Deep Learning Beyond Traditional Supervision

Shixing Chen
Wayne State University, csx1991@gmail.com

Follow this and additional works at: https://digitalcommons.wayne.edu/oa_dissertations

Part of the Computer Sciences Commons

Recommended Citation
https://digitalcommons.wayne.edu/oa_dissertations/2151

This Open Access Dissertation is brought to you for free and open access by DigitalCommons@WayneState. It has been accepted for inclusion in Wayne State University Dissertations by an authorized administrator of DigitalCommons@WayneState.
DEEP LEARNING BEYOND TRADITIONAL SUPERVISION

by

SHIXING CHEN

DISSERTATION

Submitted to the Graduate School

of Wayne State University,

Detroit, Michigan

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

2019

MAJOR: COMPUTER SCIENCE

Approved By:

Advisor Date
DEDICATION

Dedicated to my loving parents for their persistent encouragement and consistent support.
ACKNOWLEDGMENTS

It has been an unforgettable journey for me to go through the PhD program. I could have never completed it without the help and support from so many people that have influenced my life and career in the best way.

To start with, I appreciate my parents. They provided me a loving and nursing family to grow up with, and they always support me to chase my dreams and fulfill my ambition. I would always remember the encouragements from them when I encounter obstacles, and the acclamations they shared with me when I achieved my goals.

I’d like to express my appreciations to Prof. Ming Dong with my deepest respect. As an esteemed advisor, he guided me to explore the charm of science with exceptionally patient attitude and extremely professional perspectives. As a friendly mentor, he gave me so many constructive advices in work and life, which would be beneficial for long.

I would also like to thank my committee members Dr. Dongxiao Zhu, Dr. Abhilash Pandya and Dr. Alexander Kotov. They have provided meaningful inquiries and thoughtful suggestions in this dissertation. Their time and presence for the defense are greatly appreciated.

Many thanks to my co-author and dear friend Dr. Caojin Zhang. He’s been a really helpful companion for years, in work and in life. Many thanks to my colleagues in our lab as well, Dr. Raed Almomani, Dr. Haotian Xu and Hajar Emami Gohari. Your collaborative spirits and passionate discussions are truly helpful and valuable. Special thanks to Dr. Hai Jin and Xinyu Zhang, who gently helped me through some tough times and cordially shared moments of joy with me.

Finally, I’m grateful to US National Science Foundation and Ford Motor Company for their financial support under grants CNS1637312 and University Research Program 2015-9186R, respectively.
TABLE OF CONTENTS

Dedication ii
Acknowledgements iii
LIST OF FIGURES vii
LIST OF TABLES viii

CHAPTER 1 INTRODUCTION 1
  Deep Learning and Ranking Algorithms .............................. 1
  Conventional Machine Learning ........................................ 1
  Deep Neural Networks .................................................. 2
  Ranking Algorithms ..................................................... 3
  Deep Age Ranking ....................................................... 3
  Deep Transfer Learning .................................................. 4
  Transfer Learning ......................................................... 4
  Special Cases ............................................................. 5
  A Unified Framework ......................................................... 6
  Style Transfer and Relation Network ..................................... 6
  Background ................................................................. 6
  An Efficient and Scalable Model ........................................... 7
  Overview ................................................................. 8

CHAPTER 2 Ranking-CNN on Age Estimation 9
  Introduction ............................................................... 9
  Related Work ............................................................. 12
  Age Estimation ........................................................... 12
  Convolutional Neural Networks .......................................... 15
  Convergence .............................................................. 16
  Ranking-CNN for Age Estimation .......................................... 16
  Basic Binary CNNs ....................................................... 17
LIST OF FIGURES

Figure 2.1 Ranking-CNN for facial image-based age estimation. .......................... 10
Figure 2.2 Architecture of a Basic Binary CNN ............................................ 17
Figure 2.3 Representative feature maps extracted from the base CNN. ............. 18
Figure 2.4 Comparison on Cumulative Score with $L$ in $[0, 10]$. .................... 38
Figure 2.5 Precision for each age group after aggregation. ............................. 38
Figure 2.6 Accuracy of each binary ranker in ranking models. ....................... 39
Figure 3.1 The CETL framework. ................................................................. 48
Figure 3.2 Learning curves of CETL. ............................................................ 65
Figure 4.1 Person re-ID in cross-dataset one-shot setting. .............................. 68
Figure 4.2 Optimization pipeline of our model. ............................................ 71
Figure 4.3 The influence of number of ways. ................................................. 81
Figure 4.4 The influence of the ratio between the generated and real images. ... 82
Figure 4.5 Transferred examples from Market and Duke to camera styles in VIPeR. 82
## LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 2.1</td>
<td>Basic CNNs for binary age ranking: architecture and initialization.</td>
<td>33</td>
</tr>
<tr>
<td>Table 2.2</td>
<td>The age and gender information of the 54,362 random samples.</td>
<td>36</td>
</tr>
<tr>
<td>Table 2.3</td>
<td>Comparison of MAE among different combinations.</td>
<td>36</td>
</tr>
<tr>
<td>Table 2.4</td>
<td>T test outcomes of all eight combinations of features and estimators.</td>
<td>38</td>
</tr>
<tr>
<td>Table 2.5</td>
<td>Comparison with other deep learning models on the MORPH dataset.</td>
<td>41</td>
</tr>
<tr>
<td>Table 2.6</td>
<td>Age estimation results on FG-NET dataset.</td>
<td>42</td>
</tr>
<tr>
<td>Table 2.7</td>
<td>Age estimation results on the Adience benchmark.</td>
<td>42</td>
</tr>
<tr>
<td>Table 3.8</td>
<td>The properties of the benchmark datasets adopted in the experiments.</td>
<td>60</td>
</tr>
<tr>
<td>Table 3.9</td>
<td>Comparison of different configurations of CETL.</td>
<td>62</td>
</tr>
<tr>
<td>Table 3.10</td>
<td>Comparison of different source networks.</td>
<td>63</td>
</tr>
<tr>
<td>Table 3.11</td>
<td>Comparison on knowledge distillation.</td>
<td>64</td>
</tr>
<tr>
<td>Table 3.12</td>
<td>Comparison on domain adaptation.</td>
<td>64</td>
</tr>
<tr>
<td>Table 4.13</td>
<td>Comparison on the Market-1501 dataset.</td>
<td>83</td>
</tr>
<tr>
<td>Table 4.14</td>
<td>Comparison on the DukeMTMC dataset.</td>
<td>84</td>
</tr>
<tr>
<td>Table 4.15</td>
<td>Comparison on the VIPeR dataset.</td>
<td>84</td>
</tr>
</tbody>
</table>
CHAPTER 1 INTRODUCTION

Deep learning has become one of the most popular research areas in recent years. Dating back to [88], LeNet was first proposed for document recognition. Then, with the power of high performance computing [78], AlexNet boosted the revolutionary development of Convolutional Neural Networks (CNN), and it has been generalized to numerous applications in computer vision, speech recognition and natural language processing.

However, the necessity of substantial labeled data in most deep learning models has been a long overdue issue needs to be addressed. That is, experimental results have shown that satisfactory performance can be achieved for predefined tasks with general neural networks, but the majority of models are employing fully supervised learning, and can only solve a specific task when massive training data are available. This may cause severe scalability issues and fail to generalize to new application domains. Further, evidence has shown that more efficient alternatives are considered feasible in cognitive studies [119]. Thus, it is rational and natural to investigate in solving problems beyond the traditional supervised settings.

In this dissertation, we focus on exploring deep learning beyond traditional supervision from three different aspects, aiming at exposing some possibilities for this challenging field.

Deep Learning and Ranking Algorithms

In this section, we will briefly introduce our first work of deep learning beyond traditional supervision: ranking with CNN-based model. We will start with the introduction of conventional approaches, explain the rationale and lead to the application of age estimation.

Conventional Machine Learning

Machine-learning technology plays an important role in many aspects of modern society: identifying objects in images, transcribing speech into text, matching new items with users’ interests, selecting relevant results of search, and etc. Conventional machine learning
techniques were limited in their ability to process natural data in their raw form [57] [14]. For decades, constructing a pattern recognition system required careful engineering and considerable domain expertise to design a feature extractor that transforms the raw data (such as the pixel values of an image) into a suitable internal representation or feature vector. Then, a learning subsystem, often a classifier, is employed to detect or classify patterns from the input.

**Deep Neural Networks**

Different from conventional machine learning methods, deep learning provides a representation learning method that allows a machine to be fed with raw data and to automatically discover the representations needed for the specific tasks [88] [62]. It’s composed of multiple layers of nonlinear processing units to extract features from data with multiple levels of abstraction. With the composition of enough such transformations, very complex functions can be learned. It surpassed the performances of other machine learning models in computer vision tasks, such as image recognition, object detection and so on. It already became a promising topic of great attention today.

Deep neural networks can be trained in supervised (e.g., classification) and/or unsupervised (e.g., auto-encoder) manners. Supervised learning is the most common form of machine learning. One first collects a large dataset of images, each labeled with its category. During training, the machine is shown an image and produces an output in the form of a vector of scores, one for each category [29] [77]. We want the desired category to have the highest score of all categories, but this is unlikely to happen before training. We compute an objective function that measures the error (or distance) between the output scores and the desired pattern of scores. The machine then modifies its internal adjustable parameters to reduce this error. In supervised setting, deep learning models are usually trained using certain form of gradient descent via back propagation with large amount of labeled data.
However, in real-world problems, human effort in labeling such large amount of data is very expensive.

**Ranking Algorithms**

Many applications in machine learning are commonly formed as classification or regression problems [55, 77, 87], more than often, it is perfectly fine to do so. However, for some problems with ordinal labels, such assumption can’t be made arbitrarily. That is, multi-class classification can’t take the ordinal information into consideration, and regression normally over-simplifies the problem to a linear model while most problems are generally nonlinear [12]. Thus, cost-sensitive ranking techniques have been introduced [14].

Ranking, usually formalized as “learning to rank”, is the application of machine learning mostly used in the field of information retrieval. Ranking algorithms typically map the inputs to ordinal relationships, and can generally be divided into three categories [103]: pointwise, pairwise, and listwise. In pointwise approaches, the input is usually a single sample with its score being the output. For pairwise ranking, the output of partial order preference is given with sample pairs being the input. As for listwise, input sequence is ranked to reveal the order as the output.

Obviously, there exists a significant difference between classification and ranking. In classification approaches, the categories are regarded separated and not related. While in ranking, it should be taken into consideration that there are clear ordinal relationships among the categories. It is generally not a good idea to abandon this property and downgrade ranking to multi-class classification problems.

**Deep Age Ranking**

For the problem of image-based human age estimation, it is a naturally ordinal setting but was considered as a classification or regression problem before with engineered features [54] [57]. Later, some models began to consider it as a ranking problem with non-parallel hyperplanes [12–14]. More recently, age estimation models based on deep learning algorithms
have attracted great attention [161] [91]. With CNN-based models, effective features are learned during training and superior performances have been achieved [114] [125].

To combine the advantages of deep learning and ranking, we propose ranking-CNN. In terms of supervision, we no longer use categorized labels with softmax loss during training. Instead, binary labels with logistic loss are used while we formulate the ordinal problem as binary sub-problems. In this way, we can keep the ordinal information and achieve better performance.

**Deep Transfer Learning**

As our second work of learning beyond traditional supervision, we consider the machine learning tasks when only limited labels or even no labels are available using transfer learning. Transfer learning is a machine learning method that exploits data or models of related source tasks to boost the performance of the target task with only limited data [115]. Recently, the effectiveness of transfer learning has been validated in deep learning models [46] [101].

**Transfer Learning**

Traditional deep learning approaches can work fairly well when adequate labeled data is available. Transfer leaning, however, deals with the real-world situations when labeled data is very limited or even unavailable. For example, sometimes we may have sufficient labeled data in a source domain but not enough in a target domain. Generally, when there are some similarities between these two domains and the source domain is easy to acquire, the natural idea is to explore whether we can help the performance on the target domain for either discrimination or other tasks [154, 159].

In such cases, transfer learning techniques are preferred and if correctly adopted, the performance could be improved dramatically for classification, regression and clustering problems. For example, if we want to classify some images into 100 categories when there are only 500 samples available for each category [77], it would be helpful if we can take
advantage of some datasets with millions of samples for the task of image categorization as well [29].

