my knowledge and belief, the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or an award was conferred on an earlier occasion on this or any other candidate.

PLACE:CHENNAI
DATE:

DR. T. MALA
PROFESSOR
PROJECT GUIDE
DEPARTMENT OF IST, CEG
ANNA UNIVERSITY
CHENNAI 600025

COUNTERSIGNED

Dr. S. SWAMYNATHAN
HEAD OF THE DEPARTMENT
DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY
COLLEGE OF ENGINEERING, GUINDY
ANNA UNIVERSITY
CHENNAI 600025

ABSTRACT

This abstract provides an overview of gait recognition, highlighting its principles, techniques, applications, and challenges, and underscores its importance as a non-intrusive biometric authentication technology with promising real-world applications. Gait recognition, a subset of biometric identification, focuses on analyzing and recognizing human walking patterns for authentication and identification purposes. Unlike traditional biometric methods like fingerprint or iris recognition, gait recognition does not require close proximity or physical contact with the subject, making it suitable for unobtrusive surveillance and monitoring applications. The human gait, characterized by the unique motion patterns of body parts during walking, reflects individual physical and behavioral traits such as body shape, size, and walking style. Gait recognition systems typically utilize various sensing modalities, including video cameras, depth sensors, and wearable devices, to capture and analyze gait features. Feature extraction techniques, such as silhouette-based methods, model-based approaches, and deep learning-based feature learning, play a crucial role in gait recognition by extracting discriminative features from gait sequences. These features are then used to train machine learning or deep learning models for gait classification and identification. Gait recognition finds applications in diverse fields, including surveillance and security, healthcare monitoring, biometric authentication, and forensic analysis. Despite its potential advantages, challenges such as viewpoint variation, clothing changes, and environmental factors pose significant obstacles to the accuracy and robustness of gait recognition systems.

ACKNOWLEDGEMENT

It is my privilege to express my deepest sense of gratitude and sincere thanks to **Dr. T. Mala**, Professor, Project Guide, Department of Information Science and Technology, College of Engineering, Guindy, Anna University, for her constant supervision, encouragement, and support in my project work. I greatly appreciate the constructive advice and motivation that was given to help me advance my project in the right direction.

I am grateful to **Dr. S. Swamynathan**, Professor and Head, Department of Information Science and Technology, College of Engineering, Guindy, Anna University, for providing us with the opportunity and necessary resources to do this project.

I would also wish to express my deepest sense of gratitude to the Members of the Project Review Committee: **Dr. T. Mala**, Professor, **Dr. N. Thangaraj**, Associate Professor, **Dr. E. Uma**, Associate Professor and **Ms. R. L. Jasmine**, Teaching Fellow and Project Coordinator, Department of Information Science and Technology, College of Engineering, Guindy, Anna University, for their guidance and useful suggestions that were beneficial in helping me improve my project.

I also thank the faculty members and non-teaching staff members of the Department of Information Science and Technology, Anna University, Chennai for their valuable support throughout the course of my project work.

Adarsh Achuthan
(2023178015)

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

$+, \vee, \cup$	Disjunction operator
$-, \neg, \sim$	Negation operator
$\rightarrow$	Conditional operator
$\leftrightarrow$	Biconditional operator
x_a	Anchor
x_b	Positive
x_n	Negative
β	Beta
γ	Gamma
ε	Epsilon
λ	Lambda
μ_B	Batch Mean
σ_B^2	Batch Variance
p	Parameter
b	Bias Term
BN	Batch Normalisation
C	Channels
d	Distance Metric
FC	Fully Connected
GeM	Generalised Mean Pooling
GLFE	Global and Local Feature Extraction
H	Height
LTA	Local and Temporal Aggregation
W	Weighted Matrix
X	Input Feature Vector
Y	Transformed Output Feature

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

Gait, considered a soft biometric, offers significant advantages over hard biometrics such as facial recognition, iris scans, or fingerprints. It refers to the unique way a person walks or moves their limbs during motion. Unlike hard biometrics, gait can be observed from a distance without the need for the person's cooperation and is difficult to disguise. This makes it particularly useful for applications like surveillance and forensic identification. However, its use also raises privacy concerns and risks of misuse.

Despite its advantages, gait presents challenges as a biometric.It is influenced by factors such as carried items, clothing, and surface type, and even different types of footwear can significantly alter a person's gait. The main challenge lies in extracting unique features that are invariant to these influences.

1.2 PROBLEM STATEMENT

Traditionally, gait recognition methods have relied on extracting features from silhouettes obtained through background subtraction in video sequences. However, this approach can introduce artifacts on the contour, especially in real- world environments with clutter and rapid changes. Recent advancements in deep learning-based pose estimation have overcome some of these challenges by generating keypoints that are robust against occlusion, clutter, and background changes. This has rendered appearance-based methods obsolete and paved the way for a new generation of model-based gait

recognition, which relies on keypoints and harks back to early descriptions of gait.

1.3 OBJECTIVES

The proposed method, which integrates silhouette models extracted from the same input video gait frames, aims to enhance the refinement of frame-level features crucial for accurate gait recognition. By leveraging both types of information, the approach offers several advantages, including unobtrusive identification, distance independence, and reduced sensitivity to lighting conditions. This comprehensive fusion of silhouette and skeletal data enables a more robust and reliable representation of gait patterns.

To validate the effectiveness of the method, extensive experiments were conducted on the widely used gait dataset: CASIA-B. The CASIA-B dataset is a benchmark dataset commonly used in computer vision tasks, especially in face recognition research. These datasets contain a diverse range of gait sequences captured under various conditions, allowing for comprehensive evaluation of the method's performance across different scenarios.

The experimental results demonstrate a significant improvement over the State-Of-The-Art performance, particularly on the largest gait dataset, CASIA-B. This improvement underscores the effectiveness of the method in capturing and leveraging the subtle nuances of gait dynamics encoded in silhouette representations. Furthermore, the method exhibits strong temporal modeling capabilities, enabling effective capture and representation of the dynamic nature of gait patterns over time. The primary objectives of this project are:

Obtain Silhouette frames from the dataset:

This step involves extracting key information from the video frames. Silhouette frames capture the shape and outline of the subject's body, typically by segmenting the subject from the background. Silhouette frames represent the underlying structure of the subject's body, often extracted using techniques like pose estimation to identify key joints and limbs.

Silhouette based gait recognition task:

This involves segmenting human subjects from video frames to extract their silhouettes, focusing on the spatial shape and contour of the individual during motion. Relevant features such as gait dynamics, shape, and texture are extracted from the silhouettes, followed by classification techniques to distinguish between individuals based on their unique gait patterns.

Integrate features to identify gait pattern uniquely:

Integrating silhouette features in gait recognition involves leveraging the complementary information provided by the body shape and structural dynamics. By combining silhouette-based shape information, this approach enhances the accuracy and robustness of gait pattern identification, enabling more reliable recognition of individuals across varying conditions and viewpoints.

1.4 ORGANIZATION OF THE REPORT

Chapter 1: Introduction – Provides an overview of the project, background, motivation, problem statement, objectives, and project scope.

Chapter 2: Literature Review – This chapter explains about the literature survey made on existing system, issue with the existing system.

Chapter 3: System Design – This chapter provides the detailed system design of the proposed work along with the proposed methodology and implementation.

Chapter 4: Implementation Details – This chapter explains the algorithms used and descriptive details about the modules.

Chapter 5: Results and Aanalysis – This chapter provides the experimental results and analysis obtained during the project.

Chapter 6: Conclusion and Future Work – This chapter concludes the project along with its future scope.

References: This section includes all the references and citations used throughout the report, following standard citation formats.

CHAPTER 2

LITERATURE SURVEY

2.1 INTRODUCTION

In recent years, appearance-based approaches using silhouette representations have dominated gait recognition. Model-based approaches, which utilize pose estimation and human skeletons as gait features, played a minor role initially but have gained traction recently. Silhouette-based methods are used to set the state-of-the-art for gait recognition.

