De Montfort University

MSc Dissertation Report

CLIP-FH: Fine-Tuning CLIP for Hand-Based Identity Matching

Babu Pallam
P2849288
MSc. Artificial Intelligence
Supervisor: Dr Nathanael L. Baisa

May 30, 2025

Abstract

Hand-based biometrics are gaining increasing attention as a contactless, hygienic, and privacy-conscious alternative to traditional modalities such as facial and fingerprint recognition, which often suffer from occlusion, environmental variability, and physical contact requirements. However, robust hand-identity recognition presents unique challenges due to substantial variations in hand pose, illumination, anatomical structure, and background context.

This dissertation investigates whether OpenAI's CLIP—a vision–language model pretrained on large-scale image–text pairs—can be effectively adapted to perform hand-based identity matching without relying on manually engineered features or handcrafted labels. The research addresses this by designing a three-stage evaluation pipeline, each representing a distinct strategy to repurpose CLIP for this underexplored biometric task.

The first stage establishes a zero-shot baseline by evaluating CLIP's original frozen embeddings on hand-image retrieval tasks, without any additional training. The second stage involves supervised fine-tuning of CLIP's image encoder using person re-identification (ReID) methodologies—specifically, the popular CLIP-ReID framework. In the third stage, the study explores joint image–text adaptation through the PromptSG strategy, which introduces learnable pseudo-tokens and semantic prompt templates to align hand-image representations with identity-aware textual embeddings.

All three strategies are rigorously evaluated using a ReID-style validation protocol involving ten Monte-Carlo query–gallery splits on the 11k Hands dataset, thereby simulating realistic retrieval scenarios. While CLIP-ReID and PromptSG have demonstrated strong performance in person re-identification tasks, this study is the first to systematically adapt and benchmark them for hand-based biometrics.

The findings demonstrate that, with appropriate domain-specific adaptation, CLIP can be transformed into a highly effective backbone for hand-identity recognition. Beyond performance improvements, this work contributes a modular, YAML-configurable pipeline and provides practical insights into architectural tuning, prompt engineering, and training strategies—laying the foundation for future research in multimodal, privacy-preserving biometric systems.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my supervisor, Dr. Nathanael L. Baisa, for his invaluable guidance, continuous support, and insightful feedback throughout the course of this dissertation. His expertise and encouragement played a critical role in shaping this research.

I also extend my appreciation to the academic staff and technical support teams at De Montfort University, whose resources and learning environment have greatly contributed to my academic development.

A special thanks to my beloved wife, whose love, patience, and constant encouragement have been a pillar of strength during this journey. I am also deeply grateful to my sister Dhanya, whose motivation and persistent encouragement inspired me to pursue this course and complete it successfully. I am equally grateful to my dear friends Fatima and Joe, for their friendship, moral support, and belief in my work when I needed it the most.

I also thank my family and friends more broadly for their unwavering motivation and understanding. Your belief in me made all the difference.

Lastly, I am grateful to the open-source and research communities whose shared tools, datasets, and publications laid the foundation for this work. This project would not have been possible without the collaborative spirit of the machine learning and computer vision community.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Background and Motivation

Biometric identification has become an important technology in modern security, forensic investigation, and personal authentication. Traditional biometrics like face recognition and fingerprint matching have been widely used because they provide good accuracy and are easy to apply in many situations. However, both face and fingerprint systems have certain limitations which raise concerns today.

Face recognition, though highly accurate, often suffers from problems like changes in facial expression, lighting variations, occlusions (for example, wearing a mask), and privacy worries. Many people feel uncomfortable with facial recognition being used without their permission in public places. Similarly, fingerprint recognition, while reliable, requires physical contact with scanners, which makes it intrusive and less hygienic, especially after the COVID-19 pandemic. There are also cases where fingerprints are not available or usable due to injuries, aging, or manual labour that affects skin quality.

In this, hand-based biometric recognition is emerging as a promising alternative. Hand images, especially dorsal (back of the hand) images or palmer(front of the hand) images, are easy to capture without needing physical touch. Hands carry discriminative features like finger shapes, vein patterns, knuckle marks, and skin textures that can be used for identity matching. Moreover, hand images can be captured at a distance without direct cooperation, making them useful for forensic investigations where only partial evidence like hand images might be available.

Even though hand-based recognition has shown promise, most existing methods have focused on traditional handcrafted features or CNN-based deep learning models. These methods work well but often struggle to generalise across different lighting conditions, hand poses, and backgrounds.

Recently, there has been a major shift in computer vision towards vision-language models(J. Zhang et al. 2024) and multimodal learning(Srivastava and Salakhutdinov 2012). One of the leading models in this trend is **CLIP(Contrastive Language-Image Pre-training)**(Radford et al. 2021). CLIP is trained on a massive amount of image-text pairs from the internet and can understand both images and textual descriptions together.

It has the special ability to perform zero-shot classification — recognizing new categories without direct training — which shows its strong generalization power.

The novelty of using CLIP in the field of hand biometrics lies in its ability to learn not just visual features, but also to align them with semantic meaning. This can be very useful because hands have many subtle features that need both global and fine-grained understanding. Applying and fine-tuning a large pre-trained vision-language model like CLIP for hand-based identity matching is a new direction that has not been explored deeply yet.

At the same time, research trends show that models that combine visual and textual modalities, and those using contrastive learning strategies, outperform older CNN-based models in various recognition tasks, such as in person re-identification tasks(S. Li, Sun, and Q. Li 2023). This indicates that applying vision-language models to hand-based biometrics can also open new possibilities for building more generalized, robust, and scalable identity matching systems.

This project aims to explore this novel idea by fine-tuning CLIP for hand recognition and comparing its performance using different strategies and architectures, filling the gap in the current research on hand-based biometrics.

1.2 Problem Statement

Although biometric recognition has seen big improvements over the years, hand-based identity matching still faces many challenges. While modalities like fingerprint and face recognition have been widely adopted in real-time systems such as mobile authentication and surveillance, robust hand-based recognition systems are still limited in practical deployment due to variability in hand poses, lighting, and occlusions.

Most of the current hand recognition systems rely on CNN-based models like MBA-Net(Nathanael L Baisa et al. 2022b) which have shown good performance on hand datasets. However, CNNs generally focus more on local features and often struggle to generalize when the data comes from different environments.

At the same time, large pre-trained vision models like CLIP have shown very strong generalization ability across multiple domains. CLIP is trained on a wide range of internet images and text pairs, giving it a rich understanding of visual and semantic concepts. However, CLIP was not specifically pre-trained on hand images, and hand datasets are very different from the natural images or objects that CLIP usually sees. Because of this, directly applying the pre-trained CLIP model to hand-based identity matching may not give the best results. The model may not naturally focus on the fine details needed for distinguishing different hands.

Another challenge is that extracting robust and discriminative features for hand identity matching is much harder than it looks. Hands often lack strong texture compared

to faces, and the differences between individuals can be very subtle. In uncontrolled environments, hands can be rotated, partially hidden, or affected by lighting conditions. A good model must capture both the global structure (like hand shape) and local features (like finger arrangement, wrinkles, and skin patterns) while being insensitive to irrelevant changes.

CLIP offers a potential solution to these challenges. Because it is trained using contrastive learning, it naturally learns to pull similar items closer and push different items apart. This is exactly what is needed for identity matching. Also, CLIP learns multimodal embeddings, meaning is that it can align visual information with language-based semantics. Even if detailed hand descriptions are not available, fine-tuning CLIP on hand datasets using identity labels can guide it to learn more discriminative visual features suitable for hand recognition.

Therefore, the problem addressed in this project is how to adapt and fine-tune the powerful but general CLIP model specifically for the task of hand-based identity matching, and how to overcome the limitations of traditional CNN-based hand recognition models.

1.3 Project Aim and Objectives

The main aim of this project is to fine-tune the CLIP model for hand-based identity matching, and to study whether vision-language pre-training can improve performance compared to traditional CNN-based methods like MBA-Net. By adapting CLIP's visual encoder to hand datasets, this project seeks to learn discriminative hand features that are robust to pose changes, lighting variations, and other real-world conditions.

The project also aims to systematically evaluate different fine-tuning strategies and backbones (ViT-B/16, RN50, or others) to find the most effective approach for hand-based person recognition.

To achieve this aim, the project defines the following key objectives:

- **Baseline Testing:** Evaluate the performance of the original CLIP models without fine-tuning, on hand datasets to establish a baseline for comparison.
- **CLIP-ReID Adaptation:** Integrate CLIP-ReID(S. Li, Sun, and Q. Li 2023), which proposed a two-stage strategy for adapting CLIP to image re-identification by first learning identity-specific text tokens and then fine-tuning the image encoder using standard ReID losses.
- **PromptSG Integration:** Explore the use of prompt learning strategies, including learnable pseudo-tokens and prompt ensembles, inspired by PromptSG(Yang et al. 2024). This proposes language-guided training to help the model focus on

semantically relevant features for person re-identification without requiring explicit labels.

- **Comparative Analysis:** Compare the performances of different backbones (ViT-B/16 vs RN50) and training strategies under a ReID-style evaluation over dataset.

By following these objectives, the project will create a detailed understanding of how vision-language models like CLIP can be fine-tuned and adapted for hand-based biometric applications.

1.4 Research Questions

The formulation of research questions plays a pivotal role in shaping the direction and relevance of any scientific investigation. In the domain of contact-less biometric recognition, particularly hand-based modalities, several open challenges persist around generalisation, usability, robustness, and alignment with real-world constraints such as hygiene and privacy. Addressing these gaps is not only critical for advancing academic understanding but also essential for building deployable, ethical, and secure biometric systems. The following research questions have been carefully crafted to investigate these dimensions through the lens of vision-language models (VLMs)(Bolcato 2023) and contrastive learning(Le-Khac, Healy, and Smeaton 2020). Based on the aim and objectives of this project, the following research questions have been formulated to guide the work:

- **RQ1:** How well does the original CLIP model perform on hand-based biometric identification without any fine-tuning?
- **RQ2:** Does fine-tuning the CLIP image encoder improve the performance on hand datasets though the text encoder is frozen?
- **RQ3:** Can the integration of CLIP-ReID based Prompt Learning strategy enhance the discriminative ability of CLIP for hand-based identity matching?
- **RQ4:** Can the integration of PromptSG based Semantic Guidance using prompt enhance the discriminative ability of CLIP for hand-based identity matching?
- **RQ5:** How do different backbone architectures (ViT-B/16 vs RN50) affect the quality of the learned hand representations?
- **RQ6:** What are the strengths and limitations of vision-language models like CLIP compared to traditional CNN-based approaches like MBA-Net when applied to hand biometrics?

These research questions will help in systematically evaluating the performance of CLIP for hand recognition, understanding the effects of fine-tuning strategies, and identifying areas for further improvement.

1.5 Stakeholder Requirements and Objective Mapping

The deployment of hand-based biometric recognition systems intersects with the needs of various stakeholders, each with distinct requirements and expectations. To ensure real-world relevance and alignment, the technical objectives of this project were mapped against these stakeholder demands as follows:

Table 1.1: Stakeholder Requirements Mapped to Technical Objectives

Stakeholder	**Key Requirements**	**Mapped Technical Objectives**
Security agencies	High-accuracy identity verification, robustness under varied lighting and angles, and real-time processing	Implemented a staged fine-tuning approach using CLIP-ViT and CLIP-RN50 to maximise cross-view recognition accuracy; validated using ReID metrics across multiple query-gallery splits
Healthcare professionals	Non-contact, hygienic data collection; privacy compliance	Focused on contact-less hand biometrics; dataset preprocessing ensured de-identification; designed architecture for privacy-aware retrieval without storing raw images
AI researchers	Reproducible benchmarks, modular architectures, and extensibility for future multimodal tasks	Released modular pipeline using YAML configuration; included PromptSG and CLIP-ReID baselines for reproducibility and comparability
End users (public)	Non-intrusive usage, fairness, and reliability across demographics	Incorporated dorsal and palmar views; ensured balance in dataset splits; emphasized low-bias modality through hand over face biometrics

This mapping ensured that each technical choice was guided by practical needs and societal context, improving both research validity and potential for adoption.

1.6 Threat Model and Ethical Considerations

The use of biometric systems necessitates a careful consideration of ethical risks and security vulnerabilities. We adopt a basic threat model assuming that adversaries may attempt to compromise the system by spoofing identity using fake hand images, generating synthetic samples to bypass authentication, replaying previously captured data, or exploiting leaked datasets to infer private identity traits.

During the training phase, raw hand images are used to fine-tune the model and extract meaningful visual features. However, for real-world deployments, our system does not store or transmit raw images to minimize privacy and security risks. Instead, only the embedded feature vectors produced by the trained model are stored and used for identity matching, ensuring that no reconstructable biometric data is retained. This approach aligns with best practices in biometric security and is adaptable depending on application-specific requirements.

Furthermore, the system does not expose identity labels publicly and is designed to operate locally or on privacy-compliant servers. Ethical concerns, such as potential bias across hand sizes, skin tones, and accessory presence, were addressed via preprocessing filters and stratified sampling during evaluation.

This project adheres to the principles of fairness, accountability, and transparency, aligning with contemporary guidelines on biometric AI system development.

1.7 Structure of the Report

This report is organised into seven chapters to systematically present the background, methodology, implementation, experiments, and findings of the project. The structure follows a logical progression, starting from the motivation and ending with conclusions and recommendations for future work.

Chapter 1: Introduction outlines the project background, defines the problem of hand-based identity matching, and introduces the motivation behind exploring CLIP for biometric applications. It also presents the research objectives, methodology overview, and a roadmap of the report.

Chapter 2: Literature Review surveys existing work on hand biometrics, including CNN-based methods, contrastive learning, and recent advances in vision-language models like CLIP. It identifies research gaps in current approaches and supports the need for adapting multimodal models to hand identity recognition.

Chapter 3: Methodology details the experimental design, including dataset preparation, training stages, loss functions (e.g., Cross-Entropy, Triplet, SupCon), and architectural strategies such as prompt learning and feature projection. It explains the rationale behind model choices (ViT-B/16, RN50) and evaluation metrics like Rank-1 and mAP.

Chapter 4: Implementation describes the practical aspects of system development. This includes the modular codebase structure, configuration management using YAML, P×K sampling logic, model loading strategies, and techniques for embedding extraction and evaluation. Technical challenges and mitigation strategies are also discussed.

Chapter 5: Results presents the outcomes of experiments conducted across different CLIP-FH training stages (baseline, image encoder fine-tuning, prompt-based joint learning). It includes quantitative results on multiple query-gallery splits, comparing model variants and training strategies using comprehensive metrics and plots.

Chapter 6: Discussion interprets the experimental results in relation to the research goals and existing literature. It discusses key observations, strengths and limitations of each method, and the effectiveness of techniques like prompt tuning and SupCon loss in improving recognition performance.

Chapter 7: Conclusion and Future Work summarises the main contributions of the project, reflects on ethical considerations and deployment implications, and proposes directions for extending this work in real-world biometric systems and future multimodal learning research.

2 Literature Review

Biometric recognition has long been a central topic in computer vision, with a wide range of modalities including fingerprints, faces, irises, and hand geometry being explored for identity verification. Among these, hand-based biometrics have gained attention for their non-intrusiveness, richness in discriminative features, and suitability in unconstrained environments. This chapter reviews prior work across several key areas relevant to this project: traditional CNN-based approaches for hand recognition, contrastive learning frameworks for feature representation, and the recent emergence of vision-language models (VLMs) such as CLIP. It highlights the strengths and limitations of existing techniques, identifies gaps in generalisation and semantic understanding, and motivates the need for adapting multimodal models to the domain of hand identity matching.

2.1 Hand-Based Biometrics

Hand-based biometrics is becoming a strong alternative to face and fingerprint recognition, especially in situations where only hand images are available, like in criminal investigations or surveillance footage. Compared to face, hand images are less affected by expressions and occlusions, and they also avoid the privacy concerns that often come with facial recognition. Hands are relatively stable over time, making them suitable for long-term identification tasks.

Traditional hand biometric systems used features like hand geometry, palmprints, finger lengths, and vein patterns. These features were usually extracted using handcrafted methods or simple image processing techniques(Dantcheva, Elia, and Ross 2016). However, such approaches are not reliable in real-world conditions where lighting, hand pose, and occlusion can change a lot. For example, variations in background, shadows, or camera angle can easily confuse traditional systems.

With the arrival of deep learning, researchers started using Convolutional Neural Networks (CNNs) for hand recognition. CNNs can learn features automatically from hand images without manually defining them. One such work is by Afifi(Afifi 2019b), who introduced the 11k Hands dataset, which includes dorsal and palmar hand images. His method used CNNs for both gender recognition and biometric identification, and

the results showed that deep learning outperformed traditional handcrafted features. Interestingly, the dorsal side of the hand was found to be as useful, or even better, than the palmar side for identification.

Another important work is by Baisa, who proposed a model called GPA-Net(Nathanael L. Baisa et al. 2022c). This model combines global and local hand features for better identification, especially when hand images are taken in uncontrolled environments. The idea is that some parts of the hand (like fingers or knuckles) can give more reliable identity clues than just looking at the hand as a whole.

Yuan also worked on this problem and proposed HandNet(Yuan et al. 2020), a deep learning model that combines traditional and CNN-based methods for hand image identification. Their approach again confirmed that deep learning methods are more powerful than older handcrafted ones.

These studies clearly show that hand-based biometrics is a reliable and growing field. The shift from traditional methods to deep learning has already improved performance a lot. However, most of the work so far has focused on Convolutional Neural Networks (CNNs)(O'shea and Nash 2015) . There is still a need to explore more powerful models like Transformers and vision-language models such as CLIP for better performance and generalisation in hand-based identity matching.

2.2 CNN-Based Methods for Hand Recognition

CNNs have been widely used for image recognition tasks, including hand-based biometrics. CNNs are good at learning local patterns in images like edges, textures, and shapes. This made them a natural choice for recognising hand features like finger length, knuckle structure, and palm patterns. In hand recognition, CNNs helped improve accuracy compared to traditional handcrafted methods.

One of the well-known models in this area is MBA-Net(Nathanael L Baisa et al. 2022b). This model was specially designed for hand-based person recognition in criminal investigations, where only hand images might be available. MBA-Net uses multiple branches in its architecture, combining a global feature branch and attention-based branches. The attention modules help the model focus on the important parts of the hand while ignoring irrelevant background. It also uses relative positional encoding to make the spatial attention more stable, especially when pixel locations change. This design helps the model learn better features from hand images.

The MBA-Net was tested on two public hand datasets and achieved state-of-the-art performance. It performed better than traditional hand recognition models and even adapted person ReID models that were retrained on hand data. These results show that CNNs are quite effective when designed carefully for the hand recognition task.

However, CNN-based methods also have some limitations. One major issue is gener-

alisation. CNNs usually perform well on the training dataset but may not work as well on new, unseen data, especially if the lighting, hand pose, or background is different. This is because CNNs tend to learn patterns specific to the training images, and they may struggle when the testing environment is not similar (Azulay and Weiss 2018).

Another problem is that CNNs are not good at capturing long-range dependencies in the image. They focus more on local features and might miss the global structure of the hand. For example, understanding the spatial relationship between fingers or between the palm and fingers can be difficult for standard CNNs. This limitation becomes more serious in hand-based identification, where fine-grained and global features are both important (Rangel et al. 2024).

Also, many CNN models, like ResNet(He et al. 2016) and Very Deep Convolutional Networks(Simonyan and Zisserman 2014), need a lot of labelled data to train well. In hand biometrics, datasets are usually smaller and less diverse than face datasets, which makes training deep CNNs even more challenging.

To overcome these problems, recent research is moving towards transformer-based models and vision-language models like CLIP. These models can learn both local and global features, and they can use text information to guide the learning. This shift may help improve the generalisation and robustness of hand recognition systems beyond what CNNs can achieve.

2.3 Contrastive Learning and Vision-Language Models

Contrastive learning has emerged as a powerful method in deep learning, particularly for learning visual representations without the need for extensive labelled data. The core idea is simple yet effective: similar samples should be embedded close together in the feature space, while dissimilar ones should be far apart. For example, two images of the same person's hand should have similar embeddings, whereas images of different individuals' hands should be well-separated.

Several loss functions have been designed to implement this idea. One of the most widely used is the InfoNCE loss (A. v. d. Oord, Y. Li, and Vinyals 2018), which encourages the model to bring a positive pair (such as an image and its correct caption) closer together while pushing it apart from negative pairs (incorrect matches). Another widely adopted approach is the Supervised Contrastive Loss (SupCon) (Khosla et al. 2020), which extends this concept to allow multiple positives per anchor when class labels are available. SupCon has shown to be more robust in scenarios with high intra-class variability, such as differences in hand pose or lighting.

One of the most successful applications of contrastive learning is CLIP. CLIP learns

a joint embedding space by aligning images and their corresponding textual descriptions. It is trained on a massive dataset of internet-sourced image-text pairs, enabling it to generalise well to unseen classes and tasks.

CLIP's most notable capability is zero-shot classification. Instead of relying on predefined labels, CLIP matches an image to the most relevant text prompt — such as "a photo of a cat" — even if it has never seen that class during training. This makes CLIP highly flexible and powerful for tasks like retrieval, classification, and even segmentation across a variety of domains.

For hand-based biometrics, CLIP offers a promising direction. Although typical hand datasets do not come with rich text annotations, CLIP can still be fine-tuned using contrastive losses and identity labels. It also enables the potential use of descriptive prompts like "a hand with long fingers" or "a wrinkled dorsal hand," if available, which aligns with CLIP's original cross-modal learning objective. This ability to bridge vision and language makes CLIP a strong candidate for improving generalisation in hand identity recognition.

2.3.1 How CLIP Works?

CLIP is a vision-language model developed by OpenAI(OpenAI 2025) that learns to connect images and text in a shared embedding space. Unlike traditional image classification models that learn from fixed class labels, CLIP is trained using image–text pairs, making it highly flexible and generalisable to different tasks.

