Federal University of Bahia
Programme of Post-graduation in Mechatronics

On deeply learning features for automatic person image re-identification

Alexandre da Costa e Silva Franco

2016

On deeply learning features for automatic person image re-identification

Alexandre da Costa e Silva Franco

Submitted in partial fulfillment of
the requirements for the degree of
PhD in Mechatronics

Programme of Post-graduation in Mechatronics
Federal University of Bahia

under supervision of
Prof. Dr. Luciano Rebouças de Oliveira

F825 Franco, Alexandre da Costa e Silva.

On deeply learning features for automatic person image re-identification/ Alexandre da Costa e Silva Franco. – Salvador, 2016.

103 f. : il. color.

Orientador: Prof. Dr. Luciano Rebouças de Oliveira.

Tese (doutorado) – Universidade Federal da Bahia. Escola Politécnica, 2016.

1. Inteligência artificial. 2. Visão Computacional. 3. Aprendizagem de Máquina. I. Oliveira, Luciano Rebouças de. II. Universidade Federal da Bahia. III. Título.

CDD: 006.37

TERMO DE APROVAÇÃO

ALEXANDRE DA COSTA E SILVA FRANCO

ON DEEPLY LEARNING FEATURES FOR AUTOMATIC PERSON IMAGE RE-IDENTIFICATION

Tese aprovada como requisito parcial para obtenção do grau de Doutor em Mecatrônica, Universidade Federal da Bahia, pela seguinte banca examinadora:

Luciano Rebouças de Oliveira – Orientador Oliveira
Doutor em Engenharia Elétrica e de Computadores, Instituto de Sistemas e Robótica da Universidade de Coimbra, Portugal
Universidade Federal da Bahia

Leizer Schnitman – Membro interno Leizer Schnitman.
Doutor em Engenharia Eletrônica e Computação - Instituto Tecnológico de Aeronáutica
Universidade Federal da Bahia

Rubisley de Paula Lemes – Membro interno
Doutor em Informática - Universidade Federal do Paraná
Universidade Federal da Bahia

Angelo Conrado Loula – Membro externo Angelo Loula
Doutor em Engenharia Elétrica – Universidade Estadual de Campinas
Universidade Estadual de Feira de Santana

João Paulo Papa – Membro externo João Paulo Papa
Doutor em Ciência da Computação - Universidade Estadual de Campinas
Universidade Estadual Paulista Júlio de Mesquita Filho

Salvador, 13 de maio de 2016

To my mother, my father, my wife and my children

Acknowledgments

During this journey a lot of people contributed to turn my work feasible. I would like to thank all of them by the provided help, and to have been beside me during the difficult times. Among these people are the research colleagues and professors from Ivision Lab. Specially, I would like to express my particular thanks to my advisor Luciano who I owe all the given support and a friendly relationship.

To my kids Alyssa, Felipe and Matheus and to my wife Priscilla, my deep appreciation for the patience and comprehension when I had been missing. I need to say that all of you are the reason of my life.

Finally, thanks to my parents, Simone Tereza e Antonio Carlos, who supported me in all my decisions and for the love and appreciation I have for them.

Abstract

The automatic person re-identification (re-id) problem resides in matching an unknown person image to a database of previously labeled images of people. Among several issues to cope with this research field, person re-id has to deal with person appearance and environment variations. As such, discriminative features to represent a person identity must be robust regardless those variations. Comparison among two image features is commonly accomplished by distance metrics. Although features and distance metrics can be handcrafted or trainable, the latter type has demonstrated more potential to breakthroughs in achieving state-of-the-art performance over public data sets. A recent paradigm that allows to work with trainable features is deep learning, which aims at learning features directly from raw image data. Although deep learning has recently achieved significant improvements in person re-identification, found on some few recent works, there is still room for learning strategies, which can be exploited to increase the current state-of-the-art performance.

In this work a novel deep learning strategy is proposed, called here as coarse-to-fine learning (CFL), as well as a novel type of feature, called convolutional covariance features (CCF), for person re-identification. CFL is based on the human learning process. The core of CFL is a framework conceived to perform a cascade network training, learning person image features from generic-to-specific concepts about a person. Each network is comprised of a convolutional neural network (CNN) and a deep belief network denoising autoenconder (DBN-DAE). The CNN is responsible to learn local features, while the DBN-DAE learns global features, robust to illumination changing, certain image deformations, horizontal mirroring and image blurring. After extracting the convolutional features via CFL, those ones are then wrapped in covariance matrices, composing the CCF. CCF and flat features were combined to improve the performance of person re-identification in comparison with component features. The performance of the proposed framework was assessed comparatively against 18 state-of-the-art methods by using public data sets (VIPeR, i-LIDS, CUHK01 and CUHK03), cumulative matching characteristic curves and top ranking references. After a thorough analysis, our proposed framework demonstrated a superior performance.

Contents

List of Tables

List of Figures

List of Acronyms

CFL	*Coarse-to-fine learning*	4
CCF	*Convolutional covariance features*	4
CNN	*Convolutional neural network*	4
DBN	*Depp belief network*	4
DAE	*Denoising autoencoder*	4
CMC	*Cumulative matching characteristics*	14
RBM	*Restricted boltzmann machine*	15
MLP	*Multi-layer perceptron*	17
PCA	*Principal component analysis*	42
FCL	*Fine-to-coarse learning*	45
DML	*Deep metric learning*	51
SCNR	*Semantic color names and rankboost*	51
SDALF	*Symmetric-driven accumulation of local features*	51
DTR	*Domain transfer support vector ranking*	51
IDLA	*Improved deep learning architecture*	51
LAFT	*Locally aligned feature transformation*	51
RPLM	*Relaxed pairwise learned metric*	51
DFLRDC	*Deep feature learning with relative distance comparison*	51
LMNN	*Large margin nearest neighbor*	51
MCC	*Metric learning by collapsing classes*	51
HSDC	*Hybrid spatiogram and covariance descriptor*	66
DDC	*Descriptive and discriminative classification*	66

List of Symbols

$\|\cdot\|_d$	L_d-norm distance	13
Pos	Set of positive label training samples	13
Neg	Set of negative label training samples	13
$\sigma(.)$	Sigmoid function	17
S^l	Output of the l-th CNN layer	17
V^l	Output of the neuron activation functions of the l-th CNN layer	17
W	Network weight matrices	17
$E(.)$	Energy function	18
$P(.)$	Probability function	18
$\mathrm{argmax}_x f(x)$	The value of the x argument that maximize the f function.	18
$\mathbf{v}$	RBM visible layer	18
$\mathbf{h}$	RBM hidden layer	18
$\mathbf{u}$	Bias vector of the DBN hidden layers	18
$\mathbf{z}$	Bias vector of the DBN visible layers	18
$\mathbf{x}$	Autoencoder input data	20
$\mathbf{x}'$	Autoencoder output data	20
$g(.)$	Decoder function	20
$r(.)$	Encoder function	20
$\mathcal{L}(.)$	Loss function	21
$\tilde{\mathbf{x}}$	Corrupted version of the input data $\mathbf{x}$	22
$\mathcal{L}(.)$	Contrastive loss function	35
$\rho(.)$	Riemannian distance	63
$\lambda(\mathbf{C}_1, \mathbf{C}_2)$	Generalized eigenvalues of $\mathbf{C}_1$ and $\mathbf{C}_2$	63

Part I

Presentation

CHAPTER 1

Introduction

Contents

Person re-identification (re-id) consists in identifying a person across a database of images, given a source image of that person. Due to the large variation of human appearance and pose, low-resolution images, different environments and illumination change, person re-identification systems have to deal with non-trivial tasks. A comprehensive review on person re-identification can be found in [Bedagkar-Gala and Shah, 2014].

One of the main challenges of a person re-id system is to design suitable features to compose image descriptors, which need to be invariant to person appearance variations. To deal with this problem, researchers have proposed several methods to conceive discriminant person image features. Person image features can be classified as handcrafted or trainable. Handcrafted features are based on image gradients, colors, texture or any other image filter. The main difficult to deal with handcrafted features is to find appropriated features to a specific problem. A kind of handcrafted feature can be suitable to describe an object image, but can be inappropriate to another one. Finding or designing suitable handcrafted features for person re-id is a non-trivial task. Trainable features, in turn, are learned to describe a set of specific image objects. Therefore, these kind of features are used to design adaptive descriptors for a specific domain of problem. Although trainable features are promising to design robust and discriminative image descriptors, strategies to learn these features still miss a thorough investigation.

Once person features are computed (manually or learned), they are used by a distance function to calculate the similarity among a pair of person images. Each unknown person image against all target known people should provide the image pairs to be compared. The pair with higher score of similarity is considered a pair containing two images of the same person. Similarity metrics are commonly based on distance functions. As the features, a distance function can be a pre-established or trainable. The advantage of trainable functions, is that they can have learnable parameters, trained in a supervised way, in order to maximize and minimize the data inter-class and intra-class variations, respectively. Although the strategy of combining multiple distances and compute the similarity metric among these distances is important to improve the person re-id prediction, a way to choose the best strategy should be further explored. Conventional distance function learning does not take into account the learning of trainable features.

A joint way to learn features in multiple levels of abstraction and also learn the parameters of a model (like a classifier or a distance function) is provided by the deep learning paradigm. The core of a deep learning is to train the data samples, directly from its raw data, without the need of handcrafted features. Deep learning can be performed by several kind of deep neural networks and by several learning strategies. Deep networks have a large number of parameters and therefore need to be trained with a large number of training samples. The amount of available data in person re-id data sets is usually not enough to avoid the network overfitting. Thus, some strategies should be provided to overcome this lack of data when the deep learning paradigm is chosen for person re-id. Among the strategies, data set augmentation with generated images, drop out and transfer learning are some examples of techniques to control the network overfitting. While deep learning has achieved significant performance in person re-id, there is a range of deep learning strategies to be exploited in order to achieved even better person re-id performance.

Novel strategies to learn features for person re-id are proposed in this work. Some aforementioned open issues, which will be discussed throughout this chapter, were exploited by the proposed learning strategies and a framework was conceived to implement them. The proposed framework was conceived to learn person features in multiple steps, following the reasoning of human learning process – from generic to specific concepts. Our proposed framework was called coarse-to-fine learning (CFL).

Deep learning paradigm showed an appropriated resource used to implement our framework by the following aspects: (i) features were learned from raw image data in multiple levels of abstraction; (ii) some deep networks were necessary to implement the steps of the generic-to-specific human learning process; and (iii) some intermediate deep layers were necessary to extract novel person descriptors, based on the covariance descriptors proposed by Tuzel et al. [2006].

1.1 Motivation

The core of our work has been led by two reasons:

- At higher level of abstraction, by the intuitive way on how the human learning occurs, from generic to specific concepts;
- At lower level of abstraction, by the high performance of the deep networks and high discriminative power of the covariance descriptors.

The two aforementioned motivations have driven to the development of CFL to learn person features by three deep networks, which mimic the human learning process (see Chapter 3). The deep features, learned in CFL, are wrapped into a new adaptive covariance descriptor, called convolutional covariance descriptor (CCF). The deep network topology is a proposed hybrid network comprised of a set of convolutional neural networks (CNN) and a pre-trained DBN-DAE, able to learn image local features (CNN) and noise-invariant global ones (DBN-DAE). The components of our framework and the learning strategy were chosen to properly mimics the generic to specific human learning process (see Chapter 4). Our proposed CFL was published in [Franco and Oliveira, 2016a], and discussions of the proposed CCF is under revision in [Franco and Oliveira, 2016b].

1.2 Goals

The goal of our proposed CFL is to improve the discrimination of person features for person re-id by following a novel learning strategy in multiple steps, and it is implemented by a cascade hybrid deep network training.

Specifically, the goals of our work are:

- Conception of a machine learning strategy to mimic the human learning process from generic to specific concepts;
- Improvement of the discrimination of person features by the proposed CCF;
- Provide a new metric to measure the best configuration of the distances and function used to compute a similarity function;
- Provide a new learning approach that can be updated to solve other computer vision problems.

1.3 Key contributions

Our work brings three main contributions:

- A machine transfer learning approach motivated by the human skill of obtaining coarse-to-fine knowledge;

- A novel way to wrap convolutional features in convariance matrices, and the integration of those features with deep features;

- A new metric to measure the prediction performance that can be used to select the best network configurations without the need to perform the person re-id matching.

In the second contribution, CCF are integrated with the flat features of the top layer of our hybrid network. Both features can not be fused in a single array, since the covariance matrices from CCF and the flat features lie on different spaces. Then the integration of those features was accomplished via a proposed set of score distance functions, computed among CCF and the flat features. A final similarity function provides the final score, given a set of CCF and flat feature distances (see Section 4.3).

While the original covariance descriptor [Tuzel et al., 2006] extracts covariance matrices over a set of image maps created by fixed image operations (intensity, color, gradient, filter responses, etc), the CCF is achieved by a set of local covariance matrices over the feature maps of the CNN inside the hybrid deep network [Franco and Oliveira, 2016a]. Since the feature maps of the CNN layers are optimized by the training phase of the network, CCF wrap optimized features, which are expected to be more robust than those original covariance descriptors found in [Tuzel et al., 2006].

In the last contribution, a new proposed metric gives an indicative of the prediction performance computed over a relative small amount of person image pairs. Although it is necessary to perform the training and prediction of the network, it is not necessary to perform all pair comparisons to obtain the real top rank performance.

1.4 Chapter map

The reminder of this thesis is organized as follows.

- Chapter 2 presents the background of person re-identification system and essential issues to comprehend our proposed work and their relations.

- Chapter 3 discusses the CFL framework and component parts of its architecture, as well as, the framework of learning and prediction phases are presented. An experimental analysis is carried out to evaluate the performance of CFL, as well as, a comparison with 16 other state-of-the-art methods.
- Chapter 4 describes CCF as an extension of CFL approach, and details of CCF and the flat features integration are shown in this chapter. An experimental analysis is performed in order to evaluate the performance of CCF. A comparison with 18 other state-of-the-art methods is accomplished as well.
- Chapter 5 presents the conclusion, discussion and future works.

CHAPTER 2

Background

Contents

This chapter introduces some background in the field of person re-id. As the proposed method for person re-id pervasively exploits deep learning, theory about that paradigm is also presented and discussed. Also, some details about data sets used in the performance assessment of person re-id methods are given, as well as, some performance measure techniques related to person re-id evaluation are discussed. Finally, an analysis in what extension the works cited throughout the chapter is related to our work is done.

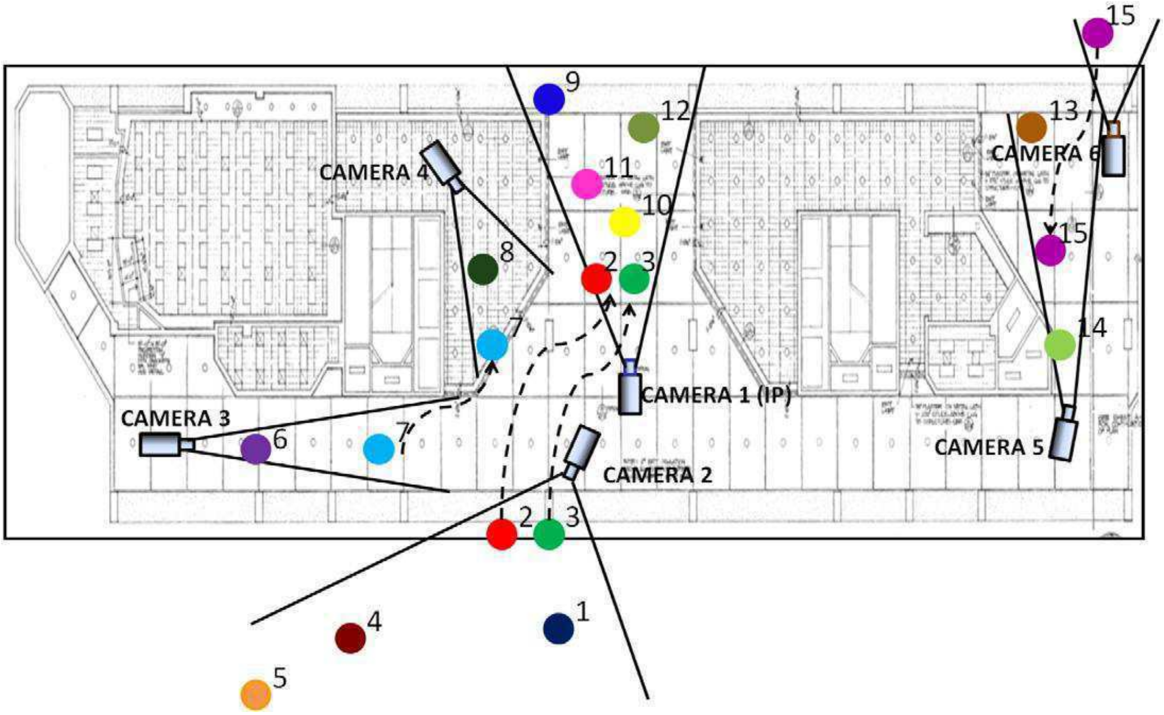


Figure 2.1: Multi-camera surveillance network illustration. An example of a person re-id system that can be applied in the scenario. The system identifies people who walk by the entrance cameras (2 and 6), and should be able to re-identify the same people labeled throughout the halls. Image taken from Bedagkar-Gala and Shah [2014].

2.1 Automatic re-identification

One scenario of application in person re-id is depicted in Fig. 2.1. At the scene in the figure, a multi-camera surveillance system identifies the people who walk by the cameras 2 or 6. By considering all cameras in the network, the re-id system should be able to re-identify a person who cross through the halls, and were initially labeled in the cameras located at the main entrances [Bedagkar-Gala and Shah, 2014]. Basically, an automatic person re-id system consists in: Given a set of target person images and a source one; considering that the target images have already been identified, the re-id system should be able to find the target image which is closer to the source one in terms of image similarity. Figure 2.2 describes the steps for a conventional person re-id system. Image features are used as unique person descriptors, being extracted on each pair of source and target images in order to be matched via a similarity measure. Each source person descriptor is compared to each target one with the goal of finding the highest similarity pair.

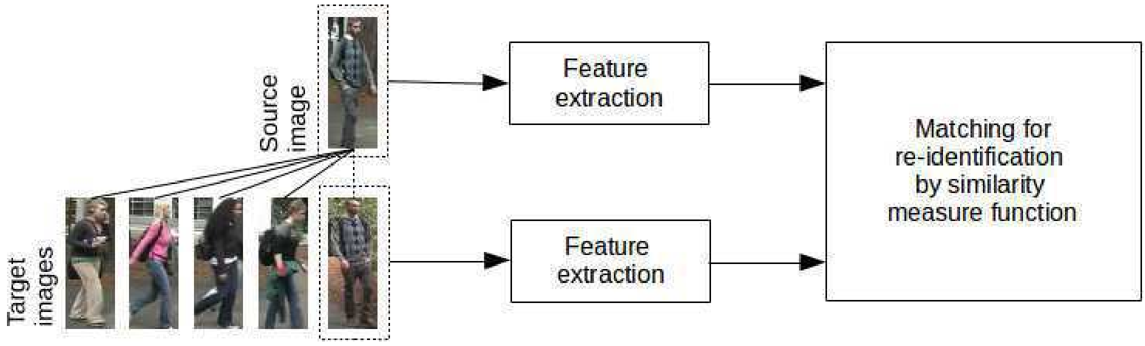

Source image
Target images
Feature extraction
Feature extraction
Matching for re-identification by similarity measure function

Figure 2.2: Pipeline of a conventional person re-id system. A source image is compared with labeled target images presented in the database. The pair with highest similarity is chosen by the re-id system.

2.1.1 Challenges

Although there are some reasons to explain the difficulty of person re-id, the main issues come from the image variations found in the wild scenarios (camera view point, lighting change, different image poses, distance from the camera, different camera perspective, and so forth), and person's appearance across different cameras and environments. Person re-id systems usually suffer from non-structured environments and non-rigid image objects. As such, the main challenge turns to find a person descriptor which represents the image person regardless its visual variations.

Another person re-id challenge is the comparison among two person descriptors. Appearance of the same person can have significant changes if he/she holds a bag, or dresses a jacket whose fabric has different appearance across cameras. In that case, due to different camera perspectives, the same person images can appear to be different, or different person images can appear to be similar. This implies that within-class variation can be large, while inter-class variation can be relatively smaller. Figure 2.3 shows examples of images of the same people with high visual variations, and different people with similar visual appearance.

2.1.2 Image person descriptors

Since one of the greatest person re-id challenge is the choice of the descriptor, one should spend relatively time to design discriminative functions to represent an image person uniquely. There are two kinds of discriminative functions which can be used as image descriptor for person re-id: handcrafted or trainable [Bedagkar-Gala and Shah, 2014]. The former can still be classified as appearance or soft-biometric descriptor. Appearance descriptors [Satta, 2013] are commonly based on the colors and the texture of clothes [Bazzani et al., 2013], and person silhouette [Wang

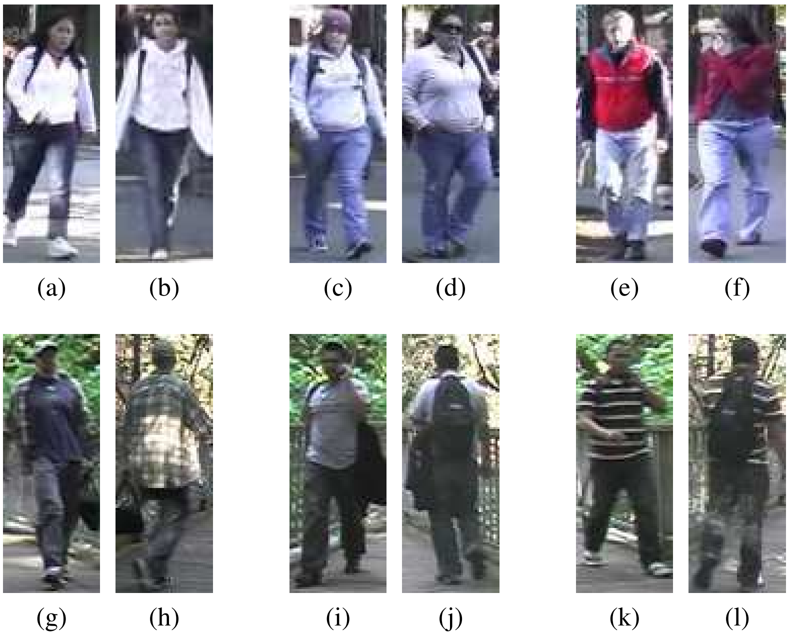


Figure 2.3: Different pairs of people with similar appearance in (a) – (b), (c) – (d) and (e) – (f). The same pairs of people with different appearance in (g) – (h), (i) – (j) and (l) – (m).

et al., 2007]; they can be generated by color [Cheng et al., 2011], [Gheissari et al., 2006], [Kuo et al., 2013], gradient [Dalal and Triggs, 2005], [Bak et al., 2010], [Zheng et al., 2009] and texture [Zheng et al., 2013], [Ma et al., 2012a] histograms, and shape-based features [Huynh and Stanciulescu, 2015]. Soft-biometric features like skin, hair and eye colour are alternative to distinguish different image people [Dantcheva et al., 2010]. While handcrafted descriptors are obtained by applying fixed image operations, trainable descriptors are used to represent dynamically the appearance of a person, or learning the weights of a linear combination of features [Schwartz and Davis, 2009], [Gray and Tao, 2008].

Regardless the choice of the kind of descriptor, fusion of features still takes place on integrating complementary or opposite image features. This is done in order to achieving superior results in comparison with the component features, representing the final descriptor itself. Descriptors based on fusion of features have demonstrated prediction improvement on person re-id [Liu et al., 2012]. The problem of this approach is that descriptor dimension can be exceedingly high, depending on the number of features [Mangaia et al., 2014]. To cope with this issue, Porikli and Kocak [2006] proposed the covariance matrices as image descriptors [Tuzel et al., 2006], [Eiselein et al., 2014], [Hirzer et al., 2011], [Zeng et al., 2015] and [Dultra et al., 2013]. Covariance descriptors demonstrated to be an appropriated way to encode the relationship among a set of features in a low dimensional space, this achieving superior performance in many object detection tasks [Tuzel et al., 2006], Porikli and Kocak [2006], [Tabia et al., 2014], [Romero et al., 2013], [Fehr et al., 2012], [Ma et al., 2012b].

