Deep Learning for Person Re-identification

Thesis of MSc Degree

Xiang He

Supervisor: **Dr Sion Hannuna**

Department of Computer Science

University of Bristol

United Kingdom

26/04/2018

Acknowledgement

First of all, I wish to express my gratitude to my supervisor Doctor Sion Hannuna for his kind advice and guidance. During this review, his suggestion helped me a lot.

Then I want to say thanks to my second supervisor Doctor Victor Ponce Lopez. He gave me much useful guidance with his experience in person re-identification.

I want to thank the academic staffs of the Department of Computer Science, especially the lecturers of Research Skills: Oliver Ray, Aisling O'Kane, Emily Grundy, Emma Tweddle, Ivan Palomares Carrascosa and Bahar Rastegari. They gave me lots of advice on how to write a research review with academic English and the skills of using LaTeX.

I also want to thank my personal tutor and friend Doctor Song Liu, who always encouraged me when I was facing troubles.

Finally, thanks to my parents for their support of my learning abroad, and my friends for their help on my project.

Contents

1 Executive Summary **4**

2 Introduction **5**
2.1 Person Re-identification 5
2.2 Surveillance System 6
2.3 Challenges 6
2.4 Extended Project Achievements and Deliverables 7
2.5 Scope 8

3 Literature Review **9**
3.1 Deep Neural Network 9
3.2 Re-identification 10
3.3 Human-engineered Person Re-ID 12
3.3.1 Feature Extraction 12
3.3.2 Distance Metric Learning 14
3.4 Deep-learning-based Person Re-ID 16
3.5 Improving Approaches 20
3.6 Datasets 22
3.7 Evaluation Methods 24
3.7.1 CMC Curve 24
3.7.2 mAP 24
3.8 Summary 25

4 Project Execution **26**
4.1 Overview 26
4.2 Datasets Used in This Project 27
4.2.1 Market1501 27
4.2.2 DukeMTMC-reID 28
4.3 Dataset Enlargement 28
4.3.1 Generative Model 28
4.3.2 Wasserstein GAN 29
4.3.3 Training of Wasserstein GAN 30
4.4 Data Pre-processing 32
4.5 Model Construction 34
4.5.1 Baseline Model 35
4.5.2 Modifications 36
4.6 Model Training and Optimisation 38
4.6.1 Model Training 39
4.6.2 Optimisation 43
4.7 Summary 46

5 Experiments and Evaluation **47**
5.1 Evaluation Results 47
5.2 Evaluation Methodology 48
5.3 Baseline Model ("Proposed" in Table 3 and 4) 49
5.4 Baseline Model + RE + RR 50
5.5 Baseline Model + RE + RR + WGAN 51
5.6 Label Assignments 53
5.7 Summary 53

6 Conclusion **54**

Appendices **61**

A Main code **61**

1 Executive Summary

Public areas need precise long-term monitoring for aiding public safety, for example in tracking missing persons and monitoring antisocial behaviour. A method for automatically managing the challenges presented by manually controlled surveillance networks would be highly desirable. The identification and re-identification of individuals can potentially be performed automatically and efficiently with the implementation of a person re-identification system. However, currently, this remains an unsolved problem due to challenges, such as illumination variance, occlusion and camera variability. The performance of the traditional human-engineered system is limited in so far as it is error-prone and does not scale well.

This project presents a bespoke person re-identification system which features a dedicated deep neural network that archives state-of-the-art performance on one publicly available dataset and competitive results on another. Performance evaluation is undertaken using Cumulative Match Characteristic (CMC) curve and mean average precision (mAP). Our approach extends and modifies an existing neural network architecture for extracting deep features and improves its performance by exploiting several data augmentation techniques and re-ranking methodologies. Furthermore, the training data was enlarged by using a generative adversarial network (GAN). The approach to model optimisation is described in detail. Note that the task of pedestrian detection will be discussed but is not the central part of this project since a perfect detection bounding box is assumed to be given for every pedestrian.

The primary contributions of this project are as follows:

- A dedicated baseline neural network model which will be regarded as a deep feature extractor. This will feature a novel architecture.
- A data augmentation method with the generated samples from Wasserstein GAN as well as a novel method for assigning these synthetic samples appropriate labels.
- The combination of random erasing and re-ranking methods with the proposed model.
- A systematic performance evaluation benchmark will be delivered.

This project will be 20% type I (software development), 60% type II (investigatory) and 20% type III (theoretical). The primary investigation task is to do related research and design an appropriate model, and then the development task is to implement it by Python. The theoretical part exists in the optimisation algorithms, including Bayesian optimisation and random search. The model construction and training are the most time-consuming parts of the implementation.

2 Introduction

2.1 Person Re-identification

Person re-identification (re-ID) is an application of computer vision based on surveillance systems. It aims to match and recognise a pedestrian through the images or videos from intra-camera or inter-cameras, for instance, images obtained from a single camera at different time (intra-camera) and images obtained from two different non-overlapping cameras (inter-cameras). Nowadays, many companies and research institutes are interested in automatic surveillance systems, and lots of significant progress has been made [25, 28, 39, 55, 58]. The re-ID system is generally used among different non-overlapping cameras in different locations or the same cameras in different time for the same individual identification and re-identification, with a large number of applications including:

- **Behaviour identification**. Identify crime or other suspicious behaviours for the consideration of public safety. Monitor some area where emergencies may happen suddenly such as swimming pool and identify these accidents and emergencies immediately to avoid bigger loss.
- **Target tracking**. Track or search specific pedestrian, for instance, lost children or elders in a large range; Track the escape route of criminals to aid the police.
- **Transportation recording**. Record extensive transportation information and identify particular vehicles and accidents for different purposes.
- **Sports application**. For large-scale sports, including basketball and soccer, Automatically track and record players' behaviours to provide a better analysis.

At present, various solutions proposed for person re-identification systems can achieve high precision due to the application of deep learning in this area, and the identification speed is increasingly fast, the anti-interference ability is much stronger than before, and the problems in the manual monitoring system can be effectively solved with a person re-identification system.

Typically, a human-engineered person re-ID system consists of three modules: person detection, person tracking and person retrieval. And most of person re-ID tasks focus on the person retrieval process. The person retrieval process is executed by calculating:

$$i^* = argmax_{i \in 1,2,\ldots,N} M(Q, G_i) \tag{1}$$

Where i^* is the identity of query image Q. M is a distance metric function to measure the similarity between query image Q and Gallery image G. Hence, the person re-ID is committed to the following two aspects of research:

1. Find a robust feature representation that is insensitive to illumination, shooting angle and other variances;

2. Learn an appropriate distance metric for person re-ID.

There are many research efforts concentrating on feature extraction and distance metric learning to improve the performance of person re-ID network. On the other hand, during these years, deep-learning-based re-ID systems become increasingly popular due to its success in the area of image classification, which will be discussed more in the next section.

2.2 Surveillance System

Person re-ID is based on surveillance systems, especially large-scale one. Currently, surveillance systems are very ubiquitous in big cities. Usually, many cameras with fixed parameters are deployed in major areas for different purposes.

However, in a large-scale surveillance network, human-based monitoring is quite time-consuming and often of limited accuracy. Operators generally manage a significant amount of monitoring devices, and they have limited ability to monitor all of them simultaneously. The probability that they make mistakes usually depend on their personal experience and skill, which can inevitably bring a certain degree of security risks, and this is a common challenge for large-scale manually managed surveillance network. On the other hand, privacy problem also needs to be considered. Thus, an automatic person re-ID system is critical for surveillance network.

Recently there are plenty of areas requiring surveillance system. The increasing demand for constructing a more stable and reliable surveillance network in public places is very rapid. There is widely held aspiration to solve the problems existing in human-based surveillance systems.

2.3 Challenges

For a practical person re-ID system, given a query image, the system needs to rank all the gallery images by the similarity between them. Because each image may be obtained with different angles, illuminations, backgrounds and sometimes different extent of occlusion issues, the match process can be very challenging. Mainly, the challenges are existed in:

- **Camera parameters**. If two images are obtained from different cameras, the parameters set on them must be considered. Different resolutions and colour spaces may occur.

- **Shooting angles**. If two images are obtained from different angles, the shape and gesture of pedestrians can vary greatly.

- **Illumination**. Since two non-overlapping cameras may obtain images at different time or area, the illumination may differ and the visual feature can be various, especially for the RGB information. Sometimes the effect of shadow also needs to be considered.

- **Background clutter**. Different cameras lead to unique background information. It is common to eliminate the background information by finding the area of interest with a detection algorithm.

- **Occlusion**. The public area is large and complex and cameras cannot be guaranteed to face pedestrians without occlusion directly. And re-ID is hard with occlusion because the human body in the image is incomplete.

In a large complex and uncontrolled environment, the performance of re-ID relying on some conventional biological features such as face recognition is quite unstable. Some visual features of the pedestrian's appearance are proved to be more reliable for identification, mostly by the clothes they were wearing and the items they were carrying, such as backpacks. However, these features individually are not sufficient for matching people, mainly because many people are wearing similar clothes, carrying same items in public area, and the same person's visual characteristics can be very different because of the camera angle and light conditions and the existence of occlusion and other issues among non-overlapping cameras. Sometimes different people's appearance can be more similar than the same people, which imposes an even more significant challenge.

2.4 Extended Project Achievements and Deliverables

The specific achievements of this project are as follows:

- Adopt 50-layers ResNet [17] as the baseline model and modify it to extract deep features for person re-ID, which makes it a very effective model and achieve a significant performance improvement.

- Enlarge the training sets by using Wasserstein Generative Adversarial Network (GAN) [2] to generate more synthetic images for the training process and propose a new way to assign them labels.

- Random erasing [62] and re-ranking [61] are combined with this project to further improve its performance and avoid over-fitting.

- Based on CMC curve and mAP, evaluation of the performance of proposed method is conducted on two large-scale public datasets: Market1501 [57] and DukeMTMC-reID [60] and and achieve a state-of-the-art performance on Market1501 and competitive result on DukeMTMC-reID.

The corresponding deliverables will be as follow:

- Dedicated convolutional neural networks (CNN) that are improved and tuned for the person re-ID task.
- A GAN [14] model which can be used to enlarge the volume of datasets.
- A general performance evaluation benchmark for person re-identification task.
- A set of improvement methods for person re-ID.
- A systematic statistical performance comparison between different datasets and features and summarise their pros and cons.

The proposed method will be able to address several challenging issues in the area of person re-identification. The dedicated deep neural network can be reused by other researchers with well-tuned parameters and the benchmark can be utilised for any person re-identification task.

2.5 Scope

As mentioned before, a practical person re-ID system can be divided into three main modules: person detection, person tracking and person retrieval. This project will mainly focus on the person retrieval module, and the person detection and person tracking parts are assumed to be well handled and beyond the scope of this project. Besides, the project will focus on the short-term image-based person re-ID rather than video-based one or the long-term one. The main techniques that underpin this project include:

- feature extraction;
- convolutional neural network;
- generative adversarial network;
- re-identification.

In section 3, the literatures that are closely related to the mentioned techniques will be critically reviewed and part of the evaluated methods will be applied to the project, which will be declared later. It is worth mentioning that the project will not explore mathematics theories too much since it is more about applications. This review aims to investigate the relevant techniques and give a comprehensive evaluation of them. Some appropriate methods will be adopted to achieve better performance on this project. The improvements mainly focus on the application level including model reconstruction and data augmentation rather than mathematics level.

3 Literature Review

3.1 Deep Neural Network

The neural network becomes increasingly popular recently due to the development of GPU hardware and parallel computing. Actually, it has a long history since 1943. At that time, the neural network was inspired by human brains and used neurons to simulate the signal propagation in our brain. However, with the development of backpropagation, the relationship between neural network and biology turned out to be less tight. Currently, we mainly regard neural networks as function approximators and use them to bridge a map from inputs to outputs. Its capability of approximating different functions mostly depends on its design and architecture. A typical neural network is implemented as a chain of matrix multiplications and element-wise non-linearities.

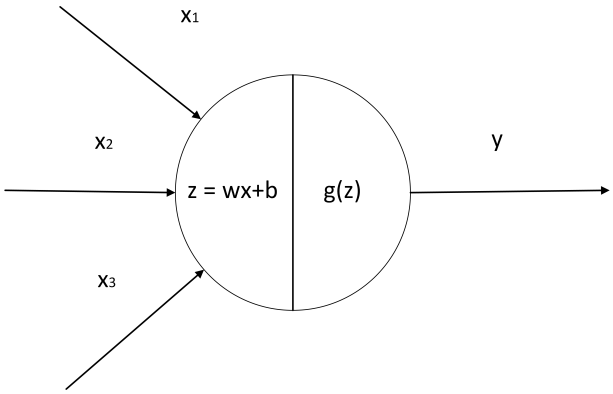
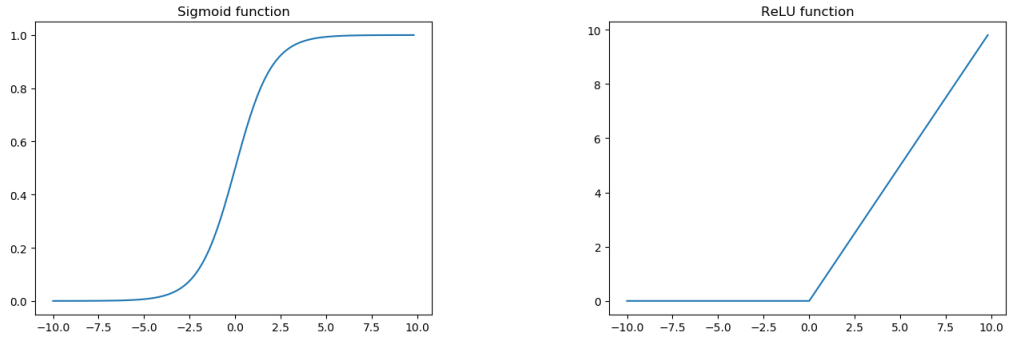


Figure 1: A single neuron

A simple one hidden layer neural network (see Figure 3) containing a finite number of neurons can approximate any continuous function according to universal approximation theorem [8]. The architecture of the neuron is shown in Figure 1. With a number of input values x, a neuron operate it with formula $z = wx + b$, where w is a matrix of parameters and b is a bias value. Then z is passed to an activation function g to output the final value $y = g(z)$. The common choices of the activation include sigmoid and ReLU [37]. To approximate a complex function, the hidden layer will be quite large to contain more neurons. Accordingly, deeper neural network with more than one hidden layer can be applied to approximate particular functions. The network with more than one hidden layer is called multilayer perceptron. Currently, DNN can have hundreds of layers for various applications. Normally this kind of network is hard to train mainly due to the vanishing gradient problem. To avoid this problem, batch normalisation [19] and ReLU [37] activation function are utilised. ReLU is a kind of function with a form of $max(0, x)$. Because it does not saturate, vanishing gradient problem with sigmoid activations can be avoided. Besides, with the help of ResNet [17] architecture, degradation problem

has also been well addressed with identity shortcut connections and the performance can be improved with the growth of layers.



(a) Sigmoid function (b) ReLU function

Figure 2: Two different activation functions

Convolutional Neural Network (CNN) is a variation of multilayer perceptron which uses many convolutional layers and pooling layers in the architecture. Mathematically, the convolution operation is a cross-correlation procedure. Also, it is inspired by biological processes in that the connection pattern between different neurons resembles the organisation of the animal visual cortex. Individual cortical neurons only respond to a restricted region of the visual field, which is called the receptive field. The receptive fields of different neurons partially overlap to make sure they cover the entire visual field without losing information. [9] The size of receptive field depends on the number of convolutional layers and their kernel sizes. With this architecture, it is very efficient to analyse visual imagery input. The pooling layer is usually utilised to reduce spatial information with an operation in a region of fixed size, including *max* and *average*. After pooling operation, the dimensions of the input, as well as the number of parameters, will be decreased, thus to gain computational performance and have less chance to over-fit.

To some extent, person re-ID task aims to find a projection from input images to output ID labels, which can be achieved by a complex function. Therefore, DNN can be implemented to person re-ID [6, 7, 28, 30, 45, 46, 53, 55, 56, 59]. Due to the excellent performance of the CNN model on computer vision tasks, we adopt it as our baseline model.