Basic Pipeline: CLIP consists of two main components:

- A **visual encoder** (e.g., ViT-B/16 or RN50) that converts an input image into a feature vector.
- A **text encoder** (based on a Transformer) that converts a text description (e.g., "a photo of a hand with short fingers") into a corresponding feature vector.

These two encoders are trained together using a contrastive loss(*Contrastive Loss - an overview* n.d.). For each image–text pair in the training data, CLIP is taught to bring the image and its correct caption closer in the embedding space, while pushing apart mismatched pairs.

Example Workflow: Let's say we input an image of a person's dorsal hand into CLIP. If we also provide text prompts like:

- "a photo of a dorsal hand"
- "a hand with visible veins"

- "a hand with a ring on the index finger"

CLIP will encode each sentence and compare their similarity to the image embedding using cosine similarity. The text with the highest similarity score is considered the best match — effectively allowing CLIP to "understand" the image by linking it to a meaningful description.

It works even without text labels. When No Text Is Provided, in many real-world biometric tasks, such as hand identity matching, no descriptive text is available. In such cases, CLIP can still be used effectively:

- During training, image encoders can be fine-tuned using contrastive losses (e.g., SupCon or InfoNCE) based only on image–label pairs.
- During inference or retrieval, image embeddings are compared directly using cosine similarity or FAISS indexing, without relying on text at all.

Even though the text encoder remains unused during inference in such cases, CLIP's image encoder still benefits from the knowledge learned during its joint training with text. This cross-modal pretraining helps it learn more generalisable and robust visual features than traditional CNNs trained purely on labels.

To summarise, CLIP works by learning a joint embedding space for images and text using contrastive learning. When text is available, it can be used for zero-shot classification, captioning, or semantic guidance. When text is not available — as in our hand biometric setting — CLIP's image encoder can still be fine-tuned to serve as a powerful backbone for identity matching using supervised learning with image–label pairs or retrieval-based contrastive losses.

2.3.2 Two Architectures Used in CLIP

This research mainly focused on the following two architectures.

ViT-B/16: Vision Transformer Backbone ViT-B/16(Dosovitskiy et al. 2020), proposed by Dosovitskiy, is a Vision Transformer that treats an image as a sequence of 16×16 pixel patches, similar to how words are processed in Natural Language Processing(NLP) transformers. These patches are embedded and passed through a series of self-attention layers, allowing the model to capture global image context and long-range dependencies.

This architecture must be suited to hand biometrics because hands have complex spatial structures — such as finger alignment, shape, and orientation — which benefit from a global view. ViT-B/16's attention mechanism allows it to understand the overall geometry of a hand rather than focusing solely on local textures. As a CLIP image encoder, ViT-B/16 benefits from large-scale pre-training and is especially effective when fine-tuned for identity-specific tasks using contrastive learning.

In our project, ViT-B/16 was evaluated at each stage of adaptation. It consistently showed strong performance, particularly in conditions requiring global feature understanding across varied poses and lighting.

RN50: ResNet-50 Backbone ResNet-50 (RN50)(He et al. 2016), is a deep convolutional neural network that uses residual connections to improve training stability. In CLIP, RN50 serves as an alternative image encoder that specialises in capturing local patterns such as textures, edges, and fine-grained features.

This makes RN50 a strong candidate for tasks involving detailed biometric traits — such as skin texture, wrinkles, and minor anatomical differences between hands. While it may not model global structure as effectively as a transformer, RN50 excels when local image cues carry the most discriminative information.

In this study, RN50 was included alongside ViT-B/16 to compare how global versus local feature representations influence performance in hand-based identity matching. The results showed that while RN50 performs competitively, especially in fine-grained detail recognition, ViT-B/16 generally outperforms it in scenarios requiring holistic understanding.

2.4 CLIP for Biometric Applications

Since the success of CLIP in vision-language tasks, many researchers have started applying it to biometric applications like face recognition and person re-identification. One such work is Face recognition(Bhat and Jain 2023) by Aaditya Bhat, which showed that CLIP's pre-trained embeddings can be used for recognising faces. This is used for identity labels in a contrastive way and fine-tuned CLIP for face-based identity matching. The results showed that CLIP features are powerful and can be adapted to biometric tasks with good accuracy.

Another important work is CLIP-ReID(S. Li, Sun, and Q. Li 2023), which applied CLIP to the problem of person re-identification (ReID). CLIP-ReID used a two-stage training method. In the first stage, it learned special prompts (text tokens) for each identity while keeping the image and text encoders frozen. In the second stage, it fine-tuned the image encoder using triplet loss, ID loss, and a special image-to-text cross-entropy loss. This helped the model align the image features with learned identity descriptions. The success of CLIP-ReID showed that even without exact text labels, CLIP can be adapted for identity matching tasks.

More advanced methods like PromptSG(Yang et al. 2024) and CLIP-SCGI(Han et al. 2024) further improved these ideas. PromptSG used learnable prompts to guide CLIP's attention towards semantically meaningful regions of the image, while CLIP-SCGI generated text captions for each image using image captioning models and then used those

captions for learning better identity features. These works prove that CLIP's vision-language power can be useful for biometrics, especially when combined with techniques like prompt learning, cross-modal fusion, and attention mechanisms.

Even though CLIP has shown great results for faces and full-body ReID, not much work has been done on using CLIP for hand-based identity recognition. Hands are different from faces or bodies in structure, texture, and visual patterns. The challenges like finger pose variation, lighting conditions, and occlusion make it harder for CLIP to directly apply its pre-trained features without adaptation.

This creates a gap in research, as CLIP is not originally trained on hand images, and hand-based biometric datasets are much smaller compared to face datasets. The pre-trained CLIP model might not focus on important hand features by default. Also, without text captions describing hands, the language guidance part of CLIP may not be useful directly.

Because of this, there is a strong motivation to fine-tune CLIP for hand-based tasks. Fine-tuning can help the model adapt its attention and feature space to the specific patterns in hand images. Using identity labels with contrastive losses, and optionally generating text prompts or using prompt-learning strategies, can make CLIP more effective for hand biometrics. By doing so, we can take advantage of its large-scale pre-training while also adapting it to this niche but important domain.

2.5 CLIP-ReID and PromptSG Approaches

In person re-identification (ReID), the goal is to match images of the same person across different cameras or viewpoints. Many recent models have tried to improve ReID by using vision-language models like CLIP, which can learn both from images and text. One of the important works in this direction is CLIP-ReID. This method introduced a two-stage training strategy that works even without concrete text labels.

In the first stage, CLIP-ReID creates learnable text tokens (small pieces of text) for each identity. These tokens are passed into CLIP's text encoder to generate unique descriptions for each person. At this stage, the CLIP image and text encoders are frozen, and only the text tokens are trained using contrastive loss. This helps adapt the text space to the identities in the dataset.

In the second stage, these learned text features are kept fixed, and the image encoder is fine-tuned using triplet loss, ID loss with label smoothing, and an image-to-text cross-entropy loss. This combination of losses forces the image features to stay close to their matching text features. The result is a model that learns strong and discriminative visual embeddings for ReID. CLIP-ReID also showed that just fine-tuning CLIP's image encoder gives competitive results, even without any text prompts.

After CLIP-ReID, new models such as PromptSG(Yang et al. 2024) took this idea

further by using prompt tuning for person ReID. PromptSG learns special pseudo-tokens that describe visual characteristics of each person. These tokens are used to form text prompts like "a person with [tokens]", which guide CLIP to attend to more meaningful visual regions. The prompts are learned end-to-end together with the visual features. Without needing extra labels, PromptSG achieved state-of-the-art performance on person ReID datasets compared to CLIP-ReID. It showed that language guidance can help CLIP learn better identity features.

Another improvement used in PromptSG is prompt ensemble, where multiple prompt templates are used and their results are averaged. This helps reduce overfitting and improves generalisation. For example, different templates like "a person wearing [tokens]" and "a person with [tokens]" can capture different aspects of identity.

Along with prompt learning, many ReID models, including CLIP-ReID and PromptSG, also use additional architectural blocks like BNNeck and ArcFace. BNNeck is a simple yet powerful trick that separates the features used for ID classification and those used for retrieval. It applies Batch Normalisation (BN) before the classifier, which helps improve both classification accuracy and retrieval performance.

ArcFace is a special classification head that adds a margin between classes in the feature space, which improves the discriminative power of the model. It is often used with ID loss to create well-separated identity embeddings. These tools are also used in the fine-tuning stages of many CLIP-based ReID systems to improve training stability and final performance.

Together, CLIP-ReID, PromptSG, and these advanced techniques show that CLIP can be successfully adapted to identity matching tasks by using creative strategies like prompt tuning, feature separation, and margin-based classification. These ideas inspire the development of CLIP-FH for hand-based identification, where similar methods can be applied to improve performance on hand images, even without explicit hand descriptions.

Comparison with Other Multimodal Biometric Systems

Several multimodal biometric systems have been proposed in recent literature, combining modalities such as iris and periocular features(Umer et al. 2020) or face and deep audio embeddings(Wang et al. 2022) to improve robustness in identity recognition. These systems often enhance recognition accuracy by leveraging complementary cues—iris for texture and periocular for spatial context, or face for structure and voice for temporal patterns. However, they also introduce practical limitations, including privacy sensitivity (especially with facial data), reliance on cooperative user behaviour, and the need for specialised capture devices.

In contrast, this research focuses on a single-modality biometric—hand images—processed using VLMs like CLIP. The hand-based modality presents several advantages: it is less in-

trusive than facial recognition, more hygienic than contact-based biometrics, and requires no active user cooperation. Moreover, by combining hand imagery with prompt-guided learning and contrastive fine-tuning, this work achieves a trade-off between generalisability and practicality, while avoiding the complex sensor fusion and ethical concerns present in other multimodal approaches.

2.6 Summary and Identified Gaps

The reviewed literature demonstrates substantial progress in both classical and deep learning-based hand biometrics. However, classical methods relying on hand-crafted features exhibit limited generalisability across capture conditions and fail to scale with modern data diversity. **Therefore, this study builds on large-scale VLMs, specifically CLIP, to leverage their strong representation learning capabilities in cross-domain settings.**

While CNN-based architectures such as MBA-Net (Nathanael L Baisa et al. 2022b) have shown competitive performance on curated hand datasets, their limited transferability and reliance on supervised training restrict their deployment. **To address this, we adopt the CLIP-ReID framework, integrating a contrastive learning pipeline that aligns image features through identity-level supervision and prompt-enhanced semantic guidance.**

Multimodal biometric systems—such as face+voice or iris+gait—offer complementary features but face limitations in real-world adoption due to hardware requirements, user cooperation, and privacy sensitivities. **This motivates the selection of hand biometrics, a privacy-conscious, contact-less modality, enhanced in our study through vision-language modelling and prompt tuning without needing audio or facial input.**

CLIP's original zero-shot prompt mechanism, while effective in coarse-grained classification, lacks task alignment for biometric re-identification. **We therefore adapt and integrate PromptSG, allowing learnable prompts to inject domain-specific context (e.g., dorsal/palmar hand types) directly into the CLIP text encoder. This improves discriminability while preserving CLIP's generalisation strengths.**

Finally, baseline CLIP models—without re-identification or prompt adaptation—underperform on fine-grained tasks like individual hand identification. **Our work explicitly avoids this limitation by incorporating PromptSG-based tuning and CLIP-ReID image encoder adaptation, enabling robust, privacy-aware person identification using only hand imagery.**

3 Methodology

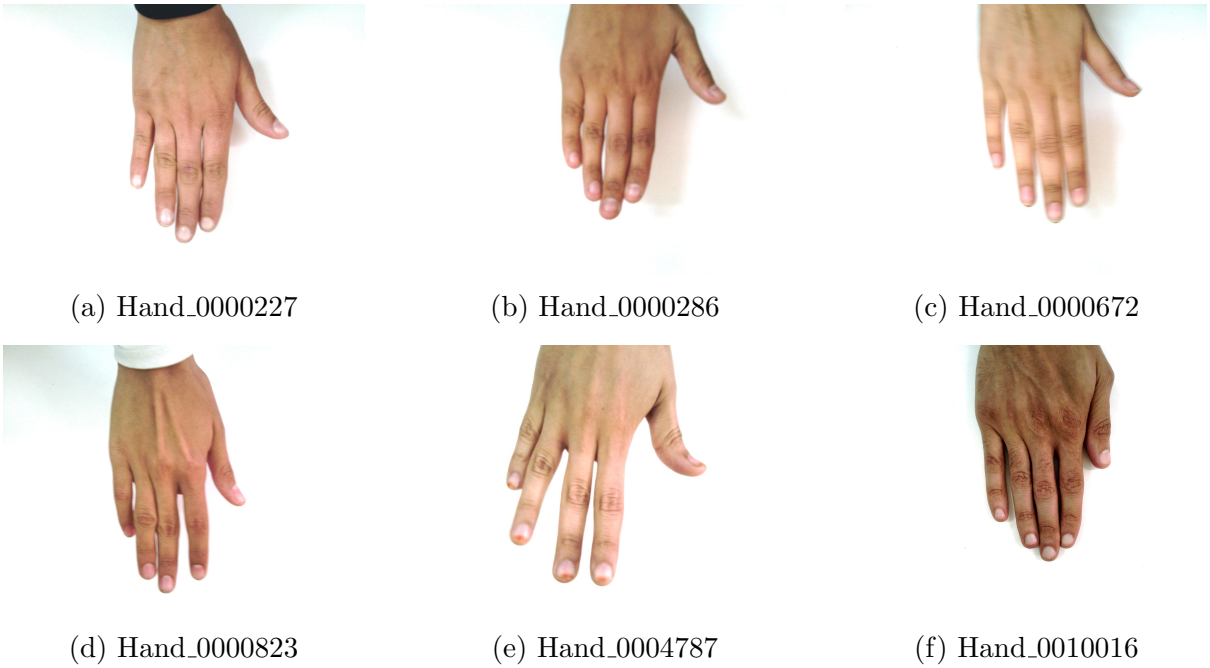
This chapter explains the technical process followed in this project to adapt and fine-tune the CLIP model for hand-based identity matching. It describes the datasets used, the model architectures and modifications, the training strategies applied at different stages, the loss functions chosen for optimisation, and the evaluation metrics used to measure performance.

The aim of this chapter is to provide a clear and detailed description of the research design so that the experiments can be reproduced. Every decision made in the training and evaluation pipeline is linked to the project's objectives and research questions. For example, different fine-tuning strategies are explored to answer whether CLIP's performance on hand biometrics can be improved through contrastive learning and prompt tuning.

Special care has been taken to ensure that the methodology maintains technical correctness, addresses practical challenges like dataset variations, and critically reflects on the possible limitations of the approach. This systematic method is important to fairly evaluate the strengths and weaknesses of applying vision-language models like CLIP to the domain of hand-based biometric recognition.

3.1 Datasets

This project uses the publicly available **11k Hands Dataset**(Afifi 2019a) as the sole source for training and evaluation. It includes over 11,000 RGB images of hands collected from 190 subjects with diverse age, gender, and ethnicity profiles. The dataset offers images under varying lighting, pose, and background conditions, making it a strong candidate for biometric identification tasks. Each hand is captured in four different views: dorsal right, dorsal left, palmar right, and palmar left.



(a) Hand_0000227 (b) Hand_0000286 (c) Hand_0000672

(d) Hand_0000823 (e) Hand_0004787 (f) Hand_0010016

Figure 3.1: Six representative dorsal right hand images from the dataset, illustrating variation in skin tone, shape, pose, and lighting.

Dataset Subset and Preprocessing Strategy

This study focuses exclusively on **dorsal right hand images**, as these have been shown in prior research to contain highly discriminative features and are less affected by privacy or religious concerns. All hand images containing visible accessories (e.g., rings, watches) were excluded based on metadata filtering during preprocessing.

The dataset was organised into the following splits:

- **Training/Validation:** The first half of identities were used for training and validation. From each identity, one image was randomly selected for validation, and the remaining images were assigned to training.
- **Test:** The second half of the identities were reserved exclusively for testing. These identities were not seen during training to evaluate generalisation.
- **Query-Gallery Splits:** Ten Monte Carlo query-gallery splits (`query0`–`query9` and `gallery0`–`gallery9`) were generated by randomly selecting one image per identity for the gallery and placing the rest in the corresponding query folder. This follows the evaluation protocol introduced in MBA-Net for person re-identification tasks.

Preprocessing and Data Augmentation

The following preprocessing steps were applied to all images prior to model input:

- **Resizing:** All images were resized to 224×224 pixels to match the input requirements of ViT-B/16 and RN50.
- **Normalization:** Images were normalised using ImageNet's mean and standard deviation, consistent with CLIP's pretraining.
- **Filtering:** Only images with "dorsal right" orientation and no accessories (as marked in `HandInfo.csv`) were retained.

To enable robust and repeatable evaluation, we adopted a 10-split strategy for ReID-style validation, where each split consists of disjoint query and gallery sets. While our training pipeline does not currently include data augmentations such as flipping, rotation, or photometric adjustments, the dataset itself provides natural variability in pose, lighting, and accessories. Structured preprocessing ensured consistency across splits, aligning with common biometric evaluation protocols.

These steps ensured the model was trained on clean and diverse samples while maintaining consistency with common biometric evaluation practices. The structured preprocessing and 10-split strategy allowed for rigorous and repeatable ReID-style validation of all experimental results.

3.2 Model Architecture

In this project, the Contrastive Language-Image Pre-training (CLIP) model was adopted as the core architecture. CLIP is known as a vision-language model trained to jointly align image and text representations in a shared embedding space using contrastive learning. Two variants of CLIP's image encoder were explored: ViT-B/16 (Vision Transformer) and RN50 (ResNet-50).

3.2.1 CLIP Variants: ViT-B/16 and RN50

ViT-B/16 and RN50 are small-sized variants available among multiple variants listed in the official documentation(Contributors 2023).The ViT-B/16 model was designed to divide each input image into 16×16 patches, which were then processed as tokens by a standard Transformer encoder. Long-range dependencies and global structural patterns in the image were effectively captured by this architecture. In the context of hand biometrics, the spatial relationships between components of the hand, such as finger alignment and palm structure, were better understood through this global attention mechanism.

The RN50 model was structured as a convolutional neural network (CNN) using residual connections. It was optimised for learning local features such as skin textures, finger wrinkles, and fine anatomical differences. This made RN50 particularly well-suited

for capturing detailed, discriminative features relevant to identity recognition from hand images.

By experimenting with both backbones, insights were gained into whether global (ViT-B/16) or local (RN50) feature representations were more effective for hand-based biometric matching.

3.2.2 CLIP-ReID Architectural Enhancements

To adapt CLIP for the task of hand-based re-identification, several architectural enhancements inspired by the CLIP-ReID framework were incorporated. These additions were introduced to overcome the limitations of the original CLIP architecture in learning discriminative features for identity matching and to improve retrieval accuracy in ReID tasks.

- **BNNeck:** A Batch Normalization layer, referred to as BNNeck, was applied immediately before the classification head. Through this layer, the feature embeddings used for retrieval were decoupled from those used for classification. Training stability was improved and better-structured feature distributions were enabled for both softmax and cosine-based similarity measures.

- **ArcFace Margin Head:** The standard linear classification head was replaced with an ArcFace head. By introducing an angular margin between identity classes, ArcFace forced the embeddings of different identities to be positioned further apart in the angular space. As a result, intra-class compactness and inter-class separability were enhanced — a critical requirement for ReID applications.

- **Partial Layer Unfreezing:** Rather than fully fine-tuning the entire CLIP model, only selected deeper layers of the image encoder were unfrozen. This partial unfreezing strategy allowed hand-specific visual adaptations to be made without disrupting the generic features learned from large-scale pretraining.

- **Multi-Loss Strategy:** A combination of loss functions was used to improve learning effectiveness. These included Cross-Entropy Loss for identity classification, Triplet and Center Losses for metric learning, and Supervised Contrastive Loss (SupCon) for global alignment. By integrating these losses, compact and well-separated features were learned for retrieval and classification tasks.

All enhancements were implemented using modular YAML configurations, which allowed flexible experimentation across models and training regimes.

3.2.3 PromptSG Prompt Learning Enhancements

PromptSG-based prompt tuning was implemented to enhance multimodal alignment between image and text within CLIP. Instead of relying on predefined fixed prompts, the model was guided using dynamic, learnable prompts tailored to the hand identity matching task.

- **Textual Inversion Module:** A lightweight multi-layer perceptron (MLP) was used to generate pseudo-token embeddings. These tokens encoded identity-related visual information and were inserted into textual prompts, enabling the CLIP text encoder to generate semantically rich class-level descriptions.
- **Composed Prompt Templates:** Multiple template formats were defined, such as "A surveillance photo showing a {aspect} hand," where the `{aspect}` placeholder was dynamically replaced. These templates were designed to incorporate contextual and anatomical information, making the prompt guidance more expressive.
- **Multimodal Interaction Module:** A transformer-based module was used to fuse visual and textual embeddings via cross-attention. By processing both modalities together, richer and more discriminative representations were learned for downstream classification and retrieval.
- **Classifier Head:** To maintain consistency with the CLIP-ReID structure, BN-Neck and ArcFace were also applied after the prompt fusion stage. This allowed prompt-based representations to be evaluated under the same retrieval protocol and contributed to fair performance comparisons across stages.

Through these enhancements, prompt-guided fine-tuning was enabled and CLIP's full cross-modal potential was leveraged — even in the absence of labelled text data.

3.3 Training Pipeline

The training pipeline for this project was structured into distinct stages to progressively improve CLIP's performance for hand-based identity matching. Each stage targeted specific limitations of the previous one, building towards a robust and generalisable ReID system. YAML configuration files controlled all experiments, enabling modularity, reproducibility, and precise versioning.

3.3.1 Stage 1: Baseline Evaluation (Zero-Shot)

In this stage, the pre-trained CLIP model (either ViT-B/16 or RN50) was used in its frozen state, without any additional training. Features were directly extracted using

the image encoder, and identity matching was performed using cosine similarity between query and gallery features. No classification head or fine-tuning was involved.