Generally, there are several types of transfer that we can do. The most straightforward idea is that if the target domain shares all/part categories with the source domain, we can use these samples directly in the training of target task to help with the performance [42]. However, this is often not the case considering the enormous diversity between tasks and objects. Then, the transfer in terms of features from the source domain is commonly considered and works well as long as the samples in both domains are similar. That is, if they could share some useful features together then the features learned for source should be helpful for target [164]. Moreover, sometimes if the performance is not the highest priority, then speed and flexibility can also be improved by transfer learning [163].

**Special Cases**

Knowledge distillation and domain adaptation are some special cases in transfer learning. For knowledge distillation [65], we consider the problem when source and target domains share the same task but target network possesses a much smaller architecture. In this case, transfer learning is downgraded to a simpler scenario while the challenge is to utilize the output from the source and make it effective for the target. Soft targets and FitNets are some successful examples [65] [124].

Domain adaptation, however, considers a problem when there’s no labeled data at all for the target domain. It is of course a more complicated situation while in fact, it is often the case for real-world applications. In this case, the basic idea would be learning a set of features that can be shared by source and target with the label loss from source and the discrepancy between the tasks. The considerations of reconstruction and generative models are well established for domain adaptation [45] [9].
A Unified Framework

Although great efforts have been put into transfer learning, knowledge distillation and domain adaptation, the field is still challenging with many unsolved problems. For example, now that these tasks are closely related, why can’t we build a unified architecture that can be applied to all these tasks with different settings? Also, do we always have to refer to source data or is fine-tuning the only way to go for the adaption of source networks? To this end, we propose a Coupled End-to-end Transfer Learning (CETL) framework. Comparing with traditional models for transfer learning, we don’t need source data (which can be exponentially larger than target data) and don’t fine-tune the source network (which can be much more complicated than target network). Also, we can adapt CETL to other scenarios including knowledge distillation and domain adaptation, which makes it flexible and efficient.

Style Transfer and Relation Network

For the third work of learning beyond traditional supervision, we leap towards the condition when no labeled data is available. That is, all input samples are not labeled, and the goal is to improve the performance for the scenario of pairwise comparison. Conventional data augmentation approaches aim at increasing labeled data [25], and thus leading to more training samples. Later on, for tasks with unlabeled images, desired synthesized output can be generated from arbitrary input images [67].

Background

Data augmentation is the technique commonly used in machine learning models to reduce overfitting and increase robustness. There are plenty of methods for data augmentation. For example, the easiest way would be adding noise to available data and generating similar samples. More commonly, applying transformation on existing data can improve generalization and performance [25] [45]. Closely related to data augmentation, image synthesis focuses on generating desired types of images based on given styles or textures [67]. Most
recently, generative models represented by [49] demonstrate impressive visual qualities for image generation [70].

Style transfer refers to rendering a content image in a style of another image. It became a topic of attention due to the extensive applications in image synthesis and transfer learning [67]. The concept of instance normalization (IN) is based on batch normalization (BN) to normalize features based on calculated statistics. Later, it was generalized as adaptive instance normalization (AdaIN) and achieved style transfer in real-time [67]. It was proved successful on the style transfer between photos and paints, however, the field is still challenging for unconstrained styles.

Relation network aims at addressing the difficulties in few-shot image recognition tasks [137]. Instead of learning the direct mapping between input images and output labels, the procedure of meta-learning was proposed to determine the relationship between pairs of training samples [37], and this procedure can be generalized to the settings of testing samples.

The task of person re-identification (re-ID) aims at recognizing the same person under variant environment including different backgrounds, cross-camera settings and various lighting conditions. Although great successes have been achieved, there are still two major obstacles restricting the real-world performance of re-ID models. That is, variety of camera styles and limited samples for each identity.

An Efficient and Scalable Model

We propose to address the major obstacles in person re-ID problem by taking advantage of style transfer and few-shot learning approaches. Specifically, our model contains three main modules: Style Transfer (ST), Feature Encoding (FE) and Relation Comparison (RC). In ST, we adopt AdaIN [67] to achieve a fast single-model style transfer between any pair of cameras. The FE module, commonly a deep convolutional network, functions to simply extract image representations in a supervised manner. Finally, the RC module formulate
the person re-ID problem into pair-wise ranking, which compared with classification-based approaches is better suited for re-ID. ST, FE and RC are seamlessly integrated through adversarial training.

**Overview**

In this dissertation, we concentrate on algorithms and models developed to emphasize the concept of deep learning beyond traditional supervision. The rest of this dissertation is arranged as follows. In Chapter 2, we propose a CNN-based ranking framework, ranking-CNN, for the problem of age estimation. In Chapter 3, we propose a coupled end-to-end framework with a coupled loss function for transfer learning. In Chapter 4, we integrate single-model arbitrary style transfer and pairwise comparison in our model through adversarial training with a novel identity preserving loss.
CHAPTER 2 Ranking-CNN on Age Estimation

Human age is considered an important biometric trait for human identification or search. Recent research shows that the aging features deeply learned from large-scale data lead to significant performance improvement on facial image-based age estimation. However, age-related ordinal information is totally ignored in these approaches. In this chapter, we propose a novel CNN-based framework, ranking-CNN, for age estimation. Ranking-CNN contains a series of basic CNNs, each of which is trained with ordinal age labels. Then, their binary outputs are aggregated for the final age prediction. We theoretically obtain a much tighter error bound for ranking-based age estimation. Moreover, we rigorously prove that ranking-CNN is more likely to get smaller estimation errors when compared with multi-class classification approaches. Through extensive experiments, we show that statistically, ranking-CNN significantly outperforms other state-of-the-art age estimation models on benchmark datasets.

Introduction

Human age is considered an important biometric trait for human identification or search. Relying on humans to supply age information from face images is often not feasible [60]. Thus, there has been a growing interest in the automatic determination of the specific age or age range of a subject based on a facial image. Some of the potential applications of automatic age estimation are in law enforcement, security control, and human computer interaction.

One major issue in age estimation models is how to extract effective aging features from a facial image. In the past decade, many efforts have been devoted to aging feature representations. Specifically, simple geometry features (e.g., distances between eyes and nose) and texture features (e.g., skin wrinkles) were first adopted [82]. Later on, Biologically Inspired Features (BIF) [57] were proposed and widely adopted in age estimation applications. More recently, Scattering Transform (ST) [12] was also proposed as an improvement over
BIF by adding filtering routes. Usually, these features can be further enhanced through manifold learning, e.g., Orthogonal Locality Preserving Projection (OLPP) [54].

The other important component in an age estimation model is the estimator. Commonly, age estimation is characterized to be a classification or regression problem. Classification models include $k$ Nearest Neighbors [53], Multilayer Perceptrons [84], and the most commonly used Support Vector Machines (SVM) [57]. For regression methods, quadratic regression [54], Support Vector Regression (SVR) [57] and multi-instance regressor [113] were considered in the literature. More recently, deep learning techniques such as CNN have been applied to human age estimation to learn aging features directly from large-scale facial data [161]. Experimental results show that the deeply-learned aging patterns lead to significant performance improvement on benchmark datasets [151] as well as unconstrained photos [91]. However, multi-class classification completely ignores the ordinal information in age labels, and regression over-simplifies it to a linear model while human aging pattern is generally nonlinear. When humans predict a person's age, it is usually easier to determine if a person is elder than a specific age than directly giving an exact age. Thus, cost-sensitive ranking techniques have recently been introduced to age estimation [12].

In this chapter, we propose a novel age ranking approach based on CNN. Specifically, we propose a ranking-CNN model that contains a set of basic CNNs, each of which has a
sequence of convolutional layers, sub-sampling layers and fully connected layers. Basic CNNs are initialized with the weights of a pre-trained base CNN and fine-tuned with the ordinal age labels through supervised learning. Then, their binary outputs are aggregated to make the final age prediction. Fig. 2.1 shows an illustration of our model. The major contribution of this chapter is summarized as follows:

• To the best of our knowledge, ranking-CNN is the first work that uses a deep ranking model for age estimation, in which binary ordinal age labels are used to train a set of basic CNNs, one for each age group. Different from the regression or the multi-class classification approaches, each basic CNN in ranking-CNN can be trained using all the labeled data, leading to better performance of feature learning and also preventing overfitting. Through extensive experiments, we show that ranking-CNN achieves superior results when compared with other state-of-the-art age estimation methods.

• From a theoretical point of view, we provide a tighter error bound for age ranking than prior work [12], which proved that the final ranking error is bounded by the sum of errors generated by all the classifiers. We divide the errors of sub-problems into two groups: overestimated errors (the sample’s actual label is less than certain age classifier but was classified as older than that age) and underestimated errors (the sample’s actual label is greater than that of certain age classifier but was classified as younger than that age). However, instead of simply aggregating errors, we rearrange them in an increasing order and go deep into the analysis of the underlying differences between any adjacent sub-classifier errors inside each group. By the accumulation of those differences, we theoretically obtain an approximation for the final ranking error, which is controlled by the maximum error produced among sub-problems. From a technical perspective, the new error bound provides very helpful guidance for the training and analysis of ranking-CNN.
Based on the new error bound, we give a Stochastic Gradient Descent (SGD) based scheme to train ranking-CNN in the context of GPU’s high performance computing [8]. We employ stochastic approximation to assert the convergence, in which the parameters are updated as a stochastic process, leading to a limit of Ordinary Differential Equation (ODE) with stationary points that approximate the minimizers of the final ranking loss.

Furthermore, we rigorously derive the expectation of prediction error of ranking-CNN and prove that ranking-CNN, by taking the ordinal relation between ages into consideration, is more likely to get smaller estimation errors when compared with multi-class classification approaches (i.e., CNNs using the softmax function).

This chapter has been published in [19, 21]. The rest of the chapter is arranged as follows. In Section , we briefly review related work in age estimation, CNN, and the convergence analysis. In Section , we first introduce ranking-CNN for age estimation. Then, we establish the theoretical error bound of ranking-CNN and show the convergence of learning ranking CNNs. Finally, we compare ranking-CNN with softmax-based multi-class CNNs and show that ranking method is preferred for age estimation. In Section , we present our age estimation results on the benchmark datasets. Finally, we conclude in Section .

Related Work

Age Estimation

One of the earliest age estimation model can be traced back to [85], in which Active Appearance Model (AAM) [28] was employed to extract shape and appearance features from facial images. Based on these features, various classifiers such as shortest-distance classifier, quadratic function and neural networks were compared. Also, two assumptions were proposed: whether human aging process is age-specific or appearance-specific. That is, whether it is identical for everyone or only people with similar appearance would have similar aging processes.
Earlier works of age estimation usually follow the latter assumption and tend to cluster similar faces before estimation. In [43], the aging process was simulated using AAM for the same individual with a series of age-ascending facial images so that specific models associated with different people’s aging processes can be constructed. Also, to interpret the long-term aging subspace of a person, Geng et al. [44] proposed AGing pattErn Subspace (AGES). AGES is a person-specific age estimation method, which fulfills the estimation by projecting the facial image into the aging subspace with best reconstruction. However, a person’s facial features might be almost identical in some age ranges. To resolve this issue, Zhang et al. [168] employed a warped Gaussian process to model a person’s age, in which both person-specific and general aging information were adopted. In general, it is hard to obtain sufficient data to derive the long-term aging process for every individual. In [138], several short-term patterns, which usually are easier to get, were integrated to construct a long-term aging sequence. More recently, Shu et al. [128] aimed to automatically render aging faces in a personalized way by learning a set of age-group specific dictionaries.

Since the available images for a specific person are typically very limited, many researchers focus on developing non-personalized approaches instead. For instance, Yang and Ai [160] adopted a real AdaBoost algorithm to build a strong classifier from a series of weak ones using Local Binary Patterns [3]. Li et al. [92] proposed a method based on ordinal discriminative feature learning, which preserves locality ordinal information and removes redundancy features. Ni et al. [112] dealt with images with noisy labels through an outlier removal step using PCA and learned a multiple-instance regression estimator. In [57], BIF features were shown to be effective for age estimation on various datasets. Meanwhile, Guo et al. [55] investigated the influence of gender and race on age estimation while Lou et al. [107] introduced a graphical model to jointly learn age and facial expression labels. In [33], Eidinger et al. adopted dropout-SVM on the age estimation of unfiltered faces.

Recently, manifold learning algorithms were incorporated to achieve better performance of age estimation. In [54], Guo et al. proposed to use aging manifold with locally
adjusted robust regressor. Dimension reduction approaches such as Principal Component Analysis (PCA) [32], Locally Linear Embedding [126] and Orthogonal Locality Preserving Projections [11] were employed to learn a low-dimensional embedding. Then, SVR was used together with SVM for data approximation and local adjustment, respectively. Meanwhile, discriminative manifold learning was adopted for better visualization results in [39]. Later, Guo and Mu [56] proposed to use kernel partial least squares regression for simultaneous dimensionality reduction and age estimation.