2.1.3 Similarity measures

After having the person descriptor selected, the similarity between the pair of source and target images should be obtained by a similarity function, *e.g.*, a distance-based metric. Distance metrics can be either used as a fixed distance function or a learned one. Yang [2006] states the aim of learning a distance function is to find metrics which are small for data points within the same classes, and large for data points of different ones. Some works that propose learned distance metrics can be found in [Bedagkar-Gala and Shah, 2014]. [Weinberger et al., 2006], [Hirzer et al., 2012], [Xiong et al., 2014], [Globerson and Roweis, 2005], [Yi et al., 2014a], [Chen et al., 2015], [Paisitkriangkrai et al., 2015], [Li and Wang, 2013], [Liao et al., 2015], [Ma et al., 2013] and [Martinel et al., 2015]. A supervised learned distance function is generally given by

$$d(x_i, x_j) = (x_i - x_j)^T D(x_i - x_j)\,, \tag{2.1}$$

where x_1, x_2,...,x_n are the descriptors of the n training samples; $d(x_i, x_j)$ is a distance metric between two samples, and D is a symmetric positive, semi-definite matrix. This problem is solved using a convex programming, according to

$$\begin{aligned} \min_{D} \quad & \sum_{(x_i,x_j)\in Pos} \|x_i - x_j\|_D^2 \\ \text{Subject to} \quad & \\ & D \succcurlyeq 0, \\ & \sum_{(x_i,x_j)\in Neg} \|x_i - x_j\|_D^2 \geqslant 1 \end{aligned} \tag{2.2}$$

where, Pos and Neg denote pairs that belong to the same person or a different one, respectively.

Deep learning is another learnable-based descriptor approach that aims at acquiring a distance metric and a person descriptor, at the same time. Deep learning will be discussed in Section 2.2.

2.1.4 Performance measure

Person re-id is a ranking driven problem, since after computing a similarity measure between pairs of people, a ranking must be formulated for the top 1 to be chosen. Performance evaluation

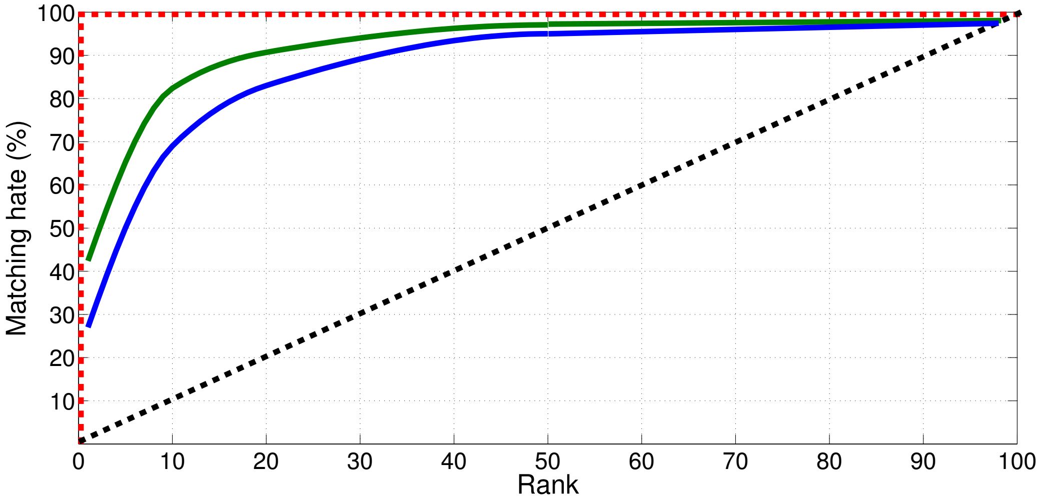


Figure 2.4: Examples of CMC curves (best viewed in color): red dashed – perfect curve, black dashed – random curve, green and blue curves: results from hypothetical methods.

in rank n consists in computing the hit rate of correct matches among each source image and n respective closer target ones. A correct matching is achieved when there is a target image correspondent to the source one, among the n targets.

Cumulative matching characteristics curves (CMC) are one of the most used metric to evaluate identification performance. By plotting the hit rate against the rank, the goal of a CMC graph is to provide an analysis of the ranking capability at an identification system. Figure 2.4 illustrates a perfect (red dashed line) and a random (black dashed line) examples of a CMC graph. In the former case, the CMC yield to a continuous value equals to 1, indicating that in any rank, a hypothetical proposed method always points to the correct match; in the latter case, an identification system is not better than the chance (dashed line, first bisectrix). The curve in green shows a better performance than that blue one, and it is closer to the perfect curve (in red). CMC help in the visual comparison of different methods.

Two other quantitative metrics used to evaluate person re-id performance are: Area under the CMC curve and top rank evaluation. The area under the CMC curve indicates the judgment of the identification system over the CMC curve (by computing the integral under the curve), and the top rank shows the top 1 performance of the system.

2.2 Deep learning

Deep learning is a novel machine learning paradigm conceived to learn concepts in a hierarchically nested way, where complex concepts are defined in relation to simpler one [I. Goodfellow and Courville, 2016]. One of the promises of deep learning is replacing handcrafted features with efficient algorithms for feature learning and hierarchical extraction [Song and Lee, 2013]. The ability to learn powerful features becomes increasingly important as the amount of data and range of machine learning applications continues to growing.

Figure 2.5 shows how a deep learning system can represent the concept of an image person by combining simpler concepts in a deep learning model. Network layers were trained to learning concepts as follow: (i) Visible layer – represents the input image pixels, the less abstract concept; (ii) first hidden layer – represents the edges of the input image, a more abstract concept then the previous layer; (iii) second hidden layer – represents the corners of the input image; (iv) third hidden layer – represents the object parts of the input image, and (v) the output layer – represents the identity of an object image, a more abstract concept. All the concepts are learned together and both classifier and features are optimized according to a specific problem domain.

A deep learning architecture is based on a multi-layer network with complex structures and a high number of network parameters. Convolutional Neural Networks (CNN) [LeCun et al., 1989], for example, are a kind of discriminative deep learning architecture, commonly used in image and video recognition (see Section 2.2.1). Restricted Boltzmann Machines (RBM) [Larochelle and Bengio, 2008] and Deep Belief Networks (DBN) [Hinton et al., 2006] are a generative deep model architecture, designed to learn a probability distribution over a set of inputs (see Section 2.2.3).

A deep network can be trained in supervised, unsupervised or semi-supervised ways. In this later training approach, a deep network is previously pre-trained in an unsupervised way, and then a supervised training to fine tune the network parameters is performed. An autoencoder is a kind of deep network conceived to learn a compressed representation of its input and/or to address the lack of sufficient data to learn [Hinton and Salakhutdinov, 2006] (see Section 2.2.4). An extension of an autoencoder, called denoising autoencoder (DAE), was introduced to learn noisy invariant image representation [Vincent et al., 2008]. In a DAE, an autoenconder is trained to reconstruct the input from its corrupted version (see Section 2.2.5).

Recently, deep learning has been adopted to solve several Computer Vision problems, such as in: Zeng et al. [2013], a jointly cascade training of a set of classifiers was proposed in order to perform a pedestrian detection; Masci et al. [2011], stacked Convolutional auto-encoders for hierarchical feature extraction were conceived; Luo et al. [2012], a deep network was trained to segment facial components; Ngiam et al. [2011], a multi-modal DBN was proposed to learn a sharing representation of a set of videos and its associated audio information; Zhu et al. [2013], a normalized representation of face images, learned by a CNN deep autoencoder, was created with the goal of generating face features invariant to pose and illumination change; Vincent

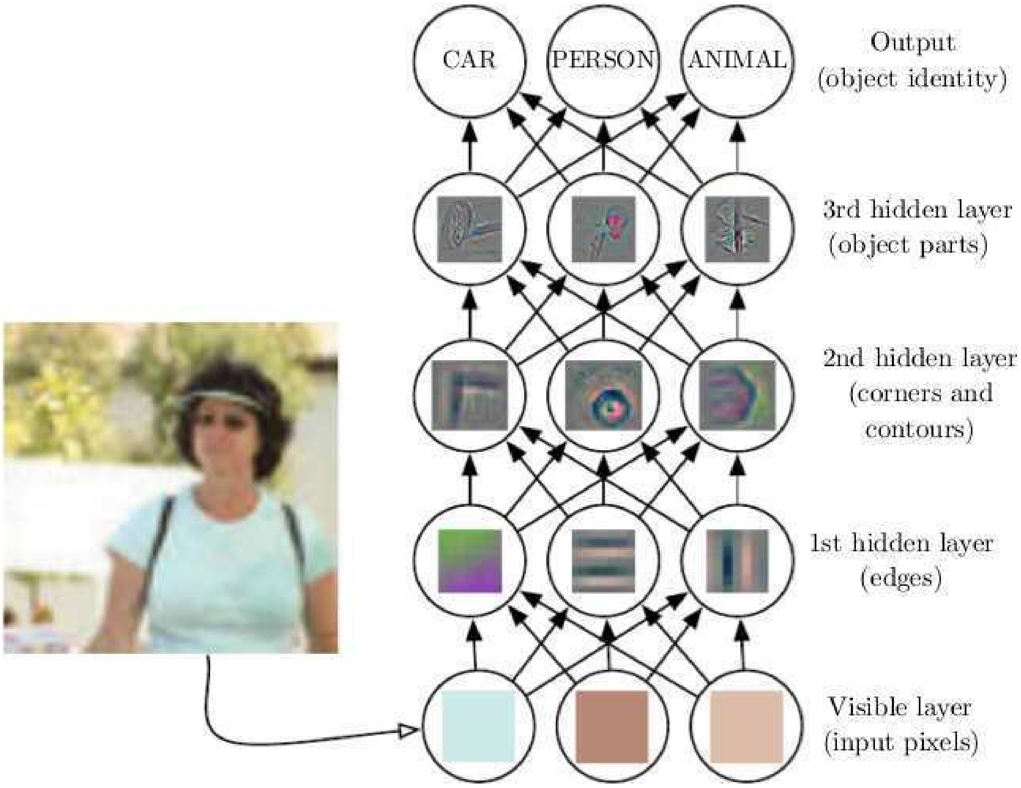


Figure 2.5: Concepts in multiple levels of abstraction, learned by a deep learning system. From the bottom to the top, less abstract concepts to more abstract ones are learned by the respective network layers: (i) Visible layers, representing the pixels of an image; (ii) first hidden layer, representing the edges of the input image; (iii) second hidden layer, representing the corner of the input image; (iv) third hidden layer, representing the object parts of the input image and (v) the output layer, representing the identity of an object image, a more abstract concepts. Image taken from I. Goodfellow and Courville [2016]

et al. [2010], a DAE was introduced to learn useful image representation; and Krizhevsky and Hinton [2011], a deep autoencoder was used to retrieve context-based image. In the context of person re-id, some deep network architectures have been introduced in [Yi et al., 2014a], [Yi et al., 2014b], [Ahmed et al., 2015], [Li et al., 2014], [Ding et al., 2015] and [Franco and Oliveira, 2016a], achieving state-of-the-art results on almost all evaluated data sets.

2.2.1 Convolutional Neural Networks

CNN are a kind of feed-forward, biologically inspired network, designed to emulate the behavior of an animal visual cortex [Hubel and Wiesel, 1968]. CNN are comprised of several layers which can be of three types:

- **Convolutional**: Consists in a set of feature maps, which are generated from a convo-

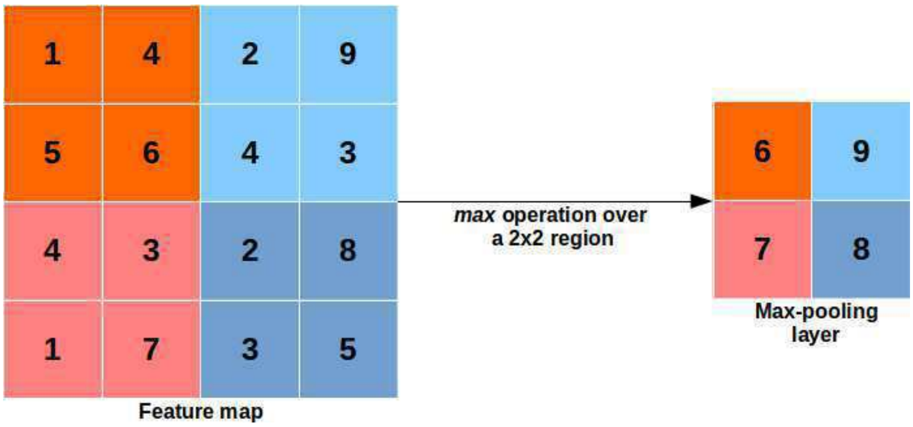


Figure 2.6: Example of a max-pooling layer obtained from an input or feature map.

lutional operation over the input data or other feature map. Each convolutional layer defines a data input representation in a certain level of abstraction. The convolutional filters, called kernels, are rectangular grids of shared weights, which are learned during the network training. Outputs S of each CNN layer are defined as

$$S^l_{(i,j,k)} = \sum_{u=1}^{m} \sum_{t=1}^{m} V^{l-1}_{(i+u,j+u,k)} W^l_k \,, \tag{2.3}$$

$$V^l_{(i,j,k)} = \sigma(S^l_{(i,j,k)}) \,, \tag{2.4}$$

where l is the output layer, i and j are image region coordinates of a feature map k, m is the size of squared region of the kernel, σ is a non-linear neuron activation function, and W represents the weight matrix of each output neuron.

- **Pooling**: The main goal is to perform a downsampling along the spatial dimension of a previous layer. The pooling operation is performed over non-overlapped regions, usually of size $2{\times}2$. A pooling operation returns a downsampled version of each non-overlapped region. One of the most commonly used pooling operation is the $\max$ function, which leads the layer to be called max-pooling. The spatial dimension operation of a layer reduces the number of network parameters, and, thus, tends to control network overfitting. Max-pooling operations also provide a way of translation invariance. Figure 2.6 illustrates a max-pooling layer, generated from an input or feature map.

- **Fully-Connected**: Fully-connected layers are the top layers of CNN, commonly used to encode the highest representation abstraction of the input data. They have full connections

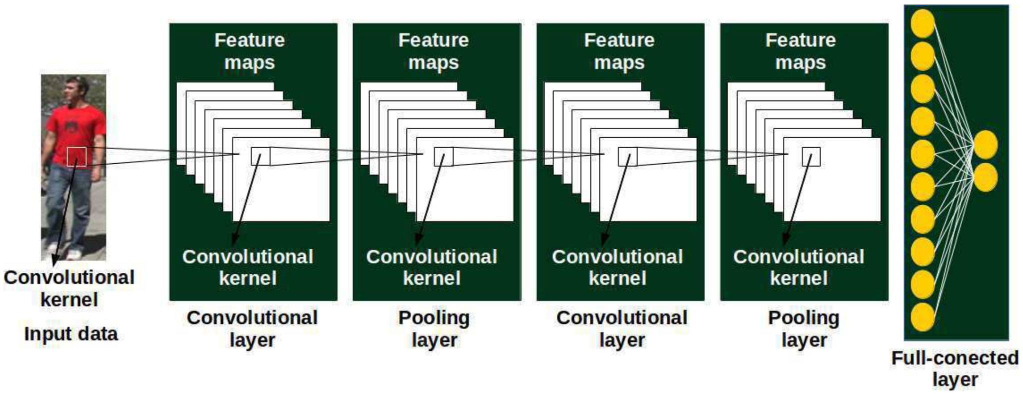


Figure 2.7: A hypothetical convolutional neural network model comprised of two convolutional, two max-pooling and two full-conected layers. This model was trained to recognize a person image. Figure adapted from Oliveira [2010]

to all neurons activations in the previous layer, as usually seen in multi-layer perceptron (MLP) networks.

Figure 2.7 illustrates a CNN model with two convolutional, two pooling and one fully-connected layers.

The standard MLP gradient descent algorithm evaluates the cost and gradient over the full training set. Due to the large amount of training data required, this algorithm is not suitable in the CNN training. An alternative algorithm and more appropriated to train a CNN is the stochastic gradient descent that uses only a single or few training examples.

2.2.2 Restricted Boltzmann Machines

RBM are undirected probabilistic graphical models proposed by Smolensky [1986], which are able to learn a probability distribution over a set of inputs. RBM are comprised of one layer of observable variables v and one hidden layer h (layer of latent variables). RBM are trained to maximize the product of the probability of a training set T, given by

$$\underset{\mathbf{W},\mathbf{z},\mathbf{u}}{\operatorname{argmax}} \prod_{\mathbf{v}\in T} P(\mathbf{v})\,, \tag{2.5}$$

where

$$P(\mathbf{v}) = \frac{1}{Z}\sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}\,, \tag{2.6}$$

, Z is a normalizing constant to ensure the probability distribution sums to 1, W is a network weight matrix, and $\mathbf{u}$ and $\mathbf{z}$ are bias vectors of the hidden and visible layers, respectively. The energy function E is defined as

$$E(\mathbf{v}, \mathbf{h}) = -\mathbf{u}^T\mathbf{v} - \mathbf{z}^T\mathbf{h} - \mathbf{v}^T\mathbf{W}\mathbf{h}\,. \tag{2.7}$$

Conditional probabilities of $P(\mathbf{h}|\mathbf{v})$ and $P(\mathbf{v}|\mathbf{h})$ are modeled by a product of Bernoulli distributions, according to

$$P(\mathbf{h}_i = 1|\mathbf{v}) = \sigma(\mathbf{u}_i + \mathbf{W}_i\mathbf{v}) \tag{2.8}$$

and

$$P(\mathbf{v}_j = 1|\mathbf{h}) = \sigma(\mathbf{z}_j + \mathbf{W}_j^T\mathbf{h})\,, \tag{2.9}$$

where $\sigma(.)$ is a $sigmoid$ function, and j and i are indexes of the visible and hidden layer units, respectively.

2.2.3 Deep Belief Networks

DBN are comprised of a set of stacked RBMs. The visible layer of an RBM is the output of the previous one, or it is the input layer of a DBN, which is trained in two steps: (i) An unsupervised training and (ii) a supervised training. The first step is performed by a cascade layer-wise training of each RBM, in n stages, where n is the number of RBM. At the first stage, the input layer of the first RBM is the DBN data input; when the first RBM is trained, a second RBM is stacked on the top of the first one – the output of the first RBM becomes the input layer of the second one. In the second step, when the new RBM is already trained, the weights of the first one is fine tuned. In the same way, latter stages follow the first and second ones. In the second DBN training step, the weights of the overall DBN are fine-tuned in other domain of the training problem.

During the first step, a DBN can learn how to probabilistically reconstruct its inputs. Then layers act as feature detectors over the inputs. In the second step, the supervised training is carried out to perform a classification. Figure 2.8 describes the steps of a DBN training in order to teach a classify how to distinguish which image is a person from those ones that are not; the left box depicts the first DBN training step, while the right box illustrates the second one.

2.2.4 Autoencoder

An autoencoder is a kind of neural network that is trained to reconstruct a copy of its input in its output. It is designed to perform feature extraction, dimensionality reduction or data compression. Internally, an autoencoder has a hidden layer $\mathbf{h}$ that encodes the input. The

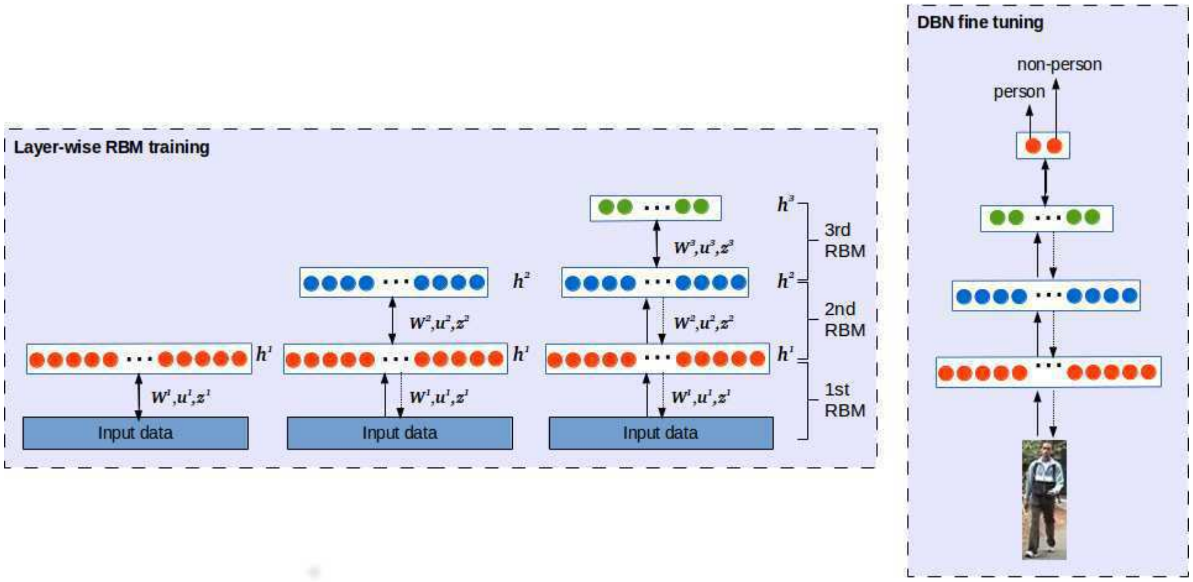


Figure 2.8: Training stages of a deep belief network. In the first stage (left box), a cascade layer-wise unsupervised training of a set of RBM is performed. In the second stage (right box), the deep belief network is fine tuned in a supervised way.

layer h should hold only the useful properties of the input. There are two kind of layers in an autoencoder network topology: The encoder, which maps the network input to a compressed representation of itself (represented by $\mathbf{h}$), and the decoder, which reconstructs an approximated version of the network input from h (see Fig. 2.9).

In the simplest form, the architecture of an autoencoder is a non-recurrent neural net which is very similar to the MLP, comprised of an input layer, an output layer, and one or more hidden layers, connecting input and output. The difference between autoencoders and MLPs is that an autoencoder must own a symmetric structure. In other words, the decoder layers of an autoenconder should be a mirrored version of the encoder ones, with their weight matrices being the transpose of the weight matrix of the encoder layers. Considering the encoder and decoder as a function r and g, respectively, the autoencoder should compute the matrix $\mathbf{W}$, corresponding to the weights of the network,

$$\mathbf{W} = \underset{\mathbf{W}}{\operatorname{argmin}} \left\| \mathbf{x} - g(r(\mathbf{x}, \mathbf{W}), \mathbf{W}^T) \right\| , \tag{2.10}$$

where x is the data input of the network.

In the simplest case, an autoencoder has only one hidden layer. In this case, it takes the

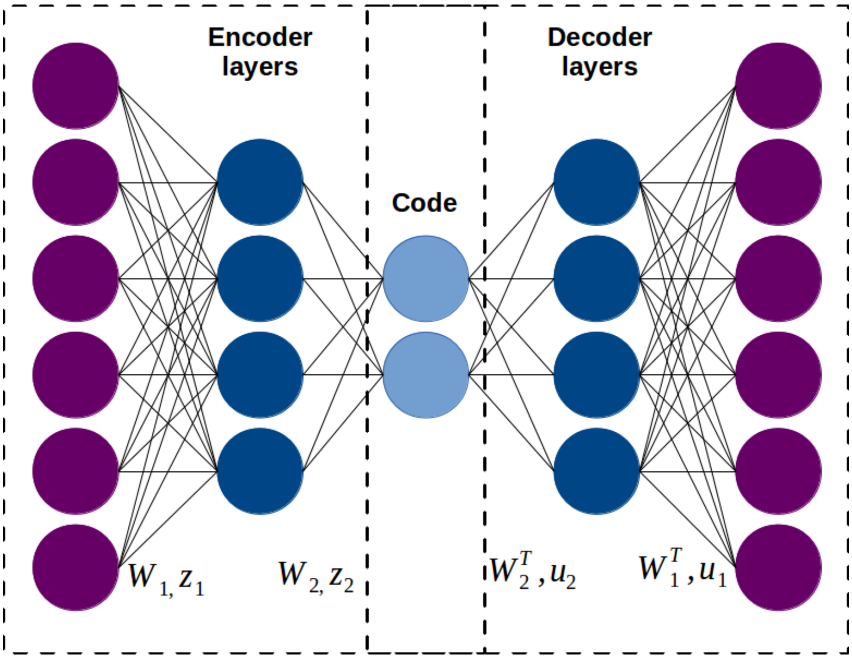


Figure 2.9: The topology of an autoencoder: $\mathbf{W}_i$ is the weight matrix associated to the connections between the layer i and $i+1$; $\mathbf{u}_i$ and $\mathbf{z}_i$ are the bias vectors of the layer i of the encoder and decoder layers, respectively.

input $\mathbf{x}$ and maps it onto $\mathbf{h}$, according to

$$\mathbf{h} = \sigma(\mathbf{W}\mathbf{x} + \mathbf{z})\,, \tag{2.11}$$

where $\mathbf{h}$ is usually referred to as code, $\mathbf{z}$ is the encoder bias and σ is an activation function.