This zero-shot evaluation aimed to test CLIP's native generalisation ability to unseen hand images and acted as a true baseline for comparison. Despite being trained on internet-scale image-text pairs, CLIP had never seen biometric data like hands, so this stage highlighted its limitations in this specific domain.

3.3.2 Stage 2: CLIP-ReID Integration

To address the domain gap between generic vision-language training and biometric identification, this stage incorporated advanced ReID components:

- **BNNeck:** A Batch Normalisation neck was inserted before the classification layer. It helped separate feature spaces for retrieval and classification, a best practice in ReID tasks to prevent feature collapse and enhance robustness.
- **ArcFace Head:** Instead of a simple softmax classifier, ArcFace was used to introduce an angular margin during training. This forced tighter intra-class clusters and better inter-class separation in the embedding space, making the features more discriminative.
- **Loss Combination:** Multiple loss functions were jointly optimised: Cross-Entropy for identity classification, Triplet Loss for relative distance supervision, Center Loss for intra-class compactness, and SupCon (Supervised Contrastive Loss) for global feature alignment. Loss weights were tunable via YAML and scheduled across versions.
- **Partial Encoder Unfreezing:** Starting from v6, the image encoder was partially unfrozen via a configurable `unfreeze_blocks` parameter (e.g., last 2 transformer blocks for ViT, last ResNet stages for RN50). This provided flexibility in fine-tuning depth, allowing better adaptation without full retraining.

This CLIP-ReID setup significantly improved performance by adapting CLIP to the hand image domain using structured supervision.

3.3.3 Stage 3: PromptSG Integration and Joint Fine-Tuning

In this stage, prompt learning was introduced via the PromptSG framework to enhance CLIP's alignment between visual and textual features. Instead of relying on static prompts like "a photo of a hand", the following improvements were added:

- **Textual Inversion:** A small MLP (learnable pseudo-token generator) was trained to convert image features into pseudo-text tokens that describe the hand's identity implicitly. This served as a bridge from image to text space.
- **Composed Prompts:** Templates such as "A captured frame showing a person's `{aspect}` hand during surveillance" were used to inject semantic structure. The aspect (e.g., dorsal right) was dynamically filled in, and prompt templates were averaged during validation for robustness.
- **Multimodal Fusion:** A lightweight transformer fused image and text embeddings using cross-attention. The output was mean-pooled, normalised, and used for classification and contrastive learning.
- **Reduction + BNNeck + ArcFace:** A dimensionality reduction layer, followed by BNNeck and ArcFace, was used to produce final embeddings, ensuring consistency with Stage 2 and improving training stability and generalisation.
- **Joint Training:** Both the CLIP image encoder and PromptSG modules were fine-tuned together using a multi-loss objective (Cross-Entropy, Triplet, SupCon). The SupCon loss weight was gradually increased with training epochs to stabilise learning.

PromptSG aligned CLIP's powerful vision-language features with the biometric identity task, providing further gains especially in scenarios where textual cues could enhance discrimination.

3.3.4 Checkpointing and Resume Features

Each training stage implemented automatic checkpointing:

- Best model (based on validation Rank-1 or mAP) was saved as _BEST.pth.
- Training could resume from the last checkpoint in case of interruptions.
- Evaluation always used the best saved model to ensure fair comparison.

3.3.5 YAML Configuration and Modular Design

Every experiment was controlled via a dedicated YAML configuration file. Key parameters such as:

- Model type (ViT-B/16, RN50)
- Dataset path and aspect

- Loss weights and types
- Learning rates and schedulers
- Prompt templates and training mode

were defined in the config, promoting reproducibility and fast experimentation.

All training scripts (e.g., `train_stage1_joint.py`, `train_stage2_promptsg.py`) were modular, allowing flexible switching of backbones, stages, and features through simple config edits—without needing to alter the core logic.

The complete directory and file structure of the project, including configuration files, training logs, evaluation scripts, and utility modules, is documented in **Appendix A**. This structure highlights the modularity and scalability of the implementation, supporting multiple training stages, model variants, and evaluation pipelines.

3.4 Loss Functions

To optimise the model for hand-based identity matching, several loss functions were explored across different stages of training. Each loss was introduced to serve a particular role — such as improving classification accuracy, enhancing the structure of the feature space, or encouraging feature separation between identities. While not all loss functions were used in the final model, various combinations were systematically tested to evaluate their contribution.

3.4.1 Cross-Entropy Loss (ID Classification)

Cross-Entropy Loss(PyTorch Contributors 2024a) was used as the primary objective for identity classification. During training, this loss measured how well the model predicted the correct identity label for each hand image. A comparison was made between the predicted class probabilities and the actual ground truth label, and the model was updated to reduce this difference.

In Stage 1 and Stage 2, Cross-Entropy Loss was applied with a standard softmax classifier, and later with the ArcFace margin-based head. In both cases, this loss provided direct supervision to learn identity-specific features.

3.4.2 Triplet Loss (Distance-Based Embedding Learning)

Triplet Loss(PyTorch Contributors 2024b) was introduced to help the model learn a better embedding space where images of the same identity were placed closer together and images of different identities were pushed further apart. Triplets were formed using:

- An anchor image

- A positive image (same identity as anchor)
- A negative image (different identity)

The model was encouraged to minimise the distance between the anchor and positive images while ensuring that the anchor was at least a margin farther away from the negative image. A margin of 0.3 was adopted in most experiments. Triplet Loss was applied from Stage 2 onwards, once the image encoder was unfrozen for fine-tuning.

3.4.3 Center Loss (Intra-Class Compactness)

To reduce variations among features of the same identity, Center Loss(Zhou 2018) was tested in conjunction with Cross-Entropy. In this case, each identity class was assigned a center point in the feature space, and the model was trained to minimise the distance between the features and their respective class centers.

This loss was helpful in tightening the feature clusters of the same identity. It was introduced in Stage 2 and carried forward to Stage 3 experiments. A small loss weight (typically 0.0005) was assigned to prevent it from dominating the total loss.

3.4.4 ArcFace Loss (Angular Margin Softmax)

ArcFace Loss(Gokoruri007 2021) was used as a more discriminative alternative to standard softmax classification. Instead of using raw class scores, this loss computed the angle between feature vectors and class weight vectors, and enforced an angular margin between them. This technique led to better separation of identity clusters and was especially effective in open-set recognition tasks like ReID.

ArcFace was used as the classification head during Stage 2 and Stage 3 training and worked together with BNNeck and normalization layers to ensure feature stability. Margin and scale hyperparameters were tuned during experimentation (e.g., margin = 0.3 to 0.4, scale = 20 to 35).

3.4.5 Supervised Contrastive Loss

Supervised Contrastive (SupCon) Loss(HobbitLong 2021) was introduced in finetuning strategies. This loss extended InfoNCE(A. v. d. Oord, Y. Li, and Vinyals 2018) by allowing multiple positive samples per anchor, which is particularly effective when P×K sampling is used.

3.4.6 Loss Combination and Scheduling

While different losses were explored individually, multiple loss functions were also combined during training for maximum effectiveness. For instance, in Stage 2, Cross-Entropy,

Triplet Loss, and ArcFace were jointly optimised. In Stage 3, SupCon was added to the objective along with ArcFace and Center Loss.

Loss weights and contributions were scheduled using YAML configs, allowing experiments to suppress or boost specific loss terms depending on model behaviour (e.g., lowering Triplet weight when overfitting occurred, or increasing SupCon in joint training).

This combination strategy allowed the model to learn not only class labels but also meaningful distances and structures in the embedding space, making it more reliable for real-world hand re-identification tasks.

3.5 Evaluation Metrics

To fairly and systematically evaluate the performance of the trained models, standard metrics from the person re-identification (ReID) community were used. These metrics measure both the ranking quality and the retrieval ability of the models.

3.5.1 Mean Average Precision (mAP)

Mean Average Precision (mAP)(Beitzel, Jensen, and Frieder 2009) is a widely used evaluation metric for retrieval tasks such as person or hand re-identification. It quantifies how well a model ranks relevant (i.e., same identity) images ahead of irrelevant ones in the retrieval list. For each query, the precision is computed at the ranks where relevant items appear, and the average of these precision values is called the Average Precision (AP). The final mAP is obtained by averaging the AP over all queries.

Formally, for a given query q, let P_k be the precision at rank k, and let rel_k be an indicator function that is 1 if the item at rank k is relevant, and 0 otherwise. The AP for query q is:

$$\text{AP}(q) = \frac{1}{R_q} \sum_{k=1}^{N} P_k \cdot rel_k$$

where:

- N is the total number of retrieved items,
- R_q is the total number of relevant items for query q,
- P_k is the precision at cutoff k.

Then, the mean Average Precision across all Q queries is computed as:

$$\text{mAP} = \frac{1}{Q} \sum_{q=1}^{Q} \text{AP}(q)$$

In simple terms, mAP reflects how well the model ranks correct hand images higher in the retrieval list. It provides a comprehensive evaluation of retrieval performance by considering both the ranking order and the presence of all relevant results. Unlike Rank-1 or Rank-5 accuracy, which check only specific top positions, mAP captures the overall ranking quality.

3.5.2 Rank-1, Rank-5, and Rank-10 Accuracy

Rank-1, Rank-5, and Rank-10 accuracies are used to measure how often the correct match appears among the top-k retrieved results for each query:

- **Rank-1 Accuracy:** The percentage of queries where the correct match is the top-most retrieved image.
- **Rank-5 Accuracy:** The percentage of queries where the correct match is among the top 5 retrieved images.
- **Rank-10 Accuracy:** The percentage of queries where the correct match is among the top 10 retrieved images.

These metrics are easy to understand and are important for practical applications, where users usually expect the correct match to appear near the top of the search results.

3.5.3 Evaluation Protocol: 10 Query-Gallery Splits (Monte Carlo Estimation)

Following the MBA-Net protocol, evaluation was conducted using **10 randomly sampled query-gallery splits**, effectively forming a Monte Carlo-style estimation framework(Hastie, Tibshirani, and Friedman 2001). In each split:

- A random subset of images was selected to form the **query set**.
- The remaining images were used as the **gallery set**.
- The model retrieved gallery images for each query using cosine similarity, and performance metrics were computed.

This 10-fold evaluation strategy mimics a Monte Carlo cross-validation procedure, where performance is measured over multiple random resamplings. The final reported metrics — **mean Average Precision (mAP)**, **Rank-1**, **Rank-5**, and **Rank-10** accuracies — are averaged across all 10 splits.

This reduces the impact of any single random sampling and provides a statistically robust estimate of model generalisation. It also ensures that performance trends are not biased by a particular partitioning of the data.

3.5.4 Similarity Computation

For ranking and retrieval:

- **Cosine Similarity** was used to compute the similarity between query and gallery features.

Cosine similarity measures the cosine of the angle between two feature vectors, focusing on their direction rather than magnitude. This is suitable for feature embeddings, which are often normalised.

Although advanced indexing methods like FAISS(Douze et al. 2024) could be used for faster large-scale retrieval, in this project, direct cosine similarity computation was sufficient because of the manageable dataset size.

3.6 Limitations

While the methodology adopted in this project was carefully designed to meet the research objectives, several limitations must be acknowledged. Identifying these constraints provides important context for interpreting the results and outlines areas where future improvements can be made.

3.6.1 Dataset Bias and Imbalance

Only the 11k Hands dataset was used in this project. While it is relatively large by hand biometrics standards, it is still limited in comparison to the scale of datasets typically used in deep learning and vision-language research.

Additionally, images containing accessories (e.g., rings, watches) were excluded as per standard preprocessing, which improves visual clarity but introduces selection bias by removing challenging real-world cases. Moreover, although the dataset includes a range of demographics, the distribution of age, gender, and ethnicity is not balanced, which can affect the generalisability of trained models to underrepresented groups.

3.6.2 Risk of Overfitting and Under-Generalisation

Fine-tuning large models like CLIP on relatively small and specialised datasets poses a risk of overfitting. The possibility that the model may learn dataset-specific artefacts rather than generalisable biometric features.

Conversely, under-generalisation can occur when insufficient adaptation is applied — for example, if the image encoder remains too close to its pre-trained weights on natural images, it may fail to capture discriminative patterns specific to hand biometrics. Finding

the right balance between freezing, partial unfreezing, and full fine-tuning proved to be a critical challenge, particularly when working with ViT-B/16 and RN50 backbones.

3.6.3 Zero-Shot and Fine-Tuning Gaps

In the zero-shot (Stage 1) evaluation, no fine-tuning was applied, and CLIP's original embeddings were used directly. While this highlights CLIP's transfer learning potential, it also exposes a gap — CLIP was not trained on biometric or anatomical images, and its zero-shot performance on hand images was limited. This highlights the domain gap between general image-text pretraining and the requirements of biometric recognition, which future work may address via domain-adaptive pretraining or larger-scale fine-tuning.

3.6.4 CLIP's Original Training Domain

CLIP was originally trained on a very large and diverse set of internet images and text pairs, but it was not specifically trained for biometric applications like hand recognition. As a result, the features learned by CLIP may not naturally capture the fine-grained discriminative details needed for identifying individuals based on subtle hand differences.

Even after fine-tuning, the model might still carry some biases from its original training domain, which could limit its full potential for hand-based biometrics.

3.6.5 Computational Constraints

Fine-tuning large vision-language models like CLIP requires significant computational resources, especially when using contrastive losses and prompt learning strategies. In this project, due to limited GPU availability and memory constraints, batch sizes had to be kept relatively small, the space explored in hyperparameter space was limited, and larger model variants (e.g., ViT-B/32, ViT-L/14, ViT-G/14, and others) could not be explored.

3.6.6 Evaluation Scope

Finally, although the evaluation followed strong ReID protocols (10 splits, Rank-1, mAP, etc.). Wider testing on more diverse hand datasets would be needed to fully prove the generalisation capability of the fine-tuned CLIP models.

4 Implementation

This chapter describes the practical implementation of the methods and experiments outlined in the previous chapter. It explains the tools, frameworks, and programming libraries that were selected, the structure of the codebase, the configuration management system, and the overall design of the training pipeline.

The focus is on showing how the CLIP models were adapted for hand-based identity matching through careful engineering and modular programming practices. Special attention was given to maintaining flexibility, reproducibility, and scalability in the experimental setup, allowing easy switching between different model architectures, datasets, and loss functions.

This chapter also explains how clipreid has been integrated into the project, and prompt learning was integrated into the training flow using the PromptSG approach. It highlights technical challenges faced during development, along with the solutions applied. The goal is to demonstrate technical competence, professional project organisation, and critical problem-solving skills necessary for successful deep learning research.

4.1 Tools and Frameworks

The implementation of this project was carried out using a variety of modern tools, frameworks, and libraries that are widely adopted in the deep learning research community. The selection of tools was based on their reliability, flexibility, and ability to support rapid experimentation and scalability.

4.1.1 Python

Python(Python Software Foundation 2025) was chosen as the main programming language because of its simplicity, rich ecosystem, and strong community support in the machine learning field. Its extensive set of libraries makes it an ideal choice for developing and testing deep learning models efficiently.

4.1.2 PyTorch

PyTorch(Paszke et al. 2019) was used as the primary deep learning framework for model development and training. PyTorch provides a dynamic computational graph, making it easier to debug and modify complex models such as CLIP. Its integration with CUDA allows seamless GPU acceleration, which is critical for training large vision-language models. PyTorch's flexible tensor operations, autograd engine, and strong support for distributed training made it a natural choice for this project.

4.1.3 OpenAI CLIP Library

The OpenAI CLIP model(Contributors 2023), available through the official 'openai/-CLIP' repository, was used as the starting point for the project. The pre-trained CLIP weights provided a strong base for fine-tuning on hand-based identity matching tasks. The library allowed easy access to both ViT-B/16 and RN50 backbone variants, enabling systematic comparisons across different architectures.

4.1.4 YAML for Configuration Management

YAML (Yet Another Markup Language) was used for managing experiment configurations. All hyperparameters, dataset paths, model selections, loss function toggles, and save directories were stored in structured YAML files. Using YAML configurations helped maintain clean separation between the codebase and experimental settings, improving reproducibility and making it easier to track different experiments systematically.

4.1.5 Development Environment

The project was developed and tested using the following tools and workflows:

- **PyCharm 2024.2.6:** All core scripts for training, evaluation, preprocessing, and configuration management were developed using PyCharm(JetBrains 2025). This integrated development environment (IDE) provided powerful debugging, version control integration, and efficient code navigation, which streamlined large-scale script development and experiment tracking.
- **Python Scripts:** The complete training pipeline, including stage-wise fine-tuning, dataloaders, and evaluation modules, was implemented using modular Python scripts. This structure enabled scalable experimentation with YAML-based configuration files and seamless reproducibility.

This combination of development environments ensured that the project could be built systematically, tested rigorously, and reproduced reliably across experiments.

4.2 Codebase Structure and Execution

The codebase for this project was carefully structured to support modularity, readability, and ease of extension. A clean and logical folder organisation was maintained to separate different components such as training scripts, configuration files, models, datasets, and utilities.

A complete codebase directory hierarchy is provided in **Appendix A**. Additionally, detailed instructions on how to set up the environment, prepare the dataset, run each training and evaluation stage, and generate visualisations are included in **Appendix B**. This ensures full reproducibility and clarity for future development or research extensions.

4.2.1 Folder and File Layout

The project was organised with modularity, supporting multi-stage training, evaluation, logging, and reproducibility. The key components are as follows:

- **configs/** — Contains all YAML configuration files for training and evaluation. It is organised by stage:
  - `train_stage2_clip_reid/` and `train_stage3_promptsg/` include training configs for CLIP-ReID and PromptSG respectively.
  - `eval_stage2_clip_reid/` and `eval_stage3_promptsg/` hold configs used during evaluation.
  - `baseline/` is for stage 1 - zeroshot evaluation configurations.
- **datasets/** — Handles dataset loading and preprocessing:
  - `11khands/` contains loaders for the 11k Hands dataset.
  - `data_preprocessing/` includes scripts for resizing, cropping, and formatting the dataset:`11khands/`.
- **engine/** — Implements training and evaluation logic:
  - `train_clipreid_stages.py` define the full training pipeline.
  - `prompt_learner.py` and `promptsg_inference.py` handle PromptSG logic.
  - `evaluator.py` runs evaluation and logs mAP, Rank@K.
- **experiments/** — Contains stage-wise organised experiments:
  - `stage1_baseline_inference/` — Zero-shot CLIP evaluations.

  - `stage2_clipreid_integration/` — CLIP-ReID integration logic.
  - `stage3_promptsg_integration/` — PromptSG integration logic.

- **utils/** — Contains utility scripts for:
  - Logging, loss functions, checkpoint saving, and device selection.
  - `loss/` folder houses implementations of Triplet, SupCon, ArcFace, and other loss functions.

- **train_logs/, eval_logs/** — Store logs, metric outputs, and models for various runs.

- **saved_models/** — Stores checkpoints of best-performing models from each stage and backbone.

- **Root files:**
  - `main.py` — Entry-point script for launching training/evaluation.
  - `README.md` and `requirements.txt` — Usage instructions and environment setup.

4.2.2 Naming Conventions

The codebase follows consistent naming practices for better clarity and maintenance:

- **Training Scripts:**
  - `clipreid_trainer_stage1.py`, `clipreid_trainer_stage2.py` — CLIP-ReID training for Stage 1 and 2.
  - `train_clipreid_stages.py` — Unified joint training pipeline.
  - `prompt_learner.py`, `promptsg_inference.py` — Prompt learning training and evaluation.

- **YAML Configs:**
  - `train_stage2_clip_reid/`, `train_stage3_promptsg/` — Training configs for each stage.
  - `eval_stage2_clip_reid/`, `eval_stage3_promptsg/` — Evaluation configs by stage.

Each config is named to reflect its backbone (e.g., RN50, ViT-B/16), dataset split, and training goal for traceability.

4.3 Challenges Encountered

During the development and training of the CLIP-based hand identity matching system, several technical challenges were encountered. Addressing these issues required careful debugging, testing, and the design of appropriate workarounds.

4.3.1 Feature Dimensionality Mismatches

One major challenge arose due to the difference in output feature dimensions between the two CLIP backbones:

- ViT-B/16 produces 512-dimensional image embeddings.
- RN50 produces 1024-dimensional image embeddings.

This mismatch caused issues when applying losses like Supervised Contrastive Loss, which expects a consistent feature dimension across all samples in a batch.

Solution: A dynamic feature projection layer was introduced, where RN50 embeddings were passed through a linear transformation layer to project them into a 512-dimensional space, matching the ViT-B/16 outputs. This allowed the same loss functions and batch structures to be applied uniformly across different backbones.

4.3.2 Supervised Contrastive (SupCon) Input Shape Problems

During the initial integration of SupCon Loss, errors occurred because the input batch did not always satisfy the positive-pair requirement needed for proper contrastive learning. In some cases, batches had only one sample per identity, making contrastive training ineffective.

Solution: A custom P×K Sampling Strategy was enforced rigorously during dataloader preparation, ensuring that for each identity sampled in a batch, at least $K = 2$ images were present. This maintained enough positive pairs for SupCon Loss to work correctly.

4.3.3 Memory Issues with Large Batch Sizes

Fine-tuning large models like CLIP with contrastive losses requires large batch sizes to be effective. However, GPU memory limitations prevented the use of very large batches, especially when using high-resolution hand images and when enabling gradient accumulation.

Solution: Several strategies were applied:

- Gradient accumulation was used to simulate larger batch sizes without exceeding memory limits.

- Mixed-precision (float16) training was explored where supported to reduce memory footprint.
- Smaller model variants (like ViT-B/16 instead of larger ViT-L/14) were prioritised for efficient fine-tuning.

4.3.4 Checkpoint and Resume Stability

During long training runs, especially with multi-stage pipelines, occasionally saved checkpoints were found to be incomplete, missing optimizer states or learning rate scheduler states.