3.2 Re-identification

Commonly, the standard scenario for re-identification is: (i) find the area of interest from the images of surveillance sensors (cameras *et al.*); (ii) generate a representation for the area of interest of every image. Some pre-processing for the images may

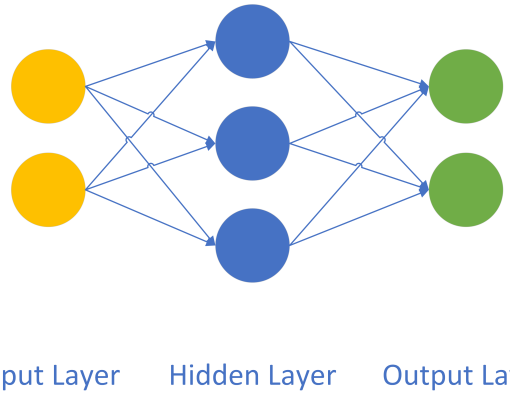


Figure 3: A basic neural network

be needed; (iii) Match the representation with a suitable method to tell which two representations are from the same object.

The first stage is usually called object detection and exists in every re-identification system. Dollár *et al.* [10] researched in this area and is worth to have a read. Object detection is usually ignored for re-identification and the primary work focus on remain two stages, described as feature extraction and distance metric learning.

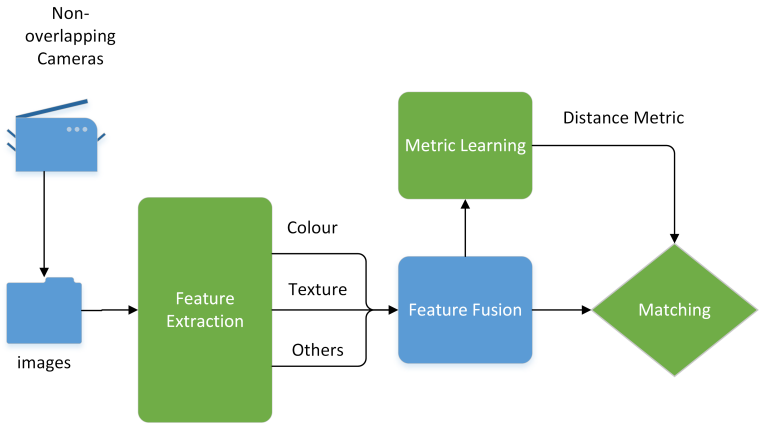


Figure 4: Basic steps for a person re-identification system

With the development of GPU computing, deep learning achieved very extraordinary performance in many areas. The architecture of re-identification based on deep learning is actually different from the one mentioned above.

When it comes to person re-ID, Zheng *et al.*[58]'s survey about the past, present and future of person re-ID gives a clear description of the development and progress in this area. Basically, person re-ID systems can be divided into two categories: human-engineered person re-ID [3, 5, 21, 25, 34, 39, 51, 52] and deep-learning-based person re-ID[6, 7, 28, 53, 55, 56, 59].

3.3 Human-engineered Person Re-ID

3.3.1 Feature Extraction

The first step for human-engineered person re-ID focuses on searching for a stable feature representation which is invariant to illumination, camera rotation, occlusion and so on. Moreover, for many computer vision task, this is an essential procedure. We hope the representation obtained can accurately represent the pedestrians' characteristics.

Currently, many different types of features such as colour, edge, shape, gait feature and so on have been proposed for person re-ID. These low-level features can be extracted from the whole bounding box of pedestrians, or divide the bounding box by body parts and extract the features separately, which are called global features and local features respectively.

To cope with the lack of datasets and some visual limitations, most of the re-ID systems are organised by extracting and combining some complementary features to work well. Usually, each visual feature is expressed in the form of a histogram using the bag-of-words [16] scheme. The compact feature representation is then formed by applying weights to connect these features, and the weights of each feature depend on their importance. Spatial information on the layout of the original images is also of great importance, for instance, the body parts. In order to blend spatial information into the representation of features, the images are usually divided into different regions or sections, from which feature extraction can be performed separately.

Concerning the low-level hand-extracted features, colour information is crucial and the most common choice since it is easy to collect and deal with. Generally, colour information will be encoded into a colour histogram [25] with different colour spaces (RGB, HSV, YCbCr). Since RGB space is very sensitive to light condition, it is seldom used individually. Hue channel in HSV space is proved to be relatively invariant to illumination. Despite that, colour is prone to all kinds of noise from the environment and can be unstable under some circumstances. For example, different people may wear similar colours in clothes and vice versa. For this project, colour cannot represent the characteristics of pedestrians effectively. Since colour is not discriminative enough, other low-level features such as shape and texture feature [5, 12, 21, 39] are usually combined with colour to form a more robust representation.

In [15], Gray and Tao combined three different colour channels (RGB, HSV, YCbCr) and two texture features (Gabor [12], Schmid [43]). This feature representation combines lots of information in it and is also used in Prosser *et al.*'s work [39]. Nevertheless, this feature gives red and blue channels roughly the same weights but significant than the weight of green channel, which means the performance of re-ID will be influenced if many people in surveillance area are wearing green-like clothes.

Apart from global features, some local features are also be explored considering the biological inspiration. In [5], Farenzena *et al.* proposed an approach called the symmetry-driven accumulation of local features (SDALF). First, the human body is divided into head, torso and legs by two horizontal axes. Also, a vertical symmetric axis is estimated for the last two body parts. At last, three different representations are calculated on each section: (i) general chromatic content formed by HSV colour histogram; (ii) colour displacement for each region via Maximally Stable Colour Regions (MSCR) [13], thus to eliminate the influence of pose variation. Although this method can enrich the feature representation and prune out the background clutter easily, it is not appropriate to use it in a DNN model.

In [3], Bak and Brémond used covariance descriptor as feature representation. Images are divided into different regions. The descriptor represents feature variances inside an image region, correlations between adjacent regions and spatial information. The descriptor is then averaged on a Riemannian manifold. With this descriptor, different types of feature can be combined into a single representation. Moreover, since the covariance matrix can absorb rotation and illumination variances, it is more robust than many other methods. However, this method is extremely computationally expensive and not appropriate for a real-world application.

Depth feature has been tested a lot due to the popularity of Microsoft Kinect which was produced in November 2010. Kinect has two different cameras and can capture RGB image and depth image simultaneously. In [51], Wu *et al.* use covariance descriptor and skeleton feature together to perform person re-identification. The so-called depth voxel covariance descriptor and the eigen-depth feature can be used to describe human body shape and have rotation invariance. Since shape and skeleton information is included in feature representation, it is proved to be robust to illumination and colour change, which is superior to other methods for long-term surveillance. Because of the expensive computational cost of covariance matrix distance in a Riemannian manifold, this paper also proved that this matrix distance is equivalent to the Euclidean distance of eigen-depth feature which extracted from the eigenvalues of covariance matrices. With this theory, the computational cost can be eliminated to a large extent. It is worth mentioning that this paper uses direct distances of skeleton structure which is sensitive to camera rotation. A superior way is to measure the relative distance. Besides, considering that RGB-D cameras have yet to be widely applied in many surveillance systems, and it will perform much worse when the sunshine is powerful, this approach is difficult to deploy in the real world.

I have researched skeleton features for person re-ID during my Bachelor's degree. Some examples of the skeleton feature are shown in Figure 5. All these features are extracted from the relative skeleton distance, which is invariant to lighting, colours and camera views.

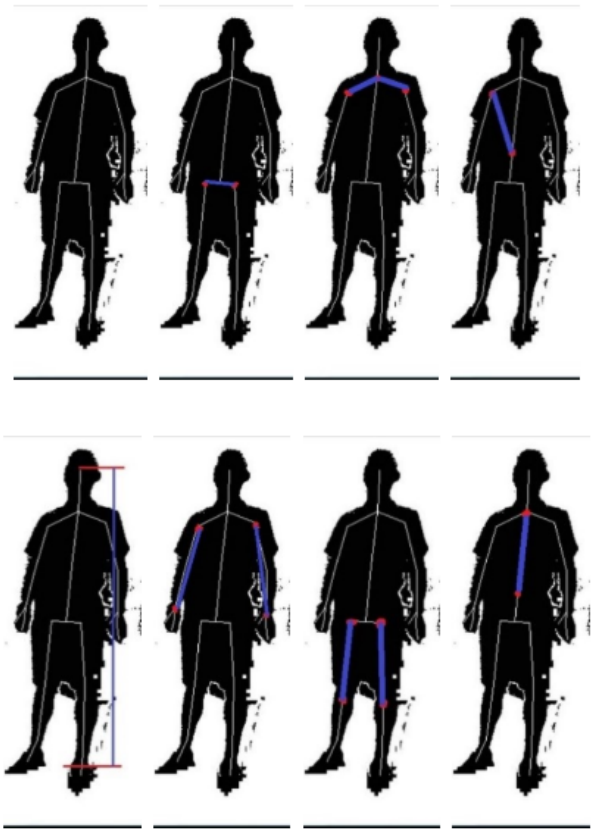


Figure 5: The examples of skeleton features

Many other hand-crafted features including Fisher vectors [34], spatial symbiosis [48], and SARC3D [4] are also widely applied in person re-ID to characterise more robust and stable features. This project concentrates on the deep-learning-based model, which generally take images as input without extracting their low-level-features. However, these low-level features can be combined with DNN to achieve better performance [52]. This project mainly focuses on the utilisation of the deep neural network, and deep features are proved to be more robust and discriminative than low-level features. Therefore, these features will not be explored.

3.3.2 Distance Metric Learning

A distance metric is a function which defines the distance between a pair of elements. Distance metric learning in person re-ID aims to learn a distance metric that can make the distance of images from the same pedestrian smaller and the distance of images from different individuals bigger.

At present, many proposed algorithms use a specific pre-set metric in the feature space. In fact, different matching problems can get benefits from using dedi-

cated metrics to enhance the relevant characteristics of the targets that need to be matched. This method attempts to learn a distance metric such that the distance between descriptors of different individuals is maximised and the distance between descriptors of the same individual is minimised. With such metric, the match and identification work will be much easier to achieve, and the performance can be improved a lot.

Basically, it is more difficult to learn an appropriate distance metric than finding a discriminative descriptor [58], and the characteristics to be emphasised for different problems are dissimilar from each other. Additionally, there are requirements on the number of samples to learn a useful distance metric thoroughly.

The most popular metric learning algorithm so far is KISSME (keep it simple and straightforward metric) [24]. KISSME is a Mahalanobis metric learning method which aims to learn a metric from equivalence constraints based on a statistical inference perspective. In general, a Mahalanobis distance metric gives the squared distance between two images:

$$d_M(Q, G) = \sqrt{(Q - G)^T \mathbf{M} (Q - G)} \tag{2}$$

where M is a positive semi-definite matrix. A likelihood ratio test is applied to determine whether an image pair belongs to the same ID or not. Moreover, the principal component analysis (PCA) is applied to the feature descriptors of images for dimension reduction since the original descriptor this paper used is a $3,465$ dimensional vector, which is intractable for distance metric learning. KISSME can handle large-scale data without complicated optimisation problems which require expensive computations.

In [50], a large margin nearest neighbour (LMNN) method which tries to improve k nearest neighbours (KNN) classification is proposed to learn a pseudometric under which that all the samples in the training set are surrounded by at least k data points that have the same ID label, which is called the target neighbours. An imposter of a data instance x_i is an instance with a different label, which is also one of the nearest neighbours. The learning algorithm aims to minimise the number of imposters for all data instances in the training set. Overall, this approach is simple but requires complex optimisation procedures to improve the classification error.

Some works are interested in learning a discriminative subspace for the matching procedure rather than learning a specific distance metric. In another word, a projection w from the original feature space to a low-dimensional subspace will be applied to the features. In practice, PCA is widely applied for the dimension reduction, and then the metric learning is performed in the PCA subspace.

In [40], Prosser *et al.* regard person re-identification as a ranking problem and the learning of subspace is formulated by applying RankSVM, which is a variant of the

support vector machine (SVM). In this algorithm, a set of weak RankSVMs are learned and combined into a stronger ranker via ensemble learning, thus to make the RankSVM tractable for the large-scale problem.

In [35], the pairwise constrained component analysis (PCCA) is proposed which can be directly applied to the data that only a restricted number of similarity information of pairs of high-dimensional data points is available. It does not require extra assumptions on the structure of the data, and additionally, it can handle the high-dimensional input without a step of dimension reduction. Thus, the original feature information remains.

Rather than using PCA to reduce the dimension of the original feature vector, in [32], Liao *et al.* propose a method called cross-view quadratic discriminant analysis (XQDA) to learn a discriminant low-dimensional subspace, and simultaneously, a QDA metric is learned in the derived subspace. The training data is from different views, for example, different cameras in person re-ID. This algorithm is similar to linear discriminant analysis (LDA) [36], and can be regarded as an extension of KISSME. Since this method can learn the subspace and distance metric simultaneously, it is more effective to apply PCA and learn a metric separately, though, in real applications, the selection of dimensionality of derived subspace is a problem to balance computation speed and performance. XQDA is appropriate to deal with person re-ID task since most of the re-ID situations are precisely under non-overlapping cameras and the data is from different views.

Although these metric learning methods achieved real competitive results with the hand-extracted features, they become less significant when the deep-learning-based person re-ID play an essential role. Based on some existed works, it is unnecessary to apply a feature mapping for the deep features. Thus the metric learning methods will not be explored in this project.

3.4 Deep-learning-based Person Re-ID

Besides human-engineered person re-ID system, deep-learning-based methods become increasingly popular after the successful application in the area of image classification, especially the Convolutional Neural Network (CNN). Also, in 2014, some researchers including Li [28] and Yi [55] began to apply the deep neural networks to person re-ID. At that time, the volume of the most re-ID datasets are relatively small, so the CNN model for image classification is not proper for the application in person re-ID. On the contrary, the siamese neural network[6] model, which uses image pairs as input, is a better choice. This is an end-to-end manner in which the matching results can be directly obtained from the model output. There is no other dedicated procedure to match the images.

Siamese neural network [6] was proposed by Bromley *et al.* to deal with signature verification. It consists of two sub-networks, and they are joined at the output layer. Two sub-networks extract features from their inputs while the joining neuron measures the distance between the two feature representations. A threshold is selected to determine whether the two inputs belong to the same category or not. Usually, the two separate networks share the same weights. With this model, the person re-ID task is regarded as a classification process rather than a retrieval process.

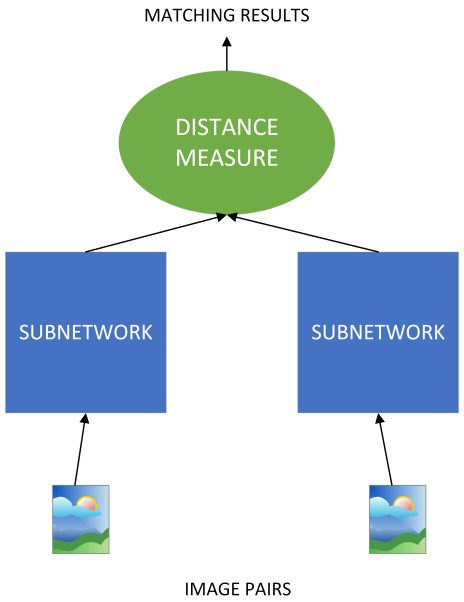


Figure 6: The Structure of Siamese Model

In [55], Yi *et al.* proposed a deep metric learning method for person re-ID. In traditional person re-ID task, feature extraction and metric learning are processed separately. With a siamese deep neural network [6], colour feature, texture feature and metric can be learned in a single framework jointly. The two sub-networks are constructed with 2 convolutional layers, 2 max-pooling layers and a fully-connected layer.

In [28], Li *et al.* proposed a method which used a novel filter pairing network (FPNN) to jointly handle misalignment, photometric, geometric transforms, occlusions and background clutter problems, which are ubiquitous challenges in existing person re-ID task. The most prominent bottleneck of deep learning is the datasets are too small for training a deep model efficiently. This paper also proposed a new re-ID dataset: CUHK03, which includes 13164 images of 1360 pedestrians and provides automatically detected bounding boxes for every pedestrian in the images.

These methods all assume that bounding boxes are given for the datasets, which is unrealistic for real-world applications. In [59], Zheng *et al.* used raw video frames

as the original data and applied some pedestrians detectors to these frames to get the bounding boxes of pedestrians. Various compounds of detectors and recognisers were tested. They also proposed a new dataset called PRW to evaluate **P**erson **R**e-identification in the **W**ild. With the help of pedestrian detection algorithm, the overall performance of person re-ID can be improved to some extent. This project will not explore pedestrian detection methods due to the limited performance improvement.

Later on, plenty of methods based on neural networks have been proposed, but still mainly focus on siamese models with image pairs as input [45, 46]. In [7], Cheng *et al.* proposed a model using three images as input. More specifically, a single network with multiple channels to learn the global full-body and local body-parts features simultaneously is constructed. So an improved triplet loss function is used to make the distance between images of same pedestrian smaller and the distance of different pedestrian bigger.

All of these siamese models have a drawback: the ID labels of pedestrians cannot be fully used since the models only care about the matching results of the image pairs.

