Skoltech

Skolkovo Institute of Science and Technology

Image-based human re-identification and recognition using deep learning methods

Doctoral Thesis

by

Evgeniya Ustinova

DOCTORAL PROGRAM IN COMPUTATIONAL AND DATA SCIENCE AND ENGINEERING

Supervisor
Professor Victor Lempitsky

Moscow

Abstract

This work considers two popular human recognition problems, namely, person re-identification and face recognition. For both of these tasks, similarity estimation for pairs of images is the core problem. Generally, no examples of test identities are exposed during the training stage, so the system should be able to learn the similarity concept from the training data and to generalize it onto unseen identities.

Training of siamese neural networks is at the center of this work. Provided with an appropriate objective function, they are able to learn a mapping from an input image space to a high-dimensional descriptor space in such a way that the semantic similarity of input images can be estimated using Euclidean distance or angular similarity between the corresponding embeddings (descriptors). Three important aspects of building siamese deep neural networks for person re-identification are considered: the architecture design, the objective function design, and deep domain adaptation techniques. Domain adaptation is additionally considered for face recognition.

While deep learning has approached human performance in face recognition, the results for person re-identification have been relatively modest during recent time. Two improvements related to training siamese neural networks for person re-identification are suggested in this work. First of them is a loss function for siamese training, called Histogram loss. The new loss does not introduce parameters that need to be tuned and shows favorable results across a range of datasets and problems compared to recently proposed alternatives. Second, a new architecture for person re-identification is suggested. As the task of re-identification is inherently associated with non-rigid appearance description, the suggested architecture is based on the deep bilinear convolutional network (Bilinear CNN) that has been proposed recently for fine-grained classification of highly non-rigid objects. Finally, domain adaptation is addressed for both person re-identification and a special case of face recognition related to using surveillance data at the test stage. Domain adversarial training is adopted for person re-identification and shown to improve the results in the cross-domain setting. Feature-level and more recent image-level deep adaptation techniques are evaluated and viable strategies are suggested for surveillance face recognition.

List of publications:

- Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand and Victor Lempitsky. Domain-adversarial training of neural networks. The Journal of Machine Learning Research, JMLR, 17(1), pages 2096-2030, 2016. *Personal contribution: developing the variant of the method for similarity learning, designing and conducting the experiments for person re-identification, text writing (partially)*

- Evgeniya Ustinova and Victor Lempitsky. Learning deep embeddings with histogram loss. Advances in Neural Information Processing Systems, NIPS, 2016. *Personal contribution: existing approaches analysis, method development, designing and conducting the experiments, text writing (partially).*

- Evgeniya Ustinova, Yaroslav Ganin and Victor Lempitsky, Multi-region bilinear convolutional neural networks for person re-identification. IEEE International Conference on Advanced Video Signal-based Surveillance, AVSS, 2017. *Personal contribution: existing approaches analysis, method development, designing and conducting the experiments, text writing (partially).*

Contents

List of abbreviations

CMC curve Cumulative Matching Characteristic curve

CNN Convolutional Neural Network

CycleGAN Cycle-Consistent Generative Adversarial Network

DML Deep Metric Learning

LSSS loss Lifted Structured Softmax Similarity loss

PCA Principal Component Analysis

ReLU Rectified Linear Unit

SGD Stochastic Gradient Descent

t-SNE t-distributed Stochastic Neighbor Embedding

Chapter 1

Introduction

1.1 Context

Today, large-scale distributed multi-camera surveillance systems are installed in many kinds of public places. While such systems are usually provided with an infrastructure to capture, store and access video data, up to this time, the work of watching and searching through such data has been mostly assigned to human operators. The growing amount of surveillance data makes its manual handling non-scalable. The situation has started to change recently with the development and adoption of methods for visual recognition. In particular, recognition of humans (either based on an image of a face or full body) is one of very demanded directions in the surveillance field. An ability to automatically retrieve and compare human images is central for a wide range of applications from long-term multi-camera tracking, forensic analysis and security to retail, and advertising.

1.1.1 Person re-identification

Person re-identification refers to visually matching pedestrian images captured by multiple cameras with (possibly) non-overlapping views. In more detail, it is usually formulated as the task of matching the query (or probe) image to a set of images, called *gallery*.

For this task, clothing is considered to be the primary source of visual information, whereas faces are assumed to be indistinguishable due to low resolution, or not visible due to individual body positions.

The main confounding factors of this task stem from the notoriously high variation of the appearance of the same person (even at short time spans) due to differences in pose,

camera viewpoint, illumination, and background clutter presence. Examples of typical person re-identification data are shown in Figure 1.1.

Most often, person re-identification is formulated as follows:

Problem 1. *Re-identification*

Input:	Query image q, gallery images $G = \{g_i\}_{i=1}^{n_G}$
Task:	Find such an image $g_j \in G$ from the gallery that g_j depicts the same identity as q: $l_q = l_{g_j}$

There are two main scenarios for the task [Zheng et al., 2016]:

- in the *closed-set* scenario, the gallery has to contain at least one image for every query identity,
- the *open-set* scenario is more challenging and admits situations when the query identity is not present in the gallery.

The latter implies that the system should be able to decide whether the query identity is contained in the gallery set at all. By the time of publishing the results presented in this work, person re-identification performance was rather low even for the closed-set protocols, so much research effort has been first dedicated to improving the results in this setting. Consequently, most of the re-identification benchmark datasets are developed and used for evaluation within the closed-set framework. This work also falls under this category and use the closed-set scenario for evaluation.

In most of person re-identification datasets, the training images are usually captured during limited time within middle-size camera networks. This is due to a large amount of manual effort required to analyze such data to find occurrences of the same person across different camera views. Therefore, commonly, each identity has a relatively small amount of different images (in the order of 10).

1.1.2 Face recognition

Face recognition also refers to the task of matching human images. But in contrast to person re-identification, faces are matched instead of full-body images. There are also two main scenarios for face recognition [Jafri and Arabnia, 2009]:

- *face identification* describes one-to-many matching, when an image of an unknown individual is compared to a database of images of known individuals in order to

(a) (b)

FIGURE 1.1: Sample images of publicly available person re-identification datasets (a) - CUHK03 dataset [Li et al., 2014], (b) - Market-1501 dataset [Zheng et al., 2015b]. In each group, each row corresponds to a person, different camera views are shown for each identity. The data demonstrate variations in pose, illumination, background and the presence of clutter

.

determine this person's identity. For example, checking the person's access level may require searching for this person in the employee database.

- *face verification* describes one-to-one matching when, given a pair of face images, one should determine whether they depict the same identity. For example, this case can be useful to ensure that the person is the individual he/she claims to be.

The formulation for face identification is similar to Problem 1, whereas face verification is:

Problem 2. *Verification*

Input:	A pair of images (q_1, q_2)
Task:	Determine whether the images q_1 and q_2 depict the same person or two different persons: $l_{q_1} ? = l_{q_2}$

Similarly to person re-identification, the difficulties of the task come from the uncontrolled environment, *e.g.*, variations in pose, lighting, different facial expressions and

hairstyle. However, unlike person re-identification, recent face recognition works have been more focused on higher quality images that are often taken by professional photographers, rather than surveillance systems. Mainly, this is due to the fact that face recognition is not limited to only surveillance-based applications. Moreover, it is a challenging task even in the case of sufficient image quality: many methods are aimed at extracting important facial information, while ignoring irrelevant variations (*e.g.*, [Chopra et al., 2005, Parkhi et al., 2015, Schroff et al., 2015]). Another circumstance can be mentioned: it is now possible to use multiple publicly available Internet resources to extract images of well-known persons (*e.g.*, actors, politicians) using their names as queries to a search engine. Therefore large training datasets for face recognition can be harvested from the Internet, where lots of face images with known identities of people in them can be mined in an automatic or semi-automatic way [Parkhi et al., 2015]. When trained on such datasets, deep face recognition networks are rivaling and exceeding human face recognition performance, when applied to similar types of images.

Yet, many, perhaps majority, of practical scenarios require building vision systems that recognize faces in images that are quite different from those harvested from the Internet. These scenarios include surveillance for security purposes, smart home applications, collaborative robotics, which all come with specific camera parameters, camera positioning, lighting conditions that are all quite different and are often of much lower quality than face images typically encountered on the Internet.

1.1.3 Relation to image retrieval and classification

For both human recognition problems, the training data most often consist of some number of training images along with their identities. This makes the tasks quite similar to classification, given that each identity corresponds to one class. Although in practice, such recognition and re-identification systems, once trained, are required to work for a large number of unseen identities. Therefore in most of the evaluation protocols, test and train data contain non-overlapping sets of identities.

The discussed property of the tasks is also inherent to general image retrieval. The latter also requires the recognition system to be able to estimate the similarity between images of probably unseen classes and, therefore, to capture the notion of similarity from the training data and generalize it.

1.1.4 The key problem

For both of the described human recognition problems, the recognition step follows several other preprocessing steps: face/full-body detection and, possibly, alignment (although some works include the alignment step into recognition, *e.g.*, [Tadmor et al., 2016]). This work is focused on the recognition step.

While various scenarios for person re-identification and face recognition exist, at a high level, the key task for both problems can be informally defined as follows:

Problem 3. *Similarity estimation*

Input: A pair of images (q_1, q_2)

Task: to measure the similarity value for (q_1, q_2) in such a way that a pair would get:

- high similarity score in case of depicting the same person (such pairs are also called *positive* pairs),
- low score in case of depicting different persons (*negative* pairs).

This requires to construct a robust representation and an appropriate similarity measure which would facilitate assigning corresponding similarity scores.

1.1.5 General approaches to human recognition

Traditionally, features/representation design and fixed distance/similarity function design have been considered as two separate steps in building the recognition system. Consequently, most of the research efforts on face recognition and person re-identification have been dedicated to these two problems. Some works focused on finding a discriminative representation [Ma et al., 2012a,b, Bazzani et al., 2013]. Others sought for a special transformation of a fixed feature space in order to improve the retrieval quality [Hirzer et al., 2012, Mignon and Jurie, 2012], or the quality of metric-based algorithms, like nearest neighbor classification [Weinberger and Saul, 2009]. Most of the works under this category formulate the task as Mahalanobis metric learning. The idea behind it is to learn a linear transformation to a new feature space where Euclidean distance can be used in order to solve Problem 3. The non-linear case is achieved by using the 'kernel trick'. Some approaches aimed at both feature design and metric learning [Liao et al., 2015].

Unlike the mentioned approaches, deep neural networks allow to learn a hierarchy of representations with different levels of abstraction. Starting from the most low-level representations that can be extracted using the first layers of the network, the task-specific information becomes more and more distilled while forwarding through a trained deep neural network. Thus deep neural networks are able to combine the two steps of feature extraction and learning a prediction model based on such features, allowing to automatically find more suitable feature extractors for a particular task. For example, in human recognition, the deep learning approach may help to take into account important clothing parts or facial features and at the same time discard the illumination and pose variations to some extent.

Deep learning methods have improved state of the art in multiple visual recognition tasks, including face recognition [Taigman et al., 2014, Parkhi et al., 2015, Schroff et al., 2015]. The situation has been rather different for person re-identification: until recently, the results of some traditional methods [Paisitkriangkrai et al., 2015] were still better than those of the first deep learning approaches to person re-identification [Li et al., 2014].

Therefore, one goal of this work, among others, is to consider deep learning methods for the person re-identification task and try to improve over the non-deep methods.

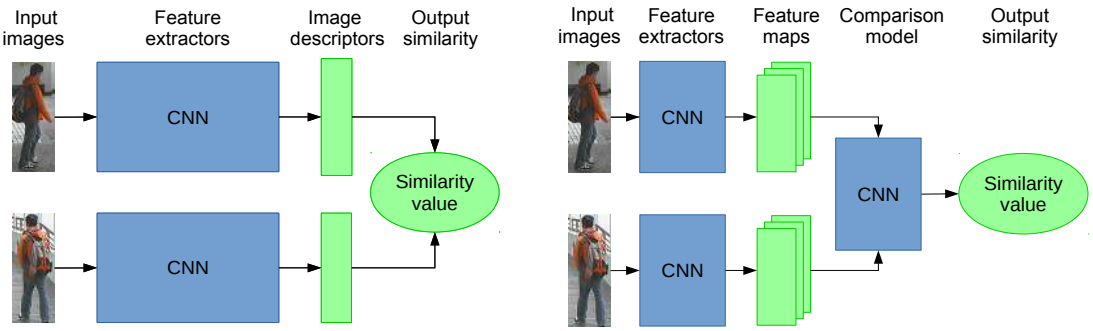
1.2 Objectives and motivation

The goal of this work is to explore building deep learning systems for human recognition.

In particular, the development of perhaps every deep recognition system includes several important tasks:

- the architecture design,
- the objective function design,
- the choice of training data.

To a certain extent, each of these tasks constitutes a challenge in the context of human recognition. The presence of a large domain shift between the training and test data (*e.g.*, when using different sets of cameras) is also a serious problem for both tasks of person re-identification and face recognition as it may cause a considerable performance drop.



(a) (b)

FIGURE 1.2: (a) - A general siamese architecture [Bromley et al., 1993, Chopra et al., 2005] scheme: two images are first mapped into a descriptor space, the similarity value is computed as a function of a pair of output descriptors; most often, negative Euclidean distance or cosine function is used as a simple similarity measure. (b) - A modification of a siamese architecture, where the similarity value is computed using an additional CNN (convolutional neural network), so the difference of images is more explicitly modeled than in the original siamese architecture.

In the reminder of this section, each of the three aspects is discussed to build a foundation for the ideas that are presented in the subsequent chapters: Chapter 3 addresses the objective function design for similarity learning, Chapter 4 considers the architecture design for person re-identification, Chapter 5 and Chapter 6 address domain adaptation for person re-identification and surveillance face recognition.

1.2.1 Approaches to building CNNs for human recognition

Deep learning allows to learn feature extractors so that their parameters are optimized for a particular task. *Siamese* architecture, that was initially suggested for signature verification by [Bromley et al., 1993], is a standard deep learning approach to retrieval problems. It typically consists of two identical subnetworks that parametrize a mapping from an image space to a descriptor space. The siamese architecture is usually trained in such a way that Euclidean or angular distance between the descriptors reflects the similarity and dissimilarity of corresponding images. For example, [Yi et al., 2014a, Parkhi et al., 2015] utilize the siamese architecture for person re-identification and face verification correspondingly.

At the same time, some other methods modify the approach to explicitly model the differences of image regions at multiple feature levels [Ahmed et al., 2015, Li et al., 2014, Chen et al., 2016]. Such architectures map a pair of images directly to a scalar similarity score and therefore a complex pairwise similarity function is learned instead of per-image embeddings. However, such approaches require multiple pairwise forward

operations at test time: a query image should be compared to each of the gallery images by computing the similarity score with a trained neural network. Generally, this is much slower than computing pairwise Euclidean/angular distances for a number of pre-computed descriptors. The described variants are shown in Figure 1.2.

Therefore this work builds on top of more standard siamese approaches in order to learn a mapping to a descriptor space so that simple similarity measures can be applied to the resulting descriptors.

However, the choice of the convolutional architecture for embedding in the case of person re-identification is far from obvious. In particular, existing "standard" architectures that combine convolutional layers followed by fully-connected layers can fail to achieve sufficient invariance to strong viewpoint changes as well as to non-rigid articulations of pedestrians, given the limited amount of training data typical for re-identification tasks and datasets.

Chapter 4 describes a person re-identification architecture that is based on the idea of bilinear convolutional networks (Bilinear CNNs) [Tsung-Yu Lin and Maji, 2015] that were originally presented for fine-grained classification tasks and later evaluated for face recognition [RoyChowdhury et al., 2015]. At the same time, the suggested architecture is built on top of the siamese neural network earlier suggested by Yi et al. [2014a] for person re-identification. It is shown to perform favorably for three publicly available person re-identification datasets, for two largest of them, state-of-the-art results were achieved (on the moment of publication).

1.2.2 Objective functions for training siamese architectures

Section 1.2.1 describes the principle of siamese neural networks. The goal of training such architectures is to build a mapping from an image space to a high-dimensional descriptor space. Such descriptors are also called *embeddings*. If similar images are mapped into points that are close in the descriptor space, and non-similar - to points that are far away from each other, the image similarity can be estimated by the distance in the descriptor space. In order to learn such mapping, a certain objective should be defined. It should encourage the proximity of embeddings of similar images and discourage it for dissimilar images.

Learning deep feed-forward embeddings still poses a challenge. While it is possible to write down a loss based on tuples of training points expressing the above-mentioned objective, optimizing such a loss rarely works "out of the box" for complex data. This is evidenced by the broad variety of losses, which can be based on pairs, triplets or

FIGURE 1.3: Matching and non-matching pairs of probe-gallery images from different person re-identification datasets. The three datasets are treated as different domains in our experiments.

quadruplets of points. Most of the existing losses come with a certain number of tunable parameters, and the quality of the final embedding is often sensitive to them.

A new objective function for training deep embeddings is suggested in Chapter 3. It is based on pairwise training data and does not incorporate usual hyper-parameters that require careful tuning for different datasets. This loss is computed in two steps: first, the two similarity distributions for positive and negative pairs are estimated, second, their overlap is computed. The goal is to minimize such overlap, so that positive pairs have higher similarity than negative pairs. The experimental section of Chapter 3 shows that the suggested loss outperforms several other point-based losses for person re-identification and also performs well for several other retrieval problems.

1.2.3 Different domains in human recognition

Cross-domain human recognition remains a hard problem for many practical scenarios. For person re-identification, the common case considered in literature is training and testing on the data obtained from different sets of cameras. Given the fact that different re-identification benchmark datasets were collected separately and independently, they are very suitable for modeling such use case. Indeed, pedestrian images differ considerably between some publicly available datasets. Such difference may come from the different illumination conditions, background, resolution and position of cameras. The examples of three different person re-identification datasets are shown in Figure 1.3.

Chapter 5 demonstrates that the performance of cross-domain person re-identification can be improved by domain-adversarial training. This method is initially developed for cross-domain classification problems and is also described in Chapter 5. Eight domain pairs are used for evaluation, where each of the domains corresponds to one of tree popular re-identification datasets.

An important cross-domain case of person re-identification is considered in Chapter 6. As it was mentioned in Section 1.1.2, the large-scale training data for face recognition are

usually harvested from the Internet and are characterized by much higher quality then surveillance data. In Chapter 6, it is demonstrated how surveillance face recognition can be approached using recent image-to-image translation techniques. The domain-adversarial training method introduced in Chapter 5 is also included in the evaluation.

1.3 Datasets

In this section, we describe several publicly available person re-identification and face verification datasets that are used in this work. Person re-identification datasets are used for training and evaluation. Face verification datasets are mostly used as training data for surveillance face verification. The specially collected surveillance data are used for evaluation and described in Chapter 6.

- PRID [Hirzer et al., 2011] contains images of 385 persons viewed from camera A and images of 749 persons viewed from camera B, 200 persons appear in both cameras. 100 persons appearing in both camera views are used for training. The images of the other 100 persons from camera A are used as probe, all images from camera B excluding those used in training (649 in total) are used as gallery at test time.
- VIPeR [Gray et al., 2007] also contains images taken with two cameras, and in total 632 persons are captured, for every person, there is one image for each of the two camera views. Images of 316 identities are used for training and all others for testing.
- CUHK01 [Li et al., 2012] contains images of 971 identities from two disjoint camera views. Each identity has two samples per camera view. 485 identities are randomly chosen for training and the other 486 for test.
- CUHK02 [Li and Wang, 2013] consists of images from five pairs of cameras, two images for each person from each of the two cameras. It contains images for 1816 identities in total, but most often, 971 from the first pair of cameras are used.
- CUHK03 [Li et al., 2014] includes $13,164$ images of $1,360$ pedestrians captured from 3 pairs of cameras. The two versions of the dataset are provided: *CUHK03-labeled* and *CUHK03-detected* with manually labeled bounding boxes and automatically detected ones accordingly. $1,360$ identities are split into $1,160$ identities for training, 100 for validation and 100 for testing.
- Market-1501 [Zheng et al., 2015b] contains $32,643$ images of 1,501 identities, each identity is captured by from two to six cameras. The dataset is randomly divided

into the test set of 750 identities and the train set of 751 identities. For each identity in the test set, one image in each camera is selected and used as a query. Manually drawn bounding boxes are used for query images, and automatically detected ones for training images as well as for the gallery images in the test set. Market-1501 also includes "distractor" images that correspond to false detections, which makes evaluation on the dataset even more realistic and challenging. Matching is done across different cameras, and in the situation when the gallery images of the same person and form the same camera as the query are ignored and not counted as true positives during the evaluation.

- YouTube Faces (YTF) [Wolf et al., 2011] consists of $3,425$ videos of $1,595$ people collected from YouTube, with an average of 2 videos per identity.
- VGG Face [Parkhi et al., 2015] contains $2,6$M images of $2,622$ identities harvested from the Internet.

Following [Li et al., 2014], we use Recall@K metric to report our results for person re-identification. Recall@K (also referred as Cumulative Matching Characteristic curve) is computed for different values of K as a percentage of the queries for which the correct match is found among the first K results, where the results are sorted by similarity to the query.

1.4 Architectures

- Deep Metric Learning (DML) architecture has been suggested in [Yi et al., 2014a] for person re-identification. Is is used in Chapter 3, Chapter 4 and Chapter 5 as a base architecture. The network incorporates independent streams, in which three overlapping parts of person images are processed separately (top, middle and bottom parts), and produces 500-dimensional descriptor as an output.

  Each of the three streams incorporates two convolutional layers of size 7×7 and 5×5, followed by the rectified linear (ReLU) non-linearity and max-pooling with the kernel size of two pixels and the stride of two pixels. 500-dimensional descriptor vector is produced as an output of a fully-connected layer, that accepts the outputs of the three convolutional streams. For each training pair of images, the cosine similarity of these descriptors is calculated.

  As in [Yi et al., 2014b], we consider the *view-invariant* architecture with the same weights shared for all views in the dataset. This case is more general as the

trained networks can be used for new out-of-sample cameras, unlike the view-specific case that lacks weights sharing (which makes networks to capture view-specific appearance peculiarities).

- VGG-face [Parkhi et al., 2015] is a powerful architecture for face recognition. It is used in Chapter 6. It consists of 5 blocks of convolution layers. Each block is followed by max-pooling. Three fully-connected layers are inserted at the end, their output dimensions are: 4096, 4096 and 2622. The layer before last is used to extract deep embeddings.

1.5 Contribution

This work presents the following contributions:

- The work presents a new loss function for training similarity-based siamese neural networks. Such training is performed to build a mapping from an image to a descriptor space so that semantically similar images are close to each other and non-similar images are distant in this descriptor space. Compared to the existing (by the time of publication) loss functions ((2.8)), the suggested loss does not incorporate special hyper-parameters, like thresholds for separating the pairs of matching images from the pairs of non-matching images. At the same time, it is based on the pairwise training and therefore does not require more complex forms of training data, like triplets or quadruplets. The experiments are performed for the two largest person re-identification datasets (the suggested loss function performed best among several others: (2.8), (2.11), (**??**)) and for two datasets for other retrieval problems (second-best result). This contribution is the subject of Chapter 3 and published as [Ustinova and Lempitsky, 2016].

- A novel architecture for person re-identification is suggested in Chapter 4. This chapter also offers an analogy between person re-identification and fine-grained recognition problems. This analogy allows to build upon the Bilinear CNN model of [Tsung-Yu Lin and Maji, 2015] and adapt it to the task: the suggested approach can be considered as a middle ground between Bilinear CNNs and regular models. The suggested architecture is shown to outperform several baselines on three popular person re-identification benchmarks. For the two largest datasets, the state-of-the-art results were achieved (on the moment of publication). The results of this part are published as [Ustinova et al., 2017].

- The deep feature-level domain adaptation model [Ganin et al., 2016] is demonstrated to be applicable to person re-identification. The modification of domain-adversarial learning, also described in Chapter 5, lies in replacing the task-specific label predictor by a siamese subnetwork that maps pedestrian images into a descriptor space. This contribution is presented in Chapter 5 and published as a part of the work [Ganin et al., 2016]. The experiments are conducted for eight domain pairs. In these domain pairs, the source and target domains are represented by two (of three in total) different publicly available person re-identification datasets. Thus, the practical situation of the domain shift between different sets of cameras is considered.

- Face recognition in the presence of a strong domain shift is considered in Chapter 6. Pixel-level domain adaptation approach based on the CycleGAN model is shown to improve the results of face recognition for surveillance images affected by complex degradation factors. The comparison is performed for the data harvested from Moscow subway and includes several baselines, namely, the feature-level domain adaptation method described in Chapter 5 and also a reverse translation from the target surveillance domain to the source higher-quality domain. Based on the demonstrated comparison and evaluation, a viable strategy is suggested for training face recognition for surveillance data.

Chapter 3, Chapter 4 and Chapter 5 use person re-identification architecture of [Yi et al., 2014a] as a baseline method (it is also described in Section 1.4). Chapter 4 is based on the results of Chapter 3: the loss function introduced in Chapter 3 is used for all the experiments in Chapter 4 as it was demonstrated to show the best performance for person re-identification. The results of Chapter 5 were chronologically the earliest among all the results presented in this work, therefore methods from Chapter 3 and Chapter 4 were not used there. Although the contributions of each of the chapters are independent, they are all parts of building a person re-identification pipeline and can be applied simultaneously. Chapter 6 considers domain adaptation for surveillance face recognition and uses the method from Chapter 5 as one of the baselines.

Chapter 2

Related work

2.1 Metric learning

There is a wide range of practical tasks where the estimation of distances or similarities between data points is the central problem. These, for example, include face recognition, person re-identification, document ranking, and also improving the effectiveness of metric-dependent methods, *e.g.*, k-nearest neighbors and k-means clustering.

Generally, the main goal of building a distance learning framework is to be able to assess the proximity of pairs of objects (here, images) given their feature representations. In more detail, the metric learning method should be able to perform the linear or non-linear mapping of an input space into a new space where the distance between the similar objects should be small, and the distance between the non-similar objects should be substantial. The particular semantics of similarity relation depends on a particular task.

[Bellet et al., 2013] name three main classes of metric learning tasks in accordance with the available level of supervision:

- supervised, where the class labels are given for each object, so the objects are considered similar if they belong to the same class [Globerson and Roweis, 2006, Weinberger et al., 2006, Weinberger and Saul, 2009],
- weakly-supervised, where the similarity labels are given for the pairs of objects [Hirzer et al., 2012, Köstinger et al., 2012, Liao et al., 2015]; it should also be mentioned that for some tasks, the information is given in the form of triplet rankings [Schultz and Joachims, 2004b],

- semi-supervised, where the labels or side information is given for a small number of examples [Xing et al., 2003, Davis et al., 2007, Mignon and Jurie, 2012].

The supervised setting can be considered as having the full side information because in this case, we can generate pairwise labels for every pair in the training set. Additionally, some related works also include more specific scenarios, for example, the case where each object may have several labels, and as a consequence, a pair of objects may have both similar and non-similar labels [Gouk et al., 2016]. Generally, the supervision level is defined by the particular problem, *e.g.*, clustering with the side information implies only knowing which pairs of points should be assigned to the same cluster, for k-NN classification, the full supervision should be available.

In terms of the classification mentioned above, training protocols for face recognition and person re-identification tasks typically imply either supervised setting (*i.e.*, class labels that correspond to the identities are given) or semi-supervised (*i.e.*, pairwise labels are given for matching and non-matching pairs of images). The available supervision is most often used to generate the training examples (constraints) of one of the following types:

- pairs [Davis et al., 2007, Mignon and Jurie, 2012, Hirzer et al., 2012, Köstinger et al., 2012, Liao et al., 2015]),
- triplets ([Zheng et al., 2013],
- or even quadruplets [Law et al., 2013].

The methods for metric learning that use more complex types of constraints are not only applicable to classification or verification tasks but may also be used for broader applications that require much richer annotations and therefore cannot be approached using only pairwise constraints. For example, tripletwise constraints allow for ranking problems formulations, whereas quadruplet-based methods can be applied to either ranking problems or hierarchical metric learning. The latter aims at learning such metric that images from sibling classes with respect to the class semantic hierarchy are more similar than images from more distant classes ([Law et al., 2013]). In other words, more flexible types of constraints allow for learning more complex data relations.

One important class of distance metric learning algorithms use Mahalanobis distance as it is a generalization of the Euclidean distance that allows for linear feature transformation. Such methods aim at solving an optimization problem with respect to the corresponding positive-semidefinite matrix $\mathbf{M}$. The goal of such optimization is to pull the similarly labeled points closer and to push further from each other those points that

are labeled differently. One of the difficulties of such methods lies in imposing the semi-positive definite matrix for the Mahalanobis distance. To handle this problem, some works suggest iterative schemes including a projection step where the current solution is projected onto the positive semi-definite cone [Globerson and Roweis, 2006, Xing et al., 2003, Weinberger et al., 2006] or more efficient eigenvalue optimization techniques [Ying and Li, 2012]. Some other methods reformulate the task that allows for more efficient solving: [Schultz and Joachims, 2004b] constraint the matrix to be diagonal and thus solve the quadratic program instead of semi-definite one. [Hirzer et al., 2012] formulate an optimization problem with equality constraints which allows for closed-form solution.

Non-linear transforms of the initial feature-space can be achieved by kernelization, and some of the Mahalanobis distance learning works [Davis et al., 2007, Globerson and Roweis, 2006, Weinberger et al., 2006, Schultz and Joachims, 2004b] suggest the kernelized versions along with the linear ones.

Mahalanobis distance learning

For the two data points x_i and x_j, the Mahalanobis distance is defined as

$$D_M(x_i, x_j) = \sqrt{(x_i - x_j)^\mathsf{T}\mathbf{M}(x_i - x_j)}, \tag{2.1}$$

where $x_i, x_j \in R^d$, $\mathbf{M} \in R^{d\times d}$ is a symmetric positive-semidefinite matrix, which ensures that D_M satisfies the pseudo-distance properties. Originally [Mahalanobis, 1936] the term referred to a case when the $\mathbf{M}$ matrix was equal to the inverse of the sample covariance matrix: $\mathbf{M} = \sum^{-1}$, but later in metric learning literature, it acquired the broader meaning. The goal of the Mahalanobis metric learning methods is to learn the matrix $\mathbf{M}$. Additionally, Mahalanobis distance defined by symmetric positive-semidefinite matrix $\mathbf{M} = \mathbf{L}^\mathsf{T}\mathbf{L}$ can be considered as Euclidean distance after the linear projection $\mathbf{L}$ of the original inputs:

$$\begin{aligned} D_M(x_i, x_j) &= \sqrt{(x_i - x_j)^\mathsf{T}\mathbf{L}^\mathsf{T}\mathbf{L}(x_i - x_j)} \\ &= \sqrt{(\mathbf{L}(x_i - x_j))^\mathsf{T}\mathbf{L}(x_i - x_j)} \end{aligned} \tag{2.2}$$

which corresponds to simple Euclidean distance in the case of identity matrix $\mathbf{L}$.

[Xing et al., 2003] are known to suggest the pioneering approach in the metric learning field. The authors aimed at learning Mahalanobis distance to improve the results of clustering with side information. The supervision was given in the form of pairwise constraints defining which data points should be assigned to the same cluster and which of them should be assigned to different clusters. [Xing et al., 2003] formulate the convex

optimization task for minimizing the sum of the distances for positive pairs and simultaneously keeping the sum of the distances between negative pairs above the predefined margin.