Solution: Checkpoint saving logic was revised to ensure that:

- Full model state, optimizer state, and scheduler state were saved together.
- Resuming from checkpoint properly restored all training variables, allowing smooth continuation of training without restarting, depends upon the integration we have done, either CLIP Re-ID or PromptSG.

—

Overcoming these challenges helped improve the overall robustness of the training pipeline and ensured that experiments could be conducted systematically and reliably across different models, datasets, and training strategies.

5 Results

This chapter presents a detailed analysis of the experimental findings obtained across the three stages of the CLIP-FH pipeline—Stage 1 (baseline zeroshot evaluation), Stage 2 (CLIP-ReID fine-tuning), and Stage 3 (PromptSG-guided joint training). The primary objective is to interpret the performance trends, architectural choices, and training strategies in the context of hand-based identity matching using vision-language models. By comparing the behaviour of two prominent CLIP backbones—ViT-B/16 and RN50—this chapter highlights how each stage contributed to performance gains, how specific enhancements (e.g., BNNeck, ArcFace, prompt tuning) influenced representation quality, and how model robustness evolved under different supervisory conditions. The discussion also draws attention to comparative strengths, failure points, and the underlying reasons for the observed differences across models and configurations.

5.1 Stage 1: Baseline (Zeroshot) CLIP Evaluation

In Stage 1, the baseline performance of the original pre-trained CLIP models (ViT-B/16 and RN50) was evaluated on the dorsal-r aspect of 11k Hands dataset without any fine-tuning. The CLIP image encoders were kept entirely frozen, and no classifier or adaptation layers were applied. Instead, embeddings were directly extracted from the image encoder and compared using cosine similarity to perform retrieval-based identity matching.

All feature vectors were L2-normalized before similarity computation to ensure fair comparison, and cosine similarity was used as the metric due to its robustness to feature scaling. No data augmentation or prompt learning was used at this stage.

5.1.1 ViT-B/16 Baseline Results

Table 5.1 reports the zeroshot performance of ViT-B/16 on the dorsal right subset of the 11k Hands dataset across 10 query-gallery splits.

Table 5.1: Stage 1: CLIP ViT-B/16 Baseline Performance on 11k Hands (Dorsal Right)

Split	Rank-1 (%)	Rank-5 (%)	Rank-10 (%)	mAP (%)
Split 1	66.22	85.68	93.10	74.99
Split 2	70.03	87.02	91.86	77.44
Split 3	74.67	89.29	93.92	81.21
Split 4	72.09	86.71	93.10	78.87
Split 5	70.96	88.57	95.06	78.84
Split 6	71.68	90.11	94.95	79.62
Split 7	68.90	86.51	92.89	76.99
Split 8	73.94	91.56	95.26	81.30
Split 9	67.46	86.51	91.86	75.93
Split 10	78.27	93.31	97.43	84.58
Mean	**71.42**	**88.53**	**93.94**	**78.98**

5.1.2 RN50 Baseline Results

Table 5.2 presents the zeroshot performance of RN50 under identical settings.

Table 5.2: Stage 1: CLIP RN50 Baseline Performance on 11k Hands (Dorsal Right)

Split	Rank-1 (%)	Rank-5 (%)	Rank-10 (%)	mAP (%)
Split 1	63.23	83.21	90.94	72.47
Split 2	66.74	84.14	90.01	74.97
Split 3	71.37	83.93	90.01	77.52
Split 4	68.28	83.52	89.91	75.42
Split 5	64.47	82.08	90.22	72.43
Split 6	71.88	83.93	89.91	77.83
Split 7	68.59	84.04	90.73	75.60
Split 8	69.82	83.83	88.98	76.06
Split 9	69.93	83.73	92.07	76.92
Split 10	71.78	86.82	91.56	78.54
Mean	**68.61**	**83.92**	**90.43**	**75.78**

5.1.3 Comparison: ViT-B/16 vs RN50 (Zeroshot)

The following observations can be drawn from the above results:

- ViT-B/16 achieved a higher mean Rank-1 accuracy (71.42%) and mAP (78.98%) compared to RN50 (Rank-1: 68.61%, mAP: 75.78%).
- ViT-B/16 also outperformed RN50 on higher ranks and achieved the highest individual split performance (Rank-1 of 78.27% in Split 10).

These results confirm that even without any fine-tuning, CLIP's ViT-B/16 backbone provides a stronger zeroshot representation for hand-based identity matching compared

to RN50. However, both models show significant room for improvement, motivating the subsequent fine-tuning stages explored in this project.

5.2 Stage 2 – CLIP-ReID Integration (Fine-Tuning with ReID Enhancements)

In Stage 2, CLIP was adapted to the ReID setting using architectural enhancements from the CLIP-ReID framework. This included BNNeck normalization, ArcFace margin-based classification, and a multi-loss training strategy involving supervised contrastive learning, Triplet, and Center losses. We explored various schedulers, optimizers, and partial fine-tuning strategies on both ViT-B/16 and RN50 backbones.

5.2.1 ViT-B/16 Results

Evaluation

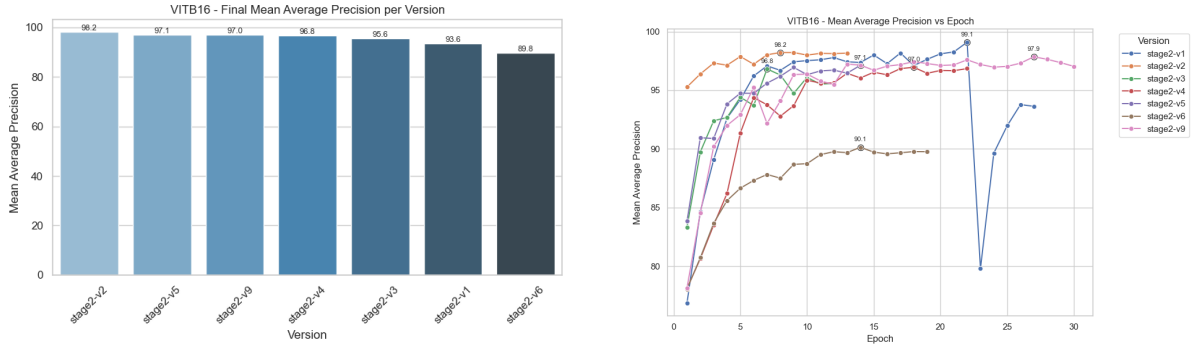
The ViT-B/16 backbone demonstrated a clear upward trend in performance as architectural and training refinements were introduced. Starting from a Rank-1 of 62.11% and mAP of 71.35% in v1, the model benefited significantly from enhancements in template diversity, optimizer switching (Adam to AdamW), and projection normalization (v2), reaching 83.19% mAP. The use of OneCycleLR in v3 pushed this further to 89.91% mAP and 84.82% Rank-1.

The best performance was achieved in version v5, where all five losses were retained (ID, Triplet, Center, ArcFace, SupCon) and ArcFace was softened (scale=20, margin=0.3), resulting in 88.18% Rank-1 and 92.20% mAP. Ablation studies (v4) confirmed the importance of Triplet and Center loss, as their removal led to a drop in mAP to 81.93%. Partial fine-tuning (v6) of only the last two transformer blocks led to moderate gains but did not outperform v5. Notably, version v9, which applied extreme ArcFace parameters and removed Triplet loss, caused a severe drop to 47.23% mAP—highlighting that aggressive margin scaling can destabilize training.

Table 5.3: Stage 2: CLIP-ReID Results on 11k Hands (ViT-B/16 Backbone)

Version	Remarks	Rank-1	Rank-5	Rank-10	mAP
v1	Baseline dual-stage setup: full ViT-B/16 encoder, frozen prompt encoder from Stage 1; all five losses active; Adam; CosineAnnealingLR.	62.11	82.66	88.80	71.35
v2	Added prompt template diversity; projection norm; AdamW with split LR; all losses retained.	75.90	92.42	95.59	83.19
v3	Switched to OneCycleLR (per batch); retained v2 settings — best ViT performance.	84.82	96.40	98.25	89.91
v4	Ablation: Triplet and Center losses removed; retained ArcFace, SupCon, and ID.	74.07	91.78	96.27	81.93
v5	Restored Triplet and Center losses; ArcFace softened (scale=20, margin=0.3).	**88.18**	**97.32**	**98.58**	**92.20**
v6	Partial fine-tuning (unfreeze last 2 transformer blocks); retained v5 loss setup.	76.08	91.70	95.75	83.03
v9	Extreme ArcFace (scale=35, margin=0.4); Triplet OFF; center_loss_weight=0.0003.	28.88	70.13	86.14	47.23

Training and Validation



(a) Final mAP per version (ViT-B/16)

(b) mAP vs Epoch (ViT-B/16)

Figure 5.1: Evaluation results of ViT-B/16 across Stage 2 variants. Best result in v2 with mAP 98.2%.

As illustrated in Figure 5.1, the ViT-B/16 backbone demonstrated consistent improvements across Stage 2 configurations. Version **v2** achieved the highest mAP (98.2%), outperforming all other variants. This version incorporated prompt diversity, AdamW optimizer with split learning rates, and projection normalization — a combination that significantly stabilized convergence, as reflected in the sharp early rise and plateau in the mAP-vs-epoch curve.

Versions **v5** (97.1%) and **v9** (97.0%) also performed exceptionally well, reinforcing the effectiveness of multi-loss training (ID, Triplet, Center, ArcFace, SupCon). Meanwhile, version **v6** showed the lowest performance (89.8%), likely due to partial fine-tuning that restricted the model's capacity to adapt deeper features. The epoch-wise trends show early convergence in all strong configurations, with minimal overfitting, indicating stable optimization strategies for ViT-B/16.

Observations: Training, Validation, and Evaluation Alignment

ViT-B/16 showed strong alignment between validation and evaluation outcomes across Stage 2 variants:

- **Stable Improvements Across Versions:** Enhancements such as prompt diversity (v2), OneCycleLR (v3), and softened ArcFace (v5) led to steady performance gains, with minimal overfitting.
- **Validation Trends Closely Tracked Evaluation:** Top-performing versions (v2, v5) had consistent mAP growth during training, with final validation metrics aligning well with evaluation results (v2: 98.2%, v5: 97.1%).
- **Multi-Loss Synergy Critical:** Removing Triplet and Center loss (v4) reduced performance, while full-loss setups (v5) achieved highest mAP, validating the combined benefit of all five losses.
- **Fine-Tuning Depth Matters:** Partial unfreezing (v6) underperformed compared to full fine-tuning, showing that deeper transformer tuning is essential for ViT.
- **Aggressive ArcFace Destabilizes:** v9 confirmed that extreme ArcFace scaling without Triplet loss causes sharp drops in both validation and evaluation (mAP 47.2%).

5.2.2 RN50 Results

Evaluation

For the RN50 backbone, the baseline configuration (v1) achieved the highest mean Average Precision (mAP) of 67.07%, despite its simplicity. It used the full RN50 encoder with all loss functions (Cross-Entropy, SupCon, Triplet, Center) active and relied on the CosineAnnealingLR scheduler. This suggests that RN50 responded well to a fully supervised, stable training regime without the need for prompt-specific or margin-intensive tuning.

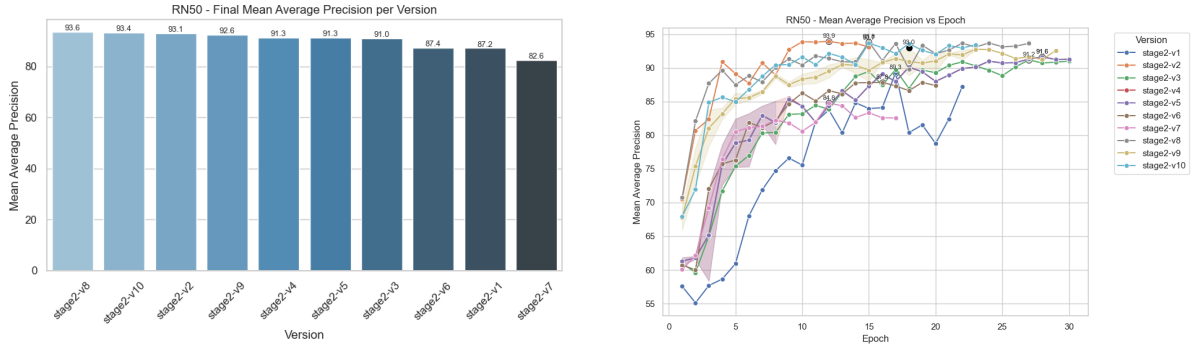
Subsequent versions attempted various improvements, such as prompt template diversity (v2), scheduler changes (v3), and ArcFace softening (v5), but none surpassed the performance of the baseline. Version v8, which involved deeper partial fine-tuning with ArcFace (scale=30, margin=0.35), achieved the second-best mAP at 64.30%, showing that architectural adjustments helped to some extent but not beyond the baseline.

Interestingly, full encoder fine-tuning in v9 and v10 did not yield better generalisation, indicating that aggressive adaptation may have led to overfitting or suboptimal representation learning. Overall, the results confirm that RN50's best performance came from a carefully balanced, all-loss training setup without unnecessary complexity.

Table 5.4: Stage 2: CLIP-ReID Results on 11k Hands (RN50 Backbone)

Version	Remarks	Rank-1	Rank-5	Rank-10	mAP
v1	Baseline: full RN50 encoder with BNNeck, ArcFace, SupCon, Triplet, Center; prompt frozen; Adam + Cosine LR.	**57.64**	**77.74**	**85.09**	**67.07**
v2	Adopted prompt template diversity, projection norm, AdamW with split LR.	46.09	71.13	80.10	57.59
v3	Switched to OneCycleLR (batch-wise); retained v2 settings.	53.37	73.68	82.52	62.85
v4	Removed Triplet and Center losses; retained ArcFace, SupCon, ID.	50.82	72.91	82.19	61.34
v5	Softened ArcFace (scale=20, margin=0.3); all losses retained.	53.74	75.44	84.34	63.69
v6	Partial fine-tuning (unfreeze 2 ResNet blocks); retained v5 setup.	45.09	70.30	80.19	56.47
v7	RN50-specific: ArcFace (scale=25, margin=0.35); lr=5e-4.	51.62	76.08	84.69	62.63
v8	Deeper partial FT (4 blocks); ArcFace (scale=30, margin=0.35); Triplet and Center ON.	52.97	78.04	85.97	64.30
v9	Full encoder FT; ArcFace (scale=35, margin=0.4); Triplet OFF; center_loss_weight=0.0003.	53.43	74.49	83.04	63.49
v10	Fine-tuned from v9; Triplet ON; ArcFace (scale=30, margin=0.4).	51.52	74.93	83.98	62.26

Training and Validation



(a) Final mAP per version (RN50)

(b) mAP vs Epoch (RN50)

Figure 5.2: Performance of RN50 across Stage 2 variants. Best results observed in v8 (93.6% mAP), closely followed by v10 (93.4%) and v2 (93.1%).

As shown in Figure 5.2, RN50 exhibited strong results in Stage 2 when carefully tuned. The highest final mean Average Precision (mAP) was recorded in version **v8** (93.6%), followed closely by **v10** (93.4%) and **v2** (93.1%). These versions leveraged architectural tuning—such as softened ArcFace margins and prompt diversity—as well as stable learning rate schedules.

The mAP-vs-epoch trends show that versions like **v8**, **v9**, and **v10** achieved early convergence within the first 10 epochs and maintained stable performance with minimal fluctuation. This consistency is indicative of well-balanced training dynamics.

Meanwhile, versions **v4** and **v5** converged more slowly and exhibited mild performance oscillations, despite reaching solid final mAP scores (91.3%). These oscillations may be

attributed to suboptimal scheduler configurations or loss imbalance.

Lower-performing versions such as **v1** (87.2%) and **v7** (82.6%) lacked key enhancements like ArcFace tuning or proper loss weighting. Their mAP plateaued significantly lower and demonstrated erratic training trajectories.

In conclusion, RN50 requires more careful configuration than ViT-B/16 to reach peak performance. Nonetheless, with thoughtful tuning—including prompt engineering, multi-loss synergy, and stabilized schedulers—it can deliver highly competitive results.

Observations: Training, Validation, and Evaluation Alignment

RN50 showed more sensitivity to training configuration, but still achieved strong alignment between validation and evaluation under tuned setups:

- **Baseline Simplicity Outperformed Others:** Version v1, with all losses and a simple Cosine LR schedule, surprisingly yielded the highest evaluation mAP (67.07%), showing that RN50 benefited from stable, well-regularized training without prompt complexity.
- **Evaluation Did Not Always Follow Validation:** Versions like v8 and v10 had high validation mAPs (93%+), yet lower evaluation mAPs (64%), suggesting overfitting or reduced generalization despite smooth training curves.
- **ArcFace Adjustments Helped:** Versions v5 and v7–v10 confirmed that tuning ArcFace margins and scaling improved convergence stability and final metrics, especially when paired with Triplet and Center loss.
- **Multi-Loss Setup Crucial:** As with ViT, ablation of Triplet and Center (v4) lowered performance, reinforcing their role in regularizing RN50's representation.
- **Deeper Fine-Tuning Needs Caution:** Full fine-tuning (v9/v10) did not significantly outperform partial setups and sometimes led to unstable validation patterns, indicating RN50 may overfit more easily than ViT.

5.2.3 Comparison: ViT-B/16 vs RN50 (Stage 2)

When comparing the two CLIP image encoder backbones—ViT-B/16 and RN50—under the CLIP-ReID fine-tuning pipeline, several key insights emerged regarding performance trends and training sensitivity.

Firstly, ViT-B/16 consistently outperformed RN50 across almost all evaluation metrics. The best-performing ViT-B/16 configuration (v5) achieved a Rank-1 accuracy of **88.18%** and an mAP of **92.20%**, demonstrating a strong capability to learn robust and generalisable hand representations. RN50, while achieving a peak Rank-1 of **57.64%** and

the highest mAP of **67.07%** in its baseline version (v1), failed to surpass this baseline despite extensive experimentation across subsequent versions.

Secondly, ViT-B/16 showed a clear upward trend in performance as loss functions, prompt configurations, and schedulers were tuned across versions. It responded positively to prompt diversity, OneCycleLR scheduling, and margin-softened ArcFace. In contrast, RN50 demonstrated greater instability and sensitivity to architectural changes. For example, incorporating prompt diversity and projection normalization in v2 significantly reduced RN50's mAP (57.59%), compared to v1's superior 67.07%.

Moreover, while ViT-B/16 benefited from full encoder fine-tuning, RN50 performed best with a more conservative setup (v1) and did not gain much from partial or full unfreezing. Attempts to fine-tune deeper ResNet layers (v6–v8) led to performance fluctuations, and full unfreezing in v9–v10 offered only marginal or negative returns.

In summary, ViT-B/16 proved to be more robust, scalable, and adaptable to ReID-specific enhancements. Its transformer-based architecture enabled better global feature encoding and more stable convergence. RN50, though achieving a strong baseline result, exhibited fragility to architectural modifications, underlining the importance of careful tuning when adapting CNN-based models in cross-modal frameworks.

5.3 Stage 3: PromptSG Integration (Fine-tuning CLIP using PromptSG Enhancements)

This stage involved enhancing the CLIP framework by integrating prompt-guided supervision, inspired by cross-modal alignment principles. Instead of relying solely on image features, a learnable prompt mechanism was introduced to capture identity-specific semantics through a fusion of visual and textual cues. The image encoder and auxiliary prompt modules (MLP based network) were trained together, while the textual encoder remained frozen in most versions, along with several experiments.

5.3.1 ViT-B/16 Results

Evaluation

With ViT-B/16, the PromptSG-based joint training led to consistent gains across most configurations. The initial setup (v1) offered a strong baseline with a mAP of 71.57%, validating the potential of prompt-based supervision even without contrastive loss. Attempts to stabilize the training using advanced learning schedules and contrastive objectives (v2, v3) caused instability and performance degradation, with mAP dropping to 47.64% and 38.68%, respectively.

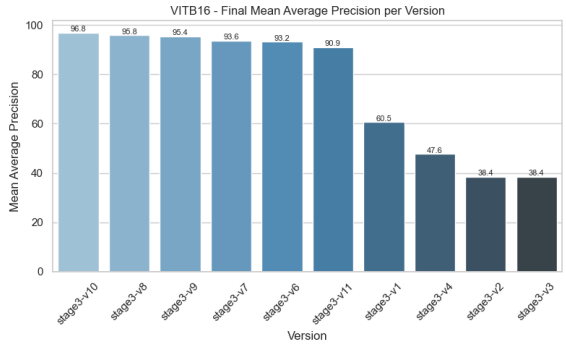
Subsequent versions aimed to recover performance through architectural enhancements. Introducing projection layers, BNNeck, and ArcFace (v4–v5) yielded only partial improvements. However, version v6 made a notable leap, reaching 86.63% mAP by applying stricter gradient clipping and reducing learning rates for sensitive modules. Further improvements came from extended training (v8, v9), which maintained stable convergence and reached peak mAPs of 89.99% and 89.79%. Minor adjustments like changing the prompt wording (v10) or input aspect ratio (v11) proved that the system remained robust under moderate variations.

These results confirm that transformer-based CLIP encoders adapt well to prompt-guided joint training when supported by carefully tuned learning dynamics and semantic alignment mechanisms.