More recently, CNN-based methods have been widely adopted for age estimation due to its superior performance over existing methods. Yi et al. [161] introduced a multi-task learning method with a relatively shallow CNN. Wang et al. [151] trained a deeper CNN for extracting features from different layers, and the features were then integrated by PCA. Based on these features, age estimation results from different regression and classification approaches were compared. In [125], Rothe et al. adopted the very deep VGG-16 architecture [129] for age estimation. In [104], Liu et al. used two large-scale deep neural networks, and fused the results from classification and regression for better performance. Zhu et al. [180] discussed an apparent age estimation problem with deeply learned features, in which the age labels are annotated by human assessors instead of the real chronological age. In both [91] and [16], CNN’s performance on unconstrained facial images were validated. Hu et al. also considered to train the neural network from the age difference information [66].

Instead of multi-class classification and regression methods, ranking techniques derived from Ranking SVM [63], RankBoost [38, 158] and RankNet [10] were introduced to the problem of age estimation. With the ranking algorithms, the ordinal information of age labels is preserved, and the nature of human aging process is reflected. In [13], the method using ranking algorithms for age estimation was first introduced, in which multiple hyperplanes parallel to each other were used in a single kernel space. Later, a cost-sensitive ordinal ranking framework was proposed with ST features [12], where non-parallel hyperplanes were adopted to allow different kernel spaces for different binary classifiers. Most recently, Niu et
al. [114] proposed to formulate age estimation as an ordinal regression problem with the use of multiple output CNN.

**Convolutional Neural Networks**

There are numerous kinds of CNN models developed in deep learning. The exact forms could vary, but the major components and computations are similar. CNN models derived from LeNet [88] consist of alternating convolutional and pooling layers followed by fully-connected layers with the input to successive layers being the feature maps from previous layers. Weights in layers are updated simultaneously for representative features and classification with a specific loss function through back propagation.

CNNs have been widely used on a variety of applications. In natural language processing, SENNA system has achieved state-of-the-art performance on tasks including language modeling, part-of-speech tagging and semantic role labeling with a convolutional architecture [27]. For text classification, CNN architectures have been widely adopted and achieved superior outcomes [72, 73].

In the computer vision field, CNN models have been applied to various tasks in the past decade. Great successes have been achieved in image classification [61, 78], object detection [46, 149, 167], face recognition [30, 135, 140] and image segmentation [18, 105]. Dating back to LeNet [87, 88], CNN was first introduced to solve the digit recognition problem using the MNIST database. The architecture of LeNet is relatively simple but effective. It contains two convolutional layers followed by two sub-sampling layers and two fully connected layers. The input is handwritten digits [89], and the output is the prediction from the network.

More recently, with the implementation using GPUs [71, 78], CNN models with deep architectures have achieved breakthroughs on object recognition problems in large-scale image datasets, e.g., the ImageNet dataset [29]. Furthermore, to build more effective CNN models, several new components were introduced: activation unit such as rectified linear unit (ReLU) [110] helps to accelerate the convergence during training and has a positive
influence on the performance [78]; regularizer like dropout prevents overfitting by setting some activation units to zero in a specific layer [132]; and batch normalization allows the use of much higher learning rates to make training faster and to improve performance [69].

Convergence

Few theoretical results for the learning algorithm of CNNs is available even though it became one of the hottest topic for machine learning nowadays. Back Propagation (BP), a widely used algorithm for training neural networks, is shown to converge to a local minimum of the least squares error in [79], using an ODE approximation method. Detailed analysis has been gone through to prove the convergence theorem for a BP neural network with a hidden layer in [153]. BP with a momentum (BPM), a variation of BP, aims at improving its convergence speed. Phansalkar and Sastry analyzed the behavior of BPM for a one layer neural network with MAE type loss function in [118] and explains why BPM achieves a faster convergence. SGD is developed to avoid unnecessary work in computing the gradient over the entire dataset and deal with new data in an online setting.

As an online gradient method, convergence of SGD can be proved by stochastic approximation. It was first introduced by Robbins and Monro [123] in the early 1950s. Kushner discussed sufficient conditions for its convergence in his book [81], and then those criterion were adopted in [79] to study adaptive algorithms. Later, more general theory was presented in [80]. In recent years, it has been the subject of an enormous literature, both theoretical and applied, due to the large number of applications and the interesting theoretical issues in the analysis of “dynamically defined” stochastic processes.

Ranking-CNN for Age Estimation

The training of ranking-CNN consists of two stages: pre-training with facial images and fine-tuning with age-labeled faces. First, a base network is pre-trained with unconstrained facial images [33] to learn a nonlinear transformation of the input samples that captures their main variation. From the base network, we then train a set of basic binary
CNNs with ordinal age labels. Specifically, we categorize samples into two groups: with ordinal labels either higher or lower than a certain age, and then use them to train a corresponding binary CNN. The fully connected layers in the binary CNN first flatten the features obtained in the previous layers and then relate them to a binary prediction. The weights are updated through SGD by comparing the prediction with the given label. Finally, all the binary outputs are aggregated to make the final age prediction. In the following, we present our system in details.

**Basic Binary CNNs**

**Architecture and Algorithms** As shown in Fig. 2.2, a basic CNN has a classic architecture: three convolutional and sub-sampling layers, and three fully connected layers. Specifically, C1 is the first convolutional layer with feature maps connected to a $5 \times 5$ neighboring area in the input. There are 96 filters applied to each of the 3 channels (RGB) of the input, followed by ReLU [110]. S2 is a sub-sampling layer with feature maps connected to corresponding feature maps in C1. In our case, we use max pooling on $3 \times 3$ regions with the stride of 2 to emphasize the most responsive points in the feature maps. S2 is followed by local response normalization (LRN) that can aid generalization [78].

C3 works in a similar way as C1 with 256 filters in 96 channels and $5 \times 5$ filter size followed by ReLU. Layer S4 functions similarly as S2, and is followed by LRN. Then, C5 is the third convolutional layer with 384 filters in 256 channels and smaller filter size $3 \times 3$, followed by ReLU.
followed by the third max pooling layer S6. We show the visualization of the feature maps after each layer later in Section.

F7 is the first fully connected layer in which the feature maps are flattened into a feature vector. There are 512 neurons in F7 followed by ReLU and a dropout layer [132]. F8 is the second fully connected layer with 512 neurons that receives the output from F7 followed by ReLU and another dropout layer. F9 is the third fully connected layer and computes the probability that an input $x$ (i.e., output after F8) belongs to class $i$ using the logistic function. Notice that we use the logistic function instead of softmax as the output of a basic CNN is binary. The optimal model parameters of a network are typically learned through minimizing a loss function. We use the negative log-likelihood as the loss function and minimize it using SGD. Detailed analysis on learning and convergence will be given in Section.

![Diagram of CNN layers](image)

Figure 2.3: Representative feature maps extracted from the base CNN.

**Feature Maps** With a single trained CNN, given an input face, we can generate a set of feature maps after each of the convolutional and pooling layers. As our model has three convolutional layers and three pooling layers, we can generate six sets of feature maps in total. The number of feature maps in each set are determined by the number of filters in the corresponding layer.

Representative feature maps extracted from the base CNN are shown in Fig. 2.3. There are six sets of feature maps, i.e., CONV1, POOL1, CONV2, POOL2, CONV3, and POOL3, and we show nine feature maps in each set. Specifically, CONV1 is the set of feature
maps after the first convolutional layer. In CONV1, there are 96 feature maps, showing the convolved results of the input image with 96 filters in layer C1. We can see that the shown nine feature maps are concentrating on different areas of the input face, some of which highlight the eyes and the mouth while others focus on the face contour. After max-pooling layer S2, we can get the corresponding set of feature maps POOL1. Feature maps in POOL1 generally have a higher contrast to pass more information to successive layers.

Then, after the second round convolution, we obtain 256 feature maps in CONV2. Clearly, these feature maps have more detailed information than CONV1 to further depict facial features. Again, the contrast in feature maps in POOL2 are enhanced to be more informative. With the filters in the third convolutional layer C3, 384 feature maps in CONV3 are generated. Now, each feature map in CONV3 concentrate on a certain area to describe the original image in a specific way. After the final pooling layer S6, the output POOL3 with 384 feature maps would be flatten in F7 as the vector to represent the face before age estimation. From these feature maps, we can generally get to know what information has been extracted by the network from the original image.

**Ranking-CNN**

Assume that \( x_i \) is the feature vector representing the \( i \)th sample and \( y_i \in \{1, ..., K\} \) is the corresponding ordinal label. To train the \( k \)-th binary CNN, the entire dataset \( D \) is split into two subsets, with age values higher or lower (or equal to) than \( k \),

\[
D^+_k = \{(x_i, +1)|y_i > k\}, \quad D^-_k = \{(x_i, -1)|y_i \leq k\}.
\] (2.1)

The binary ranking error \( \epsilon(x_i) \) is defined as,

\[
\epsilon_k(x_i) = \left[ f_k(x_i) > 0 \right][y_i \leq k] + \left[ f_k(x_i) \leq 0 \right][y_i > k],
\] (2.2)
where $f_k(x_i)$ is the output of the basic network and $[\cdot]$ denotes the truth-test operator, which is 1 if the inner condition is true, and 0 otherwise. So, $\epsilon_k(x_i) = 1$ if the ranking order is incorrect, and $\epsilon_k(x_i) = 0$ otherwise.

Based on different splitting of $D$, $K - 1$ basic networks can be trained from the base one. Note that in our model, each network is trained using the entire dataset, typically resulting in better ranking performance and also preventing overfitting. Given an unknown input $x_i$, we first use the basic networks to make a set of binary decisions and then aggregate them to make the final age prediction $r(x_i)$,

$$r(x_i) = 1 + \sum_{k=1}^{K-1} [f_k(x_i) > 0].$$  \hspace{1cm} (2.3)

It can be shown that the final ranking error is bounded by the maximum of the binary ranking errors. That is, the ranking-CNN results can be improved by optimizing the basic networks. We mathematically prove this in Section followed by the convergence analysis and theoretical comparison between ranking and softmax-based multi-class classification.

In Algorithm 1, we provide the complete training and testing procedure of ranking-CNN.

**Error Bound** In ranking-CNN, we divide an age ranking estimation problem, ranging from 1, $\cdots$, $K$, into a set of binary classification sub-problems ($K - 1$ classifiers). By aggregating the results of each sub-problem, we then obtain an estimated age $r(x)$. To assure a better overall performance of the model, a key issue is whether the ranking error can be reduced if we improve the accuracy of the binary classifiers. We rigorously address this issue with formal mathematical proof in this section.

Here, we provide a much tighter error bound for age ranking than that introduced in [12], which claims that the final ranking error is bounded by the sum of errors generated by all the classifiers. We adopt the idea in [12] that divides the errors of sub-problems into two groups: overestimated and underestimated errors. However, instead of simply
Algorithm 1 Algorithm of Ranking-CNN

1: procedure Training Procedure
2:   pretrain Base CNN
3:   top:
4:   for $k = 1$ to $K-1$ do
5:     $e_k \leftarrow k_{th}$ Basic CNN
6:   end for
7:   $k' \leftarrow$ sort $e_k$
8:   for $k' = 1$ to $K-1$ do
9:     $D^+_k = \{(x_i, +1) | y_i > k'\}$
10:    $D^-_k = \{(x_i, -1) | y_i \leq k'\}$
11:    fine-tune $k_{th}$ Basic CNN $\leftarrow e_k$
12:   end for
13:   if not converged
14:     goto top
15:   end if
16: procedure Testing Procedure
17:   for $k = 1$ to $K-1$ do
18:     $f_k(x_i) \leftarrow k_{th}$ Basic CNN
19:   end for
20:   final prediction $r(x_i) \leftarrow 1 + \sum_{k=1}^{K-1} [f_k(x_i) > 0]$
aggregating errors, we rearrange them in an increasing order and go deep into the analysis of the underlying differences between any adjacent sub-classifier errors inside each group. By the accumulation of those differences, we theoretically obtain an approximation for the final ranking error, which is controlled by the maximum error produced among sub-problems.