[Schultz and Joachims, 2004b] consider using relative constraints in the form of triplets to formulate the optimization task for Mahalanobis distance 2.1. The authors note that this type of constraints is more suited for the task of ranging documents than pairwise constraints. The learned distance was shown to reduce the percentage of the violated constraints on the test set if compared to non-learned Euclidean distance. Unlike [Xing et al., 2003], the authors avoid semi-definite programming by decomposing the $\mathbf{M}$ matrix as $\mathbf{M} = \mathbf{AWA}^\top$ where $\mathbf{W}$ is diagonal and $\mathbf{A}$ is fixed, which allowed for quadratic program formulation. This implies just learning the re-scaling of the features defined by A instead of an arbitrary linear transform.

[Weinberger et al., 2006] approach k-NN classification by introducing local constraints that are more suitable for multimodal class distributions. Like in [Schultz and Joachims, 2004b], the constraints are in the triplet form, but as opposed to the previous works, they are only used locally. In more details, at the training stage, instead of using all the matching pairs for building the constraints, the authors only use the pairs within some predefined neighborhoods of the initial space. In order to find such neighborhoods, for each training datapoint, the set of its *target* neighbors is defined using Euclidean distance. By using the above-mentioned subset of constraints, the authors ensure to preserve the initial neighborhood structure. To ensure good k-NN classification quality, the large margin approach is used: the task is formulated as minimizing the distance for similarly labeled points with triplet-based constraints. The suggested formulation is positive-semidefinite and an Alternating projection algorithm is used for optimization. The bottlenecks of the approach are the lack of regularisation and dependence on the performance of Euclidean distance.

Similar to [Weinberger et al., 2006], [Globerson and Roweis, 2006] solve nearest neighbours classification task. Unlike the aforementioned approaches, the authors do not use explicit pairwise or triplet constraints. Instead, for each point x_i, the authors consider a conditional distribution over all other points x_j depending in the distance $D_M(x_i, x_j)$. The approach aims to fit all such distributions to the ideal 'bi-level' distribution using Kullback-Leibler divergence as minimization objective. This implies pulling all the similarly labeled points into one and pushing all the points with different labels infinitely far from each other. Projected gradient approach is used for the convex optimization similar to [Xing et al., 2003].

Several important works approaching person re-identification and face recognition by metric learning should also be mentioned.

[Köstinger et al., 2012] suggest to use the special form of the $\mathbf{M}$ matrix that can be acquired in a parameter-free way without the need for optimization. Assuming the zero-mean Gaussian distribution of the difference vectors, the suggested form of $\mathbf{M}$ corresponds to the log-likelihood ratio test for the two hypotheses for the pair of points of being similar or non-similar. The method has been shown to be effective for various tasks, including face recognition and person re-identification [Köstinger et al., 2012, Roth et al., 2014] outperforming other more computationally consuming methods.

[Liao et al., 2015] extends the approach of [Köstinger et al., 2012] by additionally learning the data projection that defines a discriminant subspace. The learning is based on the fact that the distributions of the differences for positive and negative pairs have different variances (while the means are equal to zero by the earlier assumption). Therefore the idea is to learn such a projection so these variances differ as much as possible. The resulting projection is ensured to be cross-view by the choice of each pair in the training data: the two images come from the different camera viewpoints. It should also be noted, that this method performs dimensionality reduction by the choice of the dimensionality of the aforementioned subspace.

[Hirzer et al., 2012] only consider the negative pairs that contain the element invading the perimeters of some positive pair in the initial Euclidean space. Unlike [Weinberger et al., 2006], such impostors are predefined based on the pairwise Euclidean distances and need not be re-evaluated during training. The optimization objective is to minimize the sum of the squared distances for positive pairs after the linear projection $\mathbf{L}$ (2.2) and to maximize the sum of those for negative pairs. The $\mathbf{L}$ matrix is constrained to be orthogonal. The resulting optimization problem has a closed-form solution imposing efficient training.

[Mignon and Jurie, 2012] consider distance metric learning for the tasks of person re-identification and face recognition. Differently to the above-mentioned works, [Mignon and Jurie, 2012] utilize the decomposition 2.2 to formulate the non-convex optimization problem in terms of the matrix $\mathbf{L}$. This makes it possible for simultaneous dimensionality reduction by setting $\mathbf{L}$ rectangular. Another advantageous implication of optimizing $\mathbf{L}$ instead of $\mathbf{M}$ is a decrease in the number of learned parameters, which in turn allows for using higher data dimensionalities. The authors use the generalized logistic function as an objective. Only the pairwise relations are used in the formulation, which allows for sparse training data: pairwise constraints instead of full class labeling that is useful for triplet generation. Additionally, the used formulation allows for kernelization by reparametrization of the matrix $\mathbf{L}$: $\mathbf{L} = \mathbf{A}\mathbf{X}^\mathsf{T}$, where $\mathbf{X}$ is a matrix whose columns are the training datapoints.

[Law et al., 2013] limits the learned matrix to be diagonal. As opposed to the aforementioned methods, the authors use quadruplet-based constraints to formulate a convex optimization objective. Given the rich expressive properties of the quadruplets (*e.g.*, in comparison to pairwise constraints), the authors demonstrate that the method is applicable to a wide range of tasks, including relative attributes recognition and hierarchical metric learning.

Although many of the mentioned methods were initially introduced for other tasks, most of them were compared in the context of person re-identification in [Roth et al., 2014].

Learning non-linear metrics

[Kedem et al., 2012] use an ensemble of multivariate regression trees to build a mapping to a new feature space. The authors suggest to optimize the hinge loss-based objective similar to [Weinberger et al., 2006] using gradient boosting. Dimensionality reduction is naturally performed by training the regression trees with the low dimensional output.

[Weinberger and Saul, 2009] improves over the earlier work [Weinberger et al., 2006] by learning multiple local linear metrics. The training data are first partitioned into a number of disjoint clusters using either clustering or label information. The Mahalanobis distance metric is then learned separately for each of the partitions. All the metrics are learned simultaneously when solving the sole optimization problem.

[Wang et al., 2012] address the overfitting inherent to learning multiple local metrics [Weinberger and Saul, 2009] by enforcing smooth changing of the metric matrix over the data manifold. This is done by learning a local metric at each point as a linear combination of the local metrics defined by a number of predefined anchor points, the latter can be chosen as clustering centroids.

2.2 Learning to rank

Another closely related line of work is dedicated to the *learning to rank* task. The topic has been initially developed in the field of information retrieval with the search engines being its main application. It is more general then metric learning, as there are no constraints (like the particular type of distance) defining the way of computing the relevance value. Additionally, the task formulation does not only consider the binary relations between the search output items and the query ('relevant' or 'not relevant') but also implies that some items can be more relevant to the query than others. The similar type of relations is also widely used in metric learning literature [Weinberger and

Saul, 2009] for formulating the triplet-based constraints for optimization tasks, but they are most often generated from the less general pairwise of class labeling.

More formally, for the given query q and a set of items $D = \{x_1, ..., x_m\}$, the ranking system should output a ranking $r^*(q)$ that orders the items in correspondence to their relevance to the query: $x_i <_{r^*(q)} x_j \Leftrightarrow (x_i, x_j) \in r^*(q)$, where $r^*(q)$ defines weak ordering. The goal of the ranking algorithm is to estimate the ordering $r^*(q)$ by ordering $r_{f(q)}$ defined by some retrieval function $f(q)$.

[Joachims, 2002] considers the family of linear ranking functions to estimate $r^*(q)$:

$$(x_i, x_j) \in f_w(q) \Leftrightarrow w^\mathsf{T}\Phi(q, x_i) - w^\mathsf{T}\Phi(q, x_j) > 0, \tag{2.3}$$

where w is a learned weight vector, $\Phi(q, x)$ is a mapping onto features that describe the match between query q and document x (*e.g.*, the number of shared words). The work also gives a strategy for automatically acquiring the training data in the form of triplets (q, x_i, x_j) from clickthrough data that is usually available in the search engine logs by default. These training data are then used to formulate the corresponding soft constraints. Finally, the SVM [Cortes and Vapnik, 1995] optimization problem, called RankSVM, is formulated for the ranking task. The above-mentioned approach to formulating the relative constraints was then adopted in the metric learning context [Schultz and Joachims, 2004a].

The same approach was later adopted for person re-identification in [Prosser et al., 2010]. The task of re-identification was reformulated as the ranking problem where for each query all the matching gallery images are considered relevant and therefore should be ranked higher then the non-matched images that are considered irrelevant. The mapping $\Phi(q, x)$ in 2.3 is defined as the absolute difference of the arguments:

$$\Phi(q, x) = |q - x| = (|q_1 - x_1|, ..., |q_d - x_d|),$$

where d is the feature dimensionality. Further, to reduce the memory consumption, [Prosser et al., 2010] suggest the learning of RankSVM ensemble where each of the weak rankers is learned using the overlapping subsets of learning data.

[Kuo et al., 2013] suggest another ranking approach based on the RankBoost algorithm [Freund et al., 2003] for the person re-identification problem. The ranking function is based on the combination of the scores associated with different local features:

$$\begin{aligned} (x_i, x_j) \in f_\alpha(q) \Leftrightarrow H(q, x_i) - H(q, x_j) > 0, \\ H(q, x) = \sum_{m,n} \alpha_{m,n} s_{m,n} \end{aligned} \tag{2.4}$$

where $\alpha_{m,n}$ and $s_{m,n}$ are the learned coefficient and the similarity score corresponding to the particular location of image patch m and the particular feature channel n. Similar to the method suggested in [Prosser et al., 2010], the goal is to build the ranking function as the weighted sum of the similarity or dissimilarity measures corresponding to each of the features. While [Prosser et al., 2010] suggest to use the absolute difference of the corresponding feature values, the formulation of [Kuo et al., 2013] allows more generality in defining of such similarity or dissimilarity measures (*e.g.*, different similarity measures can be used for different feature channels). Differently from [Prosser et al., 2010], where the weak rankers correspond to different subsets of learning data, boosting is performed here by adding the rankers that correspond to the particular local features. Thus α weights are sequentially computed at each of the boosting rounds.

The person re-identification problem is also approached in [Paisitkriangkrai et al., 2015]. The authors aim directly at optimizing the Cumulative Matching Characteristic curve (CMC) that is used as the main performance measure for this problem. The two different formulations of the problem are considered: the first one, called $CMC^{triplet}$, is to minimize k for which the CMC value (or the rank-k recognition rate) approaches 100%, the second one, called CMC^{TOP}, is to maximize the top-k recognition rate. The second formulation is motivated by the fact that in re-identification systems, the user typically only views the limited number of top matches for the query. Therefore the authors note that in this setting, it is more reasonable to maximize the quality of the top-k search results instead of minimizing the number of resulting examples necessary to achieve 100% recognition rate. The optimization problems are formulated for each of the two tasks. The goal of the both tasks is to learn the weight vector w to combine the individual distance functions into one: $D(q,x) = w^{\mathsf{T}}\Phi(q,x)$, where $\Phi(q,x) = (d_1(q,x),...,d_T(q,x))$. For the $CMC^{triplet}$ formulation, the optimization problem coincides with the those suggested in [Joachims, 2002] and [Prosser et al., 2010].

Differently from $CMC^{triplet}$, the CMC^{TOP} formulation considers a set of constraints, each of them corresponding to one of the possible relative orderings within the training data. Each relative ordering of the training data corresponds to a number of reversions: the situations where the non-matching example is placed higher than the matched example for some query. The CMC^{TOP} formulation includes the triplet-based constraints, one for each of the possible orderings: the $w^{\mathsf{T}}\Phi(q,x_i) - w^{\mathsf{T}}\Phi(q,x_j)$ values that correspond to reversions are averaged for each ordering. The margins for the corresponding constraints are set to the number of reversions among the first k results of the corresponding ordering. Thus each additional reversion within the first k matches for one of the queries induces the new ordering and therefore the new constraint with margin value increased by one. As demonstrated in the experiments, CMC^{TOP} is most likely to result in higher recognition rate for smaller k values of the CMC curve then $CMC^{triplet}$.

2.3 Hand-crafted representations for person re-identification

While the distance metric learning and learning to rank are quite general frameworks applicable to a wide range of tasks, many works focus on the design of feature extracting methods specifically useful for human recognition and re-identification: [Liao et al., 2015, Bazzani et al., 2013, Ma et al., 2012a,b]. Moreover, some approaches, *e.g.*, the approach suggested in [Liao et al., 2015], combine the special feature design and metric learning.

In person re-identification, an ideal image representation should be robust to strong cross-view illumination changes and, at the same time, it is supposed to be highly discriminative in order to achieve successful recognition. There are many different types of image features that are mentioned in the pedestrian recognition literature. Color-based features (namely color histograms) are used in [Liao et al., 2015] and [Bazzani et al., 2013]. Texture-based features achieved by the application of Gabor [Fogel and Sagi, 1989] and Schmidt [Schmid, 2001] filters are utilized in [Gray and Tao, 2008]. Texture information captured by LBPs (Local Binary Patterns) [Ojala et al., 2002] was also used in [Köstinger et al., 2012]. [Jüngling et al., 2011] use SIFT (Scale-invariant feature transform) [Lowe, 2004], *i.e.*, the features based on interest points, for re-identification. The region based approaches include those suggested in [Tuzel et al., 2008, Bazzani et al., 2013].

It is also a common practice to combine different representation types either at the stage of computing the final image descriptor (*e.g.*, simple concatenation) or at the stage of estimating the similarity between the two images. The latter, for example, is done in [Bazzani et al., 2013], [Ma et al., 2012b] and [Ma et al., 2012a].

[Ma et al., 2012a] apply Gabor filters for different image scales and orientations followed by calculation of the Covariance descriptor suggested in [Tuzel et al., 2008] for a set of image regions. This combination is motivated by the fact the both components are known to be rather tolerant to illumination changes.

[Bazzani et al., 2013] aims at reducing the influence of background on the pedestrian image representation. The image regions based on the body structure are first defined: the two horizontal axes of asymmetry are first found, they should separate parts of an image that contain head, torso and legs correspondingly. Then the three vertical axes of symmetry are found for each of these parts. The features coming from different spatial locations are weighted in correspondence to the proximity to those symmetry axes to avoid taking background into account.

[Liao et al., 2015] introduces a feature representation called Local Maximal Occurrence (LOMO). To handle the severe illumination variation, the color restoration technique

introduced in [Jobson et al., 1997] is first applied. The local histogram-based features are then extracted from the horizontal stripes of the image, and, as opposed to the earlier methods suggested in [Zheng et al., 2011] and [Prosser et al., 2010], are maximized within each of the stripes, which allows for higher invariance to the viewpoint changes. This feature extraction scheme is repeated for the three image scales. It should also be noted that the authors also suggest a metric learning technique earlier mentioned in 2.1.

The non-trivial method for local descriptor aggregation is proposed in [Ma et al., 2012b]. The Gaussian Mixture Model is first learned for a set of local descriptors, then image representation is computed using the Fisher vector [Perronnin and Dance, 2007]. It is a notable fact, that the authors use a simple 7-dimensional local descriptors that only contain the information about the spatial location, raw pixel intensity and its derivatives. Nevertheless, it is demonstrated that the proposed representation results in better re-identification performance than other more sophisticated (*e.g.*, SIFT) types of local descriptors.

2.4 Training deep neural networks for similarity estimation

Further, we give a short description of several approaches within the deep learning field dedicated to image retrieval and specifically to human recognition.

Classification deep neural networks for retrieval

The performance of the features extracted from a deep neural network trained for the classification task has been first assessed in [Krizhevsky et al., 2012]. Images with small Euclidean distances between their deep representations were compared. It has been qualitatively demonstrated that such images are semantically close: they often represent the same class. While at the same time, they can be very different at the pixel level: the poses, orientation and illumination may vary greatly. Thus, along with the new state-of-the-art level for image classification, [Krizhevsky et al., 2012] have started the whole line of work dedicated to using deep features for image retrieval.

Afterward, [Razavian et al., 2014] have shown that the generic deep features in combination with simple classifiers (or just calculation of Euclidean distance) outperform traditional specially tuned systems for various recognition tasks, including object detection, attribute recognition, fine-grained recognition and image retrieval. Thus, the universal nature of such features has been demonstrated and additionally confirmed by the fact that the initial neural network used for the feature extraction had not been trained specially for any of the aforementioned tasks.

The using of deep representations for image retrieval has been further explored in [Babenko et al., 2014, Babenko and Lempitsky, 2015]. Unlike [Razavian et al., 2014] where the local descriptors are extracted at different locations and scales, the works of [Babenko et al., 2014] and [Babenko and Lempitsky, 2015] are focused on the construction of the holistic image descriptors either extracted from fully connected layers or computed as the aggregation of local convolutional descriptors. Thus, for a pair of images, Euclidean distance is only computed once: for the pair of their deep representations.

Siamese neural networks

The term of *siamese* neural networks has been first introduced in [Bromley et al., 1993] dedicated to the signature verification. Later works of [Chopra et al., 2005] and [Hadsell et al., 2006] continued this line of work by approaching face verification and dimensionality reduction tasks with similar techniques.

A siamese neural network consists of two identical subnetworks, often sharing their weights. Such a system is trained to optimize some predefined similarity measure (*e.g.*, L_2 distance or cosine similarity). Generally, each of the two subnetworks defines a non-linear mapping from the image space to the output descriptor space where the similarity or distance estimation is finally performed.

Differently from the above-mentioned works of [Krizhevsky et al., 2012, Razavian et al., 2014, Babenko et al., 2014, Babenko and Lempitsky, 2015], that show how classification deep neural networks can be used for extracting features useful for retrieval, *siamese* neural networks are specially trained to pull the matching images together in the output space and separate the non-matching images.

Similar to most deep learning approaches developed for a wide variety of visual recognition tasks, siamese neural networks allow to learn useful features that are specifically optimized for the target objective (incorporating similarity measure) instead of using hand-crafted features that may be sub-optimal for a particular case. This is in contrast to the methods described in the sections 2.1, 2.2 and 2.3. In more details, section 2.1 describes methods for learning a linear transform for the predefined features. Section 2.2 mentions approaches for learning weights to aggregate the per-feature similarities. Algorithms mentioned in section 2.3, aim at extracting special task-oriented features in such a way that the similarity for an image pair can be estimated by computing L_2 distance between the corresponding representations.

Thus, in light of the previously described methods, two important advantages of siamese architecture should be named.

- The ability to optimize specific similarity-based (or distance-based) objective functions. This advantage comes from the nature of siamese architecture and a large amount of the developed loss functions;
- The ability to merge the steps of feature learning and metric learning. This is an implication of the existence of the appropriate loss functions combined with a general property of neural networks to automatically learn necessary features.

This may result in learning the features that are robust to sources of variation inherent to a particular task, *e.g.*, poor alignment and even variation in pose and illumination. Indeed, [Chopra et al., 2005] note that the use of the convolutional siamese neural network helps to alleviate imprecise registration of face images and some lighting variations. Also, [Hadsell et al., 2006] show that the siamese neural network is able to capture the neighborhood relations based on pose information while ignoring strong illumination changes.

As many metric learning and ranking methods, siamese neural networks are applied to a wide variety of tasks: hashing [Lin et al., 2015], dimensionality reduction [Hadsell et al., 2006], patch matching [Simo-Serra et al., 2015], image retrieval [Song et al., 2016], face verification [Chopra et al., 2005, Sun et al., 2014, Parkhi et al., 2015, Schroff et al., 2015] and person re-identification [Yi et al., 2014a].

Pairwise losses

Contrastive loss

Contrastive loss has been first introduced in [Hadsell et al., 2006] for dimensionality reduction and visualization. Later it was used for training a face recognition system in [Sun et al., 2014] in combination with the cross-entropy classification loss. For a pair of image descriptors x_i and x_j it is defined as:

$$J_{contrastive}(x_i, x_j) = \begin{cases} \|x_i - x_j\|_2^2, & \text{if } (i,j) \text{ is a positive pair,} \\ (\max(0, M - \|x_i - x_j\|_2))^2, & \text{if } (i,j) \text{ is a negative pair,} \end{cases} \tag{2.5}$$

where M is a hyperparameter, it defines a margin, or minimal distance for a negative pair that does not affect the loss value. In other words, it states when the non-matching examples are far enough and there is no need to push them farther. The values of the Contrastive loss function for positive and negative pairs are shown in Figure 2.1a.

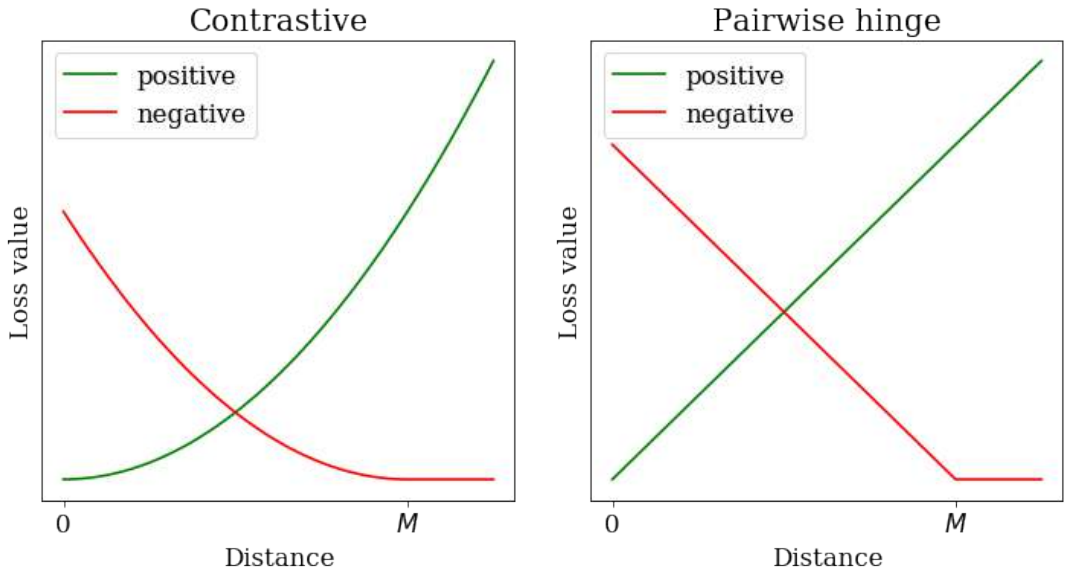


FIGURE 2.1: The values of the Contrastive loss variants: (a) - (2.5), (b) - (2.6). The losses are considered as functions of distance between a pair of image representations. The plots for positive and negative pairs are shown in green and red correspondingly.

An analogous loss function, called Coherence loss or pairwise hinge loss, was utilized in [Mobahi et al., 2009] and [Simo-Serra et al., 2015] (see Figure 2.1b for the corresponding plots):

$$J_{coherence}(x_i, x_j) = \begin{cases} \|x_i - x_j\|, & \text{if } (i,j) \text{ is a positive pair,} \\ \\ (\max(0, M - \|x_i - x_j\|)), & \text{if } (i,j) \text{ is a negative pair.} \end{cases} \tag{2.6}$$

However, [Lin et al., 2015] show that forcing every positive pair to be represented by one point may lead to performance collapse, so the authors introduce the double-margin version of the loss function:

$$J_{double}(x_i, x_j) = \begin{cases} \max(0, \|x_i - x_j\|_2^2 - M_1), & \text{if } (i,j) \text{ is a positive pair,} \\ \\ \max(0, M_2 - \|x_i - x_j\|_2^2), & \text{if } (i,j) \text{ is a negative pair,} \end{cases} \tag{2.7}$$

where M_1 and M_2 are the two margin hyperparameters for positive and negative pairs correspondingly.

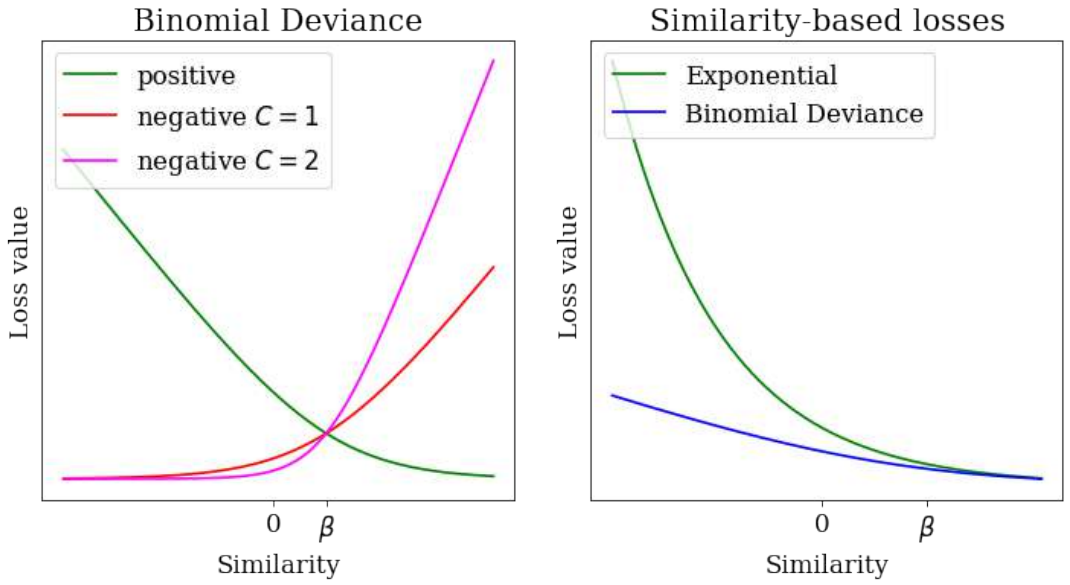


FIGURE 2.2: (a) - the values of the Binomial Deviance loss (2.8). The plot for negative pairs is shown for $C = 1$ and $C = 2$, higher values of C mean higher penalty for negative pairs violating the threshold β. (b) the values of the Binomial Deviance loss (2.8) and the Exponential loss (2.10) for positive pairs. The losses are considered as functions of distance between a pair of image representations.

Binomial Deviance loss

Binomial Deviance loss (generalized logistic loss) was considered in [Yi et al., 2014b] for training a siamese neural network for person re-identification. Differently to Constrastive loss (2.5), it is defined in terms of similarity for a pair of image descriptors. However, it should be noted, that the similar loss was also used in [Mignon and Jurie, 2012, Hu et al., 2014], but for Euclidean distance instead of cosine similarity.

Binomial Deviance loss in defined as

$$J_{dev}(s_{i,j}) = \ln(\exp^{-\alpha(s_{i,j}-\beta)m_{i,j}} + 1)\,, \tag{2.8}$$

where $s_{i,j}$ is the cosine similarity measure between ith and jth examples, α and β are the hyper-parameters. Furthermore, $m_{i,j}$ differ for positive and negative pairs:

$$m_{i,j} = \begin{cases} 1, & \text{if } (i,j) \text{ is a positive pair,} \\ \\ -C, & \text{if } (i,j) \text{ is a negative pair,} \end{cases} \tag{2.9}$$

The parameter C is the negative cost for balancing weights for positive and negative pairs that was introduced in [Yi et al., 2014b]. The authors demonstrate that this

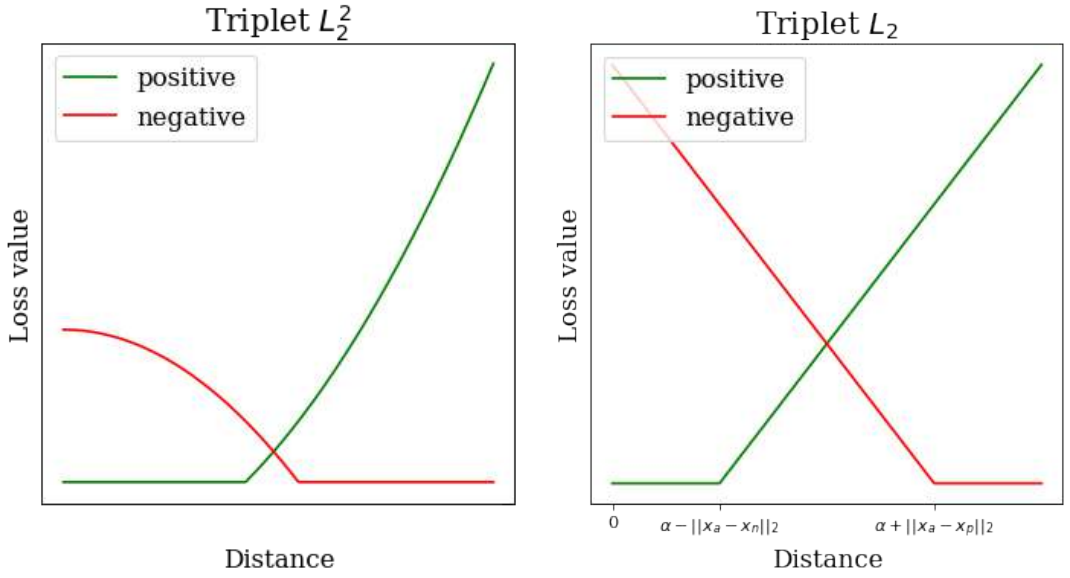


FIGURE 2.3: (a) plots the loss (2.11), (b) plots the loss (2.12). Green and red are for the loss as a function of $\|x_a - x_p\|_2$ and $\|x_a - x_n\|_2$ correspondingly (the other addend remains fixed).

parameter may influence the re-identification quality and results in better performance if tuned carefully. The plots of the Binomial Deviance loss for different similarity values and different values of C are shown in Figure 2.2a.

[Yi et al., 2014b] have additionally considered the exponential similarity-based loss:

$$J_{exp}(s_{i,j}) = \exp^{-\alpha(s_{i,j}-\beta)m_{i,j}} . \tag{2.10}$$

The corresponding plots for different similarity-based losses are shown in Figure 2.2b. The authors note that J_{exp} (2.10) assigns a higher penalty value to the most violating pairs (*i.e.*, positive pairs with low similarity), whereas J_{dev} (2.8) is more robust to outliers.

[Li et al., 2014, Ahmed et al., 2015] also use binary cross-entropy for pairs of images as an objective. But instead of using a mapping from image to descriptor space first, they map a couple of images straight to the similarity score.

Tripletwise losses

Triplet loss

[Schroff et al., 2015] and [Parkhi et al., 2015] use the triplet loss (similar to [Weinberger and Saul, 2009]) for face verification:

$$J^{L_2^2}_{triplet}(x_a, x_n, x_p) = \max(0, \alpha - \|x_a - x_n\|_2^2 + \|x_a - x_p\|_2^2) \tag{2.11}$$

The loss is defined for a triplet of image representations x_a, x_p and x_n, where (x_a, x_p) is a positive pairs and (x_a, x_n) is a negative pair. The idea of this loss is different to that of the pairwise losses: now we do not have any predefined threshold (or margin) that should separate all distances between positive pairs from all distances between negative pairs. In the triplet loss, such separation is learned independently for each anchor x_a. α is a hyperparameter that forces positive and negative distances to be additionally separated.

[Schroff et al., 2015] also point out the problem of sampling the training triplets: it is important to select such triplets that violate the condition of the ordering of x_p and x_n in terms of distance to the anchor x_a:

$$\|x_a - x_p\|_2^2 + \alpha < \|x_a - x_n\|_2^2$$

In other words, the triplet affects the learning process if the corresponding loss value (2.11) is positive. However, the authors also note that using the hardest (or most violating) triplets may lead to poor performance in practice. Therefore, the semi-hard sampling strategy was suggested. In more details, the triplet (x_a, x_p, x_n) violating a margin α is considered to be semi-hard if the following condition is satisfied:

$$\|x_a - x_p\|_2^2 < \|x_a - x_n\|_2^2$$

Later, it was also shown in [Wu et al., 2017] that the Contrastive loss (2.5) also benefits from utilizing this sampling scheme and the final results are on par with those of the triplet loss (2.11).