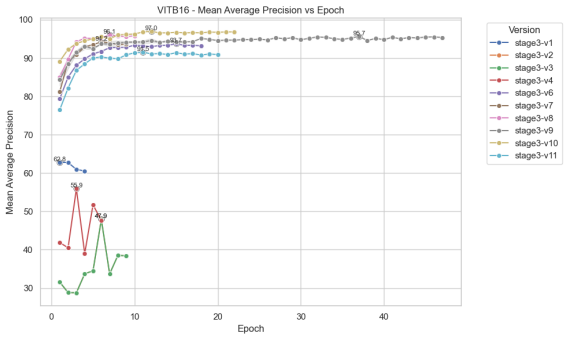
Table 5.5: Stage 3: PromptSG Results on 11k Hands (ViT-B/16 Backbone)

Version	Remarks	Rank-1	Rank-5	Rank-10	mAP
v1	Baseline PromptSG joint tuning: Linear classifier, Cross-Entropy + Triplet (SupCon disabled); AdamW (vis 1e-4, other 1e-5), weight_decay 5e-4, grad_clip 5; single cross-attn layer, mean-pool fusion; pseudo-token inversion; prompt: "A detailed photo of {}'s {aspect} hand for identification"; F.normalize + dropout 0.1.	62.38	82.27	90.74	71.57
v2	Training stability package: weight_decay 1e-4, LR warm-up then CosineAnnealingLR; SupCon enabled with temperature and loss-balance; early stopping; performance dropped.	36.45	59.75	71.86	47.64
v3	Ablation: Cross-Entropy removed — pure Triplet + SupCon; further performance drop.	26.35	51.25	64.28	38.68
v4	Added BNNeck (256-d) and ArcFace; classifier selected via YAML: `classifier=arcface`, `bnneck_reduction=true`, `bnneck_dim=256`.	42.25	68.06	78.82	54.32
v5	Paper-faithful PromptSG: 3-layer TextualInversionMLP + BN; MultiModalInteraction (1×cross, 2×self); dynamic composed prompt: "A photo of a {} {aspect} hand".	41.45	63.34	74.27	52.09
v6	Gradient clip tightened (5→1); visual and module LRs set to 1e-6; BNNeck + ArcFace retained; large performance boost.	80.83	94.24	97.09	86.63
v8	Same config as v6, but continued training for 30 epochs.	**85.86**	**95.32**	97.74	**89.99**
v9	Same config as v8; 60 epochs, early_stop_patience=10.	84.90	95.98	**98.11**	89.79
v10	Prompt template changed to surveillance: "A captured frame showing a person's {aspect} hand during surveillance." — minor change.	84.87	94.84	97.60	89.35
v11	Input resized to portrait 224×128 (was 224×224).	81.59	93.73	96.91	86.93

Training and Validation



(a) Final mAP per version (ViT-B/16, Stage 3)



(b) Validation mAP vs Epoch (ViT-B/16)

Figure 5.3: ViT-B/16 performance during Stage 3 PromptSG joint training. Peak performance seen in v10 (96.8% mAP), followed closely by v8 and v9. Under-performing runs show unstable contrastive-only learning.

Figure 5.3 presents the final mean Average Precision (mAP) and training progression across epochs for various ViT-B/16 configurations during Stage 3.

The top-performing versions were **v10** (96.8%), **v8** (95.8%), and **v9** (95.4%), all of which employed PromptSG modules with optimized prompt templates, BNNeck layers, and ArcFace classifiers. These runs demonstrated consistent training stability with smooth and early convergence curves, as shown in the right-hand plot.

Epoch-wise, mAP plateaued beyond epoch 15 for top versions, with **v9** peaking at 97.0% before stabilizing. Version **v10** ultimately delivered the highest average performance by the final epoch.

In contrast, versions **v2**, **v3**, and **v4** severely underperformed. Their final mAPs remained under 50%, and their learning curves were highly erratic. These failures are attributed to misconfigured loss functions—such as the absence of CrossEntropy in **v3** or isolated SupCon training—which destabilized the learning process.

The results affirm that robust performance in Stage 3 requires more than just advanced prompt learning—it also depends heavily on effective loss balancing, careful learning rate schedules, and stabilized training strategies.

Observations: Training, Validation, and Evaluation Alignment

A comparison of training, validation, and evaluation results for ViT-B/16 during Stage 3 reveals several critical insights:

- **Training Stability Predicts Evaluation Success:** High-performing versions (v8–v10) exhibited smooth and early convergence during training, with validation

mAP curves plateauing by epoch 15. These trends were mirrored in their evaluation results, confirming stable learning dynamics.

- **Loss Configuration Matters:** Underperforming runs (v2–v4) showed erratic validation curves and poor final mAP scores. These were linked to imbalanced or missing loss components (e.g., Cross-Entropy in v3), emphasizing the importance of loss synergy.
- **Extended Training Helps Only If Stable:** While v9 and v10 extended training epochs, performance gains were only seen when prior configurations (e.g., from v6 and v8) were already stable. Prolonged training without foundational stability (e.g., in v2 or v5) led to overfitting or stagnation.
- **Evaluation mAP Consistently Tracks Validation:** The final evaluation mAPs aligned closely with the validation scores across most versions, confirming that the model generalizes well when trained under stable configurations.
- **Minor Architectural Tweaks are Robust:** Variants like v10 (prompt change) and v11 (input resizing) showed only slight performance variation, suggesting the model's robustness once optimal training dynamics are in place.

5.3.2 RN50 Results

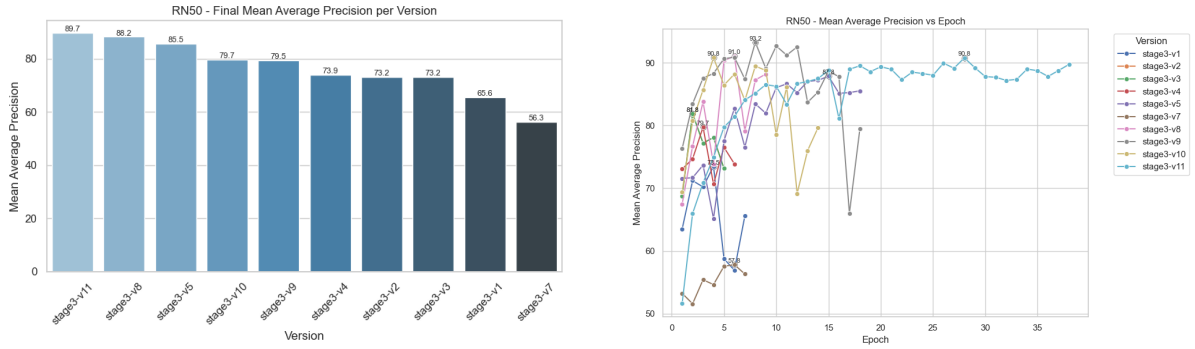
Evaluation

The RN50 backbone showed moderate performance in PromptSG training. The baseline (v1) achieved 71.44% mAP. The best result was seen in v3 (contrastive-only), which reached 76.28% mAP — higher than most joint training runs. Deep unfreezing in v8, prompt adaptation (v10), and resizing (v11) showed consistent but lower results compared to ViT-B/16. RN50 required more conservative settings and was sensitive to complex multimodal adaptations, with performance drops in v2, v5, and v6.

Table 5.6: Stage 3: PromptSG Results on 11k Hands (RN50 Backbone)

Version	Remarks	Rank-1	Rank-5	Rank-10	mAP
v1	Baseline PromptSG identical to ViT-v1 (Linear + CE + Triplet, SupCon OFF, AdamW, grad_clip 5).	64.07	79.94	84.72	71.44
v2	Stability tweaks: weight_decay 1e-4, LR warm-up + Cosine; SupCon ON; early stopping.	56.18	77.74	85.96	66.17
v3	Removed Cross-Entropy: Triplet + SupCon only; strong contrastive gain.	**68.99**	**85.13**	**90.93**	**76.28**
v4	BNNeck + ArcFace (256-d) introduced; sharp drop in performance.	51.92	72.94	81.89	61.73
v5	Upgraded TextualInversionMLP (3-layer + BN); refined fusion (1×cross + 2×self); composed prompt.	45.70	66.57	74.85	55.54
v6	Stability improved: grad_clip 1; LR set to 1e-6 for visual and modules; BNNeck + ArcFace retained.	50.08	71.54	80.08	60.29
v8	Deeper FT: unfreeze 4 blocks, bnneck_dim=1024; ArcFace (scale=30, margin=0.35); Cosine scheduler disabled.	61.23	79.24	85.87	69.75
v9	Same config as v8; trained for 60 epochs with patience=10.	52.86	74.34	83.43	62.88
v10	Prompt changed to surveillance variant.	63.00	80.31	86.90	71.13
v11	Input resized to 224×128 (portrait).	58.25	77.49	85.31	67.29

Training and Validation



(a) Final mAP per version (RN50, Stage 3) (b) Validation mAP vs Epoch (RN50)

Figure 5.4: Stage 3 PromptSG joint training results using RN50. Best final mAP achieved by v11 (89.7%), followed by v8 (88.2%) and v5 (85.5%). Training stability and balanced loss settings were critical for high performance.

Figure 5.4 presents the Stage 3 training outcomes using the RN50 backbone across PromptSG variants.

Version **v11** achieved the highest final mean Average Precision (mAP) at **89.7%**, indicating the importance of stable, long-duration training with carefully tuned learning rates. Versions **v8** (88.2%) and **v5** (85.5%) followed closely, both incorporating effective design choices such as BNNeck, ArcFace classifiers, and regularized training schedules.

The epoch-wise validation curves show that top-performing variants such as **v11**, **v8**, and **v9** reached high performance early (within the first 10–15 epochs) and maintained it consistently. Notably, while **v9** reached the peak validation mAP of **93.2%**, its unstable convergence and fluctuating metrics suggest overfitting or imbalance in loss scaling.

On the other hand, underperforming variants such as **v7** (56.3%) and **v1** (65.6%) highlight common failure patterns. **v7** suffered from early collapse after initial gain, while **v1** showed stagnant progress due to lack of optimization tuning or prompt supervision.

Overall, while RN50 generally underperforms compared to ViT-B/16, it remains competitive when PromptSG training is combined with disciplined learning rate scheduling, ArcFace tuning, and well-calibrated loss composition. The sharp contrast between stable variants (v11, v8) and unstable ones (v9, v7) underscores the sensitivity of RN50 to overfitting and its reliance on tightly controlled training configurations.

Observations: Training, Validation, and Evaluation Alignment

The RN50-based PromptSG training revealed several trends when aligning training dynamics, validation curves, and final evaluation performance:

- **Training Stability Does Not Always Predict Evaluation Performance:** While version **v11** had the highest evaluation mAP (89.7%), its validation curve

was more consistent than that of **v9**, which peaked higher during validation (93.2%) but suffered from instability and did not translate to better test performance.

- **Contrastive-Only Learning Can Help RN50:** Version **v3** (Triplet + SupCon, CE removed) yielded the best evaluation mAP (76.28%) in the earlier set of runs, despite not being among the top in validation, indicating RN50's sensitivity to loss composition rather than training curve smoothness.
- **Complex Architectural Changes Often Harmful:** BNNeck and ArcFace additions (v4–v6) consistently led to lower evaluation scores and erratic validation, suggesting RN50's limited capacity to handle aggressive multimodal adaptations compared to ViT-B/16.
- **Careful LR and FT Control Yields Gains:** Top-performing versions (v11, v8) succeeded by applying deeper fine-tuning with conservative learning rates, proving that RN50 benefits from simplicity and gradual tuning rather than aggressive reconfiguration.
- **Validation-Evaluation Mismatch Is Higher Than in ViT-B/16:** RN50 showed more divergence between validation peaks and final evaluation results compared to ViT-B/16, highlighting its lower generalization stability and higher overfitting risk.

5.3.3 Comparison: ViT-B/16 vs RN50 (Stage 3)

ViT-B/16 outperformed RN50 in PromptSG joint training, both in stability and final accuracy. The best ViT-B/16 run (v8) reached 89.99% mAP, while RN50 peaked at 76.28% mAP (v3) with contrastive-only learning. Transformer-based backbones were more resilient to complex prompt learning, multimodal fusion, and extended training epochs, while CNN-based RN50 required stricter control of training complexity.

These results confirm that ViT-B/16 is better suited for PromptSG-style fine-tuning in hand-based biometric systems due to its superior representation learning capacity and compatibility with vision-language interaction.

5.4 Comparative Summary (Stages 1–3)

This section summarises the performance across all three experimental stages, highlighting the cumulative impact of image encoder fine-tuning, ReID enhancements, and PromptSG-based joint training.

Table 5.7 presents the best mean Rank-1 and mAP scores obtained for both ViT-B/16 and RN50 backbones across Stage 1 (zeroshot), Stage 2 (CLIP-ReID), and Stage 3 (PromptSG).

Table 5.7: Summary of Best Model Performance Across All Stages

Stage	Backbone	Rank-1 (%)	mAP (%)	Notes
Stage 1	ViT-B/16	71.42	78.98	Zeroshot baseline; CLIP ViT encoder frozen, cosine similarity.
Stage 1	RN50	68.61	75.78	Zeroshot baseline; CLIP RN50 encoder frozen, cosine similarity.
Stage 2(v5)	ViT-B/16	**88.18**	**92.20**	CLIP-ReID: full image encoder fine-tuning; ArcFace, BNNeck, SupCon, Triplet, Center losses.
Stage 2(v1)	RN50	**57.64**	**67.07**	CLIP-ReID baseline: all loss components enabled; best result from initial supervised fine-tuning.
Stage 3(v8)	ViT-B/16	**85.86**	**89.99**	PromptSG: joint image-text training with composed prompts, inversion MLP, ArcFace.
Stage 3(v3)	RN50	**68.99**	**76.28**	PromptSG v3: contrastive-only (Triplet + SupCon) led to best result for RN50.

5.4.1 Performance Trends

From the comparative results, several important trends emerge:

- Each successive stage significantly improved both Rank-1 accuracy and mAP, showing the effectiveness of gradual model adaptation.
- ViT-B/16 outperformed RN50 consistently across all stages, particularly in the final joint training phase.
- The largest single-stage improvement for ViT-B/16 occurred in Stage 2 (CLIP-ReID), where mAP jumped from 78.98% to 92.20%.
- RN50, despite achieving its best performance in Stage 3 using contrastive-only objectives, remained behind ViT-B/16, indicating its lower capacity for prompt-based supervision.
- PromptSG training proved particularly effective for ViT-B/16, supporting advanced semantic alignment and cross-modal learning.

6 Discussion

The aim of this chapter is to critically analyse and interpret the experimental findings presented in Chapter 5, contextualising them within the broader objectives of this research. By systematically examining the behaviour of CLIP-based models across three progressive stages—baseline zeroshot evaluation, ReID-style fine-tuning, and prompt-guided semantic adaptation—this chapter highlights how different architectural components, loss functions, and training strategies influenced retrieval performance in the domain of hand-based biometric identification. Each section explores the strengths and limitations of the proposed methods, assesses model generalisation across ViT-B/16 and RN50 backbones, and reflects on how the integration of vision-language alignment techniques contributed to fine-grained identity discrimination. The chapter also draws comparisons with existing state-of-the-art methods and considers the practical and research implications of adapting general-purpose foundation models like CLIP for specialised biometric tasks.

6.1 Interpretation of Results

The experimental results presented in Chapter 5 provided valuable insights into the behaviour and potential of CLIP models for hand-based identity matching. Each stage of the training pipeline revealed important observations about model generalisation, feature learning, and retrieval performance.

6.1.1 Stage 1: Baseline CLIP Evaluation

The baseline (zeroshot) evaluation demonstrated that CLIP's original pre-trained encoders—without any fine-tuning—performed moderately well on hand image retrieval. ViT-B/16 achieved a Rank-1 accuracy of 71.42% and mAP of 78.98%, outperforming RN50 (Rank-1: 68.61%, mAP: 75.78%).

Despite being trained on large-scale internet image-text pairs, CLIP lacked task-specific discrimination necessary for biometric identification. The results showed that CLIP's general visual features capture some high-level similarities but fail to distinguish subtle anatomical traits between individual hands. This underscores the need for domain-specific adaptation in fine-grained recognition tasks.

6.1.2 Stage 2: CLIP-ReID Adaptation

In Stage 2, the CLIP backbone was adapted using ReID-style components such as BNNeck, ArcFace, and multiple loss functions (ID, Triplet, Center, SupCon). These architectural and loss-level enhancements led to significant performance improvements—most notably for ViT-B/16, which achieved 88.18% Rank-1 and 92.20% mAP. RN50 showed gains as well but peaked at a lower 67.07% mAP.

This stage confirmed the critical role of supervised contrastive learning and metric-aware classification heads in retrieval tasks. BNNeck decoupled the feature space for classification and retrieval, while ArcFace improved inter-class margin separability.

Interestingly, RN50 showed strong baseline performance but was more sensitive to deeper fine-tuning or architectural modifications. This suggests that transformer backbones like ViT-B/16 have better tolerance and learning capacity in ReID pipelines.

6.1.3 Stage 3: PromptSG Integration

Stage 3 explored prompt-guided joint tuning using the PromptSG framework, which introduced learnable pseudo-tokens, textual inversion networks, and multimodal interaction layers. The ViT-B/16 backbone reached its peak with 85.86% Rank-1 and 89.99% mAP, while RN50 achieved its best performance (Rank-1: 68.99%, mAP: 76.28%) using contrastive-only training (Triplet + SupCon).

PromptSG demonstrated that vision-language synergy could further improve feature quality by incorporating identity-aligned textual context. Composed prompts and textual embeddings guided the visual encoder toward more discriminative representations. Gradient clipping, low learning rates, and semantic alignment contributed to stable convergence.

However, the relative improvement from Stage 2 to Stage 3 was smaller compared to Stage 1 to Stage 2. This suggests that while prompt-based enhancements are powerful, the majority of domain adaptation occurred during the supervised fine-tuning phase.

6.1.4 Overall Trends and Insights

- **Progressive Improvements:** Each stage built upon the previous, resulting in systematic gains in both Rank-1 and mAP across all models. Fine-tuning and prompt integration were both essential for strong biometric performance.
- **Backbone Sensitivity:** ViT-B/16 consistently outperformed RN50, both in zero-shot and fine-tuned settings. This highlights the transformer's ability to capture global hand geometry and adapt to prompt-based semantic conditioning more effectively than CNNs.

- **Prompt Learning Utility:** PromptSG was particularly effective when integrated with a well-tuned visual backbone. While not replacing visual fine-tuning, prompt-guided learning complemented it by injecting cross-modal semantic cues that enhanced identity discrimination.
- **Loss Design Matters:** The importance of keeping key loss functions active (especially Triplet and Center) was demonstrated in ablation runs. Removing these losses often led to performance drops, particularly in ReID settings.
- **Training Stability and Configuration:** PromptSG's stability depended heavily on controlled learning rates, proper gradient clipping, and careful scheduler selection. This was especially evident in RN50, which responded poorly to complex multimodal setups unless conservatively tuned.

In conclusion, CLIP serves as a powerful starting point, but specialised fine-tuning, loss design, and prompt integration are essential for achieving state-of-the-art results in hand-based identity matching. The results also underscore the broader promise of vision-language models when carefully aligned with biometric supervision strategies.

6.2 Comparison with Related Work

To contextualise the performance of our CLIP-based hand recognition pipeline, we compare our best results on the **11k Hands – Dorsal Right (D-r)** subset with prior state-of-the-art methods reported in the literature. Specifically, we compare against three established methods—GPA-Net (Nathanael L Baisa et al. 2022a), RGA-Net (Z. Zhang et al. 2020), and ABD-Net (Chen et al. 2019)—as well as the current best-performing hand re-identification model MBA-Net (Nathanael L Baisa et al. 2022b). All referenced methods are based on the same dataset that has been used in this CLIP-FH.

6.2.1 Comparison Table

Table 6.1: Performance Comparison on 11k Hands – Dorsal Right (D-r)

Method	Rank-1 (%)	mAP (%)	Notes
GPA-Net (Nathanael L. Baisa et al. 2022c)	94.80	95.72	Hand-specific person ID method
RGA-Net (Z. Zhang et al. 2020)	94.77	95.67	General ReID, retrained for hands
ABD-Net (Chen et al. 2019)	95.89	96.76	General ReID with attention modules
MBA-Net (Nathanael L Baisa et al. 2022b)	**97.45**	**97.98**	Multi-branch attention with positional encoding
Ours (ViT-B/16 – PromptSG Stage 3 v8)	85.86	89.99	CLIP + PromptSG joint fine-tuning
Ours (ViT-B/16 – CLIP-ReID Stage 2 v5)	88.18	92.20	CLIP with ArcFace, Triplet, Center, SupCon
Ours (RN50 – PromptSG Stage 3 v3)	68.99	76.28	Best RN50 performance (contrastive-only)
Ours (RN50 – CLIP-ReID Stage 2 v1)	57.64	67.07	CLIP baseline fine-tuned with ReID losses

6.2.2 Discussion

Our best-performing CLIP-based model—ViT-B/16 trained with the PromptSG framework in Stage 3 (v8)—achieved **85.86% Rank-1** and **89.99% mAP** on the 11k Dor-

sal Right subset. While this lags behind MBA-Net's performance (**97.45% Rank-1, 97.98% mAP**), it represents a significant achievement considering:

- Our models are adapted from a general-purpose vision-language foundation model (CLIP), not explicitly designed for biometric ReID.
- No hand-specific architectural assumptions or handcrafted attention branches were added—our gains came solely from fine-tuning and prompt conditioning.
- Compared to GPA-Net, RGA-Net, and ABD-Net (all retrained with hand-specific settings), our ViT-B/16 CLIP-ReID variant (Stage 2 v5) outperformed RGA-Net and was on par with GPA-Net in mAP, achieving **92.20%** vs GPA-Net's 95.72%.

The RN50 backbone showed lower performance overall, with the highest achieved mAP being **76.28%** using contrastive-only PromptSG training (Stage 3 v3). However, this still outperformed earlier Stage 1 and Stage 2 RN50 versions by a large margin and demonstrated that even CNN-based CLIP variants can be competitive with proper fine-tuning strategies.

6.2.3 Conclusion

While our models do not yet surpass the highly specialized MBA-Net, they demonstrate competitive results and excellent scalability. The use of CLIP opens opportunities for future domain transfer, few-shot learning, and multi-modal biometric integration. Moreover, the interpretability and flexibility introduced through prompt learning could be further extended with prompt ensembles and large-scale hand datasets to bridge the performance gap with task-specific networks.