We denote $E^+$ as the total number of sub-classifiers that misclassified when $y \leq k$. That is, $E^+ = \sum_{k=1}^{K-1} \gamma_k^+$, where $\gamma_k^+ = [f_k(x) > 0][y \leq k]$ and $[\cdot]$ is an indicator function taking value of 1 when the condition in $[\cdot]$ holds, 0 otherwise. Similarly, we denote $E^- = \sum_{k=1}^{K-1} \gamma_k^-$ for the case of $y > k$, where $\gamma_k^- = [f_k(x) \leq 0][y > k]$.

For any observation $(x, y)$, we define the cost function (error) for each classifier as:

$$e_k(x) = \begin{cases} e_k^+ = (k - y + 1)\gamma_k^+ & y \leq k \\ e_k^- = (y - k)\gamma_k^- & y > k. \end{cases}$$  \hspace{1cm} (2.4)

Thus, we have a theorem for the error bound of final ranking error:

**Theorem 1.** For any observation $(x, y)$, in which $y > 0$ is the actual label (integer), then the following inequality holds:

$$|r(x) - y| \leq \max_k e_k(x),$$  \hspace{1cm} (2.5)

where $r(x)$ is the estimated rank of age, $k = 1, \cdots, K - 1$. That is, we can diminish the final ranking error by minimizing the greatest binary error.

**Proof**

Denote $e_k(x)$ in (2.4) as $e_k$ for simplicity. We split the proof into two parts. Firstly, we show $|E^+ - E^-| = |r(x) - y|$. Secondly, we demonstrate $\max_k e_k \geq \max\{E^+, E^\}$. By $|E^+ - E^-| < \max\{E^+, E^\}$ for $E^+$ and $E^-$ nonnegative, the inequality (2.5) follows.
Firstly, we begin by definition:

\[ r(x) = 1 + \sum_{k=1}^{K-1} [f_k(x) > 0] \]

\[ = 1 + \sum_{k=1}^{K-1} ([f_k(x) > 0][y \leq k] + [f_k(x) > 0][y > k]) \]

\[ = 1 + E^+ + \sum_{k=1}^{K-1} [f_k(x) > 0][y > k]. \tag{2.6} \]

Subtracting \((E^+ - E^-)\) on both sides, we get

\[ r(x) - (E^+ - E^-) \]

\[ = 1 + \sum_{k=1}^{K-1} [f_k(x) > 0][y > k] + \sum_{k=1}^{K-1} [f_k(x) \leq 0][y > k] \]

\[ = 1 + \sum_{k=1}^{K-1} ([f_k(x) > 0] + [f_k(x) \leq 0])[y > k] \tag{2.7} \]

\[ = 1 + \sum_{k=1}^{K-1} [y > k] \]

\[ = y. \]

Thus \( |r(x) - y| = |E^+ - E^-| \) holds.

Secondly, we extract all \( e_k^+ > 0 \) and rearrange them in an increasing order denoted as a set \( \{e^+_j, j = 1, 2, \cdots, E^+\} \). Similarly, we do the same operation on \( e_k^- \) and have the set \( \{e^-_j, j = 1, 2, \cdots, E^-\} \), where for any random variable \( \xi \), \( \xi(\cdot) \) denotes the order Statistics.

Notice that \( \{e^+_j, j = 1, 2, \cdots, E^+\} \) is a set of losses made by sub-classifiers with incorrect classification, where \( E^+ \) is the total number of sub-classifiers that misclassified when \( y \leq k \). Next, based on the definition of the loss function in (2.4), when \( y \leq k \), the loss associated with a sub-classifier must be greater than 1, i.e., \( e^+_j \geq 1 \). Moreover, the difference of losses between two adjacent classifiers is at least 1, i.e., \( e^+_j - e^+_j - 1 \geq 1 \).

Then, we get:

\[ e^+_{(E^+)} = e^+_1 + e^+_2 - e^+_1 + \cdots + e^+_{(E^+)} - e^+_1 \]

\[ \geq \underbrace{1 + 1 + \cdots + 1}_{E^+} = E^+ \tag{2.8} \]
It follows $e^+_{(E^+)} \geq E^+$. Similarly, we can show $e^-_{(E^-)} \geq E^-$. Then, $\max_k e_k = \max\{e^+_{(E^+)}, e^-_{(E^-)}\} \geq \max\{E^+, E^-\}$, which completes the proof.

**Technical Contribution of the New Error Bound**  Ranking-CNN can be seen as an ensemble of CNNs, fused with aggregation. By showing that the final ranking error is bounded by the maximum error of the binary rankers, we make significant technical contribution in the following aspects:

- Theoretically, it was mentioned in both [12] and [114] that the inconsistency issue of the binary outputs could not be resolved because that would make the training process significantly complicated. The aggregation was just carried out without explicit understanding of the inconsistency. With the tightened error bound, we can confidently demonstrate that the inconsistency doesn’t actually matter because as long as the maximum binary error is decreased, the error produced by inconsistent labels can be ignored. It would neither influence the final estimation error nor complicate the training procedure.

- Methodologically, the tightened bound provides extremely helpful guidance for the training of ranking-CNN. The training of an ensemble of deep learning models is typically very time consuming, especially when the number of sub-models is large. Based on our results, it is technically sound to focus on the sub-models with the largest errors. This training strategy will lead to more efficient training to achieve the desired performance gain. The training strategy can also be extended to ensemble learning with other decision fusion methods.

- Mathematically, based on the new error bound, we can theoretically give an explicit formula for the learning of ranking-CNN and demonstrate its convergence using stochastic approximation method. Moreover, we can rigorously derive the expectation of prediction error of ranking-CNN and prove that ranking-CNN outperforms other softmax-based deep learning models. The detailed proofs are given in following sections.
Learning and Convergence of Ranking-CNN For each ranker \( k \), given a sample \((x, y)\), consider binary target:

\[
r(x) = \begin{cases} 
1 & \text{If } y > k \\
0 & \text{Otherwise}
\end{cases}
\]  

(2.9)

Given the loss function for each ranker as:

\[
\ell(w_k) = |y - k|(-r \log P(r = 1|x) - (1 - r) \log(1 - P(r = 1|x)))
\]

(2.10)

where \( w_i \) denotes the parameters in \( k \)-th ranker.

In training ranking CNN, we implement the Back Propagation (5) using stochastic gradient decent as to minimize the maximum cross entropy loss as:

\[
w_i^{n+1} = \begin{cases} 
w_i^n - \alpha_n \nabla \ell(w_i), & \text{If } \ell(w_i) = \max \ell(w_i) \\
w_i^n, & \text{Otherwise}
\end{cases}
\]

(2.11)

We let the learning rate satisfies:

\[
\sum_n \alpha_n = \infty, \sum_n \alpha_n^2 < \infty, \lim \alpha_n = 0.
\]

(2.12)

Denote

\[
L(w_1, \ldots, w_{K-1}) = \max(\ell(w_1), \ldots, \ell(w_{K-1}))
\]

(2.13)

We concatenate all parameters \( w_i \), for \( i = 1, \ldots, K-1 \) into a vector \( W \) and interpolate the the updated parameters in each iterations as a sequence of stochastic process \( W^n(\cdot) \) as
follows:

\[ t_n = \sum_{i=1}^{n-1} \alpha_i, \quad (2.14) \]

\[ W^0(t) = W^n \quad \text{for} \quad t \in [t_n, t_{n+1}) \quad (2.15) \]

\[ W^n(t) = W^0(t + t_n). \quad (2.16) \]

Then using the stochastic approximation techniques provided in [56], or theorem 5.1 [60], we claim the sequence weakly converges to a limit ODE (convergence in distribution defined as Section II [56]):

**Theorem 2.** Let \( W^n(0) \) be fixed vectors or random vectors independent of \( \alpha_n \). Then \( W^n(\cdot) \) weakly converges to \( W(\cdot) \), where \( W(\cdot) \) satisfy the system of ODEs:

\[ \frac{\partial W(t)}{\partial t} = \frac{\partial E_{x \in D}(L(W))}{\partial W} \quad (2.17) \]

Then the parameters converges to ODEs equilibrium point \((w_1^*, \cdots, w_{K-1}^*)\) (local minimum of the loss \( L \)) by Lyapunov condition [64]. Due to the error bound, we obtain an approximation of the local minimum of the aggregation loss: \( E_{x \in D}|r(x) - y| \).

**Ranking vs. Softmax** In this section, we theoretically show that our ranking-CNN outperforms softmax method because it is more likely to get smaller ranking errors \( |r(x) - y| \). Thus, instead of a softmax classifier, ranking method is preferred for age estimation. The reason is that softmax failed to take the ordinal relation between ages into consideration.
A basic CNN in ranking-CNN differs from the softmax multi-class classification approach in the output layer. Suppose $z_1, \cdots, z_K$ are unnormalized outputs which explains the probability of a sample $x$ belonging to each class. Denote weights $a_i = e^{z_i}$ and $\hat{y}$ as the estimated age label. For softmax, the posterior probability of each class is given by:

$$P(\hat{y} = i | x) = \frac{e^{z_i}}{\sum_{k=1}^{K} e^{z_k}}$$

$$= \frac{a_i}{\sum_{k=1}^{K} a_k}, \quad (2.18)$$

for $i = 1, \cdots, K$. Then, the expected error given the label of the observation $(x, y)$ is

$$E(|r(x) - y||y) = \sum_{i=1}^{K} |i - y| P(\hat{y} = i | x). \quad (2.19)$$

For the ranking-CNN, we use $K - 1$ classifiers to determine ordinal relation between adjacent ages. The posterior probability for a prediction of age greater than a specific age $i$ is given by:

$$P(f_i(x) > 0 | x) = \frac{e^{z_{i+1}}}{e^{z_i} + e^{z_{i+1}}}$$

$$= \frac{a_{i+1}}{a_i + a_{i+1}}, \quad (2.20)$$

The expected error for a given sample is

$$E(|r(x) - y||y) = \sum_{i=1}^{K} |i - y| P(\hat{y} = i | x). \quad (2.21)$$

We present a theorem for a three ordinal class problem.

**Theorem 3.** Suppose we have classes 1, 2 and 3 with weights $a, b, c > 0$ respectively. There exists an ordinal relation: $1 < 2 < 3$. Denote the rank obtained by ranking-CNN as $r_1(x)$ and that by softmax as $r_2(x)$. Then

$$E(|r_1(x) - y|) < E(|r_2(x) - y|). \quad (2.22)$$
Proof. Given a sample with label 1, the expected error for ranking-CNN is

\[
E(|r_1(x) - y| | y = 1) = 2P(f_1(x) > 0, f_2(x) > 0 | W, U, X) \\
+ P(f_1(x) > 0, f_2(x) < 0 | W, X) \\
+ P(f_1(x) < 0, f_2(x) > 0 | W, X) \\
= \frac{2bc + b^2 + ac}{(a + b)(b + c)}. \tag{2.23}
\]

For softmax,

\[
E(|r_2(x) - y| | y = 1) = 2P(r_2(x) = 2 | W, X) \\
+ P(r_2(x) = 3 | W, X) \tag{2.24}
= \frac{2c + b}{a + b + c}.
\]

Similarly, given \( y = 2 \),

\[
E(|r_1(x) - y| | y = 2) = P(f_1(x) > 0, f_2(x) > 0 | W, X) \\
+ P(f_1(x) < 0, f_2(x) < 0 | W, X) \tag{2.25}
= \frac{ab + bc}{(a + b)(b + c)}.
\]

\[
E(|r_2(x) - y| | y = 2) = P(r_2(x) = 1 | W, X) \\
+ P(r_2(x) = 3 | W, X) \tag{2.26}
= \frac{a + c}{a + b + c}.
\]
Given $y = 3$,

\[ E(|r_1(x) - y||y = 3) = 2P(f_1(x) < 0, f_2(x) < 0|W, X) + P(f_1(x) > 0, f_2(x) < 0|W, X) + P(f_1(x) < 0, f_2(x) > 0|W, X) = 2ab + b^2 + ac \]
\[ (a + b)(b + c), \] (2.27)

and

\[ E(|r_2(x) - y||y = 3) = 2P(r_2(x) = 1|W, X) + P(r_2(x) = 2|W, X) = \frac{2a + b}{a + b + c}. \] (2.28)

For ranking-CNN, it follows

\[ E(|r_1(x) - y|) = \sum_{i=1}^{3} E(|r_1(x) - i||y = i) = 2 + \frac{ab + bc}{(a + b)(b + c)}. \] (2.29)