Rather than using end-to-end siamese neural networks, classification models can also be applied due to the growth of the size of datasets recently. For instance, MARS [56] contains $1,191,003$ images of 1261 identities from 6 non-overlapping cameras. For a classification neural network model, it is similar to traditional human-engineered person re-ID system. The output of the model is regarded as a deep feature of image input. Then metric learning can be applied to it. In [53], features are learned from multiple datasets of different domains jointly and a softmax loss function is used for classification. It is proved that the classification model outperforms the siamese model given sufficient training data. Besides, a domain guided dropout (DGD) is added. The proposed dropout scheme consistently improves the performance on all the domains, especially on the smaller-scale ones.

In this project, the scheme of dropout will be explored to improve the performance of the baseline model and avoid over-fitting. Besides, some data augmentation techniques including random cropping, random erasing [62] and so on are utilised to improve its performance further.

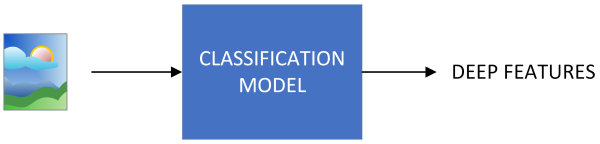


Figure 7: The Structure of Classification Model

Recently, many proposed methods are based on some pre-trained deep neural networks from ImageNet [9] data and fine-tuning them for different purposes. These

methods take images (usually image pairs or triplets) as input directly. With a pre-trained model, time will be saved a lot due to the free of training neural networks, which is quite time-consuming though the performance may be influenced. Besides, for similar tasks, a pre-trained model can provide well-extracted low-level features for the input images. Due to the time limitation, the model utilised in this project is also pre-trained on ImageNet [9], which is provided by PyTorch.

Instead of employing image input, in [52], Some low-level features including colour histograms and scale-invariant feature transform (SIFT) feature are encoded into a Fisher Vector as the input of a neural network. In this project, low-level features will not be used since it is hard to compare their performance and the improvement is limited.

Spatial information is proved to be a discriminative clue for pedestrian recognition. Thus, exploring part-level features for pedestrian image description can offer fine-grained information and has been confirmed as beneficial for person retrieval in recent literature [54]. In [44], Sun *et al.* developed a network named part-based convolutional baseline (PCB). With a single image as input, this network can output a descriptor consisting of 6 part-level features. A refined part pooling (RPP) approach was proposed to re-assign the outliers of each part from uniform partition to the parts they are closest to, which would result in refined region partition with improved within-part consistency. The visualisation of the refined part result is shown in Figure 8. This model achieved a very competitive performance on some popular public datasets. Also, another part-based method [22] achieved state-of-the-art results on DukeMTMC-reID [60], which indicates that local features from human body parts are critical for person re-ID task.

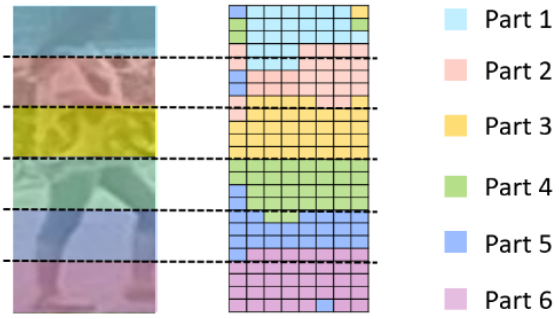


Figure 8: Visualisation of partition scheme in [44]

Attention mechanism is becoming increasingly popular in the field of deep learning. Normally, a neural attention mechanism can equip a neural network with the ability to only focus on the part of its inputs, which means it is capable of selecting specific inputs to deal with. More specifically, assuming $x \in \mathbb{R}^d$ is an input vector, $z \in \mathbb{R}^k$ is a feature vector, $a \in [0,1]^k$ is an attention vector, $g \in \mathbb{R}^k$ is an attention glimpse and

$f_\phi(x)$ is an attention network. Formally, an attention mechanism is implemented as,

$$a = f_\phi(x),\ g = a \odot z \tag{3}$$

where $\odot$ is an element-wise multiplication, and z is the output of another neural network. Typically, attention can be divided into two categories: soft attention and hard attention. Soft attention means that the elements in a are between zero and one, while in hard attention, these values are constrained to be exactly zero or one. In person re-ID task, attention selection mechanism is always utilised to calibrate misaligned images, which is illustrated in Figure 9. Li *et al.* [31] proposed a new model for jointly learning attention selection and feature representation in a single CNN model. This model can effectively eliminate the influence of misaligned images while learning a discriminative feature representation simultaneously with an end-to-end manner. We currently focus on the usual CNN model and assume the pedestrian bounding boxes are well-aligned in the datasets we are using. Thus the attention mechanism will not be explored in this project.

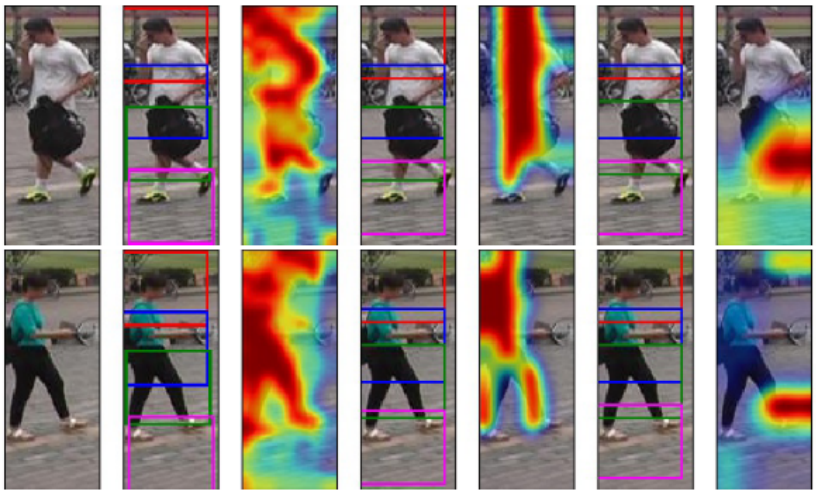


Figure 9: Attention selection in auto-detected pedestrian bounding boxes [31]

3.5 Improving Approaches

Rather than doing research on proposing new models for person re-ID, some papers mainly focused on the methods to improve the performance of existed re-ID systems. In [60], Zheng *et al.* created a semi-supervised pipeline for person re-ID that use GAN [14] to generate more samples from the original dataset. A label smoothing regularisation for outliers (LSRO) method was conducted to assign labels to the generated unlabelled images. The authors assumed that all the synthetic samples did not belong to any existed classes, and a label for a generated image was sampled from a uniform distribution over all new classes. DCGAN [42], which is based on

convolutional structure, is adopted for sample generation. This project will further explore the effective improvements by GAN-generated samples considering the relatively small data volume of the public person re-ID datasets. To address the stability of GAN training, we use Wasserstein GAN [2] as the generative model, which is proved by the authors to be more stable. More details will be given in section 4.

For computer vision task, to improve the generalisation ability of CNN models and avoid over-fitting, occlusion is always a critical factor to deal with. Zhong *et al.* proposed a new data augmentation method called random erasing to exhibit variant occlusion in training samples. Random erasing is easy to implement and has no parameter to learn. Besides, it is complementary to existed data augmentation and regularisation techniques, including random cropping, flipping, dropout [53] and batch normalisation [19]. A randomly-selected rectangle region was assigned with random values or mean pixel values. Thus, augmented images with various occlusion levels can be generated to improve the performance of the existed models. Since the datasets we will use are all obtained with widely open cameras, they exhibit limited variance of occlusion, and random erasing will be adopted as an improvement technique to avoid over-fitting further.

Apart from data augmentation, considering person re-ID as an image retrieval task, re-ranking can be a critical technique to improve its performance effectively. In [61], Zhong *et al.* proposed a re-ranking method by encoding k-reciprocal nearest neighbours [20] into a feature vector, and calculate the Jaccard distances between these features. The final distances were formed by aggregating the original distances and Jaccard distances. A local query expansion approach was developed to obtain a more robust k-reciprocal feature. After re-ranking, the mAP of person re-ID can be improved significantly, and the rank-1 accuracy will achieve a small increase. This project will encode the re-ranking into our re-ID model since it is easy to implement and requires no parameter learning. More details about re-ranking will be provided in section 4.

Usually, different datasets are from different distributions, which brings domain gaps between them. This means that when training and testing one model on different datasets, a severe performance drop is unavoidable. On the other hand, the domain gaps will make the available training data cannot be effectively leveraged for new testing domains, and lose the chance to utilise existing data effectively. To deal with this problem and relieve the expensive costs of annotating new training images, Wei *et al.* [49] developed a Person Transfer Generative Adversarial Network (PTGAN) to bridge the domain gaps between different datasets efficiently. This is another application of GAN in person re-ID besides data augmentation: image-to-image translation, which tries to learn the mapping between the original input image and the output target image. Usually, the gap between different datasets is caused by variant lighting conditions, resolutions, backgrounds and so on. PTGAN aims to

transfer pedestrians in dataset A to dataset B by keeping their identities while presenting them in a similar style (lighting, backgrounds, *etc.*) to dataset B. An illustration from the original paper is provided in Figure 10, where the images in the first row are from DukeMTMC, and the remains are transferred from Market1501 to DukeMTMC. Obviously, the images have the similar style, which proves the efficiency of PTGAN model. Although this model can significantly eliminate the domain gaps between different dataset, the performance is still in a low level when compared with other state-of-the-art methods.

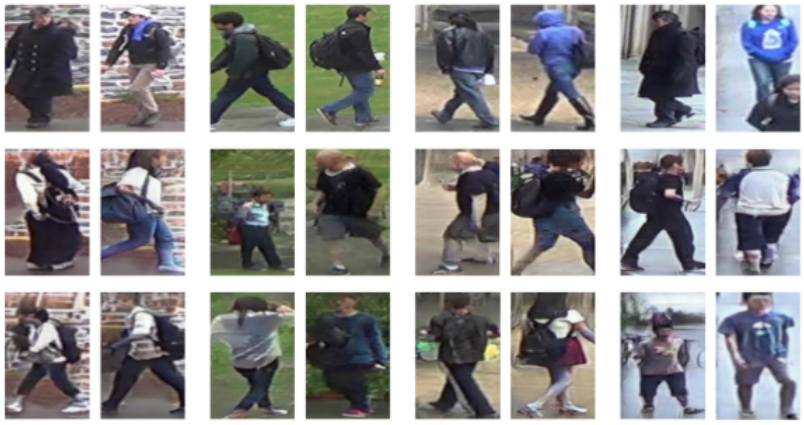


Figure 10: Image transferring from Market1501 to DukeMTMC [49]

3.6 Datasets

There are many public datasets have been proposed specifically for the task of person re-ID. Collecting data for the training of deep neural network is time-consuming work. For convenience, this project will be evaluated on these public datasets. The most popular one is VIPeR [15], which contains 1264 images of 632 pedestrians. The images are obtained from two non-overlapping cameras. Although this dataset has been tested for many times, it is still a challenging dataset. Also, since this dataset is not big enough, deep learning based methods cannot make their full use due to the lack of training data. Other main datasets [1] are shown in Table 1. These datasets were obtained from all kinds of public areas. For example, the GRID [33] dataset was obtained in a station and the iLIDS [47] was collect at an airport, so most of these pedestrians may carry a suitcase. The Market1501 dataset is collected in front of a supermarket in Tsinghua University. The CUHK01 [27], CUHK02 [26] and CUHK03 [29] were all captured in the Chinese University of Hong Kong, and the pedestrians in these datasets appear to wear a backpack with a large probability.

[1] This information is from http://robustsystems.coe.neu.edu/sites/robustsystems.coe.neu.edu/files/systems/projectpages/reiddataset.html

Dataset	Release Time	Identities	Cameras	Images	Label Method
VIPeR	2007	632	2	1264	Hand
iLIDS	2009	119	2	476	Hand
GRID	2009	1025	8	1275	Hand
CAVIAR	2011	72	2	1220	Hand
3DPeS	2011	192	8	1011	Hand
PRID2011	2011	934	2	24541	Hand
CUHK01	2012	971	2	3884	Hand
CUHK02	2013	1816	10	7264	Hand
CUHK03	2014	1467	10	13164	Hand/DPM
RAiD	2014	43	4	6920	Hand
Market1501	2015	1501	6	32217	Hand/DPM
PRW	2016	932	6	34304	Hand
MARS	2016	1261	6	1191003	DPM+GMMCP

Table 1: Person Re-identification Datasets

As shown in Table 1, the size of the datasets is increasing significantly. In the earlier years, datasets are relatively small. In recent years, datasets such as CUHK03 and MARS all have a large number of images which provide enough training samples for a deep learning re-ID model. Figure 11 shows some samples in CUHK03 dataset.

Besides, the bounding boxes of pedestrians were usually produced by hand in most of the datasets. However, recently some datasets such as CUHK03 began to use pedestrian detection algorithm (DPM [11]) to draw the bounding boxes automatically. It is vital to avoid the time-consuming hand-based process in real-world applications.

Figure 11: Samples of CUHK03 [29]

3.7 Evaluation Methods

Currently, the conventional evaluation methods for person re-ID are Cumulative Match Characteristic (CMC) curve and mean average precision (mAP) [57]. Therefore, this project will choose them as the performance analysis methods to evaluate the recall of our model considering there are usually more than one ground truths in the gallery images.

3.7.1 CMC Curve

In CMC curve, each probe sample will be compared against all gallery samples, and the resulting score will be sorted and ranked. CMC curve reports the accuracy of top k sorted samples. An example CMC curve is shown in Figure. The first point of CMC curve is called rank-1 accuracy. Although there may be more than one matched samples, only the first matched one is counted. Under this circumstance, the recall is not considered, and we need another evaluation method.

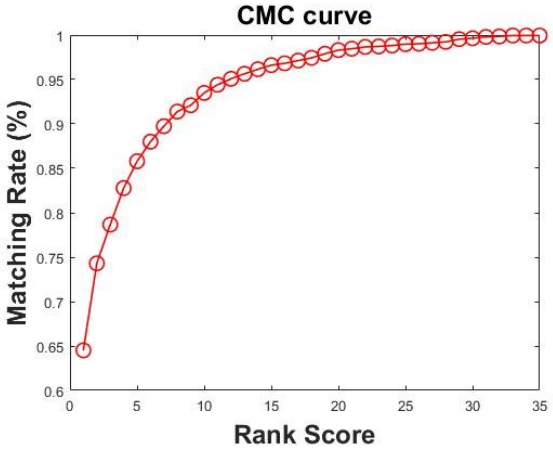


Figure 12: CMC Curve Example

3.7.2 mAP

In [57], mean average precision was introduced to evaluate person re-ID system performance for the consideration of finding all matched samples. For each probe sample, average precision (AP) is calculated. Then the mean of APs of all probe samples (mAP) is calculated, thus the precision and recall are both considered. For datasets which have more than one matched images for every query image, mAP is evaluated together with CMC curve to make a better performance evaluation.

3.8 Summary

In this section, a comprehensive review of relevant literature that underpins person re-ID task is provided. Many state-of-the-art methods for person re-ID have been critically analysed, with evaluations of their pros and cons. Besides, brief discussions on how applicable they are to this project are presented to indicate which methods will be actually implemented. The main public datasets information is given to show a comparison. At last, evaluation methods that will be used in the project are discussed.

In this project, the classification CNN model is utilised to conduct the feature extraction task, and Euclidean distance is adopted to measure the distance of the features. Two data augmentation methods are adopted: Wasserstein GAN [2] is used to enlarge the training set, and random erasing [62] is used to generate some levels of occlusion. Market1501 [57] and DukeMTMC-reID [60] are selected as the evaluation datasets due to their popularity and relatively large volume, and the evaluation is based on CMC curve and mAP.