[Wu et al., 2017] argue that the poor results of training using hardest examples come from the concave shape of the loss (2.11) with respect to the distance between the negative pair $D_n = \|x_a - x_n\|_2$. The authors point out that the derivative of the loss with respect to D_n is approaching zero for small values of D_n, which corresponds to hard triplets. At the same time the derivative of the loss with respect to $D_p = \|x_a - x_p\|_2$ remains large. This may lead to imbalance in handling positive and negative components and result in

collapsing all representations to one point. [Wu et al., 2017] suggest that this effect may be reduced by using the following loss function instead:

$$J^{L_2}_{triplet}(x_a, x_n, x_p) = \max(0, \alpha - \|x_a - x_n\|_2 + \|x_a - x_p\|_2) \tag{2.12}$$

The plots for the variants $J^{L_2^2}_{triplet}$ (2.11) and $J^{L_2}_{triplet}$ (2.12) are shown in Figure 2.3.

Lifted Structured Similarity Softmax loss

The previously described loss functions are defined for separate pairs or triplets of training examples. The loss values for every such pair or triplet are averaged resulting in the final loss value. Differently from this scheme, [Song et al., 2016] define the loss function for a whole set of examples. In practice, the loss is calculated for all the examples in the training mini-batch.

The idea of the *Lifted Structured Similarity Softmax loss* (LSSS) suggested in [Song et al., 2016] is to take into account the hardest triplets. To make the loss function smooth, the logarithm of the sum of exponents is used. The LSSS loss is defined as follows:

$$\begin{aligned} J &= \tfrac{1}{2|\mathcal{P}|} \sum_{(i,j)\subseteq\mathcal{P}} \max(0, \bar{J_{i,j}})^2, \\ \bar{J_{i,j}} &= \log\Big(\sum_{(i,k)\subseteq\mathcal{N}} \exp\{\alpha - \|x_i - x_k\|_2\} + \\ &\qquad \sum_{(j,l)\subseteq\mathcal{N}} \exp\{\alpha - \|x_j - x_l\|_2\}\Big) + \|x_i - x_j\|_2, \end{aligned} \tag{2.13}$$

where $\mathcal{P}$ and $\mathcal{N}$ are the sets of positive and negative pairs in the mini-batch, α is a hyperparameter.

The LSSS loss works with the batch of images and for each positive pair (x_i, x_j) it samples all the related negative pairs (x_i, x_k) and (x_j, x_l). Thus, this approach is based on tripletwise learning. Another feature of this loss is that it is biased towards hard negatives.

N-pair loss

[Sohn, 2016] introduce another loss function based on triplets.

$$J_{N-pair}(\{x_i, x_i^+\}_i^N) = \frac{1}{N}\sum_{i=1}^{N}\log(1+\sum_{i\neq j}\exp^{x_i^\intercal x_j^+ - x_i^\intercal x_i^+}) \\ = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{\exp^{x_i^\intercal x_i^+}}{\exp^{x_i^\intercal x_i^+}+\sum_{i\neq j}\exp^{x_i^\intercal x_j^+}}, \tag{2.14}$$

where $\{x_i, x_i^+\}_i^N$ - N example pairs from N different classes. Similar to LSSS (2.13), it also uses triplet-based tuple structure where for each example pair multiple negatives are considered.

A similar loss (previously introduced in [Goldberger et al., 2005, Globerson and Roweis, 2006]), called Proxy-NCA loss, is used by [Movshovitz-Attias et al., 2017]. The difference is that the positive and negative examples for triplets are replaced by learnable per-class proxies. This results in reduced number of triplets and alleviates the problem of sampling (choosing useful triplets for training). This loss is shown to outperform previously discussed triplet (2.11), LSSS (2.13) and N-pair (2.14) losses.

Quadruplet losses

The following loss function is suggested in [Tadmor et al., 2016] and [Wu et al., 2017]:

$$J_{margin}(x_i, x_j) = \begin{cases} \max(0, \alpha + (\|x_i - x_j\|_2 - \beta)), & \text{if } (i,j) \text{ is a positive pair,} \\ \\ \max(0, \alpha - (\|x_i - x_j\|_2 - \beta)), & \text{if } (i,j) \text{ is a negative pair,} \end{cases} \tag{2.15}$$

Where α is a hyperparameter and β is a learnable threshold separating the values of distances between positive and negative pairs. This loss is called *margin-based* loss in [Wu et al., 2017], so I follow this notation. Although this loss is defined for a pair of representations (x_i, x_j), it is still related to the quadruplet learning, as the threshold β is learned. Note, that the contrastive loss (2.5) does not allow to learn the margin parameter M as it will converge to zero (the trivial solution). The margin-based loss (3.9) also resembles the double-margin contrastive loss (2.7).

Besides the loss function choice, another important part of the training scheme is sampling. Hard-negative mining has been initially adopted by [Schroff et al., 2015] for triplet-based learning. Later has been also shown by [Wu et al., 2017] that the results of contrastive (2.5) and margin (3.9) can be affected by the sampling scheme for negative examples. [Wu et al., 2017] propose the distance-weighted sampling. [Movshovitz-Attias et al., 2017], however, avoid the sampling issue by introducing learnable proxies each of which is representing multiple training data points.

2.5 CNN architectures for human recognition

Several CNN-based methods for person re-identification have been proposed recently: [Li et al., 2014, Yi et al., 2014b, Ahmed et al., 2015, Chen et al., 2016, Varior et al., 2016a,b, Su et al., 2016].

[Yi et al., 2014b] were among the first to build *siamese* architectures that embeds of pedestrian images into the descriptor space, where they can be further compared using angular distance.

In [Yi et al., 2014b], a peculiar architecture specific to pedestrian images is proposed that includes three independent sub-networks corresponding to three regions (legs, torso, head-and-shoulders). This is done in order to take into account the variability of the statistics of textures, shapes, and articulations between the three regions.

Apart from [Yi et al., 2014b], [Li et al., 2014] and [Ahmed et al., 2015] learn classification networks that can categorize a pair of images as either depicting the same subjects or different subjects. In both approaches, the two images are fed into the network, and the difference in their mid-level representation is processed by additional special layers. Patch matching layer is used in [Li et al., 2014] and cross-input neighbourhood difference - in [Ahmed et al., 2015]). After that, extra convolutional and inner product layers are applied to the combined representation of the pair of images. Finally, the classification for the pair of images is performed using the softmax layer. The proposed deep learning approaches [Ahmed et al., 2015, Yi et al., 2014b, Li et al., 2014], while competitive, do not clearly outperform more traditional approaches based on "hand-engineered" features [Paisitkriangkrai et al., 2015, Zhao et al., 2014].

When searching for matches in a dataset, the methods proposed in [Li et al., 2014], [Ahmed et al., 2015] and [Chen et al., 2016] need to process pairs that include the query and every image in the dataset, and hence cannot directly utilize fast retrieval methods based on Euclidean and other simple distances.

Overall, the closest works to our approach described in Chapter 4 are [Yi et al., 2014a, Cheng et al., 2016] as they used separate streams to process horizontal stripes of the pedestrian image, both works consider 2-depth convolutional streams.

The later work [Li et al., 2017] (after our work) use the same approach of multistream CNN (with a deeper 4-layer architecture), but the authors additionally use Spatial Transformer Networks on each of the substreams, showing the results better than ours. Even better results were shown in [Zhao et al., 2017], where a special pose-predicting subnetwork is utilized to propose part-based regions for feature pooling.

[Saquib Sarfraz et al., 2018, Suh et al., 2018, Kalayeh et al., 2018] continue the idea of using the pose information explicitly. [Li et al., 2017] has a view point predictor as a part of architecture. [Suh et al., 2018] and [Kalayeh et al., 2018] incorporate sub-networks for pose prediction and semantic segmentation correspondingly.

2.6 Multiplicative interactions of features

Covariance descriptor and second order pooling

The multiplicative interactions between separate elements of feature vectors have been utilized for different visual recognition tasks. For example, the covariance descriptor has been introduced in [Tuzel et al., 2008] for pedestrian detection and was further used for person re-identification and face verification in [Ma et al., 2012a]. The covariance descriptor of a region R covering local features $X = \{x_i\}_1^S, x_i \in \mathbb{R}^d$ is defined as:

$$C_R = \frac{1}{S-1}\sum_{i=1}^{S}(x_i - \mu)(x_i - \mu)^\mathsf{T} \tag{2.16}$$

Later, in [Carreira et al., 2012], a similar operation has been used to pool the local descriptors in the context of semantic segmentation and classification.

Bilinear convolutional neural networks

The Bilinear architecture suggested in [Tsung-Yu Lin and Maji, 2015] are much related to the two aforementioned approaches [Tuzel et al., 2008, Carreira et al., 2012]).

The bilinear operation is defined by (4.2). It can be noticed that it performs a second-order pooling operation similar to (2.16).

Differently to [Tuzel et al., 2008] and [Carreira et al., 2012], the features outer product is computed for two sets of local features A and B instead of one set of features X. The pooling is done over all the spatial locations, so the region R in (2.16) here corresponds to the whole featuremap. The final architecture can be trained end-to-end which allows to fine-tune the feature extractors.

The Bilinear neural networks are also related to several popular feature aggregation techniques: BoVW [Csurka et al., 2004], VLAD [Jégou et al., 2010] and Fisher Vector [Perronnin et al., 2010]. The reason is that these methods incorporate the encoding step. Each feature vector is encoded by a number of soft or hard assignments to the corresponding codes in the vocabulary. The resulting descriptor can be represented as an outer product of this encoding and the other part that is specific to the particular

aggregation technique. The authors note that such encoding can be also viewed as part detectors. This analogy between the traditional feature aggregation approaches and the Bilinear architecture is described in details in [RoyChowdhury et al., 2015]. The authors also show that the Bilinear architecture outperforms the Fisher Vector pooling applied to the local features extracted using a convolutional neural network.

Interestingly, the Bilinear neural networks are able to outperform other methods that rely on the part labeling, *e.g.*, [Branson et al., 2014]. It should be mentioned that the reported performance is on par with the performance of the Spatial Transformer Network [Jaderberg et al., 2015], that may learn the pose normalization without additional pose supervision.

Another important observation made in [RoyChowdhury et al., 2015] was about the performance of the Bilinear architecture for the birds species categorization. In turned out that the Bilinear architecture performs equally well for the initial images and for their tightly cropped versions. This is a piece of evidence that the Bilinear model is able to implicitly learn to detect the important image parts.

The authors have initially suggested the Bilinear neural networks for fine-grained recognition problems. These type of recognition is characterized by high intra-category variations and low inter-category variations. The difficulty is that in some cases the intra-category variations may overcome the inter-category ones. For example, when recognizing the bird species, one should be able to take aside the pose and illumination variation an at the same time see the subtle differences in the beak shapes.

The tasks of human recognition can also be described as fine-grained recognition. Moreover, the Bilinear neural networks have been successfully applied to face recognition with high pose variation in [Tsung-Yu Lin and Maji, 2015]. Interestingly, in the latter, it was shown that the Bilinear architecture performs better than simple neural networks even without fine-tuning the feature extractors.

After computing the outer products of the feature vectors, the dimensionality of the resulting feature is squared. This does not constitute a storage or computational problem when operating the low-dimensional features like [Tuzel et al., 2008], where 8-length local descriptors were used resulting in the 64-element covariance matrices. But for a large number of feature maps, the result of applying the bilinear operation (4.2) may be overwhelming. [Gao et al., 2016] address this problem by incorporating Random Maclaurin projection into the deep learning framework, which allows for the lower-dimensional results of the bilinear operation (4.2) with only a small performance drop.

Later (after the publication of the results of this work), [Suh et al., 2018] has also successfully approached person re-identification using Bilinear architecture. The two

feature extractors explicitly model body part detection and appearance recognition: the former has been initialized by the pose detection model of [Cao et al., 2017], and the latter - is the GoogLeNet model [Szegedy et al., 2015] pre-trained for classification.

The latest work in [Kalayeh et al., 2018] also applied similar a two-stream-multiplication approach to person re-identification (but is not considered as Bilinear by the authors). Like in the discussed works, one stream extracts appearance information. The other explicitly outputs probability maps for each of the semantic regions associated with different body parts. These maps are used as masks to pool the appearance features inside each semantic region.

Attention

Another related line of work aims at explicitly modeling the attention mechanism of perceiving natural sentences and images. At a high level, this can also be described as learning "what" and "where" concepts and combining them using multiplication.

Initially, the simultaneous learning of the deep representations and the corresponding attention weights were introduced in the seminal work of [Bahdanau et al., 2015] for neural machine translation. The goal is to encode the input sentence into a sequence of so-called annotation vectors and choose which of them to use (or attend) when generating the next word of the resulting translation. To perform this selection of annotation vectors, a set of corresponding weights is learned. This technique is developed to avoid encoding the whole sentence into a single vector, which leads to poor performance in some cases, *e.g.*, for long sentences.

The following work of [Xu et al., 2015] use the attention approach to learn visual attention maps for image captioning. [Bahdanau et al., 2015] compute annotation vectors for a word sequence as activations of a bidirectional recurrent neural network, similarly, [Xu et al., 2015] use a convolutional neural network to produce annotation vectors for a set of image locations.

2.7 Domain adaptation

Domain adaptation

One popular direction in *transfer learning* is *domain adaptation*. This refers to a case when a task is being solved for two different domains, as defined in [Pan and Yang, 2010]. The goal is to adapt the predictive algorithm trained on the labeled source data to the data in the target domain.

The common assumption in domain adaptation, called *covariate shift* assumption, is that the difference between the two domains mainly comes from the difference in the feature distributions. In more details, under the *covariate shift* assumption, only the marginal distributions of two domains differ, whereas the posterior probabilities of the labels in each domain coincide. This assumption also implies that the feature space is the same for the two domains. Under this assumption, the domain adaptation task is to make the domain marginal distributions closer to each other.

[Huang et al., 2007, Pan et al., 2008, 2011, Baktashmotlagh et al., 2013] estimate the domain distribution difference using Maximum Mean Discrepancy (MMD). The latter has been introduced in [Borgwardt et al., 2006] to perform the comparison of two samples without the need to estimate the corresponding distributions (in a parametric or non-parametric way). The latter is achieved by utilizing a Reproducing Kernel Hilbert Space (RKHS) and the fact that squared MMD can be expressed in terms of the corresponding kernel function.

The empirical estimate of Maximum Mean Discrepancy is given by:

$$MMD(X_{\mathcal{D}_{\mathrm{S}}^X}, X_{\mathcal{D}_{\mathrm{T}}^X}) = \| \frac{1}{n_{\mathcal{D}_{\mathrm{S}}^X}} \sum_{i=1}^{n_{\mathcal{D}_{\mathrm{S}}^X}} \phi\left(x_i^{\mathcal{D}_{\mathrm{S}}^X}\right) - \frac{1}{n_{\mathcal{D}_{\mathrm{T}}^X}} \sum_{i=1}^{n_{\mathcal{D}_{\mathrm{T}}^X}} \phi\left(x_i^{\mathcal{D}_{\mathrm{T}}^X}\right) \|_{\mathcal{H}}, \tag{2.17}$$

where $\{x_i^{\mathcal{D}_{\mathrm{S}}^X}\}_{i=1}^{n_{\mathcal{D}_{\mathrm{S}}^X}}$ and $\{x_i^{\mathcal{D}_{\mathrm{T}}^X}\}_{i=1}^{n_{\mathcal{D}_{\mathrm{T}}^X}}$ are the samples from the source and the target domains correspondingly, $\mathcal{H}$ is RKHS and ϕ is the corresponding feature mapping.

However, the mentioned works approach reducing the Maximum Mean Discrepancy differently. [Huang et al., 2007] use sample reweighting to choose the most 'useful' examples of the source domain. The idea is to reweight the exampled from the source domain in such a way that the expected risk (associated with the task being solved), for the reweighted source data will be equal to the expected risk for the target data. Such weights are given by $\beta(x, y) = \frac{P_{\mathcal{D}_{\mathrm{T}}^X}(x,y)}{P_{\mathcal{D}_{\mathrm{S}}^X}(x,y)}$. And given the covariate shift assumption, $\beta(x) = \frac{P_{\mathcal{D}_{\mathrm{T}}^X}(x)}{P_{\mathcal{D}_{\mathrm{S}}^X}(x)}$. To avoid estimating the domain distributions, the authors learn the weights β by minimizing the empirical estimate of Maximum Mean Discrepancy between the target domain and the 'reweighted' source domain.

Differently to [Huang et al., 2007], many other approaches seek the transformation of the source feature space (or source and target together) so that the source and target distributions match better. The point of such approaches is to extract the information that is invariant across the source and target domains.

[Pan et al., 2008] aim at finding the lower dimensional space of features for this purpose, the method is called Maximum Mean Discrepancy Embedding. The motivation behind

this approach is to get rid of those variation factors of the feature space that cause the marginal distributions of the domains to differ. The authors construct a semidefinite program in terms of the kernel matrix based on (2.17) (with constraints ensuring that the distances between the nearest neighbors are preserved). PCA is then applied to the solution to get the final low-dimensional representations of the source and target examples. One disadvantage of this method is that it does not include any parametrization corresponding to a mapping onto the new feature space, therefore it cannot be used for the out-of-sample data. Such parametrization is introduced in the following work [Pan et al., 2011], where the learning problem of [Huang et al., 2007] is reformulated in terms of factorized and parametrized kernel matrix to enable the out-of-sample kernel evaluations. However, this causes the comparison of the sample means in a lower-dimension space rather than in Reproducing Kernel Hilbert Space. A somewhat similar approach correcting this drawback has been suggested in [Baktashmotlagh et al., 2013]. In the latter, the MMD minimization is formulated in terms of the linear projection of the initial data, therefore an explicit transformation of the feature space is learned.

While many approaches use Maximum Mean Discrepancy (2.17), another way of matching different domains is used in [Gong et al., 2012]. The authors model the gradual change between the domains. The two domains may have very different axes of variations, therefore the two d-dimensional subspaces, where the corresponding variances for each domain are maximized (*i.e.*, found using PCA), will be far from each other on the Grassmann manifold. The goal is to explore the intermediate subspaces on this manifold to extract the domain-invariant features. Moreover, the authors apply the kernel trick to take into account all the intermediate subspaces along the curve connecting the source and target domains.

The resampling technique similar to [Huang et al., 2007] is utilized in [Gong et al., 2013] in combination with the approach of [Gong et al., 2012]. The approach is based on the idea of solving multiple easier domain adaptation subtasks and then using their solutions as a basis for the final domain-invariant representation. Subtasks correspond different combinations of the examples in each of the two domains: to formulate each subtask, some examples are removed from the source domain and added to the target domain. Such examples are denoted *landmarks* and are chosen by thresholding the results of sample reweighting (each subtask - with different kernel parameters). Therefore the examples from the source domain that are closest to the target domain are chosen in different ways. This allows to solve easier adaptation tasks, moreover, this gives multiple different perspectives on the initial task. The representations achieved by solving each of the subtasks are scaled and concatenated, and the scaling weights are learned discriminatively using the landmark examples and their labels.

Deep domain adaptation

Feature-level domain adaptation. Like many of the above-mentioned methods, deep learning methods for domain adaptation aim at learning a data representation that would be the best at explaining both domains simultaneously. However, deep learning gives the tools for learning such features at multiple levels, whereas non-deep techniques work with data descriptions that are fixed and extracted beforehand. Deep learning methods, in contrast, are able to work with the raw input and extract a hierarchy of features. Thus, like for many other recognition tasks, multiple successful deep learning methods have been suggested for domain adaptation.

One of the first deep domain adaptation methods was suggested in [Glorot et al., 2011] for sentiment analysis. The Denoising Autoencoder [Vincent et al., 2008] is trained on the mixed data from both domains to capture their common concepts. Like in the non-deep approaches, described above, the transformed labeled data are then used for training the final domain-invariant classifier.

[Chopra et al., 2013] combines the ideas of [Gong et al., 2012] and [Glorot et al., 2011] to propose a deep learning framework interpolating between the two domains. Unlike [Glorot et al., 2011], both the task-specific prediction algorithm and domain-invariant feature extractors are neural networks. This allows finetuning the parameters of both these parts simultaneously by propagating the gradients from the task-specific network to the feature extractors. The gradual change between the domains is modeled by constructing multiple variants of the mixture of the two domains: different proportions of the data from the two domains are used. Each of the feature extractors is trained in an unsupervised manner using one of such mixtures. The final representation is composed as a concatenation of outputs of all of these feature extractors.

Finally, [Long et al., 2015] suggest a deep neural network, called DAN (Deep Adaptation Network), that is trained with the two objectives: one is a task-specific loss and another is to match the distributions of the two domains using Maximum Mean Discrepancy (2.17). The matching is performed for several higher-level layers of the main neural network to ensure that the domains are aligned well, as the last layers of a neural network are considered to be the most domain-specific ones. [Tzeng et al., 2014] similarly use (2.17) as an additional objective to train a domain-invariant neural network. In the latter, the authors use only one layer for matching the domain distributions. This layer is chosen among the last layers of the neural network so that the corresponding MMD value is minimized.

Notably, the deep approaches of [Chopra et al., 2013, Tzeng et al., 2014, Long et al., 2015] outperform the previous non-deep methods, *e.g.*, the strong method of [Gong et al., 2012], by a large margin on the cross-domain classification.

Image-level domain adaptation. Since the CycleGAN [Zhu et al., 2017] architecture for image-to-image translation and stylization appeared, domain adaptation has become one of its active fields of application. This approach differs from the feature-level domain adaptation techniques of [Long et al., 2015] and [Tzeng et al., 2014] or the method presented in the Chapter 5, because rather than finding deep domain-invariant representations, it works on the pixel level and aims at building mappings between the image domains. Thus the domain adaptation is done in two steps: building a mapping the source domain to target and retraining the predictor on the transferred source data. (Although it is also possible to combine these two steps into one optimization process.)

The methods of [Hoffman et al., 2018, Deng et al., 2018, Murez et al., 2018] employ CycleGAN framework to change the domain of training data for various tasks, including classification, segmentation, and person re-identification.

In particular, for person re-identification, the pixel-level domain adaptation with CycleGAN has been applied in several recent works [Zhong et al., 2018, Deng et al., 2018] (after the results of the Chapter 5 were published). Some of them consider different datasets as domains, others aim at utilizing synthetic re-identification data to improve the results on real data [Bak et al., 2018]. As demonstrated by these works, image-to-image translation may help a lot to overcome the visual (*e.g.*, caused by different illumination) differences between the source and target camera sets.

[Hong et al., 2017] and [Sohn et al., 2017] approach face recognition tasks for low-quality image domain. They consider image degradation as a domain shift and perform feature-level unsupervised domain adaptation based on adversarial learning showing better recognition results. In works [Hong et al., 2017] and [Sohn et al., 2017] it is shown that domain-specific data augmentation is essential for training face recognition systems. However, in both works, the data augmentation is performed 'by hand' (the degradation types and hyper-parameters for transforms are chosen and fixed).

Chapter 3

Learning Deep Embeddings with Histogram Loss

3.1 Motivation

The problem of learning deep embeddings is considered in this chapter. Usually, such learning is performed using siamese architectures described in Section 1.2.1 to solve the similarity estimation Problem 3. Under this approach, complex input patterns (*e.g.*, images) are mapped into a high-dimensional space through a chain of feed-forward transformations, while the parameters of the transformations are learned from a large amount of supervised data. The *objective* of the learning process is to achieve the proximity of semantically related patterns (*e.g.*, images of the same person) and avoid the proximity of semantically unrelated (*e.g.*, images of different people) in the target space. It should be noted that this work is focused on simple similarity measures such as Euclidean distance or scalar products, as they allow fast evaluation.

The formulation of an objective function for learning embeddings depends on the exact formulation of Problem 3. It can be formalized in several different ways depending on the setting:

1. whether we need all the positive pairs to have higher similarity values than *all* the negative pairs,
2. or we only need this condition fulfilled for each separate identity.

In more detail, the latter means that positive pairs $(x_i^a, x_j^a)_{i,j}$ that consist of images of some identity a should have the higher similarity value compared only to corresponding

negative pairs $(x_i^a, x_k^n)_{i,k}$, that contain one image for the identity a and the second image - for some other identity n.

According to one of the two described assumptions, the training data for the similarity-learning tasks may be formed in one of the following basic ways:

- a set of positive pairs and a set of negative pairs:

$$\begin{aligned} S &= \{(x_i^a, x_j^a) : x_i^a \text{ and } x_j^a \text{ are similar}\}, \\ D &= \{(x_i^b, x_j^c) : x_i^b \text{ and } x_j^c \text{ are not similar}\}; \end{aligned} \tag{3.1}$$

- a set of triplets:

$$R = \{(x_i^a, x_j^a, x_k^n) : x_i^a \text{ is more similar to } x_j^p \text{ than to } x_k^n\}; \tag{3.2}$$

- a set of quadruplets:

$$Q = \{(x_i^a, x_j^a, x_k^b, x_l^c) : x_i^a \text{ and } x_j^a \text{ are more similar than } x_k^b \text{ and } x_l^c\}. \tag{3.3}$$

Training with pairs and quadruplets

The training data forms 3.1 and 3.3 correspond to the assumption 1. The pairwise data 3.1 require comparing the similarity of each pair to some pre-defined threshold in the process of training (such threshold should separate similarity values of positive and negative pairs). The using of quadruplet data 3.3 usually implies comparing the similarity values between the independent pairs (so no predefined threshold is necessary). While pairwise data case is more rigid and requires more assumptions about the data (the threshold value), handling of quadruplet data is more computationally expensive (number of comparisons will be quadratic in the number of pairs).

The pairwise form of training data is widely used in metric learning [Xing et al., 2003, Globerson and Roweis, 2006, Davis et al., 2007, Köstinger et al., 2012, Liao et al., 2015, Mignon and Jurie, 2012], and also for training siamese architectures by [Bromley et al., 1993, Hadsell et al., 2006], including those for face verification by [Sun et al., 2014, Hu et al., 2014] and person re-identification by [Yi et al., 2014b,a].

Quadruplet-based learning is less popular. The metric learning method of [Law et al., 2013] is one of the examples under this category. The loss function suggested by [Tadmor et al., 2016] (it was published simultaneously with the corresponding results of this work) on deep face verification can also be considered quadruplet-based. The reason is that the

threshold for separating similarities of positive and negative pairs is a learned parameter in this work.

Training with triplets

The triplet form of training data 3.2 corresponds to the assumption 2. Differently to pairwise 3.1 and quadruplet data 3.3, for triplets, only relative similarities/distances are compared. This means that the neighborhoods corresponding to different training classes are allowed to be of a different radius in the descriptor space. This is in contrast to using pairs and quadruplets, which implies that all distance (negative similarity) value for positive pairs should not exceed a certain threshold.

Training data organized as triplets are usually used for ranking formulations: either linear ranking functions like in methods by [Joachims, 2002, Prosser et al., 2010, Kuo et al., 2013, Paisitkriangkrai et al., 2015] or non-linear ranking functions parametrized by CNN [Chen et al., 2016]. Triplet-based approach is also used for metric learning [Schultz and Joachims, 2004b, Weinberger and Saul, 2009], and for training siamese neural networks for re-identification [Song et al., 2016] and even for the verification formulation of the face recognition problem [Schroff et al., 2015, Parkhi et al., 2015].

Classification objective

It has been observed in [Krizhevsky et al., 2012] and confirmed later in multiple works (*e.g.*, [Razavian et al., 2014]) that deep networks trained for classification can be used for deep embedding. In particular, it is sufficient to consider an intermediate representation arising in one of the last layers of the deep network. In human recognition task, identity numbers can be used as class labels in order to apply the classification training. [Taigman et al., 2014] dramatically improved the results for face verification using the classification objective for training a CNN. [Parkhi et al., 2015] use pre-training for classification followed by triplet-based training for the same task.

Overall, using classification objectives can help the final results [Sun et al., 2014], however, this work considers objective functions that are specially designed for learning deep embeddings.

Sampling

It has been later demonstrated in literature [Sohn, 2016, Wu et al., 2017], that sampling scheme for training tuples may be very important and can affect the results. In this chapter, we, however, use simple sampling: we generate all the possible pairs within the mini-batch.

Motivation for a new loss function

As it was mentioned in the several last paragraphs, the training objectives for embedding learning are most often based on using point-wise constraints 3.1, 3.2, 3.3. This leads to

- either presence of hyperparameters, (like thresholds, whose optimal values can vary for different training datasets),
- or using more complex structures of training data (like triplets or quadruplets).

In this chapter a new loss function that helps to avoid both of these issues is suggested. In designing this function we strive to avoid highly-sensitive parameters such as margins or thresholds of any kind. While processing a batch of data points, the proposed loss is computed in two stages.

1. Firstly, the two one-dimensional distributions of similarities in the embedding space are estimated, one corresponding to similarities between matching (*positive*) pairs, the other corresponding to similarities between non-matching (*negative*) pairs. The distributions are estimated in a simple non-parametric way (as histograms with linearly-interpolated values-to-bins assignments).
2. In the second stage, the overlap between the two distributions is computed by estimating the probability that the two points sampled from the two distribution are in a wrong order, i.e. that a random negative pair has a higher similarity than a random positive pair.

It should be noted that the way of estimating the overlap between the similarity distributions in the second stage is quite important. The reason we choose the reverse order probability described above is that this loss forces the similarity distribution for positive pairs to the right and the distribution for negative pairs - to the left with respect to the similarity value axis. The latter means that during the process of training, each positive pair gets a pulling signal and each negative pair - a repelling signal. For such measures as Kullback-Leibler divergence [Kullback and Leibler, 1951] or Bhattacharyya distance [Bhattacharyya, 1943], this may not be a case, especially if the similarity distributions given by untrained network are highly overlapping. The problem is that, in contrast to the suggested method, these measures do not imply the ordering of distribution means. However, it should be noted that this problem could be fixed by introducing an additional optimization objective, *e.g.*, the difference between means of the two distributions.

The two stages are implemented in a piecewise-differentiable manner, thus allowing to minimize the loss (i.e. the overlap between distributions) using standard backpropagation. The number of bins in the histograms is the only tunable parameter associated

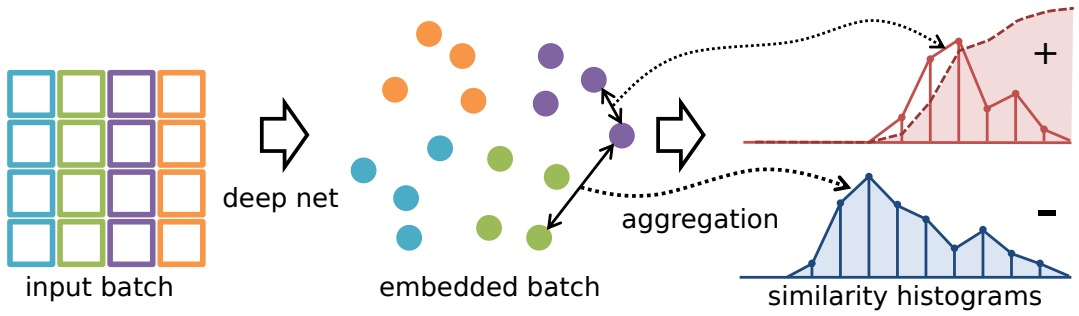


FIGURE 3.1: The histogram loss computation for a batch of examples (color-coded; same color indicates matching samples). After the batch (left) is embedded into a high-dimensional space by a deep network (middle), we compute the histograms of similarities of positive (top-right) and negative pairs (bottom-right). We then evaluate the integral of the product between the negative distribution and the cumulative density function for the positive distribution (shown with a dashed line), which corresponds to a probability that a randomly sampled positive pair has smaller similarity than a randomly sampled negative pair. Such histogram loss can be minimized by backpropagation. The only associated parameter of such loss is the number of histogram bins, to which the results have very low sensitivity.

with our loss, and it can be set according to the batch size independently of the data itself. In the experiments, we fix this parameter (and the batch size) and demonstrate the versatility of the loss by applying it to four different image datasets of varying complexity and nature. Comparing the new loss to state-of-the-art reveals its favorable performance. Overall, we hope that the proposed loss will be used as an "out-of-the-box" solution for learning deep embeddings that requires little tuning and leads to close to the state-of-the-art results.