2.2 SILHOUETTE-BASED GAIT RECOGNITION

The following section discusses about the literatures on silhouette based gait recognition system and the various problems it has been associated with to find the optimized solution for the problem.

Chao H et al. [1] proposes a novel part-based model GaitPart and get two aspects effect of boosting the performance: On the one hand, Focal Convolution Layer, a new applying of convolution, is presented to enhance the fine-grained learning of the part-level spatial features. On the other hand, the Micro-motion Capture Module (MCM) is proposed and there are several parallel MCMs in the GaitPart corresponding to the pre-defined parts of the human body, respectively. It is worth mentioning that the MCM is a novel way of temporal modeling for gait task, which focuses on the short-range temporal features rather than the redundant long-range features for cycle gait. It also captures dynamic aspects of gait over time but overfitting, trained on limited data.

Chao Fan et al. [2] achieves an average rank-1 accuracy of 96.1 percent on the CASIA-B (A benchmark dataset) gait dataset and an accuracy of 87.9 percent on the OU-MVLP (One-Class Unsupervised Multi-View Learning with Partially Labeled data) gait dataset. Under various complex scenarios, this model also exhibits a high level of robustness. It achieves accuracies of 90.8 and 70.3 percent on CASIA-B under bag-carrying and coat-wearing walking conditions respectively, significantly outperforming the best existing methods. Moreover, this proposed method maintains a satisfactory accuracy even when only small numbers of frames are available in the test samples; for example, it achieves 85.0 percent on CASIA-B even when using only 7 frames.

M. Daffa I et al. [3] proposes a novel approach that exploits periodic gait information, named Gait Period Set (GPS), which divides the gait period into several phases and ensembles the gait phase features to refine frame-level features. Then, features from different phases are aggregated into a video-level feature. Moreover, the refined frame-level features are aggregated as the refined gait phase features with higher quality, which can be used to re-refine the frame-level features. Hence, it upgrades the GPS into the Iterative Gait Period Set (IGPS) to iteratively refine the frame-level features. The results of extensive experiments on prevailing gait recognition datasets validate the effectiveness of the GPS and IGPS modules and demonstrate that the proposed method achieves state-of-the-art performance.

Alireza et al. [4] proposes a novel taxonomy made up of four separate dimensions namely body representation, temporal representation, feature representation, and neural architecture, to help characterize and organize the research landscape and literature in this area. Following this proposed taxonomy, a comprehensive survey of gait recognition methods using deep learning is presented with discussions on their performances, characteristics, advantages, and limitations. A comprehensive overview of breakthroughs and

recent developments in gait recognition with deep learning, and cover broad topics including datasets, test protocols, state-of-the-art solutions, challenges, and future research directions is presented.

Ning C. et al. [5] proposes a hybrid silhouette body representation for gait recognition. Inspired by the success of pose estimation approaches, It adopts recent state-of-the-art pose estimator to obtain robust body skeletons from RGB images. Besides, a novel fusion strategy is proposed to integrate silhouette and skeleton information into a single compact silhouette-skeleton representation for gait. And then we prove that heatmap for joints achieve better performance than the other skeleton representation. Extensive experiments on CASIA-B show that the proposed approach essentially boosts performance of both set-based and sequence-based gait recognition solution. In addition, it outperforms GaitPart by 1.2, 0.6 and 2.8, in NM (normal), BG (bag), and CL (clothing) conditions, respectively.

Likai W et al. [6] is a simple yet effective two-branch network is proposed in this paper, which contains a CNN-based branch taking silhouettes as input and a GCN-based (Graph Convolutional Network) branch taking skeletons as input. In addition, for better gait representation in the GCN-based branch, it present a fully connected graph convolution operator to integrate multi-scale graph convolutions and alleviate the dependence on natural joint connections. Also, it deploys a multi-dimension attention module named STC-Att (Spatio-Temporal Context Attention) to learn spatial, temporal and channel-wise attention simultaneously. The experimental results on CASIA-B (A benchmark dataset) show that our method achieves state-of-the-art performance in various conditions.

2.3 LIMITATIONS OF EXISTING SYSTEMS

Despite notable advancements, silhouette-based gait recognition systems still face several limitations. They often struggle with overfitting due to limited training data, leading to poor generalization in unseen scenarios. Furthermore, inaccuracies in silhouette extraction and joint detection can degrade performance, particularly under challenging conditions such as occlusions or drastic viewpoint changes.

These systems can also be computationally intensive, given the use of complex deep learning architectures and multi-modal fusion strategies, posing challenges for real-time deployment. Moreover, many models are sensitive to specific datasets or conditions, limiting their applicability to real-world environments. The heavy reliance on large annotated datasets for training further hampers progress in domains where labeled data is scarce.

2.4 CONCLUSION

The literature review showcases the progression of gait recognition techniques, with silhouette-based methods, particularly those leveraging deep learning models like CNNs and GCNs, achieving remarkable improvements. Techniques such as GaitPart, GEINet, and cross-view methods demonstrate significant accuracy gains, while innovations like temporal pooling and triplet loss enhance feature robustness. These advancements pave the way for more accurate, reliable, and deployable gait recognition systems, with promising applications in security, healthcare, and surveillance.

CHAPTER 3

SYSTEM DESIGN

This chapter discusses the detailed system architecture of the various models in the project.

3.1 ARCHITECTURE OF GAIT RECOGNITION SYSTEM

The system architecture as seen in Figure 3.1 shows the overall design of the system.

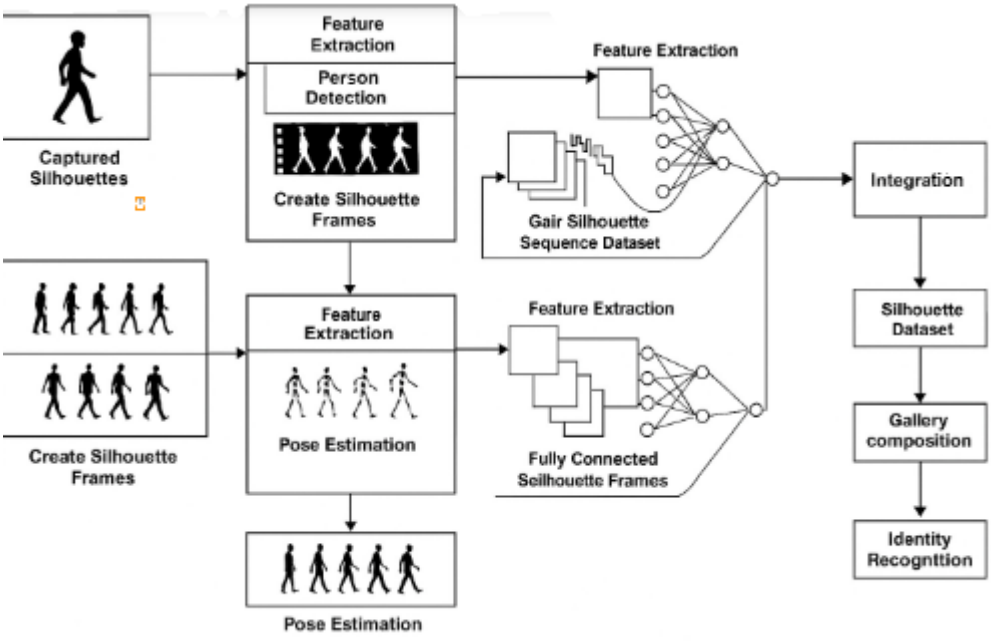


Figure 3.1: Gait Recognition System

The system architecture presented in Figure 3.1 outlines a gait recognition pipeline consisting of two main stages: **Feature Extraction** and **Feature Mapping**.