4 Project Execution

4.1 Overview

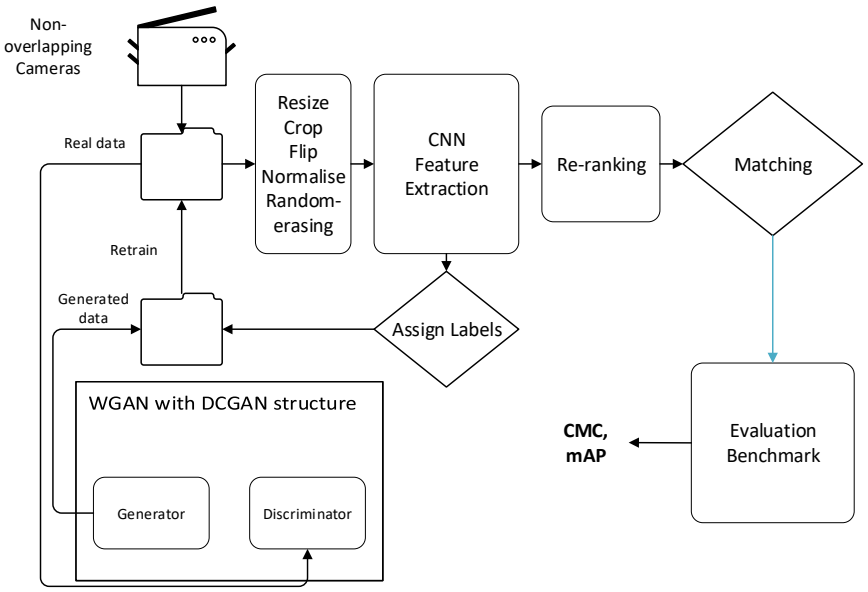


Figure 13: The structure of proposed person re-ID system

The structure of this project is shown in Figure 13. The details of every module will be provided later. Normally, to regard a person re-ID task as a retrieval process, there will be some query images and gallery images from different non-overlapping cameras. The main task is to rank the gallery images based on their similarity with the query image. Thus, our re-ID result would be a ranking list with respect to the query input image. The main steps of our project are:

1. Use the pre-processed original datasets to train the baseline model;
2. Use the original datasets to train the Wasserstein GAN;
3. Generate synthetic samples with the trained generator from Wasserstein GAN;
4. Assign labels to the generated samples with the trained baseline model;
5. Add the labelled synthetic samples to the training set;
6. Retrain the baseline model with the enlarged training set;
7. Use the retrained model to extract the features of query and gallery images;
8. Use Euclidean distance to rank the gallery images with respect to different query images;

9. Adopt re-ranking to optimise the ranking results;
10. Use CMC curve and mAP to evaluate the ranking results.

To our knowledge, this is the first try to adopt Wasserstein GAN [2] as a generative model for person re-ID task. Also, we will prove that our new label assignment method is superior to the existing methods.

4.2 Datasets Used in This Project

In section 3, many public person re-ID datasets have been introduced. Unfortunately, not all of them are large enough to train a deep neural network considering that it usually requires many training samples. Market1501 [57] and DukeMTMC-reID [60] are selected for their relatively large volume and the popularity among person re-ID tasks.

4.2.1 Market1501

Recall that Market1501 is collected in front of a supermarket in Tsinghua University. There are 6 different cameras, including five high-resolution cameras and one low-resolution camera. It contains 32,668 annotated bounding boxes of 1,501 identities, where 12,936 of them are used for training.[2] Some sample images from different cameras are shown in Figure 14.

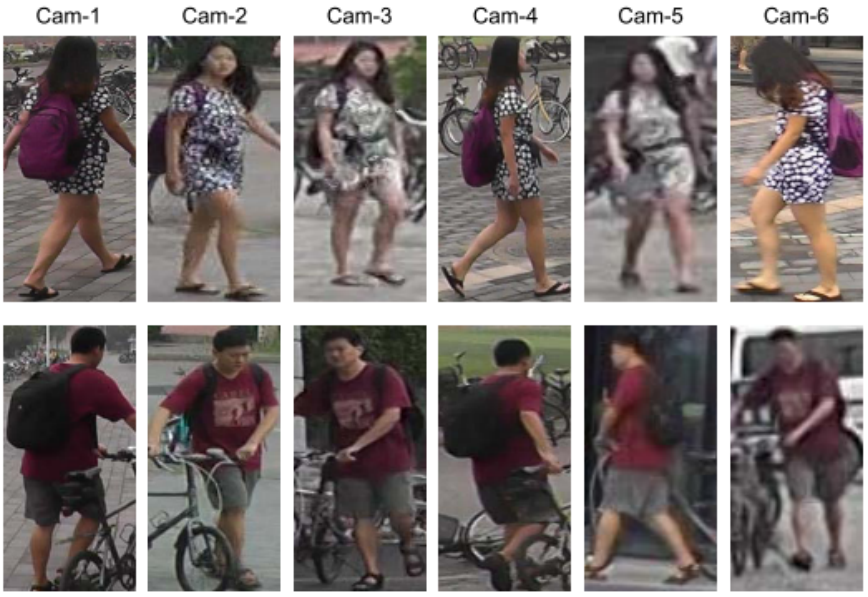


Figure 14: Sample images of Market1501 [57]

[2]The information is from http://www.liangzheng.org/Project/project_reid.html

4.2.2 DukeMTMC-reID

DukeMTMC-reID is a subset of DukeMTMC, which is a manually annotated dataset recorded outdoors on the Duke University campus with 8 synchronised non-overlapping cameras. It contains 36,411 annotated bounding boxes of 1,404 identities, where 16522 of them are used for training.[3] Some sample images are provided in Figure 15. In this project, we will use the terms DukeMTMC and DukeMTMC-reID interchangeably.

Figure 15: Sample images of DukeMTMC-reID [60]

4.3 Dataset Enlargement

Deliverables: An extended and redistributed version of the original Wasserstein GAN [2] as long as 12,000 generated images. The code is available on GitHub for future researchers. (redistribution of Wasserstein GAN)

As mentioned before, both Market1501 and DukeMTMC-reID have less than 20,000 training samples. For a deep learning task, the data volume is relatively small. Thus, in this project, Wasserstein GAN [2] is adopted as the generative model for synthetic sample generation. Also, a new label assignment method is proposed since the generated images are unlabelled.

4.3.1 Generative Model

Typically, to form a generative model, we need to learn a probability distribution, and the classic solution is to learn a probability density. Usually, a parametric family of densities $(P_\theta)_{\theta \in \mathbb{R}^d}$ is defined and the one that can maximise the likelihood of the

[3]The information is from http://vision.cs.duke.edu/DukeMTMC/

data the density we need. If the real data distribution $\mathbb{P}_r$ has a probability density and $\mathbb{P}_\theta$ is the probability distribution of the parametrised probability density P_θ, this is equivalent to minimise the Kullback-Leibler divergence $KL(\mathbb{P}_r||\mathbb{P}_\theta)$.

Under some circumstances, the density P_θ may do not exist. To address this issue, Gaussian noise with relatively high variance is added to cover all the examples. However, the added noise will degrade the quality of the samples, which means it is incorrect for the problem but is necessary to conduct a maximum likelihood on the model.

An alternative way is to define a random variable Z with a fixed probability distribution $p(z)$ and map it through a parametrised function $g_\theta : \mathcal{Z} \to \mathcal{X}$ that can directly generate data which follow a specific distribution $\mathbb{P}_\theta$ without estimating the density of real distribution $\mathbb{P}_r$. Usually, the function is a neural network. For the consideration of image superresolution or data augmentation, the ability to generate synthetic samples is often more important than finding the numerical value of the probability density.

4.3.2 Wasserstein GAN

GAN [14] and Variational Auto-Encoder (VAE) [23] are both famous examples of the method mentioned before. VAE mainly draws attention to the approximate likelihood of the samples, thus share the limitation of the first method mentioned before. GAN is more flexible with Jensen-Shannon distance as the measurement of two different distributions. Intuitively, GAN is implemented by a system of two networks (generator and discriminator) contesting with each other. The generator network takes random numbers as input and is trained to output samples that can cheat the discriminator, and the discriminator is optimised to tell an input sample is synthetic or real. However, the training of GAN is unstable due to some reasons that are well investigated in [1]. Wasserstein GAN was developed to deal with the many problems remained in the training of GAN by directing attention on the various methods to measure the distance between the real distribution and model distribution.

More specifically, the Earth-Mover (EM) distance

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \prod(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{(x,y)\sim\gamma}[\ \|x - y\|\ ] \tag{4}$$

is utilised to measure how close of the real distribution $\mathbb{P}_r$ and the model-generated distribution $\mathbb{P}_g$, where $\gamma \in \prod(\mathbb{P}_r, \mathbb{P}_g)$ denotes the set of all joint distributions $\gamma(x, y)$ whose marginals are respectively $\mathbb{P}_r$ and $\mathbb{P}_g$. $\gamma(x, y)$ represents how much mass must be transported from x to y in order to transform from the distributions $\mathbb{P}_r$ to the distribution $\mathbb{P}_g$. The EM distance measures the effort of the optimal transformation

plan. Wasserstein GAN is trained to optimise the EM distance.

With the EM distance as the objective function, this project applies DCGAN [42] as the structures of generators and discriminators. To take $64 \times 64 \times 3$ images as input, the discriminator has 5 convolutional layers, and a batch normalisation [19] layer follows every one of them before the ReLU [37] activation function. The first 4 convolutional layers are configured as **kernel size = 4, stride = 2, padding = 1**, while the last one is constructed as **kernel size = 4, stride = 1, padding = 0**. With this structure, the discriminator outputs a single value that represents whether the input image is real or not. When it comes to the generator, there are 5 transposed convolutional layers which transform the 100-dimensional random input to the output of $64 \times 64 \times 3$ images.

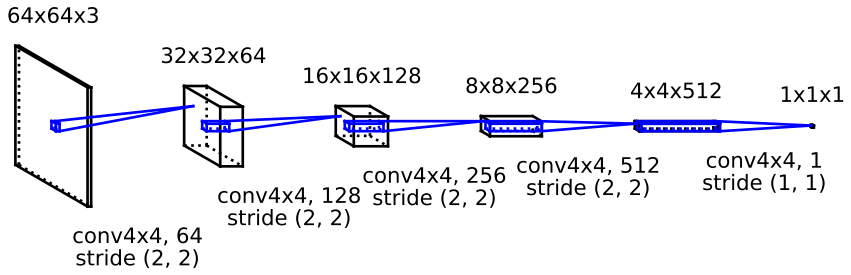


Figure 16: The structure of discriminator

4.3.3 Training of Wasserstein GAN

The training procedures are as follows:

1. Train the discriminator on the real data for 100 iterations;
2. Generate 100-dimensional fake random inputs for the generator and train the discriminator on fake data;
3. Train the generator with the output of the discriminator;
4. Repeat step 1 to 3 for 4000 epochs.

It is worth mentioning that the after 25 iterations of the generator training, the number of training of discriminator for every repeating can decrease to 5 for the consideration of speeding up. It is not necessary to balance the training of the discriminator and the generator with the excellent characteristic of Wasserstein GAN.

Before training, image resizing is necessary since all the images in Market1501 [57] are 128×64 and the sizes of images in DukeMTMC are not uniform. If the original

images are directly used as input, the structure of the discriminator and generator are both needed to be altered to adapt the input size considering the original version is designed to deal with square images, which is complicated and requires more learning procedure for the new parameters. I tried to change the kernel size and the number of convolutional layers to adapt 2:1 images and the training became very slow and hard to converge. However, I still believe that changing the structure will bring a better performance, which can be a future work to deal with. If the input images are directly resized the to 128×128, the pedestrians in the images can be 'weird' visually and may lose semantic information, which is illustrated in Figure 17.

Figure 17: The problem of resizing images from 128×64 to 128×128

A simple and effective method is to pad the original images to 128×128 with white colours and resize it to 64×64 as the inputs of the discriminator. Under this circumstance, the structure of discriminator and generator remain the same, and the training is still efficient since the numerical input of the padding area is constant and imposes no redundant information, which is shown in Figure 18. It only costs less than 5 epochs for the network to work out that the useful information is located in the centre of the samples. The padding area of these generated images are cropped, and the size of the images are reset to 128×64 to be constant with the original data.

The training of Wasserstein GAN is conducted on NVIDIA Tesla P100 16GB. It costs 10 hours to train the generator for $300k$ times. From Figure 19, it is easy to conclude that the training error of the discriminator is well associated with the quality of generated images, which is demonstrated in [2]. After $300k$ iterations of the generator, the training error remains the same, and the iteration is stopped there. With this generator, we generate 12,000 images and 16,000 images for enlarging the training set of Market1501 and DukeMTMC respectively.

Figure 18: Generated samples with padding area

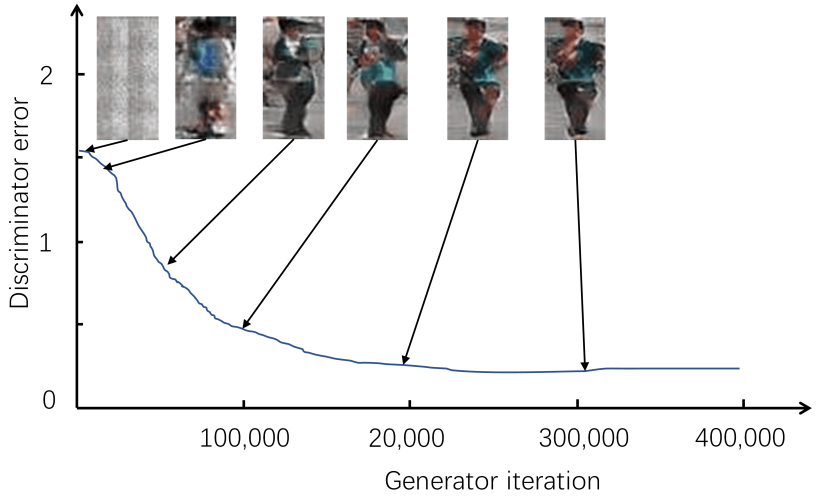


Figure 19: The training process of discriminator on Market1501

In conclusion, we adopt the DCGAN [42] as the structure of our generative model and train it with respect to EM distance by following Wasserstein GAN [2]. Additionally, we apply constant padding to the input images and use the trained generator to generate 12,000 and 16,000 synthetic samples for Market1501 and DukeMTMC-reID respectively.

4.4 Data Pre-processing

Deliverables: A novel and dedicated pre-processing architecture for person re-ID on Market1501 and DukeMTMC. (**complete own work**)

Although there are many public datasets for person re-ID task, we mainly focus on

the **Market1501** [57] and **DukeMTMC-reID** [60] due to the time limitation and data volume.

Before applying the baseline model to the data, there are some pre-processing procedures which are necessary to carry out to make the model achieve better performance.

Image resizing and cropping: The first step is to make sure that all the input images have the same size and aspect ratio considering the samples in DukeMTMC have different size with samples from Market1501. In this project, the input images are first resized to the size of 288×144, and then randomly centre-cropped to 256×128. With these two operations together, we reduce the contribution of the background in the original images while keeping the image size consistent.

Normalisation: Data normalisation is a crucial step which ensures that each input parameter (pixels) has a similar data distribution, thus to make the convergence of the model faster while training it. Usually, normalisation is conducted by subtracting the mean of the images from each pixel and then dividing it by their standard deviation. After normalisation, the distribution of the input data would resemble a Gaussian curve centred at zero. According to [60], we normalise the input images of Market1501 with mean of $[0.485, 0.456, 0.406]$ and standard deviation of $[0.229, 0.224, 0.225]$, which are scaled from $[0, 255]$ to $[0, 1]$. As for DukeMTMC, we use the same mean and standard deviation. Figure 20 provides several examples of the normalised images from Market1501.

Figure 20: Data normalising for Market1501

Data augmentation: Usually, some data augmentation techniques are applied before the training process, including rotation, flipping and so on. Data augmentation is conducted to expose the model to a wide variety of variations to make sure it is less likely that the neural network recognises unwanted characteristics in the dataset. In this project, all the training images are horizontally flipped randomly. Besides, the random erasing method [62] is utilised to improve the efficiency of data augmentation further. We will show in the evaluation section that random erasing

can effectively improve the performance of our person re-ID system. Also, the generated samples from Wasserstein GAN [2] can be regarded as a data augmentation technique.

Figure 21: Horizontally random flipping examples from Market1501

After image resizing, cropping, normalisation and data augmentation, the images are well organised under a Gaussian distribution with identical size.

4.5 Model Construction

Deliverables: A dedicated CNN baseline model for person re-ID. This is a new architecture developed specifically for this project. (**complete own work**)

In this part, the details of the baseline model: 50-layers deep residual network and the reason why it is selected in this project will be discussed comprehensively. Additionally, we make some crucial modifications to the baseline model, which is one of our primary contributions and makes the model significantly surpass the original one.

This part is the most time-consuming procedure in this project because if the model is not optimal, the training would cost much time and achieve nothing.

4.5.1 Baseline Model

The first step is the selection of the baseline model. Since time is limited, it is relatively impossible to construct a new model from scratch for our project. Here we will give the details of the architecture of the model selected in this project: ResNet50.