In a trained autoencoder, $\mathbf{h}$ can be mapped onto the reconstruction $\mathbf{x}'$ of the same size as $\mathbf{x}$, according to

$$\mathbf{x}' = \sigma(\mathbf{W}^T\mathbf{h} + \mathbf{u})\,, \tag{2.12}$$

where $\mathbf{u}$ is the decoder bias.

Autoencoders are trained to minimize a loss function, $\mathcal{L}$, defined as

$$\mathcal{L}(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|^2 = \left\|\mathbf{x} - \sigma(\mathbf{W}^T\sigma(\mathbf{W}^T\mathbf{x} + \mathbf{z}) + \mathbf{u})\right\|^2 \tag{2.13}$$

Recently, some works have provided autoencoders to train generative models [Kingma and Welling, 2013], [Hinton et al., 2006], [Zhu et al., 2013]. Figure 2.10 shows a hypothetical DBN model to train an autoencoder.

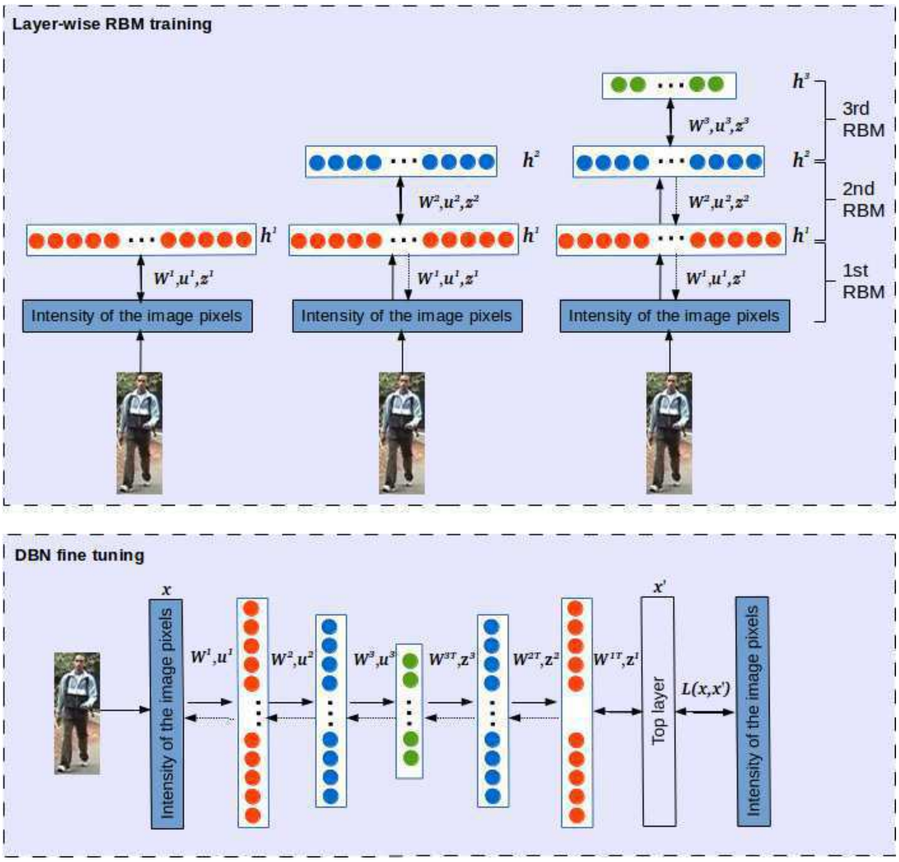


Figure 2.10: An example of a deep belief network autoencoder. After the first stage (upper box), the DBN encoder layers are pre-trained. In the second stage (lower box), the DBN is fine-tuned. The encoder top layer should denotes a compressed representation of a person image, after the second stage.

2.2.5 Denoising Autoencoder

A DAE is a kind of autoencoder that maps a corrupted version of the data input back to the original one. The main goal of the DAE is to create a similar representation of a data regardless its variations, commonly induced by noise.

Given a corrupted version $\tilde{\mathbf{x}}$ of $\mathbf{x}$, a DAE should compute the weight matrix $\mathbf{W}$ in a similar way to Eq. 2.10, according to

$$\mathbf{W} = \operatorname*{argmin}_{\mathbf{W}} \left\| \mathbf{x} - g(r(\tilde{\mathbf{x}}, \mathbf{W}), \mathbf{W}^T) \right\| \,, \tag{2.14}$$

The difference between Eq. 2.14 and Eq. 2.10 is that, in the first, the g function has $\tilde{\mathbf{x}}$ instead of $\mathbf{x}$, as input. In a conventional DAE, $\tilde{\mathbf{x}}$ is a corrupted version of $\mathbf{x}$. However, a DAE can be conceived to work with any other relationship between $\tilde{\mathbf{x}}$ and $\mathbf{x}$. For example, $\mathbf{x}$ can represent an object image in a frontal view, while $\tilde{\mathbf{x}}$ is an image of the same object in a different view. A set of training samples comprised of $\tilde{\mathbf{x}}$ examples can be used to train a DAE, having $\mathbf{x}$ as a label in order to create a normalized representation of this object. The use of a DAE to solve problems that are different from those solved by the conventional one can be found in [Vincent et al., 2010], [Luo et al., 2012], [Ngiam et al., 2011], [Zhu et al., 2013] and [Krizhevsky and Hinton, 2011].

2.3 Data sets

Performance of person re-id systems is assessed over public data sets. There are two ways of evaluating a re-id system, those are via: still images or videos. Still images are used when it is considered that the detection stage of a re-id system is quite perfect so that the proposed novel method is evaluated in isolation over cropped person images. Video images are commonly used to evaluate the whole re-id system by considering the detection or the identification stage in a wild environment. The idea of video performance assessment is to demonstrate the maturity of a whole system in a practical application.

Our proposed person re-id method was thoroughly evaluated over cropped person image data sets, with the goal of acquiring a first performance experimental evaluation. This indicates we are taking in advance that our person detector is simply perfect and isolating the method's performance. For that, four public and state-of-the-art data sets were used: VIPeR [Gray et al., 2007], i-LIDS [iLI, 2007], CUHK01 [Li et al., 2013] and CUHK03 [Li et al., 2014]. VIPeR data set is comprised of 632 pedestrian image pairs taken from two non-overlapping cameras. i-LIDS data set contains 476 images of 119 pedestrians taken from two non-overlapping cameras. CUHK01 contains 971 people, taken from two camera views in a campus environment, while the CUHK03 is composed of 1360 people captured by six different cameras.

2.4 Relation to our work

Our work approaches the person re-id problem by using local and global trainable descriptors learned in a novel deep learning framework, called CFL. The rationale of CFL framework is to mimic the generic-to-specific human learning process, exploiting a transfer learning technique, in the point of view of machine learning field (see Chap. 3). In the prediction stage of the proposed framework, instead of matching a pair of source and target images, the input image pairs are increased by an artificial noisy version of the original images. The final identification score is given by the distances generated among all the combination of original and included noisy images. The last contribution of our work is the conception of an adaptive covariance descriptor, called CCF, obtained by computing covariance matrices over the deep features (see Chap. 4). Next, we present and contextualize the relation between our contributions and the background discussed in this chapter.

The transfer learning technique is not novel and is used in some works: In [Masci et al., 2011], [Luo et al., 2012], [Ngiam et al., 2011], [Vincent et al., 2010] and [Zhu et al., 2013], a deep network is trained in two stages with a transfer learning among them. Transfer learning in these cited works is performed only to avoid the network overfitting. In our proposed CFL framework, a transfer learning strategy was conceived to simulate the human learning process from generic to specific concepts in order to learn person descriptors in multiple steps. This showed to be an efficient strategy, thus increasing the person re-id performance.

While in [Yi et al., 2014a] and [Yi et al., 2014b], a pre-estabilished score function was used to predict the re-id, we proposed a novel metric to select a score function among a set of candidate ones. This metric was also used to choose the best configuration of the input data in the prediction phase. The core of the network used in the CFL framework is comprised of a DBN, which borrows the parameters and topology from [Yi et al., 2014a], and also a CNN with the same structure to the one proposed in [Luo et al., 2012].

CCF were motivated by the original covariance descriptors proposed in [Porikli and Kocak, 2006], and exploited in [Eiselein et al., 2014], [Hirzer et al., 2011], [Zeng et al., 2015] and [Dultra et al., 2013] to solve the problem of person re-identification. In contrast to the original descriptors, CCF are adaptive, because it is extracted directly from the intermediate layers of a trained CNN. Since the CNN, in the way that was proposed, provide only the local features, a new way to integrate the CCF and the global features found in the CFL framework is also done.

Part II

Proposed framework

CHAPTER 3

Coarse-to-fine learning framework

Contents

3.1 Introduction

A framework to learn trainable and discriminative features for person re-id was proposed here. This framework was conceived to acquire knowledge from generic-to-specific concepts, as follows: (i) recognition of global characteristics of a visual object, regardless their variations; (ii) recognition of what a person is (generic concept about person, usually learned in childhood); (iii) gender discrimination (male/female - specific concept about a person), and, finally, (iv) comparison of each local and global features of the target person with a data base.

Figure 3.1 describes the steps followed by our framework and examples of the system input and correspondent system response in each step. From the first to the last step, the proposed framework learns from a more generic concept to more specific ones. The aim of the first step is to learn global features of a person image (entire image of a person), which are expected to be robust to some image variations. The rationale in this step is that **the image person is still an image person regardless its visual variations, even without any knowledge about what a person is**. In this stage, there is a direct relationship between input and response, that is, both of them are an image person, but the response should be a normalized image regardless the variations of the input. It is noteworthy this concept can be applied to any other object image after specific changes in the framework.

In CFL framework, global features are learned in the first step and tuned in the next ones. Local features (body parts) of an image person start to be learned from the second step, and tuned in the last two ones. During the second and third steps, features are optimized by the learning of two concepts, respectively: “Is the input image a person or not?” and “if the image is a person, which is the gender of that person?”. Global and local features are finally optimized in the last step. Different images of the same person should have similar feature representations, which should be as distinct as possible for images of different people.

A question arises from all this discussion: A machine learning framework, based on a generic-to-specific concept, can achieve better performance than that conventional one? Conventional machine learning here is comprehended by one that learns a knowledge, only considering a particular domain of a problem, without any other previous domain. Implementation of the proposed framework and several experiments will give some evidences that the answer for that question is ‘yes’.

3.2 CFL framework

CFL was implemented by a cascade network training, driven by a transfer learning technique, and performed in four steps (depicted in Fig. 3.2). They are:

- Step 1: Pre-training of the person global features is performed by a proposed DBN-DAE (see Section 3.2.1 for details). The goal of these global features are to be invariant to some image variations (for simplicity, these variations are considered here as noises).

- Step 2: Once pre-trained, the DBN-DAE encode layers are integrated to a set of CNNs in a proposed hybrid deep network (see Section 3.2.2 and Fig. 3.2a). CNN will learn local features. Local features will be optimized and the global ones will be tuned in a binary

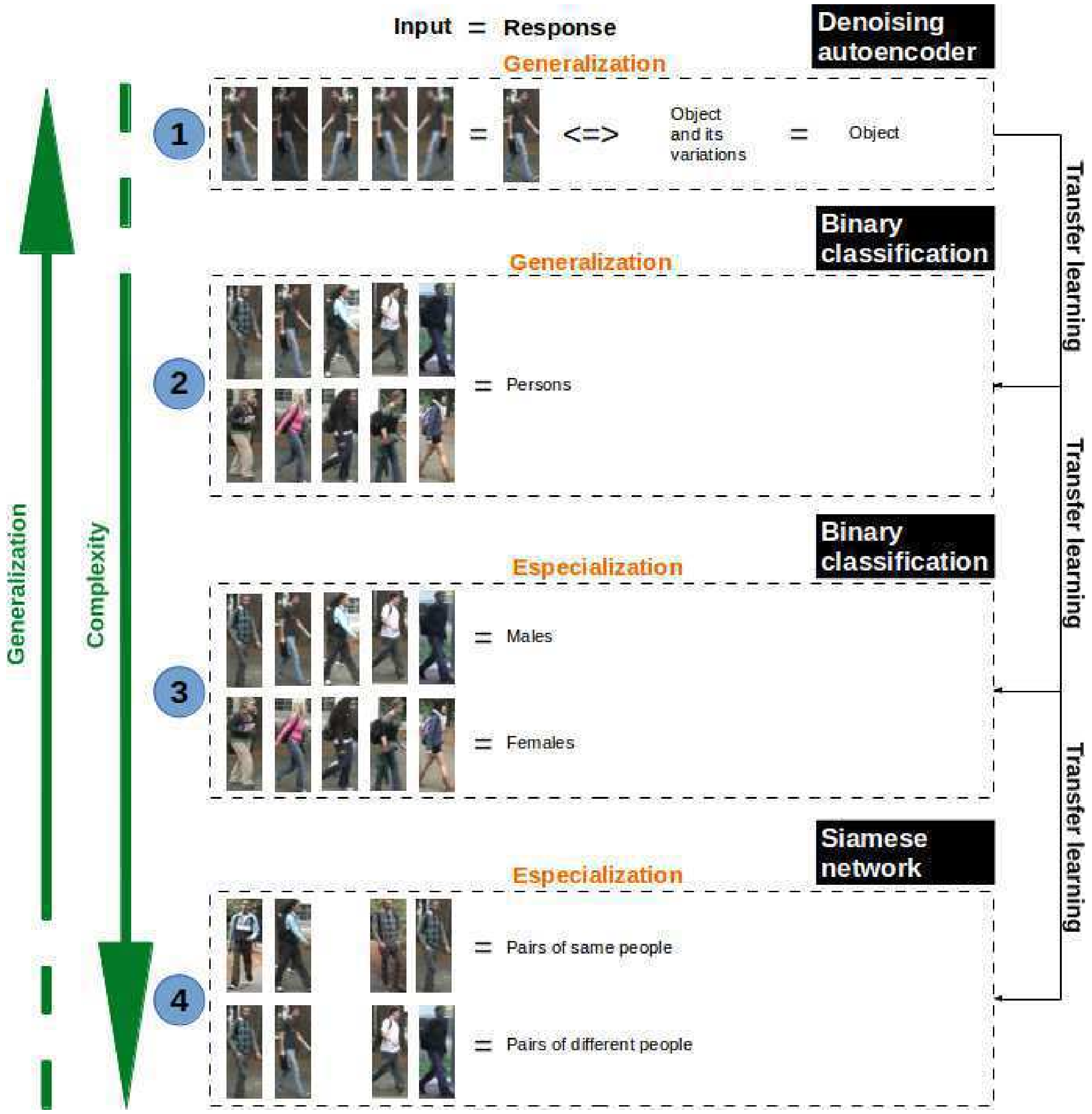


Figure 3.1: Steps of CFL. Steps ① and ② are associated with learning by generalization, while ③ and ④ with learning by specialization. The problem is less complex and more generic in the first two steps and, conversely, it is more complex and more specific in the last two steps.

classification fashion. Particularly, the network learns to classify which image is or is not a person.

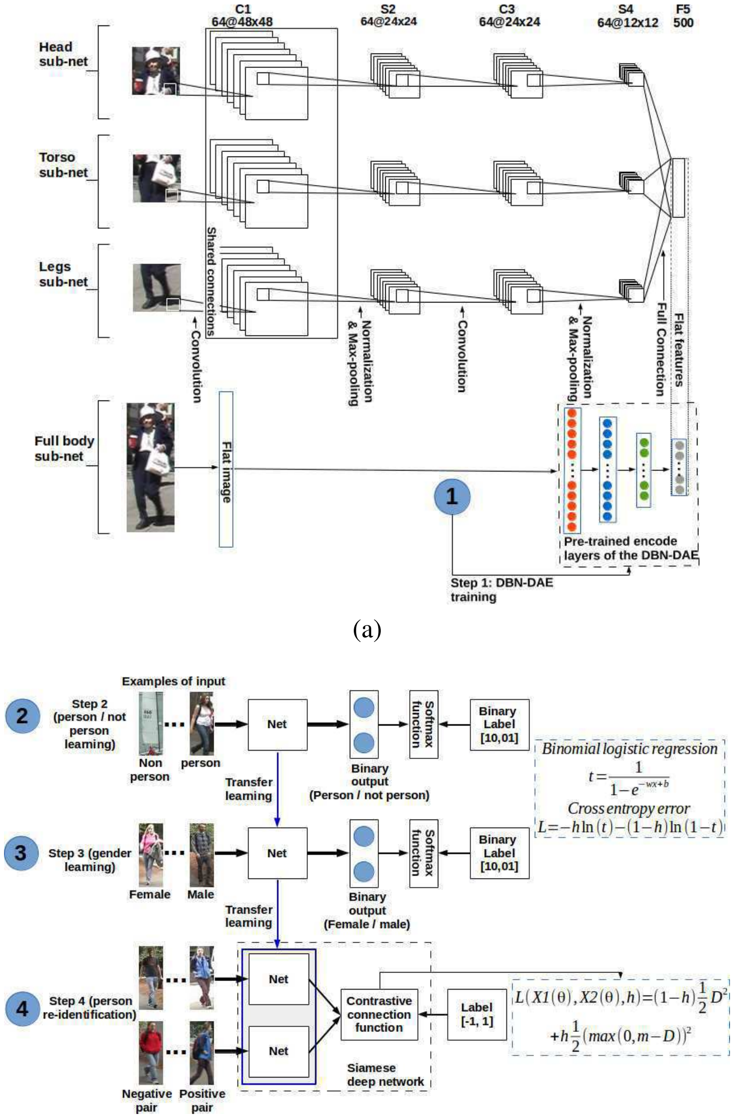
- Step 3: Global and local features will be tuned in the same way as in step 2, but in a different domain of classification problem. Here, the hybrid network is trained to identify two genders: male and female.

- Step 4: A copy of two identical hybrid network, pre-trained in the previous step, will be trained in order to optimize the person features for person re-identification. In this step, there is a need to train two networks together, since the training is accomplished over a pair of person images in order to measure the similarity between them. The two hybrid networks are trained into a network topology called Siamese network (see Section 3.2.3).

Figure 3.2 depicts the outline of CFL. The ”Net” box depicted in Fig. 3.2b concerns a hybrid deep network comprised of a CNN and a DBN-DAE, and it is illustrated in more detail in Fig. 3.2a. In the hybrid architecture (see Section 3.2.2), while the CNN extract local features from the person images, the pre-trained encode layers of a DBN-DAE network (see Section 3.2.1) select global features, which are expected to be invariant to certain types of noises, such as brightness changing, horizontal mirroring, blurring and small image distortions. While a conventional DAE aims at reconstructing an image from its corrupted version, our proposed DAE attempts to reconstruct a person image from a noisy image version. For each original image, a set of randomly brightness change, horizontal mirroring, blurring and distorted images was generated. Each one of those types of noise is incorporated in the network training phase by applying some noise generating functions in the images of the original data sets (see examples illustrated in Fig. 3.3). That augmented data set also works to reduce the overfitting caused by the large number of deep network parameters. Although all CNNs or all DBN-DAEs would be possible in the ”Net” box, experiments demonstrated that a DBN-DAE along with a CNN form the best configuration (see Section 3.3 for a framework experimental evaluation). After having the DBN-DAE pre-trained (step 1), the hybrid network is now able to be trained in the three further steps. The learning is transferred (see Section 3.2.4) from the person network (step 2) to the gender network (step 3), and then finally to the Siamese Network (step 4) in order to perform the final person re-id (see Fig. 3.2b). The Siamese (Section 3.2.3) learns which pairs of images belong to the same or different people.

3.2.1 Proposed DBN-DAE

The topology of our DBN-DAE is structured by four pre-trained Restricted Boltzmann Machines (RBM) layers. According to Fig. 3.4, $\mathbf{v}$ and $\mathbf{h}^i$, with $i = 1$ to 4, are the visible and hidden units, respectively. $\mathbf{v}$ is the flat version of a noisy image $\mathbf{I}_N$. $\widetilde{\mathbf{v}}$ is the output of the network, while $\mathbf{l}$ is the flat version of the original image $\mathbf{I}_o$. The weights among each pair of layers are represented by $\mathbf{W}^i$ vectors. $\mathbf{z}^i$ and $\mathbf{u}^i$ are the offset vectors for input and hidden units, respectively. There are two steps to reach a fully trained DBN-DAE to fulfil the first CFL training step: (i) A stacked layer-wise training for each one of the four RBM, as shown in Fig. 3.4a,



(a)

(b)

Figure 3.2: Outline of our proposed framework and CFL steps. (a) Proposed hybrid network comprised of three CNNs (each one for each human body part – head, torso and legs) and a pre-trained DBN-DAE. (b) The three steps of the overall hybrid network training. The ”Net” box contains the hybrid network, illustrated in (a). The blue circles indicate the four CFL training steps: ① DBN-DAE training, described in more details in Fig. 3.4; ② Person / not person learning; ③ Person gender learning; and ④ Person re-id. The steps follow the order from more generic to more specific learning about person.

and described in Section 2.2.3 (each RBM is trained according to the procedures discussed in Section 2.2.2), and (ii) a DBN fine-tuning to minimize the cross-entropy error between $\widetilde{\mathbf{v}}$ and $\mathbf{l}$, as shown in Fig. 3.4b.