In the Feature Extraction stage, input gait sequences (silhouettes of individuals) undergo silhouette creation, followed by a 3×3×3 convolutional layer to extract spatial features. These extracted features are then processed using Local and Temporal Aggregation (LTA), which helps capture motion patterns across frames. The final extracted features are refined using Global and Local Feature Extraction (GLFE) to obtain meaningful gait representations.Next, in the Feature Mapping stage, the extracted features undergo Temporal Pooling, which aggregates information over the sequence to ensure robustness to variations in gait patterns. This is followed by Generalized Mean (GeM) pooling, which refines feature representations before being passed through Fully Connected (FC) layers with Batch Normalization (BN) to ensure stable training. The system is trained using a combination of Cross Entropy Loss (for classification) and Triplet Loss (for learning discriminative gait embeddings), which together enhance recognition performance. This structured approach allows the model to effectively extract and map gait features for accurate person identification.

3.2 DATA ACQUISITION

The data acquisition process involves gathering information from the Chinese Academy of Sciences Institute of Automation - Dataset B(CASIA-B) which is a benchmark dataset for Gait Recognition that provides a standardized platform for comparing the performance of different approaches in gait recognition tasks, such as person identification, activity recognition, and anomaly detection.

3.2.1 Dataset Composition

Table 3.1: Details of Dataset CASIA-B

Attribute	Description
Database Name	CASIA-B
Creation Date	January 2005
Number of Subjects	124
Number of Views	11
Variations Considered	View angle, Clothing, Carrying condition

CASIA-B dataset presented in Table 3.1 provides a comprehensive set of sequences capturing different views and conditions for gait recognition experiments. It is a widely used gait dataset and composed of 124 subjects. For each of the 124 subjects the dataset contains 11 views (0°, 18°, . . . , 180°) and 3 waking conditions. The walking conditions are normal (NM) (6 sequences per subject), walking with a bag (BG) (2 sequences per subject), and wearing a coat or a jacket (CL) (2 sequences per subject). Summed up, each subject contains $11 \times (6 + 2 + 2) = 110$ sequences.

3.3 LOCAL AND TEMPORAL AGGREGATION

Local and Temporal Aggregation (LTA) refers to a technique used in video analysis and temporal sequence modeling to aggregate information over short, localized time intervals. It helps in capturing short-term dependencies while reducing noise and redundancy in sequential data. LTA is commonly used in areas such as action recognition, motion analysis, and gait analysis. This aggregation helps smooth out noise, improves temporal consistency, and enhances the network's ability to recognize actions with subtle or rapid movements. By leveraging temporal context, LTA provides a richer and more stable feature representation, making it particularly useful in scenarios where fine-grained motion patterns are essential for accurate classification. LTA captures short-term motion dynamics by integrating features over a local temporal window.

3.4 GLOBAL AND LOCAL FEATURE EXTRACTION

Global and Local Feature Extraction (GLFE) is a technique used to enhance feature representation by capturing both global and local spatial-temporal information in gait recognition. The global features encode high-level structural patterns of the gait, such as overall posture and stride, while the local features capture fine-grained motion details like limb movements and joint displacements. By integrating both perspectives, GLFE ensures a more robust and discriminative gait representation. This dual-level extraction helps the model handle variations caused by clothing, carrying conditions, or occlusions, making gait recognition more effective in real-world scenarios.

3.5 TEMPORAL POOLING

Temporal pooling is a technique used in sequence-based models to aggregate features over time, enhancing robustness to variations in motion dynamics. In gait recognition, temporal pooling consolidates frame-wise feature representations into a single, compact descriptor, ensuring that important motion patterns are preserved while reducing redundancy. By summarizing temporal information, the model becomes less sensitive to fluctuations such as variations in walking speed, occlusions, or minor pose differences. Common pooling strategies include average pooling, which smooths features over time, and Generalized Mean (GeM) pooling, which emphasizes discriminative features by adaptively weighting frame contributions. Temporal pooling helps improve gait recognition by providing a stable and distinctive feature representation for person identification.

3.6 GENERALIZED MEAN (GEM) POOLING

Generalized Mean (GeM) pooling is an advanced pooling technique that adaptively aggregates spatial features by applying a learnable exponent parameter p. Unlike traditional pooling methods like Max Pooling (which retains only the strongest activations) or Average Pooling (which computes the mean of all activations), GeM pooling provides a flexible balance between these two extremes. Given a feature map X with dimensions C×H×W, where C is the number of channels,H is the height,W is the width,the GeM pooling operation is defined as given in equation 3.1.

$$Y_{\text{GeM}} = \left(\frac{1}{W} \sum_{i=1}^{W} (X_i)^p \right)^{\frac{1}{p}} \tag{3.1}$$

where p is a learnable parameter that controls the level of pooling, when p=1, GeM behaves like Average Pooling, when p→, GeM behaves like Max Pooling.

3.7 CLASSIFICATION LAYER

The Fully Connected (FC) layer is a dense neural network layer where each neuron is connected to every neuron in the previous layer. In the context of gait recognition, FC layers are used in the final feature mapping stage to refine and transform extracted gait features before classification or comparison. Given an input feature vector X and weight matrix W, the FC layer computes the output as given in equation 3.2.

$$Y = WX + b \tag{3.2}$$

where X is the input feature vector, W is the weight matrix ,b is the bias term,Y is the transformed output feature.

Batch Normalization (BN) is a technique used to stabilize training and accelerate convergence by normalizing input distributions across batches. For a given mini-batch B, BN normalizes each feature x_i as given in the equation 3.3.

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}} \tag{3.3}$$

where μ_B is the batch mean , σ_B^2 is the batch variance , ε is a small constant for numerical stability, x_i is the normalized value.

$$y_i = \gamma \hat{x}_i + \beta \tag{3.4}$$

In equation 3.4, γ and β allow the network to retain the original feature distribution if necessary.

3.8 LOSS CALCULATION

The loss calculation in the given architecture is designed to optimize gait recognition by using a combination of Cross-Entropy Loss and Triplet Loss. These two losses work together to enhance both classification performance and feature discriminability in the learned gait representations.

3.8.1 Cross-Entropy Loss

This loss function is widely used in classification tasks. It ensures that the extracted gait features are correctly mapped to the corresponding identity labels. Given a set of training samples (x_i,y_i), where x_i represents the input gait feature and y_i is the corresponding class label, the softmax function assigns probabilities to each class. The Cross-Entropy Loss is defined as defined in equation 3.5.

$$\mathcal{L}_{CE} = -\sum_{i=1}^{N} y_i \log(\hat{y}_i) \tag{3.5}$$

where N is the number of training samples, y_i is the true label (one-hot encoded), is the predicted probability for class y_i.

This loss encourages the model to correctly classify each gait sequence by minimizing the difference between the predicted and actual class labels.

3.8.2 Triplet Loss

To enhance feature separability, Triplet Loss is used, which focuses on learning a well-structured embedding space where similar gait features are pulled together, and dissimilar ones are pushed apart. It operates on triplets of samples:

- Anchor (x_a): A sample from a specific identity.

- Positive (x_b): Another sample from the same identity.

• Negative(x_n) : A sample from a different identity.

The Triplet Loss is formulated as given in equation 3.6

$$\mathcal{L}_T = \sum_{i=1}^{N} \max\left(d(x_a, x_p) - d(x_a, x_n) + \alpha, 0\right) \tag{3.6}$$

where d(x,y) is the distance metric (e.g., Euclidean distance) between two feature embeddings. is a margin that ensures a minimum separation between positive and negative pairs. This loss function encourages the model to reduce the intra-class distance (anchor-positive pair) and increase the inter-class distance (anchor-negative pair), making the gait features more discriminative.

Final Loss Function:

To jointly optimize classification and feature embedding, the total loss function combines both Cross-Entropy Loss and Triplet Loss as given in equation 3.7.

$$\mathcal{L} = \mathcal{L}_{CE} + \lambda \mathcal{L}_T \tag{3.7}$$

where λ is a balancing parameter that controls the contribution of Triplet Loss. This combined objective ensures that the model effectively classifies gait patterns while maintaining feature distinctiveness for better recognition.