3.2 Histogram loss

We now describe our loss function and then relate it to the quadruplet-based loss. Our loss (Figure 3.1) is defined for a batch of examples $X = \{x_1, x_2, \dots x_N\}$ and a deep feedforward network $f(\cdot;\theta)$, where θ represents learnable parameters of the network. We assume that the last layer of the network performs length-normalization, so that the embedded vectors $\{y_i = f(x_i;\theta)\}$ are $L2$-normalized.

We further assume that we know which elements should match to each other and which ones are not. Let m_{ij} be $+1$ if x_i and x_j form a positive pair (correspond to a match) and m_{ij} be -1 if x_i and x_j are known to form a negative pair (these labels can be derived from class labels or be specified otherwise). Given $\{m_{ij}\}$ and $\{y_i\}$ we can estimate the two probability distributions p^+ and p^- corresponding to the similarities in positive and negative pairs respectively. In particular $\mathcal{S}^+ = \{s_{ij} = \langle x_i, x_j \rangle \,|\, m_{ij} = +1\}$ and $\mathcal{S}^- = \{s_{ij} = \langle x_i, x_j \rangle \,|\, m_{ij} = -1\}$ can be regarded as sample sets from these two

distributions. Although samples in these sets are not independent, we keep all of them to ensure a large sample size.

Given sample sets $\mathcal{S}^+$ and $\mathcal{S}^-$, we can use any statistical approach to estimate p^+ and p^-. The fact that these distributions are one-dimensional and bounded to $[-1; +1]$ simplifies the task. Perhaps, the most obvious choice in this case is fitting simple histograms with uniformly spaced bins, and we use this approach in our experiments. We therefore consider R-dimensional histograms H^+ and H^-, with the nodes $t_1 = -1, t_2, \ldots, t_R = +1$ uniformly filling $[-1; +1]$ with the step $\Delta = \frac{2}{R-1}$. We estimate the value h_r^+ of the histogram H^+ at each node as:

$$h_r^+ = \frac{1}{|\mathcal{S}^+|} \sum_{(i,j)\,:\,m_{ij}=+1} \delta_{i,j,r} \tag{3.4}$$

where (i, j) spans all positive pairs of points in the batch. The weights $\delta_{i,j,r}$ are chosen so that each pair sample is assigned to the two adjacent nodes:

$$\delta_{i,j,r} = \begin{cases} (s_{ij} - t_{r-1})/\Delta, & \text{if } s_{ij} \in [t_{r-1}; t_r], \\ (t_{r+1} - s_{ij})/\Delta, & \text{if } s_{ij} \in [t_r; t_{r+1}], \\ 0, & \text{otherwise}\,. \end{cases} \tag{3.5}$$

We thus use linear interpolation for each entry in the pair set, when assigning it to the two nodes. The estimation of H^- proceeds analogously. Note, that the described approach is equivalent to using "triangular" kernel for density estimation; other kernel functions can be used as well.

Once we have the estimates for the distributions p^+ and p^-, we use them to estimate the probability of the similarity in a random negative pair to be more than the similarity in a random positive pair (*the probability of reverse*). Generally, this probability can be estimated as:

$$p_{\text{reverse}} = \int_{-1}^{1} p^-(x) \left[\int_{-1}^{x} p^+(y)\, dy \right] dx = \int_{-1}^{1} p^-(x)\, \Phi^+(x)\, dx = \mathbb{E}_{x \sim p^-} [\Phi^+(x)]\,, \tag{3.6}$$

where $\Phi^+(x)$ is the CDF (cumulative density function) of $p^+(x)$. The integral (3.6) can then be approximated and computed as:

$$L(X, \theta) = \sum_{r=1}^{R} \left(h_r^- \sum_{q=1}^{r} h_q^+ \right) = \sum_{r=1}^{R} h_r^- \phi_r^+\,, \tag{3.7}$$

where L is our loss function (the *histogram loss*) computed for the batch X and the embedding parameters θ, which approximates the reverse probability; $\phi_r^+ = \sum_{q=1}^{r} h_q^+$ is

the cumulative sum of the histogram H^+.

Importantly, the loss (3.7) is differentiable w.r.t. the pairwise similarities $s \in \mathcal{S}^+$ and $s \in \mathcal{S}^-$. Indeed, it is straightforward to obtain $\frac{\partial L}{\partial h_r^-} = \sum_{q=1}^{r} h_q^+$ and $\frac{\partial L}{\partial h_r^+} = \sum_{q=r}^{R} h_q^-$ from (3.7). Furthermore, from (3.4) and (3.5) it follows that:

$$\frac{\partial h_r^+}{\partial s_{ij}} = \begin{cases} \frac{+1}{\Delta|\mathcal{S}^+|}, & \text{if } s_{ij} \in [t_{r-1}; t_r], \\ \frac{-1}{\Delta|\mathcal{S}^+|}, & \text{if } s_{ij} \in [t_r; t_{r+1}], \\ 0, & \text{otherwise}, \end{cases} \tag{3.8}$$

for any s_{ij} such that $m_{ij} = +1$ (and analogously for $\frac{\partial h_r^-}{\partial s_{ij}}$). Finally, $\frac{\partial s_{ij}}{\partial x_i} = x_j$ and $\frac{\partial s_{ij}}{\partial x_j} = x_i$. One can thus backpropagate the loss to the scalar product similarities, then further to the individual embedded points, and then further into the deep embedding network.

Relation to quadruplet loss.
Our loss first estimates the probability distributions of similarities for positive and negative pairs in a semi-parametric ways (using histograms), and then computes the probability of reverse using these distributions via equation (3.7). An alternative and purely non-parametric way would be to consider all possible pairs of positive and negative pairs contained in the batch and to estimate this probability from such set of pairs of pairs. This would correspond to evaluating a quadruplet-based loss similarly to [Law, Zheng et al., 2013]. The number of pairs of pairs in a batch, however tends to be quartic (fourth degree polynomial) of the batch size, rendering exhaustive sampling impractical. This is in contrast to our loss, for which the separation into two stages brings down the complexity to quadratic in batch size. Another efficient loss based on quadruplets is introduced in [Tadmor et al., 2016]. The training is done pairwise, but the threshold separating positive and negative pairs is also learned.

We note that quadruplet-based losses as in [Law, Zheng et al., 2013] often encourage the positive pairs to be more similar than negative pairs by some non-zero margin. It is also easy to incorporate such non-zero margin into our method by defining the loss to be:

$$L_\mu(X, \theta) = \sum_{r=1}^{R} \left(h_r^- \sum_{q=1}^{r+\mu} h_q^+ \right) , \tag{3.9}$$

where the new loss effectively enforces the margin $\mu\,\Delta$. We however do not use such modification in our experiments (preliminary experiments do not show any benefit of introducing the margin).

Computational complexity.
The complexity of computing the Histogram loss in the batch size n and the number of histogram bins R is $O(n^2 \log R + R)$, as first each pair is assigned to the corresponding bin, then the estimates of two cumulative density functions are computed, and finally, the loss (3.7) is computed.

3.3 Experiments

In this section we present the results of embedding learning. We compare our loss to state-of-the-art pairwise and triplet losses, which have been reported in recent works to give state-of-the-art performance on these datasets.

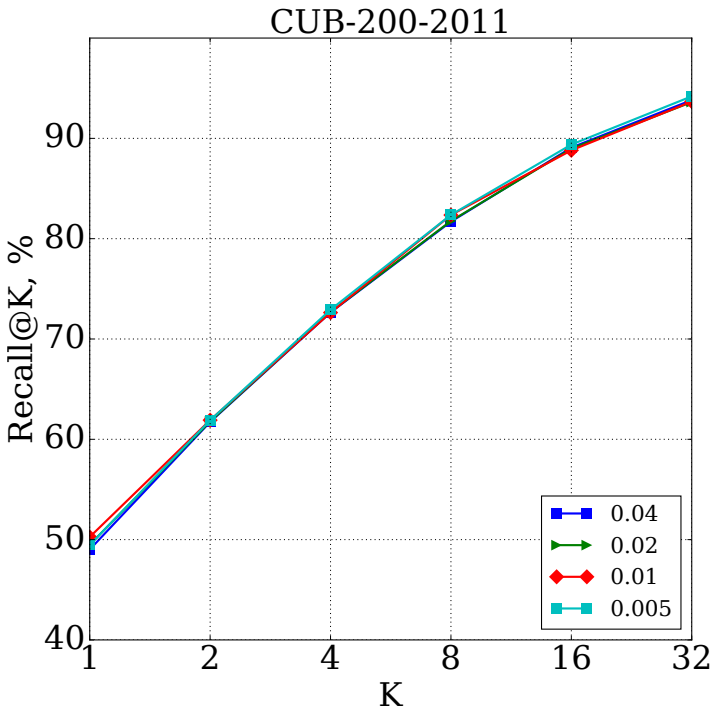
Baselines.
In particular, we have evaluated the Binomial Deviance loss (2.8) [Yi et al., 2014a]. While we are aware only of its use in person re-identification approaches, in our experiments it performed very well for product image search and bird recognition significantly outperforming the baseline pairwise (contrastive) loss reported in [Song et al., 2016], once its parameters are tuned.

Our experimental results suggest that the quality of the embedding is sensitive to parameter C of the Binomial Deviance loss. Therefore, in the experiments we report results for the two versions of the loss: with $C = 10$ that is close to optimal for re-identification datasets, and with $C = 25$ that is close to optimal for the product and bird datasets.

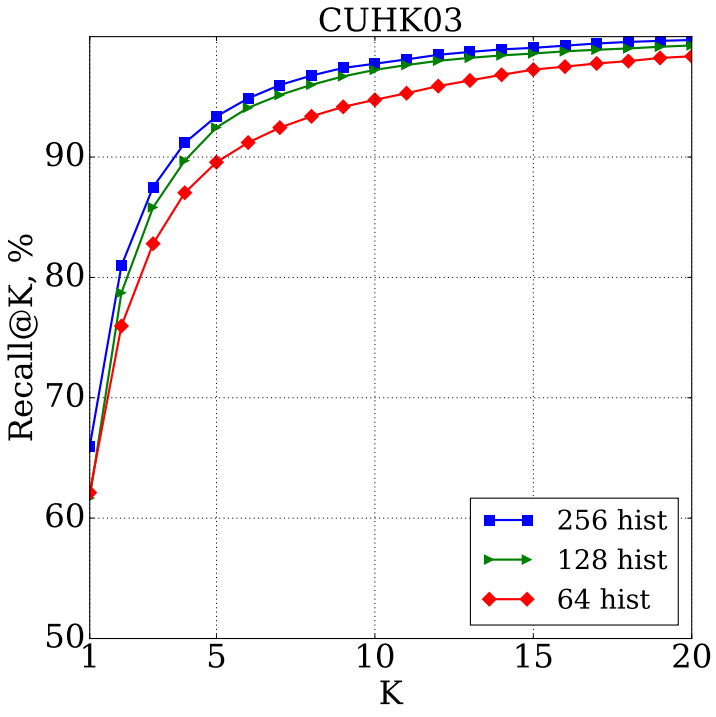
We have also computed the results for the Lifted Structured Similarity Softmax (LSSS) loss (2.13) [Song et al., 2016] on CUB-200-2011 [Wah et al., 2011] and Online Products [Song et al., 2016] datasets and additionally applied it to re-identification datasets. Lifted Structured Similarity Softmax loss is triplet-based and uses sophisticated triplet sampling strategy that was shown in [Song et al., 2016] to outperform standard triplet-based loss.

Additionally, we performed experiments for the triplet loss [Schroff et al., 2015] that uses "semi-hard negative" triplet sampling. Such sampling considers only triplets violating the margin, but still having the positive distance smaller than the negative distance.

Datasets and evaluation metrics.
We have evaluated the above mentioned loss functions on the four datasets : CUB200-2011 [Wah et al., 2011], CUHK03 [Li et al., 2014], Market-1501 [Zheng et al., 2015a] and Online Products [Song et al., 2016]. All these datasets have been used for evaluating methods of solving embedding learning tasks. Specifically, the Online Products dataset

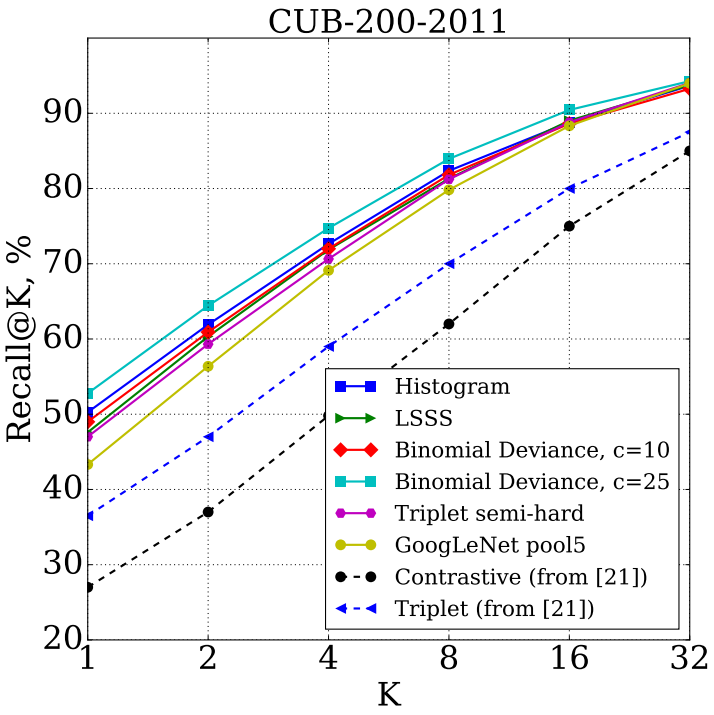


(a)

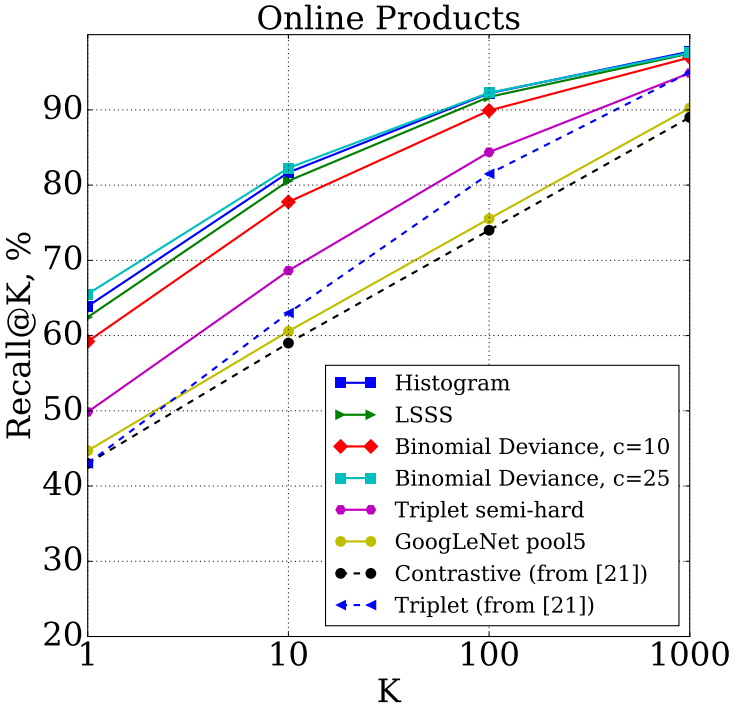


(b)

FIGURE 3.2: (a) - Recall@K for the CUB-200-2011 dataset for the Histogram loss (3.7). Different curves correspond to variable histogram step Δ, which is the only parameter inherent to our loss. The curves are very similar for CUB-200-2011. (b) - Recall@K for the CUHK03 labeled dataset for different batch sizes. Results for batch size 256 is uniformly better than those for smaller values.



(a)



(b)

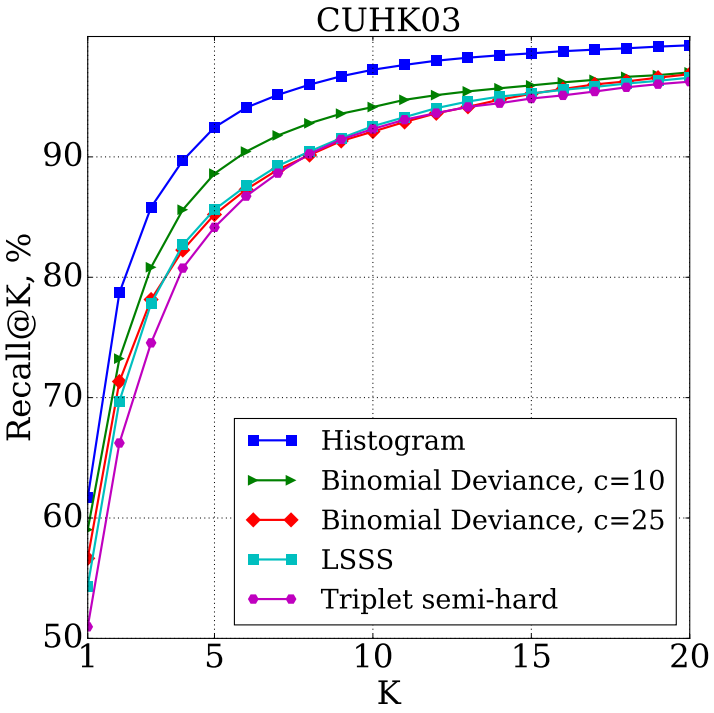
FIGURE 3.3: Recall@K for (a) - CUB-200-2011 and (b) - Online Products datasets for different methods. Results for the Histogram loss (3.7), Binomial Deviance (2.8), LSSS (2.13) [Song et al., 2016] and Triplet (2.11) [Schroff et al., 2015] losses are present. Binomial Deviance loss for $C = 25$ outperforms all other methods. The best-performing method is Histogram loss. We also include results for contrastive and triplet losses from [Song et al., 2016].

introduced in [Song et al., 2016] along with CUB200-2011 was used for training a deep image retrieval model [Song et al., 2016]. CUHK03 and Market-1501 are the largest datasets for person re-identification available at the moment.

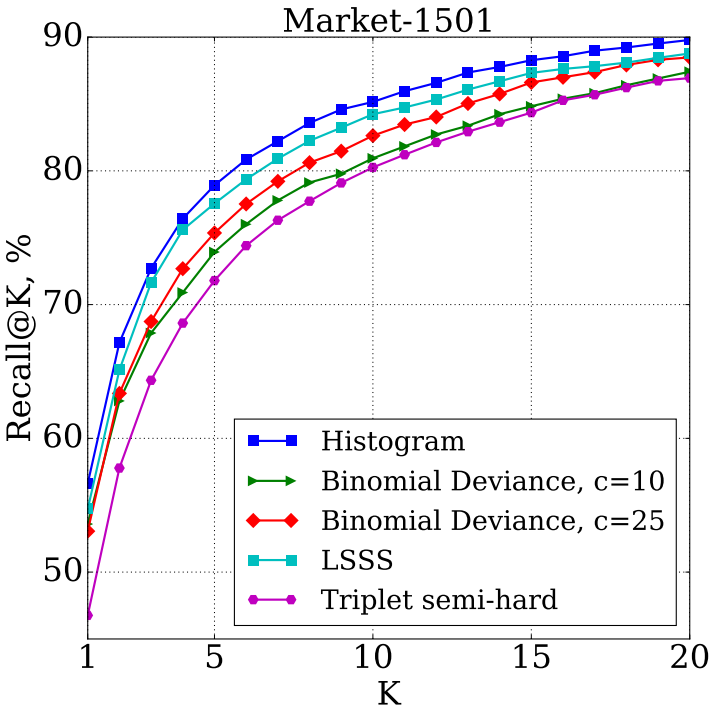
Commonly, for the retrieval tasks training and testing are done using disjoint sets of classes. For many datasets, including above mentioned ones, the standard splits for evaluation are provided. The CUB-200-2011 dataset includes 11,788 images of 200 classes corresponding to different birds species. As in [Song et al., 2016] we use the first 100 classes for training (5,864 images) and the remaining classes for testing (5,924 images). The Online Products dataset includes 120,053 images of 22,634 classes. Classes correspond to a number of online products from eBay.com. There are approximately 5.3 images for each product. We used the standard split from [Song et al., 2016]: 11,318 classes (59,551 images) are used for training and 11,316 classes (60,502 images) are used for testing. The images from the CUB-200-2011 and the Online Products datasets are resized to 256 by 256, keeping the original aspect ratio (padding is done when needed). The Recall@K metric is calculated for the set of queries consisting of all the images in the test set.

Following [Song et al., 2016, Yi et al., 2014a, Zheng et al., 2015a], we report Recall@K metric for all the datasets. For CUB-200-2011 and Online products, every test image is used as the query in turn and remaining images are used as the gallery correspondingly. In contrast, for CUHK03 *single-shot* results are reported. This means that one image for each identity from the test set is chosen randomly in each of its two camera views. Recall@K values for 100 random query-gallery sets are averaged to compute the final result for a given split. For the Market-1501 dataset, we use the *multi-shot* protocol (as is done in most other works), as there are many images of the same person in the gallery set.

Architectures used.
For training on the CUB-200-2011 and the Online Products datasets we used the same architecture as in [Song et al., 2016], which conincides with the GoogleNet architecture [Szegedy et al., 2015] up to the 'pool5' and the inner product layers, while the last layer is used to compute the embedding vectors. The GoogleNet part is pretrained on ImageNet ILSVRC [Russakovsky et al., 2015] and the last layer is trained from scratch. As in [Song et al., 2016], all GoogLeNet layers are fine-tuned with the learning rate that is ten times less than the learning rate of the last layer. We set the embedding size to 512 for all the experiments with this architecture. We reproduced the results for the LSSS loss (2.13) [Song et al., 2016] for these two datasets. For the architectures that use the Binomial Deviance loss, Histogram loss and Triplet loss the iteration number and the parameters value (for the former) are chosen using the validation set.



(a)



(b)

FIGURE 3.4: Recall@K for (a) - CUHK03 and (b) - Market-1501 datasets. The Histogram loss (3.7) outperforms Binomial Deviance, LSSS and Triplet losses.

TABLE 3.1: Final results for CUHK03-labeled and Market-1501. For CUHK03-labeled results for 5 random splits were averaged. Batch of size 256 was used for both experiments.

Dataset	r = 1	r = 5	r = 10	r = 15	r = 20
CUHK03	65.77	92.85	97.62	98.94	99.43
Market-1501	59.47	80.73	86.94	89.28	91.09

For training on CUHK03 and Market-1501 we used the Deep Metric Learning (DML) architecture introduced in [Yi et al., 2014a]. It has been described in Section 1.4.

Implementation details.

For all the experiments with loss functions (3.7) and (2.8) we used quadratic number of pairs in each batch (all the pairs that can be sampled from batch). For triplet loss "semi-hard" triplets chosen from all the possible triplets in the batch are used. For comparison with other methods the batch size was set to 128. We sample batches randomly in such a way that there are several images for each sampled class in the batch. We iterate over all the classes and all the images corresponding to the classes, sampling images in turn. The sequences of the classes and of the corresponding images are shuffled for every new epoch. CUB-200-2011 and Market-1501 include more than ten images per class on average, so we limit the number of images of the same class in the batch to ten for the experiments on these datasets. We used ADAM [Kingma and Ba, 2014] for stochastic optimization in all of the experiments. For all losses the learning rate is set to $1e-4$ for all the experiments except ones on the CUB-200-2011 datasets, for which we have found the learning rate of $1e-5$ more effective. For the re-identification datasets the learning rate was decreased by 10 after the 100K iterations, for the other experiments learning rate was fixed. The iterations number for each method was chosen using the validation set.

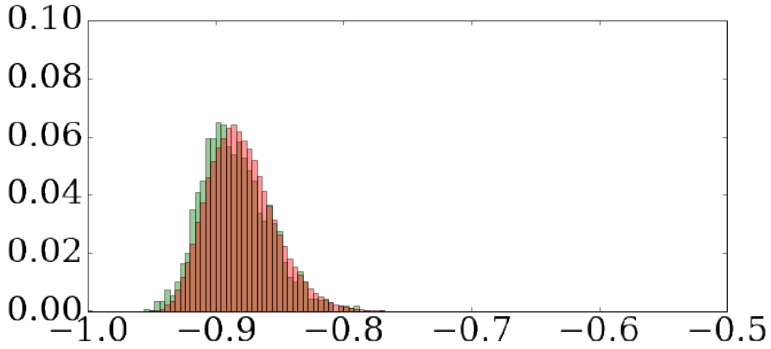
Results.

The Recall@K values for the experiments on CUB-200-2011, Online Products, CUHK03 and Market-1501 are shown in Figure 3.3 and Figure 3.4. The Binomial Deviance loss (2.8) gives the best results for CUB-200-2011 and Online Products with the C parameter set to 25. We previously checked several values of C on the CUB-200-2011 dataset and found the value $C = 25$ to be the optimal one. We also observed that with smaller values of C the results are significantly worse than those presented in the Figure 3.3a (for C equal to 2 the best Recall@1 is 43.50%). For CUHK03 the situation is reverse: the Histogram loss gives the boost of 2.64% over the Binomial Deviance loss with $C = 10$ (which we found to be optimal for this dataset). The results are shown in the figure

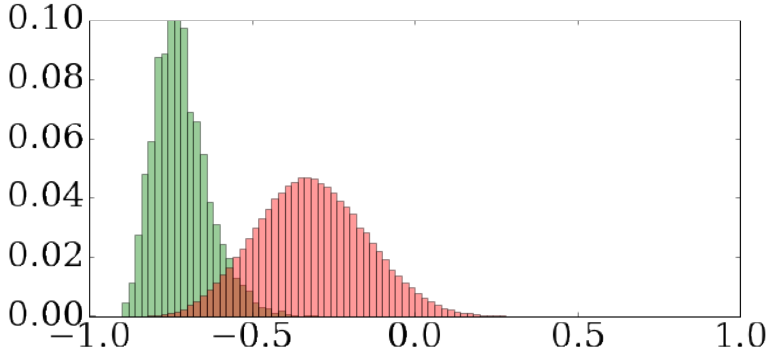
Figure 3.4a. Embedding distributions of the positive and negative pairs from CUHK03 test set for different methods are shown in Figure 3.5. For the Market-1501 dataset our method also outperforms the Binomial Deviance loss for both values of C. In contrast to the experiments with CUHK03, the Binomial Deviance loss appeared to perform better with C set to 25 than to 10 for Market-1501. We have also investigated how the size of the histogram bin affects the model performance for the Histogram loss. As shown in the Figure 3.2a, the results for CUB-200-2011 remain stable for the sizes equal to 0.005, 0.01, 0.02 and 0.04 (these values correspond to 400, 200, 100 and 50 bins in the histograms). In our method, distributions of similarities of training data are estimated by distributions of similarities within mini-batches. Therefore we also show results for the Histogram loss for various batch size values (Figure 3.2b). The larger batches are more preferable: for CUHK03, Recall@K for batch size equal to 256 is uniformly better than Recall@K for 128 and 64. We also observed similar behaviour for Market-1501. Additionally, we present our final results (batch size set to 256) for CUHK03 and Market-1501 in Table 3.1. For CUHK03, Rekall@K values for 5 random splits were averaged. To the best of our knowledge, these results corresponded to state-of-the-art on CUHK03 and Market-1501 at the moment of submission. To summarize the results of the comparison: the new (Histogram) loss gives the best results on the two person re-identification problems. For CUB-200-2011 and Online Products it came very close to the best loss (Binomial Deviance with $C = 25$). Interestingly, the histogram loss uniformly outperformed the triplet-based LSSS loss (2.13) [Song et al., 2016] in our experiments including two datasets from [Song et al., 2016]. Importantly, the new loss does not require to tune parameters associated with it (though we have found learning with our loss to be sensitive to the learning rate).

3.4 Conclusion

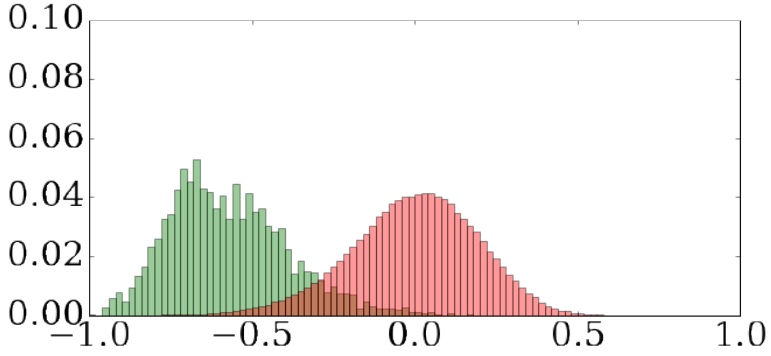
In this chapter we have suggested a new loss function for learning deep embeddings, called the Histogram loss. Like most previous losses, it is based on the idea of making the distributions of the similarities of the positive and negative pairs less overlapping. Unlike other losses used for deep embeddings, the new loss comes with virtually no parameters that need to be tuned. It also incorporates information across a large number of quadruplets formed from training samples in the mini-batch and implicitly takes into account all of such quadruplets. We have demonstrated the competitive results of the new loss on a number of datasets. In particular, the Histogram loss outperformed other losses for the person re-identification problem on CUHK03 and Market-1501 datasets. The code for Caffe [Jia et al., 2014] is available at: `https://github.com/madkn/HistogramLoss`.



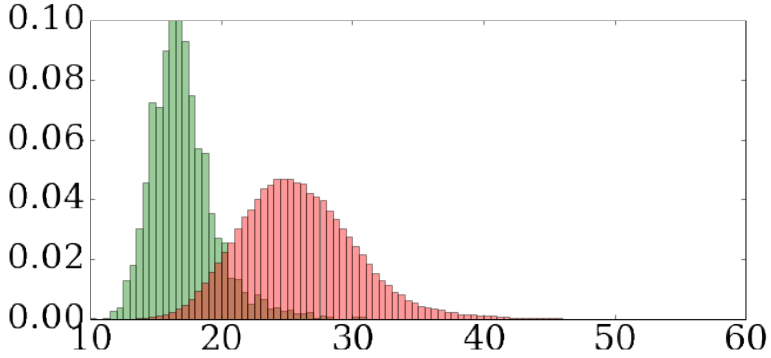
(a)



(b)



(c)



(d)

FIGURE 3.5: Histograms for positive and negative distance distributions on the CUHK03 test set for: (a) Initial state: randomly initialized net, (b) Network training with the Histogram loss, (c) same for the Binomial Deviance loss, (d) same for the LSSS loss. Red is for negative pairs, green is for positive pairs. Negative cosine distance measure is used for Histogram and Binomial Deviance losses, Euclidean distance is used for the LSSS loss. Initially the two distributions are highly overlapped. For the Histogram loss the distribution overlap is less than for the LSSS.

Acknowledgement: This research is supported by the Russian Ministry of Science and Education grant RFMEFI57914X0071.

Chapter 4

Multi-region Bilinear Convolutional Neural Networks for Person Re-Identification

4.1 Motivation

Fine-grained recognition tasks are usually characterized by small visual differences between the categories and possibly large intra-category differences. The examples of the former are different bird species of similar color and pedestrians wearing similar clothes, the latter is often due to high variations caused by such factors as pose and lighting.

The task of person re-identification shares considerable similarity with fine-grained categorization (see the example pairs in Figure 4.1), as the matching process in both cases often needs to resort to the analysis of fine texture details and parts that are hard to localize.