There are many proposed state-of-the-art deep neural networks, including VGG, ResNet, CaffeNet and so on. While normalised initialisation and batch normalisation [19] have primarily addressed the vanishing gradient problem, the degradation problem has been exposed among very deep neural networks: with the network depth increasing, the accuracy of training can get saturated and then degrade rapidly [17]. To avoid this problem in this project, a fine-grained 50-layers Deep Residual Network (ResNet50) [17] is deployed for feature extraction. ResNet50 is a very popular CNN structure due to its simplicity and excellent performance on computer vision tasks. Different from traditional multi-layer perceptron (MLP), the residual network adds a shortcut connection between the convolutional layers, which is shown in Figure 22. ResNet has many different versions, from 18 to 152 layers (actually we can even add more layers). We choose the 50-layers version because it has a competitive performance while requiring reasonable computational resources (3.8×10^9 FLOPs [17]). If the 101-layers or 152-layers version is selected, we believe that the performance can be further improved at the expense of training time and more powerful machines (it requires more GPU memory to store the massive amounts of parameters).

Considering $\mathcal{H}(x)$ as the underlying mapping to be fit by a few stacked non-linear layers, where x represents the inputs of the first layer. If we assume that multiple non-linear layers (for example, convolutional layers) can asymptotically approximate a complicated non-linear function, it is equivalent to hypothesise that they can asymptotically approximate a residual function $\mathcal{H}(x) - x$ (assuming that the input and output are of the same dimensions). Therefore, rather than expect stacked layers to approximate $\mathcal{H}(x)$, we explicitly let these layers approximate a residual function $\mathcal{F}(x) = \mathcal{H}(x) - x$. The original function thus becomes $\mathcal{F}(x) + x$. Although both forms should be able to approximate the desired functions asymptotically, the ease of learning might be different, and it is proved by He *et al.* that this structure can be easier to optimise, and avoid degradation problem and gain accuracy from considerably increased depth simultaneously.[17].

The building block used here is stacked with three convolutional layers, with kernel sizes of 1×1, 3×3 and 1×1 respectively. The strides of every layer are set to 1, and the padding of the second convolutional layer is set to 1, with another two of padding 0 (default). With this design, He *et al.* argued that it could efficiently decrease the training time by using the front and end 1×1 convolutional layers to reduce and increase the dimensions and leave the 3×3 layer a smaller input and output dimensions. The ResNet50 model is illustrated in Figure 23. In convolutional block 1, the first two numbers of maps are 64, and the last one is 256. All of

these maps are expanded to 4 times with the increase of blocks.

In conclusion, the shortcut connection of ResNet makes it a very efficient model which can avoid degradation, and it is adopted as the baseline model in this project.

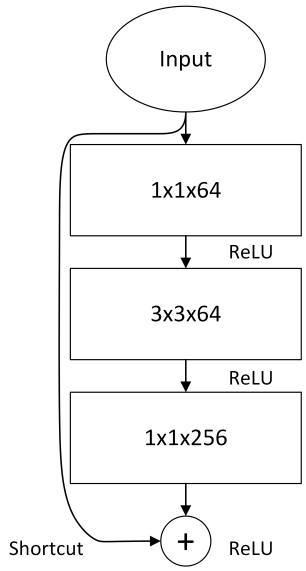


Figure 22: Deep block of ResNet

4.5.2 Modifications

The modification to the ResNet50 is one of my main contributions. Due to the limitation of time computer resources, it is difficult to make sure the model can converge to the optima.

To conduct a classification task on Market1501 [57] and DukeMTMC-reID [60], we add another fully-connected layer and a classification layer, where the fully-connected layer is composed with linear layer, batch normalisation [19], ReLU activation [37] and dropout regularisation [18]. The dropout probability is set to 0.5. The number of output is set to 751 or 702 with respect to the number of identities in the related two datasets.

First, batch normalisation [19] is utilised in every layer of the model. Batch normalisation has been widely used in all kinds of deep neural networks due to its efficient ability to address internal covariance shift. Considering that we are using mini-batch gradient descent in this project, the distributions of different mini-batches are inconsistent. For every mini-batch of data, we train our model to adapt its distribution, which would cause the training process unstable and consuming more time to converge. Batch normalisation addresses this problem by normalising the output

of the previous layer. More specifically, it subtracts the batch mean of the output of the previous layer and divides it by the batch standard deviation. However, this operation will change the original distribution, and consequently, the weights of the next layer will be no longer optimal. Thus, batch normalisation adds two learnable parameters to each layer: γ and β, so the normalised output of the previous layer output is multiplied by a "standard deviation" parameter (γ) and added with a "mean" parameter (β):

$$y_i = \gamma x_i + \beta \tag{5}$$

In another word, the last step of batch normalisation does the "denormalisation" by learning only these two parameters for each activation, instead of losing the stability of the network by changing all the weights. With this operation, we can use a higher learning rate since the activation of every layer is fixed within a range. Also, it has a slight regularisation effect since it adds some noise to each layer.

According to [17], we use batch normalisation for every convolutional layer and fully-connected layer, except for the classification layer. This can help our model to converge faster and be more stable while training. After training, the batch normalisation use the expectations of mean and variance of every mini-batch from training set as the mean and variance of the testing process.

Although the batch normalisation can provide a slight regularisation effect, dropout [18] is necessary to achieve a better regularisation effect and avoid over-fitting. We do not apply dropout to the original ResNet50 by following [17] as they argue that the design of the blocks can impose regularisation on the model. At the beginning, we did not implement the dropout on the added fully-connected layer. With this architecture, the training procedure is smooth while the testing result is always not optimal, which means our model is over-fitting to some extent. Since we enlarged our training samples with Wasserstein GAN [2], the problem is not raised by the size of the training set. The problem is we did not use other regularisation methods. The three most popular options are L1 regularisation, L2 regularisation and dropout, while the dropout is the most common in deep learning. Thus, we modified the fully-connected layer by adding the dropout with the probability of 0.5. Also, dropout can be applied to the convolutional layers. Since the dropout will cause the input information to get lost, and our model is very deep with many convolutional layers, we did not adopt it to the convolutional layers. Under this circumstance, the testing performance can be improved by **8%**. It is worth mentioning that the probability of dropout is a hyper-parameter and can be tuned to improve the performance further. Besides, equipping convolutional layers with dropout is worth a try in the future.

We use leaky ReLU rather than original ReLU as the activation function of the new fully-connected layer. The motivation is ReLU is easily to be "die": if the input of a ReLU activation function is consistently negative, then the output will be 0, and the gradient through it will also be 0. Under this circumstance, the error signal

propagated from later layers gets multiplied by this 0. Thus no error signal will pass to earlier layers. On the contrary, if we use leaky ReLU instead, the model is capable of mapping the negative input appropriately without losing the advantage of ReLU: does not saturate and can avoid vanishing gradient.

Following the deep block structure, the added fully-connected layer outputs a 512-dimensional vector from the 2048-dimensional input. Then the 512-dimensional vector is transformed into the number of classes with the classification layer. Every numerical value in the output represents the probability that the input image belongs to that class, and the maximum of them is selected as the prediction.

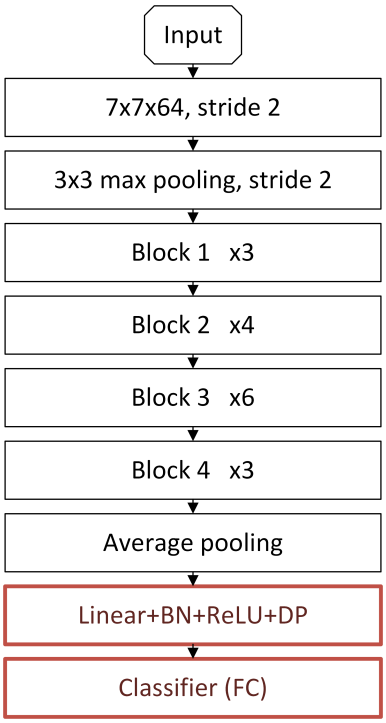


Figure 23: The baseline ResNet50 network

4.6 Model Training and Optimisation

Deliverables: A complete train and optimisation system for the proposed model as well as the extracted features for the selected datasets. (own work for model training and redistribution of re-ranking and random erasing from GitHub codes of [61] and [62] respectively)

In this part, the training mechanism of the model as long as the hyper-parameters setting will be delivered. After that, the analysis of two optimisation techniques: re-ranking [61] and random erasing [62] will be provided. The combination of Wasser-

stein GAN, random erasing and re-ranking is one of our contributions.

4.6.1 Model Training

In brief, this project uses mini-batch gradient descent with momentum [41] as the optimisation method to train the model, and the objective function is set to cross-entropy loss. The training process is conducted twice: first with the original dataset and then with the enlarged dataset. The mechanism of the mini-batch gradient descent will be given in details.

Usually, the training of deep neural networks is time-consuming since there are a large number of parameters to tune. Following [58], we pre-train our baseline model on the ImageNet [9] rather than training it from the scratch to save time. In this project, the primary target is demonstrating the effectiveness of data augmentation with Wasserstein GAN [14] and some improvement techniques rather than tuning the hyper-parameters to get better performance. Thus, many hyper-parameters including learning rate and mini-batch size have not been well-tuned during the training procedure. Therefore, most of the values of hyper-parameters that will be mentioned later are selected with experiences and guidance from other papers [60, 44, 31].

As illustrated in Figure 13, the training is divided into two steps: (1) before adding the generated data, (2) after adding the generated data.

For the training with the original data, we simply regard it as a classification task. We set the batch size to 32 and train the model with the cross-entropy loss for 60 epochs. The cross-entropy loss, also known as log loss, is usually utilised to measure the performance classification models where the output is a set of probability values from 0 to 1. Assuming M is the number of classes, and p_i is the output probability of the class i. Then the cross-entropy loss is calculated as,

$$F = \sum_{i=1}^{M} y \log p \tag{6}$$

where y is a binary value to indicate that the class label i is correct or not.

To simplify the training process, we choose the first image of every identity as the validation set and regard the remaining samples as the training set. With a single image as input, the model will output 751 or 702 values which represent the probability that the input image belongs to each class. The maximum value is selected as the predicted label, and the cross-entropy loss is calculated with the predicted label and the ground truth.

The optimisation scheme is a mini-batch gradient descent with a momentum of 0.9 and weight decay as 5e-4. Gradient descent is an iterative method to optimise the objective function (here is the cross-entropy loss). During every iteration, the cross-entropy loss is calculated with respect to the predicted labels and ground truths of every image in the mini-batch. Then the loss is backpropagated by calculating the loss with respect to the weights in each layer. Finally, the weights W are updated as,

$$W = W - \alpha \nabla F(W) \tag{7}$$

where α is the learning rate and $\nabla F(W)$ is the gradient of objective function $F(W)$ with respect to W. The learning rate is also a hyper-parameter. Selecting an appropriate learning rate can be difficult since a learning rate that is too small leads to a slow convergence process, while a learning rate which is too large will cause the objective function to fluctuate around the minimum or even to diverge. Also, since stochastic gradient descent is unstable and requires much time to converge, we use mini-batch gradient descent in this project. The input images are grouped as a batch and the loss is calculated by averaging all the losses of the input images. Momentum [41] is a method that is inspired from physics process. It can help accelerate stochastic gradient descent in the relevant direction and dampens oscillations. More specifically, it adjusts the current direction of weights updating with a part of the previous one:

$$v_t = \beta v_{t-1} + (1 - \beta) \nabla F(W),\ W = W - \alpha v_t \tag{8}$$

With momentum, the derivative is estimated within a small batch (decided by β), which means the weights are not updated in the optimal direction, because the exact derivatives are "noisy". Besides, it can help model avoid local optima. Intuitively, supposing we are pushing a ball down a hill. The goal is to find a way down the hill quickly. The original stochastic gradient descent is to choose the steepest path to achieve it. However, as we all know, The ball can accumulate momentum when it rolls downhill, thus becoming increasingly faster on the way until it reaches the final velocity with the balance of gravity and the air resistance. Therefore, the direction of pushing the ball should be adjusted by the direction of the accumulated momentum to achieve a higher velocity.

Since our model is pre-trained on the ImageNet [9], the weights of the pre-trained part are close to optimal and can be updated with a smaller learning rate. Thus, we set their initial learning rate to 0.01, while setting the learning rate of the added two layers to 0.1. We decrease the learning rate to 10 times smaller for every 40 epochs to train the model to optima. Some important setting is shown in Table 2.

The training and validation loss and error (1 - accuracy) are shown in Figure 25. It is easy to find that the model was close to saturation and did not get to the optima when after 40 epochs of training. However, after decreasing the learning rate, the model was keeping learning for a couple of iterations. Finally, the model gets a near

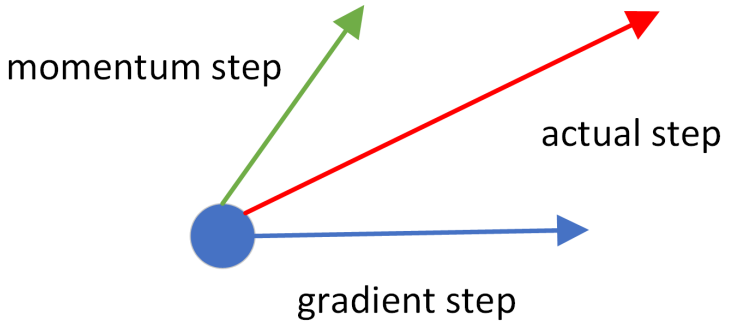


Figure 24: The intuition of momentum

Training epochs	60
Batch size	32
Loss function	Cross-entropy loss
Optimiser	Stochastic gradient descent
Weight decay	5e-4
Momentum	0.9
Initial learning rate	0.01 for pre-trained part and 0.1 for added part
Learning rate decay	Every 40 epochs with gamma of 0.1
Validation data	The first image of every class

Table 2: Training setting

100% accuracy on both the training and validation set. It is easy to find that the training error does not fluctuate severely mainly because the momentum technique is applied on the stochastic gradient descent. The parameter of momentum $\beta = 0.9$ means the current gradient is averaged with the previous 9 values.

After training, we use the saved model to extract the features of query and gallery images for re-ID. We remove the last two added layers (fully-connected and classification layer) to make the model output a 2048-dimensional vector as the feature representation of the input image. Actually, we did not test the classification accuracy on query and gallery images since it is unnecessary for re-ID tasks.

Then we use the generator form Wasserstein GAN to generate 12,000 unlabelled samples for Market1501. The intuition is, the original dataset contains 12936 training images, and we want to double the size of it. After data generating, the previously trained model is utilised to assign labels for these unlabelled images. More specifically, we extract the features for both the training set and generated images, and then employ Euclidean distance to calculate the distance between them. For every synthetic image, the label of the closest training sample is assigned to it as a pseudo label. Then all these images are added to the original training set to form a new training set.

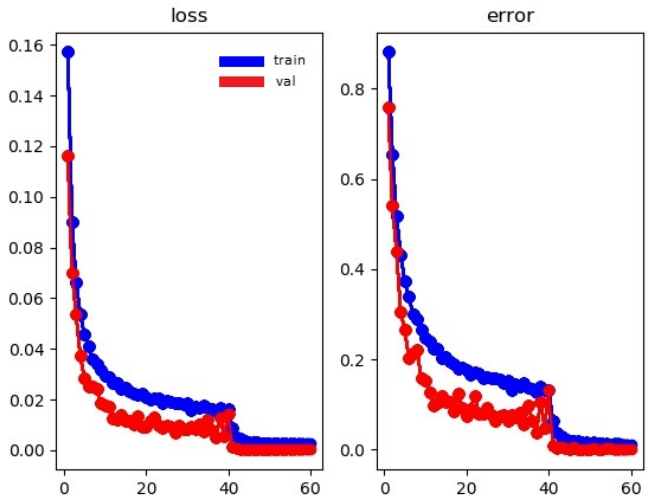


Figure 25: Training process

Since the input of the model is RGB colour information, the label assignment tends to assign an image label of other classes that the images are visually similar to it. However, since the model has not seen the synthetic images, it may assign some labels that are "inappropriate", which is illustrated in Figure 26. The first generated image is visually similar to the real images that belong to the same identity. The second one looks blurry but still has similar colours. The last one is entirely different from the real images. However, it is believed that this can improve the generalisation ability of the model and proved by the results of the experiments. Some papers manually assign some images of the original dataset wrong labels to achieve a similar result with the proposed method.

Figure 26: The example of label assignment

The second step is trained with the enlarged dataset while the other settings remain unchanged. However, since the volume of the dataset is nearly doubled, to train it

to optima, we need to iterate it for 100 epochs.

To make a comparison with other methods of label assigning and exhibit our superior performance, we also applied "All in one" [38] and LSRO [60] to the generated data, which we will go into details in the next section.