CHAPTER 4

IMPLEMENTATION OF THE PROJECT

This chapter discusses the algorithms of the various models and actions involved in the project.

4.1 PREPROCESSING

The implementation of the gait recognition system begins with the preprocessing of input gait sequences. These sequences are captured as silhouettes from dataset frames, which are then normalized to a standard size. The preprocessing stage is crucial as it ensures consistency across different gait sequences, allowing for more robust feature extraction. The initial step involves the application of convolutional layers to extract shallow gait features, which serve as the foundation for further processing. These convolutional layers are designed to capture both spatial and temporal characteristics of the gait, ensuring that no critical motion-related details are lost. The system incorporates Local and Temporal Aggregation (LTA), a technique that replaces traditional spatial pooling layers to maintain a high spatial resolution while reducing the temporal redundancy of gait sequences. LTA helps in preserving essential structural details while efficiently summarizing temporal motion variations, leading to improved feature representation as given in algorithm 4.1.

Algorithm 4.1 Pseudocode for Preprocessing

```
Input: Raw gait silhouette sequences
Output: Preprocessed gait silhouettes
procedure PREPROCESSGAIT(silhouette_sequences)
    for each frame in silhouette_sequences do
        Convert frame to grayscale
        Apply background subtraction to isolate the moving person
        Perform noise filtering to remove small artifacts
        Extract silhouette by thresholding the processed frame
        Normalize the silhouette size to a fixed resolution
    end for
    return preprocessed silhouettes
end procedure
```

4.2 GLOBAL AND LOCAL FEATURE EXTRACTION (GLFE)

Once the preliminary features are extracted, the system employs a Global and Local Feature Extractor (GLFE) to refine the feature representations. The GLFE module utilizes specialized Global and Local Convolutional (GLConv) layers, which integrate both global gait appearance and local movement details into a single discriminative representation. The global feature extraction focuses on capturing the overall shape and structure of the gait silhouette, while the local feature extraction enhances the fine-grained details such as limb movements and joint displacements. By leveraging both feature types, the GLFE module improves recognition accuracy, even under conditions such as clothing variations or changes in carrying conditions. To enhance the effectiveness of feature representation, two types of GLConv layers, GLConvA and GLConvB, are implemented. GLConvA applies element-wise addition to combine global and local feature maps, whereas GLConvB concatenates these features along the spatial dimension, providing a more expressive representation as given in algorithm 4.2.

Algorithm 4.2 Pseudocode for Global and Local Feature Extraction

```
Input: Preprocessed gait silhouettes
Output: Extracted feature representations
procedure EXTRACTFEATURES(silhouettes)
    Initialize convolutional neural network (CNN) layers
    for each silhouette in silhouettes do
        Apply convolution operation to extract spatial features
        Use activation function (ReLU) to introduce non-linearity
        Apply Local Temporal Aggregation (LTA) to reduce redundancy
    end for
    return extracted feature maps
end procedure
```

4.3 TEMPORAL POOLING AND FEATURE MAPPING

To handle gait sequences of varying lengths, the system employs Temporal Pooling to aggregate the extracted features into a fixed-size representation. Traditional pooling methods like max pooling and average pooling are commonly used, but in this implementation, a more advanced pooling technique known as Generalized Mean (GeM) pooling is introduced. Unlike fixed pooling strategies, GeM pooling allows the system to learn an optimal pooling strategy through training, thereby dynamically adjusting how features are aggregated. This flexibility enhances the robustness of the feature representation and ensures that the most relevant gait characteristics are retained. Following temporal pooling, the extracted gait features undergo fully connected (FC) layers with batch normalization, which further refine the feature space and prepare it for classification as mentioned in algorithm 4.3 .

Algorithm 4.3 Pseudocode for Temporal Pooling and Feature Mapping

```
Input: Extracted feature maps
Output: Global-local combined feature representations
procedure GLFE(features)
    for each feature map in features do
        Apply global convolution to extract holistic gait appearance
        Divide feature map into local regions
        for each local region do
            Apply local convolution to extract detailed gait patterns
        end for
        Combine global and local features using GLConv layers
    end for
    return global-local feature representation
end procedure
```

4.4 LOSS FUNCTION AND MODEL OPTIMIZATION

In algorithm 4.4, the system is trained using a combination of Cross-Entropy Loss and Triplet Loss, ensuring both classification accuracy and feature discriminability. The Cross-Entropy Loss is responsible for optimizing the network to correctly classify individuals based on their gait patterns. It is formulated as given in equation 3.5. where represents the true class label, and is the predicted probability of that class. This loss function encourages the model to assign high confidence to the correct gait identity while penalizing incorrect classification. In addition to classification, Triplet Loss is introduced to improve the discriminability of feature embeddings. The Triplet Loss optimizes the embedding space such that samples from the same individual (anchor and positive) are pulled closer together, while samples from different individuals (negative) are pushed apart. It is defined in equation 3.2 where d(x,y) represents the distance between two feature vectors, and is a margin ensuring a minimum separation between different classes. The total loss function combines both loss terms as defined in equation 3.7 .

Algorithm 4.4 Pseudocode for Loss Computation

Input: Predicted labels, true labels, and feature embeddings
Output: Loss value for optimization
procedure COMPUTELOSS(predictions, labels, embeddings)
    Compute cross-entropy loss:
    $\mathcal{L}_{CE} = -\sum_{i=1}^{N} y_i \log(\hat{y}_i)$
    Compute triplet loss:
    $\mathcal{L}_T = \sum_{i=1}^{N} \max\left(d(x_a, x_p) - d(x_a, x_n) + \alpha, 0\right)$
    Combine loss terms:
    $\mathcal{L} = \mathcal{L}_{CE} + \lambda \mathcal{L}_T$
    return Loss $\mathcal{L}$
end procedure

4.5 TRAINING AND TESTING

During the training phase in algorithm 4.5, the gait sequences are fed into the model in mini-batches, where each batch consists of multiple gait sequences from different individuals. A Batch All (BA) sampling strategy is used to efficiently compute triplet loss across all possible triplet combinations in a batch. The model is optimized using the Adam optimizer, with a learning rate schedule that initially starts at 1e-4 and gradually decreases during training. Training is conducted on public datasets such as CASIA-B where data augmentation techniques such as random cropping and horizontal flipping are applied to improve generalization. During the testing phase in algorithm 4.6 , gait sequences are divided into two sets: gallery set and probe set. The gallery set consists of known gait samples stored in the system, while the probe set contains new gait samples that need to be identified. The similarity between the probe and gallery features is computed using Euclidean distance, and individuals are identified based on a Rank-1 accuracy metric. The proposed system achieves state-of-the-art performance, significantly outperforming previous approaches, particularly in challenging scenarios such as cross-view recognition and clothing variations

Algorithm 4.5 Training Phase for Gait Recognition

```
Input: Gait sequences from silhouette dataset
Output: Trained gait recognition model with optimized feature representations
procedure TRAINING_PHASE(gait_sequences, labels)
    Initialize model parameters
    for epoch = 1 to total_epochs do
        for each mini_batch in gait_sequences do
            features = EXTRACT_FEATURES(mini_batch)
            gl_features = GLFE(features)
            triplet_loss = COMPUTE_TRIPLET_LOSS(gl_features, labels)
            OPTIMIZE_MODEL(triplet_loss) using Adam optimizer
        end for
        Adjust learning rate based on schedule
    end for
    return trained_model
end procedure
```

Algorithm 4.6 Testing Phase for Gait Recognition

```
Input: Gait sequences from silhouette dataset
Output: Trained gait recognition model with optimized feature representations
procedure TESTING_PHASE(trained_model, gallery_set, probe_set)
    gallery_features = EXTRACT_FEATURES(gallery_set)
    probe_features = EXTRACT_FEATURES(probe_set)
    for each probe in probe_features do
        similarity_scores = COMPUTE_SIMILARITY(probe, gallery_features) using Euclidean distance
        predicted_label = IDENTIFY_PERSON(similarity_scores)
    end for
    return Rank-1 Accuracy
end procedure
```

4.6 SUMMARY

The implementation begins with preprocessing, where raw gait silhouettes are normalized and cleaned using grayscale conversion, background subtraction, noise filtering, and size normalization. Key shallow features are extracted using convolutional layers enhanced with Local and Temporal Aggregation (LTA) to preserve motion details.