The DBN-DAE is comprised of an encoder and decoder layers (see Figure 3.4b). The en-

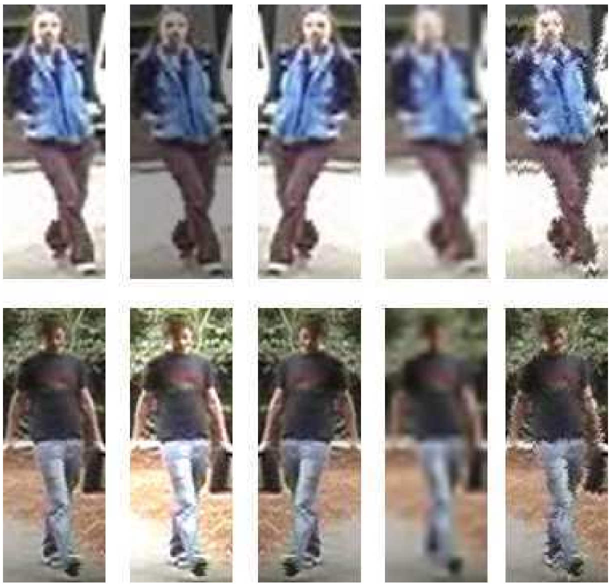


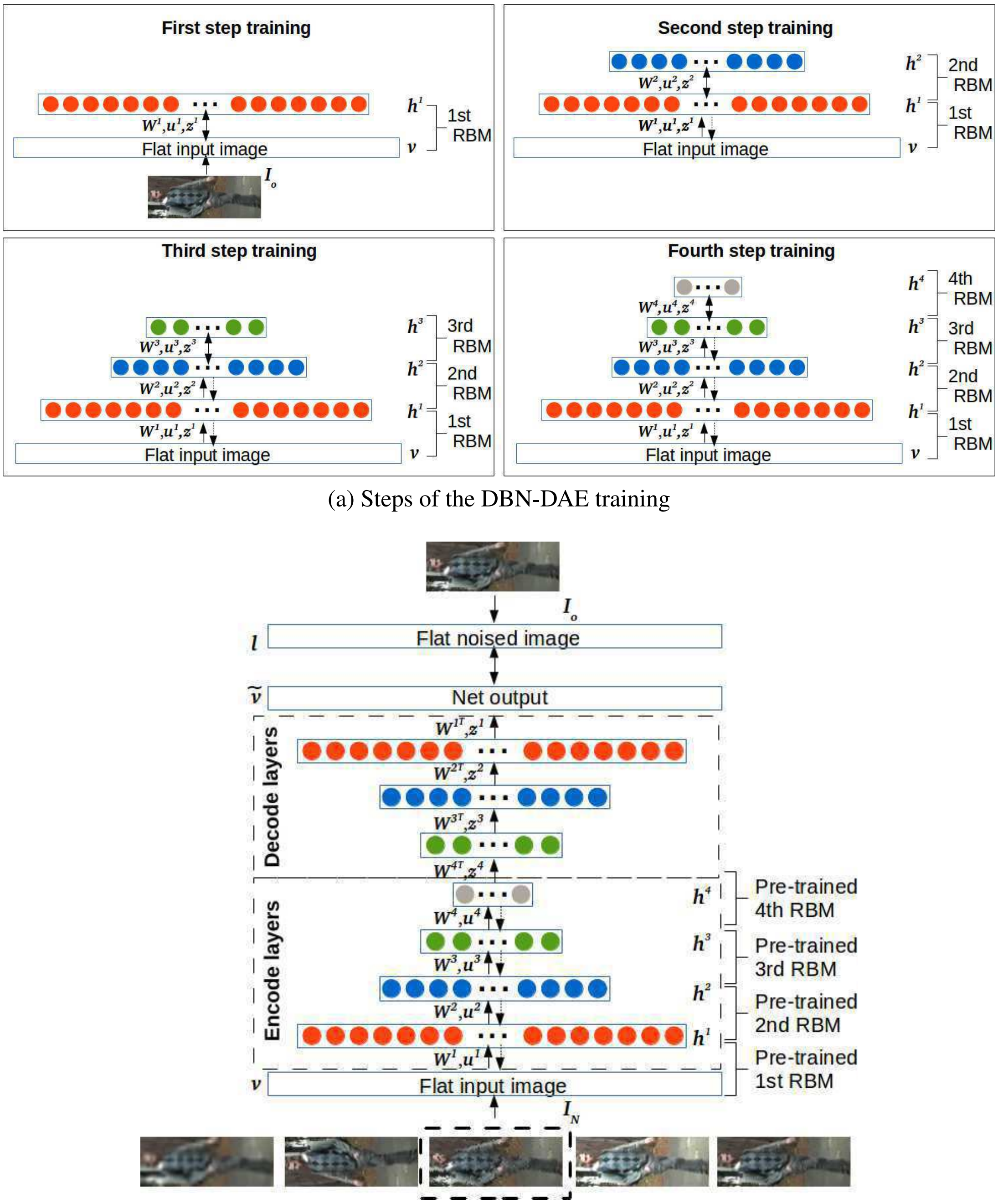
Figure 3.3: Noise generating functions applied on samples of VIPeR data set [Gray et al., 2007]. From left to right: Original image, randomly brightness change, horizontal mirroring, and blurring and image distortions.

coder is comprised of four stacked RBM, trained after the last cascade RBM training step (network from the fourth step of the cascade RBM training in Fig. 3.4a). The decoder layers are a mirrored version of the encoder ones (see Section 2.2.4). The weights of the decoder layers are the transpose of the weights of the encoder ones. The number of DBN-DAE units for the input and output layers are 6912 (flat version of the 48×48 re-sized image with the 3 RGB channels), following the work in [Luo et al., 2012]. As in [Luo et al., 2012], the number of hidden units are 4000 in $\mathbf{h}^1$, 2000 in $\mathbf{h}^2$, 1000 in $\mathbf{h}^3$ and 500 in $\mathbf{h}^4$.

The encoder layers of the pre-trained DBN-DAE is coupled to CNN to form the hybrid network (see Fig. 3.2a). The top layer of the encoder corresponds to the global features of the person image that will be tuning after the step 2 of the CFL training.

3.2.2 Hybrid network

The hybrid network has four sub-nets: One for each fixed image region (head, torso and legs) and a full-body sub-net (see Fig. 3.2a). Each fixed image regions were extracted from the entire image in a square of 48×48 pixels wide. Before dividing each person body-part image, each fixed image region was overlapped one another (see Fig. 3.5), to guarantee that the body-parts image must be square in the input of the CNNs. Each one of the three body-part sub-nets is a CNN with two convolutional layers (C1 and C3), two max-pooling layers (S2 and S4), and



(a) Steps of the DBN-DAE training

(b) DBN-DAE fine-tuning with the flat noisy image layer

Figure 3.4: DBN-DAE topology and training steps. In (a), a cascade of layer-wise RBM training is performed in four steps. In the first step, the input layer of the first RBM is the flat version of a person image; when the first RBM is trained, a second RBM is stacked on the top of the first one; the output of the first RBM becomes the input layer of the second one. In the second step, while the new RBM is trained, the weights of the first one is fine-tuning. In the same way, the third and fourth steps follow the first and second ones. In (b), a fully symmetric DBN-DAE with all pre-trained RBM is trained to minimize a cross-entropy error between the output of the network and the flat original image, having a noise image as input.

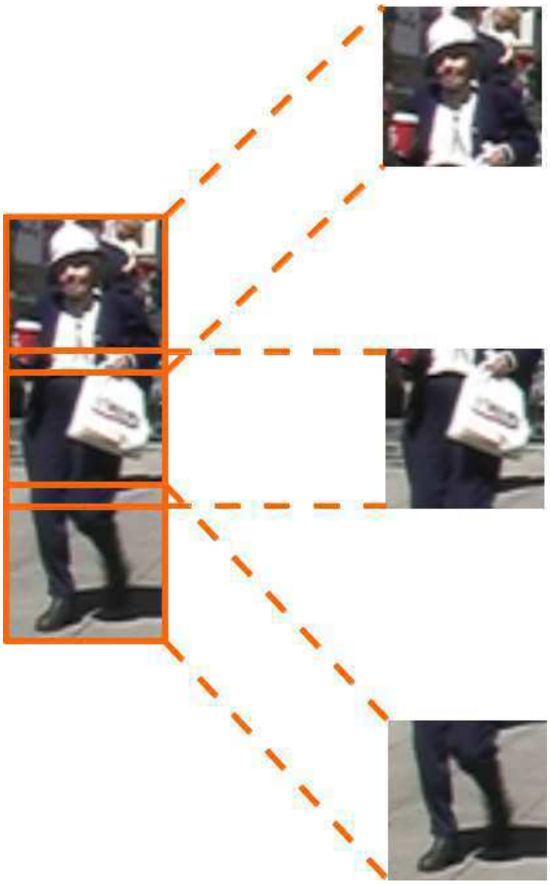


Figure 3.5: Person image body parts. The size of each part is 48×48 pixels wide. Each one of the parts overlaps the other in approximately 10%.

one full-connected layer (F5). This latter one has 500 units shared with the three body-part sub-nets. CNN topology and network parameters for these three top networks follow the work in Yi et al. [2014a]. The full-body sub-net is comprised of the pre-trained DBN-DAE, which provides a 500-dimensional feature vector in its output. At the end, the hybrid network output is given by the F5 layer concatenated with the output of the pre-trained DBN-DAE, forming a 1000-dimensional flat features.

3.2.3 Siamese Network

Proposed by [Yi et al., 2014a], the goal of a Siamese network is to learn a similarity function between a pair of input data. Within a Siamese topology, the outputs of the two networks are usually connected by a connection and a cost functions. The connection function evaluates the relationship between the two network outputs, while the cost function converts this relationship into a cost. A sample in the supervised training phase of the Siamese is composed of a pair of images and a label, y. In our Siamese Network, the two networks are connected by a contrastive connection function, which ultimately measures the similarity between the two network outputs and the cost, at the same time. The contrastive function, L, is defined as

$$\begin{aligned} L(X1(\phi), X2(\phi), y) = &(1-y)\frac{1}{2}D^2 + \\ &y\frac{1}{2}(max(0, m-D))^2 \,, \end{aligned} \tag{3.1}$$

where $D = \|X1(\phi) - X2(\phi)\|_2$, and $X1(\phi)$ and $X2(\phi)$ denote the output of the Nets (step 4 of Fig. 3.2b), m represents a constant (in our case, equal to 1), and ϕ represents the network parameters. The Siamese is trained to find the values of ϕ that minimize L. The contrastive function is not used in the prediction phase and the features of the two networks are evaluated by a distance function. The smaller the distance, the higher the similarity between the two people in the input of the Siamese. In [Yi et al., 2014a], two original images and their mirrored version from each pair of people, to be compared, generate four score distances and the final score was computed by the mean among the distances. Further we investigate which image combinations and which final score function improve the performance prediction (see Section 3.2.5 for more detail).

3.2.4 Learning strategy

Before the Siamese Network training step, the CFL takes place by means of a cascade of transfer learning (person → gender → identification). The goal of the transfer learning is to initialize the parameters of a network by using those pre-trained parameters of the previous ones. Particularly in steps 2 and 3 of Fig. 3.2b, as the problem domain resides in a binary classification, a binary layer in the output of the networks was included. In a current step, the learning rate in the training process is decreased by 10 times regarding to the previous step. This is so since the network of a higher step is tuning the parameters already learned in the previous one.

The network training in the three steps was performed using a stochastic gradient descent with mini-batch size equals to 100, and 30,000 iterations. The learning rate of the DBN-DAE was set to 0.01 and 0.00001, during the DBN-DAE training and CFL, respectively. The learning rate of the CNN was set to 0.01 in the step 1, 0.001 in the step 2 and 0.0001 in the step 3. The best network training parameters were chosen after a performance evaluation by using a variation of the holdout cross-validation method found in [Kohavi, 1995]. While the original method in [Kohavi, 1995] selects one pair of training/testing samples for performance evaluation, we have chosen four pairs. For each training sample, the CFL was trained with a set of CNN and DBN-DAE learning rate configurations, and the configuration that achieved the best

Table 3.1: Outcomes of the test to find the best function f (see Eq. 3.2) to compute the similarity score function, over VIPeR and iLIDS data sets. The test was accomplished over ten subsets by computing the value of y from each one of them, as well as, each one of the candidate functions.

Testing subset	VIPeR data set		
	max	min	mean
1	0.5062	0.5550	0.5278
2	0.5105	0.5445	0.5193
3	0.5088	0.5567	0.5264
4	0.5172	0.5598	0.5308
5	0.5205	0.5605	0.5255
6	0.5048	0.5478	0.5332
7	0.5155	0.5554	0.5342
8	0.5168	0.5594	0.5448
9	0.5064	0.5499	0.5292
10	0.4998	0.5632	0.5341
Average	0.5106	0.5552	0.5305
Testing subset	i-LIDS data set		
	max	min	mean
1	0.5176	0.5377	0.5274
2	0.5092	0.5340	0.5222
3	0.5222	0.5402	0.5341
4	0.5150	0.5425	0.5273
5	0.5108	0.5331	0.5236
6	0.5043	0.5442	0.5302
7	0.5200	0.5398	0.5324
8	0.5115	0.5382	0.5227
9	0.5145	0.5401	0.5331
10	0.5202	0.5422	0.5312
Average	0.5043	0.5392	0.5284

performance prediction was chosen as the network training parameters.

The same noisy functions (used to generate the artificial noisy images in the DBN-DAE training) were also used in the other steps of the CFL training, due to the large number of network parameters. This was done to increase the number of samples, in order to reduce the probability of the network overfitting. From each one of the original image were generated 15 randomly noisy ones, of which: Eight with different intensity of brightness; four with different image distortions; one mirrored; one blurred; and one mirrored and blurred, counting a total of 16 images per person. Hence, the size of training samples of the network from step 2 to step 3 (in Fig. 3.2b) was increased by 16 times the samples of the original data sets. Since a pair of images is the input of a Siamese network, the total amount of pairs could be equals to the

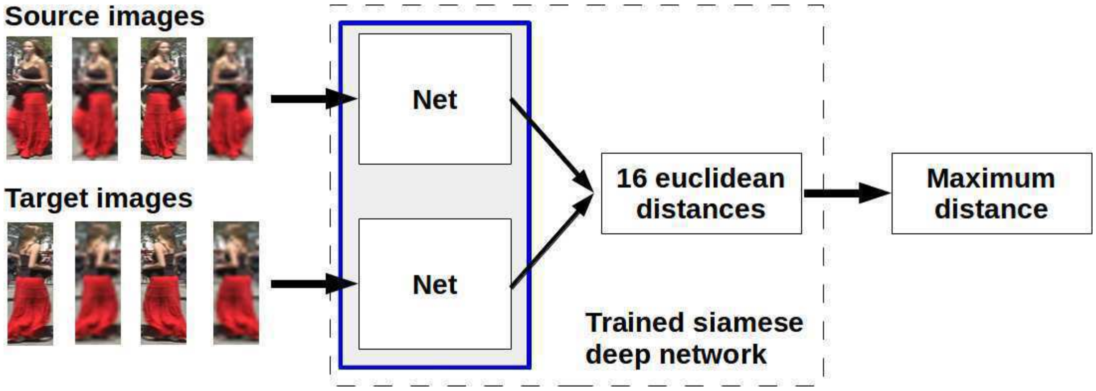


Figure 3.6: The structure of the Siamese network in the identification phase. The similarity between two people is computed by finding the maximum value among 16 euclidean distances, generated by the distances among the flat features of each source image and target ones (original and noisy versions).

number of combinations arising from the matching of each image against all the others. Then the size of training samples for this network could be equal to the size of the training samples of the original data set raised to square. By considering the noisy images, the total of samples for the Siamese network training can be further increased by 256 times. We can conclude that even a data set with a small number of examples, the number of produced training samples can be high, due to the addition of the artificial noisy images, as well as, the structure of the Siamese network. It was used then 20% of the total number of image pairs, which were randomly chosen in order to reduce the time of the Siamese network training without the loss of network generalization capability, and to fit in the computer memory. The drop-out method in [Srivastava et al., 2014] was also used in the learning phase to prevent network overfitting. In each training stage, this method consists in either to ”drop out” the incoming and outgoing connections of a network node with $1 - p$ probability, or to keep a network node with p probability, so that a reduced network is achieved. Only the reduced network is trained on the data in that stage. The removed connections are then reinserted into the network with their original weights.

3.2.5 Person re-identification: Distance space prediction

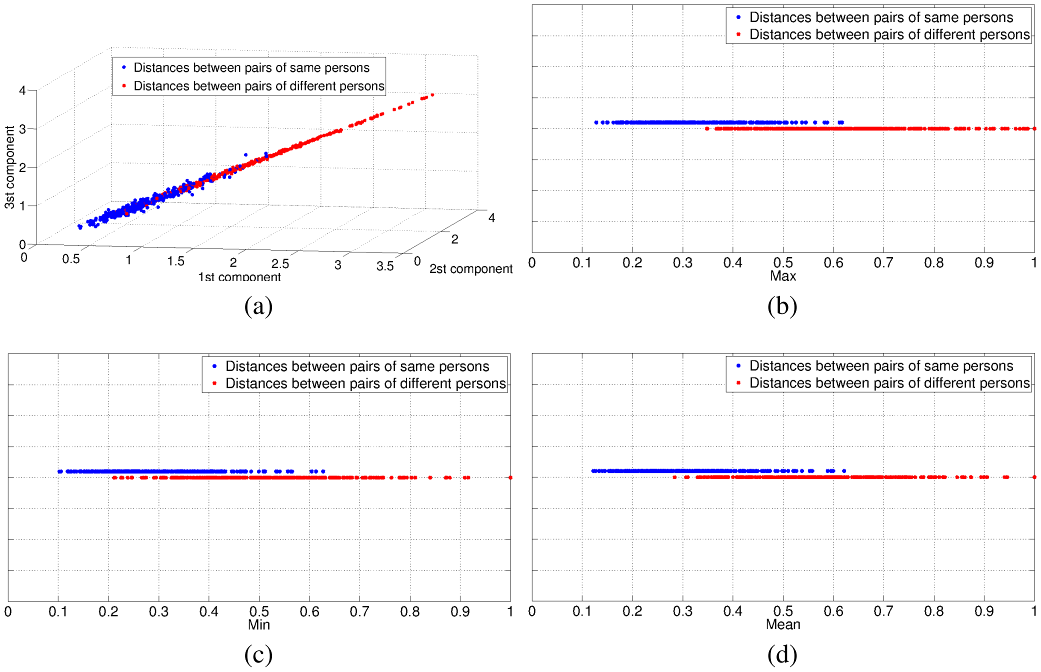
The aim of the person re-identification prediction is to find the more similar target person image, given a source one. This is achieved by computing a similarity score function among the flat features (the top layers of the hybrid network) of the source and target images. The person in the target image that obtains the highest score (it could be the lowest, depending on the score function) is considered the one who best matches the person in the source image. In our work, we have used the Euclidean distance as the base function to compute the similarity

Table 3.2: The same test as in Table 3.1, but over CUHK01 and CUHK03 data sets.

Testing subset	CUHK01 data set		
	max	min	mean
1	0.4765	0.5340	0.5111
2	0.4788	0.5321	0.5061
3	0.4892	0.5462	0.5233
4	0.4755	0.5298	0.5211
5	0.4663	0.5341	0.5198
6	0.4728	0.5401	0.5156
7	0.4872	0.5332	0.5201
8	0.4801	0.5341	0.5143
9	0.4912	0.5299	0.5200
10	0.4771	0.5279	0.5203
Mean	0.4803	0.5341	0.5172
Testing subset	CUHK03 data set		
	max	min	mean
1	0.4811	0.5231	0.5344
2	0.4946	0.5132	0.5256
3	0.4822	0.5233	0.5323
4	0.4805	0.5213	0.5399
5	0.4901	0.5245	0.5401
6	0.4888	0.5302	0.5345
7	0.4803	0.5322	0.5297
8	0.4922	0.5345	0.5356
9	0.4935	0.5298	0.5345
10	0.4899	0.5235	0.5399
Mean	0.4873	0.5256	0.5346

score between the flat features of the source and target images, from each pair of people to be compared. As in [Yi et al., 2014a], the prediction phase is performed by the Siamese network.

Although it is common to use only the original images to compute the prediction, Yi et al. [2014a] found a significant improvement in prediction performance when the data augmentation trick [Cireşan et al., 2011] is used over the testing set.Two original images and their mirrored versions were used, for each pair of people to be compared, producing four score distances. The average value among these distances was used as the final similarity score function. Although Yi et al. [2014a] have achieved better performance using this strategy, they did not perform any analysis to verify the possibility of other images and final similarity score function configurations, which might further improve the prediction performance. Here, we propose to use a so called distance space analysis, in order to check the prediction performance of a set of configurations. Distance space prediction represents the distance space related with the best



(a)

(b)

(c)

(d)

Figure 3.7: Distance graphical analysis over VIPeR data set. The blue and red points correspond to the distances between pairs of the same and different people, respectively. (a) 16 distances of each pair of people were projected into the first three principal components. (b), (c) and (d) graphical results after applying $\max$, $\min$ and mean functions over 16 distances. Distance values are normalized.

configurations, selected by the proposed analysis. The person re-identification prediction is accomplished in that space.

The proposed analysis gave us a hint of which similarity function, and target and source image configurations should be used to achieve the best prediction performance. Furthermore, it was also useful to validate the performance of the hybrid network topology and the CFL approach, as it will be shown in Section 3.3. The distance space analysis is based on a straightforward metric, which evaluates the ratio between the pair distances of the same people and the pair distances of different ones. By calculating y, given by

$$y = \frac{\sum_{n=1}^{t_1} f(\mathbf{s}_n)}{\sum_{n=1}^{t_2} f(\mathbf{d}_n)} , \tag{3.2}$$

it is possible to evaluate the best prediction parameters in the distance space.

The basic idea of this metric is to provide an indicative of high prediction performance when the distances among the same pair of people is low, and the distances among different ones is

Table 3.3: Prediction performance analysis of 25 source and target image configurations, over VIPeR data set. The minimum average value of y indicates that the prediction of the pair of target and source people, comprised of original, mirroring, blurring, and blurring and mirroring images (see Fig. 3.6), achieves the best performance.

Source and target image configurations	Average of y
Original	0.6544
Original + Brightness	0.6205
Original + Mirroring	0.6168
Original + Blurring	0.6250
Original + Blurring and mirroring	0.5944
Original + Distortion	0.5905
Original + Brightness + Mirroring	0.5748
Original + Brightness + Blurring	0.5855
Original + Brightness + Blurring and Mirroring	0.5698
Original + Brightness + Distortion	0.5724
Original + Mirroring + Blurring	0.5802
Original + Mirroring + Blurring and Mirroring	0.5665
Original + Mirroring + Distortion	0.5700
Original + Blurring + Blurring and Mirroring	0.5772
Original + Blurring + Distortion	0.5696
Original + Blurring and Mirroring + Distortion	0.5805
Original + Brightness + Mirroring + Blurring	0.5423
Original + Brightness + Mirroring + Blurring and Mirroring	0.5356
Original + Brightness + Mirroring + Distortion	0.5307
Original + Brightness + Blurring + Blurring and Mirroring	0.5286
Original + Brightness + Blurring + Distortion	0.5360
Original + Brightness + Blurring and Mirroring + Distortion	0.5200
Original + Mirroring + Blurring + Blurring and Mirroring	**0.5106**
Original + Mirroring + Blurring and Mirroring + Distortion	0.5189
Original + Blurring + Blurring and Mirroring + Distortion	0.5201

high. In Eq. 3.2, $\mathbf{s}_n$ and $\mathbf{d}_n$ are k-dimensional vectors comprised of the k Euclidean distances of the n-th person pair and $\mathbf{s}_{\{1..t_1\}}$ and $\mathbf{d}_{\{1..t_2\}}$ contain the distances of t_1 and t_2 pairs of the same and different people, respectively. It is easy to note the lower the value of y, the higher the prediction performance. Hence it is necessary to select a f that minimizes y.

Three candidate f functions were chosen to compute the similarity score function: $\max$, $\min$ and mean. The test to choose the suitable f was conducted over VIPeR, iLIDS, CUHK01 and CUHK03 data sets. All individuals in these data sets were randomly divided into two subsets: Training and testing sets, containing half of the available individuals in each subset, with no overlapping regarding person identities. Ten pairs of the aforementioned subsets were selected, and the CFL learning and prediction were performed.

For each combination of testing subset, data set and candidate function ($\max$, $\min$ and $mean$), the distance space metric was computed. Since 10 testing subset were selected, 10 values of y were obtained, for each candidate function and data set. Each distance space metric was computed over 100 selected pairs of the same and different people, during the prediction phase. The average obtained among the 10 values of y was used as a reference to select the appropriated f (see Tables 3.1 and 3.2). The $\max$ function was one that achieved the minimum reference value among the candidate functions, as we can see in Tables 3.1 and 3.2, and it was the selected f to compute the similarity score function.

The same distance space analysis was performed to chose the best configuration for the source and target images. The minimum average of y was achieved for the configuration showed in Fig. 3.6 (the original, mirroring, blurring, and mirroring and blurring source and target image

Table 3.4: The same analysis showed in Table 3.3, over iLIDS data set.

Source and target image configurations	Average of y
Original	0.6584
Original + Brightness	0.6305
Original + Mirroring	0.6218
Original + Blurring	0.6210
Original + Blurring and mirroring	0.5934
Original + Distortion	0.6012
Original + Brightness + Mirroring	0.5841
Original + Brightness + Blurring	0.5857
Original + Brightness + Blurring and Mirroring	0.5708
Original + Brightness + Distortion	0.5726
Original + Mirroring + Blurring	0.5822
Original + Mirroring + Blurring and Mirroring	0.5681
Original + Mirroring + Distortion	0.5708
Original + Blurring + Blurring and Mirroring	0.5805
Original + Blurring + Distortion	0.5732
Original + Blurring and Mirroring + Distortion	0.5816
Original + Brightness + Mirroring + Blurring	0.5495
Original + Brightness + Mirroring + Blurring and Mirroring	0.5406
Original + Brightness + Mirroring + Distortion	0.5322
Original + Brightness + Blurring + Blurring and Mirroring	0.5386
Original + Brightness + Blurring + Distortion	0.5401
Original + Brightness + Blurring and Mirroring + Distortion	0.5225
Original + Mirroring + Blurring + Blurring and Mirroring	**0.5043**
Original + Mirroring + Blurring and Mirroring + Distortion	0.5199
Original + Blurring + Blurring and Mirroring + Distortion	0.5208

Table 3.5: The same analysis showed in Table 3.3, over CUHK01 data set.