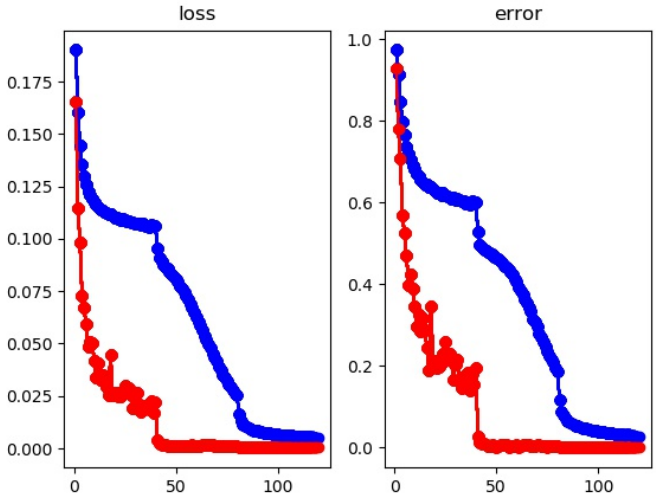


Figure 27: Training process

4.6.2 Optimisation

There are many optimisation techniques for person re-ID have been proposed in recent years. We choose two of them to optimise our model: random erasing [62] and re-ranking [61], which are both proved to be effective for person re-ID.

Random erasing is proposed by Zhong *et al.*. The basic idea is randomly choosing rectangle regions of arbitrary sizes and assign the pixels with random values or their mean values. Their motivation is related to the generalisation ability of CNN models. Occlusion is quite common in the surveillance system due to the complexity of the real world. Thus, a re-ID system always needs to deal with pedestrian images with a certain level of occlusion. A robust model can still achieve a high accuracy when some parts of the images are occluded. However, the training samples of popular re-ID datasets usually exhibit limited variance in occlusion while it is a critical factor considering to avoid over-fitting.

With random erasing, various levels of occlusion can be generated without more parameters to train and learn. It can be regarded as a new data augmentation way, which is similar to perform dropout on the image level.

Assuming that the randomly selected rectangle region of an training sample is I_e , and the size of the training sample is $W \times H$. The area of the sample is $S = W \times H$. Following the [62], we randomly initialise the area of erasing rectangle region to S_e, where S_e/S is in range specified by minimum s_l and maximum s_h. The aspect ratio of erasing rectangle region is randomly initialised from r_1 and r_2, we set it to r_e. The size of the region I_e is decided with $H_e = \sqrt{S_e \times r_e}$ and $W_e = \sqrt{S_e/r_e}$. Then, we randomly choose a point $P = (x_e, y_e)$ in the original image I. If $x_e + W_e \leq W$ and $y_e + H_e \leq H$, the region $I_e = (x_e, y_e, x_e + W_e, y_e + H_e)$ is selected as the region to erase. Otherwise, the above process is repeated until an appropriate rectangle region is selected. In this project, with the selected erasing region I_e, each pixel in it is assigned to the value of 255 (black), respectively. We fix S_l to 0.02, $S_h = 0.4$, $r_1 = 1/r_2 = 0.3$.

Since it is easy to implement and combine with our baseline model, we adopt this technique as one of our optimisation methods. We combine it with random flipping and random cropping as a data pre-processing procedure. In this project, we set the probability of random erasing to 0.5, which means half of the training samples are processed with random erasing. Although the probability is a hyper-parameter which needs to be tuned through a series of comprehensive experiments, we simply follow the setting of [62] since it is not our main objectives to tune all these parameters. The Figure 28 illustrates the erasing results on Market1501 [57].

Figure 28: Random erasing results on Market1501

The other optimisation technique is re-ranking [61]. Regarding person re-ID as a retrieval process, the ranking procedure is always an essential step since it is directly related to the results of re-ID. Thus, re-ranking is critical to improving its accuracy, especially the mAP.

This method is also proposed by Zhong *et al.*. Suppose $N(p, k)$ represents the k-nearest neighbours of the probe image p, which can be defined as:

$$N(p, k) = g_1, g_2, ..., g_k \tag{9}$$

As to the k-reciprocal nearest neighbours $R(p,k)$ where both images are ranked top-k when the other image is taken as the query, which can be defined as,

$$R(p,q) = g_i | (g_i \in N(p,k)) \wedge (p \in N(g_i,k)) \tag{10}$$

The k-reciprocal nearest neighbours are proved to be more robust than k-nearest neighbours [61]. However, due to the influence of different illuminations, poses of pedestrians, camera views and occlusions, the positive images of the probe can be excluded from the k-reciprocal nearest neighbours. Thus, the authors incrementally add the $\frac{1}{2}$-reciprocal nearest neighbours of each candidate in $R(p,k)$ into a more robust set $R*(p,k)$ following the condition

$$R*(p,k) \leftarrow R(p,k) \cup R(q,\frac{1}{2}k) s.t.\ |R(p,k) \cap R(q,\frac{1}{2}k)| \geq \frac{2}{3}|R(q,\frac{1}{2}k)|,\ \forall q \in R(p,k) \tag{11}$$

Then, the k-reciprocal nearest neighbours set of every image in the query and gallery sets is formed as a feature vector and Jaccard distance is utilised to calculate the distance between these features. More specifically, the Jaccard distance is calculated as,

$$D_{Jaccard}(p,g_i) = 1 - \frac{|R*(p,k) \cap R*(g_i,k)|}{|R*(p,k) \cup R*(g_i,k)|} \tag{12}$$

where $|\bullet|$ refers to the number of candidates in the corresponding set. A weighted aggregation of the Jaccard distance and the original distance (Euclidean distance) is calculated as the final distance for ranking the gallery images. In this project, we set the weights of both distances as 0.5, which is:

$$D_{Final} = (D_{Original} + D_{Jaccard}) \ / \ 2 \tag{13}$$

Also, this method is easy to implement and requires no extra parameters to learn.

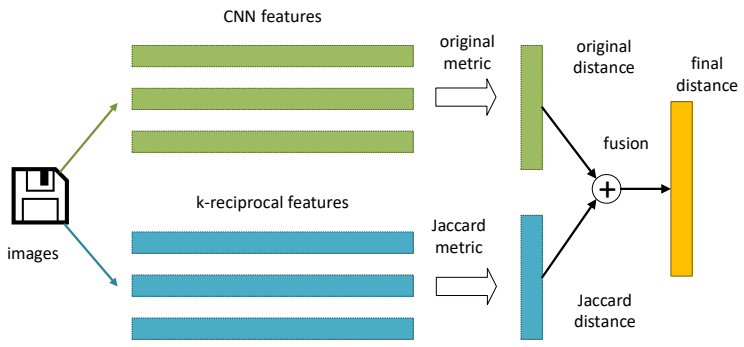


Figure 29: Re-ranking procedure

In conclusion, random erasing is utilised at the data pre-processing stage to generate occlusions and re-ranking is conducted to improve the performance of the baseline

model further. After the re-ranking procedure, the final ranking results for Market1501 and DukeMTMC-reID are obtained. In the next part, the performance of the proposed method will be analysed systematically.

4.7 Summary

In this section, an overview of the proposed person re-ID system is provided, and the order of execution steps is given. First, we talked about the use of Wasserstein GAN [2] and why it works better than the original GAN [14], as long as the proposed new label assignment method: one of the main contributions. Then we gave the details of data pre-processing and the reasons for applying these techniques. Next, the most important part: model construction was well argued with the intuition of modifying the original ResNet [17]: one of another main contribution. At last, the details of training and optimisation of the baseline model are provided to show some problems we faced during the project.

With all the steps mentioned before, we get the deep features of the query images and gallery images in Market1501 and DukeMTMC-reID. Then the gallery images are ranked with respect to the distance between these features. In the next section, the obtained ranking list will be used to evaluate the performance of the proposed model.

5 Experiments and Evaluation

5.1 Evaluation Results

A series of systematic experiments are carried out based on the model and the optimisation methods mentioned in section 4, where RE refers to random erasing [62], RR represents re-ranking [61] and WGAN stands for Wasserstein GAN [2]. Also, the performance evaluation results are provided in Table 3, Figure 30, Table 4 and Figure 31. Briefly, the proposed person re-ID system achieves a state-of-the-art performance on Market1501 [57] and a competitive result on DukeMTMC-reID [60], which demonstrates that the proposed method is effective on person re-ID task.

Table 3: Final evaluation results on Market1501

Market1501	Single Query		Multi Query	
Model	Rank 1	mAP	Rank 1	mAP
IDE(ResNet)	73.69	51.48	81.47	63.95
IDE+LSRO [60]	83.97	66.07	88.42	76.10
HACNN [31]	91.2	75.7	**93.8**	**82.8**
Proposed	88.54	70.16	91.69	77.35
Proposed + RE(0.5)	89.31	73.56	92.28	80.45
Proposed + RE(0.5) + RR	92.19	88.34	-	-
Proposed + RE(0.5) + RR + WGAN	**92.89**	**89.84**	-	-

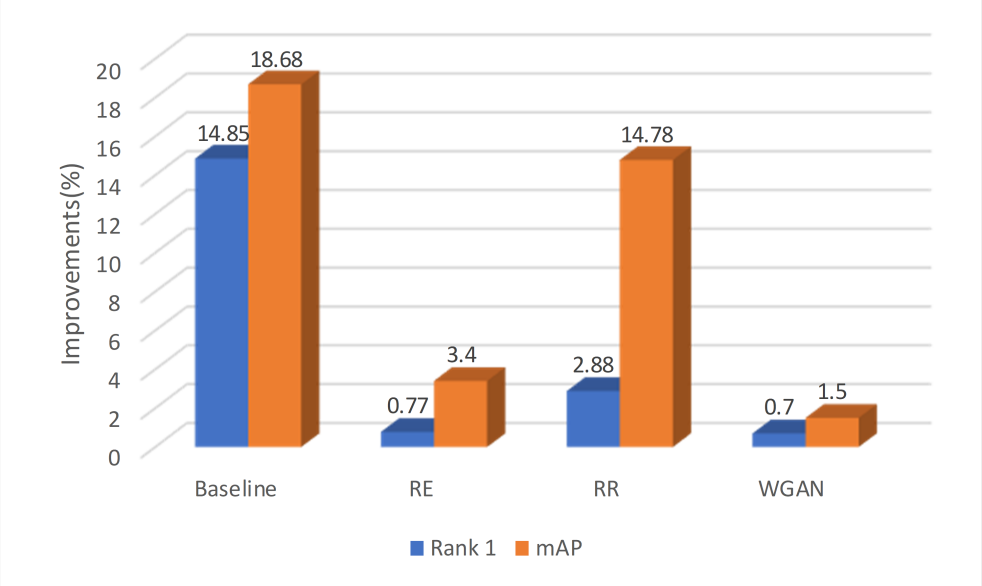


Figure 30: The performance improvements on Market1501

Table 4: Final evaluation results on DukeMTMC

DukeMTMC	Single Query	
Model	Rank 1	mAP
BoW+KISSME	25.13	12.17
LOMO+XQDA	30.75	17.04
Proposed	76.84	55.04
Proposed + RE(0.5)	77.53	57.81
Proposed + Re(0.5) + RR	81.78	77.08
Proposed + Re(0.5) + RR + WGAN	82.28	**78.18**
SPreID (Res-152)	**85.95**	73.34

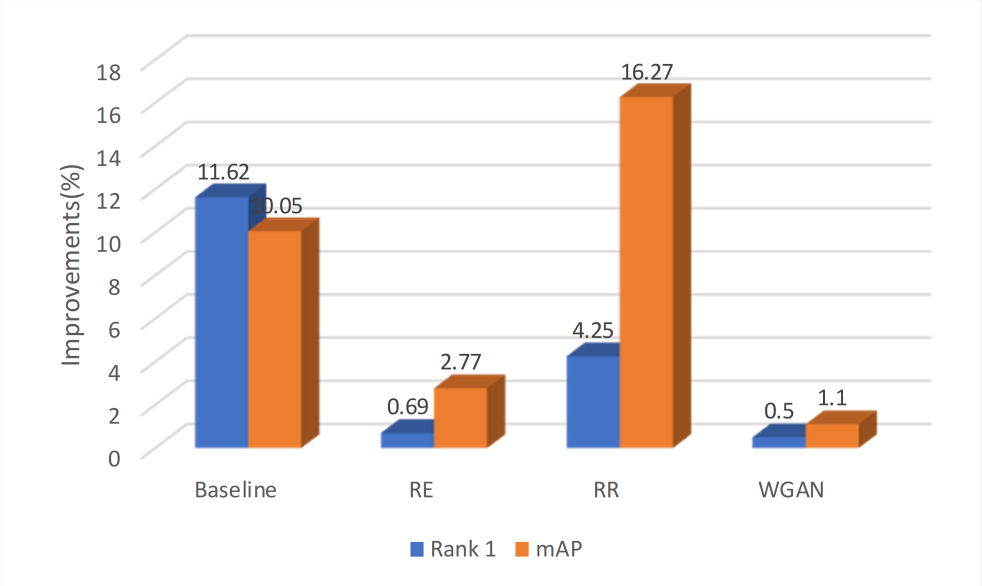


Figure 31: The performance improvements on DukeMTMC

5.2 Evaluation Methodology

After extracting all the features of query and gallery images, we save them to a file and prepare for the ranking process. We use CMC curve and mAP to evaluate the performance in this project. CMC curve represents the precision among all the ranking results in the gallery images by giving a query input while mAP can evaluate the recall at the same time. Since there are more than one ground truths in the gallery images, mAP is more comprehensive to describe how well all the ground truths are ranked.

While ranking the gallery images with respect to the query images, we regard all the images which have same labels and same camera identities as "bad matches" since what we want to do are to match the pedestrians from different cameras, and they will be removed from the corresponding ranking list. On the other hand, it is

neither a positive sample nor a negative sample.

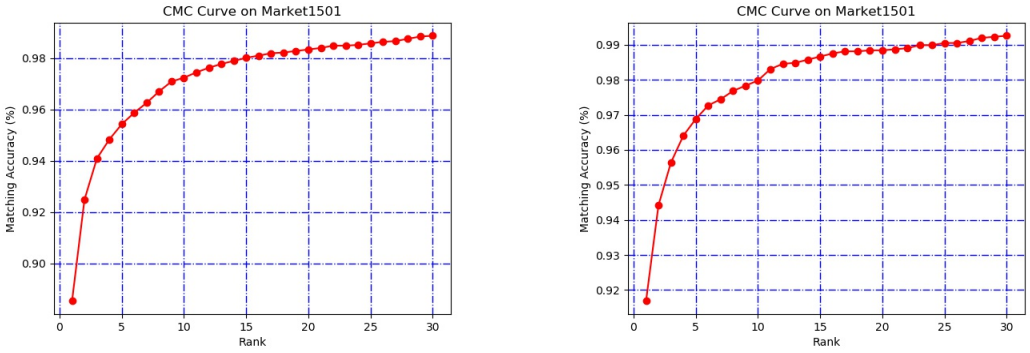
It is worth mentioning that there are two main modes of query: single-query and multi-query. Under the single-query mode, there is only one query image, and the gallery images are ranked based on the distance between the query image and every gallery image; and under the multi-query mode, there are more than one query images provided and the gallery images are ranked based on the average distance between these query images and every gallery image. Therefore, multi-query is more robust and usually achieves a higher accuracy by considering multiple query images. Unfortunately, not all datasets are capable of using multi-query mode. In this project, Market1501 will be evaluated in two modes while DukeMTMC will only be evaluated under single-query mode.

As mentioned before, the mAP is essential to show the recall of the proposed model. For Market1501 dataset, there are on average 14.8 ground truths from different cameras for each query image [57]. For each query, the area under the Precision-Recall curve is calculated, which is known as average precision (AP). Then, mAP is obtained by averaging the value of APs among all query images. Both precision and recall of the model are well reflected by mAP. Thus it can provide a more comprehensive evaluation than CMC curve.

5.3 Baseline Model ("Proposed" in Table 3 and 4)

For comparison, first, the performance of the baseline model is evaluated on the original dataset without adding random erasing and re-ranking procedures. On Market1501, we achieve 88.54%, 95.43%, 97.24% of rank 1, 5 and 10 accuracies, and we arrive at mAP = 70.16%. This result significantly outperforms the result reported in [60] (**+14.85%** on rank 1 and **+18.68%** on mAP), which demonstrate that the revised ResNet50 is superior to the original version. Also, since Market1501 support multi-query mode, We tested the proposed model under multi-query mode and reported 91.69% and 77.35% on rank 1 accuracy and mAP, which are also effective improvements. The CMC curves of the single-query mode and multi-query mode on Market1501 are shown in Figure 32.