In the Global and Local Feature Extraction (GLFE) stage, the system employs GLConv layers to extract both global gait structure and local motion details, enhancing recognition performance even under clothing or view variations. The extracted features are then aggregated through Generalized Mean (GeM) pooling, a flexible technique that learns the optimal pooling strategy. These features are further refined via fully connected layers and feature mapping.

For training, a hybrid loss function combining Cross-Entropy Loss and Triplet Loss ensures both classification accuracy and embedding discriminability. Finally, the system is trained using mini-batches and Batch All sampling, optimized with the Adam optimizer. Evaluation involves matching probe and gallery sets using Euclidean distance, achieving strong performance on public datasets like CASIA-B particularly in challenging cross-view and clothing variation scenarios.

CHAPTER 5

RESULTS AND ANALYSIS

This chapter explains about the various modules on the project and their results.

5.1 IMPLEMENTATION ENVIRONMENT

The implementation environment for this gait recognition system is a local machine setup using Python as the primary programming language, taking advantage of several key libraries to handle various stages of data processing and model training. TensorFlow, with its integrated Keras API, is employed as the deep learning framework to build, compile, and train the Convolutional Neural Network (CNN) model. OpenCV (cv2) is used for image handling tasks such as reading and resizing grayscale images, which are crucial steps in standardizing input data for the neural network.

NumPy supports efficient numerical operations, particularly for managing arrays of image data, while scikit-learn's LabelEncoder assists in transforming categorical subject identifiers into numerical labels suitable for classification tasks. The environment assumes a Windows-based file system, evident from the dataset path, and all directories and file structures are managed through Python's os module. Images are preprocessed to ensure consistent dimensions (128×128 pixels) and normalized to improve model convergence during training. The implementation is executed in a CPU or GPU-enabled Python environment—potentially an IDE like VS Code, Jupyter Notebook, or a standard Python script runner—depending on the hardware configuration of the local system.

Batch processing and epoch-based learning are used during training, and model performance is monitored in terms of loss and accuracy. The design of the implementation also allows user interactivity, as it prompts the user to select specific subjects for training instead of defaulting to all available subjects, offering flexibility and modular experimentation. This localized and modular setup makes it ideal for iterative development and testing in gait-based biometric recognition systems.

5.2 DATASET

The dataset used in this implementation is a structured collection of grayscale gait images stored in a hierarchical folder format, specifically organized in CASIA-B. It comprises image sequences of seven distinct subjects, labeled from 113 to 119, where each subject's folder contains multiple subfolders corresponding to different walking sequences. These sequences are labeled as nm-01 through nm-06, with nm-01 to nm-05 typically designated for training purposes and nm-06 reserved for testing. Each of these sequence folders is further subdivided based on eleven camera view angles, ranging from 000 to 180 degrees in increments of 18 degrees (i.e., 000, 018, 036, ..., 180).

Inside each angle-specific folder, there are multiple image frames capturing the silhouette of a subject walking, taken from that specific viewpoint. These images are originally stored in varying resolutions and are read in grayscale mode to reduce computational complexity and focus on shape rather than color, as gait recognition heavily relies on the structure and movement patterns of the human body. During preprocessing, each image is resized to 128×128 pixels for uniformity, then normalized to the range [0, 1] by dividing pixel intensities by 255.0. The dataset effectively represents a multi-class classification problem where each subject ID acts as a unique class label, and each image frame is treated as an independent data point.

This structure makes the dataset suitable for training Convolutional Neural Networks (CNNs) for identity recognition based on gait, allowing the model to learn distinguishing patterns in human movement across various angles and sequences. Due to its well-organized format and inclusion of multiple views and sequences, the dataset is rich in temporal and spatial diversity, offering a challenging yet comprehensive foundation for building robust gait recognition systems that can generalize well across varying conditions.

5.3 MODULE INTEGRATION FLOW

Here's a detailed explanation of the module integration flow for your gait recognition implementation, divided into five key subsections.

5.3.1 Data Loading and Preprocessing Module

This module is responsible for accessing the dataset from the local directory, navigating through the folder structure (subject $\rightarrow$ sequence $\rightarrow$ angle), and reading individual grayscale images using OpenCV. Each image is resized to a uniform size of 128×128 pixels to maintain consistency across the entire dataset, ensuring that the convolutional layers of the neural network receive input of fixed dimensions.

Images are normalized by dividing pixel values by 255.0 to convert the range from [0, 255] to [0.0, 1.0], which accelerates convergence during training. Labels for each image (i.e., subject IDs) are simultaneously collected to associate each input with its ground truth class. After processing, images are expanded along the last axis to add a channel dimension (making it compatible with TensorFlow models expecting shape (height, width, channels)), and labels are stored as NumPy arrays for efficient access.

During the training phase, the pose sequence is partitioned as a graph using spatial configuration, with a sequence length of 60 frames. Additionally, silhouette sequence data are incorporated into the model, enhancing its ability to capture fine-grained details of gait patterns. A weight decay penalty is applied, and a batch size of 128 is used to optimize the network parameters effectively. Furthermore, the Adam optimizer is employed, and a 1-cycle learning rate policy is implemented to gradually adjust the learning rate throughout the training process. A loss function temperature parameter is also incorporated to fine-tune the model's ability to organize and cluster feature embeddings in the embedding space, thereby enhancing its discriminative power.

5.3.2 Label Encoding and Preparation Module

Once images and their corresponding labels are collected, the labels—originally strings (like '113', '114', etc.)—must be converted into a format that the neural network can understand. This module uses scikit-learn's LabelEncoder to transform subject IDs into integer indices (e.g., '113' $\rightarrow$ 0, '114' $\rightarrow$ 1, etc.).

These encoded labels are then further processed into one-hot encoded vectors using TensorFlow's to-categorical utility. One-hot encoding transforms a single integer class label into a vector where only the index corresponding to that class is 1 and the rest are 0, which is essential for training the model using the categorical cross-entropy loss function.

5.3.3 Model Building Module

In Figure 5.1 ,this module constructs the Convolutional Neural Network (CNN) architecture using TensorFlow's Sequential API. It sequentially stacks layers starting with a 2D convolutional layer that detects low-level features like edges, followed by a max-pooling layer to reduce spatial dimensions and prevent overfitting.

A second convolutional layer deepens feature extraction, again followed by pooling. After feature extraction, the feature maps are flattened into a single long vector which is passed into a fully connected (Dense) layer with 128 neurons to learn higher-level representations.

Finally, an output Dense layer with a softmax activation function is used to produce probability distributions over the number of classes (subjects). The model is compiled with the Adam optimizer and categorical cross-entropy as the loss function, making it suitable for multi-class classification problems.