Similarly, for softmax,

\[ E(|r_2 - y|) = \sum_{i=1}^{3} E(|r_2(x) - i||y = i) = 2 + \frac{a + c}{a + b + c}. \] (2.30)

Since

\[ \frac{a + c}{a + b + c} - \frac{ab + bc}{(a + b)(b + c)} = \frac{a^2c + c^2a}{(a + b)(b + c)(a + b + c)} > 0, \] (2.31)
then we conclude

$$E(|r_1(x) - y|) < E(|r_2(x) - y|).$$

(2.32)

Furthermore, the case for \( K = 4, 5, \cdots \) could be shown in a similar way by induction. However, when the number of class \( K \) increases, the analytic expression of the distribution for each class \( i = 1, 2 \cdots K \), becomes

$$P(\hat{y} = i | y) = \sum_{A \in \mathcal{F}_i} \prod_{j \in A} p_j \prod_{j \in A^c} (1 - p_j),$$

(2.33)

satisfying a Poisson-Binomial distribution, where \( p_j = \frac{a_j}{a_{j-1} + a_j} \), \( \mathcal{F}_i \) is the subset of \( i \) integers that could be selected from \( \{1, 2, \cdots, K\} \) and \( A^c \) is the complement of \( A \). Notice that \( \mathcal{F}_i \) represents \( C_K^i \) possible cases. Then, to compute the expected value becomes hopeless since listing all the probability out as we did in theorem 3 looks impractical. Though Le Cam, L. [86] gives an approximation of Poisson-Binomial by a Poisson distribution, the computation for the

$$E(|r_1(x) - y|) = K \sum_{y=1}^{K} \sum_{r=1}^{K} |r - y| P(\hat{y} = r)$$

(2.34)

is not an easy task. To overcome this, statistics provides us a powerful tool with no need for knowing the actual distributions. To further strengthen that ranking-CNN wins over softmax in age estimation, we propose a t-test with hypothesis that compared with softmax, our ranking-CNN does reduce the ranking error in the sense of statistical significance. The details will be discussed later in the experiment section.

**Age Estimation**

When humans predict a person’s age, it is generally easier to determine if a person is elder than a specific age than directly giving an exact age. With ranking-CNN, it provides
a framework for simultaneous feature learning and age ranking based on facial images. The rationale of using ranking-CNN for age estimation is that the age labels are naturally ordinal, and ranking-CNN can keep the relative ordinal relationship among different age groups.

We adopt a general pre-processing procedure for face detection and alignment before feeding the raw data to the networks. Specifically, given an input color image, we first perform face detection using Harr-based cascade classifiers [148]. Then, face alignment is conducted based on the location of eyes. Finally, the image is resized to a standard size of $256 \times 256 \times 3$ for network training and age estimation.

**Experiments**

In this section, we demonstrate the performance of ranking-CNN through extensive experiments. We first choose the appropriate architecture for the basic CNN by evaluating it on binary age ranking problems. Then, we move to multiple age estimation problems and evaluate ranking-CNN.

For multiple age estimation, we compared the features learned by ranking-CNN with the ones obtained through BIF+OLPP [57], ST [12], and multi-class CNN. BIF features are implemented with Gabor filters in 8 orientations and 8 scales and followed by max-pooling. In addition, OLPP is employed to learn the age manifold based on BIF features, in which the top 1,000 eigenvectors are used. In ST, the Gabor coefficients are scattered into 417 routes in two convolutional layers and pooled with Gaussian smoothing. Multi-class CNN is commonly used for age estimation [91, 161], but it completely ignores the ordinal information in age labels. Its structure is similar to a basic CNN (three convolutional and pooling layers and three fully connected layers) with the exception that the last fully-connected layer contains multiple outputs corresponding to the number of ages to be classified instead of the binary one. As for the age estimators, SVM is selected for comparison due to its proved performance [57]. In ranking-based approach (Ranking-SVM), following [12], SVM is used as the binary classifier for each age label and the results are aggregated to give the final
output. Finally, we also directly compare age estimation results obtained by ranking-CNN with the ones reported in the literature by leading deep learners on benchmark datasets.

The comparison and evaluation of different methods in our experiments are reported in terms of precision of each age group, accuracy of each binary ranker as well as two widely adopted performance measures [12,114]: Mean Absolute Error (MAE) and Cumulative Score (CS). MAE computes the absolute costs between the exact and the predicted ages (the lower the better):

$$MAE = \sum_{i=1}^{M} e_i / M,$$

(2.35)

where $e_i = |\hat{l}_i - l_i|$ is the absolute cost of misclassifying true label $l_i$ to $\hat{l}_i$, and $M$ is the total amount of testing samples. CS indicates the percentage of data correctly classified in the range of $(l_i - L, l_i + L)$, a neighbor range of the exact age $l_i$ (the larger the better):

$$CS(L) = \sum_{i=1}^{M} [e_i \leq L] / M,$$

(2.36)

where $[\cdot]$ is the truth-test operator and $L$ is the parameter representing the tolerance range.

Also, we used paired t-test to demonstrate the statistical significance of our empirical comparison. Suppose $\{\epsilon_i\}_{i=1}^{N}$ are the errors obtained through the test set $\{(x_i, y_i)\}_{i=1}^{N}$ by ranking-CNN, and $\{\tau_i\}_{i=1}^{N}$ are errors in testing by another method. We employ paired t-test to determine if the former significantly outperforms the latter. A two-sample t-statistic with unknown but equal variance is computed as:

$$t = \frac{\bar{\epsilon} - \bar{\tau}}{S_{1,2} \sqrt{\frac{2}{n}}},$$

(2.37)
where $\mu_1$ and $\mu_2$ are the mean of two sets of errors respectively, $S_{1,2} = \sqrt{\frac{S_1^2 + S_2^2}{2}}$, and $S_1$, $S_2$ are unbiased estimators of variances of two samples, where

\[
\begin{align*}
S_1^2 &= \frac{1}{N-1} \sum_{n=1}^{N} (\epsilon_n - \mu_1)^2 \\
S_2^2 &= \frac{1}{N-1} \sum_{n=1}^{N} (\tau_n - \mu_2)^2
\end{align*}
\] (2.38)

Define $H_0$: $\mu_1 - \mu_2 = 0$ (the performance of ranking-CNN is not significantly improved), $H_1$: $\mu_2 - \mu_1 > 0$ (otherwise). In the hypothesis test, we compute the $p$ value at 1% significance level. If the $p$ value is small enough, we reject the hypothesis $H_0$.

**Basic CNN on Binary Age Ranking**

Table 2.1: Basic CNNs for binary age ranking: architecture and initialization.

<table>
<thead>
<tr>
<th></th>
<th>20-29 vs. 40-49</th>
<th>&lt;20 vs. &gt;50</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Structure</strong></td>
<td>2+2 2+2 3+3 3+3</td>
<td>2+2 2+2 3+3 3+3</td>
</tr>
<tr>
<td><strong>Weight Initialization</strong></td>
<td>XAVIER GAUSSIAN XAVIER GAUSSIAN</td>
<td>XAVIER GAUSSIAN XAVIER GAUSSIAN</td>
</tr>
<tr>
<td><strong># of Samples</strong></td>
<td>3000 3000 3000 3000</td>
<td>1500 1500 1500 1500</td>
</tr>
<tr>
<td><strong>Accuracy</strong></td>
<td>89.20% 88.13% 93.95% 96.32%</td>
<td>95.35% 94.98% 96.28% 98.72%</td>
</tr>
<tr>
<td></td>
<td>±0.21% ±0.15% ±0.13% ±0.18%</td>
<td>±0.19% ±0.17% ±0.14% ±0.12%</td>
</tr>
</tbody>
</table>

We implemented two architectures for the basic CNN in the GPU mode with Caffe [71], namely, 2+2 and 3+3. For the 2+2 architecture, it is derived from LeNet [88]. It contains two convolutional layers with 20 and 50 filters in each layer respectively, followed by max-pooling layers and two fully connected layers. In the first fully connected layer, there are 500 outputs, and the number of outputs in the second fully connected layer is decided by the number of categories. For the 3+3 architecture, it is similar to our basic CNN shown in Fig. 2.2. It is derived from a simplified version of the ImageNet CNN [78] with fewer layers for higher efficiency [91].
The networks are initialized with random weights generated in two methods. For the weights following Gaussian distribution, the mean is 0, and standard deviation is 0.01. For the Xavier initialization [47], the weights \( W \sim U(-scale, scale) \) follow a uniform distribution with the range inversely proportional to the number of incoming and outgoing nodes:

\[
\begin{align*}
    scale &= \sqrt{3/n} \\
    n &= \frac{fan_{in} + fan_{out}}{2} \\
    fan_{in} &= num_{channel} \times columns_{filter} \times rows_{filter} \\
    fan_{out} &= num_{output} \times columns_{filter} \times rows_{filter}
\end{align*}
\]

where in our case, for example, in the first convolutional layer C1, \( num_{channel} \) is 3, \( num_{output} \) is 96, \( columns_{filter} \) and \( rows_{filter} \) are both 5.

We evaluated the architectures of the networks on two binary age ranking problems: age groups 20-29 vs. 40-49, and age groups <20 vs. >50 on MORPH dataset. MORPH contains 55,134 facial images with the age range from 16 to 77. It provides specific age, gender and ethnicity information for each individual. Based on the availability of samples, we randomly selected 6,000 and 3,000 images from MORPH, respectively, for the two problems. The selection is balanced over age groups. In our experiments, 80% of the data is used for training and the rest 20% for testing (no overlapping with training). The averaged accuracy is reported with standard deviations over 10 runs. In each run, the network is trained using supervised training, and the maximum number of iterations is set at 100,000. We consider the training converges when the change of training error between two adjacent iterations is less than 0.001.

As we can see in Table 2.1, the 3+3 architecture and Gaussian initialization \( N(0, 0.01^2) \) gives the highest classification accuracy in both problems. For the same architecture, Xavier initialization generates comparable results better than all combinations with 2+2 architecture. For 2+2 CNNs, Xavier initialization actually gives higher accuracy than Gaussian.
complex situation (i.e., “20-29 vs. 40-49”), the accuracy decreases dramatically. Since 2 + 2 CNNs are generally trained faster than 3 + 3 CNNs, we can infer that if the problem is not too complicated and computing resource is limited, then 2 + 2 CNNs could still be considered. In our case, since we have to distinguish between adjacent ages, we select the 3 + 3 architecture and Gaussian initialization for best performance. It is used for all the basic networks in ranking-CNN to complete the remaining experiments.

For our hardware settings, we use a single GTX 980 graphics card (including 2,048 CUDA cores), i7-4790K CPU, 32GB RAM, and 2TB hard disk drive. The training time for the base CNN with the selected 3 + 3 architecture is around 6 hours. Fine-tuning takes about 20 to 30 minutes for each basic CNN. Totally, it takes about 30 hours to pre-train the base CNN and fine-tune 50 basic CNNs.

Multiple Age Estimation

In this section, we evaluate the performance of Ranking-CNN on three benchmark datasets: MORPH Album 2 [121], FG-NET [1] and Adience Faces benchmark [33].

MORPH  To further demonstrate the performance improvement of ranking-CNN, we consider the age estimation problem in the range between 16 and 66 years old on the most commonly used age estimation benchmark dataset MORPH Album 2, and compare ranking-CNN with other state-of-the-art feature extractors and age estimators. First, we pre-train a base network with 26,580 image samples from the unfiltered faces dataset [33]. The age group labels for these images are used in training as surrogate labels [31]. Then, we fine-tune our ranking-CNN model on MORPH.

In our experiments, when fine-tuning from the pre-trained base CNN to basic CNNs, we set the learning rate for the last fully-connected layer 10 times of the one used in the previous layers. Thus, the majority of the weights in the basic CNNs has only a slight difference, all similar to the ones in the base CNN. In principle, this training procedure works similarly as weight sharing, but with the additional benefit of easier parallelization.
That is, the 50 basic CNNs can be fine-tuned parallelly on a distributed computing platform, while traditional weight-sharing has to be done sequentially.

Following the settings used in some recent work on age estimation [14, 17, 114, 151], we randomly select 54,362 samples in the age range between 16 and 66 from the MORPH dataset. The age and gender information of the selected samples is shown in Table 2.2. Note that these images are not used in the pre-training stage. All the selected samples are then divided into two sets: 80% of the samples are used for basic networks training and the rest 20% samples for testing. There is no overlapping between the training and testing sets, and we repeat 10 independent runs to evaluate the performance during experiments.

Table 2.2: The age and gender information of the 54,362 random samples.