Bilinear convolutional neural networks have been first suggested by [Tsung-Yu Lin and Maji, 2015] for fine-grained categorization tasks. The idea of this architecture is to utilize multiplicative interactions between different features computed using two feature extractor subnetworks. The intuition behind this suggests that such factorization may allow the model to learn such functions as detection (some parts may be detected, *e.g.*, birds beaks, heads) and recognition (*e.g.*, colors and textures). When the features carrying information about parts are combined in a multiplicative way with the features carrying some texture information, the result may capture more complex concepts, *e.g.*, particular parts of particular color/texture.

The bilinear layer suggested by [Tsung-Yu Lin and Maji, 2015] is defined as follows:

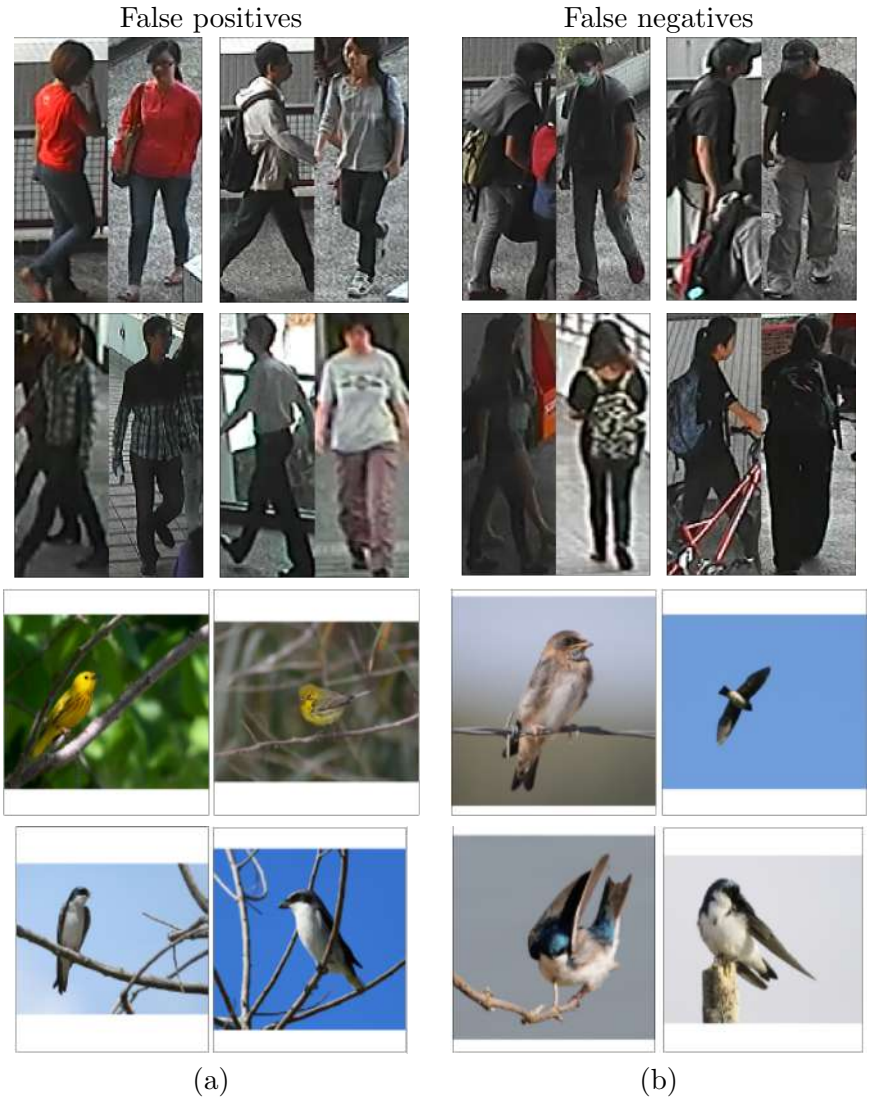


FIGURE 4.1: Difficult re-identification cases in the CUHK03 dataset and the CUB-Birds dataset for the fine-grained classification [Wah et al., 2011]. (a) – the pairs of very similar images depicting different persons/bids species (b) – the pairs of dissimilar images depicting the same person/bird species. There is a clear similarity between the challenges posed by the two tasks. Both re-identification and fine-grained classification deal with strong viewpoint variations, and often need to focus on small-scale fragments in order to distinguish subjects/classes. At the same time the re-identification task has a greater degree of alignment, and we therefore suggest a modification of the bilinear CNN exploiting the presene of such weak alignment in the re-identification case.

$$x_{bilinear} = A^T B, \tag{4.1}$$

where A and B are the outputs of the two deep feature extractors f_A and f_B (convolutional neural networks). $A \in \mathbb{R}^{S \times M}$, $B \in \mathbb{R}^{S \times N}$, S is the number of spatial locations, M and N are the number of the featuremaps that the extractors f_A and f_B produce. The result of this operation is a matrix of size $M \times N$, but is reshaped to a vector $MN \times 1$.

Another characteristic that differs Bilinear CNNs from regular models is related to the way of handling the image spatial information. Non-bilinear architectures (*e.g.*, [Simonyan and Zisserman, 2014]) apply fully-connected layers to the output of the last convolution layer to compute the final representation. In this case, each output element is computed by applying different weights to the input elements. It is easy to see that such operation may preserve some spatial information by assigning bigger weights for particular regions of the input feature maps.

Bilinear CNNs, however, rather radically discard spatial information in the process of the sum-pooling. For very deep and powerful architectures like [Simonyan and Zisserman, 2014], such spatial information may be rather irrelevant and therefore can be easily discarded. In contrast, for more shallow architectures, like those that have been recently most popular for person re-identification, the radical pooling over all spatial locations may lead to a noticeable performance drop.

Moreover, as evidenced by the experiments of [Tsung-Yu Lin and Maji, 2015], Bilinear CNNs with global pooling and powerful feature extractors are very appropriate for the tasks where pose variability may be very high, *e.g.*, bird species recognition (for example of very different poses, see the lower picture in Figure 4.1b). At the same time, the variability of geometric pose and viewpoints in re-identification problems is more restricted.

Thus the CNN architecture for person re-identification should be developed in the light of the discussed circumstances of this task, namely:

- moderate-size architectures being used for person re-identification,
- more restricted pose variation of pedestrians, compared to other fine-grained recognition problems.

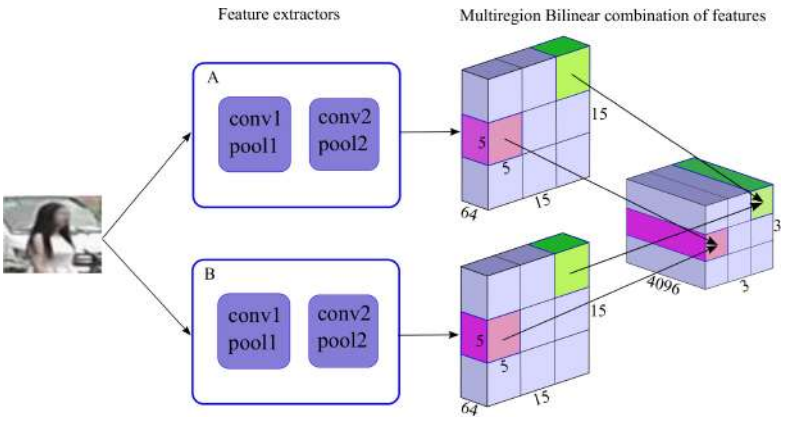
Given these preliminaries, it is appropriate to consider some intermediate variant that would preserve the features of both Bilinear and standard architectures. Indeed, such a variant, called Multi-region Bilinear CNN, is discussed later in this chapter. It can be regarded as a middle ground between the traditional CNNs and the Bilinear CNNs. In the experiments presented in this chapter, it is also shown that such a compromise achieves an optimal performance across a range of person re-identification benchmarks, while also performing favorably compared to the previous state-of-the-art. The success of such architecture confirms the promise hold by deep architectures with multiplicative interactions such as Bilinear CNNs and the Multi-region Bilinear CNNs for hard pattern recognition tasks.

It should also be mentioned, that the Bilinear CNN with global pooling has been successfully applied to person re-identification by [Suh et al., 2018] (after this work). The authors used powerful models of [Szegedy et al., 2015] and [Cao et al., 2017] (pre-trained for human pose estimation) for the recognition and detection streams correspondingly. This work presents some earlier (and lower) results, for which no explicit body part detection has been used. However, this work is the earliest to consider person re-identification a fine-grained problem and to adopt the Bilinear architecture for it.

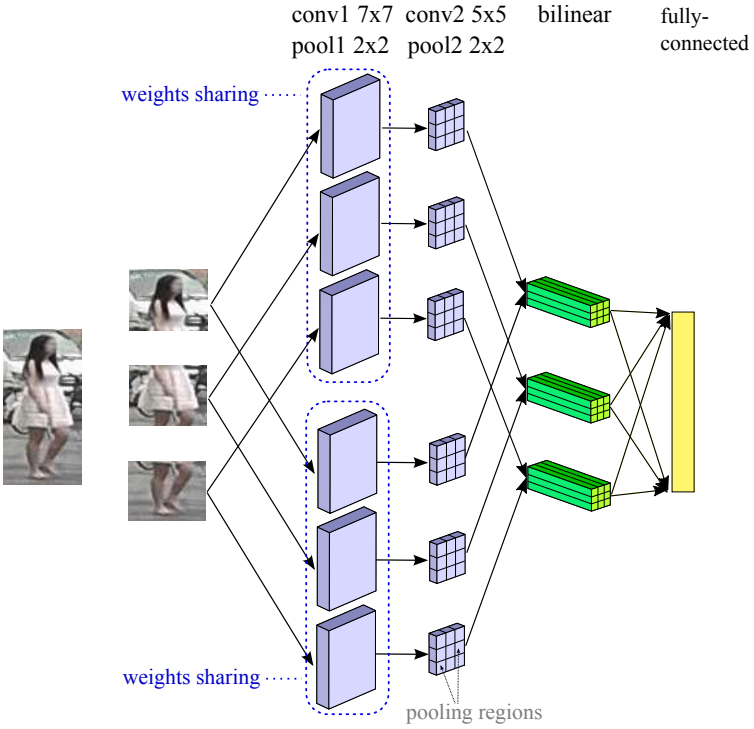
4.2 The architecture

Our solution combines the state-of-the-art method for person re-identification (Deep Metric Learning [Yi et al., 2014b], described in details in Section 1.4) and the state-of-the-art fine-grained recognition method (bilinear CNN [Tsung-Yu Lin and Maji, 2015]). Modifying the bilinear CNNs by performing multi-region pooling boosts the performance of this combination significantly. Below, we introduce the notations and discuss the components of the system in detail.

Multi-region Bilinear Model.
Bilinear CNNs are motivated by the specialized pooling operation that aggregates the correlations across maps coming from different feature extractors. The aggregation however discards all spatial information that remains in the network prior to the application of the operation. This is justified when the images lack even loose alignment (as e.g. in the case of some fine-grained classification datasets), however is sub-optimal in our case, where relatively tight bounding boxes are either manually selected or obtained using a good person detector. Thus some loose geometric alignment between images is always present. Therefore we modify bilinear layer and replace it with the *multi-region bilinear layer*, which allows us to retain some of the geometric information. Our modification is, of course, similar to many other approaches in computer vision, notably to the classical spatial pyramids of [Lazebnik et al., 2006]. In more detail, similarly to [Tsung-Yu Lin and Maji, 2015], we introduce the bilinear model for image similarity as follows:
$\mathcal{B} = (f_A, f_B, \mathcal{P}, \mathcal{S})$, where f_A and f_B are feature extractor functions (implemented as CNNs), $\mathcal{P}$ is the pooling function, $\mathcal{S}$ is the similarity function. The feature function takes an image $\mathcal{I}$ at location $\mathcal{L}$ and outputs the feature of determined dimension $\mathcal{D}$ (unlike [Tsung-Yu Lin and Maji, 2015], we use vector notation for features for simplicity): $f : \mathcal{I} \times \mathcal{L} \rightarrow \mathcal{R}^{1 \times \mathcal{D}}$. In this work, two convolutional CNNs (without fully-connected layers) serve as the two feature extractors f_A and f_B. For each of the two images in the pair at each spatial location, the outputs of two feature extractors f_A and f_B are

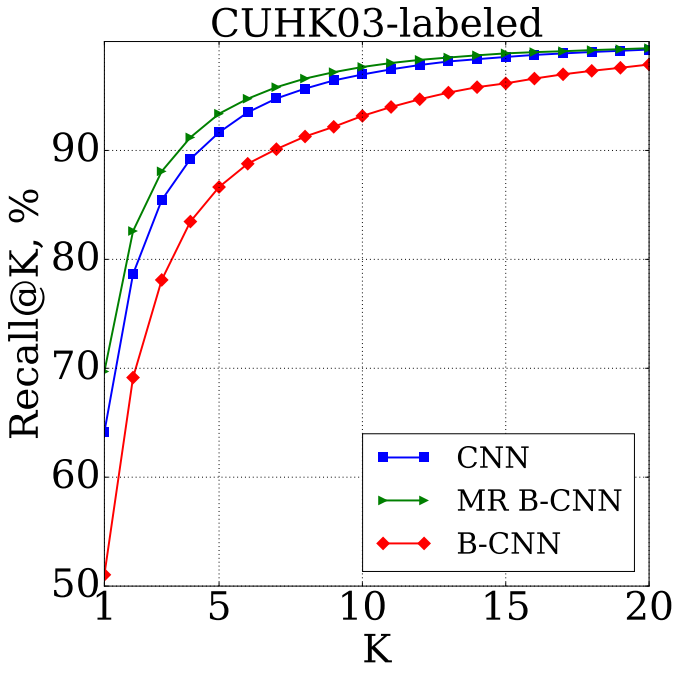


(a)

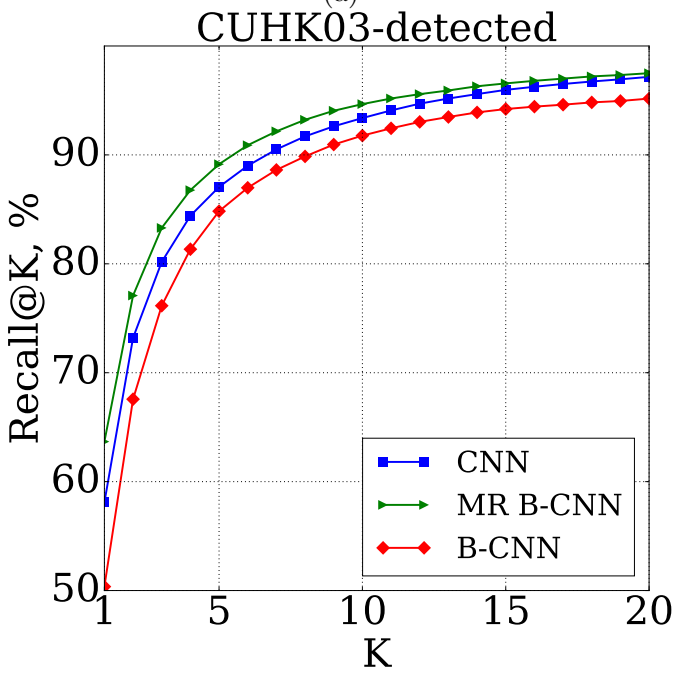


(b)

FIGURE 4.2: The proposed architecture for person re-identification: (a) - multi-region bilinear sub-network used for each of the three parts of the input image, (b) - the whole multi-region Bilinear CNN architecture that uses bilinear pooling over regions rather than the entire image. The new architecture achieves state-of-the-art performance over a range of benchmark datasets.

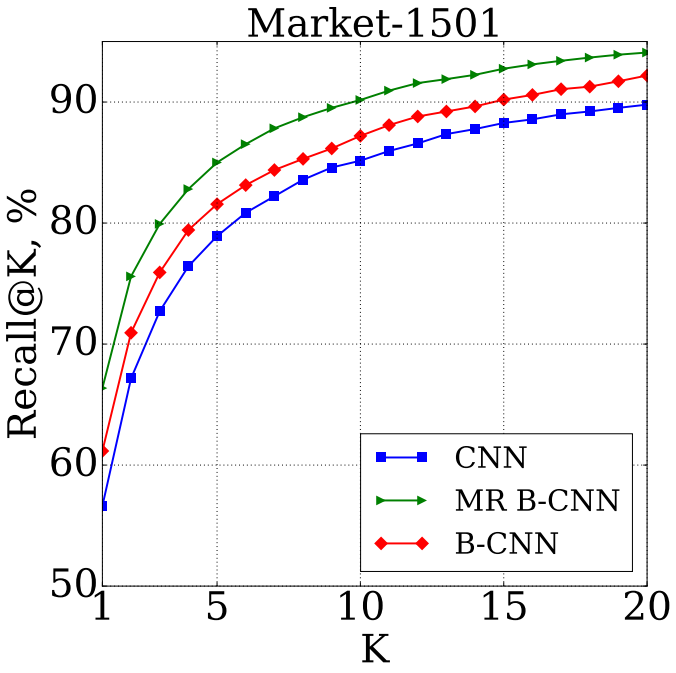


(a)

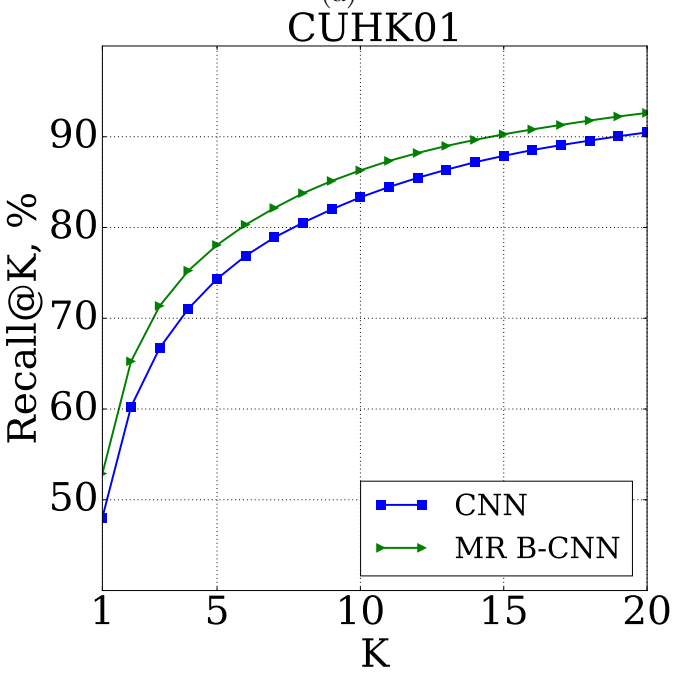


(b)

FIGURE 4.3: Recall@K results for the (a) CUHK03-labeled, (b) CUHK03-detected. MR B-CNN uniformly outperforms other architectures.



(a)



(b)

FIGURE 4.4: Recall@K results for the (a) Market-1501 (b) CUHK01, datasets. MR B-CNN uniformly outperforms other architectures.

combined using the bilinear operation [Tsung-Yu Lin and Maji, 2015]:

$$\text{bilinear}(l, im, f_A, f_B) = f_A(l, im)^T f_B(l, im), \tag{4.2}$$

where $l \in \mathcal{L}, im \in \mathcal{I}$. Using the operation (4.2), we compute the bilinear feature vector for each spatial location l of the image im. If the feature extractors f_A and the f_B output local feature vectors of size M and N correspondingly, their bilinear combination will have size $M \times N$ or $MN \times 1$, if reshaped to the column vector.

We then suggest to aggregate the obtained bilinear features by pooling across locations that belong to a predefined set of image regions: $r_1, ..., r_R$, where R is number of chosen regions. After such pooling, we get the pooled feature vector for each image region i (as opposed to the feature vector that is obtained in [Tsung-Yu Lin and Maji, 2015] for the whole image):

$$\phi_{r_i}(im) = \phi(im_{r_i}) = \sum_{l \in r_i} \text{bilinear}(l, im, f_A, f_B) \tag{4.3}$$

Finally, in order to get a descriptor for image im, we combine all region descriptors into a matrix of size $R \times MN$:

$$\phi(im) = [\phi_{r_1}(im)^T; \phi_{r_2}(im)^T; ...; \phi_{r_R}(im)^T]. \tag{4.4}$$

To pick the set of regions, in our experiments, we simply used the grid of equally-sized non-overlapping patches (note that the receptive fields of the units from different regions are still overlapping rather strongly). The scheme of Multi-region Bilinear CNN architecture is shown in figure 4.2a.

We incorporate the multi-region bilinear operation (4.4) into the convolutional architecture in the following way: instead of using one sub-network for each image part, we use two feature extractors with the same convolutional architecture described above. The outputs are combined by the multi-region bilinear operation (4.4) after the second convolution. Three bilinear outputs for each of the image parts are then concatenated and turned into a 500-dimensional image descriptor by an extra fully connected layer. The overall scheme of Multi-region Bilinear CNN net for each of the two siamese sub-networks used is this work is shown in figure 4.2b.

4.3 Experiments

Datasets and evaluation protocols.

We investigate the performance of the CNN method and its Bilinear variant (figure 4.2)

TABLE 4.1: Recall@K for the CUHK03-labeled dataset.

Method	r = 1	r = 5	r = 10	r = 20
FPNN [Li et al., 2014]	20.65	51.50	66.50	80.00
LOMO+XQDA [Liao et al., 2015]	52.20	82.23	92.14	96.25
ImrovedDeep [Ahmed et al., 2015]	54.74	86.50	93.88	98.10
ME [Paisitkriangkrai et al., 2015]	62.10	89.10	94.30	97.80
DiscrNullSpace [Zhang et al., 2016]	62.55	90.05	94.80	98.10
CNN	64.15	91.66	96.97	99.26
MR B-CNN	**69.7**	**93.37**	**98.91**	**99.39**

TABLE 4.2: Recall@K for the CUHK03-detected dataset. The new architecture (MR B-CNN) outperforms other methods.

Method	r = 1	r = 5	r = 10	r = 20
FPNN [Li et al., 2014]	19.89	50.00	64.00	78.50
ImrovedDeep [Ahmed et al., 2015]	44.96	76.01	83.47	93.15
LOMO+XQDA [Liao et al., 2015]	46.25	78.90	88.55	94.25
DiscrNullSpace [Zhang et al., 2016]	54.70	84.75	**94.80**	95.20
SiamLSTM [Varior et al., 2016b]	57.3	80.1	88.3	-
GatedSiamCNN [Varior et al., 2016a]	61.8	80.9	88.3	-
CNN	58.09	87.06	93.38	97.17
MR B-CNN	**63.67**	**89.15**	94.66	**97.5**

for three re-identification datasets: CUHK01 [Li et al., 2012], CUHK03 [Li et al., 2014] and Market-1501 [Zheng et al., 2015b]. Following [Li et al., 2014], we use Recall@K metric to report our results.

Architectures.

In the experiments, we compare the baseline CNN architecture of [Yi et al., 2014b] as one of the baselines (it is described in Section 1.4). We also evaluate the baseline Bilinear CNN (*"B-CNN"*) architecture where bilinear features are pooled over all locations for each of the three image parts. This corresponds to the formula (4.4), where whole image is used for pooling. Finally, we present the results for the Multi-region Bilinear CNN (*"MR B-CNN"*) introduced in this paper (figure 4.2).

Implementation details.

To learn the embeddings, we use the Histogram loss introduced in Chapter 3. As in [Yi et al., 2014b], we form training pairs inside each batch consisting of 128 randomly chosen training images (from all cameras). The training set is shuffled after each epoch, so the network can see many different image pairs while training. All images are resized to height 160 and width 60 pixels. Cosine similarity is used to compute the distance between a pair of image descriptors.

TABLE 4.3: Recall@K for the Market-1501 dataset. The proposed architecture (MR B-CNN) outperforms other methods.

Method	r = 1	r = 5	r = 10	mAP
DeepAttrDriven [Su et al., 2016]	39.4	-	-	19.6
DiscrNullSpace [Zhang et al., 2016]	61.02	-	-	35.68
SiamLSTM [Varior et al., 2016b]	61.60	-	-	35.31
GatedSiamCNN [Varior et al., 2016a]	65.88	-	-	39.55
CNN	56.62	78.92	85.15	32.97
MR B-CNN	**66.36**	**85.01**	**90.17**	**41.17**

TABLE 4.4: Recall@K for the CUHK01 dataset. For CNN and MR B-CNN, single-shot protocol with 486 queries was used. We include some results for this dataset, although we are not sure which protocol is used in [Zhang et al., 2016]. Other works use the same protocol as ours.

Method	r = 1	r = 5	r = 10	r = 20
ImrovedDeep [Ahmed et al., 2015]	47.53	71.60	80.25	87.45
ME [Paisitkriangkrai et al., 2015]	53.40	76.40	84.40	90.50
DiscrNullSpace [Zhang et al., 2016]	69.09	86.87	91.77	95.39
CNN	48.04	74.34	83.33	90.48
MR B-CNN	52.88	78.08	86.3	92.63

We train networks with the weight decay rate of 0.0005. The learning rate is changing according to the "step" policy, the initial learning rate is set to 10^{-4} and it is divided by ten when the performance on the validation set stops improving (which is roughly every $100,000$ iterations). The dropout layer with probability of 0.5 is inserted before the fully connected layer. The best iteration is chosen using the validation set. Following [Ahmed et al., 2015], for CUHK01 we finetune the net pretrained on CUHK03.

Variations of the Bilinear CNN architecture.

We have conducted a number of experiments with the varying pooling area for bilinear features (MR B-CNN), including full area pooling (B-CNN), on the CUHK03-labeled. Here we demonstrate results for our current MR B-CNN architecture with 5×5 pooling area, as this architecture has been found to be the most beneficial for the CUHK03 dataset. We also compare results for B-CNN architecture, where no spatial information is preserved. In Figure 4.4a and Figure 4.4b B-CNN is shown to be outperformed by other two architectures by a large margin. This result is not specific to a particular loss, as we observed the same in our preliminary experiments with the Binomial Deviance loss (2.8) [Yi et al., 2014b]. Interestingly, in our previous experiments with the Binomial Deviance loss, the simple B-CNN variant slightly outperformed baseline CNN architecture on CUHK03-detected. The MR B-CNN architecture shows uniform improvement over baseline CNN architecture on all three datasets (Figure 4.4a,b,c,d).

Comparison with the state-of-the-art methods.
To our knowledge, Multi-region Bilinear CNN networks introduced in this paper outperform previously published methods on the CUHK03 (both 'detected' and 'labeled' versions), and Market-1501 datasets. Recall@K for several rank values are shown in Table 4.1, Table 4.2 and Table 4.3 (singe query setting was used). For the Market-1501 dataset, mean average precision value is additionally shown. The results for CUHK01 are shown in Table 4.4.

4.4 Conclusion

In this chapter we demonstrated an application of new Multi-region Bilinear CNN architecture to the problem of person re-identification. Having tried different variants of the bilinear architecture, we showed that such architectures give state-of-the-art performance on larger datasets (by the time of submission for publication). In particular, Multi-region Bilinear CNN allows to retain some spatial information and to extract more complex features, while increase the number of parameters over the baseline CNN without overfitting. We have demonstrated notable gap between the performance of the Multi-region Bilinear CNN and the performance of the standard CNN [Yi et al., 2014b]. The code for Caffe [Jia et al., 2014] is available at: `https://github.com/madkn/MultiregionBilinearCNN-ReId`.

Acknowledgement: This research is supported by the Russian Ministry of Science and Education grant RFMEFI57914X0071.

FIGURE 4.5: The result of the proposed architecture on the Market-1501 dataset for a random subset of queries. The queries are shown on the left, and the remaining images in each row show the closest matches in the gallery dataset.

Chapter 5

Domain-adversarial adaptation by backpropagation

5.1 Motivation

In many practical cases, we do not have access to a sufficient amount of labeled training data for supervised learning. This problem is especially critical for learning deep neural networks that often incorporate millions of parameters. However, it is often the case that the training data are available, and the labeling corresponds to the task of interest, but the data differ from those that our predictor will encounter during test time. Such difference between the training data (source) and possible test data (target) is referred to as a *domain shift* or *covariate shift*. In more detail, this means that the source and target data share the same feature space, and only their marginal distributions are assumed to be different.

One important example of the described situation is training on synthetic images that can be generated automatically (*e.g.*, by 3D rendering systems). This is a particularly practical use case, as collecting the necessary amount of analogous real data and its labeling may be very time-consuming. In this way, the ability to overcome the domain shift between the available synthetic and target data is crucial for the final performance of the predictor.

Although there exist many powerful methods for domain adaptation Huang et al. [2007], Pan et al. [2008, 2011], Baktashmotlagh et al. [2013], Gong et al. [2012, 2013], they most often work for the fixed feature representations and therefore are limited by these representations.

Several works have also approached domain adaptation in the framework of deep learning. Glorot et al. [2011] and Chopra et al. [2013] train the denoising autoencoders on a mixture of data from the two domains to get the domain-invariant feature extractors. The task-specific training is performed as the second step (although, at this step, Chopra et al. [2013] finetunes both task-specific neural network and the pre-trained feature extractor using the task-specific loss). In contrast to Glorot et al. [2011] and Chopra et al. [2013], Long et al. [2015] and Tzeng et al. [2014] apply a special loss function (based on Maximum Mean Discrepancy) to pull the two domains closer. Such an objective can be optimized simultaneously with the task-specific loss. This allows to learn the representations that are useful for the task of interest and at the same time are domain-invariant.

A new way of deep domain adaptation is suggested in this chapter. Like in the works of Long et al. [2015] and Tzeng et al. [2014], the goal is to incorporate domain adaptation into the process of training a classification neural network, so that the learned deep representations are

- discriminative (*i.e.*, useful for classification),
- domain-invariant (*i.e.*, the feature distributions are close for the two domains)

. This is achieved by jointly optimizing the two deep discriminative classifiers:

- the task-specific label predictor,
- the domain classifier that discriminates between the source and the target domains.

The parameters of these two classifiers are optimized in order to minimize their error on the training set. What provides discriminativeness and domain-invariance of the learned features is that these features are optimized simultaneously to

- minimize the error of the task-specific classifier,
- maximize the error of the domain classifier.

Such combination of objectives can be optimized for feed-forward networks using standard backpropagation algorithm based on stochastic gradient descent. To perform a domain-adversarial training of feature representation, the gradient reversal layer is introduced. No labeling is needed for the target domain, so the approach is suitable for *unsupervised domain adaptation*.

It should be noted that an idea related to ours is described in [Goodfellow et al., 2014]. While their goal is quite different (building generative deep networks that can synthesize samples), the way they measure and minimize the discrepancy between the distribution of the training data and the distribution of the synthesized data is very similar to the way our architecture measures and minimizes the discrepancy between feature distributions for the two domains.

First, the described approach is evaluated for an image classification task. This chapter includes the results only for digit classification (on the MNIST dataset [LeCun et al., 1998]). The extensive evaluation for many other classifications tasks can be found in the corresponding publication [Ganin et al. [2016]].

Domain-adversarial training is also demonstrated to be applicable to person re-identification. Several publicly available datasets are considered as different domains. To adapt the described approach to person re-identification, we consider a *descriptor predictor* trained with a Siamese-like loss instead of the label predictor trained with a classification loss. In a series of experiments, we demonstrate that domain-adversarial learning can improve cross-dataset re-identification considerably.