Source and target image configurations	Average of y
Original	0.6364
Original + Brightness	0.6184
Original + Mirroring	0.6047
Original + Blurring	0.6111
Original + Blurring and mirroring	0.5895
Original + Distortion	0.5875
Original + Brightness + Mirroring	0.5684
Original + Brightness + Blurring	0.5901
Original + Brightness + Blurring and Mirroring	0.5756
Original + Brightness + Distortion	0.5648
Original + Mirroring + Blurring	0.5758
Original + Mirroring + Blurring and Mirroring	0.5594
Original + Mirroring + Distortion	0.5584
Original + Blurring + Blurring and Mirroring	0.5643
Original + Blurring + Distortion	0.5495
Original + Blurring and Mirroring + Distortion	0.5694
Original + Brightness + Mirroring + Blurring	0.5304
Original + Brightness + Mirroring + Blurring and Mirroring	0.5184
Original + Brightness + Mirroring + Distortion	0.5034
Original + Brightness + Blurring + Blurring and Mirroring	0.5064
Original + Brightness + Blurring + Distortion	0.5153
Original + Brightness + Blurring and Mirroring + Distortion	0.4986
Original + Mirroring + Blurring + Blurring and Mirroring	**0.4876**
Original + Mirroring + Blurring and Mirroring + Distortion	0.4992
Original + Blurring + Blurring and Mirroring + Distortion	0.4978

configuration), as shown in Tables 3.3, 3.4, 3.5 and 3.6. The selected target and source image configuration generated 16 distances from the matches of each one of the source image with all the target ones (see Fig. 3.6). Although the tests to choose the appropriate f, and the target and the source configuration are separated here, they were jointly performed.

Figures 3.7, 3.8, 3.9 and 3.10 show graphic analyses of the distances of the person pairs, arising from the first selected testing subset over VIPeR, iLIDS, CUHK01 and CUHK03 data sets, respectively. In Figures 3.7a, 3.8a, 3.9a and 3.10a, the 16 distances among the pairs were projected into the three principal components by computing the principal component analysis (PCA) in order to visualize the distributions of the distances. Figures 3.7b, 3.7c and 3.7d over VIPeR, Figures 3.8b, 3.8c and 3.8d over iLIDs, Figures 3.9b, 3.9c and 3.9d over CUHK01 and Figures 3.10b, 3.10c and 3.10d over CUHK03, show 1-dimensional projection of the distances by performing the $\max$, $\min$ and mean candidate functions over the original 16-dimensional

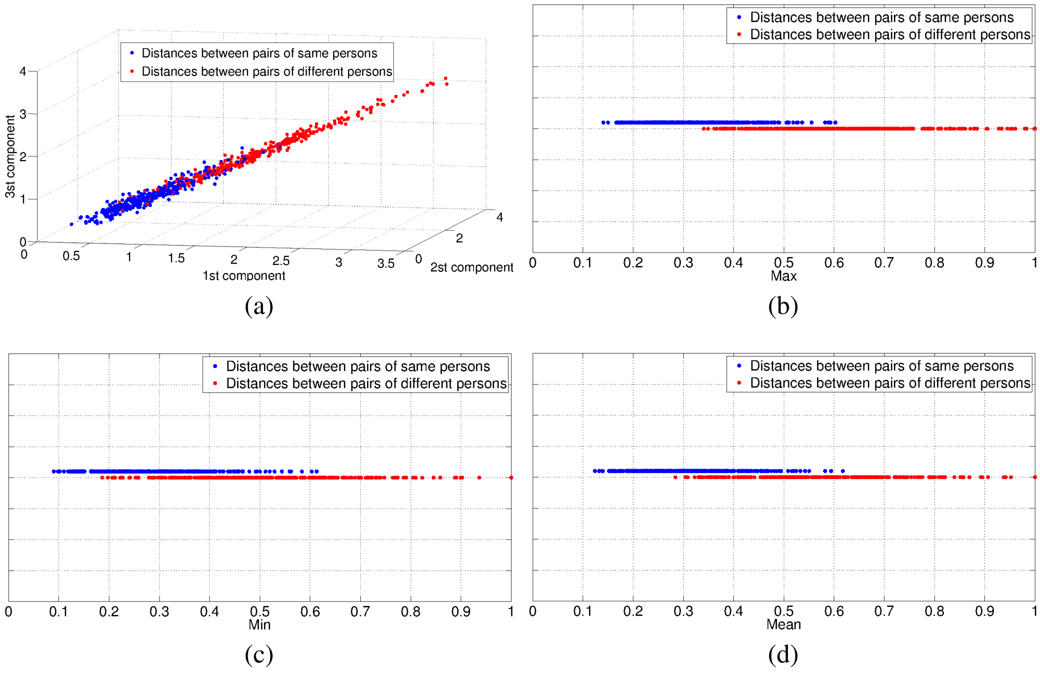
Table 3.6: The same analysis showed in Table 3.3, over CUHK03 data set.

Source and target image configurations	Average of y
Original	0.6312
Original + Brightness	0.6132
Original + Mirroring	0.6134
Original + Blurring	0.6075
Original + Blurring and mirroring	0.5822
Original + Distortion	0.5798
Original + Brightness + Mirroring	0.5567
Original + Brightness + Blurring	0.5835
Original + Brightness + Blurring and Mirroring	0.5721
Original + Brightness + Distortion	0.5611
Original + Mirroring + Blurring	0.5634
Original + Mirroring + Blurring and Mirroring	0.5511
Original + Mirroring + Distortion	0.5545
Original + Blurring + Blurring and Mirroring	0.5587
Original + Blurring + Distortion	0.5385
Original + Blurring and Mirroring + Distortion	0.5526
Original + Brightness + Mirroring + Blurring	0.5310
Original + Brightness + Mirroring + Blurring and Mirroring	0.5056
Original + Brightness + Mirroring + Distortion	0.5011
Original + Brightness + Blurring + Blurring and Mirroring	0.5004
Original + Brightness + Blurring + Distortion	0.5101
Original + Brightness + Blurring and Mirroring + Distortion	0.4932
Original + Mirroring + Blurring + Blurring and Mirroring	**0.4811**
Original + Mirroring + Blurring and Mirroring + Distortion	0.4945
Original + Blurring + Blurring and Mirroring + Distortion	0.5001

distances, respectively. In these figures, blue and red points represent the distances between pairs of the same and different people, respectively. In Figures 3.7b, 3.8b, 3.9b and 3.10b, it is worthy noting two characteristics of the $\max$ function that can reduce the probability of false positives and false negative: (i) Overlapping region between red and blue points is smaller than the intersection region from the other 1-dimensional functions; and (ii) red points distribution in the regions of high distance values is more evident then those found in the other functions.

3.3 Experimental Analysis

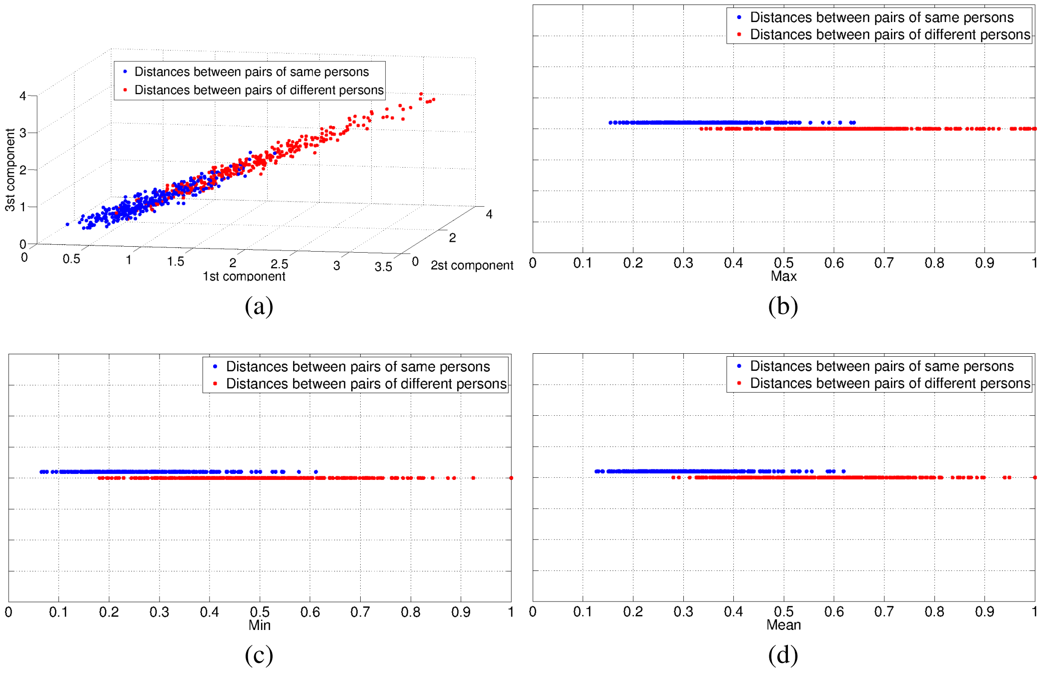
The performance of the framework was evaluated by considering the hybrid network topology, CFL configuration training and the comparative evaluation with 16 other state of the arts methods. All the tests were performed over VIPeR, i-LIDS, CUHK01 and CUHK03 data sets.



(a) (b)

(c) (d)

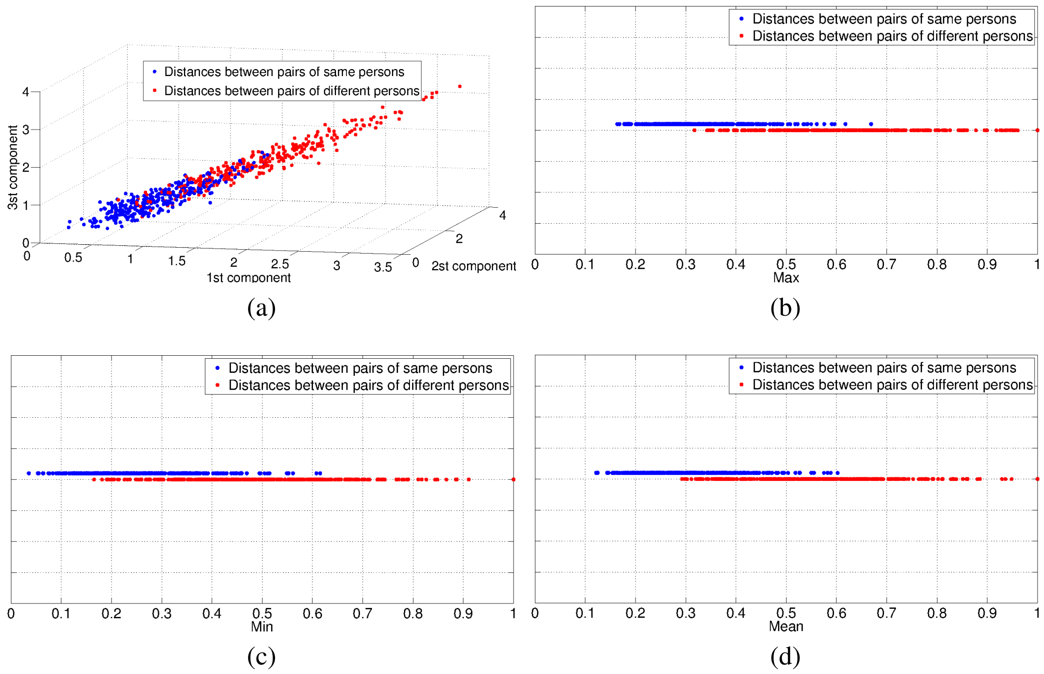
Figure 3.8: The same analysis as in Figure 3.7, applied in the iLIDS testing set.



(a) (b)

(c) (d)

Figure 3.9: The same analysis made in Figure 3.7, applied in the CUHK01 testing set.



(a)

(b)

(c)

(d)

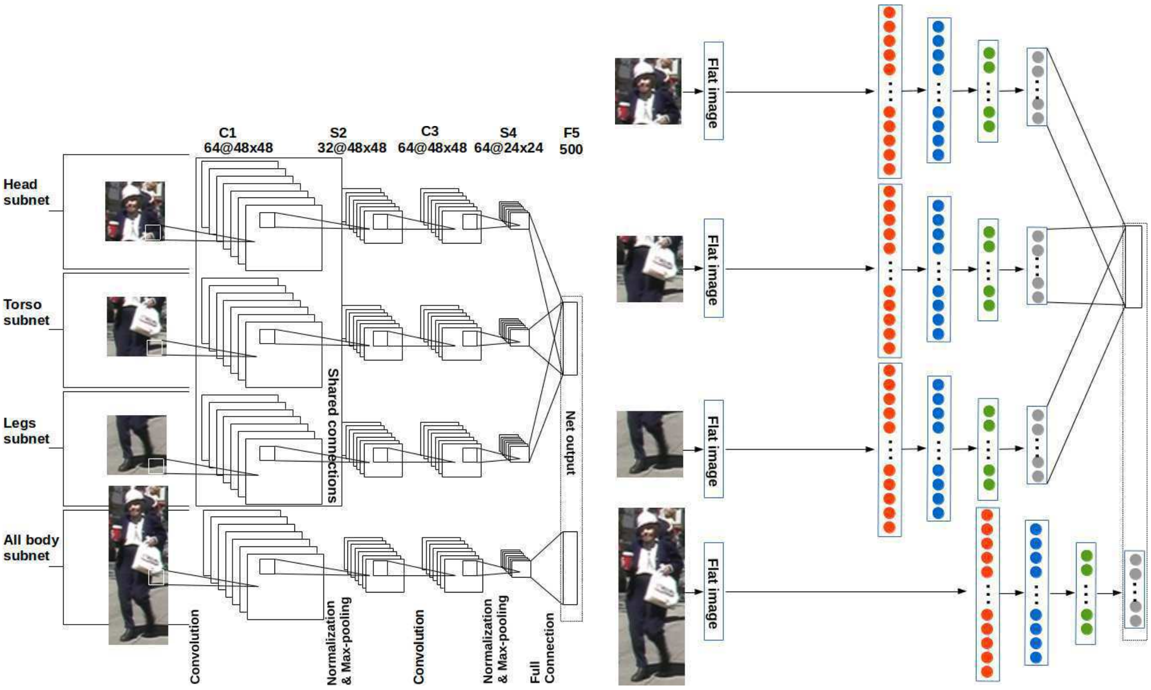
Figure 3.10: The same analysis as in Figure 3.7, applied in the CUHK03 data set testing set.

3.3.1 Methodology

Here we have followed the same testing protocol of the compared works. Except on the CUHK03 data set, and for the comparative evaluation with the method in [Chen et al., 2015] over i-LIDs data set, training and testing sets were selected in same way as in Section 3.2.5 with ten new randomly selected subsets. Chen et al. [Chen et al., 2015] divided the training and testing set into three groups over i-LIDs data set: (i) 89/39 training/testing people; (ii) 69/50 training/testing people; and (iii) 39/80 training/testing people. In CUHK03, the individuals in the training/testing sets were split to 1260/100, as it was made in [Li et al., 2014]. As in Section 3.2.5, the tests were repeated 10 times, and the final results were the average values among all those tests.

3.3.2 Selection of the hybrid network topology and training approach

The first step in the performance assessment of the proposed method is to define the best hybrid architecture, which is lately used inside the CFL framework. Three types of hybrid network were experimentally evaluated, considering a general structure depicted in Fig. 3.2a, but varying the type of network inside: (i) all CNNs, (ii) all DBN-DAEs and (iii) CNN and DBN-DAE (see Fig. 3.11). The second step is to assess the performance of the overall network depicted in



(a) (b)

Figure 3.11: Two topology configurations of the hybrid network which differ from the proposed topology: (a) All the hybrid network comprised of CNN; (b) All the hybrid network comprised of pre-trained DBN-DAE

Fig. 3.2b by varying its architecture, according to: (i) person and gender transfer learning (all CFL); (ii) fine-to-coarse learning (FCL) – training gender before person; (iii) only person transfer learning (Person CFL); (iv) only gender transfer learning (Gender CFL); and, (v) with only the Siamese deep network (without CFL). For simplicity, we are considering, here forth the ”without CFL” term as being the CFL training without perform the step 1, 2 and 3 of the proposed CFL in Figure 3.2b. Figure 3.12 shows that the use of the hybrid network with **CNN and DBN-DAE with full CFL training** increases the hit rate, in the top rank of our model, by 11, 13, 21 and 21 percentage points, respectively, over VIPeR, i-LIDS, CUHK01 and CUHK03, in comparison with the single CNN Siamese deep network without CFL. The use of the hybrid topology, instead of the network with only CNNs, increases the top rank performance of our model by at least 7 percentage points over the all data sets. The network pre-trained by full CFL increases the hit rate of our model, in the top rank performance, by at least 4 percentage points, in comparison with the network without CFL.

The graphic analysis of the pair distances, accomplished in Section 3.2.5, was also considered. Figures 3.13, 3.14, 3.15 and 3.16 show the distribution of the distances, projected on the

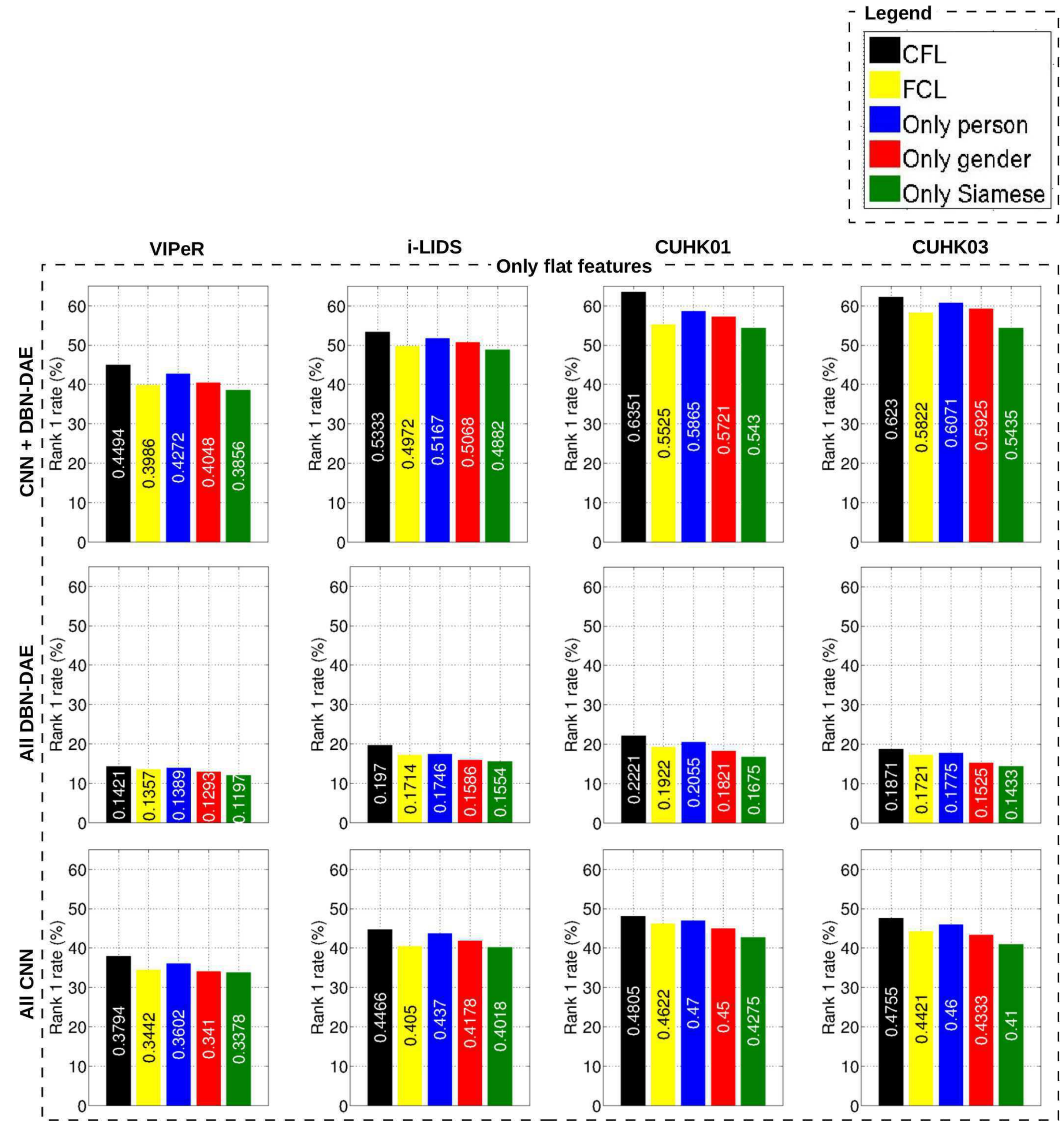
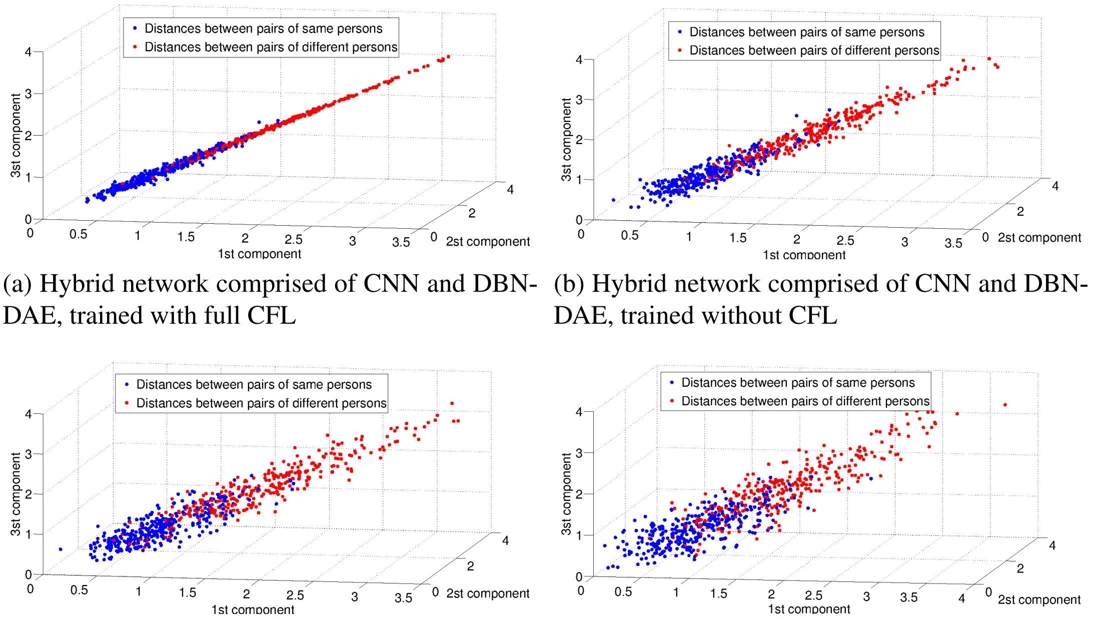


Figure 3.12: Comparative evaluation of different network topology and configurations: (i) CNN and DBN-DAE, (ii) all the networks comprised of DBN-DAEs, and (iii) all the networks comprised of CNNs. This comparative evaluation was done in the four data sets: VIPeR, i-LIDS, CUHK01 and CUHK03. Black, yellow, blue, red and green bars depict the network performance: With full CFL, FCL, with only knowledge about person, with only knowledge about gender and without CFL, respectively (see the legend in the top right of the figure).