<table>
<thead>
<tr>
<th></th>
<th>&lt;20</th>
<th>20-29</th>
<th>30-39</th>
<th>40-49</th>
<th>&gt;50</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Male</td>
<td>6543</td>
<td>13849</td>
<td>12322</td>
<td>9905</td>
<td>3321</td>
<td>45940</td>
</tr>
<tr>
<td>Female</td>
<td>829</td>
<td>2291</td>
<td>2886</td>
<td>1975</td>
<td>441</td>
<td>8422</td>
</tr>
<tr>
<td>Total</td>
<td>7372</td>
<td>16140</td>
<td>15208</td>
<td>11880</td>
<td>3762</td>
<td>54362</td>
</tr>
</tbody>
</table>

As there are 51 age groups in this age range, 50 binary rankers are needed for ranking approaches (i.e., ranking-CNN and ranking-SVM). In our experiments, 43,490 samples (80% of all the randomly selected samples) with binary labels are selected to train each basic network or SVM in ranking-CNN and ranking-SVM, respectively. The exactly same set of samples with multi-class labels are used to train multi-class CNN and SVM, respectively. The rest 10,872 samples were used for testing results. All experiments are repeated with 10 independent runs.

Table 2.3: Comparison of MAE among different combinations.

<table>
<thead>
<tr>
<th></th>
<th>Engineered Features</th>
<th>Learned Features</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BIF + OLPP ST</td>
<td>CNN Feature Ranking-CNN Feature</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Classification</td>
<td>SVM</td>
<td>4.99 ± 0.035 5.35 ± 0.040</td>
<td>3.05 ± 0.028 3.65 ± 0.028</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Model</td>
<td>Multi-class CNN</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ranking</td>
<td>Ranking-SVM</td>
<td>5.03 ± 0.028 4.88 ± 0.030</td>
<td>-</td>
<td>3.63 ± 0.019</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Model</td>
<td>ranking-CNN</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2.96 ± 0.015
Basically, we have three sets of features: engineered features (i.e., BIF+OLPP and ST), learned classification features (multi-class CNN) and learned ranking features (ranking-CNN). CNN feature and ranking-CNN feature are the output after layer F8 of multi-class CNN and Ranking-CNN respectively. Also, we have two sets of age estimators: classification methods (i.e., SVM and Multi-class CNN) and ranking methods (ranking-CNN and ranking-SVM). We report MAE of all possible combinations of feature extractors and age estimators (eight in total) in Table 2.3. A dash in the table means that the selected feature set is not applicable to the selected estimator.

As shown in Table 2.3, ranking-CNN with its features achieves the lowest MAE of 2.96±0.015 in all the combinations. Ranking-CNN features with Ranking-SVM achieves the second best MAE result, and this validates the effectiveness and generality of ranking-CNN features. In comparison, the lowest MAE achieved by the learned classification features is 3.65±0.028. Note the multi-class CNN represents the commonly used CNN-based age estimation methods [91, 161]. Our experimental results strongly support the theoretical results (ranking vs. softmax) we presented before. Another fact we can see is that the performance of CNN-based features gets weakened when combined with SVM-based estimators. The lowest MAE achieved by engineered features is 4.88±0.030 by ST+ranking-SVM. Notice that ST works better with ranking-SVM, and BIF+OLPP works better with SVM. This could be caused by the fact that in the literature specific features were manually selected for certain estimators to achieve the best performance.

The comparison in terms of CS of the eight combinations of features and estimators are given in Fig. 2.4. Clearly, ranking-CNN outperforms all others across the entire range of $L$ (age error tolerance range) from 0 to 10. Specifically, Ranking-CNN can reach the accuracy of 89.90% for $L = 6$, and 92.93% for $L = 7$. The other fact we notice is that four CNN-based methods reach a higher accuracy for $L = 10$ than the others.
Figure 2.4: Comparison on Cumulative Score with $L$ in $[0, 10]$.

Table 2.4: T test outcomes of all eight combinations of features and estimators.

<table>
<thead>
<tr>
<th></th>
<th>RANKING-CNN</th>
<th>RANKING-CNN FEATURE</th>
<th>ST+RANKING-SVM</th>
<th>BIF+OLPP+RANKING-SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>RANKING-CNN</td>
<td>NaN</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>RANKING-CNN FEATURE</td>
<td>6.36e$^{-148}$</td>
<td>NaN</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>+ RANKING-SVM</td>
<td>0</td>
<td>0</td>
<td>NaN</td>
<td>1</td>
</tr>
<tr>
<td>ST+RANKING-SVM</td>
<td>0</td>
<td>0</td>
<td>1.79e$^{-135}$</td>
<td>NaN</td>
</tr>
<tr>
<td>BIF+OLPP+RANKING-SVM</td>
<td>0</td>
<td>0</td>
<td>1.94e$^{-121}$</td>
<td>2.00e$^{-4}$</td>
</tr>
<tr>
<td>MULTI-CLASS CNN</td>
<td>0</td>
<td>0.14</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>CNN FEATURE+SVM</td>
<td>4.12e$^{-276}$</td>
<td>8.90e$^{-184}$</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>ST+SVM</td>
<td>0</td>
<td>0</td>
<td>1.94e$^{-121}$</td>
<td>2.00e$^{-4}$</td>
</tr>
<tr>
<td>BIF+OLPP+SVM</td>
<td>0</td>
<td>0</td>
<td>4.56e$^{-90}$</td>
<td>0.18</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>MULTI-CLASS CNN</th>
<th>CNN FEATURE+SVM</th>
<th>ST+SVM</th>
<th>BIF+OLPP+SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>RANKING-CNN</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>RANKING-CNN FEATURE</td>
<td>0.85</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>+ RANKING-SVM</td>
<td>0</td>
<td>0</td>
<td>1.99</td>
<td>0.81</td>
</tr>
<tr>
<td>ST+RANKING-SVM</td>
<td>0.14</td>
<td>0</td>
<td>0.99</td>
<td>0.81</td>
</tr>
<tr>
<td>BIF+OLPP+RANKING-SVM</td>
<td>NaN</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>MULTI-CLASS CNN</td>
<td>5.43e$^{-24}$</td>
<td>NaN</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>CNN FEATURE+SVM</td>
<td>0</td>
<td>0</td>
<td>NaN</td>
<td>3.66e$^{-6}$</td>
</tr>
<tr>
<td>ST+SVM</td>
<td>0</td>
<td>0</td>
<td>NaN</td>
<td>3.66e$^{-6}$</td>
</tr>
<tr>
<td>BIF+OLPP+SVM</td>
<td>0</td>
<td>0</td>
<td>0.99</td>
<td>NaN</td>
</tr>
</tbody>
</table>

Figure 2.5: Precision for each age group after aggregation.
In addition, in Fig. 2.5, we compare the estimation precision of each age category for the eight combinations. The precision is defined as below:

\[
\text{precision} = \frac{\text{samples}_{\text{correct}} \cap \text{samples}_{\text{all}}}{\text{samples}_{\text{all}}} \tag{2.40}
\]

where \( \text{samples}_{\text{correct}} \) denotes the samples correctly classified to a certain age category, and \( \text{samples}_{\text{all}} \) denotes the total number of samples classified to this age category. It is obvious that ranking-CNN has a more consistent performance on each of the age groups. For methods like BIF+OLPP+SVM, there are many age categories with 0 precision. Taking a closer look, we found out that this is caused by the unbalanced classification of the multi-class estimators, where samples are mostly classified to certain categories instead of all the categories. The problem can be alleviated by ranking-CNN to some extent. In fact, none of the methods reach the age categories after 54. This is mainly because comparing with more than 50K samples in total, there are too few samples for age categories after 54 (averagely around 80 samples in each age category).

In Fig. 2.6, we further compared the four ranking-based methods and report their performance on each binary ranker. Again, ranking-CNN demonstrates a consistent outstanding performance throughout all binary problems. Note that when the data for the binary rankers are not balanced (and thus higher baseline accuracy, e.g., age < 20 and age > 48), all rankers seem to perform quite well. However, when it comes to the age range with more balanced...
data (and thus lower baseline accuracy, age 20 – 48), the superior performance of ranking-CNN is shown, and this would lead to better overall performance of age estimation. Again, our results clearly illustrated the remarkable improvement of using ranking-CNN for age estimation.

To demonstrate that the experimental results we obtained do not happen simply by chance, we report in Table 2.4 the p-values from paired t-test. We report the p values of the paired t-test at significant level 1%. In Table 2.4, if $p < 0.01$, we reject the null hypothesis. Otherwise, we don’t. For example, when comparing “ranking-CNN” with “ranking-CNN feature+ranking SVM”, the p-value $6.36 \times 10^{-148}$ is much less than 0.01, which means that we reject the null hypothesis that “the performance of ranking-CNN is not significantly improved”. The “NaN” in the table means we could not compare a method with itself. As we can see, statistically, ranking-CNN significantly outperforms all other methods, which implies if we repeat the experiments for numerous times, then in 99% of those experiments, ranking-CNN would outperform. From the table, Ranking-CNN Feature+Ranking SVM and the Multi-Class CNN tied for the second place, followed by CNN Feature+SVM. ST+Ranking SVM stands out among the engineered feature-based methods. Lastly, BIF+OLPP+Ranking-SVM ties with BIF+OLPP+SVM, and ST+SVM has no significant improvement than any other methods.

Furthermore, in Table 2.5, we compare ranking-CNN with other deep learning-based age estimation models, i.e., Ordinal Regression with CNN (OR-CNN) [114], Metric Regression with CNN (MR-CNN) [114], Deep EXpectation (DEX) [125] and GoogLeNet in [66]. Since all the experiments are carried out on the MORPH dataset and we followed the same setting for data partition, we can directly compare the MAE of Ranking-CNN with the ones obtained by these deep learners. Notice that in order to make a fair comparison among all the deep learners, all existing results are reported without pre-training using additional facial images. For ranking-CNN, results with and without pre-training are both reported. Clearly, ranking-CNN outperforms these deep learning models in both cases.
Finally, we show the efficiency brought by the new error bound with a modified training strategy. According to [12] and our experiment results, the basic CNNs between age groups 36 and 45 get the largest training errors as they have more balanced training data (and thus lower baseline accuracy). For these basic CNNs, we train them until the change of the training errors between two adjacent iterations is less than 0.001. For all other basic CNNs (16 to 35 and 46 to 65), we only train them until the change is less than 0.01. In this experiment setting, 80% of the basic CNNs are trained with dramatically less epochs (60% less on average), leading to much faster training. Yet, we still achieved very competitive results on age estimation, an MAE of 3.07±0.017.

Table 2.5: Comparison with other deep learning models on the MORPH dataset.

<table>
<thead>
<tr>
<th>Model</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ranking-CNN</td>
<td>2.96±0.015</td>
</tr>
<tr>
<td>MR-CNN [114]</td>
<td>3.27±0.14</td>
</tr>
<tr>
<td>OR-CNN [114]</td>
<td>3.34±0.28</td>
</tr>
<tr>
<td>DEX [125]</td>
<td>3.25</td>
</tr>
<tr>
<td>GoogLeNet [66]</td>
<td>3.13</td>
</tr>
<tr>
<td>Ranking-CNN (without pre-training)</td>
<td>3.03±0.018</td>
</tr>
</tbody>
</table>

**FG-NET** The FG-NET dataset is another benchmark dataset for age estimation. Since there are merely 1,002 photos in this dataset, it is not suitable for direct training of deep learners. Thus, we evaluate the performance on this dataset by fine-tuning the ranking-CNN model trained on the MORPH dataset. The age range we considered in Section is 16 to 66, so we select the 405 samples from FG-NET in the same range for this experiment. Similarly, we use 80% of these samples for training and 20% for testing and compare the MAE results with prior arts.

As shown in Table 2.6, ranking-CNN outperforms other models on FG-NET dataset as well, and achieves the lowest MAE of 4.13. This further demonstrates the effectiveness and generalization ability of ranking-CNN. CSOHR achieves the second best MAE result...
Table 2.6: Age estimation results on FG-NET dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ranking-CNN</td>
<td>4.13</td>
</tr>
<tr>
<td>DEX [125]</td>
<td>4.63</td>
</tr>
<tr>
<td>CSOHR [12]</td>
<td>4.48</td>
</tr>
<tr>
<td>BIF+OLPP+SVM [57]</td>
<td>4.77</td>
</tr>
<tr>
<td>RankBoost [158]</td>
<td>5.67</td>
</tr>
</tbody>
</table>

of 4.48 while the MAE of DEX is 4.63. For BIF+OLPP+SVM and RankBoost, the MAE results are 4.77 and 5.67 respectively.