DukeMTMC does not provide multi-query mode (there is only one query image for every identity from a single camera). Thus we only tested it with the single-query setting. With the baseline model, the rank 1 accuracy and mAP are improved from 65.22% and 44.99% to 76.84% (**+11.62%**) and 55.04% (**+10.05%**) respectively. Although the improvements are not as significant as achieved on the Market1501, this is still a competitive result. There are some reasons. Firstly, the data normalisation procedure for both Market1501 and DukeMTMC is configured consistently. Obviously, the mean and standard deviation for normalising DukeMTMC should be different from those for Market1501. We simply use the same setting, and the



(a) Single-query (b) Multi-query

Figure 32: The CMC curves on Market1501

performance will be influenced. Also, since different datasets are from different data distributions, one single model may be more suitable for one of them while performing worse on other datasets. The CMC curve is illustrated in Figure 33.

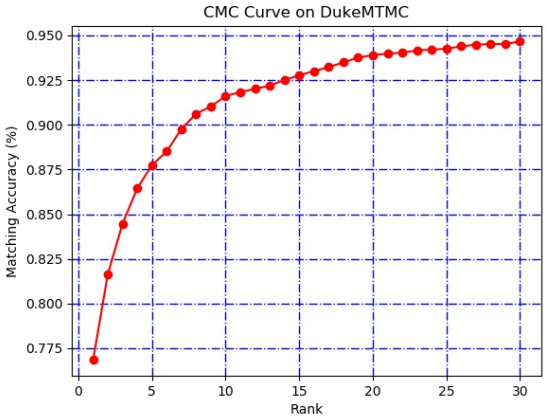
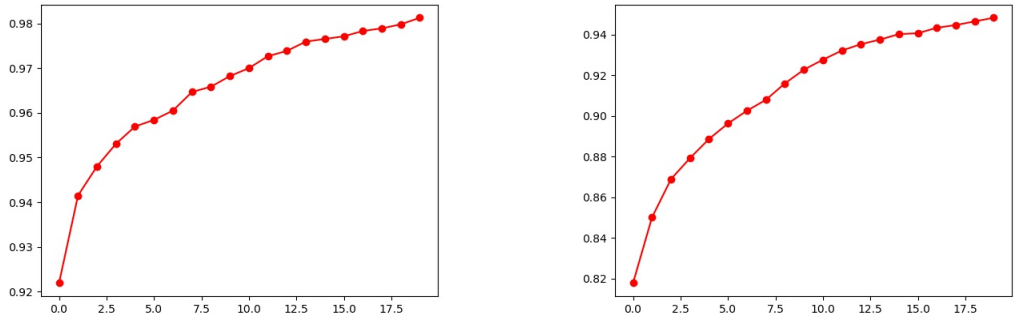


Figure 33: The CMC curve on DukeMTMC

5.4 Baseline Model + RE + RR

Then I want to explore the efficiency of random-erasing and re-ranking on the baseline model. It is worth mentioning that re-ranking cannot be applied to the model with a multi-query setting due to its mechanism. Thus, to simplify the evaluation procedure, all the testing are configured to single-query mode. After adding random erasing to our model, we gained improvements of **+0.77%** on rank 1 and **+3.40%** on mAP with Market1501, and **0.69%** on rank 1 and **2.77%** on mAP with DukeMTMC, which prove that random erasing is an effective data augmentation technique. After re-ranking procedure, the performance on Market1501 is

boosted to **92.19%** on rank 1 accuracy and **88.34%** on mAP. As we can tell from the results, re-ranking is a critical procedure to improve the performance of person re-ID task, especially the mAP since it improves the recall effectively. Also, when testing the re-ranking on DukeMTMC, I arrive at **rank 1 = 81.78%**, **mAP = 77.08%**, which is very competitive.



(a) Market1501 (b) DukeMtMC

Figure 34: The CMC curves with random erasing and re-ranking

To visually show the efficiency of the re-ranking technique, one random query image is selected, and its rank 21 gallery images before and after applying re-ranking are given in Figure 35. At this time, the "bad matches" are not removed to make a comprehensive comparison. The original ranking result already achieves a good result while the re-ranking further improve it.

5.5 Baseline Model + RE + RR + WGAN

One last improvement and one of my main contributions is the training set enlarging with the generated samples from Wasserstein GAN.

In this project, 12,000 synthetic images are generated for Market1501 by the generator of Wasserstein GAN. This number is not well tuned and can be improved by more experiments. Our motivation is to enlarge the training sample to near double size. The synthetic images are then labelled through the original baseline model and added to the original training set. We do not care about the camera identities of the generated samples since they will be removed at the testing stage. All the generated images are assumed to be obtained from the same camera. After adding these generated samples, the model is re-trained for 100 epochs to reach its optimal. We arrive at **rank 1 = 92.89%** and **mAP = 89.84%** with the final improved model on Market1501. All the evaluation results on Market1501 is shown in Table 3, as long as some other state-of-the-art methods for comparison. As shown in the table, our method achieved the state-of-the-art with the single-query setting, which

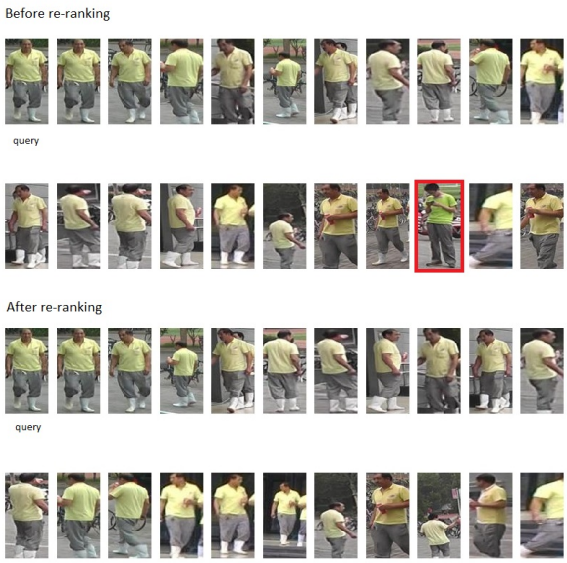


Figure 35: Rank 21 results before and after applying re-ranking

indicates our model is effective and competitive. Despite that the improvement with Wasserstein GAN is not significant, considering that the generated samples aa little blurry, we argue that this improvement is still impressive and demonstrate the efficiency of more training data.

From the Figure 30, the most significant improvements are obtained from the modification of the baseline model. It is easy to understand that the mapping function approximator, CNN baseline model has the most considerable effect on the performance. Re-ranking is critical to improving the recall of the proposed model, thus providing a substantial enhancement on the mAP. Random erasing and adding more data with WGAN are both data augmentation techniques, which improve the model from the perspective of avoiding over-fitting. The model itself is not improved, so the improvements are less effective. This result provides a good hint that the proposed model performs well on avoiding over-fitting.

As for DukeMTMC, the number of generated samples is increased to 16,000 for the consideration of doubling the 16,522 training samples in the original dataset. The improvements are **+0.5%** and **+1.1%** on rank 1 and mAP respectively, which are less than I got on Market1501. The reason is the original training set of DukeMTMC is larger than Market1501, and the help that enlarging the training set can provide is less significant. There are around 3.7% gap between the proposed model and the current state-of-the-art methods.

5.6 Label Assignments

Also, to prove the effectiveness of our new label assignment method, we make a comparison with the other two methods: "all in one" [38] and LSRO [60]. "All in one" method assumes that all the generated samples are from a same new class. LSRO assigns them new labels from a uniform distribution. Both of them regard the synthetic images are from new classes while the method proposed in this project join them to the existed classes. From the 5, it is easy to conclude that our method outperforms the other two methods. The LSRO [60] can provide significant performance improvement in their original paper, while even perform worse in our model since our baseline model already achieves a good result.

Table 5: The comparison of different label assignment methods

Method	Rank 1	mAP
All in one	- 2.3%	- 0.7%
LSRO [60]	- 0.9%	+ 1.1%
Ours	**+ 0.7%**	**+ 1.5%**

5.7 Summary

In this section, a systematic experiment is conducted, and an evaluation benchmark is developed to evaluate the proposed method and show its effectiveness on two public datasets: Market1501 [57] and DukeMTMC [60]. On Market1501 the model arrives at a state-of-the-art performance while on DukeMTMC the model achieves relatively competitive results. The main reason is that there is a domain gap between these two different datasets. As we can tell from the results Table 3 and Table 4, one method that can achieve an excellent result on one dataset will perform less well on the other dataset. On the other hand, since our model use RGB images as the input of the model, nearly all the miss-ranked images are influenced by the gallery images that look very similar to the query image. They usually wear the clothes of same or similar colours, which is visually confusing. For the long-term surveillance system, our model will perform much worse. An improved method is to use RGB-D (depth) camera to extract the RGB colour information and depth information simultaneously and weight these two different features to compose a new compact feature that is invariant to colour changing to some extent.

6 Conclusion

In this project, a bespoke person re-ID system with a deep neural network as the baseline model was developed. This system demonstrated state-of-the-art performance on a widely researched dataset. A data augmentation method with Wasserstein GAN [2] and a new way of assigning labels to the generated samples was proposed to further improve the performance of the model. Random erasing [62] and re-ranking [61] were combined with the proposed model to yield further performance improvements. Finally, with an evaluation benchmark based on CMC curve and mAP, state-of-the-art performance on Market1501 was achieved [57] and a competitive result on DukeMTMC-reID [60] was obtained.

Our approach's performance is evaluated systematically with a well-developed evaluation benchmark and encouragingly the results show that the proposed method is capable of dealing with person re-ID tasks with competitive performance on two popular public datasets. However, the proposed model may still be improved. For example, the heavy reliance on RGB information might be considered a shortcoming and inconsistent performance on different datasets would ideally be addressed - although these limitations are shared by other leading approaches.

Most of our original aims were achieved, except the use of metric learning since we argue that the CNN model can map the features to well-constructed feature space and the use of dedicated metric may not offer an improvement. There is support in the literature for this argument from [60] and [57]. Additionally, one of our original goals was to compare performance between the siamese model and classification model. However, we found the performance of the siamese model to be significantly worse than that of classification, which makes this comparison meaningless.

Future work, aiming to further improve the performance of the proposed model, include the following:

- Use RGB-D camera to obtain RGB information and depth information simultaneously;
- Explore part-level features;
- Jointly train the model with more than one datasets;
- Apply attention mechanisms to calibrate misaligned images.

Furthermore, this project mainly focuses on short-term image-based person re-ID in a relatively simple environment and ignore person detection procedure. Also, the proposed method requires some time to perform, which cannot meet the real-time requirement in real applications. Thus, the following works can be explored in the future:

- Video-based person re-ID model;

- Person detection problem in person re-ID;
- Real-time person re-ID system;
- Person re-ID in a complex environment.

References

[1] M. Arjovsky and L. Bottou. Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862*, 2017.

[2] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

[3] S. Bak and F. Brémond. *Re-identification by Covariance Descriptors*, pages 71–91. Springer London, London, 2014.

[4] D. Baltieri, R. Vezzani, and R. Cucchiara. 3dpes: 3d people dataset for surveillance and forensics. In *Proceedings of the 2011 Joint ACM Workshop on Human Gesture and Behavior Understanding*, J-HGBU '11, pages 59–64, New York, NY, USA, 2011. ACM.

[5] L. Bazzani, M. Cristani, and V. Murino. *SDALF: Modeling Human Appearance with Symmetry-Driven Accumulation of Local Features*, pages 43–69. Springer London, London, 2014.

[6] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah. Signature verification using a "siamese" time delay neural network. In *Proceedings of the 6th International Conference on Neural Information Processing Systems*, NIPS'93, pages 737–744, San Francisco, CA, USA, 1993. Morgan Kaufmann Publishers Inc.

[7] D. Cheng, Y. Gong, S. Zhou, J. Wang, and N. Zheng. Person re-identification by multi-channel parts-based cnn with improved triplet loss function. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1335–1344, June 2016.

[8] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 5(4):455–455, Dec 1992.

[9] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, June 2009.

[10] P. Dollar, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: An evaluation of the state of the art. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(4):743–761, April 2012.

[11] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, Sept 2010.

[12] I. Fogel and D. Sagi. Gabor filters as texture discriminator. *Biological Cybernetics*, 61(2):103–113, Jun 1989.

[13] P. E. Forssen. Maximally stable colour regions for recognition and matching. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2007.

[14] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.

[15] D. Gray and H. Tao. Viewpoint invariant pedestrian recognition with an ensemble of localized features. In D. Forsyth, P. Torr, and A. Zisserman, editors, *Computer Vision – ECCV 2008*, pages 262–275, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[16] Z. Harris. Distributional structure. *Word*, 10(23):146–162, 1954.

[17] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[18] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, abs/1207.0580, 2012.

[19] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015.

[20] H. Jegou, H. Harzallah, and C. Schmid. A contextual dissimilarity measure for accurate and efficient image search. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[21] S. B. k, E. Corvee, F. Bremond, and M. Thonnat. Person re-identification using haar-based and dcd-based signature. In *2010 7th IEEE International Conference on Advanced Video and Signal Based Surveillance*, pages 1–8.

[22] M. M. Kalayeh, E. Basaran, M. Gökmen, M. E. Kamasak, and M. Shah. Human semantic parsing for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1062–1071, 2018.

[23] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[24] M. Ktinger, M. Hirzer, P. Wohlhart, P. M. Roth, and H. Bischof. Large scale metric learning from equivalence constraints. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2288–2295.

[25] I. Kviatkovsky, A. Adam, and E. Rivlin. Color invariants for person reidentification. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(7):1622–1634, July 2013.

[26] W. Li and X. Wang. Locally aligned feature transforms across views. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3594–3601, June 2013.

[27] W. Li, R. Zhao, and X. Wang. Human reidentification with transferred metric learning. In K. M. Lee, Y. Matsushita, J. M. Rehg, and Z. Hu, editors, *Computer Vision – ACCV 2012*, pages 31–44, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[28] W. Li, R. Zhao, T. Xiao, and X. Wang. Deepreid: Deep filter pairing neural network for person re-identification. 06 2014.

[29] W. Li, R. Zhao, T. Xiao, and X. Wang. Deepreid: Deep filter pairing neural network for person re-identification. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 152–159, June 2014.

[30] W. Li, X. Zhu, and S. Gong. Harmonious attention network for person re-identification. *CoRR*, abs/1802.08122, 2018.

[31] W. Li, X. Zhu, and S. Gong. Harmonious attention network for person re-identification. *CoRR*, abs/1802.08122, 2018.

[32] S. Liao, Y. Hu, X. Zhu, and S. Z. Li. Person re-identification by local maximal occurrence representation and metric learning. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2197–2206, June 2015.

[33] C. C. Loy, C. Liu, and S. Gong. Person re-identification by manifold ranking. In *2013 IEEE International Conference on Image Processing*, pages 3567–3571, Sept 2013.

[34] B. Ma, Y. Su, and F. Jurie. Local descriptors encoded by fisher vectors for person re-identification. In A. Fusiello, V. Murino, and R. Cucchiara, editors, *Computer Vision – ECCV 2012. Workshops and Demonstrations*, pages 413–422, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[35] A. Mignon and F. Jurie. PCCA: A new approach for distance learning from sparse pairwise constraints. In *IEEE conf. on Computer Vision and Pattern Recognition*, pages 2666–2672, France, 2012.

[36] S. Mika, G. Ratsch, J. Weston, B. Scholkopf, and K. R. Mullers. Fisher discriminant analysis with kernels. In *Neural Networks for Signal Processing IX: Proceedings of the 1999 IEEE Signal Processing Society Workshop (Cat. No.98TH8468)*, pages 41–48, Aug 1999.

[37] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, pages 807–814, USA, 2010. Omnipress.

[38] A. Odena. Semi-supervised learning with generative adversarial networks. *arXiv preprint arXiv:1606.01583*, 2016.

[39] B. Prosser, W.-S. Zheng, S. Gong, and T. Xiang. Person re-identification by support vector ranking. In *Proceedings of the British Machine Vision Conference*, pages 21.1–21.11. BMVA Press, 2010. doi:10.5244/C.24.21.

[40] B. Prosser, W.-S. Zheng, S. Gong, and T. Xiang. Person re-identification by support vector ranking. In *Proceedings of the British Machine Vision Conference*, pages 21.1–21.11. BMVA Press, 2010. doi:10.5244/C.24.21.

[41] N. Qian. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12(1):145–151, 1999.

[42] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[43] C. Schmid. Constructing models for content-based image retrieval. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 2, pages II–39–II–45 vol.2, 2001.

[44] Y. Sun, L. Zheng, Y. Yang, Q. Tian, and S. Wang. Beyond part models: Person retrieval with refined part pooling. *arXiv preprint arXiv:1711.09349*, 2017.

[45] R. R. Varior, M. Haloi, and G. Wang. Gated siamese convolutional neural network architecture for human re-identification. *CoRR*, abs/1607.08378, 2016.

[46] R. R. Varior, B. Shuai, J. Lu, D. Xu, and G. Wang. A siamese long short-term memory architecture for human re-identification. *CoRR*, abs/1607.08381, 2016.