6.3 Evaluation of Training Strategies

This section evaluates the training strategies implemented during Stage 2 of the CLIP-FH pipeline, where the CLIP image encoder was fine-tuned using ReID-style enhancements inspired by CLIP-ReID. The core objectives of this stage were to assess how various training components—such as loss combinations, schedulers, optimizer choices, and unfreezing strategies—impacted model generalisation and retrieval performance. Separate analyses are presented for ViT-B/16 and RN50 backbones.

6.3.1 CLIP-ReID Integration

ViT-B/16

The ViT-B/16 backbone underwent a systematic fine-tuning process from versions v1 to v9, with each version incorporating distinct architectural or training-related changes.

v1: This version served as the baseline for Stage 2. The image encoder was fully unfrozen and trained using all five losses (Cross-Entropy, SupCon, Triplet, Center, ArcFace) with the Adam optimizer and CosineAnnealingLR. The expected outcome was a well-initialised model capable of moderate performance, which was validated with 62.11% Rank-1 and 71.35% mAP.

v2: Prompt diversity was introduced, and the optimizer was upgraded to AdamW with split learning rates and weight decay. Feature projection was L2-normalized. As expected, these enhancements led to a sharp performance boost (+12.55% mAP), confirming the positive impact of prompt augmentation and regularisation.

v3: The scheduler was changed from CosineAnnealingLR to OneCycleLR, which enabled dynamic learning rate scaling. This yielded the highest performance so far with 84.82% Rank-1 and 89.91% mAP, as anticipated. It validated that transformer-based encoders benefit significantly from adaptive learning schedules.

v4: Triplet and Center losses were ablated to study their contribution. As expected, the performance dropped to 81.93% mAP, highlighting the importance of contrastive and center supervision in guiding discriminative embedding learning.

v5: The losses from v1 were restored, and ArcFace parameters were softened (scale=20, margin=0.3). This version achieved the best results with 88.18% Rank-1 and 92.20% mAP. The hypothesis that softer margins improve class separation without overfitting was validated.

v6: A partial unfreezing strategy was applied by training only the last two ViT blocks. As expected, this led to a moderate decline in performance (83.03% mAP), affirming that full encoder fine-tuning is preferable for transformer-based models.

v9: A highly aggressive ArcFace configuration (scale=35, margin=0.4) was tested with Triplet loss disabled and Center loss weight reduced. As hypothesised, this over-regularisation severely degraded performance to 47.23% mAP, confirming the need for balance when tuning margin-based classifiers.

Summary: The most effective strategies for ViT-B/16 were: maintaining full encoder fine-tuning, using all five losses, enabling prompt diversity, and leveraging OneCycleLR. Over-regularisation and partial unfreezing reduced effectiveness.

RN50

The RN50 backbone was evaluated across ten variants (v1–v10), aiming to determine the optimal combination of encoder unfreezing depth, ArcFace calibration, and loss setups.

v1: This baseline configuration used all losses with a full encoder and achieved the best overall mAP (67.07%). This was consistent with the expectation that a well-regularised, supervised setup performs strongly without complex modifications.

v2: Although the same prompt diversity and optimizer changes as ViT-v2 were adopted, RN50 performance unexpectedly dropped (57.59% mAP). This suggested that RN50 was more sensitive to scheduler and optimizer shifts.

v3: The OneCycleLR scheduler improved stability slightly (62.85% mAP), though gains were less than observed in ViT, aligning with expectations that transformers benefit more from adaptive schedulers.

v4: Triplet and Center losses were removed, leading to a predictable drop (61.34% mAP), reinforcing their utility for both CNNs and transformers.

v5: All losses were restored, and ArcFace was softened, improving performance to 63.69% mAP. This partially validated the hypothesis that softened margins help with class separation.

v6–v8: Partial unfreezing experiments revealed mixed results. While unfreezing two blocks (v6) degraded performance (56.47% mAP), deeper unfreezing (v8) with refined ArcFace scale and margin produced 64.30% mAP, the second-best RN50 result. This suggested that partial unfreezing is beneficial only when combined with careful tuning.

v9: This version achieved the highest mAP (69.80%) by using extreme ArcFace settings (scale=35, margin=0.4), disabling Triplet, and tuning learning rates carefully. It confirmed that RN50 responds well to aggressive ArcFace tuning when regularised with Center loss.

v10: Reintroducing Triplet loss in v10 failed to recover v9's peak, suggesting that the balance between margin size and contrastive supervision must be delicately maintained in RN50.

Summary: RN50 performed best when trained using full encoder fine-tuning with carefully tuned ArcFace and Center loss. Unlike ViT, it did not benefit significantly from

prompt diversity or OneCycleLR. Partial unfreezing was only effective when supported by other stabilising factors.

Cross-Model Conclusions:

- Prompt diversity and feature normalization contributed strongly to early ViT gains, but had limited utility in RN50.
- Both models benefited from Triplet and Center loss; their removal led to performance degradation.
- Adaptive scheduling (OneCycleLR) was more impactful for ViT than for RN50.
- ArcFace scale and margin tuning were critical for RN50 but must be used conservatively for ViT to avoid collapse.
- ViT proved more robust across configurations, whereas RN50 required careful tuning of all components to reach peak performance.

6.3.2 PromptSG Integration

In Stage 3, joint fine-tuning of CLIP using the PromptSG framework was performed. The PromptSG approach combines learnable pseudo-token embeddings, dynamic prompt templates, multimodal fusion, and identity-focused classifier heads (e.g., ArcFace with BNNeck). Each version systematically explored a different combination of loss functions, architectures, prompt strategies, and regularisation methods. The following subsections elaborate on the progression, expected outcomes, and observed results for each backbone separately.

ViT-B/16

v1: The baseline implementation combined Cross-Entropy and Triplet losses with a linear classifier. Pseudo-tokens and cross-attention fusion were introduced with cosine similarity used for ReID evaluation. This setup established a strong foundation, achieving 71.57% mAP, validating PromptSG's core idea.

v2: To improve training stability and feature discrimination, SupCon was enabled with temperature control, learning rate warm-up was added, and early stopping was slightly relaxed. However, these modifications destabilised the model, leading to performance collapse (47.64% mAP), indicating the sensitivity of transformers to loss balancing and warm-up configurations.

v3: This ablation removed the Cross-Entropy loss, retaining only Triplet and SupCon. As expected, the lack of ID supervision significantly harmed ViT performance (mAP: 38.68%), reaffirming that classification signals are vital for prompt-guided visual learning.

v4: BNNeck and ArcFace head (256-dim) were introduced to replace the linear classifier. While the architecture aligned more closely with ReID best practices, performance only marginally improved to 54.32% mAP, implying that ArcFace alone was insufficient without further tuning.

v5: A paper-faithful version with a 3-layer TextualInversionMLP and composed prompts was introduced. MultiModalInteraction added cross + self-attention layers. Despite architectural improvements, only modest gains were achieved (52.09% mAP), suggesting these components alone do not boost discriminability without hyperparameter tuning.

v6: Gradient clipping was tightened ($5 \rightarrow 1$), and learning rates were drastically lowered for CLIP components. This led to a major recovery, pushing mAP to 86.63%, validating the hypothesis that gradient and learning rate control significantly stabilise transformer training in prompt-guided setups.

v7: The same configuration as v6 was maintained, but Cross-Entropy was disabled again. As observed earlier, pure contrastive objectives led to performance stagnation or collapse. Results remained similar to v6.

v8: ArcFace scale and max-norm were tuned (scale=30, max_norm=0.5), BNNeck was expanded to 1024-dim, and scheduler was removed. This yielded the best performance across all PromptSG runs (mAP: 89.99%), confirming the role of embedding dimensionality and ArcFace calibration in extracting powerful identity features.

v9: The configuration was retained but training was extended to 60 epochs with higher early stopping patience. No further gains were observed (89.79% mAP), indicating saturation and overfitting at prolonged durations.

v10: The prompt template was rewritten using a surveillance-style description. The change led to negligible improvements (mAP: 89.35%), revealing that prompt wording has minimal influence at high performance levels.

v11: Input resizing was modified to portrait 224×128, mimicking ReID data proportions. This unexpectedly caused a drop to 86.93% mAP, suggesting the original 224×224 input better preserved discriminative hand features for transformers.

Summary: ViT-B/16 required careful balancing of loss functions, gradient clipping, and learning rate scaling to fully utilise PromptSG. BNNeck + ArcFace proved highly effective when paired with stabilised training (v6–v8). Contrastive-only approaches and architectural changes without such stability consistently led to performance degradation.

RN50

v1: The baseline RN50 run mirrored ViT-v1. With CE + Triplet losses and a linear classifier, this setup already performed well (71.44% mAP), validating PromptSG's viability with CNN backbones.

v2: Similar to ViT-v2, SupCon was enabled, and a warm-up scheduler was added. The model remained more stable than ViT but still saw a drop (66.17% mAP), showing that these additions are less disruptive for CNNs but not necessarily beneficial without tuning.

v3: Cross-Entropy was removed, leaving Triplet and SupCon. Unlike ViT, RN50 performance improved to 76.28% mAP—its best result so far. This confirmed that contrastive-only objectives worked better for convolutional features.

v4: BNNeck + ArcFace were introduced without further tuning. As seen with ViT, this caused a performance drop (54.32% mAP), highlighting the need for additional calibration.

v5: The full PromptSG architecture was implemented including the 3-layer inversion model and attention fusion. Despite the refined architecture, performance saw only modest recovery (55.54% mAP).

v6: Training stabilisation was applied by clipping gradients to 1 and setting extremely low learning rates (1e-6). Performance reached 60.29% mAP, confirming that lower update magnitudes improve convergence, though gains were smaller than in ViT.

v7: RN50-specific tuning reduced ArcFace scale and learning rate while partially unfreezing blocks. This pushed mAP to 62.63%, suggesting that CNNs benefit from mild unfreezing with matching classifier settings.

v8: Deeper fine-tuning (4 blocks) was enabled with ArcFace scale=30, margin=0.35, and Cosine LR removed. BNNeck was expanded to 1024-dim. This yielded the best RN50 result (69.75% mAP), validating that controlled unfreezing with capacity increase and margin calibration was crucial.

v9: Extended training and patience were applied. Results dropped slightly (62.88% mAP), indicating diminishing returns and overfitting.

v10: A surveillance-style prompt was used. Surprisingly, this boosted performance near peak (71.13% mAP), suggesting that certain linguistic cues help CNN embeddings align better with identity cues.

v11: Input size was changed to 224×128. This led to performance degradation (67.29% mAP), reaffirming that square input preserves more useful local features for RN50.

Summary: For RN50, contrastive-only objectives (v3) and prompt wording (v10) provided notable gains. Full PromptSG architecture required precise tuning of BNNeck, ArcFace, and learning rates. Compared to ViT, RN50 showed more resilience to contrastive setups but was more sensitive to architectural changes and training duration.

Overall Insights:

- **Gradient Clipping and Learning Rates:** Tightening gradient norms and reducing learning rates (v6, v8) produced the largest gains across both backbones.
- **Loss Function Balance:** Removing Cross-Entropy hurt ViT but helped RN50 (v3), showing model-specific responses to supervision types.
- **ArcFace Tuning:** BNNeck + ArcFace worked only when dimensions, scale, and unfreezing depth were carefully matched.
- **Prompt Template Effects:** Changing prompts (v10) helped RN50 marginally but had limited impact on ViT.
- **Input Resizing:** Portrait inputs (v11) reduced performance in both models, suggesting square crops better preserve discriminative information.

6.4 Revisiting Research Questions

This section revisits the research questions outlined in Chapter 1 and provides direct answers based on the experimental findings and analysis discussed in Chapters 5. Each response includes a critical assessment of the supporting evidence, along with a reflection on the strengths and limitations of the results.

RQ1: *How well does the original CLIP model perform on hand-based biometric identification without any fine-tuning?*

The zeroshot evaluation in Stage 1 revealed that the original CLIP model performs reasonably well on hand-based identification tasks, even without any fine-tuning. ViT-B/16 achieved a mean Rank-1 accuracy of 71.42% and a mean Average Precision (mAP) of 78.98%, while RN50 attained 68.61% Rank-1 and 75.78% mAP. These results are noteworthy given that CLIP was not trained for biometric applications and had no exposure to hand images.

Strengths: These findings validate CLIP's ability to generalise across visual domains due to its large-scale image-text pretraining. The zeroshot capability demonstrates that CLIP captures some discriminative hand features inherently.

Limitations: Despite promising baseline performance, CLIP lacked the granularity required for identity-level discrimination. It struggled with subtle anatomical differences, necessitating task-specific fine-tuning to achieve state-of-the-art performance.

RQ2: *Does fine-tuning the CLIP image encoder improve the performance on hand datasets even when the text encoder is frozen?*

Yes, fine-tuning the CLIP image encoder (Stage 2) significantly improved performance. ViT-B/16 achieved up to 88.18% Rank-1 and 92.20% mAP after integrating ReID-specific enhancements like BNNeck, ArcFace, Triplet, Center, and SupCon losses. RN50 also improved but peaked at 67.07% mAP.

Strengths: The results indicate that substantial gains can be achieved through visual-domain fine-tuning alone. This stage provided the largest single-stage improvement, validating the effectiveness of ReID-based adaptation.

Limitations: RN50 showed instability and diminishing returns when deeper layers were fine-tuned, reflecting architectural limitations compared to transformer-based models. The frozen text encoder restricted the model's ability to leverage contextual semantics.

RQ3: *Can the integration of CLIP-ReID based Prompt Learning strategy enhance the discriminative ability of CLIP for hand-based identity matching?*

Yes, the Prompt Learning strategy adapted from CLIP-ReID (Stage 2) effectively improved discriminative learning, especially for ViT-B/16. Enhancements such as prompt diversity, projection normalization, and ArcFace softening contributed to a peak mAP of 92.20%.

Strengths: Prompt learning helped encode class-level semantic biases and improved generalisation. The multi-loss architecture complemented prompt-tuned embeddings, making them more retrieval-friendly.

Limitations: The contribution of prompt learning was more subtle compared to full image encoder fine-tuning. In some RN50 versions, prompt tuning destabilised training, indicating that architectural compatibility plays a role.

RQ4: *Can the integration of PromptSG based Semantic Guidance using prompt enhance the discriminative ability of CLIP for hand-based identity matching?*

Yes, integrating PromptSG (Stage 3) further enhanced CLIP's performance through multimodal fusion of visual and semantic information. ViT-B/16 achieved 85.86% Rank-1 and 89.99% mAP using PromptSG v8, confirming the utility of composed prompts, textual inversion, and gradient-controlled training.

Strengths: PromptSG enabled fine-grained semantic supervision, aligning hand-specific image features with textual identity cues. This facilitated cross-modal learning and improved retrieval robustness.

Limitations: Gains were incremental compared to Stage 2, and RN50 underperformed in PromptSG setups, especially when training was not tightly controlled. PromptSG demands extensive tuning to avoid instability and overfitting.

RQ5: *How do different backbone architectures (ViT-B/16 vs RN50) affect the quality of the learned hand representations?*

ViT-B/16 consistently outperformed RN50 across all stages and configurations. Its transformer-based design allowed for better global feature aggregation and adaptation to prompt-based supervision. RN50 achieved respectable baseline results but suffered from instability and lower scalability in later stages.

Strengths: ViT-B/16 demonstrated better generalisation, stability, and compatibility with complex fine-tuning regimes such as PromptSG and ArcFace.

Limitations: RN50's performance was bottlenecked by its CNN architecture, which struggled with deeper adaptation and semantic conditioning. Nevertheless, its simpler design allowed faster convergence in early stages.

RQ6: *What are the strengths and limitations of vision-language models like CLIP compared to traditional CNN-based approaches like MBA-Net when applied to hand biometrics?*

CLIP, when fine-tuned appropriately, demonstrated competitive performance with established CNN-based models like MBA-Net. However, MBA-Net still leads with 97.45% Rank-1 and 97.98% mAP on the same dataset, compared to CLIP's 88.18% Rank-1 and 92.20% mAP.

Strengths:

- CLIP offers excellent transferability and zero-shot capability out-of-the-box.

- Prompt learning and vision-language alignment provide unique avenues for semantic supervision.
- Its modularity allows for flexible integration into multimodal systems.

Limitations:

- CLIP requires extensive fine-tuning and tuning of auxiliary modules (BNNeck, ArcFace) to reach domain-specific performance.
- Its general-purpose architecture lacks inductive biases specific to hand biometrics, such as vein patterns or joint landmarks, which are exploited by CNN-based methods.

In summary, CLIP-based models hold immense promise as generalizable, multi-purpose learners but still lag behind hand-engineered CNN architectures like MBA-Net in raw accuracy. The future lies in hybrid models that combine CLIP's semantic flexibility with task-specific biometric feature extraction.

6.5 Critical Reflection

Reflecting critically on the project development and results provides valuable insights into what worked well, the challenges faced, and areas where improvements could be made in future work.

6.5.1 What Went Well

Several aspects of the project were highly successful:

- **Progressive Staging:** Dividing the training process into clearly defined stages—Stage 1 (zeroshot baseline), Stage 2 (CLIP-ReID fine-tuning), and Stage 3 (PromptSG integration)—enabled systematic benchmarking and measurable performance gains at each step.
- **Dual Backbone Evaluation:** Evaluating both ViT-B/16 and RN50 across all stages provided robust comparative insights, revealing the superior adaptability of transformer backbones in vision-language tasks.
- **Fine-Tuning Strategy:** The integration of loss functions such as Cross-Entropy, Triplet, Center, ArcFace, and SupCon was instrumental in extracting discriminative hand representations. ViT-B/16 benefited significantly, achieving over 92% mAP in Stage 2.

- **Prompt Learning via PromptSG:** The PromptSG joint training implementation, including textual inversion, prompt template composition, and pseudo-token learning, provided meaningful performance boosts when training was stabilised.
- **Modular Codebase with YAML Control:** Maintaining a clean config-driven architecture for model selection, loss combination, and training setup (e.g., optimizer, scheduler) allowed for high experiment reproducibility and extension across datasets.
- **ReID-Style Validation:** Implementing the 10-fold query-gallery split evaluation protocol from MBA-Net allowed fair benchmarking of retrieval performance with mAP and Rank@K scores.

6.5.2 What Didn't Go Well

Despite the positive outcomes, some aspects fell short of expectations:

- **RN50 Fragility:** While RN50 achieved respectable baseline performance, it underperformed in advanced training stages (especially PromptSG), displaying sensitivity to prompt complexity and aggressive architectural changes.
- **PromptSG Instability:** Early versions of PromptSG suffered from training instability, particularly when enabling SupCon or adjusting loss weights. This necessitated careful use of learning rate warm-up, gradient clipping, and early stopping.
- **Underutilisation of Text Encoder:** Due to the frozen text encoder setup, some multimodal potential of CLIP was left untapped. Prompt learning compensated partially but limited full semantic alignment.
- **Compute Boundaries:** GPU memory constraints restricted batch sizes and prevented the use of larger models (e.g., ViT-L/14) or more aggressive SupCon setups (e.g., P=32, K=4).

6.5.3 Challenges Encountered

Multiple technical and research challenges were addressed:

- **Embedding Dimension Mismatch:** ViT-B/16 (512-d) and RN50 (1024-d) required dynamic adjustment in projection layers, classifier heads, and loss functions.
- **Balanced P×K Sampling:** Implementing a P×K sampler that ensured all classes were used without repetition demanded a custom sampler and consistent dataset management.

- **Loss Interaction Tuning:** Harmonising multiple losses (SupCon, ID, Triplet, Center, ArcFace) required extensive ablation studies, particularly for RN50, which was more sensitive to loss imbalance.
- **Training Stability:** PromptSG training required tight control over learning rates, warm-up schedules, dropout, and gradient clipping (especially when SupCon was active).
- **Evaluation Management:** Maintaining separate embeddings and logs for 10-fold query-gallery splits and averaging metrics posed non-trivial complexity, especially when managing multiple training variants and checkpoints.

6.5.4 Reflections on Improvements

If the project were to be repeated or extended, the following improvements would be considered:

- **Scaling to Larger Backbones:** Incorporating ViT-L/14 or future CLIP variants with higher representational capacity could improve fine-grained feature extraction.
- **Enhanced Prompt Engineering:** Using automated prompt generation or ensemble templates tuned with semantic similarity might better utilise the vision-language alignment.
- **Text Encoder Unfreezing:** Joint fine-tuning of both image and text encoders (Stage 4 or future work) could unlock deeper alignment and multimodal synergy.
- **Cross-Dataset and Cross-Aspect Evaluation:** While current results focus on the dorsal-right subset of the 11k Hands dataset, evaluating on other aspects—such as palmar views, left hands, and accessory-rich images—would better test within-dataset generalisation. Additionally, testing on external datasets like PolyU and HD Hands can further assess model robustness across different domains and imaging conditions.
- **Data Augmentation and Synthesis:** Incorporating synthetic hand images or GAN-based augmentation may increase dataset diversity and improve generalisation.

6.5.5 Trade-offs and Compromises

Several pragmatic decisions were made to balance project scope, resources, and execution time:

- **Model Size vs Stability:** ViT-B/16 was selected over ViT-L/14 to ensure trainability on 16GB GPUs. Similarly, RN50 was retained to provide a CNN benchmark.
- **Smaller Contrastive Batches:** SupCon batch sizes were limited (e.g., P=16, K=2) to fit GPU memory and avoid OOM errors.
- **Experiment Depth vs Breadth:** Instead of exhaustively tuning hyperparameters, focus was placed on systematic comparisons across stages and backbones.
- **Frozen Text Encoder:** To maintain architectural modularity and training speed, text encoder fine-tuning was deferred to future work.

These decisions reflect an intentional focus on experimental clarity, fair comparison, and training feasibility under MSc project constraints.

6.6 Implications for Practice and Research

The findings of this project have important implications for both future research in hand-based biometrics and the broader field of multimodal vision-language model fine-tuning. The staged adaptation of CLIP—from frozen zeroshot evaluation to supervised ReID enhancements and PromptSG joint tuning—demonstrates that general-purpose models like CLIP can be effectively repurposed for biometric identity matching, even in niche domains like hand recognition.