Adience There are 26,580 photos in the Adience benchmark dataset of unfiltered faces. The samples are categorized into eight age groups with labels “0-2, 4-6, 8-13, 15-20, 25-32, 38-43, 48-53 and over 60”, so we train seven basic CNNs for this task. Following the same settings in some recent work [16, 33, 91], we randomly select 80% of the samples as the training set and the rest 20% as the testing set.

Table 2.7: Age estimation results on the Adience benchmark.

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ranking-CNN</td>
<td>53.7±4.4</td>
</tr>
<tr>
<td>CNN [91]</td>
<td>50.7±5.1</td>
</tr>
<tr>
<td>Cascaded CNN [16]</td>
<td>52.88±6</td>
</tr>
<tr>
<td>Dropout-SVM [33]</td>
<td>45.1±2.6</td>
</tr>
</tbody>
</table>

As shown in Table 2.7, the mean accuracy±standard error over all the age categories by ranking-CNN are compared with several results recently reported in the literature. It is obvious that ranking-CNN outperforms other methods and achieves the highest accuracy of 53.7±4.4 for age categorization on Adience. Other CNN-based models also achieve good results. The accuracy of Cascaded CNN is 52.88±6, and the multi-class CNN which has the architecture similar to the base CNN in ranking-CNN achieved 50.7±5.1. The dropout-SVM method has the lowest accuracy of 45.1±2.6 among the compared models.
Conclusion

In this chapter, we proposed ranking-CNN, a novel deep ranking framework for age estimation. Our model contains a set of basic CNNs, each of which is initialized with the pre-trained base CNN and fine-tuned with ordinal labels. The binary output of basic CNNs are aggregated to make the final age prediction. From a theoretical perspective, we established a much tighter error bound for ranking-based age estimation, based on which, we mathematically proved the convergence of SGD-based training of ranking-CNN using a novel stochastic approximation approach and rigorously showed that ranking-CNN, by taking the ordinal relation between ages into consideration, is more likely to get smaller estimation errors when compared with multi-class classification approaches. Through extensive experiments, we show that ranking-CNN outperforms other state-of-the-art age estimation methods on benchmark datasets.
CHAPTER 3 Coupled End-to-end Transfer Learning

In transfer learning, one seeks to transfer related information from source tasks with sufficient data to help with the learning of target task with only limited data. In this chapter, we propose a novel Coupled End-to-end Transfer Learning (CETL) framework, which mainly consists of two convolutional neural networks (source and target) that connect to a shared decoder. A novel loss function, the coupled loss, is used for CETL training. From a theoretical perspective, we demonstrate the rationale of the coupled loss by establishing a learning bound for CETL. Moreover, we introduce the generalized Fisher information to improve multi-task optimization in CETL. From a practical aspect, CETL provides a unified and highly flexible solution for various learning tasks such as domain adaption and knowledge distillation. Empirical result shows the superior performance of CETL on cross-domain and cross-task image classification.

Introduction

In computer vision, deep learning models such as Convolutional Neural Networks (CNNs) have successfully been applied to analyzing images, e.g., ImageNet [78], and achieved superior performance than other machine learning methods. However, such advances are often on account of the availability of a large amount of labeled training data. In many cases, manually labeling data can be very expensive, and when the labeled data is limited, CNN’s performance will be compromised.

Transfer learning provides a framework to address this challenge. In transfer learning, one seeks to transfer related information from source tasks with sufficient data to help with the learning of target task with only limited data [115]. Recently, the ability to learn and transfer representations in CNN models has been shown to be important and effective [46, 142]. In [164], the transferability of features from various layers in neural networks was discussed. More recently, in [101], several factors (including width, depth, density, etc.) affecting the transferability for CNNs were compared.
As a special case of transfer learning, domain adaptation considers the problem when no labels of the target domain are available. It assumes that only source domain is labeled, and source and target domains have different distributions (domain discrepancy) but share the same task [115]. In recent years, various works [40, 45, 48, 50, 106] attempt to address the domain adaptation problem for deep CNNs. Usually, the domain discrepancy is modeled using Kullback-Leibler divergence or Maximum Mean Discrepancy (MMD). Then, a target domain network is fine-tuned from the source network by jointly minimizing source domain classification errors and the domain discrepancy. However, due to the relative low model accuracy and extra optimization procedures, domain adaptation remains a challenging research problem.

Knowledge distillation [65] can be considered as another special case of transfer learning, in which the knowledge from a teacher CNN is transferred to a much more concise student CNN by emulating teacher’s soft-targets (a variation of softmax outputs). In this setting, teacher and student networks share the same data distribution and classification objectives. Later, FitNets [124] was proposed to include the transfer between intermediate feature maps of CNNs to improve the performance of the student CNN.

In this work, we propose a Coupled End-to-end Transfer Learning (CETL) framework to transfer the knowledge between CNNs for related tasks, and address the issues caused by domain discrepancy. Our major contributions are summarized as follows:

- CETL provides a unified transfer learning solution that can also be adapted for knowledge distillation and domain adaptation tasks, while prior work typically only considers one of these problems. In addition, through its novel architecture, CETL has great flexibility on the choice of the source network and on the architecture of the target network.

- Different from most prior work on transfer learning, the training of CETL neither uses the source data nor directly tunes on the source network. From a computation
perspective, this is critical as the source dataset is usually large, and the pre-trained source network can be very big, both leading to a long training time.

- We propose a novel loss function, the coupled loss, for CETL training. From a theoretical point of view, we demonstrate the rationale of the new loss function by establishing a learning bound for CETL.

- We introduce the Generalized Fisher Information (GFI) to improve multi-objective optimization in CETL. GFI conducts a dynamic allocation of shared and private weights for multi-tasks to overcome the catastrophic forgetting and preserve useful parameters for the new task. Empirical result shows the superior performance of CETL on cross-domain and cross-task image classifications.

This chapter has been published in [20]. The rest of this chapter is arranged as follows. In Section , we briefly review related work in transfer learning and its applications on image classification. In Section , we introduce the architecture of CETL, give the definition of GFI, and demonstrate the theoretical soundness of the coupled loss employed in CETL training. In Section , we present our image classification results on benchmark datasets. Finally, we conclude in Section .

**Related Work**

Deep CNNs achieved state-of-the-art performance in a wide range of tasks and applications in computer vision. However, in supervised learning of a CNN, a large amount of labeled data is necessary, or the model may encounter generalization issues. Thus, how to transfer useful knowledge from a source network to boost the performance of a target network with limited labeled data becomes an important research topic. In transfer learning [115], we aim to learn a new task in a domain of interest called target domain when we only have sufficient data to learn a similar but different task on a source domain with different data distribution. A learning bound was introduced by [6], which claimed the error of target task is bounded by the sum of the error of the task on source and the domain discrepancy.
The research of transfer learning on deep CNN emerged recently. Yosinski et al. [164] gave one of the earliest empirical study about the feature transferability in various layers of CNN. Littwin et al. [101] proposed a framework to transfer the source data representation learned using a set of orthogonal class classifiers. Azizpour et al. [4] discussed several factors influencing the transferability of features learned by CNN.

Knowledge distillation can be considered as a special case of transfer learning, in which the features learned by a teacher network are exploited to improve the performance of a relatively concise student network for the same task. Hinton et al. [65] adopted soft-targets to distill knowledge from a series of ensemble of CNNs into a single model. Following Hinton’s work, Romero et al. added a difference loss between two intermediate layers to improve the performance [124]. In [163], Yim et al. defined the distill knowledge as the Flow of Solution Procedure (FSP) matrix where the training of the student network was implemented by mimicking the FSP matrices generated by the teacher.

For domain adaptation, prior work focuses on improving deep learning models when domain discrepancy arises. A direct way is to reweigh or select samples from the source domain that are similar to the ones in the target domain [42, 48]. Rendering synthetic data is an alternative. Recently, Bousmalis et al. [9] adopted the Generative Adversarial Networks to transform source images into the target style. Most deep domain adaptation works resolve the training problem by jointly minimizing the source label classification errors and the domain discrepancy. Ganin and Lempitsky [40] addressed domain discrepancy by training a CNN that minimizes the loss of label classification while maximizing the loss of a domain classifier in an end-to-end style. Weighted Maximum Mean Discrepancy (WDA) [157] was proposed later to take class weight bias into account. Tzeng et al. [143] proposed the Adversarial Discriminative Domain Adaptation (ADDA) method, where the label classifier and domain classifier are trained separately in an adversarial manner.

The proposed CETL framework is motivated by two considerations. The first one is to gradually tweak feature representations through target data reconstruction to minimize
domain disparity. As in [23], Chopra et al. mitigated the domain discrepancy by layer-wise pre-training a CNN using a series of autoencoders. Later, Ghifary et al. [45] designed the model combining a traditional CNN for source label prediction with a convolutional autoencoder for target data reconstruction. The second is that the non-linear mapping between cross-modal data provides helpful deep feature representation for robust object detection with various backgrounds. For example, Xu et al. addressed the pedestrian detection problem under adverse illumination conditions, in which they exploited features in the non-linear mapping from RGB image to its corresponding thermal data [156]. Mao et al. [108] proposed a HyperLearner, which is an architecture that reconstructs various channel features (e.g., apparent-to-semantic features, temporal features and depth features) as well as performs pedestrian detection. In CETL, by the multi-task of simultaneous classification and reconstruction, a pre-trained source network exploits the target data for cross domain feature generation. Further, it is coupled with the target network to reconstruct those features as well as perform classification on the target data.

**Coupled End-to-end Transfer Learning**

In this section, we provide details on CETL. First, we show the architecture of CETL and explain the learning procedure with the coupled loss. Then, GFI is introduced for dynamic allocation of shared and private weights in multi-task learning. The theoretical
analysis for the coupled loss follows immediately. Last, we show how CETL can be adapted to various tasks of transfer learning, knowledge distillation and domain adaptation.

The Architecture

As shown in Fig. 3.1, CETL mainly consists of two CNNs with softmax outputs (source and target) that connect to a shared decoder $T_1$ containing deconvolution and unpooling layers for reconstruction. The pre-trained source CNN, denoted as $S$, aims at extracting cross domain features. The detailed steps of training in CETL are given by the numbers in the figure. Specifically, by passing the target data through $S$, we obtain the feature maps from each layer in $S$. Then, by connecting a specific layer in $S$ to a reversed target CNN $T_1$, we consider $S$ as an encoder and $T_1$ as a decoder. In training, we update the weights in $T_1$ with the reconstruction loss while keeping the weights in $S$ unchanged. Since the feature maps in $S$ reflect the activations of the source CNN with the input of target data, by decoding these feature back into the input space, $T_1$ is updated to represent the weights encoded in $S$ in a backward manner.

Denote the datasets of target and source domains as $D_{src} = \{x_{src}, y_{src}\}$ and $D_{tgt} = \{x_{tgt}, y_{tgt}\}$ with distribution $P$ and $Q$, respectively. The source CNN $S$ is pre-trained using a supervised cross entropy loss based on the source data:

$$L^s_c(S(\theta_S), x_{src}, y_{src}) = \frac{1}{N} \sum_{i=1}^{N} \log P(y_{src}^i | x_{src}^i, \theta_S)$$  \hspace{1cm} (3.1)

where $\theta_S$ denotes the model parameters while $T_1$ is trained using an unsupervised reconstruction loss on the target data:

$$L^s_r(T_1(\theta_{T_1}), x_{tgt}) = |T_1 \circ S(x_{tgt}) - x_{tgt}|^2$$  \hspace{1cm} (3.2)
By doing so, we find an underlying feature representation across two datasets $D_{src}$ and $D_{tgt}$. It is decoded in the reconstruction $T_1 \circ S(x)$, which is a resemblance to the channel features in [108].

The target CNN denoted as $T_2$ is also connected with decoder $T_1$ to conduct the coupled learning, in which the following combined loss is minimized:

$$\lambda L^c_c(T_2(\theta_{T_2}), x_{tgt}, y_{tgt}) + (1 - \lambda)L^c_r(T_1(\theta_{T_1}), T_2(\theta_{T_2}), x_{tgt})$$  \hspace{1cm} (3.3)

where

$$L^c_c(T_2(\theta_{T_2}), x_{tgt}, y_{tgt}) = \frac{1}{N} \sum_{i=1}^{N} \log P(y^i_{tgt} | x^i_{tgt}, \theta_{T_2})$$  \hspace{1cm} (3.4)

is the classification loss on the target data, and

$$L^c_r(T_1(\theta_{T_1}), T_2(\theta_{T_2}), x_{tgt}) = |T_1 \circ T_2(x_{tgt}) - T_1 \circ S(x_{tgt})|^2$$  \hspace{1cm} (3.5)

is the loss of reconstructing output of $T_1 \circ S$ using $x_{tgt}$ as the input.