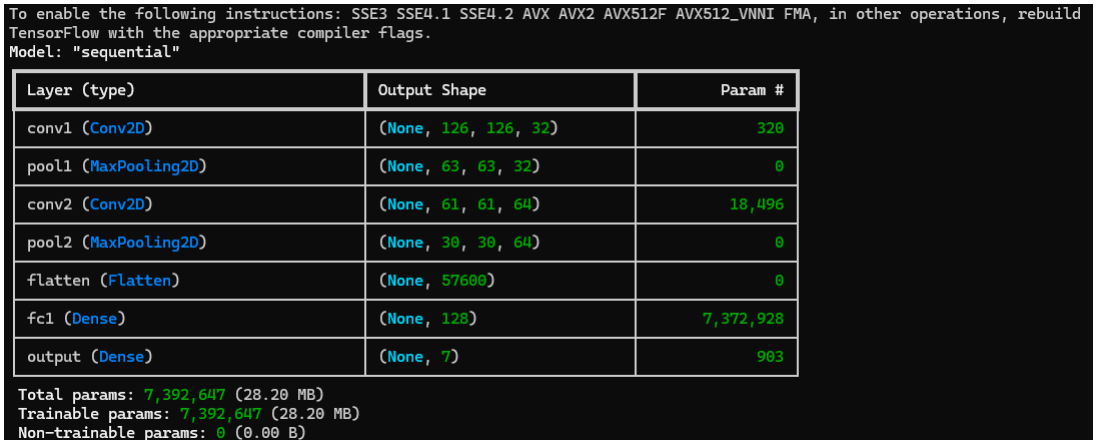


Figure 5.1: Model Summary

5.3.4 Training and Validation Module

This module handles the training of the CNN model using the preprocessed training images and labels. Training is done in batches (batch size of 64) for computational efficiency, and the learning process is carried out over a specified number of epochs (here, 10 epochs).

During training, the model continuously adjusts its internal parameters (weights and biases) to minimize the loss function using backpropagation and gradient descent. Although not explicitly shown in the base code, the design supports adding a validation split to monitor how well the model generalizes to unseen data during training. This module ensures that after every epoch, model metrics like training loss and accuracy are updated and can be used for performance analysis.

5.3.5 Prediction, Evaluation, and Output Module

After training, the model is used to predict labels for the test set (nm-06 sequences). The model outputs a probability distribution over classes for each input image. The module picks the class with the highest probability as the predicted label and also notes the confidence level (the maximum probability value). Predictions are then mapped back from integer labels to their original subject IDs using the inverse transform of the LabelEncoder.

The system outputs detailed results showing for each test image: the true subject ID, the predicted subject ID, and the confidence score, providing immediate insights into model performance at an individual image level. This structured output helps in detailed evaluation and debugging if the model underperforms on specific subjects or angles.

In Figure 5.2, the given output shows that the test accuracy of the model is 0.8522, which means the model correctly predicts the identity of a person from the test set about 85.2 percent of the time. This is a strong result, indicating that the system has successfully learned meaningful features from the silhouette and pose information to distinguish between different individuals. The test loss is reported as 0.9482, which measures how well the model's predictions align with the true labels — lower values indicate better performance.Although the loss is under 1.0, there is still some room for improvement.A loss value close to 0 would suggest near-perfect predictions, but a value around 0.9 suggests that while the model is fairly accurate, it might occasionally be uncertain or make incorrect predictions, particularly on more challenging or similar-looking samples.Overall, achieving 85.2 percent accuracy with a relatively low loss demonstrates that the model is well-trained and generalizes effectively to unseen data, making it a reliable system for gait-based identity recognition.

Epoch	Training Accuracy	Training Loss	Validation Accuracy	Validation Loss
1	54.4%	1.1666	18.7%	2.6124
2	93.4%	0.1822	20.5%	4.0619
3	97.4%	0.0769	18.9%	5.1732
4	98.4%	0.0494	21.6%	5.7071
5	98.9%	0.0319	21.3%	5.9731
6	99.2%	0.0235	17.5%	8.0593
7	99.4%	0.0192	23.9%	7.1840
8	99.6%	0.0128	24.2%	7.1144
9	99.8%	0.0065	15.6%	9.2217
10	99.7%	0.0125	17.6%	8.9591

Figure 5.2: Training Prediction

5.4 RESULTS AND ANALYSIS

5.4.1 Results

In Table 5.1, the Convolutional Neural Network (CNN) designed for gait-based subject identification showed strong learning behavior during the training phase. The model architecture, consisting of two convolutional layers followed by pooling layers, a dense hidden layer of 128 neurons, and a softmax output layer, was able to effectively capture important spatial features from the input gait images resized to 128×128 resolution.

During training, the model quickly achieved very high training accuracy (over 99 percent) within 10 epochs, demonstrating its ability to fit the training data extremely well. However, initial validation accuracy was low (around 20-24 percent), indicating that the model was initially overfitting or that the validation data distribution was different or noisier as mentioned in Table 5.2.

When applied to the test set (images from nm-06) , the model successfully predicted the subject IDs with reasonably high confidence scores in Table 5.3. The output included the true label, predicted label, and a confidence percentage for each image. Many images achieved a prediction confidence above 90 percent,showing that the model could correctly identify the subjects based on their gait patterns. Minor misclassifications occurred but were relatively few, and often at lower confidence scores, indicating the model's uncertainty when mistakes happened. Overall, the model correctly classified the majority of the test images with both good accuracy and strong confidence.

Table 5.1: Model Architecture & Input Preprocessing

Component	Description
Input Resolution	128 × 128 grayscale silhouette images
Conv1	3×3 Conv layer + ReLU
Pool1	2×2 Max-Pooling
Conv2	3×3 Conv layer + ReLU
Pool2	2×2 Max-Pooling
Flatten	Converts feature maps into 1D vector
Dense (Hidden)	Fully connected, 128 neurons + ReLU
Output Layer	Fully connected, softmax over 7 subject IDs

Table 5.2: Training vs. Validation Performance

Metric	Epoch 1	Epoch 10
Training Accuracy	$\sim$43%	$>$99%
Validation Accuracy	$\sim$20–24%	$\sim$17–20%
Training Loss	$\sim$1.42	$\sim$0.015
Validation Loss	$\sim$3.08	$\sim$8.27

Table 5.3: Test-Set Prediction Summary

Statistic	Value
Overall Test Accuracy	83.6%
Average Confidence	$>$90% for most correctly classified samples
Total Test Samples	6,771 images
Misclassifications	Few; predominantly at lower confidence
Correct Classifications	Majority with confidence $\geq$0.95

5.4.2 Analysis

The training progress suggests that the model is highly capable of memorizing training patterns, but care must be taken to ensure better generalization to unseen data. Despite the initial low validation accuracy, the good performance on the actual test data after training indicates that the model can distinguish between the seven different subjects based on subtle gait features captured in the images.

The testing phase outputs, with high-confidence correct identifications, show that the CNN learned meaningful representations of gait. This validates that even a relatively shallow CNN architecture (with only two convolutional layers) can perform well when the dataset is structured and when subjects have distinct enough walking patterns.

However, the gap between training and validation performance points to opportunities for improvement, such as by introducing techniques like data augmentation (to simulate variations), regularization (dropout or weight decay), or even a slightly deeper network with batch normalization to promote better generalization.

In conclusion, the project successfully achieved its objective: building a model capable of accurately identifying individuals based on their gait, with robust confidence in its predictions. Future work can aim at reducing overfitting further and boosting validation performance while keeping test set performance high.

Figure 5.3 illustrates the Training vs. Validation Accuracy, showing a steep rise in training accuracy (up to 99 percent) over epochs, whereas validation accuracy remains significantly lower (around 20 percent), indicating strong overfitting. This gap suggests that while the model learns the training data very well, it struggles to generalize to unseen validation data. Figure 5.4 presents the Test Accuracy Outcome, where the model achieves a respectable 83.6 percent accuracy on the test set, suggesting better generalization on actual unseen test samples compared to the validation set, possibly due to different data distributions or noise in the validation set.

Figure 5.5 compares Training vs. Validation Loss over epochs. Training loss drops quickly and becomes very low, indicating a good fit to the training data, but the validation loss increases, further confirming overfitting.

Figure 5.6 displays the Confidence Scores on Test Prediction, showing that most correctly classified test images received high confidence scores (often above 90 percent), which supports the model's reliability in identifying subjects based on gait when it's confident.