5.2 Domain Adaptation

We consider classification tasks where X is the input space and $Y = \{0, 1, \ldots, L-1\}$ is the set of L possible labels. Moreover, we have two different distributions over $X \times Y$, called the *source domain* $\mathcal{D}_\mathrm{S}$ and the *target domain* $\mathcal{D}_\mathrm{T}$. An *unsupervised domain adaptation* learning algorithm is then provided with a *labeled source sample* S drawn *i.i.d.* from $\mathcal{D}_\mathrm{S}$, and an *unlabeled target sample* T drawn *i.i.d.* from $\mathcal{D}_\mathrm{T}^X$, where $\mathcal{D}_\mathrm{T}^X$ is the marginal distribution of $\mathcal{D}_\mathrm{T}$ over X.

$$S = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n} \sim (\mathcal{D}_\mathrm{S})^n\,; \quad T = \{\mathbf{x}_i\}_{i=n+1}^{N} \sim (\mathcal{D}_\mathrm{T}^X)^{n'},$$

with $N = n + n'$ being the total number of samples. The goal of the learning algorithm is to build a classifier $\eta : X \to Y$ with a low *target risk*

$$R_{\mathcal{D}_\mathrm{T}}(\eta) \;= \Pr_{(\mathbf{x},y)\sim\mathcal{D}_\mathrm{T}} \Big(\eta(\mathbf{x}) \neq y\Big)\,,$$

while having no information about the labels of $\mathcal{D}_\mathrm{T}$.

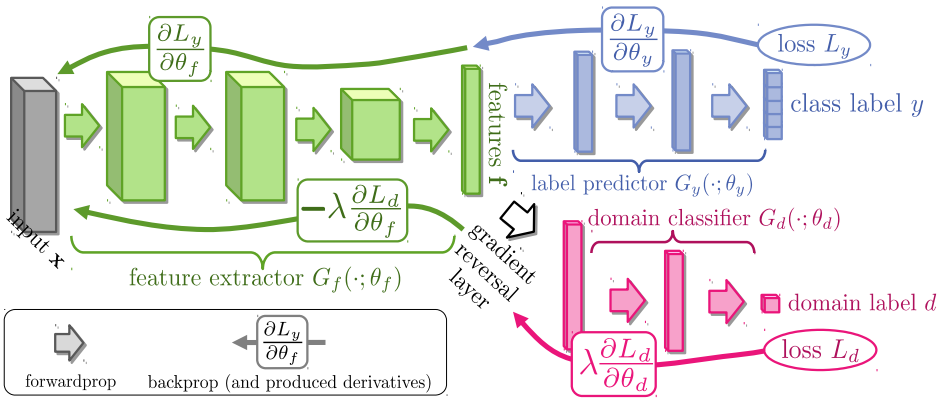


FIGURE 5.1: The **proposed architecture** includes a deep *feature extractor* (green) and a deep *label predictor* (blue), which together form a standard feed-forward architecture. Unsupervised domain adaptation is achieved by adding a *domain classifier* (red) connected to the feature extractor via a *gradient reversal layer* that multiplies the gradient by a certain negative constant during the backpropagation-based training. Otherwise, the training proceeds standardly and minimizes the label prediction loss (for source examples) and the domain classification loss (for all samples). Gradient reversal ensures that the feature distributions over the two domains are made similar (as indistinguishable as possible for the domain classifier), thus resulting in the domain-invariant features.

5.3 Domain-Adversarial Neural Networks (DANN)

To learn a model that can generalize well from one domain to another, we ensure that the internal representation of the neural network contains no discriminative information about the origin of the input (source or target), while preserving a low risk on the source (labeled) examples.

In this section, we detail the proposed approach for incorporating a "domain adaptation component" to neural networks.

For illustration purposes, we've so far focused on the case of a single hidden layer DANN. However, it is straightforward to generalize to other sophisticated architectures, which might be more appropriate for the data at hand. For example, deep convolutional neural networks are well known for being state-of-the-art models for learning discriminative features of images [Krizhevsky et al., 2012].

Let us now use a more general notation for the different components of DANN. Namely, let $G_f(\cdot;\theta_f)$ be the D-dimensional neural network feature extractor, with parameters θ_f. Also, let $G_y(\cdot;\theta_y)$ be the part of DANN that computes the network's label prediction output layer, with parameters θ_y, while $G_d(\cdot;\theta_d)$ now corresponds to the computation of the domain prediction output of the network, with parameters θ_d.

We will note the prediction loss and the domain loss respectively by

$$\begin{aligned}\mathcal{L}_y^i(\theta_f, \theta_y) &= \mathcal{L}_y\big(G_y(G_f(\mathbf{x}_i; \theta_f); \theta_y), y_i\big)\,, \\ \mathcal{L}_d^i(\theta_f, \theta_d) &= \mathcal{L}_d\big(G_d(G_f(\mathbf{x}_i; \theta_f); \theta_d), d_i\big)\,.\end{aligned}$$

Training DANN then consists in optimizing

$$E(\theta_f, \theta_y, \theta_d) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_y^i(\theta_f, \theta_y) - \lambda\left(\frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_d^i(\theta_f, \theta_d) + \frac{1}{n'}\sum_{i=n+1}^{N} \mathcal{L}_d^i(\theta_f, \theta_d)\right), \tag{5.1}$$

by finding the saddle point $\hat{\theta}_f, \hat{\theta}_y, \hat{\theta}_d$ such that

$$(\hat{\theta}_f, \hat{\theta}_y) = \operatorname*{argmin}_{\theta_f, \theta_y} E(\theta_f, \theta_y, \hat{\theta}_d)\,, \tag{5.2}$$

$$\hat{\theta}_d = \operatorname*{argmax}_{\theta_d} E(\hat{\theta}_f, \hat{\theta}_y, \theta_d)\,. \tag{5.3}$$

As suggested previously, a saddle point defined by Equations (5.2-5.3) can be found as a stationary point of the following gradient updates:

$$\theta_f \longleftarrow \theta_f - \mu\left(\frac{\partial \mathcal{L}_y^i}{\partial \theta_f} - \lambda\frac{\partial \mathcal{L}_d^i}{\partial \theta_f}\right), \tag{5.4}$$

$$\theta_y \longleftarrow \theta_y - \mu\frac{\partial \mathcal{L}_y^i}{\partial \theta_y}\,, \tag{5.5}$$

$$\theta_d \longleftarrow \theta_d - \mu\lambda\frac{\partial \mathcal{L}_d^i}{\partial \theta_d}\,, \tag{5.6}$$

where μ is the learning rate. We use stochastic estimates of these gradients, by sampling examples from the dataset.

The updates of Equations (5.4-5.6) are very similar to stochastic gradient descent (SGD) updates for a feed-forward deep model that comprises feature extractor fed into the label predictor and into the domain classifier (with loss weighted by λ). The only difference is that in (5.4), the gradients from the class and domain predictors are subtracted, instead of being summed (the difference is important, as otherwise SGD would try to make features dissimilar across domains in order to minimize the domain classification loss). Since SGD, and its many variants, is the main learning algorithm implemented in most libraries for deep learning, it would be convenient to frame an implementation of our stochastic saddle point procedure as SGD.

Fortunately, such a reduction can be accomplished by introducing a special *gradient reversal layer* (GRL), defined as follows. The gradient reversal layer has no parameters

associated with it. During the forward propagation, the GRL acts as an identity transformation. During the backpropagation however, the GRL takes the gradient from the subsequent level and changes its sign, *i.e.*,multiplies it by -1, before passing it to the preceding layer. Implementing such a layer using existing object-oriented packages for deep learning is simple, requiring only to define procedures for the forward propagation (identity transformation), and backpropagation (multiplying by -1). The layer requires no parameter update.

The GRL as defined above is inserted between the feature extractor G_f and the domain classifier G_d, resulting in the architecture depicted in Figure 5.1. As the backpropagation process passes through the GRL, the partial derivatives of the loss that is downstream the GRL (*i.e.*,, $\mathcal{L}_d$) w.r.t. the layer parameters that are upstream the GRL (*i.e.*,, θ_f) get multiplied by -1, *i.e.*,, $\frac{\partial \mathcal{L}_d}{\partial \theta_f}$ is effectively replaced with $-\frac{\partial \mathcal{L}_d}{\partial \theta_f}$. Therefore, running SGD in the resulting model implements the updates of Equations (5.4-5.6) and converges to a saddle point of Equation (5.1).

Mathematically, we can formally treat the gradient reversal layer as a "pseudo-function" $\mathcal{R}(\mathbf{x})$ defined by two (incompatible) equations describing its forward and backpropagation behaviour:

$$\mathcal{R}(\mathbf{x}) = \mathbf{x}\,, \tag{5.7}$$

$$\frac{d\mathcal{R}}{d\mathbf{x}} = -\mathbf{I}\,, \tag{5.8}$$

where $\mathbf{I}$ is an identity matrix. We can then define the objective "pseudo-function" of $(\theta_f, \theta_y, \theta_d)$ that is being optimized by the stochastic gradient descent within our method:

$$\begin{aligned}\tilde{E}(\theta_f, \theta_y, \theta_d) = &\frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_y\left(G_y(G_f(\mathbf{x}_i;\theta_f);\theta_y), y_i\right) \\ &- \lambda\left(\frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_d\left(G_d(\mathcal{R}(G_f(\mathbf{x}_i;\theta_f));\theta_d), d_i\right) + \frac{1}{n'}\sum_{i=n+1}^{N} \mathcal{L}_d\left(G_d(\mathcal{R}(G_f(\mathbf{x}_i;\theta_f));\theta_d), d_i\right)\right).\end{aligned} \tag{5.9}$$

Running updates (5.4-5.6) can then be implemented as doing SGD for (5.9) and leads to the emergence of features that are domain-invariant and discriminative at the same time. After the learning, the label predictor $G_y(G_f(\mathbf{x};\theta_f);\theta_y)$ can be used to predict labels for samples from the target domain (as well as from the source domain).

Figure 5.2: Examples of domain pairs used in the experiments. See Section 5.4 for details.

Method	Source	MNIST
	Target	MNIST-M
Source only		.5225
SA [Fernando et al., 2013]		.5690 (4.1%)
DANN		**.7666** (52.9%)

Table 5.1: Classification accuracies for digit image classifications for MNIST and MNIST-M. The first row corresponds to the lower performance bound (*i.e.*, if no adaptation is performed). The last row corresponds to training on the target domain data with known class labels (upper bound on the DA performance). For each of the two DA methods [ours and Fernando et al., 2013] we show how much of the gap between the lower and the upper bounds was covered (in brackets). For all five cases, our approach outperforms Fernando et al. [2013] considerably, and covers a big portion of the gap

5.4 Experiments

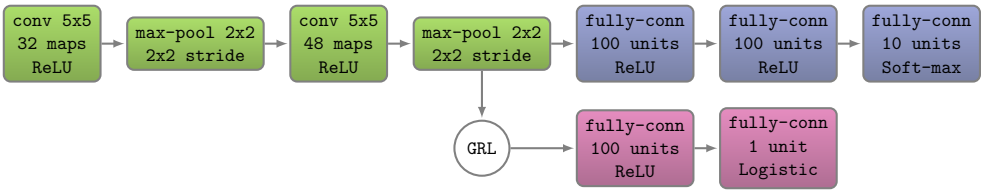
Experiments with deep networks on image classification

Baselines

The following baselines are evaluated in the experiments of this subsection.

The *source-only* model is trained without consideration for target-domain data (no domain classifier branch included into the network).

In addition, we compare our approach against the recently proposed unsupervised DA method based on *subspace alignment (SA)* [Fernando et al., 2013], which is simple to setup and test on new datasets. To boost the performance of this baseline, we pick its most important free parameter (the number of principal components) from the range $\{2, \ldots, 60\}$, so that the test performance on the target domain is maximized. To apply SA in our setting, we train a source-only model and then consider the activations of the last hidden layer in the label predictor (before the final linear classifier) as descriptors/features, and learn the mapping between the source and the target domains [Fernando et al., 2013].



(A) MNIST architecture; inspired by the classical LeNet-5 [LeCun et al., 1998].

FIGURE 5.3: CNN architecture used in experiments for the MNIST dataset. Boxes correspond to transformations applied to the data. Color-coding is the same as in Figure 5.1.

Since the SA baseline requires training a new classifier after adapting the features, and in order to put all the compared settings on an equal footing, we retrain the last layer of the label predictor using a standard linear SVM [Fan et al., 2008] for all four considered methods (including ours; the performance on the target domain remains approximately the same after the retraining).

CNN architectures and Training Procedure

In general, we compose feature extractor from two convolutional layers, picking their exact configurations from previous works. For MNIST, where we used two fully-connected layers ($x \to 100 \to 2$). Admittedly these choices for domain classifier are arbitrary, and better adaptation performance might be attained if this part of the architecture is tuned.

For the loss functions, we set $\mathcal{L}_y$ and $\mathcal{L}_d$ to be the logistic regression loss and the binomial cross-entropy respectively.

The learning rate is adjusted during the stochastic gradient descent using the following formula:

$$\mu_p = \frac{\mu_0}{(1 + \alpha \cdot p)^\beta} \,,$$

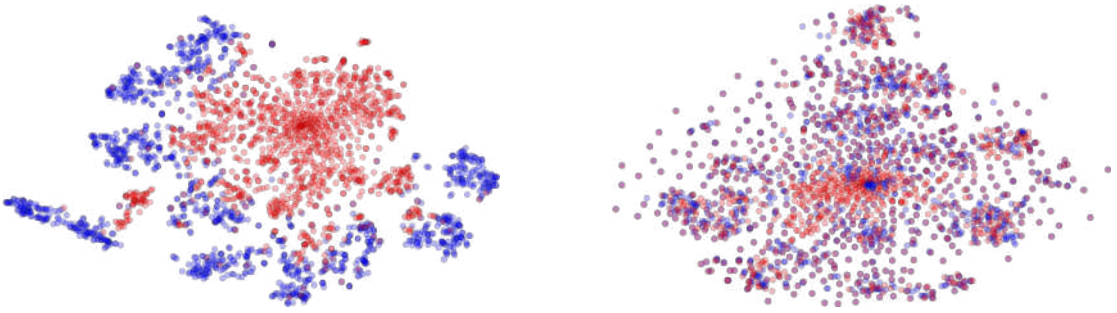
where p is the training progress linearly changing from 0 to 1, $\mu_0 = 0.01$, $\alpha = 10$ and $\beta = 0.75$ (the schedule was optimized to promote convergence and low error on the *source* domain). A momentum term of 0.9 is also used.

The domain adaptation parameter λ is initiated at 0 and is gradually changed to 1 using the following schedule:

$$\lambda_p = \frac{2}{1 + \exp(-\gamma \cdot p)} - 1 \,,$$

where γ was set to 10 in all experiments (the schedule was not optimized/tweaked). This strategy allows the domain classifier to be less sensitive to noisy signal at the early stages of the training procedure. Note however that these λ_p were used only for updating the

MNIST → MNIST-M: top feature extractor layer



(A) Non-adapted (B) Adapted

FIGURE 5.4: The effect of adaptation on the distribution of the extracted features (best viewed in color). The figure shows t-SNE [Maaten and Hinton, 2008] visualizations of the CNN's activations **(a)** in case when no adaptation was performed and **(b)** in case when our adaptation procedure was incorporated into training. *Blue* points correspond to the source domain examples, while *red* ones correspond to the target domain. In all cases, the adaptation in our method makes the two distributions of features much closer.

feature extractor component G_f. For updating the *domain classification* component, we used a fixed $\lambda = 1$, to ensure that the latter trains as fast as the *label predictor* G_y.[1]

Finally, note that the model is trained on 128-sized batches (images are preprocessed by the mean subtraction). A half of each batch is populated by the samples from the source domain (with known labels), the rest constitutes the target domain (with labels not revealed to the algorithms except for the train-on-target baseline).

Visualizations

We use t-SNE [Maaten and Hinton, 2008] projection to visualize feature distributions at different points of the network, while color-coding the domains (Figure 5.4). There is a strong correspondence between the success of the adaptation in terms of the classification accuracy for the target domain, and the overlap between the domain distributions in such visualizations.

Results

We now discuss the experimental settings and the results. In each case, we train on the source dataset and test on a different target domain dataset, with considerable shifts between domains (see Figure 5.2). The results are summarized in Table 5.1.

MNIST → MNIST-M

Our first experiment deals with the MNIST dataset [LeCun et al., 1998] (source). In

[1] Equivalently, one can use the same λ_p for both feature extractor and domain classification components, but use a learning rate of μ/λ_p for the latter.

order to obtain the target domain (MNIST-M) we blend digits from the original set over patches randomly extracted from color photos from BSDS500 [Arbelaez et al., 2010]. This operation is formally defined for two images I^1, I^2 as $I^{out}_{ijk} = |I^1_{ijk} - I^2_{ijk}|$, where i, j are the coordinates of a pixel and k is a channel index. In other words, an output sample is produced by taking a patch from a photo and inverting its pixels at positions corresponding to the pixels of a digit. For a human the classification task becomes only slightly harder compared to the original dataset (the digits are still clearly distinguishable) whereas for a CNN trained on MNIST this domain is quite distinct, as the background and the strokes are no longer constant. Consequently, the source-only model performs poorly. Our approach succeeded at aligning feature distributions (Figure 5.4), which led to successful adaptation results (considering that the adaptation is unsupervised). At the same time, the improvement over source-only model achieved by subspace alignment (SA) [Fernando et al., 2013] is quite modest, thus highlighting the difficulty of the adaptation task.

Experiments with deep image descriptors for re-identification

Datasets

For cross-domain experiments, we use PRID [Hirzer et al., 2011], VIPeR [Gray et al., 2007], CUHK02 [Li and Wang, 2013] as target datasets for our experiments. The *PRID* dataset exists in two versions, we use a single-shot variant. We use all images in the first pair of cameras of CUHK02 instead of choosing one image of a person from each camera view. We refer to the subset of this dataset that includes the first pair of cameras only as *CUHK02/p1* (as most papers use this subset). We also performed two experiments with all images of the whole CUHK02 dataset as source domain and VIPeR and PRID datasets as target domains similar to [Yi et al., 2014a].

We perform extensive experiments for various pairs of datasets, where one dataset serves as a source domain, *i.e.*,, it is used to train a descriptor mapping in a supervised way with known correspondences between probe and gallery images. The second dataset is used as a target domain, so that images from that dataset are used without probe-gallery correspondence.

In more detail, CUHK02/p1 is used for experiments when CUHK02 serves as a target domain and two settings ("whole CUHK02" and CUHK02/p1) are used for experiments when CUHK02 serves as a source domain. Given PRID as a target dataset, we randomly choose 100 persons appearing in both camera views as training set. The images of the other 100 persons from camera A are used as probe, all images from camera B excluding those used in training (649 in total) are used as gallery at test time. For VIPeR, we use

random 316 persons for training and all others for testing. For CUHK02, 971 persons are split into 485 for training and 486 for testing.

We use all images in the first pair of cameras of CUHK02 instead of choosing one image of a person from each camera view. We also performed two experiments with all images of the whole CUHK02 dataset as source domain and VIPeR and PRID datasets as target domains as in the original paper [Yi et al., 2014a].

Following Yi et al. [2014a], we augmented our data with mirror images, and during test time we calculate similarity score between two images as the mean of the four scores corresponding to different flips of the two compared images. In case of CUHK02, where there are four images (including mirror images) for each of the two camera views for each person, all 16 combinations' scores are averaged.

CNN architectures and Training Procedure

In our experiments, we use DML siamese architecture [Yi et al., 2014a] described in Section 1.4.

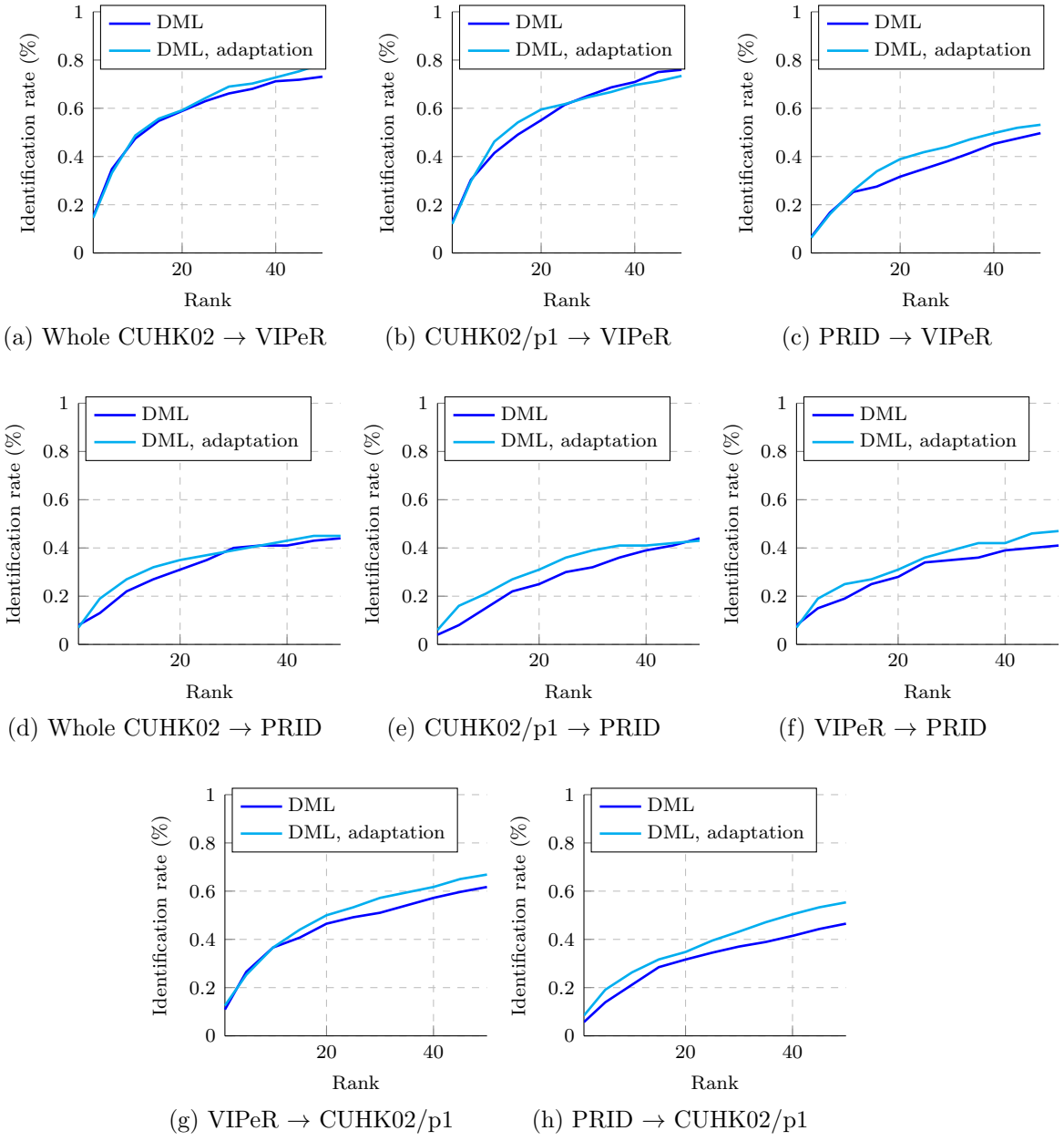
To perform domain-adversarial training, we construct a DANN architecture. The feature extractor includes the two convolutional layers (followed by max-pooling and ReLU) discussed above. The label predictor in this case is replaced with *descriptor predictor* that includes one fully-connected layer. The domain classifier includes two fully-connected layers with 500 units in the intermediate representation ($x \to 500 \to 1$).

For the verification loss function in the descriptor predictor we used Binomial Deviance loss (2.8), also used in Yi et al. [2014a] with similar parameters: $\alpha = 2$, $\beta = 0.5$, $c = 2$ (the asymmetric cost parameter for negative pairs). The domain classifier is trained with logistic loss as in Section 6.3.

We used learning rate fixed to 0.001 and momentum of 0.9. The schedule of adaptation similar to the one described in Section 6.3 was used. We also inserted dropout layer with rate 0.5 after the concatenation of outputs of the second max-pooling layer. 128-sized batches were used for source data and 128-sized batches for target data.

Results

Figure 5.5 shows results in the form of Rekall@K metric for eight pairs of datasets. Depending on the hardness of the annotation problem we trained either for 50,000 iterations (CUHK02/p1 $\to$ VIPeR, VIPeR $\to$ CUHK02/p1, PRID $\to$ VIPeR) or for 20,000 iterations (the other five pairs).



(a) Whole CUHK02 → VIPeR (b) CUHK02/p1 → VIPeR (c) PRID → VIPeR

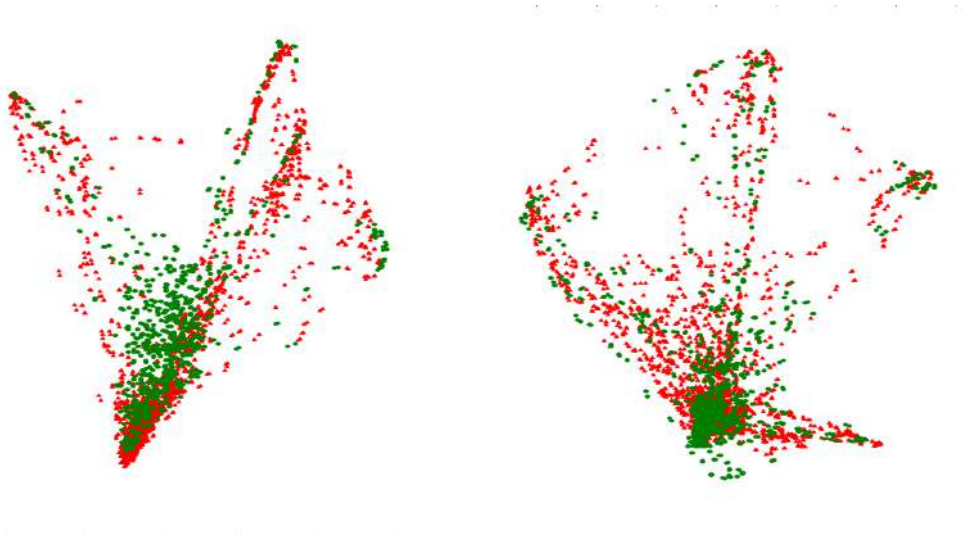
(d) Whole CUHK02 → PRID (e) CUHK02/p1 → PRID (f) VIPeR → PRID

(g) VIPeR → CUHK02/p1 (h) PRID → CUHK02/p1

FIGURE 5.5: Results on VIPeR, PRID and CUHK02/p1 with and without domain-adversarial learning. Across the eight domain pairs domain-adversarial learning improves re-identification accuracy. For some domain pairs the improvement is considerable.

After the sufficient number of iterations, domain-adversarial training consistently improves the performance of re-identification. For the pairs that involve PRID dataset, which is more dissimilar to the other two datasets, the improvement is considerable. Overall, this demonstrates the applicability of the domain-adversarial learning beyond classification problems.

Figure 5.6 further demonstrates the effect of adaptation on the distributions of the



(a) DML (b) DML, adaptation

FIGURE 5.6: The effect of adaptation shown by t-SNE visualizations of source and target domains descriptors in a VIPeR → CUHK02/p1 experiment pair. VIPeR is depicted with *green* and CUHK02/p1 - with *red*. As in the image classification case, domain-adversarial learning ensures a closer match between the source and the target distributions.

learned descriptors in the source and in target sets in VIPeR → CUHK02/p1 experiments, where domain adversarial learning once again achieves better intermixing of the two domains.

5.5 Conclusion

The chapter describes a new approach to domain adaptation of feed-forward neural networks, which allows large-scale training based on large amount of annotated data in the source domain and large amount of unannotated data in the target domain. Similarly to many previous shallow and deep DA techniques, the adaptation is achieved through aligning the distributions of features across the two domains. However, unlike previous approaches, the alignment is accomplished through standard backpropagation training.

The main idea behind DANN is to enjoin the network hidden layer to learn a representation which is predictive of the source example labels, but uninformative about the domain of the input (source or target).

A convenient aspect of our approach is that the domain adaptation component can be added to almost any neural network architecture that is trainable with backpropagation. Towards this end, we have demonstrated experimentally that the approach is not confined to classification tasks but can be used in other feed-forward architectures, e.g.

for descriptor learning for person re-identification. The full set of experiments, including those for classification, is presented in the corresponding publication [Ganin et al., 2016].

Chapter 6

Getting Off the Internet: Practical Domain Adaptation for Face Recognition

6.1 Motivation

For face recognition, one important cross-domain scenario is related to using powerful models pretrained on professional photographs for surveillance data. As it has been already mentioned in Section 1.1.2, large-scale training datasets for face recognition most often include high-quality images that are very different from those captured by surveillance systems. In person re-identification, the domain difference comes mainly from the illumination condition variations between the camera sets (if the difference of the camera positions is put aside). In contrast, surveillance face recognition implies a complex domain shift caused by a combination of low resolution, compression and illumination conditions.

In particular, in this chapter we study the unsupervised domain adaptation scenario, where face recognition is trained using an annotated Internet face dataset and an unannotated dataset of faces collected from a surveillance camera network with low image quality. We mostly focus on the recent class of methods that consider domain-adaptation at the image level. We thus investigate how image transformation achieved with recent unsupervised image transformation techniques such as CycleGAN [Zhu et al., 2017] can be used for face recognition under strong domain shifts.

We compare and evaluate several strategies, such as transferring test data to the Internet image domain, transferring training data to the target domain followed by retraining the

network. As a baseline, we also compare to adversarial domain adaptation at feature level described in Chapter 5.

Our comparison suggests that image transformation (without explicit modeling of separate degradation factors) can be used for unsupervised domain adaptation of face recognition. We, however, demonstrate that special care needs to be taken in order to make such domain adaptation work better than baselines, and come up with practical suggestions on how such improvement can be achieved.

6.2 Evaluated approaches

Face recognition for the low-quality image domain

In this work we consider and compare two main approaches to face recognition for surveillance data: 1) restoration-based approach and 2) domain adaptation of existing face recognition neural networks.

We consider two facial image domains:

- domain $T : \{X_i^T\}_{i=0}^{N_T}$ that includes low-quality facial images X_i^T captured using surveillance cameras. Usually, there are no identity labels provided, as assigning identity labels is quite challenging and may not even be feasible.
- domain $S : \{(X_i^S, Y_i^S)\}_{i=0}^{N_S}$ that includes facial images X_i^S harvested from the Internet. These images are usually of higher quality and are taken in good lighting conditions. We assume that the data in this domain are supplied with identity labels Y_i.

According to the available labeling, we can consider two different pipelines for building face recognition systems for surveillance data. The first option is the restoration-based approach when we use transform $F^{T \to S} : T \longrightarrow S$ as a face restoration method and then apply existing recognition neural network R^S that is pre-trained on images from the domain S. The second option is to use the transform $F^{S \to T} : S \longrightarrow T$ to transfer the large collections of labeled training data to the target domain of surveillance images. In this scenario, we retrain the existing face recognition networks resulting in the new adapted model R^T.

More formally, we consider the following two pipelines for face recognition in the domain T. We denote d^T and d^S the descriptors produced by the domain-specific face recognition models R^T and R^S. These descriptors may be used e.g. to identify matching and non-matching faces based on the distances between them.