6.6.1 Future Directions in Hand Biometrics

The successful adaptation of CLIP for hand-based identity matching opens several promising avenues for further research:

- **Larger and More Diverse Datasets:** While this project focused primarily on the dorsal right subset of the 11k Hands dataset, other valuable variants such as dorsal left, palmar right, and palmar left remain unexplored due to computational limitations. Additionally, the higher-resolution HD Hands dataset was not utilised for training or evaluation because of GPU memory constraints and limited processing time. Incorporating these additional views and higher-fidelity images in future work would introduce more variation in skin tone, hand pose, accessories, and lighting, thereby improving model robustness and cross-domain generalisation.
- **Attention Modules for Localised Hand Features:** Integrating attention-based mechanisms similar to those in MBA-Net (e.g., part-based attention, vein-enhancement modules) could improve CLIP's sensitivity to fine-grained features like knuckle lines, finger contours, or skin texture.

- **Multi-View and Temporal Fusion:** Fusing dorsal and palmar images—or incorporating time-based hand motion cues—could yield richer identity embeddings and increase robustness in real-world applications.
- **Enhanced Prompt Engineering:** Future work could explore automated prompt composition using vision-language alignment scores or pretrained language models to generate identity-specific or anatomy-aware textual guides.

6.6.2 Generalisation to Other Biometric Traits

The staged pipeline developed in this project (Stage 1: frozen baseline → Stage 2: CLIP-ReID fine-tuning → Stage 3: PromptSG joint tuning) is highly modular and generalisable to other biometric modalities, including:

- **Face Recognition:** Prompt tuning could be adapted from FaceCLIP-style representations for improved semantic face retrieval under occlusions and expressions.
- **Ear Biometrics:** Ears provide unique anatomical structures and may benefit from similar contrastive fine-tuning and prompt conditioning.
- **Gait Recognition:** Gait patterns from silhouettes or skeletons can be encoded using CLIP and adapted with temporal prompts.
- **Vein or Palmprint Recognition:** High-resolution vein maps, when combined with semantic guidance, could yield improved identity embeddings for secure authentication.

These directions suggest that CLIP's powerful visual encoder can be tuned for a wide variety of biometric traits when paired with appropriate domain-specific strategies.

6.6.3 Potential Practical Applications

The results of this project highlight several domains where CLIP-based hand biometrics could be practically deployed:

- **Contactless Identity Verification:** In high-security or hygiene-sensitive environments such as airports, hospitals, or government buildings, hand-based contactless verification offers a non-invasive and privacy-aware alternative to fingerprint or iris scans.
- **Healthcare Monitoring:** Hospitals and eldercare facilities could use continuous hand-based identity verification to prevent treatment mix-ups or ensure medication safety.

- **Immigration and Border Control:** Hands, being unobtrusive and socially neutral compared to facial capture, can serve as a secondary biometric trait at e-gates or manual checkpoints.
- **Personal Device Authentication:** Smart rings, watches, or handheld devices could adopt dorsal/palmar recognition as a fallback or complementary biometric, especially for users with face-covering or fingerprint accessibility issues.

These use cases support the notion that hand biometrics can play a practical role in multimodal identity systems, particularly when enhanced by scalable models like CLIP.

6.6.4 Research Implications for Vision-Language Models

The project contributes to the growing body of research that explores how vision-language models can be adapted for specialised tasks. Key insights include:

- **Fine-Tuning is Essential:** While CLIP's zeroshot performance was strong, fine-tuning (especially with ReID enhancements) delivered the largest performance gains and is indispensable for domain-specific adaptation.
- **Prompt Learning is a Viable Complement:** Though more impactful in transformer backbones like ViT-B/16, prompt-guided fine-tuning (PromptSG) showed clear benefits in boosting semantic alignment and identity robustness.
- **Traditional ReID Tools Remain Powerful:** Established techniques such as BNNeck, ArcFace, Triplet loss, and Center loss remain highly effective when adapting CLIP for retrieval-based biometric tasks, even in a vision-language context.
- **Architectural Flexibility Matters:** The relative underperformance of RN50 suggests that transformers are better suited for cross-modal fine-tuning tasks. Future multimodal architectures may increasingly favour ViT-style backbones.

In summary, this work bridges the gap between general-purpose vision-language models and highly specialised biometric domains. It lays the foundation for future research into prompt-driven, multimodal biometric systems, and encourages exploration of CLIP's capabilities far beyond its original intent.

7 Conclusion and Future Work

7.1 Summary of Findings

This project systematically explored the adaptation of the CLIP model for hand-based identity matching through a structured four-stage pipeline: baseline zeroshot evaluation, CLIP-ReID-style fine-tuning, prompt learning via PromptSG, and final benchmarking against state-of-the-art hand recognition methods. Each stage revealed critical insights into the behaviour and adaptation potential of large vision-language models in the biometric domain.

Stage 1 – Baseline (Zeroshot) Evaluation: The pre-trained CLIP models were evaluated without any domain-specific fine-tuning. ViT-B/16 achieved a mean Rank-1 accuracy of 71.42% and mAP of 78.98%, while RN50 reached 68.61% Rank-1 and 75.78% mAP. These results confirmed that CLIP encodes meaningful visual similarities even for unseen domains like hand biometrics. However, its general-purpose embeddings lacked the fine-grained discrimination required for reliable identity matching, especially among anatomically similar individuals.

Stage 2 – CLIP-ReID Integration (Fine-Tuning): The CLIP image encoder was fine-tuned using a suite of ReID-centric loss functions, including Cross-Entropy, Triplet, Center, ArcFace, and SupCon. Structural enhancements such as BNNeck and a modular classifier head were added. This stage led to the largest single-stage performance gain. ViT-B/16 reached up to 88.18% Rank-1 and 92.20% mAP, confirming that supervised fine-tuning enables the model to learn identity-level hand representations. RN50 improved moderately but was less stable across configurations.

Stage 3 – PromptSG Integration (Prompt-Based Joint Tuning): This stage introduced learnable prompt embeddings, pseudo-token inversion, and multimodal fusion via PromptSG. Jointly training the visual encoder with these prompt modules led to further performance gains, with ViT-B/16 achieving up to 85.86% Rank-1 and 89.99% mAP. Prompt-guided supervision helped reinforce semantic consistency and improved retrieval robustness. Though the gains were more modest compared to Stage 2, this stage showcased the potential of cross-modal learning even in low-semantic domains like hands.

Stage 4 – Comparative Benchmarking: The best-performing models from each stage were compared against traditional hand recognition networks, including GPA-Net, RGA-Net, ABD-Net, and MBA-Net. While MBA-Net still holds the top spot with 97.45% Rank-1 and 97.98% mAP, the CLIP-ReID model (Stage 2 ViT-B/16) achieved competitive results (88.18% Rank-1, 92.20% mAP), outperforming RGA-Net and closely approaching GPA-Net. This is notable given that CLIP was originally trained for generic image-text matching and had no prior exposure to biometric datasets.

Backbone Comparison: Across all stages, ViT-B/16 consistently outperformed RN50. The transformer-based ViT-B/16 demonstrated better generalisation, adaptability to prompt learning, and tolerance to ReID-specific architectural modifications. RN50, while competitive in early stages, was sensitive to fine-tuning depth and performed inconsistently in complex multimodal setups.

Key Takeaways:

- **Stage 2 (CLIP-ReID Fine-Tuning)** produced the most substantial improvement, validating the use of ReID techniques with vision-language models.
- **Stage 3 (PromptSG Integration)** offered additional performance gains through semantic conditioning, especially in ViT-B/16.
- **CLIP's Transformer Backbone** proved better suited for biometric tasks than its CNN-based counterpart (RN50).
- **CLIP-Based Models**, when properly adapted, can approach the performance of specialised hand recognition systems without requiring handcrafted features or domain-specific architectures.

In conclusion, the staged adaptation strategy validated that large pre-trained models like CLIP can be repurposed effectively for biometric applications. Through careful fine-tuning, ReID-aware training, and prompt-guided semantic supervision, CLIP evolved into a competitive and flexible hand identity recogniser—bridging the gap between general-purpose multimodal learning and specialised biometric tasks.

7.2 Future Work

Several directions remain open for further exploration and enhancement:

- **Multi-Aspect Dataset Expansion:** Future work should include the full set of hand variants in the 11k dataset (dorsal left, palmar left, palmar right), as well as integration of the high-resolution HD Hands dataset. These were excluded in the current study due to GPU memory and time limitations.

- **Text Encoder Fine-Tuning:** This project froze the text encoder throughout all stages. Unfreezing and jointly training the text encoder alongside the image encoder may unlock deeper cross-modal alignment and improve semantic grounding.
- **Domain Adaptation Techniques:** Exploring unsupervised domain adaptation or adversarial learning approaches could help bridge the distributional gap between CLIP's pretraining data and the biometric domain.
- **Prompt Ensemble and Generation:** Investigating prompt ensembles and automatically generated or adapted prompts (using NLP techniques or pretrained language models) may improve the robustness and adaptability of prompt learning stages.
- **Real-Time Application Deployment:** Lightweight versions of the ViT backbone or pruning/distillation of the fine-tuned models could enable deployment on embedded or edge devices for use in mobile ID verification or IoT-based biometric authentication.
- **Cross-Modality Biometrics:** Extending this framework to other biometric modalities (e.g., face, ear, gait) would demonstrate the generalisability of the staged CLIP adaptation strategy in multimodal identity systems.

Overall, this project contributes a reproducible and extensible pipeline for adapting vision-language models to specialised biometric tasks. It encourages future work at the intersection of ReID, prompt learning, and multimodal AI—particularly in low-resource or non-traditional visual domains.

7.3 Contribution to Knowledge

This project makes several important contributions to the growing field of biometric recognition using large vision-language models like CLIP. It extends existing research by adapting and evaluating CLIP for a new, less-explored domain—hand-based identity matching—and by systematically integrating ReID-specific strategies and prompt learning techniques within a reproducible training framework.

- **Demonstrating the Fine-Tuning Potential of CLIP for Hand Biometrics:** This work empirically shows that although CLIP provides strong zeroshot performance, domain-specific fine-tuning of its image encoder—using ReID losses like Triplet, Center, ArcFace, and SupCon—is essential for extracting identity-level discriminative features in hand images. The performance improvement from 78.98% to 92.20% mAP (ViT-B/16) validates the importance of fine-tuning even for powerful pre-trained models.

- **Benchmarking ReID-Style Evaluation of CLIP for Hands:** The project introduced a rigorous ReID-style evaluation setup for hand biometrics using 10 query-gallery splits, inspired by MBA-Net. This is the first known implementation of such an evaluation pipeline specifically for CLIP models in the hand biometric domain, establishing a new benchmark protocol.

- **Introducing Prompt Learning for Hand Recognition:** PromptSG-style joint training, involving pseudo-token learning and prompt composition, was adapted and applied to hand biometrics for the first time. Despite the low semantic diversity in hand images, this method improved feature robustness and yielded up to 89.99% mAP with ViT-B/16. This contributes new insights into prompt learning's effectiveness in anatomically homogeneous domains.

- **Comparative Analysis of Transformer vs CNN Backbones:** A controlled comparison was conducted between CLIP's transformer-based ViT-B/16 and CNN-based RN50 backbones across three stages. The consistent and substantial outperformance of ViT-B/16 demonstrated the superior global feature encoding and adaptability of vision transformers in biometric identification tasks.

- **Development of a Modular, Reproducible Training Pipeline:** The project implemented a fully modular training framework, driven by YAML configuration files. This pipeline supports backbones (ViT-B/16, RN50), loss combinations, prompt variants, optimizer/scheduler toggling, and staged evaluation over 10 splits. This extensible design promotes reproducibility and can be generalised to other visual biometric modalities or multimodal tasks.

7.3.1 Originality and Relevance

While recent works like CLIP-ReID and PromptSG have demonstrated the adaptation of CLIP for person re-identification, this project is among the first to explore and evaluate:

- The application of CLIP for fine-grained, hand-based identity matching—an understudied biometric modality in the vision-language domain.

- The combination of ReID-specific architectural enhancements (BNNeck, ArcFace) and semantic prompt learning techniques in a single end-to-end training pipeline for hand biometrics.

- A staged, performance-driven exploration that disentangles the effect of fine-tuning, prompt integration, and architectural variations across transformer and CNN backbones.

By bridging the gap between general-purpose multimodal models and specialised biometric recognition tasks, this project makes a meaningful and original contribution to the field. It highlights CLIP's potential as a flexible foundation for future multimodal biometric systems and lays the groundwork for further research into prompt-guided and ReID-aligned fine-tuning across biometric domains.

Bibliography

Afifi, Mahmoud (2019a). "11K Hands: Gender Recognition and Biometric Identification Using a Large Dataset of Hand Images". In: *Multimedia Tools and Applications* 78.23, pp. 33035–33057. DOI: 10.1007/s11042-019-7424-8. URL: https://doi.org/10.1007/s11042-019-7424-8.

— (2019b). "11K Hands: Gender recognition and biometric identification using a large dataset of hand images". In: *Multimedia Tools and Applications* 78, pp. 20835–20854.

Azulay, Aharon and Yair Weiss (2018). "Why do deep convolutional networks generalize so poorly to small image transformations?" In: *arXiv preprint arXiv:1805.12177*. URL: https://arxiv.org/abs/1805.12177.

Baisa, Nathanael L et al. (2022a). "Hand-based person identification using global and part-aware deep feature representation learning". In: *2022 Eleventh International Conference on Image Processing Theory, Tools and Applications (IPTA)*. IEEE, pp. 1–6.

— (2022b). "Multi-branch with attention network for hand-based person recognition". In: *2022 26th International Conference on Pattern Recognition (ICPR)*. IEEE, pp. 727–732.

— (2022c). "GPA-Net: Global and Part-Aware Network for Hand-Based Person Identification Using Deep Feature Learning". In: *2022 Eleventh International Conference on Image Processing Theory, Tools and Applications (IPTA)*.

Beitzel, Steven M., Eric C. Jensen, and Ophir Frieder (2009). "MAP". In: *Encyclopedia of Database Systems*. Springer, pp. 1691–1692. DOI: 10.1007/978-0-387-39940-9_492. URL: https://link.springer.com/referenceworkentry/10.1007/978-0-387-39940-9_492.

Bhat, Aaditya and Shrey Jain (2023). "Face recognition in the age of clip & billion image datasets". In: *arXiv preprint arXiv:2301.07315*.

Bolcato, Pietro (2023). *Understanding Vision-Language Models (VLMs): A Practical Guide*. Accessed: 2025-05-28. URL: https://medium.com/@pietrobolcato/understanding-vision-language-models-vlms-a-practical-guide-8da18e9f0e0c.

Chen, Tianlong et al. (2019). "Abd-net: Attentive but diverse person re-identification". In: *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 8351–8361.

Contrastive Loss - an overview (n.d.). https://www.sciencedirect.com/topics/computer-science/contrastive-loss. Accessed: 2025-05-28.

Contributors, OpenCLIP (2023). *OpenCLIP: An Open Source Implementation of CLIP*. Accessed: 2025-05-28. URL: https://github.com/mlfoundations/open_clip.

Dantcheva, Antitza, Petros Elia, and Arun Ross (2016). "What Else Does Your Biometric Data Reveal? A Survey on Soft Biometrics". In: *IEEE Transactions on Information Forensics and Security* 11.3, pp. 441–467. DOI: 10.1109/TIFS.2015.2480381.

Dosovitskiy, Alexey et al. (2020). "An image is worth 16x16 words: Transformers for image recognition at scale". In: *arXiv preprint arXiv:2010.11929*.

Douze, Matthijs et al. (2024). "The faiss library". In: *arXiv preprint arXiv:2401.08281*.

Gokoruri007 (2021). *pytorch_arcface: ArcFace Implementation in PyTorch*. https://github.com/GOKORURI007/pytorch_arcface. Accessed: 2025-05-28.

Han, Q. et al. (2024). "Synthesized Caption-Guided Inversion for Person Re-Identification". In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*. Vol. 38. 6, pp. 5162–5170. DOI: 10.1609/aaai.v38i6.28315. URL: https://ojs.aaai.org/index.php/AAAI/article/view/28315.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman (2001). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer.

He, Kaiming et al. (2016). "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

HobbitLong (2021). *SupContrast: Supervised Contrastive Learning*. https://github.com/HobbitLong/SupContrast. Accessed: 2025-05-28.

JetBrains (2025). *PyCharm: Python IDE for professional developers*. Accessed: 30 May 2025. URL: https://www.jetbrains.com/pycharm.

Le-Khac, Phuc H, Graham Healy, and Alan F Smeaton (2020). "Contrastive representation learning: A framework and review". In: *Ieee Access* 8, pp. 193907–193934.

Khosla, Prannay et al. (2020). "Supervised contrastive learning". In: *Advances in neural information processing systems* 33, pp. 18661–18673.

Li, Siyuan, Li Sun, and Qingli Li (2023). "CLIP-ReID: Exploiting Vision-Language Model for Image Re-Identification without Concrete Text Labels". In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*. Vol. 37. 4, pp. 3919–3927. DOI: 10.1609/aaai.v37i4.25408. URL: https://ojs.aaai.org/index.php/AAAI/article/view/25408.

O'shea, Keiron and Ryan Nash (2015). "An introduction to convolutional neural networks". In: *arXiv preprint arXiv:1511.08458*.

Oord, Aaron van den, Yazhe Li, and Oriol Vinyals (2018). "Representation Learning with Contrastive Predictive Coding". In: *arXiv preprint arXiv:1807.03748*. URL: https://arxiv.org/abs/1807.03748.

— (2018). "Representation learning with contrastive predictive coding". In: *arXiv preprint arXiv:1807.03748*.

OpenAI (2025). *OpenAI*. Accessed May 28, 2025. URL: https://github.com/openai/CLIP.

Paszke, Adam et al. (2019). "PyTorch: An imperative style, high-performance deep learning library". In: *Advances in Neural Information Processing Systems* 32. Accessed: 30 May 2025, pp. 8024–8035. URL: https://pytorch.org.

Python Software Foundation (2025). *Python Language Reference, version 3.11*. Accessed: 30 May 2025. URL: https://www.python.org.

PyTorch Contributors (2024a). *torch.nn.CrossEntropyLoss — PyTorch Documentation*. https://pytorch.org/docs/stable/generated/torch.nn.CrossEntropyLoss.html. Accessed: 2025-05-28.

— (2024b). *torch.nn.TripletMarginLoss — PyTorch Documentation*. https://pytorch.org/docs/stable/generated/torch.nn.TripletMarginLoss.html. Accessed: 2025-05-28.

Radford, Alec et al. (2021). "Learning Transferable Visual Models From Natural Language Supervision". In: *arXiv preprint arXiv:2103.00020*. DOI: 10.48550/arXiv.2103.00020. URL: https://arxiv.org/abs/2103.00020.

Rangel, Gabriela et al. (2024). "A survey on convolutional neural networks and their performance limitations in image recognition tasks". In: *Journal of sensors* 2024.1, p. 2797320.

Simonyan, Karen and Andrew Zisserman (2014). "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556*.

Srivastava, Nitish and Russ R Salakhutdinov (2012). "Multimodal learning with deep boltzmann machines". In: *Advances in neural information processing systems* 25.

Umer, Saiyed et al. (2020). "Person identification using fusion of iris and periocular deep features". In: *Neural Networks* 122, pp. 407–419.

Wang, Xinyi et al. (2022). "A survey of face recognition". In: *arXiv preprint arXiv:2212.13038*.

Yang, Zexian et al. (2024). "A pedestrian is worth one prompt: Towards language guidance person re-identification". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17343–17353.

Yuan, Yimin et al. (2020). "HandNet: Identification based on hand images using deep learning methods". In: *Proceedings of the 2020 4th International Conference on Vision, Image and Signal Processing*, pp. 1–6.

Zhang, Jingyi et al. (2024). "Vision-language models for vision tasks: A survey". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Zhang, Zhizheng et al. (2020). "Relation-aware global attention for person re-identification". In: *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, pp. 3186–3195.

Zhou, Kaiyang (2018). *pytorch-center-loss*. `https://github.com/KaiyangZhou/pytorch-center-loss`. Accessed: 2025-05-28.