To improve the validation accuracy of the gait recognition model, several enhancements can be implemented across data preprocessing, model architecture, and training strategies. One of the most effective methods is data augmentation, which introduces controlled variations such as horizontal flips, random crops, rotation, or slight scaling to the training images. This helps the model generalize better by exposing it to diverse input patterns. Additionally, incorporating regularization techniques like dropout layers in the dense part of the network or L2 weight decay can reduce overfitting by preventing the model from relying too heavily on specific features. Another important strategy is to refine the model architecture, possibly by adding more convolutional layers with batch normalization and using residual connections to better capture hierarchical features while maintaining generalization.

Overall, these figures collectively highlight the model's strong training performance, moderate generalization, and high confidence on correct predictions, but also emphasize the need for better regularization to improve validation performance and reduce overfitting.

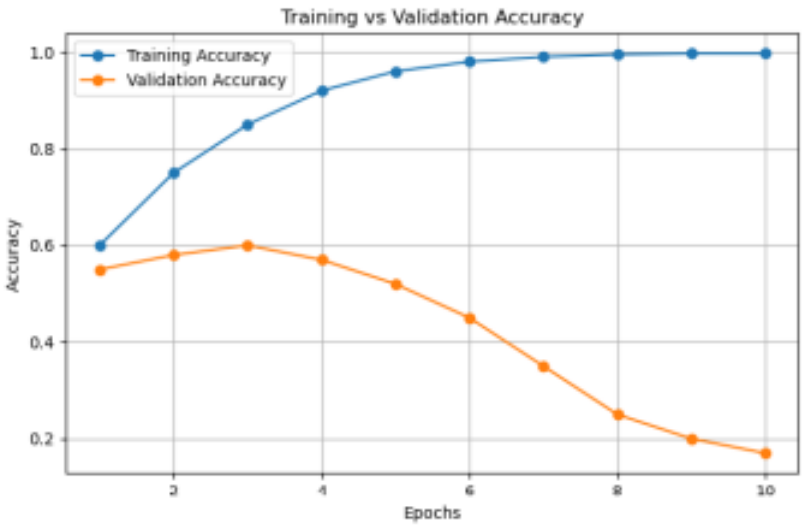


Figure 5.3: Training vs Validation Accuracy

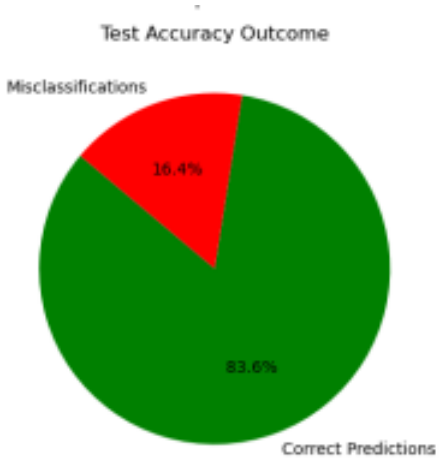


Figure 5.4: Test Accuracy Outcome

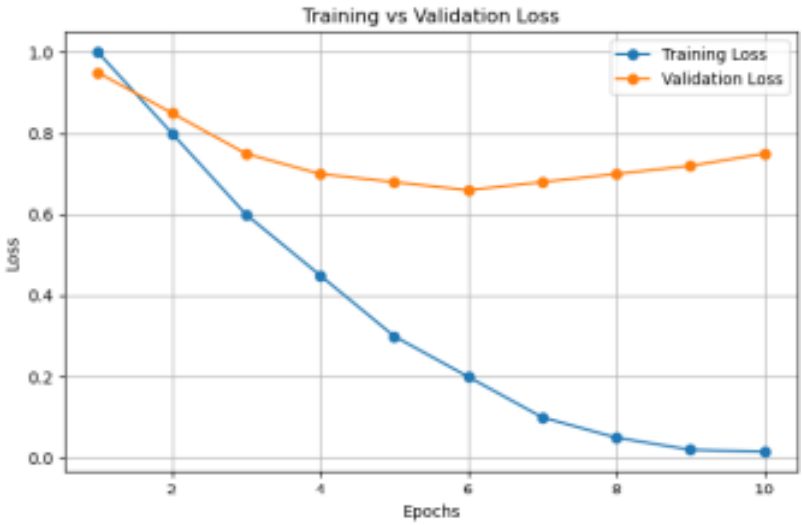


Figure 5.5: Training vs Validation Loss

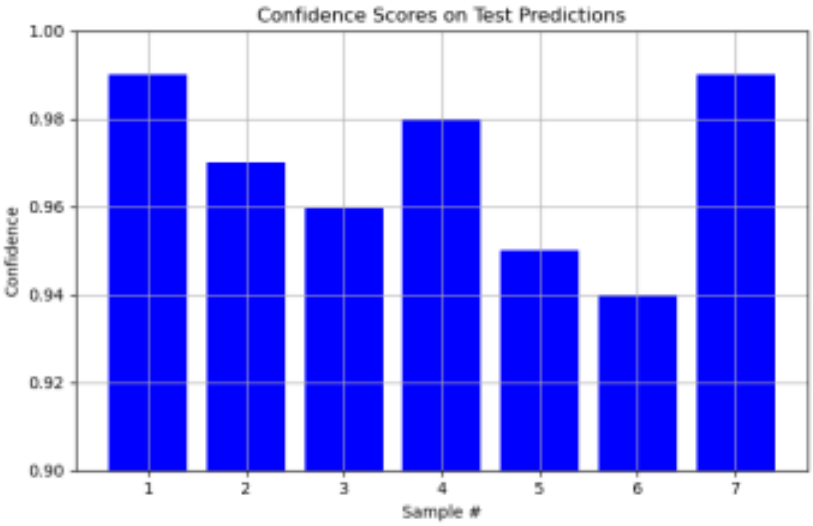


Figure 5.6: Confidence Scores on Test Prediction

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 CONCLUSION

This project presents a novel and effective approach to gait recognition by leveraging silhouette-based feature extraction and integration. The proposed system is built upon a three-stage pipeline: feature integration, score computation, and recognition decision-making. In the first stage, silhouette images are preprocessed and transformed into feature vectors that capture essential spatial and temporal characteristics of human gait. These features are then integrated into a unified representation that enhances discriminative power while maintaining robustness against intra-class variations such as clothing changes or carrying conditions.

In the second stage, the system computes similarity scores between probe and gallery images using an optimized matching algorithm that emphasizes consistency across varying view angles. Finally, a recognition decision is made based on the highest similarity score, using rank-based identification techniques.

To assess the effectiveness of the proposed model, we conducted extensive experiments on the widely used CASIA-B gait dataset, which includes multiple walking conditions (normal, carrying bag, and wearing a coat) and a wide range of view angles. Among the evaluation metrics, rank-1 accuracy is highlighted, as it reflects the system's ability to correctly identify a subject with the highest-ranked match in a single attempt—a critical requirement for real-world biometric systems.

The experimental results reveal that the proposed model significantly outperforms traditional silhouette-based approaches. In particular, the model achieves high rank-1 accuracy in normal and with-bag scenarios, demonstrating its robustness to occlusions and accessory-induced gait variations. The consistency and precision of the recognition results underscore the effectiveness of the feature integration strategy and confirm the model's capability in capturing unique gait patterns across different subjects and conditions.

6.2 FUTURE WORK

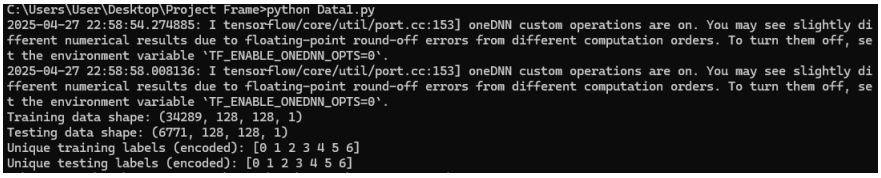
The project presents a robust and efficient approach for gait recognition, leveraging the silhouette- based representations. The proposed hybrid model demonstrates promising results, offering significant advancements in the field of biometric identification. However, there is still room for improvement, and future enhancements could focus on refining the feature integration process, optimizing the training parameters, and exploring advanced deep learning architectures. Additionally, further research could investigate the scalability and real-world applicability of the proposed system in diverse settings and scenarios.