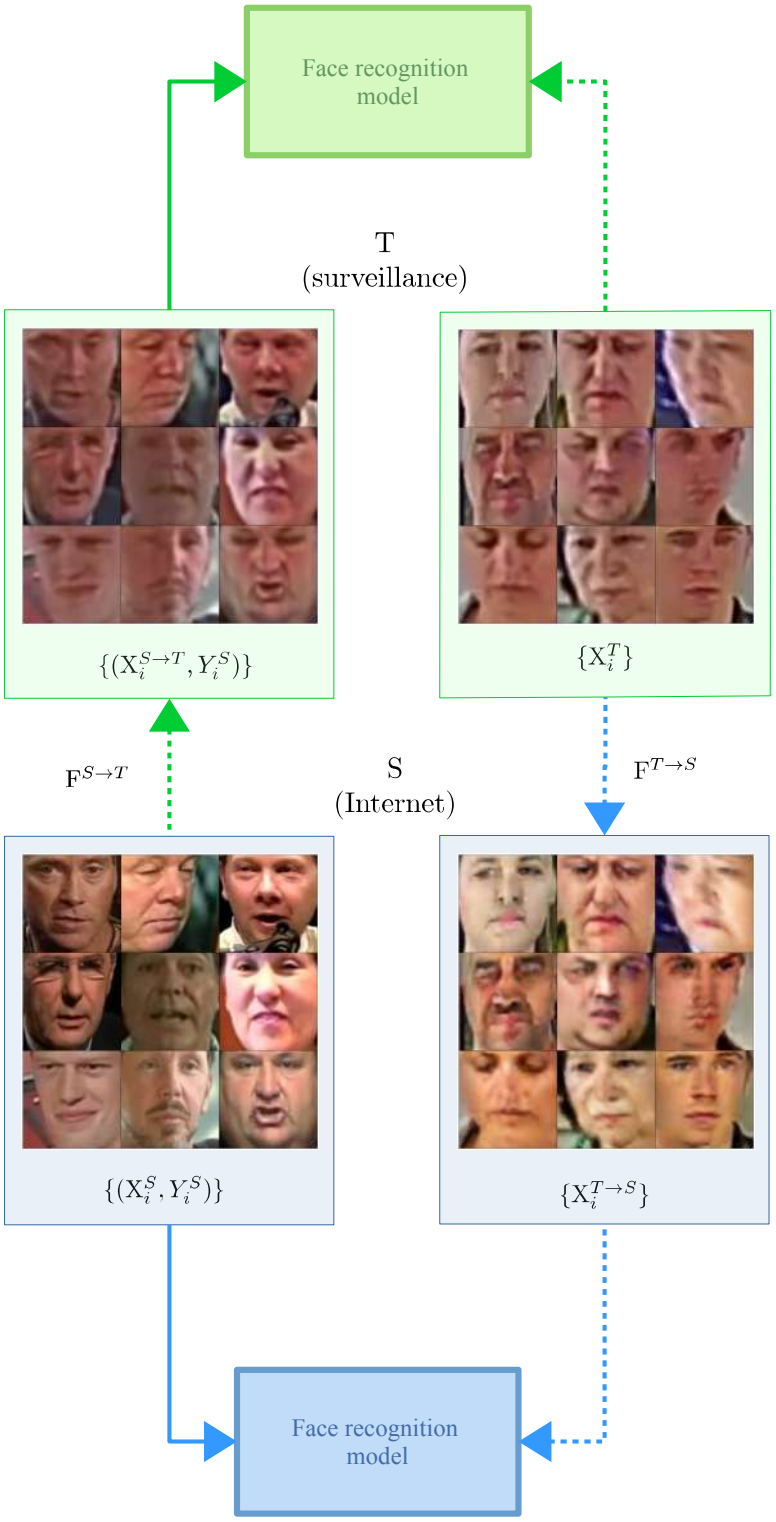


FIGURE 6.1: The overall scheme of the two possible approaches to the face recognition problem considered in our work. Surveillance and Internet image domains are denoted with green and blue rectangles correspondingly (the data examples are taken from our surveillance dataset and the YouTube faces dataset). The *face restoration* approach (blue lines) transfers the surveillance data images to the Internet domain using the transform $F^{T\to S}$. It then uses the "blue" face recognition model trained on annotated internet images to compute descriptors for the transfered images. Meanwhile, the *domain adaptation* approach (green lines) transfers the annotated internet data to the surveillance domain, and then uses the transfered data to train the "green" face recognition model, which is then applied to unannotated surveillance images. Our work evaluates and compares several variants of both approaches.

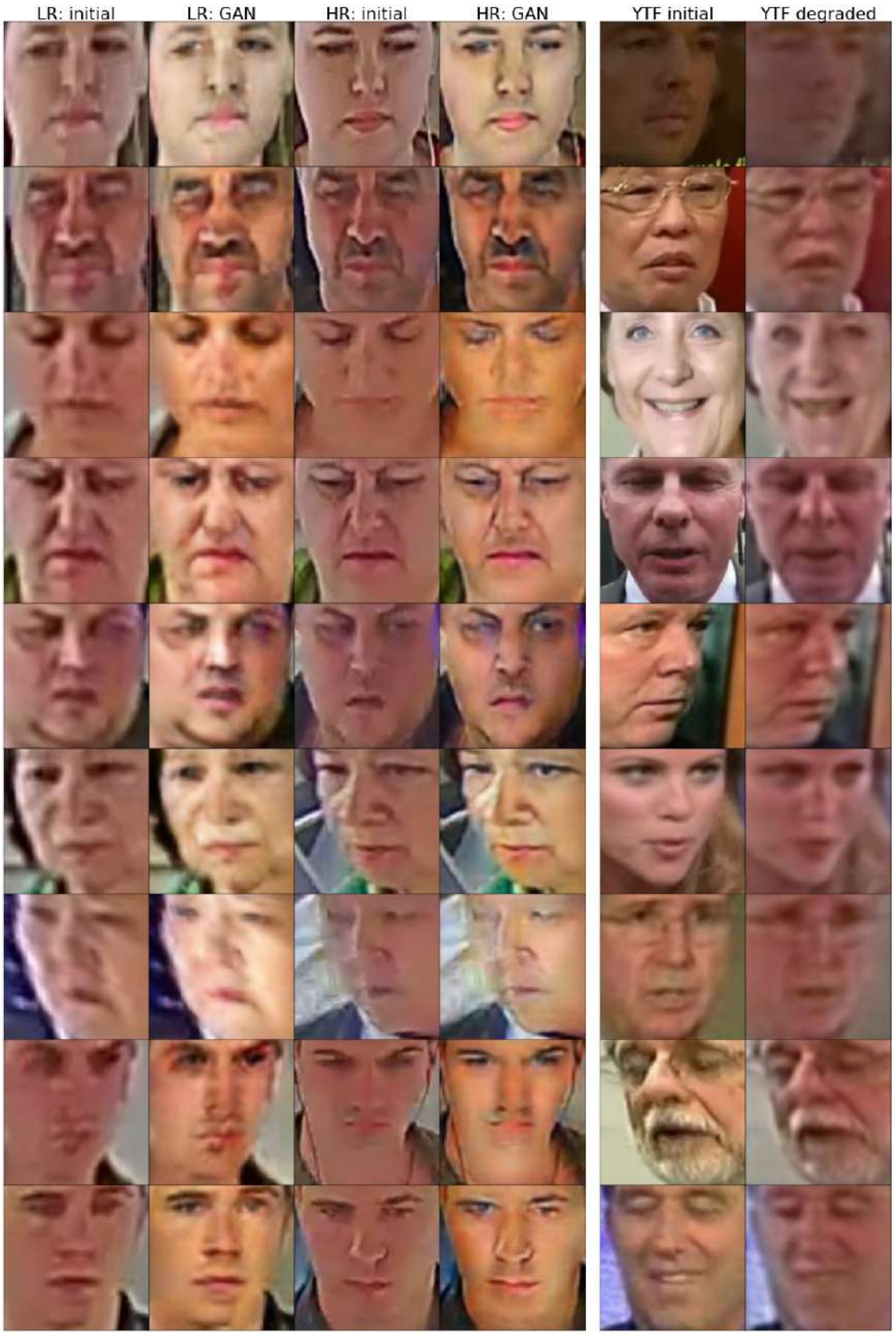


FIGURE 6.2: Columns one and three show images from our test surveillance data, while columns two and four contain the corresponding images transformed to the Internet data domain. See section 6.2 for the details. The last two columns show the examples of the reverse transformation from the Internet image domain to the surveillance image domain for the YouTube faces dataset.

$F^{T\to S} : T \longrightarrow S$ and $F^{S\to T} : B \longrightarrow A$ are the image-level domain transfer mappings. In the restoration-based approach, we train the recognition model R^S using labeled data $\{(X_i^S, Y_i^S)\}$ where $X_i^S \subseteq B$. We then test the learned model by computing the descriptors d^S after applying the network $F^{T\to S}$: $d^S = R^S(X^{T\to S}) = R^S(F^{T\to S}(X^T))$

In the domain adaptation approach, we train the recognition network R^T using labeled data $\{(X_i^{S\to T}, Y_i^S)\}$, where the training examples $X_i^{S\to T} = F^{S\to T}(X_i^S)$, $X_i^S \subseteq S$ are obtained by transforming the high-quality images to the low-quality domain using the learned transformation $F^{S\to T}$. In this case, we apply the learned network directly to the low-quality images by computing and working with their descriptors $d^T = R^T(X^T)$. The two approaches are compared below.

Learning domain transfer mappings

We use the CycleGAN approach [Zhu et al., 2017] to simultaneously learn the domain transfer mappings in both directions: $F^{T\to S} : T \longrightarrow S$ (restoration-based approach) and $F^{S\to T} : S \longrightarrow T$ (domain adaptation approach). Here we describe the objective functions used for learning the domain transfer architecture.

We use the variant of CycleGAN similar to the one introduced in [Liu et al., 2017] as we found it resulting in more stable and visually more plausible results for our task than the original framework [Zhu et al., 2017]. Following [Liu et al., 2017], we decompose the domain transfer mappings into the compositions of encoders and generators: $F^{T\to S} = G^S \odot E^T$ and $F^{S\to T} = G^T \odot E^S$. Here, the encoders E^T and E^S transfer input images to the latent space, and generators G^S and G^T map the input latent codes to the domains S and T.

For inputs $X^T \subseteq T$ and $X^S \subseteq S$ the results of their transfer to the opposite domain will be:

$$X^{T\to S} = F^{T\to S}(X^T; \theta_F^T) = G^S(E^T(X^T)) \tag{6.1}$$

$$X^{S\to T} = F^{S\to T}(X^S; \theta_F^S) = G^T(E^S(X^S)) \tag{6.2}$$

The objective function used by the CycleGAN approach for learning is composed of the two symmetric parts:

$$\mathcal{L} = \mathcal{L}^T + \mathcal{L}^S, \tag{6.3}$$

where $\mathcal{L}^T$ further decomposes as:

$$\mathcal{L}^T = \mathcal{L}^T_{\text{GAN}} + \lambda_1 \mathcal{L}^T_{\text{cycle}} + \lambda_2 \mathcal{L}^T_{\text{rec}}, \tag{6.4}$$

while $\mathcal{L}^S$ has same structure as $\mathcal{L}^T$:

$$\mathcal{L}^S = \mathcal{L}^S_{\text{GAN}} + \lambda_1 \mathcal{L}^S_{\text{cycle}} + \lambda_2 \mathcal{L}^S_{\text{rec}}, \tag{6.5}$$

We now describe each of the terms in (6.4). The GAN loss serves as the optimization objective for the domain transfer:

$\text{L}^A_{\text{GAN}} = \min_{\theta^S_F} \max_{\theta^T_D} \mathbb{E}_{x \sim p_{X^T}} \log D^T(x) + \mathbb{E}_{x \sim p_{X^S}} \log \left(1 - D^T(F^{S \to T}(x))\right)$

Here, $D^T(X; \theta^T_D)$ and $D^S(X; \theta^S_D)$ are discriminators for the domains T and S that are trained in parallel with the training of the domain transforms.

The other two terms are the so-called cycle consistency loss:

$$\mathcal{L}^T_{\text{cycle}} = L_1(F^{S \to T}(F^{T \to S}(X^T)), X^T) \tag{6.6}$$

and the reconstruction loss:

$$\mathcal{L}^T_{\text{rec}} = L_1(G^T(E^T(X^T)), X^T) \tag{6.7}$$

In both terms, $L_1(\cdot, \cdot)$ denotes the L_1 distance.

We show the results of transferring the Internet and surveillance images to the other domain in figure 6.2. While these results look interesting, we do not analyze their visual quality, as we are ultimately interested in the recognition performance rather than obtained visually-convincing images.

Learning face recognition models

In both scenarios that we compare in this paper, we need to train a face recognition model that turns images into vectorial descriptors. This happens either in domain S (in the face restoration approach) or in domain T (in the domain adaptation approach).

In either case, the goal of the training is to build a deep convolutional network that converts face images to the descriptors, such that matching face images have close descriptors and non-matching face descriptors have dissimilar descriptors. We use the Binomial Deviance loss [Yi et al., 2014a] to perform such training (2.8). We note that the choice of a particular metric learning loss is orthogonal to our study.

Alternatively to the models trained using the setting discussed above, we also consider reusing the VGG face model trained by the authors of [Parkhi et al., 2015] on the VGG-face dataset.

6.3 Experiments

We now perform evaluation and the comparison of the two approaches and their variants.

Datasets and protocols

Surveillance data

For the surveillance image domain (A, as denoted in Section 6.2), we have obtained a surveillance dataset comprising faces from five cameras in Moscow subway. Our dataset consists of two subsets of images, denoted as `LR` (low-resolution) and `HR` (high-resolution) according to their size in pixels. Sizes are calculated as the face bounding box heights. Mean face heights for `LR` and `HR` subsets were 49.72 and 106.19 pixels correspondingly. The `LR` images range from 37 to 63 pixels, and `HR` images from 75 to 224. The DLib [King, 2009] library was used for face detection and subsequent alignment.

Generally, the identities of people occurring in the video are unknown, and therefore we mine matching faces in the dataset by considering temporal tracks in videos. We assume that face images from different tracks are non-matching. The matching pairs then correspond to pairs of faces from the same track, such that one image belongs to the LR subset and the other belongs to the HR subset. Columns 1 and 3 in Figure 6.2 show some examples of matched pairs. Usually, the quality of face images increases when the person approaches the camera, and therefore `HR`-subset images are often (but not always) visually better than those present in the `LR` subset. The division into `LR` and `HR` subsets is introduced to ensure that the matched pairs of frames correspond to distinct frames of the temporal tracks. We stress that the mined matching pairs are used for evaluation (testing) only and are not used for training of any networks in our experiments.

We use 96 identities that are present in both `LR` and `HR` for parameter validation and 279 identities for test. For each of the test identity, there is a pair of matching tracks (one LR track and one HR track). The goal of algorithms is then to build descriptors that would be similar for frames coming from the HR and the LR tracks of the same identity, and would be dissimilar for the HR and the LR tracks corresponding to different identities. Overall, $3,891$ images were used for test.

$4,979$ images of identities not present in the test set are used for training the unsupervised domain transfer described in Section 6.2 (tracking-based information was not used to train the ConvNets). The mean number of frames for each identity in `LR` and `HR` test data are 18.62 and 17.72 correspondingly.

To evaluate the recognition quality, we match identities across the LR and HR in the following way: we calculate the cosine similarity between the frame set $t^{LR}_{id_1}$ of identity id_1 and the frame set, $t^{HR}_{id_2}$ of identity id_2 by averaging the similarities of each pair of frames:

$$S\left(t^{LR}_{id_1}, t^{HR}_{id_2}\right) = \frac{1}{|t^{LR}_{id_1}| * |t^{HR}_{id_2}|} \sum_{i=0,j=0}^{|t^{LR}_{id_1}|,|t^{HR}_{id_2}|} S\left(f^{LR}_{id_1,i}, f^{LR}_{id_2,j}\right), \tag{6.8}$$

$$S\left((f^{LR}_{id_1,i}, f^{LR}_{id_2,j}\right) = cos\left(d^{LR}_{id_1,i}, d^{LR}_{id_2,j}\right), \tag{6.9}$$

where $d^{LR}_{id_1,i}$,$d^{HR}_{id_2,j}$ are the descriptors of the corresponding frames calculated by face recognition neural network R 6.2.

Evaluation metrics

We focus our evaluation on the surveillance data domain. As already mentioned above, during evaluation, we compare pairs of *tracks*, where the first track comes from the LR subset and the second track comes from the HR subset.

When comparing the two tracks using the recognition metrics, we match all possible pairs of frames and compute the mean average cosine similarity between the computed descriptors over all pairs (more sophisticated schemes involving minimal pairs did not result in better performance). Depending on whether the mean average cosine similarity is higher or lower than a certain threshold τ, we treat a certain pair of tracks as matching or non-matching.

By considering various τ, we then compute the *ROC curve*, the area under the ROC curve (ROC AUC), the 100% - EER (Equal Error Rate) statistics, and the average precision (AP) metrics.

Internet data

For the Internet image domain (B, as denoted in Section 6.2) we use three face recognition datasets: Celeb-A [Liu et al., 2015], YouTube Faces (YTF) [Wolf et al., 2011] and the VGG Face datasets [Parkhi et al., 2015].

The Celeb-A dataset [Liu et al., 2015] consists of 202, 599 images of high quality and is used for training CycleGAN-based domain transfer described in Section 6.2.

The Youtube Faces (YTF) dataset [Wolf et al., 2011] and the VGG Face dataset [Parkhi et al., 2015] are used for the finetuning of the face recognition neural network as described

in Section 6.2. YTF consists of 3,425 videos of 1,595 people collected from YouTube, with an average of 2 videos per identity. The VGG Face dataset contains 2,6M images of 2,622 identities. We show that face recognition improves, when using our CycleGAN-based data augmentation when trained on either YTF or VGG Face. Generally, YTF and VGG Face represent two different types of face images that can be mined from the Internet (with YTF having lower quality).

Compared variants of recognition networks

We compare the following adaptation/transfer strategies for training the recognition networks:

- *no ft* – the pre-trained VGG-face model [Parkhi et al., 2015] with no re-training is used to compute descriptors of the surveillance-domain images.
- *ft initial* – the VGG-face model is fine-tuned using the original version of the YTF or the VGG Face datasets (no domain adaptation).
- *ft degraded* – the VGG-face model fine-tuned using the degraded version of the YTF or VGG Face datasets transferred to the target (surveillance) domain (using domain adaptation),
- *ft union* – the VGG-face model fine-tuned using **both** the initial and the degraded versions of the YTF or VGG Face datasets (using domain adaptation).

The YTF dataset images and the corresponding degraded images are shown in Figure 6.2 (the two last columns).

Training details

The CycleGAN-based domain transfer consists of 3 types of modules (see Section 6.2). Encoders E_A and E_B have the following architecture:

#conv layer	1	2	3	4	5	6	7
num of filters	32	64	128	128	128	128	128
kernel size	3	3	3	3	3	3	3
stride/pad	1/0	2/1	2/1	1/1	1/1	1/1	1/1
#res block	-	-	-	0	0	1	1

The architecture of generators G_A, G_B is as follows:

#conv layer	1	2	3	4	5	6	7
num of filters	128	128	128	128	64	32	3
kernel size	3	3	3	3	3	3	1
stride/pad	1/1	1/1	1/1	1/1	1/1	1/1	1/0
#res block	0	0	1	1	-	-	-

Instance normalization [Ulyanov et al., 2017] and Leaky ReLU [He et al., 2015] with negative slope set to 0.01 are inserted after each of the convolution layers. Except for the last layers of G_A and G_B, where `tanh` non-linearity is used (for the subsequent feeding of the result into the discriminator). To keep the input image size unchanged, $\times 2$ nearest neighbor upsampling is done before the convolutions 3 and 4. All input images are normalized to 64×64 pixels (output images are therefore of the same size).

Finally, discriminators D_A and D_B have the following architecture:

#conv layer	1	2	3	4	5
num of filters	64	128	256	256	1
kernel size	3	3	3	3	3
stride/pad	2/1	2/1	2/1	2/1	1/0

Leaky ReLU [He et al., 2015] with negative slope parameter set to 0.01 is used as an activation. The final fully-connected layer with one output unit is added to D_A and D_B.

According to the strategies described in Section 6.3, we fine-tune the VGG-face model, having added new 128-dimensional embedding layer instead of the initial classification layer (*fc8*). All the results are reported for the *fc7* layer though, as we found it to work better across all compared methods.

ADAM optimization [Kingma and Ba, 2014] was used for both optimization objectives of the domain transfer task and the face recognition task, the learning rates were set to $1e-4$ and $1e-7$ correspondingly. Batch sizes were 16 and 64. Learning processes took 50 epochs for the domain transfer, and from 80 to 200 epochs for the face recognition task. Parameters λ_1 and λ_2 were set to 10 in the loss (6.4). Parameters α, β and C of the loss (2.8) were set to 2, 0.5 and 10.

For training the face recognition models described in Section 6.3, the training batches had the following structure: in each batch, there were up to 3 examples for each class for the VGG Face dataset and up to 10 examples for the YTF dataset. For the *ft union* model, each training batch was formed out of the examples of one of the domains. The sampling process alternated between the domains at each iteration.

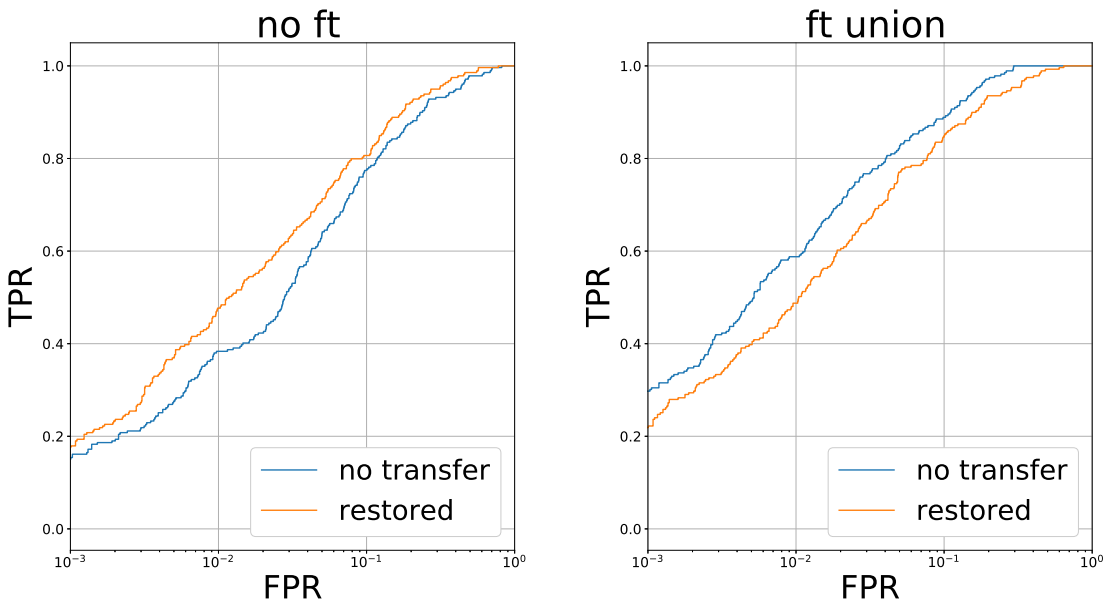


FIGURE 6.3: The ROC curves for our face recognition models for different types of test data. Left – *no ft* model, test data transformation (the curve is denoted by 'restored') improves recognition. Middle – *ft initial*, the results for test data transformation are not clearly better than the results for initial images. Right – *ft union* model, the best results are for initial test data, without transformation. See 6.3 for the details.

Does the translation to the Internet domain help recognition?

We start by assessing the improvement that the restoration process brings to the recognition. For this we evaluate the performance of the three different recognition networks discussed above, when they are applied either to untransformed surveillance-domain images or to surveillance-domain images transformed to the Internet image domain (using the learned mapping $F^{A \to B}$). See Figure 6.2 for the example results of the Internet domain transfer.

The ROC-curves in Figure 6.3 shows that while the restoration process helps for the *no ft* network, it actually *hurts* for the better-performing *ft union* network. While trying to improve the results of the reverse transfer, we have also tried to transfer only the LR subset of the training images, while keeping the HR subset intact, but this lead to uniformly worse results.

We conclude that restoration does not necessarily help the recognition process in our setting.

In the course of our experiments, we have also tried several different options for training CycleGAN. For example, we observed that the CycleGAN training scheme from [Zhu et al., 2017] did not lead to visually good results for our data. Second, for the CycleGAN version mentioned in the paper, we have also tried an approach similar to that of [Sungatullina et al., 2018]. In more details, we not only fed images to the higher-quality

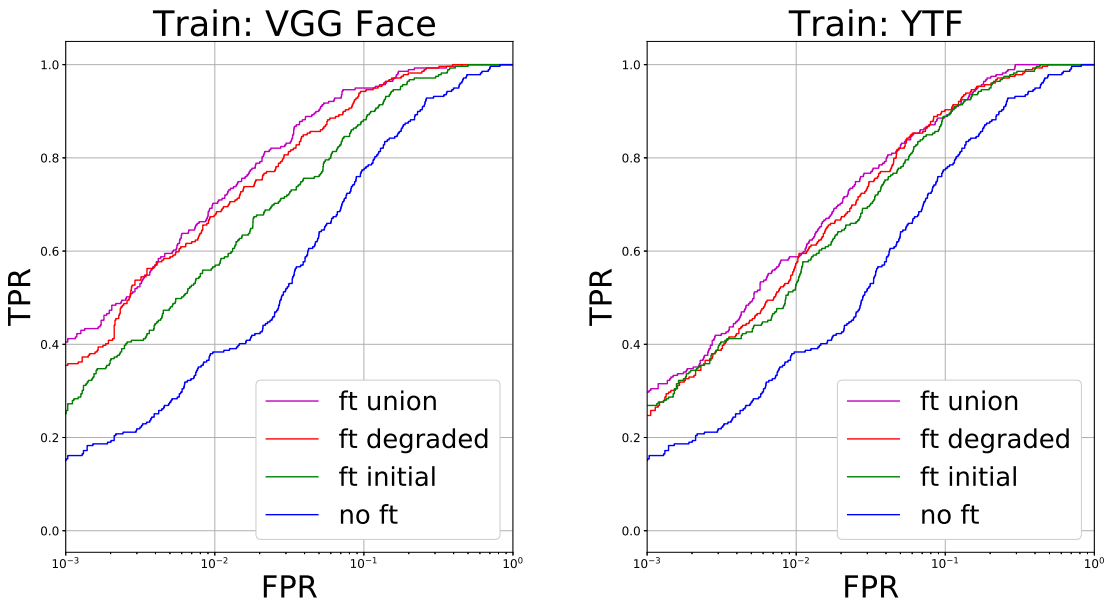


FIGURE 6.4: The ROC curves for the fine-tuning strategies described in Section 6.3. Our surveillance data is used for test. *ft degraded* and *ft union* models that use our image-level domain adaptation are better than other models.

TABLE 6.1: Quality metrics for different face recognition models compared in this work (6.3). Surveillance data are used for evaluation. The best model is *ft union*, it is trained on both initial and degraded data.

	train: VGG Face			train: YTF		
Fine-tuning type	100%−eer	roc auc	AP	100%−eer	roc auc	AP
no ft	83.89	91.86	22.04	83.89	91.86	22.04
ft initial	89.12	96.25	42.02	89.24	96.19	40.81
ft degraded	91.55	97.70	52.55	**90.04**	96.54	42.07
ft union	**92.94**	**98.08**	**57.30**	89.37	**96.96**	**45.45**

domain discriminator D^I, but also intermediate features extracted from a pretrained face recognition network. As a result, we observed that restoration transfer gave very visually plausible results, but recognition quality after such transform dropped considerably and was uniformly worse than all the results in Figure 6.3. This might be due to non identity-preserving way of facial features change.

Does domain adaptation help recognition?

We now consider the second scenario based on domain adaptation and thus compare the performance of different recognition networks described above on surveillance data. Our findings for image level domain adaptation are summarized in Table 6.1, which shows metric values for the compared training settings. Finally, Figure 6.4 shows the final ROC curves.

TABLE 6.2: Quality metrics of our approach compared to that of [Ganin et al., 2016] and [Sohn et al., 2017] and some combinations of these methods with ours. Surveillance data are used for evaluation. 'fm loss' and 'fr loss' stand for Feature Matching loss and Feature Reconstruction loss as defined in [Sohn et al., 2017]. 'adv loss' column shows if the method of [Ganin et al., 2016] for feature-level domain adaptation is used. 'aug type' column shows which augmentation method is used for each of the models: 'CycleGAN' means our CycleGAN based augmentation, 'predefined' means the procedure used in [Sohn et al., 2017]. The *ft union* model is denoted A in this table. Model B combines feature-level ('adv loss') and image-level domain adaptation ('aug type' is set to 'CycleGAN'), it outperforms model A slightly. Overall, augmentation choice is very important: models D and I are trained using scheme from [Sohn et al., 2017] with different augmentation methods, and model D performs better than model I and on par with B.

	fm loss	fr loss	adv loss	aug type	100% - eer	roc auc	AP
ours							
A	-	-	-	CycleGAN	89.37	**96.96**	45.45
B	-	-	✓	CycleGAN	**89.75**	96.72	**46.22**
C	-	-	✓	-	88.30	95.63	39.01
our implementation of [Sohn et al., 2017]							
D	✓	✓	✓	CycleGAN	**90.34**	**96.83**	**43.08**
E	-	-	✓	CycleGAN	89.26	96.52	42.75
F	-	✓	✓	CycleGAN	89.86	96.51	40.10
G	✓	-	✓	CycleGAN	88.05	96.06	40.89
H	✓	✓	-	CycleGAN	89.79	96.76	42.56
I	✓	✓	✓	predefined	**89.22**	**96.16**	**40.66**
J	-	-	✓	predefined	88.69	95.92	38.92

The following can be observed. First, fine-tuning the VGG Face model on either VGG Face dataset or the YTF dataset using the BinDev loss (*ft initial*) lead to a considerable improvement over the original state of the network. Furthermore, we found that fine-tuning in the *ft degraded* setting is clearly beneficial compared to *ft initial* setting in the case of the VGG Face training dataset, and a little bit better for the YTF training data, overall making a case for image-level domain adaptation. The results of the *ft union* setting are uniformly better than *ft initial* and *ft degraded* suggesting that the automatically degraded data are a useful augmentation, but that the original data should not be discarded.

Finally, Figure 6.6 shows that the feature distribution of our surveillance data is more intermixed with those of the degraded version of the YTF dataset than its initial version. This aligns with the performance improvement we achieved with image-level domain adaptation.

Comparison to other domain adaptation methods

We have additionally compared our image-level domain adaptation (all the experiments and discussions above) to the feature-level domain-adversarial adaptation approach [Ganin et al., 2016] as well as to the approach described in [Sohn et al., 2017].

FIGURE 6.5: Examples of degraded images produced using a predefined procedure from [Sohn et al., 2017]. See Table 6.2 for comparison with our augmentation.

[Ganin et al., 2016] proposes a feature-level adaptation technique based on reversing the sign of gradients of the domain discriminator with respect to the intermediate representations in order to make those representations more domain-invariant. We thus built a DANN (Deep Adversarial Neural Network) based on the VGG-face network. The domain classifier consists of three fully-connected layers: 512 units in the first two layers (Leaky ReLU [He et al., 2015] non-linearities were used) and one classification layer with 1 unit. Dropout with 0.5 probability was inserted before the classification layer. The Gradient Reversal layer is attached after the *fc6* layer of VGG-face. We found that schedule for the adaptation parameter λ is very important for this task. Instead of the schedule suggested in [Ganin et al., 2016] (which did not lead to good results in our comparison), we set λ to $1e-3$ and increased it to $1e-2$ and $1e-1$ after the first and the fifth epochs correspondingly. In these experiments, we fine-tune one of our pre-trained models (either *ft initial* or *ft union*) with additional feature-level domain adaptation objective.