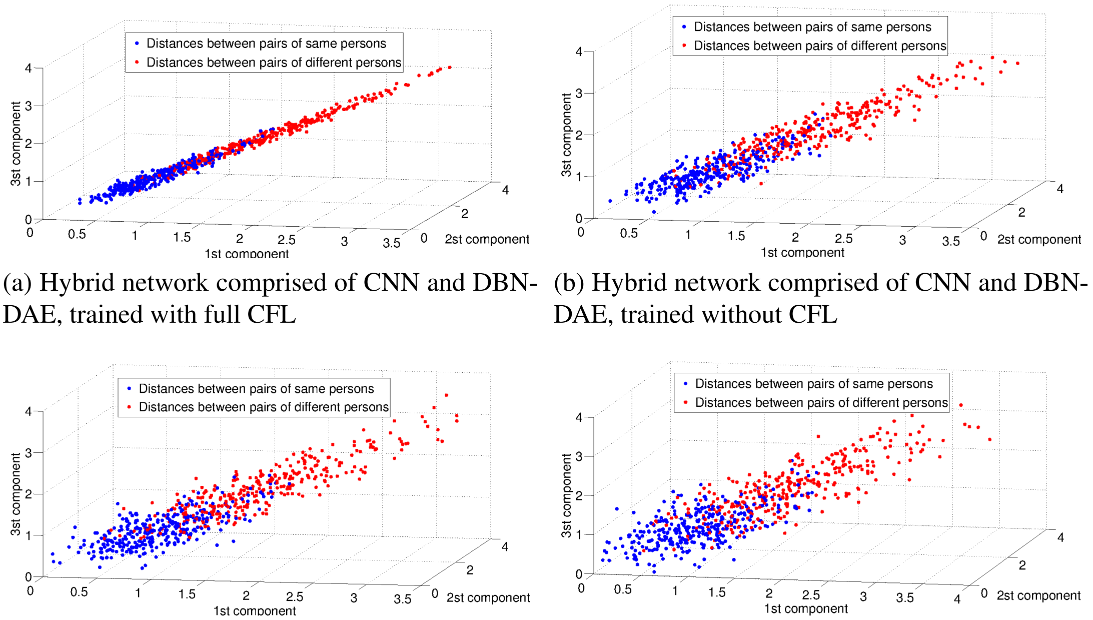
(a) Hybrid network comprised of CNN and DBN-DAE, trained with full CFL

(b) Hybrid network comprised of CNN and DBN-DAE, trained without CFL

(c) All the hybrid network comprised of CNN, trained with full CFL

(d) All the hybrid network comprised of CNN, trained without CFL

Figure 3.13: Distance distributions over VIPeR data set, projected into the first three principal components, of four network topology and training configurations.



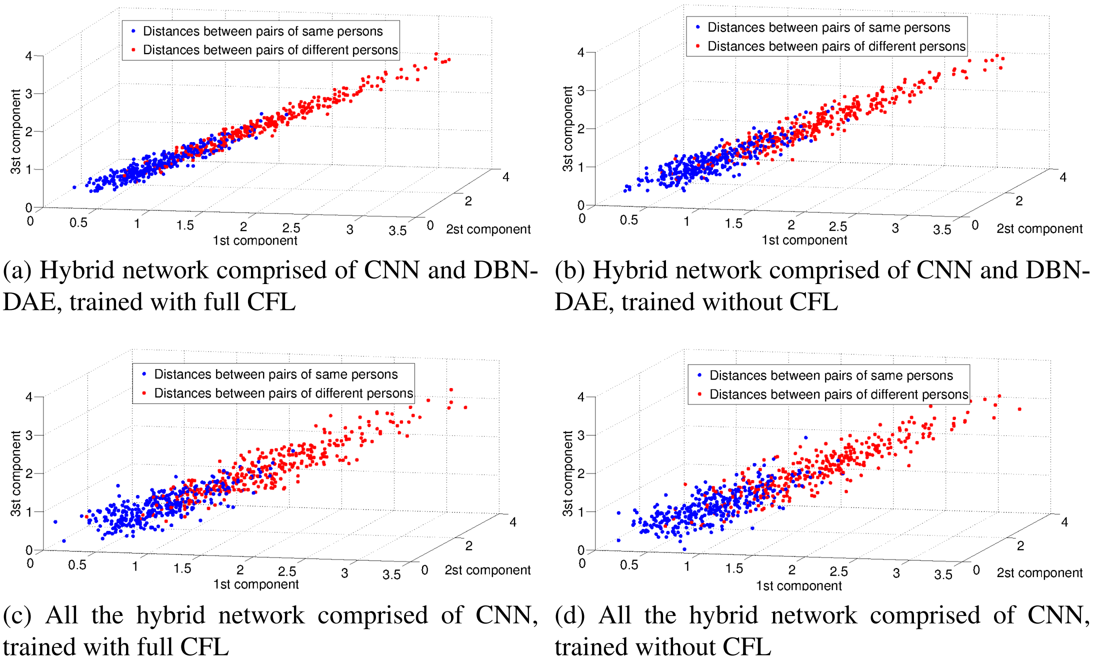
(a) Hybrid network comprised of CNN and DBN-DAE, trained with full CFL

(b) Hybrid network comprised of CNN and DBN-DAE, trained without CFL

(c) All the hybrid network comprised of CNN, trained with full CFL

(d) All the hybrid network comprised of CNN, trained without CFL

Figure 3.14: The same plot showed in Fig. 3.13 over iLIDS data set



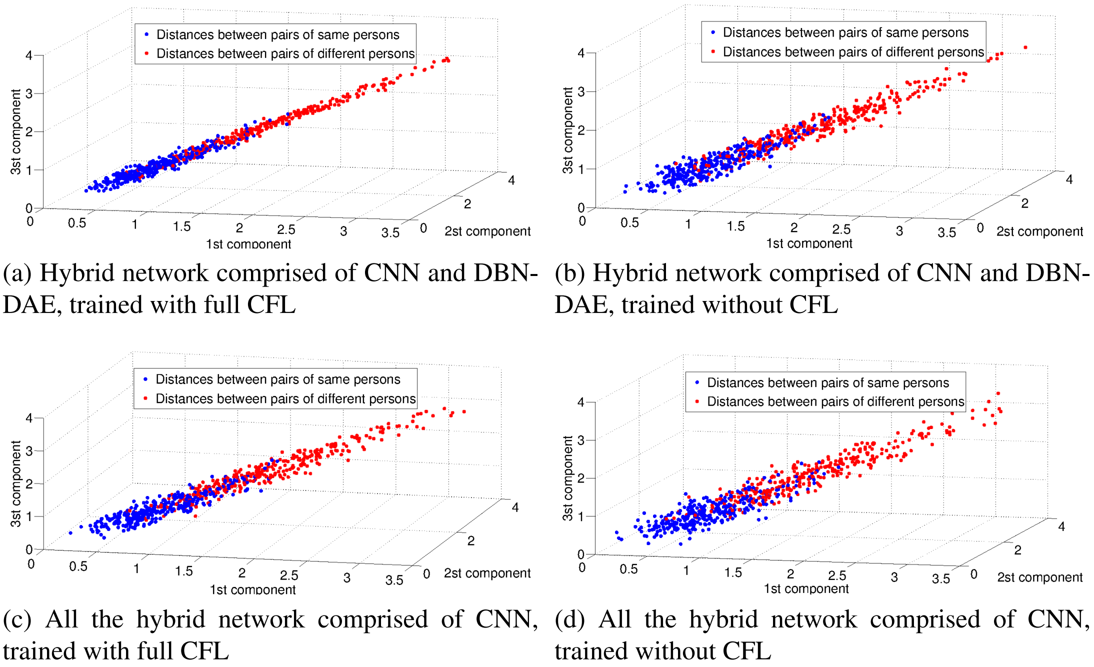
(a) Hybrid network comprised of CNN and DBN-DAE, trained with full CFL

(b) Hybrid network comprised of CNN and DBN-DAE, trained without CFL

(c) All the hybrid network comprised of CNN, trained with full CFL

(d) All the hybrid network comprised of CNN, trained without CFL

Figure 3.15: The same plot showed in Fig. 3.13, over CUHK01 data set.



(a) Hybrid network comprised of CNN and DBN-DAE, trained with full CFL

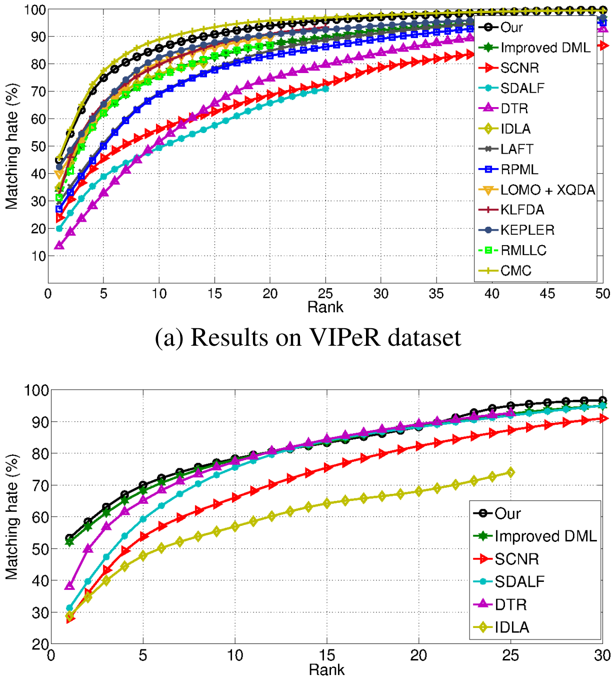
(b) Hybrid network comprised of CNN and DBN-DAE, trained without CFL

(c) All the hybrid network comprised of CNN, trained with full CFL

(d) All the hybrid network comprised of CNN, trained without CFL

Figure 3.16: The same plot showed in Fig. 3.13 over CUHK03 data set

three first principal components, considering four network topology and training configurations: (i) Hybrid network comprised of CNN and DBN-DAE, trained with full CFL (our best config-



(a) Results on VIPeR dataset

(b) Results on i-LIDS dataset

Figure 3.17: Cumulative curves of our CFL against other methods over VIPeR and i-LIDS datasets

uration); (ii) Hybrid network comprised of CNN and DBN-DAE trained without CFL (only the siamese network training); (iii) All the hybrid network comprised of CNN, trained with full CFL; and (iv) all the hybrid network comprised of CNN, trained without CFL. As in the Section 3.2.5, we choose the first selected testing subset to conduct this analysis. It is noteworthy that the distance between the center of mass of the red and blue point cloud is the greatest in our best topology and training configuration. This latter observation gives a hint about why our best framework configuration achieve the best performance. Table 3.7 and 3.8 show the average value among ten values of y, computed from each selected testing set, for each network topology and training configurations. Each one of the ten values was computed with the max function. The minimum average values, presented in Table 3.7 and 3.8, were achieved with our best network configuration.

Table 3.9 shows the number of the Siamese training iterations just after reaching a mean value, e, of a loss function, in five training configurations with our best hybrid network topology: (i) Full CFL training; (ii) FCL training; (iii) Only person network training before Siamese; (iv) Only gender network training before Siamese; and (v) Only siamese network training (without CFL). The number of iterations was the lowest one when the network was trained with full CFL.

Table 3.7: Hybrid network topology and training configuration performance measured by the mean value among ten values of y. Each one of the ten value was computed by the max function over ten selected training subsets, on VIPeR and iLIDS data sets.

VIPeR data set	
Hybrid network topology and training configuration	Average of y
CNN + DBN DAE with full CFL	**0.5126**
CNN + DBN DAE with FCL	0.5409
CNN + DBN DAE with only person CFL	0.5348
CNN + DBN DAE with only gender CFL	0.5392
CNN + DBN DAE without CFL	0.5482
All hybrid network comprised of DBN DAE with full CFL	0.7194
All hybrid network comprised of DBN DAE with FCL	0.7248
All hybrid network comprised of DBN DAE with only person CFL	0.7215
All hybrid network comprised of DBN DAE with only gender CFL	0.7267
All hybrid network comprised of DBN DAE without CFL	0.7344
All hybrid network comprised of CNN with full CFL	0.5894
All hybrid network comprised of CNN with FCL	0.6245
All hybrid network comprised of CNN with only person CFL	0.5921
All hybrid network comprised of CNN with only gender CFL	0.6247
All hybrid network comprised of CNN without CFL	0.6401
iLIDS data set	
Hybrid network topology and training configuration	Average of y
CNN + DBN DAE with full CFL	**0.5024**
CNN + DBN DAE with FCL	0.5205
CNN + DBN DAE with only person CFL	0.5182
CNN + DBN DAE with only gender CFL	0.5195
CNN + DBN DAE without CFL	0.5310
All hybrid network comprised of DBN DAE with full CFL	0.6314
All hybrid network comprised of DBN DAE with FCL	0.6515
All hybrid network comprised of DBN DAE with only person CFL	0.6443
All hybrid network comprised of DBN DAE with only gender CFL	0.6478
All hybrid network comprised of DBN DAE without CFL	0.6673
All hybrid network comprised of CNN with full CFL	0.5100
All hybrid network comprised of CNN with FCL	0.5422
All hybrid network comprised of CNN with only person CFL	0.5312
All hybrid network comprised of CNN with only gender CFL	0.5343
All hybrid network comprised of CNN without CFL	0.5495

3.3.3 Comparative performance evaluation

After choosing the best overall architecture (illustrated in Fig. 3.2b), the performance of our proposed framework was compared with 16 state-of-the-art methods: Improved Deep Met-

Table 3.8: The same performance measure as in Table 3.7, over CUHK01 and CUHK03 data sets.

CUHK01 data set	
Hybrid network topology and training configuration	Average of y
CNN + DBN DAE with full CFL	**0.5013**
CNN + DBN DAE with FCL	0.5399
CNN + DBN DAE with only person CFL	0.5288
CNN + DBN DAE with only gender CFL	0.5297
CNN + DBN DAE without CFL	0.5482
All hybrid network comprised of DBN DAE with full CFL	0.6992
All hybrid network comprised of DBN DAE with FCL	0.7164
All hybrid network comprised of DBN DAE with only person CFL	0.7199
All hybrid network comprised of DBN DAE with only gender CFL	0.7145
All hybrid network comprised of DBN DAE without CFL	0.7246
All hybrid network comprised of CNN with full CFL	0.5786
All hybrid network comprised of CNN with FCL	0.6137
All hybrid network comprised of CNN with only person CFL	0.5893
All hybrid network comprised of CNN with only gender CFL	0.6134
All hybrid network comprised of CNN without CFL	0.6422
CUHK03 data set	
Hybrid network topology and training configuration	Average of y
CNN + DBN DAE with full CFL	**0.5003**
CNN + DBN DAE with FCL	0.5197
CNN + DBN DAE with only person CFL	0.5135
CNN + DBN DAE with only gender CFL	0.5178
CNN + DBN DAE without CFL	0.5299
All hybrid network comprised of DBN DAE with full CFL	0.6213
All hybrid network comprised of DBN DAE with FCL	0.6463
All hybrid network comprised of DBN DAE with only person CFL	0.6421
All hybrid network comprised of DBN DAE with only gender CFL	0.6456
All hybrid network comprised of DBN DAE without CFL	0.6587
All hybrid network comprised of CNN with full CFL	0.5122
All hybrid network comprised of CNN with FCL	0.5356
All hybrid network comprised of CNN with only person CFL	0.5223
All hybrid network comprised of CNN with only gender CFL	0.5343
All hybrid network comprised of CNN without CFL	0.5464

ric Learning (DML) [Yi et al., 2014a], Semantic Color Names and Rankboost (SCNR) [Kuo et al., 2013], Symmetric-driven accumulation of local features (SDALF) [Bazzani et al., 2013], Domain Transfer support vector Ranking (DTR) [Ma et al., 2013], Improved Deep Learning Architecture (IDLA) [Ahmed et al., 2015], Locally Aligned Feature Transformation (LAFT)

Table 3.9: Number of training iterations in the Siamese network, just after reaching a mean value, e, of the loss function ($e = 10^{-3}$)

VIPeR data set	
Training configuration	Number of iterations
Full CFL training	**7520**
FCL training	12030
Only person network training before siamese	13522
Only gender network training before siamese	14050
Only siamese network training (without CFL)	18423
i-LIDS data set	
Training configuration	Number of iterations
Full CFL training	**8232**
FCL training	12234
Only person network training before siamese	13722
Only gender network training before siamese	15225
Only siamese network training (without CFL)	21423
CUHK01 data set	
Training configuration	Number of iterations
Full CFL training	**7124**
FCL training	13243
Only person network training before siamese	14343
Only gender network training before siamese	14877
Only siamese network training (without CFL)	19356
CUHK03 data set	
Training configuration	Number of iterations
Full CFL training	**8542**
FCL training	13454
Only person network training before siamese	13723
Only gender network training before siamese	15562
Only siamese network training (without CFL)	20023

[Li and Wang, 2013], Relaxed Pairwise Learned Metric (RPLM) [Hirzer et al., 2012], Local Maximal Occurrence Representation and Metric Learning (LOMO+XQDA) [Liao et al., 2015], Kernel-based Metric Learning (KFLDA) [Xiong et al., 2014], Deep Feature Learning with Relative Distance Comparison (DFLRDC) [Ding et al., 2015], Large Margin Nearest Neighbor (LMNN) [Weinberger et al., 2006], Metric Learning by Collapsing Classes (MCC) [Globerson and Roweis, 2005], Filter Paring Neural Network (FPNN) [Li et al., 2014], Relevance metric learning by exploiting listwise similarities (RMLLC) [Chen et al., 2015], Learning to rank with metric ensembles (CMC) [Paisitkriangkrai et al., 2015] and Kernelized saliency-based through multiple metric learning (KEPLER) [Martinel et al., 2015].

Table 3.10: Comparative analysis of our CFL on VIPeR data set. Each value corresponds to a hit rate score of a method in a specific rank (*rank 14 instead of 15, for IDLA).

Rank Method	1	5	10	15*	20	25	30	50
Our	0.4494	0.7500	0.8576	0.9082	0.9399	0.9589	0.9715	**0.9968**
CMC	**0.4590**	**0.7750**	**0.8890**	-	**0.9580**	-	-	0.9950
improved DML	0.3440	0.6215	0.7589	0.8256	0.8722	0.8965	0.9228	0.9652
SCNR	0.2392	0.4557	0.5623	0.6266	0.6873	0.7278	0.7880	0.8671
SDALF	0.1987	0.3889	0.4937	0.5759	0.6573	0.7089	-	-
DTR	0.1345	-	0.5158	-	0.7468	-	-	0.9272
IDLA	0.3481	0.6424	0.7627	0.8038	-	-	-	-
LAFT	0.2960	-	0.6931	-	-	0.8870	-	0.9680
RPML	0.2700	-	0.6900	-	0.8300	-	-	0.9500
LOMO+XQDA	0.4000	-	0.8051	-	-	0.9108	-	-
KLFDA	0.3233	0.6578	0.7972	0.8699	0.9095	0.9346	-	-
KEPLER	0.4241	-	0.8237	-	0.9070	-	-	0.9706
RMLLC	0.3127	0.6212	0.7531	-	0.8671	-	-	-

Table 3.11: Comparative analysis of our CFL on i-LIDS data set. Each value corresponds to a hit rate score of a method in a specific rank.

Rank Method	1	5	10	15	20	25	30
Our	**0.5333**	**0.7000**	**0.7833**	0.8333	0.8832	**0.9333**	**0.9500**
CMC	0.5034	-	-	-	-	-	-
DFLRDC	0.5210	0.6820	0.7800	0.8360	0.8880	-	0.9500
LMNN	0.2800	0.5380	0.6610	0.7550	0.8230	-	0.9100
MCC	0.3130	0.5930	0.7560	0.8400	0.8830	-	0.9500
KLFDA	0.3802	0.6512	0.7738	**0.8440**	**0.8919**	0.9267	-
SDALF	0.2880	0.4778	0.5696	0.6424	0.6804	0.7405	-

Table 3.12: Comparative analysis of our CFL on CUHK01 and CUHK03 data sets.

Data set Method	CUHK01	CUHK03	Data set Method	CUHK01	CUHK03
Our	**0.6351**	**0.6230**	IDLA	0.4753	0.5474
CMC	0.5340	0.6210	FPNN	0.2787	0.2065
SDALF	0.1033	-	LOMO+XQDA	0.6321	0.5230
DTR	0.0804	-	LMNN	0.1598	-

Table 3.13: Comparative analysis between our CFL and RMLLC method Chen et al. [2015] on i-LIDs in multiple training/testing data samples.

Methods	Number of training/testing people: 89/30			
	Rank=1	Rank=5	Rank=10	Rank=20
Our	**0.5667**	**0.8700**	**0.9333**	0.9633
RMLLC	0.5653	0.8448	0.9316	**0.9873**
Methods	Number of training/testing people: 69/50			
	Rank=1	Rank=5	Rank=10	Rank=20
Our	**0.4660**	**0.7440**	0.8420	0.9200
RMLLC	0.4653	0.7301	**0.8457**	**0.9345**
Methods	Number of training/testing people: 39/80			
	Rank=1	Rank=5	Rank=10	Rank=20
Our	**0.3525**	0.5862	**0.7263**	0.8513
RMLLC	0.3513	**0.5969**	0.7257	**0.8523**

In Figure 3.17, it is noteworthy that our CFL approach has demonstrated superior performance on i-LIDS and it was competitive on VIPeR, against all compared methods. Tables 3.10 and 3.11 summarize the discrete values of the cumulative curves of our CFL and other methods on VIPeR and i-LIDs, respectively. On VIPeR, our method achieved the second best performance in the ranks equals to or below 30 (CMC method achieved the best performance). On i-LIDs, our method was superior at almost all ranks, except in the ranks 15 an 20, where our method showed a slightly lower performance than KLDFA. The best performance in the top rank was achieved by our method on CUHK01 and CUHK03 data sets (see Table 3.12).

Table 3.13 shows a different protocol on i-LIDs, followed by the RMLLC method Chen et al. [2015]. When compared to ours, we obtained superior performance at least in top rank.

3.4 Closure

In this chapter, a framework to learn discriminative and trainable features for the problem of person re-identification was proposed. From the philosophical point-of-view, the framework tries to mimic the human learning process. The basic idea was to learn features from generic to specific concepts. From the technical point-of-view, the topology of the framework was comprised of three CNN and a DBN-DAE. The local features were learned by the CNN and the global ones were learned by the DBN-DAE, during four framework learning steps: 1 - DBN-DAE training, to leaning global features; 2 - Person classification; 3 - Gender classification; and 4 - Person re-identification. From one step to the next one, the learning was transferred by the machine transfer learning technique. This machine learning approach was called CFL.

The experimental analysis shows the best performance of our CFL training approach and network topology, in comparison with other topology and training configurations. Also, comparison of our framework with 16 other state-of-art methods shows that our proposed approach was competitive over ViPER, obtaining the best performance over i-LIDS, CUHK01 and CUHK03 data sets.

CHAPTER 4

Wrapping CFL features into covariance matrices

Contents

4.1 Introduction

According to Tuzel et al. [2006], the joint representation of several different features through histograms is exponential with the number of features. Instead of a joint distribution of the features, Tuzel et al. [2006] proposed a new way to integrate features in a reduced dimensional space by using covariance matrices. Covariance descriptors are derived from a set of image statistics, computed inside image regions, after applying a set of filters. In [Tuzel et al., 2006], nine filters were applied in a input image, generating nine image maps, as illustrated in Fig. 4.1.

After carefully observing Fig. 4.1, it is possible to find a close similarity between the image maps and the feature maps – this latter one presented in the intermediate layers of a CNN. Although with similar structures, the way of extracting each type of map is different. While image maps are computed from previously designed convolutional filters, feature maps of a

CNN are achieved in a trainable way (see Section 2.2.1). This characterizes the feature maps as adaptive to the domain of the problem. Fixed filters, as computed in the image maps in Fig. 4.1, can be suitable for a particular problem, while not efficient for the others.

Considering the adaptive nature of the feature maps of a CNN, we propose to use the covariance descriptors over the intermediate CNN layers. The hypothesis is that by considering adaptive maps, it is possible to increase the discriminative power of covariance descriptors. We called those descriptors by convolutional covariance features (CCF). The proposed CCF, their integration with CFL features and their performance analysis will be presented in the remainder of this chapter.

4.2 Convolutional covariance features

The idea of the covariance descriptors is not only to reduce the image dimension, but also to provide a descriptor robust to lighting change and nonrigid motion. A d×d region covariance descriptor, $\mathbf{C}_R$, in a image map region, R, is extracted as

$$\mathbf{C}_R = \frac{1}{n-1}\sum_{k=1}^{n}(\mathbf{z}_k - \mu)(\mathbf{z}_k - \mu)^T \,, \tag{4.1}$$

where $\{\mathbf{z}_k\}_{k=1..n}$ is the d-dimensional feature points, and μ is the d-dimensional vector, providing the means of each feature map region. The kth $\mathbf{z}$ and its jth element is a point into the region R of the jth image feature, given by

$$\{F_j(x,y)\}_{j=1..d} = \phi(I,x,y) \,, \tag{4.2}$$

where ϕ can be any mapping function, such as intensity, color, gradient and filter responses of n_{th} derivatives of the image pixels (see Fig. 4.1). Nine mapping functions were used in [Tuzel et al., 2006], defined as

$$\begin{aligned}\mathbf{F}(\mathbf{x},\mathbf{y}) = \Bigg[x \quad y \quad R(x,y) \quad G(x,y) \quad B(x,y) \quad \left|\frac{\partial I(x,y)}{\partial x}\right| \\ \left|\frac{\partial I(x,y)}{\partial y}\right| \quad \left|\frac{\partial^2 I(x,y)}{\partial x^2}\right| \quad \left|\frac{\partial^2 I(x,y)}{\partial y^2}\right| \Bigg] \,,\end{aligned} \tag{4.3}$$

where R, G and B are RGB color values, I denotes color intensity, and x and y are pixel coordinates. Figure 4.1 shows the visual representation of $\mathbf{F}$.