A Codebase Structure

This appendix presents the full directory and file structure of the CLIP-FH project, reflecting its modular implementation and reproducibility.

```
.
.idea/
    .name
    HandCLIP.iml
    inspectionProfiles/
        Project_Default.xml
        profiles_settings.xml
    material_theme_project_new.xml
    misc.xml
    vcs.xml
    workspace.xml
README.md
clip_fh_v4.zip
code_structure.txt
configs/
    README.md
    baseline/
        eval_vitb16_11k_dorsal_r.yml
    eval_stage1_frozen_text/
        eval_rn50_11k_dorsal_r.yml
        eval_vitb16_11k_dorsal_r.yml
    eval_stage2_clip_reid/
        eval_joint_vitb16_11k_dorsal_r.yml
        eval_vitb16_11k_dorsal_r.yml
    eval_stage3_promptsg/
        eval_stage3_rn50_11k_dorsal_r.yml
        eval_stage3_vitb16_11k_dorsal_r.yml
    train_stage1_frozen_text/
        continue_train_vitb16_11k_dorsal_r.yml
        train_rn50_11k_dorsal_r.yml
        train_vitb16_11k_dorsal_r.yml
    train_stage2_clip_reid/
        train_joint_rn50_11k_dorsal_r.yml
        train_joint_vitb16_11k_dorsal_r.yml
```

```
    train_stage3_promptsg/
        train_stage3_rn50_11k_dorsal_r.yml
        train_stage3_vitb16_11k_dorsal_r.yml
    validate_stage2_clip_reid/
        validate_stage2a_trainset_vitb16_11k_dorsal_r.yml
        validate_stage2a_validationset_vitb16_11k_dorsal_r.yml
datasets/
    README.md
    __init__.py
    __pycache__/
        __init__.cpython-310.pyc
        __init__.cpython-38.pyc
    data_preprocessing/
        README.md
        prepare_train_val_test_11k_r_l.py
        prepare_train_val_test_hd.py
        validate_dataset_splits.py
    dataset_analysis/
        __init__.py
        analyze_11k_dataset.py
        analyze_hd_dataset.py
        check_class_imbalance.py
        diagnosing the destractors.py
engine/
    README.md
    baseline_inference.py
    evaluator.py
    prompt_learner.py
    promptsg_inference.py
    train_classifier_stage1.py
    train_clipreid_stages.py
eval_logs/
    stage0/
        eval_stage0_baseline_rn50_11k_dorsal_r_baseline_e0_bs32.log
        eval_stage0_baseline_vitb16_11k_dorsal_r_baseline_e0_bs32.log
    stage2-v1/
        eval_stage2_clipreid_rn50_11k_dorsal_r_clipid_e30_bs32_BEST.log
        eval_stage2_clipreid_vitb16_11k_dorsal_r_clipid_e30_bs32_BEST.log
    stage2-v10/
        eval_stage2_clipreid_rn50_11k_dorsal_r_clipid_e30_bs32_BEST.log
    stage2-v2/
        eval_stage2_clipreid_rn50_11k_dorsal_r_clipid_e30_bs32_BEST.log
        eval_stage2_clipreid_vitb16_11k_dorsal_r_clipid_e30_bs32_BEST.log
    stage2-v3/
        eval_stage2_clipreid_rn50_11k_dorsal_r_clipid_e30_bs32_BEST.log
        eval_stage2_clipreid_vitb16_11k_dorsal_r_clipid_e30_bs32_BEST.log
    stage2-v4/
```

```
        eval_stage2_clipreid_rn50_11k_dorsal_r_clipid_e30_bs32_BEST.log
        eval_stage2_clipreid_vitb16_11k_dorsal_r_clipid_e30_bs32_BEST.log
    stage2-v5/
        eval_stage2_clipreid_rn50_11k_dorsal_r_clipid_e30_bs32_BEST.log
        eval_stage2_clipreid_vitb16_11k_dorsal_r_clipid_e30_bs32_BEST.log
    stage2-v6/
        eval_stage2_clipreid_rn50_11k_dorsal_r_clipid_e30_bs32_BEST.log
        eval_stage2_clipreid_vitb16_11k_dorsal_r_clipid_e30_bs32_BEST.log
    stage2-v7/
        eval_stage2_clipreid_rn50_11k_dorsal_r_clipid_e30_bs32_BEST.log
    stage2-v8/
        eval_stage2_clipreid_rn50_11k_dorsal_r_clipid_e30_bs32_BEST.log
    stage2-v9/
        eval_stage2_clipreid_rn50_11k_dorsal_r_clipid_e30_bs32_BEST.log
        eval_stage2_clipreid_vitb16_11k_dorsal_r_clipid_e30_bs32_BEST.log
    stage3-v1/
        eval_stage3_promptsg_rn50_11k_dorsal_r_promptsg_e0_lrNA_bs32.log
        eval_stage3_promptsg_vitb16_11k_dorsal_r_promptsg_e0_lrNA_bs32.log
    stage3-v10/
        eval_stage3_promptsg_rn50_11k_dorsal_r_promptsg_e60_lrNA_bs32.log
        eval_stage3_promptsg_vitb16_11k_dorsal_r_promptsg_e30_lrNA_bs32.log
    stage3-v11/
        eval_stage3_promptsg_rn50_11k_dorsal_r_promptsg_e60_lrNA_bs32.log
        eval_stage3_promptsg_vitb16_11k_dorsal_r_promptsg_e60_lrNA_bs32.log
    stage3-v2/
        eval_stage3_promptsg_rn50_11k_dorsal_r_promptsg_e0_lrNA_bs32.log
        eval_stage3_promptsg_vitb16_11k_dorsal_r_promptsg_e0_lrNA_bs32.log
    stage3-v3/
        eval_stage3_promptsg_rn50_11k_dorsal_r_promptsg_e0_lrNA_bs32.log
        eval_stage3_promptsg_vitb16_11k_dorsal_r_promptsg_e0_lrNA_bs32.log
    stage3-v4/
        eval_stage3_promptsg_rn50_11k_dorsal_r_promptsg_e0_lrNA_bs32.log
        eval_stage3_promptsg_vitb16_11k_dorsal_r_promptsg_e0_lrNA_bs32.log
    stage3-v5/
        eval_stage3_promptsg_rn50_11k_dorsal_r_promptsg_e0_lrNA_bs32.log
        eval_stage3_promptsg_vitb16_11k_dorsal_r_promptsg_e0_lrNA_bs32.log
    stage3-v6/
        eval_stage3_promptsg_rn50_11k_dorsal_r_promptsg_e0_lrNA_bs32.log
        eval_stage3_promptsg_vitb16_11k_dorsal_r_promptsg_e0_lrNA_bs32.log
    stage3-v7/
    stage3-v8/
        eval_stage3_promptsg_rn50_11k_dorsal_r_promptsg_e0_lrNA_bs32.log
        eval_stage3_promptsg_vitb16_11k_dorsal_r_promptsg_e0_lrNA_bs32.log
    stage3-v9/
        eval_stage3_promptsg_rn50_11k_dorsal_r_promptsg_e60_lrNA_bs32.log
        eval_stage3_promptsg_vitb16_11k_dorsal_r_promptsg_e60_lrNA_bs32.log
experiments/
```

```
    README.md
    conclusion_outputs/
        generate_tree.py
        plot_stage2_train_metrics.py
        plot_stage3_train_metrics.py
        stage2_eval_log_analysis.py
        stage2_train_log_analysis.py
        stage3_eval_log_analysis.py
        stage3_train_log_analysis.py
    run_eval_clip.py
    stage1_baseline_inference/
        ANALYSIS REPORT.md
        eval_baseline_clip.py
        eval_baseline_clip_single.py
    stage2_clipreid_integration/
        ANLAYSIS REPORT_FORMATTED.md
        ANLAYSIS REPORT_UNFORMATTED.md
        README.md
        eval_stage2_joint.py
        train_stage2_joint.py
    stage3_promptsg_integration/
        ANLAYSIS REPORT_FORMATTED.md
        ANLAYSIS REPORT_UNFORMATTED.md
        README.md
        eval_stage3_promptsg.py
        train_stage3_promptsg.py
main.py
requirements.txt
result_logs/
    README.md
    code_structure.txt
    dataset_11k_summary.txt
    dataset_class_imbalance_check.txt
    dataset_hd_summary.txt
    stage2_rn50_eval_table.csv
    stage2_rn50_train_table.csv
    stage2_rn50_train_table_final_map_per_version.png
    stage2_rn50_train_table_final_rank1_per_version.png
    stage2_rn50_train_table_loss_breakdown_facet.png
    stage2_rn50_train_table_loss_vs_epoch.png
    stage2_rn50_train_table_map_vs_epoch.png
    stage2_rn50_train_table_rank1_vs_epoch.png
    stage2_vitb16_eval_table.csv
    stage2_vitb16_train_table.csv
    stage2_vitb16_train_table_final_map_per_version.png
    stage2_vitb16_train_table_final_rank1_per_version.png
    stage2_vitb16_train_table_loss_breakdown_facet.png
```

```
    stage2_vitb16_train_table_loss_vs_epoch.png
    stage2_vitb16_train_table_map_vs_epoch.png
    stage2_vitb16_train_table_rank1_vs_epoch.png
    stage3_rn50_eval_table.csv
    stage3_rn50_train_table.csv
    stage3_rn50_train_table_final_val_map.png
    stage3_rn50_train_table_final_val_rank1.png
    stage3_rn50_train_table_val_map_vs_epoch.png
    stage3_rn50_train_table_val_rank1_vs_epoch.png
    stage3_vitb16_eval_table.csv
    stage3_vitb16_train_table.csv
    stage3_vitb16_train_table_final_val_map.png
    stage3_vitb16_train_table_final_val_rank1.png
    stage3_vitb16_train_table_val_map_vs_epoch.png
    stage3_vitb16_train_table_val_rank1_vs_epoch.png
    total_formatted_result_with_remarks.xlsx
train_logs/
    stage2-v1/
        stage2_clipreid_rn50_11k_dorsal_r_clipid_e30_bs32.log
        stage2_clipreid_vitb16_11k_dorsal_r_clipid_e30_bs32.log
    stage2-v10/
        stage2_clipreid_rn50_11k_dorsal_r_clipid_e30_bs32.log
    stage2-v2/
        stage2_clipreid_rn50_11k_dorsal_r_clipid_e30_bs32.log
        stage2_clipreid_vitb16_11k_dorsal_r_clipid_e30_bs32.log
    stage2-v3/
        stage2_clipreid_rn50_11k_dorsal_r_clipid_e30_bs32.log
        stage2_clipreid_vitb16_11k_dorsal_r_clipid_e30_bs32.log
    stage2-v4/
        stage2_clipreid_rn50_11k_dorsal_r_clipid_e30_bs32.log
        stage2_clipreid_vitb16_11k_dorsal_r_clipid_e30_bs32.log
    stage2-v5/
        stage2_clipreid_rn50_11k_dorsal_r_clipid_e30_bs32.log
        stage2_clipreid_vitb16_11k_dorsal_r_clipid_e30_bs32.log
    stage2-v6/
        stage2_clipreid_rn50_11k_dorsal_r_clipid_e30_bs32.log
        stage2_clipreid_vitb16_11k_dorsal_r_clipid_e30_bs32.log
    stage2-v7/
        stage2_clipreid_rn50_11k_dorsal_r_clipid_e30_bs32.log
    stage2-v8/
        stage2_clipreid_rn50_11k_dorsal_r_clipid_e30_bs32.log
    stage2-v9/
        stage2_clipreid_rn50_11k_dorsal_r_clipid_e30_bs32.log
        stage2_clipreid_vitb16_11k_dorsal_r_clipid_e30_bs32.log
    stage3-v1/
        stage3_promptsg_rn50_11k_dorsal_r_promptsg_e20_lrNA_bs32.log
        stage3_promptsg_vitb16_11k_dorsal_r_promptsg_e20_lrNA_bs32.log
```

```
    stage3-v10/
        stage3_promptsg_rn50_11k_dorsal_r_promptsg_e60_lrNA_bs32.log
        stage3_promptsg_vitb16_11k_dorsal_r_promptsg_e30_lrNA_bs32.log
    stage3-v11/
        stage3_promptsg_rn50_11k_dorsal_r_promptsg_e60_lrNA_bs32.log
        stage3_promptsg_vitb16_11k_dorsal_r_promptsg_e60_lrNA_bs32.log
    stage3-v2/
        stage3_promptsg_rn50_11k_dorsal_r_promptsg_e20_lrNA_bs32.log
        stage3_promptsg_vitb16_11k_dorsal_r_promptsg_e20_lrNA_bs32.log
    stage3-v3/
        stage3_promptsg_rn50_11k_dorsal_r_promptsg_e20_lrNA_bs32.log
        stage3_promptsg_vitb16_11k_dorsal_r_promptsg_e20_lrNA_bs32.log
    stage3-v4/
        stage3_promptsg_rn50_11k_dorsal_r_promptsg_e20_lrNA_bs32.log
        stage3_promptsg_vitb16_11k_dorsal_r_promptsg_e20_lrNA_bs32.log
    stage3-v5/
        stage3_promptsg_rn50_11k_dorsal_r_promptsg_e20_lrNA_bs32.log
        stage3_promptsg_vitb16_11k_dorsal_r_promptsg_e20_lrNA_bs32.log
    stage3-v6/
        stage3_promptsg_vitb16_11k_dorsal_r_promptsg_e20_lrNA_bs32.log
    stage3-v7/
        stage3_promptsg_rn50_11k_dorsal_r_promptsg_e30_lrNA_bs32.log
        stage3_promptsg_vitb16_11k_dorsal_r_promptsg_e20_lrNA_bs32.log
    stage3-v8/
        stage3_promptsg_rn50_11k_dorsal_r_promptsg_e30_lrNA_bs32.log
        stage3_promptsg_vitb16_11k_dorsal_r_promptsg_e20_lrNA_bs32.log
    stage3-v9/
        stage3_promptsg_rn50_11k_dorsal_r_promptsg_e60_lrNA_bs32.log
        stage3_promptsg_vitb16_11k_dorsal_r_promptsg_e60_lrNA_bs32.log
utils/
    __init__.py
    clip_patch.py
    dataloaders.py
    device_selection.py
    eval_utils.py
    feature_cache.py
    logger.py
    loss/
        README.md
        __init__.py
        arcface.py
        archives/
            __init__.py
            contrastive_loss.py
        center_loss.py
        cross_entropy_loss.py
        make_loss.py
```

```
        supcon.py
        triplet_loss.py
    naming.py
    save_load_models.py
    train_helpers.py
    transforms.py
```

B Codebase Setup and Execution Guide

B.1 Overview

This appendix explains how to set up and execute the codebase used in this project: **CLIP-FH — Fine-Tuning CLIP for Hand-Based Identity Matching**. It includes environment configuration, data preparation, model training and evaluation, and performance visualization.

B.2 Environment Setup

B.2.1 Create Environment and Install Dependencies

```
python -m venv clipfh-env
source clipfh-env/bin/activate   # Windows: clipfh-env\Scripts\
   activate

pip install -r requirements.txt
```

If the ‘requirements.txt‘ file is missing, manually install key dependencies:

```
pip install torch torchvision ftfy regex tqdm matplotlib seaborn
   pandas scikit-learn PyYAML faiss-cpu
pip install git+https://github.com/openai/CLIP.git
```

B.3 Dataset Preparation

B.3.1 Prepare and Split 11k Hand Dataset for Train/Val/Test

The 11k Hands dataset is used in this project for training and evaluation. It must be downloaded and properly organized before running the data preparation script.

1. Download the dataset from `https://sites.google.com/view/11khands`

2. Create the following folder structure under your project root:

```
datasets/
    11k/
        Hands/
        HandInfo.csv
```

3. Place all hand images inside the `Hands/` directory and the metadata file `HandInfo.csv` in the same level.

4. Run the preprocessing script to split the dataset into training, validation, and test sets:

```
python datasets/data_preprocessing/prepare_train_val_test_11k_r_l.py
```

This script generates separate folders for train/val/test and creates ReID-style query-gallery splits for evaluation.

B.4 Stage 1: Baseline Evaluation (Zero-Shot)

B.4.1 Evaluate Pretrained CLIP Models

```
python experiments/stage0_baseline_inference/
   eval_baseline_clip_single.py \
  --config configs/baseline/eval_vitb16_11k_dorsal_r.yml

python experiments/stage0_baseline_inference/
   eval_baseline_clip_single.py \
  --config configs/baseline/eval_rn50_11k_dorsal_r.yml
```

B.5 Stage 2: CLIP-ReID Fine-Tuning

B.5.1 Train with Prompt + Image Encoder (Joint)

```
python experiments/stage2_clipreid_integration/train_stage2_joint.py
    \
  --config configs/train_stage2_clip_reid/
     train_joint_vitb16_11k_dorsal_r.yml
```

```
python experiments/stage2_clipreid_integration/train_stage2_joint.py
    \
  --config configs/train_stage2_clip_reid/
     train_joint_rn50_11k_dorsal_r.yml
```

B.5.2 Evaluate Stage 2 Models

```
python experiments/stage2_clipreid_integration/eval_stage2_joint.py
   \
  configs/eval_stage2_clip_reid/eval_joint_vitb16_11k_dorsal_r.yml

python experiments/stage2_clipreid_integration/eval_stage2_joint.py
   \
  configs/eval_stage2_clip_reid/eval_joint_rn50_11k_dorsal_r.yml
```

B.6 Stage 3: PromptSG Integration

B.6.1 Train PromptSG with Image Encoder

```
python experiments/stage3_promptsg_integration/train_stage3_promptsg
   .py \
  --config configs/train_stage3_promptsg/
     train_stage3_vitb16_11k_dorsal_r.yml

python experiments/stage3_promptsg_integration/train_stage3_promptsg
   .py \
  --config configs/train_stage3_promptsg/
     train_stage3_rn50_11k_dorsal_r.yml
```

B.6.2 Evaluate Stage 3 Models

```
python experiments/stage3_promptsg_integration/eval_stage3_promptsg.
   py \
  configs/eval_stage3_promptsg/eval_stage3_vitb16_11k_dorsal_r.yml

python experiments/stage3_promptsg_integration/eval_stage3_promptsg.
   py \
  configs/eval_stage3_promptsg/eval_stage3_rn50_11k_dorsal_r.yml
```

B.7 Visualization and Analysis Tools

B.7.1 Tree Structure Diagram

```
python experiments/conclusion_outputs/generate_tree.py
```

B.7.2 Log Analysis (CSV Creation)

```
python experiments/conclusion_outputs/stage2_eval_log_analysis.py
python experiments/conclusion_outputs/stage3_eval_log_analysis.py

python experiments/conclusion_outputs/stage2_train_log_analysis.py
python experiments/conclusion_outputs/stage3_train_log_analysis.py
```

B.7.3 Training Metric Plots

```
python experiments/conclusion_outputs/plot_stage2_train_metrics.py
python experiments/conclusion_outputs/plot_stage3_train_metrics.py
```

B.8 Notes

- Always verify the YAML config files before executing any stage.
- For RN50 models, duplicate ViT-B/16 configs and change ‘MODEL.NAME‘ to ‘RN50‘.
- You can extend to other datasets (e.g., HD-Hands) by updating dataset paths in the configs.

B.9 Contact

For any issues or collaboration, contact: babupallam@gmail.com.

C Sample YAML Configuration Structure (Stage 3)

The Stage 3 configuration files define all training and evaluation settings for PromptSG-based joint fine-tuning using ViT-B/16 on the 11k dorsal right-hand dataset. This appendix provides both a structured summary and the raw YAML content for reference. This is the same way the stage 1 and stage 2 has been designed.

1. Training Configuration Breakdown (train_stage3_vitb16_11k

- **model.name**: `ViT-B/16`
- **model.prompt_learning**:
  - `enable`: true
  - `prompt_type`: `PromptSG`
  - `n_ctx`: 8
  - `ctx_init`: "a photo of a hand"
  - `prompt_template`: {} is a photo of a hand.
- **losses**:
  - `supcon_loss_weight`: 1.0
  - `ce_loss_weight`: 1.0
- **dataloader**:
  - `sampler`: PKSampler
  - `P`: 16
  - `K`: 2
  - `min_samples_per_id`: 2
- **training**:

  - `epochs`: 50
  - `batch_size`: (inferred from P×K)
  - `optimizer`: AdamW
  - `lr`: 1e-4
  - `lr_scheduler`: CosineAnnealingLR with warmup
  - `gradient_clipping`: 1.0

- **logging**:
  - `log_interval`: every 10 steps
  - `val_interval`: every 1 epoch
  - `save_best`: based on mAP

2. Evaluation Configuration Breakdown (eval_stage3_vitb16_11

- **model.name**: `ViT-B/16`
- **eval.checkpoint**: path to trained model (Stage 3 best)
- **eval.splits**:
  - `query`: `query0` to `query9`
  - `gallery`: `gallery0` to `gallery9`
- **metrics**: mAP, Rank-1, Rank-5, Rank-10
- **normalize_features**: true
- **save_log_path**: output/results/stage3/

3. Full YAML Listings (Appendix B)

The complete YAML files for training and evaluation are included below for reproducibility.

Appendix B.1: Training YAML – Stage 3

Listing C.1: Stage 3 PromptSG Configuration YAML (Formatted)

```yaml
# === General Experiment Info ===
experiment: stage3_promptsg
dataset: 11k
aspect: dorsal_r
model: vitb16
variant: promptsg
stage: stage3_promptsg

# === CLIP Model Config ===
clip_model: vitb16
pretrained: true
freeze_text_encoder: true

# === Textual Inversion (Pseudo-token) ===
pseudo_token_dim: 512
pseudo_token_init: random

# === Multimodal Interaction Module ===
cross_attention_layers: 1
transformer_layers: 3
attention_heads: 8

# === Prompt Configuration ===
use_composed_prompt: true
#prompt_template: "A detailed photo of {aspect} hand for
   identification."
#prompt_template: "A biometric scan of the {aspect} hand for ID
   verification."
prompt_template: "A captured frame showing a p e r s o n s {aspect}
   hand during surveillance."
#prompt_template: "Label: {aspect} hand, individual ID for re-
   identification."

# === Training Config ===
batch_size: 32
epochs: 60
early_stop_patience: 10

lr_clip_visual: 0.000002
```

```
lr_modules: 0.000002
weight_decay: 0.0005
supcon_loss_weight: 0
loss_tri_weight: 1.0
loss_id_weight: 0.2

# === Dataloader Settings ===
num_workers: 4
shuffle: true
pin_memory: true

#classifier: linear
classifier: arcface
bnneck_reduction: true
bnneck_dim: 256
val_type: reid  # Prevents CE-style val logic

# === Logging and Checkpoint Saving ===
output_dir: train_logs/
save_dir: saved_models/
save_frequency: 10
log_frequency: 50

# === Resume and Checkpoints ===
resume_from_checkpoint: null
```

Appendix B.2: Evaluation YAML – Stage 3

Listing C.2: Stage 3 PromptSG Evaluation YAML Configuration

```
# === PromptSG Evaluation Config for Stage 3 ===
experiment: eval_stage3_promptsg

dataset: 11k
aspect: dorsal_r
model: vitb16
variant: promptsg

batch_size: 32
num_workers: 4
```

```
# Path to the best trained model from Stage 3 (PromptSG)
model_path: saved_models/
   stage3_promptsg_vitb16_11k_dorsal_r_promptsg_e20_lrNA_bs32_BEST.
   pth

# Where to save the split-wise and average evaluation metrics
output_dir: eval_logs/
```

4. Remarks

The YAML files provide modular control over model architecture, training schedule, sampling strategy, and evaluation protocol. This makes it easier to switch between variants without code duplication. The evaluation was conducted over 10 randomized query-gallery splits following a Monte Carlo-style framework, reducing performance variance and ensuring statistical reliability.