[47] T. Wang, S. Gong, X. Zhu, and S. Wang. Person re-identification by discriminative selection in video ranking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(12):2501–2514, Dec 2016.

[48] X. Wang, G. Doretto, T. Sebastian, J. Rittscher, and P. H. Tu. Shape and appearance context modeling. pages 1–8, 01 2007.

[49] L. Wei, S. Zhang, W. Gao, and Q. Tian. Person transfer GAN to bridge domain gap for person re-identification. *CoRR*, abs/1711.08565, 2017.

[50] K. Q. Weinberger, J. Blitzer, and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. In Y. Weiss, B. Schölkopf, and J. C. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 1473–1480. MIT Press, 2006.

[51] A. Wu, W. S. Zheng, and J. H. Lai. Robust depth-based person re-identification. *IEEE Transactions on Image Processing*, 26(6):2588–2603, June 2017.

[52] L. Wu, C. Shen, and A. van den Hengel. Deep linear discriminant analysis on fisher networks: A hybrid architecture for person re-identification. *CoRR*, abs/1606.01595, 2016.

[53] T. Xiao, H. Li, W. Ouyang, and X. Wang. Learning deep feature representations with domain guided dropout for person re-identification. *CoRR*, abs/1604.07528, 2016.

[54] H. Yao, S. Zhang, Y. Zhang, J. Li, and Q. Tian. Deep representation learning with part loss for person re-identification. *arXiv preprint arXiv:1707.00798*, 2017.

[55] D. Yi, Z. Lei, and S. Z. Li. Deep metric learning for practical person re-identification. *CoRR*, abs/1407.4979, 2014.

[56] L. Zheng, Z. Bie, Y. Sun, J. Wang, C. Su, S. Wang, and Q. Tian. Mars: A video benchmark for large-scale person re-identification. In *European Conference on Computer Vision*. Springer, 2016.

[57] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, and Q. Tian. Scalable person re-identification: A benchmark. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1116–1124, Dec 2015.

[58] L. Zheng, Y. Yang, and A. G. Hauptmann. Person re-identification: Past, present and future. *CoRR*, abs/1610.02984, 2016.

[59] L. Zheng, H. Zhang, S. Sun, M. Chandraker, and Q. Tian. Person re-identification in the wild. *CoRR*, abs/1604.02531, 2016.

[60] Z. Zheng, L. Zheng, and Y. Yang. Unlabeled samples generated by gan improve the person re-identification baseline in vitro. *arXiv preprint arXiv:1701.07717*, 3, 2017.

[61] Z. Zhong, L. Zheng, D. Cao, and S. Li. Re-ranking person re-identification with k-reciprocal encoding. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 3652–3661. IEEE, 2017.

[62] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang. Random erasing data augmentation. *arXiv preprint arXiv:1708.04896*, 2017.

Appendices

A Main code

Here I will provide some main parts of codes in this project. The complete project is provided on my GitHub.

```
# Basic residual block
class BasicBlock(nn.Module):
    expansion = 1

    def __init__(self, inplanes, outplanes, stride=1, downsample=None):
        super(BasicBlock, self).__init__()
        self.conv1 = nn.Conv2d(inplanes, outplanes, kernel_size=3, stride=stride,
                               padding=1, bias=False)
        self.bn1 = nn.BatchNorm2d(outplanes)
        self.relu = nn.ReLU(inplace=True)  # save a little memory usage
        self.conv2 = nn.Conv2d(outplanes, outplanes, kernel_size=3, stride=stride,
                               padding=1, bias=False)
        self.bn2 = nn.BatchNorm2d(outplanes)
        self.downsample = downsample   # a method for downsampling
        self.stride = stride

    def forward(self, x):
        # (identity) shortcut connection
        residual = x

        # Here dim represents height x width, size means dim x depth (number of feature maps)
        # 3x3 convolutional layer with padding of 1
        # output_dim = (input_dim - kernel_size + 2 * padding) / stride + 1
        output = self.conv1(x)
        output = self.bn1(output)
        output = self.relu(output)

        # 3x3 convolutional layer with padding of 1
        # output_dim = (input_dim - kernel_size + 2 * padding) / stride + 1
        output = self.conv2(output)
        output = self.bn2(output)

        # Apply downsampling (if not None) to make sure the size remain the same
        # With this operation, the shortcut is no longer identity mapping
        if self.downsample is not None:
            residual = self.downsample(x)

        output += residual
```

```
39          output = self.relu(output)
40
41          return output
42
43  # Deeper bottleneck architecture (for more than 50 layers)
44  # for the concerns on the training time
45  class Bottleneck(nn.Module):
46      # With Bottleneck architecture, there are 3 convolutional layers
        and the depth of
47      # the last layer is 4 times of the first two layers
48      expansion = 4
49
50      def __init__(self, inplanes, outplanes, stride=1, downsample=None):
51          super(Bottleneck, self).__init__()
52          self.conv1 = nn.Conv2d(inplanes, outplanes, kernel_size=1, bias
        =False)
53          self.bn1 = nn.BatchNorm2d(outplanes)
54          self.relu = nn.ReLU(inplace=True)
55          self.conv2 = nn.Conv2d(outplanes, outplanes, kernel_size=3,
        stride=stride,
56                                 padding=1, bias=False)
57          self.bn2 = nn.BatchNorm2d(outplanes)
58          self.conv3 = nn.Conv2d(outplanes, outplanes * self.expansion,
        kernel_size=1, bias=False)
59          self.bn3 = nn.BatchNorm2d(outplanes * self.expansion)
60          self.downsample = downsample
61          self.stride = stride
62
63      def forward(self, x):
64          # (identity) shortcut connection
65          residual = x
66
67          # Here dim represents height x width, size means dim x depth (
        number of feature maps)
68          # 1x1 convolutional layer, output_dim = input_dim
69          output = self.conv1(x)
70          output = self.bn1(output)
71          output = self.relu(output)
72
73          # 3x3 convolutional layer with padding of 1
74          # output_dim = (input_dim - kernel_size + 2 * padding) / stride
        + 1
75          output = self.conv2(output)
76          output = self.bn2(output)
77          output = self.relu(output)
78
79          # 1x1 convolutional layer, output_dim = input_dim
80          output = self.conv3(output)
81          output = self.bn3(output)
82
83          # Apply downsampling (if not None) to make sure the size remain
        the same
```

```
84          # With this operation, the shortcut is no longer identity
    mapping
85          if self.downsample is not None:
86              residual = self.downsample(x)
87
88          output += residual
89          output = self.relu(output)
90
91          return output
92
93  # Residual network construction
94  class ResNet(nn.Module):
95
96      def __init__(self, block_style, num_layers, num_classes=1000):
97          self.inplanes = 64
98          super(ResNet, self).__init__()
99          self.conv1 = nn.Conv2d(3, 64, kernel_size=7, stride=2,
100                                 padding=3, bias=False)
101         self.bn1 = nn.BatchNorm2d(64)
102         self.relu = nn.ReLU(inplace=True)
103         self.maxpool = nn.MaxPool2d(kernel_size=3, stride=2, padding=1)
104         self.layer1 = self._make_layer(block_style, 64, num_layers[0])
105         self.layer2 = self._make_layer(block_style, 128, num_layers[1],
      stride=2)
106         self.layer3 = self._make_layer(block_style, 256, num_layers[2],
      stride=2)
107         self.layer4 = self._make_layer(block_style, 512, num_layers[3],
      stride=2)
108         self.avgpool = nn.AvgPool2d(7, stride=1)
109         self.fc = nn.Linear(512 * block_style.expansion, num_classes)
110
111         # initialization
112         for m in self.modules():
113             if isinstance(m, nn.Conv2d):
114                 nn.init.kaiming_normal_(m.weight, mode='fan_out',
    nonlinearity='relu')
115             elif isinstance(m, nn.BatchNorm2d):
116                 nn.init.constant_(m.weight, 1)
117                 nn.init.constant_(m.bias, 0)
118
119     def _make_layer(self, block_style, outplanes, num_layers, stride=1)
    :
120         downsample = None
121
122         # Define the downsample method for shortcut connection to make
    sure the sizes of
123         # tensors remain the same (shortcut no longer performs identity
     mapping)
124         # This corresponds to option B of the original paper
125         if stride != 1 or self.inplanes != outplanes * block_style.
    expansion:
126             downsample = nn.Sequential(
```

```
                nn.Conv2d(self.inplanes, outplanes * block_style.expansion,
                          kernel_size=1, stride=stride, bias=False),
                nn.BatchNorm2d(outplanes * block_style.expansion),
            )

        layers = [block_style(self.inplanes, outplanes, stride, downsample)]
        # The number of input channels of the remain parts changed
        self.inplanes = outplanes * block_style.expansion

        for i in range(1, num_layers):
            layers.append(block_style(self.inplanes, outplanes))

        return nn.Sequential(*layers)

    def forward(self, x):
        x = self.conv1(x)
        x = self.bn1(x)
        x = self.relu(x)
        x = self.maxpool(x)

        x = self.layer1(x)
        x = self.layer2(x)
        x = self.layer3(x)
        x = self.layer4(x)

        x = self.avgpool(x)
        x = x.view(x.size(0), -1)
        x = self.fc(x)

        return x

# 50-layers ResNet
# Notice that the block style change to Bottleneck for the rest models
def resnet50(pretrained=False, **kwargs):

    model = ResNet(Bottleneck, [3, 4, 6, 3], **kwargs)

    if pretrained:
        model.load_state_dict(model_zoo.load_url(model_urls['ResNet50']))
    return model
```

Listing 1: ResNet

```
# Define the initialization methods for new fc and classification layer
def init_fc(m):
    classname = m.__class__.__name__

    if classname.find('Conv') != -1:
        init.kaiming_normal_(m.weight.data, a=0, mode='fan_in')
    elif classname.find('Linear') != -1:
        init.kaiming_normal_(m.weight.data, a=0, mode='fan_out')
```

```
9       # Fill the batch normalization layer with random values (mean = 1
        and std = 0.02)
10      # and set the bias to 0
11      elif classname.find('BatchNorm1d') != -1:
12          init.normal_(m.weight.data, 1.0, 0.02)
13          init.constant_(m.bias.data, 0.0)
14
15
16  def init_classifier(m):
17      classname = m.__class__.__name__
18
19      # Fill the linear layer with random values (mean = 1 and std =
        0.02)
20      # and set the bias to 0
21      if classname.find('Linear') != -1:
22          init.normal_(m.weight.data, std=0.001)
23          init.constant_(m.bias.data, 0.0)
24
25
26  # Define the new fully-connected and classifier layers
27  class NewBlock(nn.Module):
28      def __init__(self, input_dim, num_classes, dropout=True, relu=True,
        num_bottlenecks=512):
29          super(NewBlock, self).__init__()
30          new_fc = []
31          new_fc += [nn.Linear(input_dim, num_bottlenecks)]
32          new_fc += [nn.BatchNorm1d(num_bottlenecks)]
33          if relu:
34              # Use leaky ReLU as activation function
35              new_fc += [nn.LeakyReLU(0.1)]
36          if dropout:
37              # Use dropout with 0.5 probability
38              new_fc += [nn.Dropout(p = 0.5)]
39
40          new_fc = nn.Sequential(*new_fc)
41          new_fc.apply(init_fc)
42
43          new_classifier = []
44          new_classifier += [nn.Linear(num_bottlenecks, num_classes)]
45          new_classifier = nn.Sequential(*new_classifier)
46          new_classifier.apply(init_classifier)
47
48          self.new_fc = new_fc
49          self.new_classifier = new_classifier
50
51      def forward(self, x):
52          x = self.new_fc(x)
53          x = self.new_classifier(x)
54
55          return x
56
57  # Define the base ResNet50 model
58  class ResNet50Baseline(nn.Module):
```

```
    def __init__(self, num_classes):
        super(ResNet50Baseline, self).__init__()
        # Use pretrained ResNet50 as baseline
        baseline = resnet50(pretrained=True)
        # Modify the average pooling layer
        baseline.avgpool = nn.AdaptiveAvgPool2d((1, 1))

        self.model = baseline
        self.classifier = NewBlock(2048, num_classes)

    def forward(self, x):
        x = self.model.conv1(x)
        x = self.model.bn1(x)
        x = self.model.relu(x)
        x = self.model.maxpool(x)

        x = self.model.layer1(x)
        x = self.model.layer2(x)
        x = self.model.layer3(x)
        x = self.model.layer4(x)

        x = self.model.avgpool(x)
        x = torch.squeeze(x)
        x = self.classifier(x)

        return x
```

Listing 2: model

```
    # Training function
    def train_process(model, criterion, optimizer, scheduler,
    num_epochs):
        # Record the training time
        begin_time = time.time()

        best_weights = model.state_dict()
        best_acc = 0.0

        for epoch in range(num_epochs):
            print('Epoch {} of {}'.format(epoch + 1, num_epochs))
            print('-' * 15)

            # train and validation modes respectively
            for mode in ['train', 'val']:
                if mode == 'train':
                    scheduler.step()
                    model.train(True)
                else:
                    model.train(False)

                loss = 0.0
                num_corrects = 0

                # Iterate over the train and validation data
```

```
25                 for data in data_itr[mode]:
26                     inputs, labels = data
27                     # Data format is [batch_size, channels, height,
   width]
28                     now_batch_size, c, h, w = inputs.shape
29                     if now_batch_size == 1:
30                         continue
31 
32                     # If GPU is available, map the data to GPU
33                     if use_gpu:
34                         inputs = Variable(inputs.cuda())
35                         labels = Variable(labels.cuda())
36                     else:
37                         inputs = Variable(inputs)
38                         labels = Variable(labels)
39 
40                     # Zero the gradients of parameters
41                     optimizer.zero_grad()
42 
43                     # Forward process
44                     outputs = model(inputs)
45                     _, predictions = torch.max(outputs.data, 1)
46                     entropy_loss = criterion(outputs, labels)
47 
48                     # Backward process, (only in train mode)
49                     if mode == 'train':
50                         entropy_loss.backward()
51                         optimizer.step()
52 
53                     loss += entropy_loss.item()
54                     num_corrects += torch.sum(predictions == labels.
   data)
55 
56                 epoch_loss = loss / data_size[mode]
57                 epoch_acc = num_corrects.item() / data_size[mode]
58 
59                 print('{} Loss: {:.4f}, Accuracy: {:.4f}'.format(mode,
   epoch_loss, epoch_acc))
60 
61                 y_loss[mode].append(epoch_loss)
62                 y_err[mode].append(1 - epoch_acc)
63 
64                 if mode == 'val':
65                     last_weights = model.state_dict()
66                     if epoch%10 == 9:
67                         # Save the trained networks
68                         save_network(model, epoch + 1)
69                     draw_curve(epoch + 1)
70 
71             print()
72 
73         end_time = time.time()
74         used_time = end_time - begin_time
```

```
75          print('Training process completes in {:.0f}m {:.0f}s'.format(
76              used_time // 60, used_time % 60
77          ))
78
79          model.load_state_dict(last_weights)
80          save_network(model, 'last')
81
82          return model
83
84      model = ResNet50Baseline(len(class_names))
85      print(model)
86      if use_gpu:
87          model = model.cuda()
88      # Use cross entropy loss
89      criterion = nn.CrossEntropyLoss()
90
91      # Decide how to optimize the loss function (SGD here)
92      ignored_params = list(map(id, model.model.fc.parameters())) + list(
        map(id, model.classifier.parameters()))
93      base_params = filter(lambda p: id(p) not in ignored_params, model.
        parameters())
94      # Stochastic gradient descent with momentum
95      # Set the learning rate of pretrained parameters as 0.01
96      # Set the learning rate of the other parameters as 0.1
97      optimizer_ft = optim.SGD([
98          {'params': base_params, 'lr': 0.01},
99          {'params': model.model.fc.parameters(), 'lr': 0.1},
100         {'params': model.classifier.parameters(), 'lr': 0.1}
101     ], weight_decay=5e-4, momentum=0.9, nesterov=True)
102
103     # Sets the learning rate of each parameter group to the initial lr
104     # decayed by gamma every step_size epochs
105     exp_lr_scheduler = lr_scheduler.StepLR(optimizer_ft, step_size=40,
        gamma=0.1)
106
107     dir_name = os.path.join('./model', model_name)
108     if not os.path.isdir(dir_name):
109         os.mkdir(dir_name)
110
111     # save the arguments to a json file
112     with open('%s/args.json' % dir_name, 'w') as fp:
113         json.dump(vars(ag), fp, indent=1)
114
115     # Train and validate for 60 epochs
116     model = train_process(model, criterion, optimizer_ft,
        exp_lr_scheduler, num_epochs=num_epoch)
```

Listing 3: train