While the number of image maps proposed in [Tuzel et al., 2006] is nine, the number of

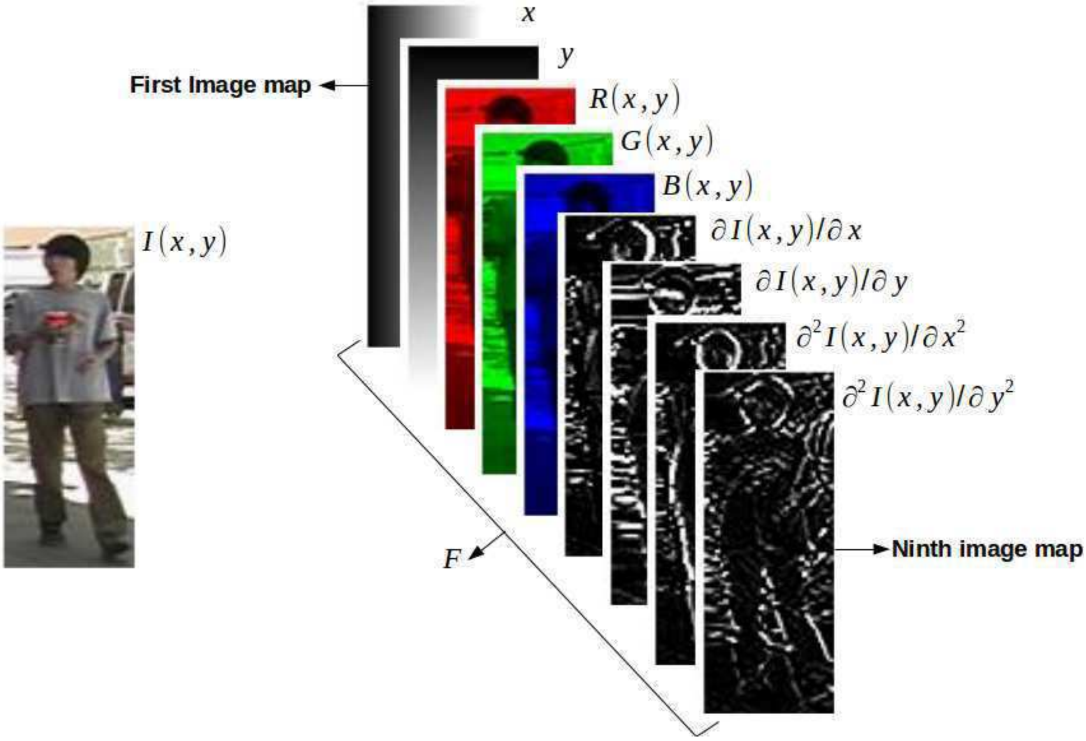


Figure 4.1: Original covariance descriptor extraction, as described in [Tuzel et al., 2006]. Each image represents an image map.

feature maps is 64, located in the intermediate CNN layers of the hybrid network (see Fig. 4.2). A feature map in an intermediate CNN layer is similar to an image map in $\mathbf{F}$. Figure 4.3 depicts the scheme of extraction of the CCF, which is comprised of a set of local covariance matrices computed over R regions of a feature map, according to Eq. 4.1. The size of a covariance matrix in the CCF is 64×64, since there are 64 feature maps in each intermediate layer. Examples of R regions are depicted by the green boxes in the left upper and right down corners of the zoomed area in Fig. 4.3.

A total of twelve CCF are computed on the three CNN sub-net of the hybrid network: Head, Torso and Legs, over each CNN layer – C1, S2, C3 and S4 (see Fig. 4.2 for a complete description of the CNN layer). Each local covariance matrix, $\mathbf{C}_R$, is extracted in an 8×8 region; In turn, each region is overlapped to its neighbors by 50% of the region size in horizontal and vertical directions. A total of 121 $\mathbf{C}_R$ on C1, 25 on S2 and C3, and 4 on S4 layers, counting a total of 175 local covariance matrix for each CNN body part subnet. Algorithm 1 describes how the CCF are extracted step-by-step, starting with the hybrid network CFL training approach.

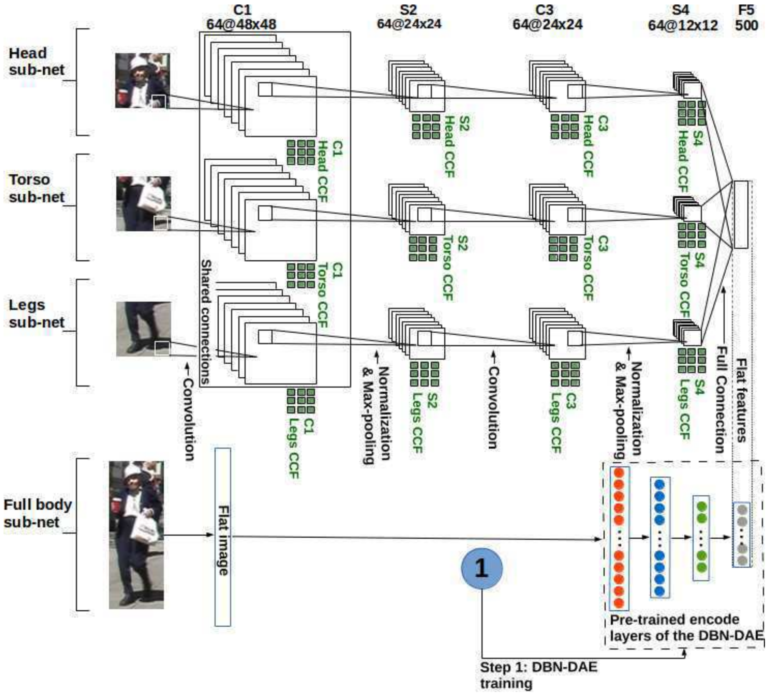


Figure 4.2: Outline of our proposed CCF. Twelve CCF (green boxes) are extracted from each CNN layer, after CFL training.

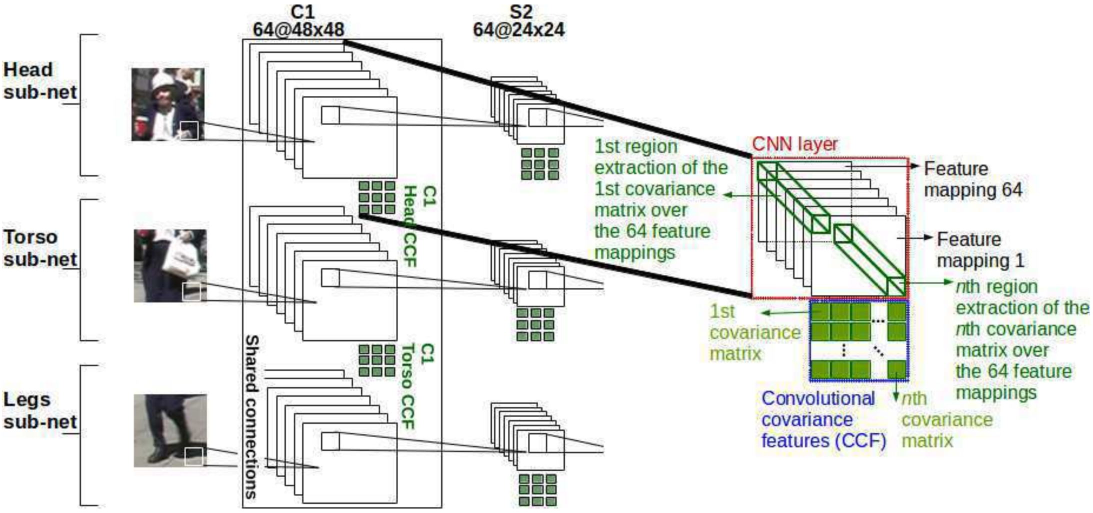


Figure 4.3: CCF extraction. The zoomed region depicts how the CCF are extracted. The CCF are comprised of a set of covariance matrices, each of them extracted in a region R (3D green rectangle) over the feature maps of a convolutional layer.

4.3 Integration of CCF and flat features

Since CCF are only extracted from the three body part subnets of the hybrid network, CCF do not take into account the global features from the top encoder layer of the DBN-DAE, and the

Algorithm 1 Extracting the CCF

```
procedure CFL
    Train the DBN-DAE (see Section 3.2.1)
    Form the hybrid network NET with three human body CNN sub-nets and the encode layer of the DBN-DAE, according to Fig. 4.2
    Train $NET$ with the person network (see Step 1 in Fig. 3.2b)
    Train $NET$ with the gender network (see Step 2 in Fig. 3.2b)
    Create the Siamese Network ($SN$) with two identical $NET$s (see Fig. 3.2b)
    Train the Siamese Network ($SN$) (see Step 3 in Fig. 3.2b)
return $SN$
end procedure
procedure CCFExtration($SN$)
    CCF$_{set1}$ ← {}        ▷ Store all the CCF extracted from the first $NET$ of $SN$
    CCF$_{set2}$ ← {}        ▷ Store all the CCF extracted from the second $NET$ of $SN$
    for each $NET \in SN$ do
        for each CNN sub-net $S \in NET$ do
            for each CNN layer $c \in S$ do
                $M$ ← A set of 64 feature maps of $c$
                $B$ ← A set of 8×8 overlapped regions $R$ extracted from $M$
                CCF ← {}        ▷ Current CCF (A set of local covariance matrices)
                for each region $R \in B$ do
                    Compute $C_R$ into the region $R$ by Equation 4.1
                    Include $C_R$ in the CCF
                end for
                Include CCF into CCF$_{set1}$ for the first $NET$ or CCF$_{set2}$ for the second one
            end for
        end for
    end for
return CCF$_{set1}$, CCF$_{set2}$
end procedure
```

local features from the top layer of CNN subnets. That is because the DBN-DAE is comprised of full-connected layers, and there is no way to extract covariance matrices from them. Therefore a way of integration of CCF and the flat features is necessary to be conceived in order to keep the features not covered by the CCF. This integration can not be done in a vector space, since CCF lie in a Riemann space, while the flat features are in an Euclidean space. Then, we propose a method to integrate the CCF and the flat features, which follows the steps (depicted in Fig. 4.4): Four pairs of source and target images (each image in the pair contains the original data set image and three noised ones) were used to compute the distances of the CCF and the flat features among the two people in the pair to be evaluated. A total of 16 image pairs, arising from the combination of the source images and the target one, produces 16 final CCF and 16

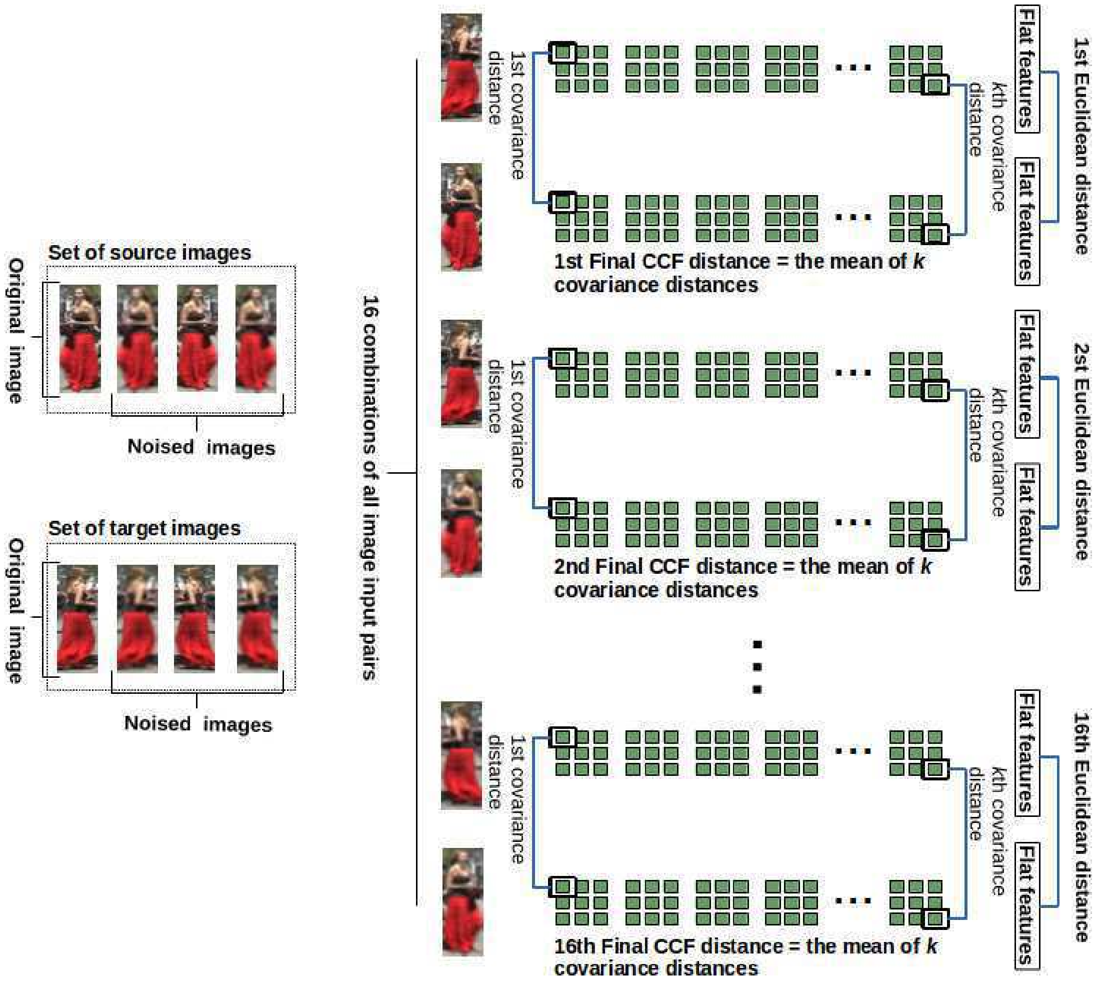


Figure 4.4: Integration of the CCF and flat features. Four source and target images are used to evaluate the similarity between two people. CCF and flat features of the 16 image pairs are extracted by the Siamese Network. For each of those 16 pairs, one final CCF and one Euclidean distance is computed. A final CCF distance is the mean of k covariance distances computed between two covariance matrices from the CCF pairs. The final similarity score is the maximum value among those 32 distances.

Euclidean distances. The final 16 CCF are achieved by computing the mean of the k covariance distances of each pair of initial CCF distance, according to Fig. 4.4. Euclidean distances are computed among each pair of flat features (source and target images). As in CFL framework, the final similarity score is the maximum value among the 16 final CCF and 16 euclidean distances (according demonstration in the last chapter).

The function proposed in [Förstner and Moonen, 2003] was used to compute the distance,

$\rho(.)$, between two covariance matrices of a pair of CCF, defined as

$$\rho(\mathbf{C}_1, \mathbf{C}_2) = \sqrt{\sum_{i=1}^{n} \ln^2 \lambda_i(\mathbf{C}_1, \mathbf{C}_2)}\,, \tag{4.4}$$

where $\mathbf{C}_1$ and $\mathbf{C}_2$ denotes two covariance matrices, $\{\lambda_i(\mathbf{C}_1, \mathbf{C}_2)\}_{i=1..n}$ are the generalized eigenvalues of $\mathbf{C}_1$ and $\mathbf{C}_2$ computed by

$$\lambda_i \mathbf{C}_1 \mathbf{x}_i - \mathbf{C}_2 \mathbf{x}_i = 0 \qquad i = 1..d\,, \tag{4.5}$$

where $\mathbf{x}_i \neq 0$ are the generalized eigenvectors.

The Algorithm 2 describes in details how our proposed feature integration was carried out.

4.4 Experimental analysis

Although the analysis in the previous chapter has found the best hybrid network topology and CFL training configuration, the best configuration can be changed by the addition of the CCF. Therefore, the experiments performed in the previous chapter are replicated here, taking into account the CCF. As in the last chapter, a similar comparative analysis with other state-of-the-art methods was also done here, adding two new methods found in [Zeng et al., 2015] and [Hirzer et al., 2011]. Both of them use covariance descriptors to solve the problem of person re-identification. The addition of these two new methods was useful to present the improvement of the CCF in comparison with covariance descriptors based methods.

4.4.1 Selection of the hybrid network topology, CFL and CCF

As in Section 3.3.2, two steps in the performance assessment of the proposed CFL were followed: (i) Analysis of the hybrid network architecture, and (ii) Analysis of the CFL approach. While including the proposed CCF, three more performance assessments were added on the previous combinations: (i) Using only flat features (the same configuration of CFL performance evaluation), (ii) using only CCF, and (iii) using the integration of the flat and CCF features.

As in Fig. 3.12, the best result is found by considering the intersection between the columns in each plot of each data set and the rows containing the type of deep architecture inside the hybrid network (see Fig. 4.5). Bars into the plots correspond to the performance of the type of transfer learning approach, considering: (i) CFL, (ii) FCL, (iii) only person, (iv) only gender and (v) only Siamese network, exactly as in the previous analysis accomplished.

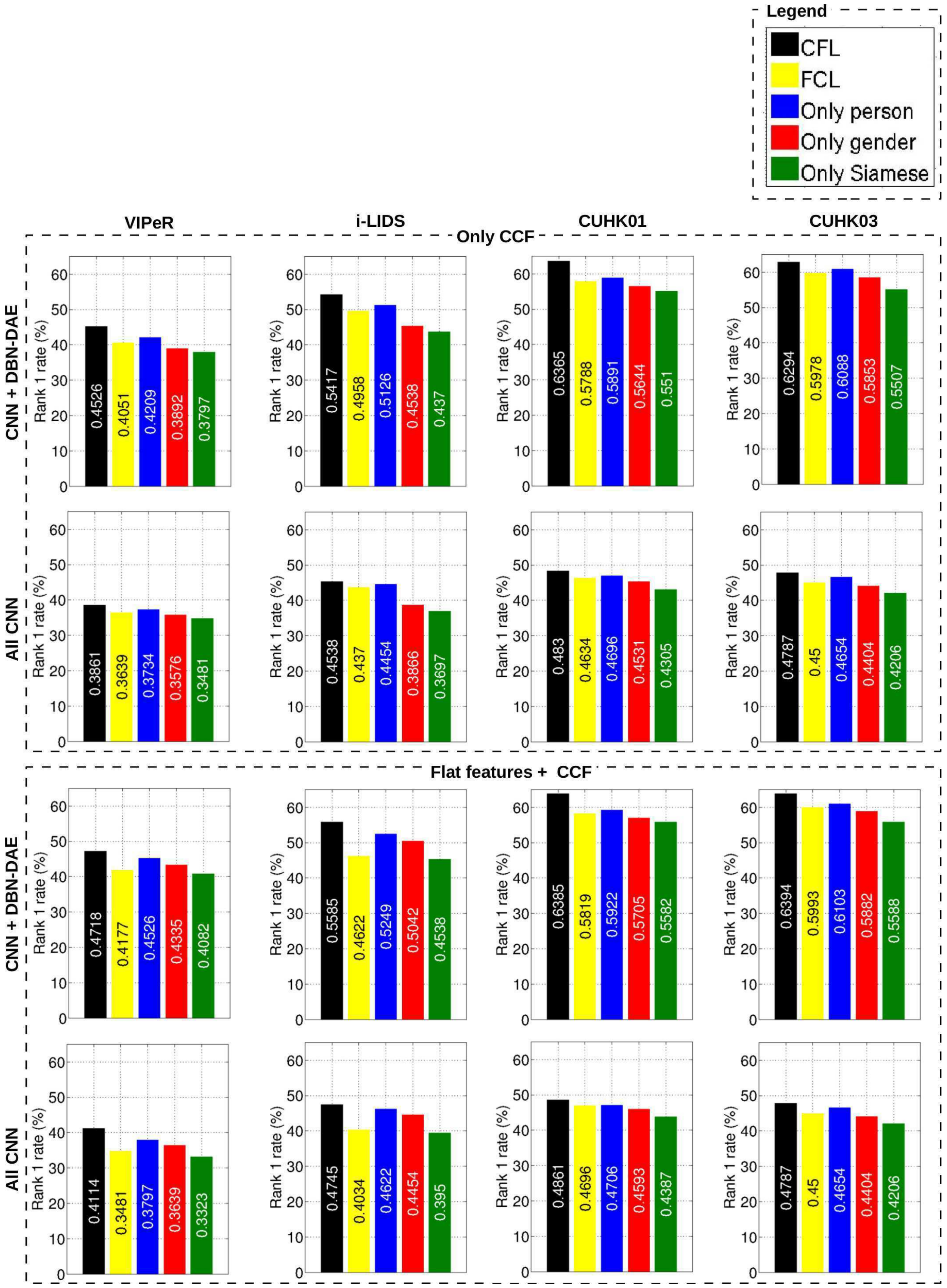


Figure 4.5: Comparative evaluation of different network topology and feature configurations. Plots on the top show results using only CCF, while plots on the bottom shows integration of flat features and CCF. All results are over VIPeR, i-LIDS, CUHK01 and CUHK03 data sets. Black, yellow, blue, red and green bars depict the network performance: With full CFL, FCL, with only knowledge about person, with only knowledge about gender and with only the Siamese network (without transfer learning), respectively.

Algorithm 2 Computing the similarity score between two people

```
procedure COMPUTESIMILARITY($I_s$,$I_t$,$SN$)    ▷ $I_s$ and $I_t$ are the 3D image source and target, respectively, and $SN$ is a trained Siamese Network.
    Create a set of image source $I_{sset}$ containing $I_s$ and three noised version of $I_s$ (see Fig. 4.4)
    Create a set of image targt $I_{tset}$ containing $I_t$ and three noised version of $I_t$ (see Fig. 4.4)
    $FCD_{set} \leftarrow \{\}$    ▷ The set of final CCF distances
    $ED_{set} \leftarrow \{\}$    ▷ The set of Euclidean distances
    for each image $i_s \in I_{sset}$ do
        for each image $i_t \in I_{tset}$ do
            Perform the feedforward algorithm in the $SN$ with the $I_s$ and $I_t$ inputs
            [CCF$_{set1}$, CCF$_{set2}$] $\leftarrow$ CCFExtration($SN$) (see Algorithm 1)
            $RD \leftarrow \{\}$    ▷ The set of local covariance distances
            for each CCF$_1 \in$ CCF$_{set1}$ and CCF$_2 \in$ CCF$_{set2}$ do
                for each covariance matrix $C_{R_1} \in$ CCF$_1$ and $C_{R_1} \in$ CCF$_2$ do
                    Compute the distance $\rho$ between $C_{R_1}$ and $C_{R_1}$ by Equation 4.4
                    Include $\rho$ into $RD$
                end for
            end for
            $FCD \leftarrow$ mean($RD$)
            Include $FCD$ into $FCD_{set}$
            Compute the Euclidean distance $ED$ between the flat features of the $SN$
            Include $ED$ into $ED_{set}$
        end for
    end for
    $FS \leftarrow$ max($ED_{set}$,$FCD_{set}$)    ▷ Final similarity score (see Fig. 4.4)
return $FS$
end procedure
```

The highest performance is achieved when **CNN and DBN-DAE are used as the topology of the hybrid deep network inside a full CFL strategy, and considering the integration of the CCF and the flat features** (black bars for these configurations contain the greatest results of top 1 rank). Results of the hybrid network shows that the network composed by CNN and DBN-DAE with full CFL improves the performance of the method, at least, in 6, 8, 14 and 14 percentage points on VIPeR, i-LIDs, CUHK01 and CUHK03 data sets, respectively, in comparison with the network with only CNN (type of deep network used in all other deep learning-based methods). It is also noteworthy that the full architecture achieved, at least, a performance improvement of 6, 4, 8 and 8 percentage points in comparison with the Siamese without transfer learning strategy, on VIPeR, i-LIDS, CUHK01 and CUHK03 data sets, re-

Table 4.1: Analysis of the CCF and flat features integration.