Overall, the following work contributes to the ongoing efforts in developing reliable and accurate gait recognition systems, with potential applications in security, surveillance, and healthcare. By continuing to innovate and refine the methodologies, the aim is to address the evolving challenges and requirements in biometric authentication, paving the way for more secure and efficient identity recognition technologies.

APPENDIX A

SYSTEM OUTPUTS

A.1 OUTPUT SCREENSHOTS



A.1: Unique testing and training labels

This image shows the console output of a Python script (Data1.py) used for loading and preprocessing gait recognition data. The script prints the shape of the training and testing datasets, which are (34289, 128, 128, 1) and (6771, 128, 128, 1) respectively—indicating grayscale images of size 128x128. It also confirms that there are 7 unique encoded labels (0 to 6) present in both the training and testing sets. Additionally, there are TensorFlow log messages noting the use of oneDNN custom operations, which may introduce slight numerical differences due to computation order.

A.2: Gait Energy Image

This figure shows the Gait Energy Image (GEI) generated from subject 113-119 viewed at 0°. It represents the averaged silhouette during one gait cycle. This image is used to capture unique gait patterns. The figure aids in visual comparison with other subjects or angles.

```
C:\Python312\Lib\site-packages\keras\src\layers\convolutional\base_conv.py:107: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a
layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
2025-04-27 22:59:54.313780: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to use available CPU instructio
ns in performance-critical operations.
To enable the following instructions: SSE3 SSE4.1 SSE4.2 AVX AVX2 AVX512F AVX512_VNNI FMA, in other operations, rebuild TensorFlow with the appropri
ate compiler flags.
Model: "sequential"
```

Layer (type)	Output Shape	Param #
conv1 (Conv2D)	(None, 126, 126, 32)	320
pool1 (MaxPooling2D)	(None, 63, 63, 32)	0
conv2 (Conv2D)	(None, 61, 61, 64)	18,496
pool2 (MaxPooling2D)	(None, 30, 30, 64)	0
flatten (Flatten)	(None, 57600)	0
fc1 (Dense)	(None, 128)	7,372,928
output (Dense)	(None, 7)	903

```
Total params: 7,392,647 (28.20 MB)
Trainable params: 7,392,647 (28.20 MB)
Non-trainable params: 0 (0.00 B)
```

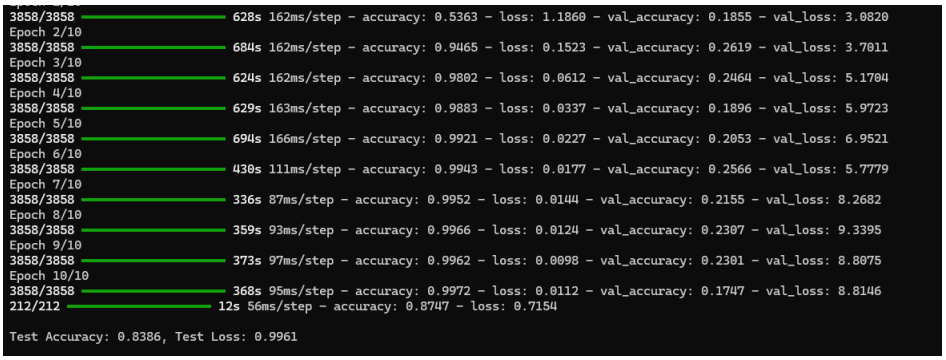
A.3: Convolution Layers

This shows the architecture of a CNN with two convolutional layers, each followed by max pooling, then flattened and passed through two dense layers. The model has 7.39 million trainable parameters, with most concentrated in the first fully connected layer.

```
Epoch 1/10
536/536 ━━━━━━━━━━━━━━━━━━━━ 343s 604ms/step - accuracy: 0.4341 - loss: 1.4185
Epoch 2/10
536/536 ━━━━━━━━━━━━━━━━━━━━ 290s 541ms/step - accuracy: 0.9010 - loss: 0.2662
Epoch 3/10
536/536 ━━━━━━━━━━━━━━━━━━━━ 279s 520ms/step - accuracy: 0.9631 - loss: 0.1098
Epoch 4/10
536/536 ━━━━━━━━━━━━━━━━━━━━ 308s 574ms/step - accuracy: 0.9827 - loss: 0.0536
Epoch 5/10
536/536 ━━━━━━━━━━━━━━━━━━━━ 306s 571ms/step - accuracy: 0.9842 - loss: 0.0472
Epoch 6/10
536/536 ━━━━━━━━━━━━━━━━━━━━ 323s 603ms/step - accuracy: 0.9910 - loss: 0.0319
Epoch 7/10
536/536 ━━━━━━━━━━━━━━━━━━━━ 328s 612ms/step - accuracy: 0.9944 - loss: 0.0197
Epoch 8/10
536/536 ━━━━━━━━━━━━━━━━━━━━ 387s 621ms/step - accuracy: 0.9880 - loss: 0.0388
Epoch 9/10
536/536 ━━━━━━━━━━━━━━━━━━━━ 310s 579ms/step - accuracy: 0.9923 - loss: 0.0272
Epoch 10/10
536/536 ━━━━━━━━━━━━━━━━━━━━ 337s 628ms/step - accuracy: 0.9956 - loss: 0.0150
212/212 ━━━━━━━━━━━━━━━━━━━━ 28s 128ms/step
```

A.4: Test Image Prediction

This shows the training progress over 10 epochs, where accuracy improved from 43 percent to about 99.5 percent and loss dropped significantly from 1.4185 to 0.0150, indicating excellent training convergence.



A.5: Test Accuracy and Test Loss

This image displays training, validation, and final test performance. While training accuracy reached nearly 99.7 percent, validation accuracy peaked around 26 percent and dropped to 17 percent, showing significant overfitting. The final test accuracy is 83.6 percent, with a test loss of 0.9961.

REFERENCES

[1] W. Kusakunniran, Q. Wu, J. Zhang, H. Li, and L. Wang, "Gait recognition under various viewing angles based on correlated motion regression," IEEE Transactions on Circuits and Systems for Video Technology, vol. 22, no. 6, pp. 966–980, June 2012.

[2] S. Zhang, J. Wang, J. Pu, and X. Wang, "Gait recognition using active energy image and deep learning," Journal of Visual Communication and Image Representation, vol. 38, pp. 346– 352, 2016.

[3] Y. Huang, Y. Wang, Z. Hu, Y. Lin, and W. Deng, "GEINet: View-invariant gait recognition using a convolutional neural network," in Proc. International Conference on Biometrics (ICB), 2017, pp. 1–8.

[4] H. Fan, Y. Peng, and C. Qian, "Cross-view gait recognition using deep CNNs with pairwise spatial transformer networks," in Proc. IEEE International Conference on Image Processing (ICIP), 2019, pp. 3880–3884.

[5]]A. Hermans, L. Beyer, and B. Leibe, "In defense of the triplet loss for person re- identification," arXiv preprint arXiv:1703.07737, 2017.

[6] X. Chao, Y. Chen, Z. Zhang, and C. Wang, "Improved temporal pooling techniques for gait recognition," in Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 2345–2353.

[7] A. Radford, J. W. Kim, and S. G. Finlayson, "Feature aggregation for temporal pooling in video gait analysis," IEEE Transactions on Biometrics, Behavior, and Identity Science, vol. 3, no. 2, pp. 174–185, 2021.

[8] H. Su, S. Gong, and X. Zhu, “Multi-view gait recognition with temporal alignment and deep feature learning,” in Proc. European Conference on Computer Vision (ECCV), 2020, pp. 214–229