The results for training on the YTF dataset with method [Ganin et al., 2016] are shown in Table 6.2: model A corresponds to our *ft union* model, model B combines the image-level domain-adaptation (the same as for *ft union*) and feature-level domain adaptation, model C uses feature-level domain adaptation only. Feature-level domain adaptation improves the performance slightly if combined with the image-level domain adaptation (model B compared to model A), whereas applied alone, it leads to overfitting (model C).

We have also compared our approach to the method described in [Sohn et al., 2017]. It aims at performing feature-level adaptation and restoration using artificially-degraded data as a bridge between the two domains. The goal is to train such a model that would be tolerant to image degradation and at the same time, retain the representations of

the source domain data. The authors also incorporate a feature-level adversarial loss, and we use that of [Ganin et al., 2016] for our implementation of [Sohn et al., 2017] for better comparability. Instead of the Image Classification loss used in [Sohn et al., 2017], we use the BinDev loss as for the other experiments in this work. Two other losses that were used in [Sohn et al., 2017], apart for Image Classification loss and Domain-Adversarial loss, are Feature Matching and Feature Restoration losses. To implement the scheme from [Sohn et al., 2017], we added these two losses to the final objective function. Additionally, it should be noted that in [Sohn et al., 2017], the network is initialised by the weights of the model trained on the source domain (*ft initial* in our case). We also used the predefined degradation procedure described in [Sohn et al., 2017] to generate augmentation data.

The results of our implementation of the scheme from [Sohn et al., 2017] are shown in Table 6.2 in rows D-J. All the models D-J are fine-tuned with additional losses from [Sohn et al., 2017] (with changes discussed above) starting at the state of *ft inital*. Models D and I differ in the utilized augmentation method. Model I uses pre-defined augmentation procedure ('predefined', see Figure 6.5 for examples) from [Sohn et al., 2017], whereas our CycleGAN-based augmentation is used for D. We can see that our augmentation results in better performance in comparison to that of [Sohn et al., 2017] for our data. We also compared the two augmentation methods for our training scheme *ft-union* and observed very similar results (not shown here). To ensure the sanity of our implementation of [Sohn et al., 2017], we also show the role of different subsets of losses in the rows D-H of Table 6.2. Indeed, we can see that model D, that is trained with the full set of losses, outperforms other models E-H.

Finally, we can see from Table 6.2, that our approach (models A and B) outperforms the scheme from [Sohn et al., 2017] (model I), and as it can be seen from the comparison, the difference mostly comes from the augmentation method: model D is trained using our CycleGAN-based augmentation and the resulting performance is close to that of our model B.

6.4 Discussion and Conclusions

In this chapter, we have compared the image-based domain adaptation techniques for face recognition in the presence of strong image degradation. We consider the recently proposed CycleGAN approach for learning mappings between the two domains of surveillance data and Internet face images. We demonstrate that the strategy of transferring the labeled Internet data to the surveillance domain and subsequent retraining the face recognition network helps to improve the recognition quality on real surveillance test

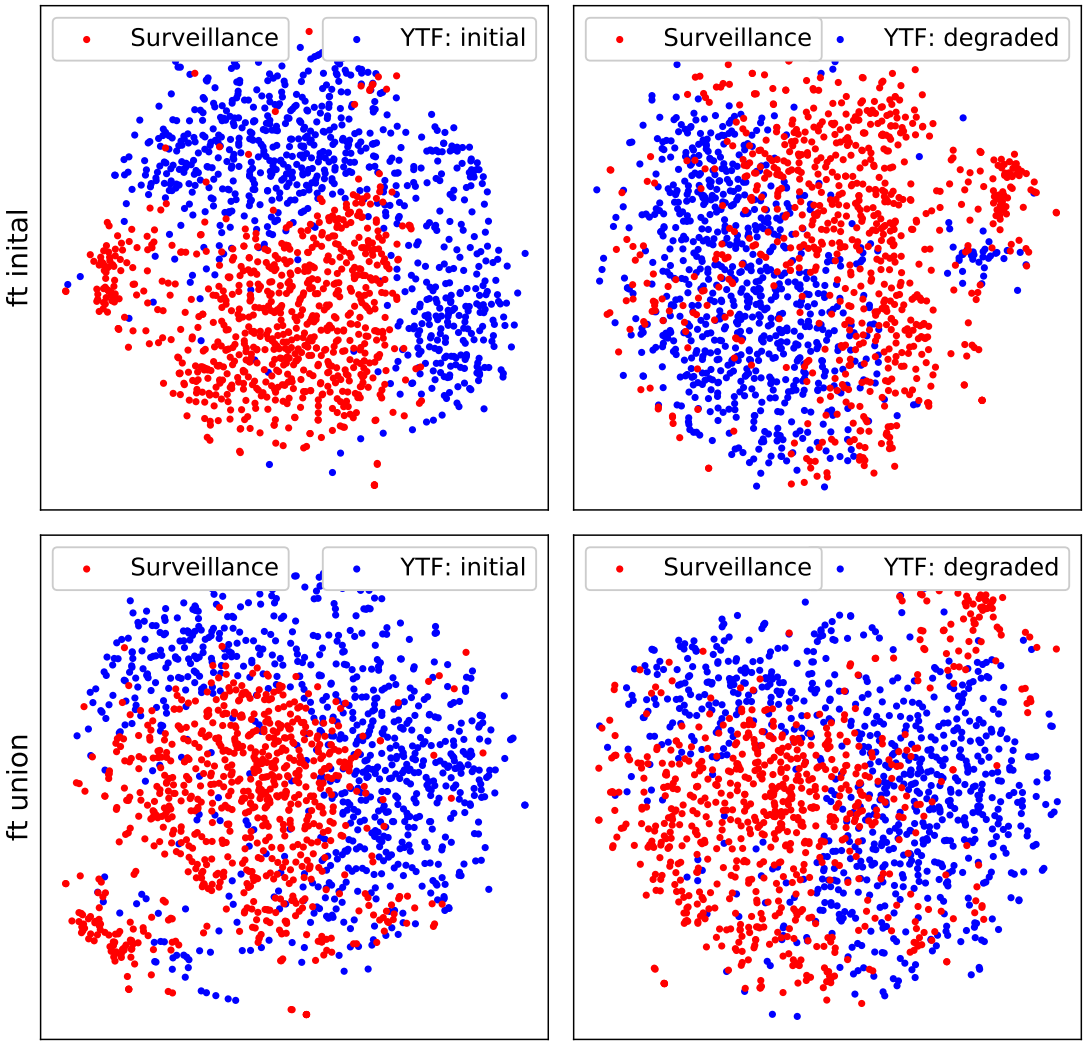


FIGURE 6.6: t-SNE [Maaten and Hinton, 2008] visualizations for deep features extracted with: first row – *ft initial*, second row – *ft union* neural nets (See 6.3 for the descriptions of the models). For both nets, surveillance data distribution is more intermixed with the degraded version of the YTF dataset than with its initial version.

data. We have investigated the variants of this approach, and have demonstrated that training on both transferred and original Internet data leads to the optimal performance. Finally, we show that in the case of our domain pair, the image-level adaptation approach outperforms feature-level domain adaptation.

We found our results consistent with [Sohn et al., 2017], where face recognition for the low-quality is also considered. In [Sohn et al., 2017], verification accuracy was improved using carefully chosen data augmentation. However, we also show that for our data, the approach analogical to [Sohn et al., 2017] benefits considerably from using our automatic augmentation.

In our experiments, best results were achieved with training on both domains. An

explanation can be given that the domain transfer model is imperfect and may push different images of the same identity too far as we do not use any kind of verification loss for training (ideally, this would require another pre-trained network for the low-quality domain, which essentially is the goal of this work). Therefore, keeping the initial data in the training set may result in less overfitting.

As an important negative result, we show that using CycleGAN-based restoration of lower-quality domain images by transferring them to the higher-quality domain does not bring consistent improvement to the recognition performance, which is also consistent with the results of [Sohn et al., 2017]. We speculate that such transfer may distort some details of the images in a non-identity preserving way.

Our study considers a practically important domain of image data. We note, however, that our findings might not transfer to other pairs of domains in image adaptation as optimal strategies may vary for different domains of data. Here were are experimenting with the two domains, one of which is associated with the information loss, including identity information crucial for our main task (face verification). In our case, the strategy of translation from the target domain to the source domain (restoration) did not quite work, or it was harder to make it work than the opposite approach. However, this strategy might be more applicable to some other domains, e.g. high-quality city photos and synthetic city images.

Acknowledgement: This research is supported by VisionLabs and the Ministry of Education and Science of the Russian Federation (grant 14.756.31.0001).

Chapter 7

Conclusion

In this work, three important aspects of building deep learning systems for human recognition are addressed: the architecture design, the objective function design and deep domain adaptation techniques. Most of the results demonstrated in the work are for person re-identification; cross-domain learning has been also considered for surveillance face recognition.

A new loss function, called the Histogram loss, for learning deep embeddings using siamese neural networks has been suggested in this work. Like many other existing loss functions, it encourages higher similarity values for semantically related examples, and lower similarity values for unrelated examples. Unlike other loss functions that are computed for tuples of descriptors (*e.g.*, pairs), the suggested loss is computed for two one-dimensional distributions of similarity values: one is for similarity values for *positive* (matching) pairs of embeddings, and another is for similarity values for *negative* (non-matching) pairs. The goal of this loss function is to reduce the overlap between such distributions, which results in higher similarity values for positive pairs compared to those for negative pairs. Experiments have shown that such loss performs favorably for a range of problems, including person re-identification for CUHK03 and Market-1501 datasets, retrieval of bird species for the CUB-200-2011 dataset and online products for the Stanford Online Products dataset. Notably, the performance of the Histogram loss has appeared to be the best for person re-identification among several losses based on tuples. In more detail, for person re-identification, it outperformed well-tuned pairwise Binomial Deviance loss (2.8) and the Lifted Structured Similarity Softmax loss (2.13), that is biased to hard negative examples.

A person re-identification architecture, called Multi-region Bilinear CNN, is also presented in this work. It is based on the idea of Bilinear CNNs that learn deep representations in a factorized form: an outer product of two stacks of feature maps is

computed for each spatial location, followed by global sum-pooling. Such architectures are able to model non-rigid transformations and view-point changes in fine-grained recognition problems, however, such global pooling appears to be too radical for person re-identification. Multi-region Bilinear CNN has been proposed as a certain compromise between Bilinear CNNs and regular CNN architectures that usually consist of convolution layers followed by non-linearity and fully-connected layers. The pooling of bilinear features is performed separately for a set of regions instead of pooling over all the spatial locations. Such modification allows preserving some spatial information in the output descriptor. The proposed architecture has been shown to perform better than a number of baselines, including Bilinear CNN and a regular CNN, for three popular re-identification datasets: CUHK03, CUHK01 and Market-1501. For the two largest datasets (CUHK03, Market-1501), state-of-the-art results have been achieved (on the moment of publication).

Cross-domain human recognition has also been addressed in this work. For person re-identification, the case of training and testing on disjoint sets of cameras has been considered. Namely, domain-adversarial training approach, also described in Chapter 5, has been demonstrated to improve the results for cross-domain training, *i.e.*, when the re-identification siamese neural network is trained and tested using data from different pedestrian datasets. In more detail, domain-adversarial training allows to learn deep representations that are useful for the task of interest (*e.g.*, classification), and at the same time are domain-invariant. This is achieved by simultaneous training of two predictors: the main task classifier and the binary domain classifier, that predicts to which of the two domains the input data belong. At the same time, the underlying deep representation is trained to minimize the task-specific objective and to maximize the domain prediction loss. For person re-identification, the label predictor is replaced by a descriptor predictor. The experiments show the advantage of using domain-adversarial training for a total of eight domain pairs, where each of the domains corresponds to one of three publicly available person re-identification datasets: VIPER, PRID and CUHK02.

The challenging case of surveillance face recognition has been also considered. Having a face recognition neural network trained on labeled publicly available data, that are harvested from the Internet and mostly consist of professional photographs, the goal has been to adapt it to unlabeled surveillance data of much lower quality. The domain shift, in this case, is characterized by a complex combination of different degradation factors such as compression, blur, and illumination change. An image-level domain adaptation technique, based on cycle-consistent image-to-image translation model (CycleGAN), has been evaluated and compared to several baselines including domain-adversarial training, also described in Chapter 5. The image-level translation allows to build two mappings to transfer the images of one domain to the other and vice versa. The approach works

in two stages: first, the labeled source data are transferred to the target domain, and then the recognition network is retrained using such transferred data. The evaluated baselines also include the backward translation from the target surveillance domain to the source Internet domain. Based on performed evaluations and comparisons, a strategy for surveillance face recognition has been suggested. In brief, the most effective approach was to use a mixture of the source Internet data and its version transferred to the surveillance domain.

Bibliography

Ejaz Ahmed, Michael J. Jones, and Tim K. Marks. An improved deep learning architecture for person re-identification. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2015.

Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):898–916, 2010.

Artem Babenko and Victor Lempitsky. Aggregating deep convolutional features for image retrieval. *arXiv preprint arXiv:1510.07493*, 2015.

Artem Babenko, Anton Slesarev, Alexandr Chigorin, and Victor Lempitsky. Neural codes for image retrieval. In *European Conference on Computer Vision, ECCV*, 2014.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*, 2015.

Slawomir Bak, Peter Carr, and Jean-Francois Lalonde. Domain adaptation through synthesis for unsupervised person re-identification. In *Conference on Computer Vision (ECCV)*, pages 189–205, 2018.

Mahsa Baktashmotlagh, Mehrtash T Harandi, Brian C Lovell, and Mathieu Salzmann. Unsupervised domain adaptation by domain invariant projection. In *International Conference on Computer Vision, ICCV*, 2013.

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

Aurélien Bellet, Amaury Habrard, and Marc Sebban. A survey on metric learning for feature vectors and structured data. *arXiv preprint arXiv:1306.6709*, 2013.

Anil Bhattacharyya. On a measure of divergence between two statistical populations defined by their probability distributions. *Bull. Calcutta Math. Soc.*, 35:99–109, 1943.

Karsten M Borgwardt, Arthur Gretton, Malte J Rasch, Hans-Peter Kriegel, Bernhard Schölkopf, and Alex J Smola. Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics*, 22(14):e49–e57, 2006.

Steve Branson, Grant Van Horn, Serge Belongie, and Pietro Perona. Bird species categorization using pose normalized deep convolutional nets. *arXiv preprint arXiv:1406.2952*, 2014.

Jane Bromley, James W Bentz, Léon Bottou, Isabelle Guyon, Yann LeCun, Cliff Moore, Eduard Säckinger, and Roopak Shah. Signature verification using a siamese time delay neural network. *International Journal of Pattern Recognition and Artificial Intelligence*, 7(04):669–688, 1993.

Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2017.

Joao Carreira, Rui Caseiro, Jorge Batista, and Cristian Sminchisescu. Semantic segmentation with second-order pooling. In *European Conference on Computer Vision, ECCV*, 2012.

Shi-Zhe Chen, Chun-Chao Guo, and Jian-Huang Lai. Deep ranking for person re-identification via joint representation learning. *IEEE Transactions on Image Processing*, 25(5):2353–2367, 2016.

De Cheng, Yihong Gong, Sanping Zhou, Jinjun Wang, and Nanning Zheng. Person re-identification by multi-channel parts-based cnn with improved triplet loss function. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1335–1344, 2016.

Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2005.

Sumit Chopra, Suhrid Balakrishnan, and Raghuraman Gopalan. Dlid: Deep learning for domain adaptation by interpolating between domains. In *ICML workshop on challenges in representation learning*, volume 2, 2013.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20 (3):273–297, 1995.

Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, 2004.

Jason V Davis, Brian Kulis, Prateek Jain, Suvrit Sra, and Inderjit S Dhillon. Information-theoretic metric learning. In *International conference on Machine learning, ICCV*, 2007.

Weijian Deng, Liang Zheng, Qixiang Ye, Guoliang Kang, Yi Yang, and Jianbin Jiao. Image-image domain adaptation with preserved self-similarity and domain-dissimilarity for person re-identification. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2018.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

Basura Fernando, Amaury Habrard, Marc Sebban, and Tinne Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *International Conference on Computer Vision, ICCV*, 2013.

Itzhak Fogel and Dov Sagi. Gabor filters as texture discriminator. *Biological cybernetics*, 61(2):103–113, 1989.

Yoav Freund, Raj Iyer, Robert E Schapire, and Yoram Singer. An efficient boosting algorithm for combining preferences. *Journal of machine learning research, JMLR*, 4 (Nov):933–969, 2003.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research, JMLR*, 17 (1):2096–2030, 2016.

Yang Gao, Oscar Beijbom, Ning Zhang, and Trevor Darrell. Compact bilinear pooling. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2016.

Amir Globerson and Sam T Roweis. Metric learning by collapsing classes. In *Advances in neural information processing systems, NIPS*, 2006.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *International conference on machine learning, ICML*, 2011.

Jacob Goldberger, Geoffrey E Hinton, Sam T Roweis, and Ruslan R Salakhutdinov. Neighbourhood components analysis. In *Advances in neural information processing systems, NIPS*, pages 513–520, 2005.

Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2012.

Boqing Gong, Kristen Grauman, and Fei Sha. Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *International Conference on Machine Learning, ICML*, 2013.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014.

Henry Gouk, Bernhard Pfahringer, and Michael J. Cree. Learning distance metrics for multi-label classification. In *Proceedings of The 8th Asian Conference on Machine Learning, ACML 2016, Hamilton, New Zealand, November 16-18, 2016.*, pages 318–333, 2016.

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

Douglas Gray and Hai Tao. Viewpoint invariant pedestrian recognition with an ensemble of localized features. In *European Conference on Computer Vision, ECCV*, 2008.

Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *Conference on Computer Vision and Pattern Recognition CVPR*, 2006.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *International Conference on Computer Vision, ICCV*, 2015.

Martin Hirzer, Csaba Beleznai, Peter M. Roth, and Horst Bischof. H.: Person re-identification by descriptive and discriminative classification. In *Scandinavian Conference on Image Analysis, SCIA*, 2011.

Martin Hirzer, Peter M. Roth, and Horst Bischof. Person re-identification by efficient impostor-based metric learning. In *International Conference on Advanced Video and Signal-Based Surveillance, AVSS*, 2012.

Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A. Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *International Conference on Machine Learning, ICML*, 2018.

Sungeun Hong, Woobin Im, Jongbin Ryu, and Hyun Seung Yang. SSPP-DAN: deep domain adaptation network for face recognition with single sample per person. In *International Conference on Image Processing, ICIP*, 2017.

Junlin Hu, Jiwen Lu, and Yap-Peng Tan. Discriminative deep metric learning for face verification in the wild. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2014.

Jiayuan Huang, Arthur Gretton, Karsten Borgwardt, Bernhard Schölkopf, and Alex J Smola. Correcting sample selection bias by unlabeled data. In *Advances in neural information processing systems, NIPS*, 2007.

Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in neural information processing systems, NIPS*, 2015.

Rabia Jafri and Hamid R Arabnia. A survey of face recognition techniques. *Journal of Information Processing Systems*, 5(2):41–68, 2009.

Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. Aggregating local descriptors into a compact image representation. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2010.

Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

Thorsten Joachims. Optimizing search engines using clickthrough data. In *ACM SIGKDD international conference on Knowledge discovery and data mining*, 2002.

Daniel J Jobson, Zia-ur Rahman, and Glenn A Woodell. A multiscale retinex for bridging the gap between color images and the human observation of scenes. *IEEE Transactions on Image processing*, 6(7):965–976, 1997.

Kai Jüngling, Christoph Bodensteiner, and Michael Arens. Person re-identification in multi-camera networks. In *Conference on Computer Vision and Pattern Recognition, CVPR Workshops*, 2011.

Mahdi M Kalayeh, Emrah Basaran, Muhittin Gökmen, Mustafa E Kamasak, and Mubarak Shah. Human semantic parsing for person re-identification. In *Computer Vision and Pattern Recognition, CVPR*, 2018.

Dor Kedem, Stephen Tyree, Fei Sha, Gert R. Lanckriet, and Kilian Q Weinberger. Non-linear metric learning. In *Advances in Neural Information Processing Systems, NIPS*. 2012.

Davis E. King. Dlib-ml: A machine learning toolkit. *The Journal of Machine Learning Research JMLR*, 10:1755–1758, 2009.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Martin Köstinger, Martin Hirzer, Paul Wohlhart, Peter M. Roth, and Horst Bischof. Large scale metric learning from equivalence constraints. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2012.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems (NIPS)*, 2012.

Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

Cheng-Hao Kuo, Sameh Khamis, and Vinay D. Shet. Person re-identification using semantic color names and rankboost. In *IEEE Workshop on Applications of Computer Vision, WACV*, 2013.

Marc T Law, Nicolas Thome, and Matthieu Cord. Quadruplet-wise image similarity learning. In *International Conference on Computer Vision, ICCV*, 2013.

Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2006.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Dangwei Li, Xiaotang Chen, Zhang Zhang, and Kaiqi Huang. Learning deep context-aware features over body and latent parts for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 384–393, 2017.

Wei Li and Xiaogang Wang. Locally aligned feature transforms across views. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2013.

Wei Li, Rui Zhao, and Xiaogang Wang. Human reidentification with transferred metric learning. In *Asian Conference on Computer Vision, ACCV*, 2012.

Wei Li, Rui Zhao, Tong Xiao, and Xiaogang Wang. Deepreid: Deep filter pairing neural network for person re-identification. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2014.

Shengcai Liao, Yang Hu, Xiangyu Zhu, and Stan Z. Li. Person re-identification by local maximal occurrence representation and metric learning. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2015.

Jie Lin, Olivier Morere, Vijay Chandrasekhar, Antoine Veillard, and Hanlin Goh. Deephash: Getting regularization, depth and fine-tuning right. *arXiv preprint arXiv:1501.04711*, 2015.

Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems, NIPS*, 2017.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *International Conference on Computer Vision, ICCV*, 2015.

Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I. Jordan. Learning transferable features with deep adaptation networks. In *International conference on machine learning, ICML*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 97–105. JMLR.org, 2015.

David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision, IJCV*, 60(2):91–110, 2004.

Bingpeng Ma, Yu Su, and Frédéric Jurie. Bicov: a novel image representation for person re-identification and face verification. In *British Machine Vision Conference, BMVC*, 2012a.

Bingpeng Ma, Yu Su, and Frédéric Jurie. Local descriptors encoded by fisher vectors for person re-identification. In *European Conference on Computer Vision, ECCV, Workshops*, 2012b.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *The Journal of Machine Learning Research, JMLR*, 9:2579–2605, 2008.

Prasanta Chandra Mahalanobis. On the generalised distance in statistics. volume 2, pages 49—55, 1936.

Alexis Mignon and Frédéric Jurie. PCCA: A new approach for distance learning from sparse pairwise constraints. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2012.

Hossein Mobahi, Ronan Collobert, and Jason Weston. Deep learning from temporal coherence in video. In *International Conference on Machine Learning, ICML*, 2009.

Yair Movshovitz-Attias, Alexander Toshev, Thomas K Leung, Sergey Ioffe, and Saurabh Singh. No fuss distance metric learning using proxies. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 360–368, 2017.

Zak Murez, Soheil Kolouri, David Kriegman, Ravi Ramamoorthi, and Kyungnam Kim. Image to image translation for domain adaptation. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2018.

Timo Ojala, Matti Pietikäinen, and Topi Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence, TPAMI*, (7):971–987, 2002.

Sakrapee Paisitkriangkrai, Chunhua Shen, and Anton van den Hengel. Learning to rank in person re-identification with metric ensembles. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2015.

Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.

Sinno Jialin Pan, James T Kwok, Qiang Yang, et al. Transfer learning via dimensionality reduction. In *AAAI*, volume 8, 2008.

Sinno Jialin Pan, Ivor W Tsang, James T Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2):199–210, 2011.

Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, et al. Deep face recognition. In *British Machine Vision Conference, BMVC*, 2015.

Florent Perronnin and Christopher Dance. Fisher kernels on visual vocabularies for image categorization. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2007.

Florent Perronnin, Jorge Sánchez, and Thomas Mensink. Improving the fisher kernel for large-scale image classification. In *European Conference on Computer Vision, ECCV*. 2010.

Bryan Prosser, Wei-Shi Zheng, Shaogang Gong, and Tao Xiang. Person re-identification by support vector ranking. In *British Machine Vision Conference, BMVC*, 2010.

Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. CNN features off-the-shelf: An astounding baseline for recognition. In *Conference on Computer Vision and Pattern Recognition, CVPR Workshops*, 2014.

Peter M. Roth, Martin Hirzer, Martin Köstinger, Csaba Beleznai, and Horst Bischof. Mahalanobis distance learning for person re-identification. In *Person Re-Identification*, pages 247–267. 2014.

Aruni RoyChowdhury, Tsung-Yu Lin, Subhransu Maji, and Erik Learned-Miller. Face identification with bilinear cnns. *arXiv preprint arXiv:1506.01342*, 2015.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3), 2015.

M Saquib Sarfraz, Arne Schumann, Andreas Eberle, and Rainer Stiefelhagen. A pose-sensitive embedding for person re-identification with expanded cross neighborhood re-ranking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 420–429, 2018.

Cordelia Schmid. Constructing models for content-based image retrieval. In *Proceedings of Computer Vision and Pattern Recognition, CVPR*, volume 2, 2001.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2015.

Matthew Schultz and Thorsten Joachims. Learning a distance metric from relative comparisons. In *Advances in neural information processing systems*, 2004a.

Matthew Schultz and Thorsten Joachims. Learning a distance metric from relative comparisons. In *Advances in neural information processing systems, NIPS*, 2004b.

Edgar Simo-Serra, Eduard Trulls, Luis Ferraz, Iasonas Kokkinos, Pascal Fua, and Francesc Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *International Conference on Computer Vision, ICCV*, 2015.

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

Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in Neural Information Processing Systems, NIPS*, pages 1857–1865, 2016.

Kihyuk Sohn, Sifei Liu, Guangyu Zhong, Xiang Yu, Ming-Hsuan Yang, and Manmohan Chandraker. Unsupervised domain adaptation for face recognition in unlabeled videos. In *International Conference on Computer Vision, ICCV*, 2017.

Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Chi Su, Shiliang Zhang, Junliang Xing, Wen Gao, and Qi Tian. Deep attributes driven multi-camera person re-identification. In *European Conference on Computer Vision, ECCV*, 2016.

Yumin Suh, Jingdong Wang, Siyu Tang, Tao Mei, and Kyoung Mu Lee. Part-aligned bilinear representations for person re-identification. In *European Conference on Computer Vision, ECCV*, 2018.

Yi Sun, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation by joint identification-verification. In *Advances in Neural Information Processing Systems, NIPS*, 2014.

Diana Sungatullina, Egor Zakharov, Dmitry Ulyanov, and Victor Lempitsky. Image manipulation with perceptual discriminators. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 579–595, 2018.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2015.

Oren Tadmor, Tal Rosenwein, Shai Shalev-Shwartz, Yonatan Wexler, and Amnon Shashua. Learning a metric embedding for face recognition using the multibatch method. In *Advances in neural information processing systems, NIPS*, 2016.

Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Conference on Conference on Computer Vision and Pattern Recognition, CVPR*, 2014.

Aruni RoyChowdhury Tsung-Yu Lin and Subhransu Maji. Bilinear cnn models for fine-grained visual recognition. In *International Conference on Computer Vision (ICCV)*, 2015.

Oncel Tuzel, Fatih Porikli, and Peter Meer. Pedestrian detection via classification on riemannian manifolds. *Transactions on pattern analysis and machine intelligence, TPAMI*, 30(10):1713–1727, 2008.

Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014.

Dmitry Ulyanov, Andrea Vedaldi, and Victor S. Lempitsky. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2017.

Evgeniya Ustinova and Victor Lempitsky. Learning deep embeddings with histogram loss. In *Advances in Neural Information Processing Systems, NIPS*, 2016.

Evgeniya Ustinova, Yaroslav Ganin, and Victor Lempitsky. Multi-region bilinear convolutional neural networks for person re-identification. In *IEEE International Conference on Advanced Video and Signal Based Surveillance, AVSS*, 2017.

Rahul Rama Varior, Mrinal Haloi, and Gang Wang. Gated siamese convolutional neural network architecture for human re-identification. In *European Conference on Computer Vision, ECCV*, 2016a.

Rahul Rama Varior, Bing Shuai, Jiwen Lu, Dong Xu, and Gang Wang. A siamese long short-term memory architecture for human re-identification. In *European Conference on Computer Vision, ECCV*, 2016b.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *International conference on Machine learning, ICML*, 2008.

C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. (CNS-TR-2011-001), 2011.

Jun Wang, Alexandros Kalousis, and Adam Woznica. Parametric local metric learning for nearest neighbor classification. In *Advances in Neural Information Processing Systems*, pages 1601–1609, 2012.

Kilian Q Weinberger and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research, JMLR*, 10 (Feb):207–244, 2009.

Kilian Q Weinberger, John Blitzer, and Lawrence K. Saul. Distance metric learning for large margin nearest neighbor classification. In *Advances in Neural Information Processing Systems, NIPS*. 2006.

Lior Wolf, Tal Hassner, and Itay Maoz. Face recognition in unconstrained videos with matched background similarity. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2011.

Chao-Yuan Wu, R Manmatha, Alexander J Smola, and Philipp Krahenbuhl. Sampling matters in deep embedding learning. In *International Conference on Computer Vision, ICCV*, 2017.

Eric P Xing, Michael I Jordan, Stuart J Russell, and Andrew Y Ng. Distance metric learning with application to clustering with side-information. In *Advances in neural information processing systems, NIPS*, 2003.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning, ICML*, 2015.

Dong Yi, Zhen Lei, and Stan Z Li. Deep metric learning for practical person re-identification. *arXiv preprint arXiv:1407.4979*, 2014a.

Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z Li. Deep metric learning for person re-identification. In *International Conference on Pattern Recognition, ICPR*, 2014b.

Yiming Ying and Peng Li. Distance metric learning with eigenvalue optimization. *Journal of Machine Learning Research, JMLR*, 13(Jan):1–26, 2012.

Li Zhang, Tao Xiang, and Shaogang Gong. Learning a discriminative null space for person re-identification. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2016.

Haiyu Zhao, Maoqing Tian, Shuyang Sun, Jing Shao, Junjie Yan, Shuai Yi, Xiaogang Wang, and Xiaoou Tang. Spindle net: Person re-identification with human body region guided feature decomposition and fusion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1077–1085, 2017.

Rui Zhao, Wanli Ouyang, and Xiaogang Wang. Person re-identification by saliency learning. *arXiv preprint arXiv:1412.1908*, 2014.

Liang Zheng, Liyue Shen, Lu Tian, Shengjin Wang, Jingdong Wang, and Qi Tian. Scalable person re-identification: A benchmark. In *International Conference on Computer Vision, ICCV*, 2015a.

Liang Zheng, Liyue Shen, Lu Tian, Shengjin Wang, Jingdong Wang, and Qi Tian. Scalable person re-identification: A benchmark. In *International Conference on Computer Vision, ICCV*, 2015b.

Liang Zheng, Yi Yang, and Alexander G Hauptmann. Person re-identification: Past, present and future. *arXiv preprint arXiv:1610.02984*, 2016.

Wei-Shi Zheng, Shaogang Gong, and Tao Xiang. Person re-identification by probabilistic relative distance comparison. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2011.

Wei-Shi Zheng, Shaogang Gong, and Tao Xiang. Reidentification by relative distance comparison. *IEEE Transactions on Pattern Analysis and Machine Intelligence, TPAMI*, 35(3):653–668, 2013.

Zhun Zhong, Liang Zheng, Zhedong Zheng, Shaozi Li, and Yi Yang. Camera style adaptation for person re-identification. In *Conference on Computer Vision and Pattern Recognition CVPR*, pages 5157–5166, 2018.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *International Conference on Computer Vision, ICCV*, 2017.