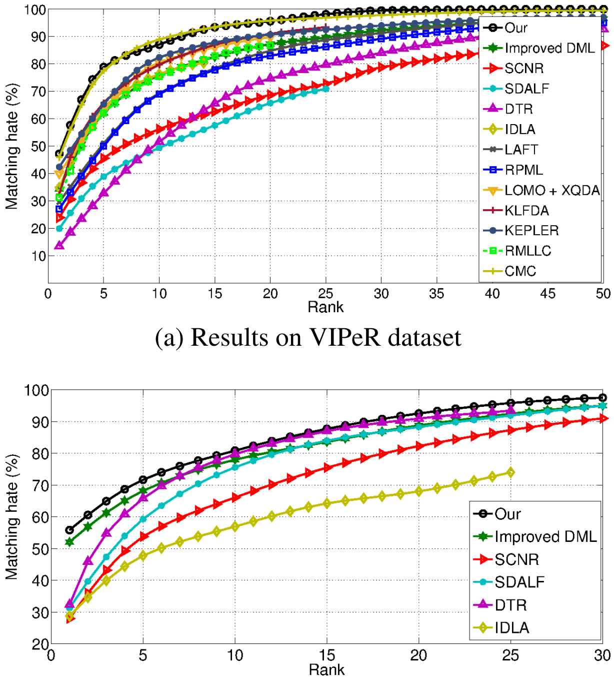
	VIPeR dataset						
Features	Rank=1	Rank=5	Rank=10	Rank=15	Rank=20	Rank=25	Rank=30
Flat	0.4494	0.7500	0.8576	0.9082	0.9399	0.9589	0.9715
Covariance	0.4526	0.7660	0.8640	0.9210	**0.9687**	0.9781	0.9843
Both	**0.4718**	**0.7884**	**0.8708**	**0.9338**	0.9559	**0.9813**	**0.9947**
	i-LIDS dataset						
Features	Rank=1	Rank=5	Rank=10	Rank=15	Rank=20	Rank=25	Rank=30
Flat	0.5333	0.7000	0.7833	0.8333	0.8832	0.9333	0.9500
Covariance	0.5417	0.7000	0.8085	0.8333	0.9000	**0.9417**	0.9752
Both	**0.5585**	**0.7168**	**0.8085**	**0.8771**	**0.9255**	0.9285	**0.9752**
	CUHK01 dataset						
Features	Rank=1	Rank=5	Rank=10	Rank=15	Rank=20	Rank=25	Rank=30
Flat	0.6351	0.7785	0.8797	0.9241	0.9557	0.9778	0.9905
Covariance	0.6365	0.7742	0.8733	0.9275	0.9577	0.9751	0.9943
Both	**0.6385**	**0.7800**	**0.8895**	**0.9371**	**0.9633**	**0.9800**	**0.9975**
	CUHK03 dataset						
Features	Rank=1	Rank=5	Rank=10	Rank=15	Rank=20	Rank=25	Rank=30
Flat	0.6230	0.7675	0.8733	0.9198	0.9495	0.9705	0.9899
Covariance	0.6294	0.7785	0.8798	0.9133	0.9445	0.9788	0.9922
Both	**0.6394**	**0.7833**	**0.8845**	**0.9377**	**0.9605**	**0.9833**	**0.9965**

spectively. Table 4.1 shows that by using the integration of the CCF and the flat features, the performance gain reaches more than 2 percentage points, in the experiments over VIPeR and i-LIDS, and more than 1 percentage point over CUHK01 and CUHK03 data sets, in comparison with the use of only flat features. Also in Table 4.1, results show that the best performance of the proposed feature integration was obtained until at least top 15 over all the evaluated data sets.

4.4.2 Comparative evaluation

Here, we followed the same comparative evaluation from Section 3.3.3, including two new methods, based on covariance descriptors: Hybrid Spatiogram and Covariance Descriptor (HSCD) [Zeng et al., 2015]; and Descriptive and Discriminative Classification (DDC) [Hirzer et al., 2011].

In Figure 4.6, it is noteworthy that our complete approach (CFL with CCF integration) has consistently demonstrated superior performance on VIPeR and i-LIDS data sets, against all compared methods. On VIPeR, our method obtained the best performance at all ranks, except in rank 10 and 20, against all compared methods (see Table 4.2). On i-LIDs, our method was



(a) Results on VIPeR dataset

(b) Results on i-LIDS dataset

Figure 4.6: Cumulative curves of the methods over VIPeR and i-LIDS datasets

Table 4.2: Comparative analysis on VIPeR data set. Each value corresponds to a hit rate score of a method in a specific rank (rank 14 instead of 15, for IDLA).

Rank / Method	1	5	10	15*	20	25	30	50
Our	**0.4718**	**0.7884**	0.8708	**0.9338**	0.9559	**0.9813**	**0.9947**	**1**
CMC	0.4590	0.7750	**0.8890**	-	**0.9580**	-	-	0.9950
improved DML	0.3440	0.6215	0.7589	0.8256	0.8722	0.8965	0.9228	0.9652
SCNR	0.2392	0.4557	0.5623	0.6266	0.6873	0.7278	0.7880	0.8671
SDALF	0.1987	0.3889	0.4937	0.5759	0.6573	0.7089	-	-
DTR	0.1345	-	0.5158	-	0.7468	-	-	0.9272
IDLA	0.3481	0.6424	0.7627	0.8038	-	-	-	-
LAFT	0.2960	-	0.6931	-	-	0.8870	-	0.9680
RPML	0.2700	-	0.6900	-	0.8300	-	-	0.9500
LOMO+XQDA	0.4000	-	0.8051	-	-	0.9108	-	-
KLFDA	0.3233	0.6578	0.7972	0.8699	0.9095	0.9346	-	-
KEPLER	0.4241	-	0.8237	-	0.9070	-	-	0.9706
RMLLC	0.3127	0.6212	0.7531	-	0.8671	-	-	-
DDC	0.1900	-	0.5200	-	-	0.6900	-	0.8000
HSCD	0.3120	-	0.8650	-	-	-	-	-

Table 4.3: Comparative analysis over i-LIDS data set. Each value corresponds to a hit rate score of a method in a specific rank.

Rank Method	1	5	10	15	20	25	30
our	**0.5585**	**0.7168**	**0.8085**	**0.8771**	**0.9255**	**0.9585**	**0.9752**
CMC [Paisitkriangkrai et al., 2015]	0.5034	-	-	-	-	-	-
DFLRDC	0.5210	0.6820	0.7800	0.8360	0.8880	-	0.9500
LMNN	0.2800	0.5380	0.6610	0.7550	0.8230	-	0.9100
MCC	0.3130	0.5930	0.7560	0.8400	0.8830	-	0.9500
KLFDA	0.3802	0.6512	0.7738	0.8440	0.8919	0.9267	-
SDALF	0.2880	0.4778	0.5696	0.6424	0.6804	0.7405	-
HSCD	0.3900	-	0.6600	-	-	-	-

Table 4.4: Top rank comparative analysis on CUHK01 and CUHK03 data set.

Data set Method	CUHK01	CUHK03
Our	**0.6385**	**0.6394**
CMC	0.5340	0.6210
SDALF	0.1033	-
DTR	0.0804	-
IDLA	0.4753	0.5474
FPNN	0.2787	0.2065
LOMO+XQDA	0.6321	0.5230
LMNN	0.1598	-

Table 4.5: Comparative analysis between our method and RMLLC method over i-LIDs data set in multiples training/testing data samples.

Methods	Number of training/testing persons: 89/30			
	Rank=1	Rank=5	Rank=10	Rank=20
Our	**0.5775**	**0.8892**	**0.9484**	0.9788
RMLLC	0.5653	0.8448	0.9316	**0.9873**
Methods	Number of training/testing persons: 69/50			
	Rank=1	Rank=5	Rank=10	Rank=20
Our	**0.4799**	**0.7622**	**0.8575**	**0.9384**
RMLLC	0.4653	0.7301	0.8457	0.9345
Methods	Number of training/testing persons: 39/80			
	Rank=1	Rank=5	Rank=10	Rank=20
Our	**0.3674**	0.5945	**0.7362**	**0.8677**
RMLLC	0.3513	**0.5969**	0.7257	0.8523

superior at all ranks (see Table 4.3), as well as, on CUHK01 and CUHK03 data sets (see Table 4.4). Even when comparing our method with covariance-based methods in Tables 4.2 and 4.3, our method showed the best performance against methods found in Zeng et al. [2015] and Hirzer et al. [2011].

When the RMLLC method is compared to ours, we obtained superior performance at all ranks, except in rank 20, when considering a training/testing rate of $89/30$, where our method looses by less than 1 percentage point (see Table 4.5).

4.5 Closure

This chapter showed that the features of CNN intermediate layers can also be useful for the data representation. Noting the similarities among the structure of a CNN intermediate layers and the image maps of the covariance descriptors, we proposed an adaptive covariance descriptor, called CCF. CCF were extracted from the CNN intermediate layers of the proposed hybrid network in order to improve the prediction performance of the person re-identification. Since CCF do not take into account the person global features from the top of the DBN-DAE and the local features from the top of the CNN layer, an integration of the CCF and the flat features was proposed. CCF and their integration with the flat features, learned during the CFL training, have improved the performance of our framework in all compared data sets. This new way to extract the covariance descriptors achieved the best performance in comparison with the state-of-the-arts methods that use conventional covariance descriptors for person re-identification. The reason for the highest performance with CCF seems to be due to their adaptive characteristics, since CCF are extracted from the trained convolutional layers of the CNN.

Part III

Closure

CHAPTER 5

Discussion and Conclusion

In this work, a novel coarse-to-fine deep learning approach has been proposed to learn discriminative features for person re-identification. CFL deep-based framework relies on acquiring the necessary knowledge to identify a person by transferring the learning achieved in each step of the network training. CCF and its integration with deep features have also been proposed. The integration relies on a novel way of applying the covariance descriptors over the convolutional layers of a deep hybrid network, as well as assembling flat features (in vector space) with CCF (in Riemann space). In this earlier stage of the method conception, the idea was to implement, to evaluate and to raise all the necessary information about the characteristics of the proposed method in isolation. Real-time implementation, as well as the evaluation of the method in video, were not covered.

Although a quite superior top 1 performance over 18 other state-of-the-art methods, it is noteworthy that the application of the method in real scenarios can take time with other issues being faced. Also, without measuring processing time or evaluating complexity of the algorithms involved in the CFL framework, it is observable that some issues must be dealt, such as: the way that the covariances are computed, indexing and retrieving the target image data sets (which lately can increase exponentially during a real-time application), time-consuption during the prediction phase due to the distance space measurement and parallelism in processing the convolutional deep features. All these issues were open in our work, but can be further investigated in future researches.

The training stage of CFL framework was carried out on a multi-core graphical processing unit (GPU) using CAFFE framework [Jia et al., 2014]. However computer memory was a bottleneck, limiting the number of training samples. This limitation definitely did not lead the performance of CFL to all its potential. This situation can be overcame by exploiting multiple parallel machines or clusters.

The complete accomplishment of a person re-id system demands a previous detection methods. In our work, detection was considered perfect by using cropped images for performance assessment. Howeve,r in real applications, it will be necessary to include a previous detection stage that ultimately uses context-aware or semantic information to deal with the wilderness of situations in real environments. Some works that exploit semantic or context-aware information by scene element detection in order to help object recognition can be found in [Tavanai et al., 2014], [Goferman et al., 2012], [Vu et al., 2015]. Long time person re-id was not also tackled in our work. This means that we should cope with clothes and other changes of the people in the scenario. This can be subject to another study in the future.

Even not considering what was not exploited in our work, there are some points that have place to be improved in CFL. First, CCF can be incorporated into the CNN training by allowing the covariance learning inside network training with Riemannian loss function. This might result in weights optimized taking into consideration the covariance matrices. In other word, network parameters are learnt along with the covariance matrices, and not before, as currently done. CFL parameters were chosen by relying in other works. However, by evaluating parameter space, one can find an optimal parameter configuration, increasing identification performance.

Finally, although our work has achieved the best result in top 1 rank against other 18 state-of-the-art works, there are open issues that can be exploited in the method itself or in a real-time application of the proposed method. Also, one can investigate a way to adapt the proposed CFL for other Computer Vision related problems.

References

Uk home office, i-lids multiple camera tracking scenario definition, 2007. `http://www.homeoffice.gov.uk/science-research/hosdb/i-lids/`.

E. Ahmed, M. Jones, and T. Marks. An improved deep learning architecture for person re-identification. *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

S. Bak, E. Corvee, F. Bremond, and M. Thonnat. Person re-identification using spatial covariance regions of human body parts. In *Advanced Video and Signal Based Surveillance (AVSS), Seventh IEEE International Conference on*, pages 435–440, 2010.

L. Bazzani, M. Cristani, and V. Murino. Symmetry-driven accumulation of local features for human characterization and re-identification. *Computer Vision and Image Understanding*, 117(2):130–144, 2013.

A. Bedagkar-Gala and S. K. Shah. A survey of approaches and trends in person re-identification. volume 32, pages 270–286. 2014.

J. Chen, Z. Zhang, and Y. Wang. Relevance metric learning for person re-identification by exploiting listwise similarities. *Image Processing, IEEE Transactions on*, 24(12):4741–4755, 2015.

D. S. Cheng, M. Cristani, M. Stoppa, L. Bazzani, and V. Murino. Custom pictorial structures for re-identification. In *Proceedings of the British Machine Vision Conference*. BMVA Press, 2011.

D. C. Cireşan, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber. Flexible, high performance convolutional neural networks for image classification. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Two*, IJCAI'11, pages 1237–1242. AAAI Press, 2011.

N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, 2005.

A. Dantcheva, C. Velardo, A. D'Angelo, and J. Dugelay. Bag of soft biometrics for person identification. *Multimedia Tools and Applications*, 51(2):739–777, 2010.

S. Ding, L. Lin, G. Wang, and H. Chao. Deep feature learning with relative distance comparison for person re-identification. *Pattern Recognition*, 48(10):2993–3003, 2015. Discriminative Feature Learning from Big Data for Visual Recognition.

C. R. S. Dultra, W. R. Schwartz, T. Souza, R. Alves, and L. Oliveira. Re-identifying people based on indexing structure and manifold appearance modeling. In *SIBGRAPI Conference on Graphics, Patterns, and Images*, 2013.

V. Eiselein, G. Sternharz, T. Senst, I. Keller, and T. Sikora. Person re-identification using region covariance in a multi-feature approach. In *Image Analysis and Recognition*, volume 8815 of *Lecture Notes in Computer Science*, pages 77–84. Springer International Publishing, 2014.

D. Fehr, A. Cherian, R. Sivalingam, S. Nickolay, V. Morellas, , and N. Papanikolopoulos. Compact covariance descriptors in 3d point clouds for object recognition. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 1793–1798, 2012.

W. Förstner and Bo. Moonen. A metric for covariance matrices. In *Geodesy-The Challenge of the 3rd Millennium*, pages 299–309. Springer Berlin Heidelberg, 2003.

A. Franco and L. Oliveira. A coarse-to-fine deep learning for person re-identification. In *Applications of Computer Vision (WACV), IEEE Winter Conference on*, 2016a.

A. Franco and L. Oliveira. Integrating coarse-to-fine deep features with covariance descriptors in person re-identification. (submitted). *Pattern recognition*, 2016b.

N. Gheissari, T.B. Sebastian, P.H. Tu, J. Rittscher, and R. Hartley. Person reidentification using spatiotemporal appearance. volume 2, pages 1528–1535, 2006.

A. Globerson and S. Roweis. Metric learning by collapsing classes. In *NIPS*, 2005.

S. Goferman, L. Zelnik-Manor, and A. Tal. Context-aware saliency detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(10):1915–1926, 2012.

D. Gray and H. Tao. *Computer Vision - ECCV: 10th European Conference on Computer Vision, Marseille, France, October 12-18, Proceedings, Part I*, chapter Viewpoint Invariant Pedestrian Recognition with an Ensemble of Localized Features, pages 262–275. 2008.

D. Gray, S. Brennan, and H. Tao. Evaluating appearance models for recognition, reacquisition, and tracking. In *IEEE International Workshop on Performance Evaluation for Tracking and Surveillance*, 2007.

G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

G. E. Hinton, S. Osindero, and Y. Teh. A fast learning algorithm for deep belief nets. *Neural Comput.*, 18(7):1527–1554, 2006.

M. Hirzer, C. Beleznai, P. M. Roth, and H. Bischof. Person re-identification by descriptive and discriminative classification. In *Proceedings of the 17th Scandinavian Conference on Image Analysis*, pages 91–102. Springer-Verlag, 2011.

M. Hirzer, P. Roth, M. Köstinger, and H. Bischof. Relaxed pairwise learned metric for person re-identification. In *Computer Vision - ECCV*, volume 7577 of *Lecture Notes in Computer Science*, pages 780–793. Springer, 2012.

D. Hubel and T. Wiesel. Receptive fields and functional architecture of monkey striate cortex. In *The Journal of Physiology 195(1)*, pages 215–243, 1968.

O. Huynh and B. Stanciulescu. Person re-identification using the silhouette shape described by a point distribution model. In *IEEE Winter Conference on Applications of Computer Vision*, pages 929–934, 2015.

Y. Bengio I. Goodfellow and A. Courville. Deep learning. Book in preparation for MIT Press, 2016.

Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *ArXiv e-prints*, 2013.

Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'95, pages 1137–1143, San Francisco, CA, USA, 1995.

A. Krizhevsky and G. Hinton. Using very deep autoencoders for content-based image retrieval. In *ESANN*, 2011.

C. Kuo, S. Khamis, and V. Shet. Person re-identification using semantic color names and rankboost. In *IEEE Winter Conference on Applications of Computer Vision*, pages 281–287, 2013.

H. Larochelle and Y. Bengio. Classification using discriminative restricted Boltzmann machines. pages 536–543, 2008.

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Comput.*, 1(4):541–551, 1989.

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

W. Li, R. Zhao, and X. Wang. Human reidentification with transferred metric learning. In *Computer Vision - ACCV*, volume 7724 of *Lecture Notes in Computer Science*, pages 31–44. Springer Berlin Heidelberg, 2013.

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

S. Liao, Y. Hu, X. Zhu, and S. Z. Li. Person re-identification by local maximal occurrence representation and metric learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

C. Liu, S. Gong, C. C. Loy, and X. Lin. Person re-identification: What features are important? In *Proceedings of the 12th International Conference on Computer Vision - Volume Part I*, pages 391–401, 2012.

P. Luo, X. Wang, and X. Tang. Hierarchical face parsing via deep learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2480–2487. IEEE Computer Society, 2012.

A.J. Ma, P.C. Yuen, and L. Jiawei. Domain transfer support vector ranking for person re-identification without target camera label information. In *IEEE International Conference on Computer Vision*, pages 3567–3574, 2013.

B. Ma, Y. Su, and F. Jurie. Bicov: a novel image representation for person re-identification and face verification. In *Proceedings of the British Machine Vision Conference*. BMVA Press, 2012a.

B. Ma, Y. Wu, and F. Sun. *Foundations of Intelligent Systems: Proceedings of the Sixth International Conference on Intelligent Systems and Knowledge Engineering, Shanghai,* *China(ISKE2011)*, chapter Affine Object Tracking Using Kernel-Based Region Covariance Descriptors, pages 613–623. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012b.

U. G. Mangaia, S. Samantaa, S. Dasa, and P. R. Chowdhuryb. A survey of decision fusion and feature fusion strategies for pattern classification. In *IETE Technical Review*, pages 293–307, 2014.

N. Martinel, C. Micheloni, and G. Foresti. Kernelized saliency-based person re-identification through multiple metric learning. *Image Processing, IEEE Transactions on*, 2015.

J. Masci, U. Meier, D. Ciresan, and J. Schmidhuber. Stacked convolutional auto-encoders for hierarchical feature extraction. In *International Conference on Artificial Neural Networks and Machine Learning*, volume 6791 of *Lecture Notes in Computer Science*, pages 52–59. Springer, 2011.

J. Ngiam, A. Khosla, M. Kim, J. Nam amd H. Lee, and A. Y. Ng. Multimodal deep learning. In *International Conference on Machine Learning*, 2011.

L. Oliveira. *Semantically Integrating Laser and Vision in Pedestrian Detection.* PhD thesis, University of Coimbra. Department of Electrical and Computer Engineering, 2010.

S. Paisitkriangkrai, C. Shen, and A. V. D. Hengel. Learning to rank in person re-identification with metric ensembles. *CoRR*, abs/1503.01543, 2015.

F. Porikli and T. Kocak. Robust license plate detection using covariance descriptor in a neural network framework. In *2006 IEEE International Conference on Video and Signal Based Surveillance*, pages 107–107, 2006.

A. Romero, M. Gouiffés, and L. Lacassagne. *Computer Vision - ACCV International Workshops, Revised Selected Papers, Part II*, chapter Covariance Descriptor Multiple Object Tracking and Re-identification with Colorspace Evaluation, pages 400–411. Springer Berlin Heidelberg, 2013.

R. Satta. Appearance descriptors for person re-identification: a comprehensive review. *CoRR*, 2013.

W. R. Schwartz and L. S. Davis. Learning discriminative appearance-based models using partial least squares. In *2009 XXII Brazilian Symposium on Computer Graphics and Image Processing*, pages 322–329, 2009.

P. Smolensky. Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1. chapter Information Processing in Dynamical Systems: Foundations of Harmony Theory, pages 194–281. Cambridge, MA, USA, 1986.

H. A. Song and S. Lee. *Neural Information Processing: 20th International Conference, ICONIP. Proceedings, Part I*, chapter Hierarchical Representation Using NMF, pages 466–473. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15: 1929–1958, 2014.

H. Tabia, H. Laga, D. Picard, and P. H. Gosselin. Covariance descriptors for 3d shape matching and retrieval. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 4185–4192, June 2014.

A. Tavanai, M. Sridhar, F. Gu, A. G. Cohn, and D. C. Hogg. Context aware detection and tracking. In *Pattern Recognition (ICPR), 2014 22nd International Conference on*, pages 2197–2202, 2014.

O. Tuzel, F. Porikli, and P. Meer. Region covariance: A fast descriptor for detection and classification. In *Computer Vision - ECCV*, volume 3952 of *Lecture Notes in Computer Science*, pages 589–600. 2006.

P. Vincent, H. Larochelle, Y. Bengio, and P. Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, pages 1096–1103, New York, NY, USA, 2008. ACM.

P. Vincent, H. Larochelle, I. Lajoie, B. Yoshua, and P. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.*, 11:3371–3408, 2010.

T. Vu, A. Osokin, and I. Laptev. Context-aware cnns for person head detection. *CoRR*, abs/1511.07917, 2015.

X. Wang, G. Doretto, T. Sebastian, J. Rittscher, and P. Tu. Shape and appearance context modeling. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8, 2007.

K. Weinberger, J. Blitzer, and L. Saul. Distance metric learning for large margin nearest neighbor classification. In *NIPS*. MIT Press, 2006.

F. Xiong, M. Gou, O. Camps, and M. Sznaier. Person re-identification using kernel-based metric learning methods. In *Computer Vision - ECCV*, pages 1–16. Springer, 2014.

L. Yang. Distance metric learning: A comprehensive survey, 2006.

D. Yi, Z. Lei, and S. Z. Li. Deep metric learning for practical person re-identification. *CoRR*, 1407.4979, 2014a.

D. Yi, Z. Lei, S. Liao, and S. Z. Li. Deep metric learning for person re-identification. In *ICPR*, 2014b.

M. Zeng, Z. Wu, C. Tian, L. Zhang, and L. Hu. Efficient person re-identification by hybrid spatiogram and covariance descriptor. In *Computer Vision and Pattern Recognition Workshops (CVPRW), IEEE Conference on*, pages 48–56, 2015.

X. Zeng, W. Ouyang, and X. Wang. Multi-stage contextual deep learning for pedestrian detection. In *IEEE International Conference on Computer Vision*, pages 121–128, 2013.

W. Zheng, S. Gong, and T. Xiang. Associating groups of people. In *Proceedings of the British Machine Vision Conference*. BMVA Press, 2009.

W. S. Zheng, S. Gong, and T. Xiang. Reidentification by relative distance comparison. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(3):653–668, 2013.

Z. Zhu, P. Luo, X. Wang, and X. Tang. Deep learning identity-preserving face space. In *IEEE International Conference on Computer Vision*, pages 113–120, 2013.