**Learning:** The classifier of $S$ is pre-trained using $D_{src}$. First, we train the $T_1 \circ S$ using unlabeled $D_{tgt}$. Then, we train the target network $T_2$ by optimizing the combination of classification loss $L^c_c(T_2(\theta_{T_2}), x_{tgt}, y_{tgt})$ using labelled $D_{tgt}$ and the reconstruction loss $L^c_r(T_1(\theta_{T_1}), T_2(\theta_{T_2}), x_{tgt})$ using all $D_{tgt}$. Alternatively, CETL can be trained in an end-to-end style. That is, we train the coupled networks using sum of all losses (coupled loss) simultaneously:

$$L_{coupled} = \lambda_1 L^c_c(T_1(\theta_{T_1}), x_{tgt}) + \lambda_2 L^c_c(T_2(\theta_{T_2}), x_{tgt}, y_{tgt})$$
$$+ \lambda_3 L^c_r(T_1(\theta_{T_1}), T_2(\theta_{T_2}), x_{tgt})$$  \hspace{1cm} (3.6)

where $\lambda_i (i = 1, 2, 3)$ denotes the constant weights. We will demonstrate the rationale of the coupled loss in Section .
Generalized Fisher Information

We introduce GFI as a novel contribution for transfer learning in this section. Different from previous work [74], we take the correlation of two tasks into account and dynamically allocate shared and private weights for the corresponding tasks.

In CETL, the coupled networks contain multi-objectives with shared parameters. That is,

\[ L^s_r(T_1(\theta_{T_1}), x_{tgt}) \quad \text{and} \quad L^t_c(T_2(\theta_{T_2}), x_{tgt}) \] (3.7)

share parameters in \( T_1 \).

\[ L^l_c(T_2(\theta_{T_2}), x_{tgt}, y_{tgt}) \quad \text{and} \quad L^l_r(T_1(\theta_{T_1}), T_2(\theta_{T_2}), x_{tgt}) \] (3.8)

share parameters in \( T_2 \).

Thus, the issues of catastrophic forgetting tend to override model parameters learned in previous tasks, leading to impaired performance [74].

Fisher information (FI) is a way of measuring the amount of information that an observable random variable \( X \) carries for an unknown parameter \( \theta \) of a distribution that models \( X \). In [74], Fisher information \( F_i \) with respect to certain weights \( \theta_i \) of a neural network is derived from the cross-entropy loss and used to measure parameters’ importance to a given task. The Elastic Weight Consolidation loss using FI from the given task is designed as a regularization to keep the weights with large \( F_i \) unchanged in order to avoid catastrophic forgetting.

However, when all parameters are determined as important by the prior task, update of weights with respect to the new task will be trivial. The training of the new task may fail to converge. Thus, we introduce a new measure, Relative Fisher (RF) information, to determine the correlation of FIs derived from the two tasks. Denote the losses for two tasks
as $L_1$ and $L_2$, with shared parameters $\theta_i$, $i = 1, \cdots, m$, where $m$ denotes the total number of parameters, we have:

$$RF_i = I(F_{1,i}, F_{2,i} | \theta_i^*)$$

(3.9)

where $I(\cdot, \cdot)$ denotes the mutual information, $F_{1,i}$ and $F_{2,i}$ are random variables representing the FI with respect to $L_1$ and $L_2$, respectively. The higher $RF_i$ is, the more probable the two tasks may share the weights $\theta_i$. Finally, the Generalized Fisher Information (GFI) is defined as:

$$GFI_i = \begin{cases} 0 \text{ with probability } p & \text{if } RF_i < u \\ F_i \text{ with probability } 1 - p & \\ 0 \text{ with probability } 1 - p & \text{if } RF_i \geq u \\ F_i \text{ with probability } p & \end{cases}$$

(3.10)

In this way, the normalized mutual information is used to indicate whether two tasks should share the same weights. The hyperparameters $u$ and $p$ are set at 0.5 and 0.9 respectively. Thus, when $RF_i \geq 0.5$, weights will be shared, and we set the $GFI_i = 0$ with a low probability $1 - p$ to retain flexibility. Otherwise (when $RF_i < 0.5$), $F_i$ has a high probability $p$ to be dropped, and thus the new task can be better learned without regularization.

We define the Dynamic Weight Allocation (DWA) loss as the regularization term:

$$DWA = \sum_i \frac{\lambda}{2} GFI_i (\theta_i - \theta_i^*)^2$$

(3.11)

which allows dynamic allocations of shared and private parameters for different tasks. We apply it on the joint optimization of the multi-objectives in Eqs. (3.7) and (3.8).
Theoretical Analysis of the Coupled Loss

In this section, we derive an error bound for CETL learning, which provides a rigorous theoretical explanation on the rationale for the coupled loss function adopted in CETL (Eq. (3.6) in Section).

We assume the ground truth concept for the source and target as $c_{src}$ and $c_{tgt}$, respectively. Denote $T_1 \circ T_2(x)$ for $x \in P$ as $T_1 \circ T_2 \in P$, and for $x \in Q$ as $T_1 \circ T_2 \in Q$, respectively. The similar notations work for $T_1 \circ S$ as well. We denote all constant numbers in proofs as $C$ for simplicity. In addition, $E_Z$ denotes the expectation on distribution $Z$, and sup represents taking the maximum value over the collection of functions $f$.

**Lemma 4.** If $E_P|S - c_{src}| \leq C$, $C > 0$, $E_Q|c_{src} - c_{tgt}| \leq \lambda_1$, $\sup_f |E_P f - E_{T_1 \circ S \in P} f| \leq \lambda_2$, for any $f \in P, Q$, $\lambda_1, \lambda_2 > 0$, then there exists some constant $C > 0$, for any measurable function $f \in P, Q$, $f > 0$,

\[
E_Q|T_2 - c_{tgt}| \leq C + |E_{T_1 \circ S \in P} f - E_Q f| + E_Q|T_2 - S| + 2 \sup_f |E_{T_1 \circ S(x) \in Q} f - E_{x \in Q} f| + 2 \sup_f |E_{T_1 \circ T_2(x) \in Q} f - E_{x \in Q} f| \tag{3.12}
\]

**Proof**

\[
E_Q|T_2 - c_{tgt}| \leq E_P|S - c_{src}| + |E_Q|T_2 - c_{tgt}| - E_Q|S - c_{src}| + |E_Q|S - c_{src}| - E_P|S - c_{src}| \tag{3.13}
\]

Using triangle inequality,

\[
|E_Q|T_2 - c_{tgt}| - E_Q|S - c_{src}|| \leq |E_Q|T_2 - c_{tgt} - S + c_{src}| \leq E_Q|T_2 - S| + E_Q|c_{tgt} - c_{src}| \tag{3.14}
\]
It follows:

$$\begin{align*}
E_Q|T_2 - c_{tgt}| & \leq E_P|S - c_{src}| + E_Q|T_2 - S| + E_Q|c_{tgt} - c_{src}| \\
|E_Q|S - c_{src}| - E_P|S - c_{src}| & \leq C + \lambda_1 + E_Q|T_2 - S| \\
& + |E_Q|S - c_{src}| - E_P|S - c_{src}| \\
\end{align*}$$

(3.15)

Then, we focus on the analysis of the last term,

$$\begin{align*}
|E_Q|S - c_{src}| - E_P|S - c_{src}| & \\
& \leq \sup_f |E_Qf - E_Pf| \\
& \leq \lambda_2 + \sup_f |E_{T_1 \circ S \in P}f - E_{T_1 \circ S \in Q}f| \\
& + \sup_f |E_{T_1 \circ S \in Q}f - E_{T_1 \circ T_2 \in Q}f| \\
& + \sup_f |E_{T_1 \circ T_2 \in Q}f - E_Qf| \\
\end{align*}$$

(3.16)

Since

$$|E_{T_1 \circ S \in Q}f - E_{T_1 \circ T_2 \in Q}f|$$

(3.17)

$$\leq |E_{T_1 \circ S \in Q}f - E_Qf| + |E_{T_1 \circ T_2 \in Q}f - E_Qf|$$

and

$$|E_{T_1 \circ S \in P}f - E_{T_1 \circ S \in Q}f|$$

(3.18)

$$\leq |E_{T_1 \circ S \in P}f - E_Qf| + |E_{T_1 \circ S \in Q}f - E_Qf|$$

Combining (3.16), (3.17), (3.18), we get

$$\begin{align*}
|E_Q|S - c_{src}| - E_P|S - c_{src}| & \\
& \leq \lambda_2 + \sup_f |E_{T_1 \circ S \in P}f - E_Qf| \\
& + 2\sup_f |E_{T_1 \circ S \in Q}f - E_Qf| \\
& + 2\sup_f |E_{T_1 \circ T_2 \in Q}f - E_Qf| \\
\end{align*}$$

(3.19)
Substituting (3.19) into (3.15), the desired result follows.

**Lemma 5.** Assume $E_{T_1 \circ S \in P}|c_{src} - c_{tgt}| \leq C$, $E_{T_1 \circ S \in Q}|c_{src} - c_{tgt}| \leq C$, $E_{T_1 \circ S \in P}|c_{tgt}| \leq C$, $E_{T_1 \circ S \in Q}|c_{src}| < C$, for some $C > 0$, then there exists some constant $C > 0$, such that for any measurable function $f > 0$, and $f \in P, Q$,

$$
sup_f |E_{T_1 \circ S \in P}f - E_Qf| \leq C + E_{T_1 \circ S \in P}|S - c_{src}|
+ sup_f |E_{T_1 \circ S \in Q}f - E_Qf|
+E_Q|T_2 - S|
$$

(3.20)

**Proof** Using the definition of sup, for $S$, there exists some constant $C > 0$, such that:

$$
sup_f |E_{T_1 \circ S \in P}f - E_Qf| \leq |E_{T_1 \circ S \in P}S - E_QS| + C
$$

(3.21)

It follows:

$$
sup_f |E_{T_1 \circ S \in P}f - E_Qf| \leq |E_{T_1 \circ S \in P}S - E_{T_1 \circ S \in P}|c_{src} - c_{tgt}|
+E_{T_1 \circ S \in Q}|c_{src} - c_{tgt} - E_QS| + C
$$

(3.22)

Using triangle inequality, we get

$$
sup_f |E_{T_1 \circ S \in P}f - E_Qf| \leq E_{T_1 \circ S \in P}|S - c_{src}| + C
+ |E_{T_1 \circ S \in Q}|c_{src} - c_{tgt}| - E_QS| + C
\leq C + E_{T_1 \circ S \in P}|S - c_{src}|
+ |E_{T_1 \circ S \in Q}|c_{tgt} - E_QT_2| + E_Q|T_2 - S|
\leq C + E_{T_1 \circ S \in P}|S - c_{src}|
+ sup_f |E_{T_1 \circ S \in Q}f - E_Qf| + E_Q|T_2 - S|
$$

(3.23)
Theorem 6. If all conditions in Lemma 4 and 5 hold. We have the bound for CETL as:

\[ E_Q |T_2 - c_{tgt}| \leq E_P |S - c_{src}| + 2E_Q |T_2 - S| \\
+ 3 \sup_f |E_{T_1 \circ S(x) \in Q} f - E_{x \in Q} f| \\
+ 2 \sup_f |E_{T_1 \circ T_2(x) \in Q} f - E_{x \in Q} f| \]  

(3.24)

Proof

Combining the results of Lemma 4 and 5, the desired result follows.

Remark 7. The left hand side (LHS) of Eq. (3.24) is the expected classification error on the target domain. It is the ultimate objective to be minimized, but direct optimization is virtually impossible. This is our motivation and rationality to provide our theoretical analysis for the error bound. Specifically, we derived the upper bound of LHS as the right hand side (RHS) in Eq. (3.24) and proposed to minimize RHS instead. It further guides us to define the coupled loss in Eq. (3.6). RHS and Eq. (3.6) correspond as follows. Classification loss: the first term in RHS and the second term in Eq. (3.6). Cross domain loss: the second term in RHS and the third term in Eq. (3.6). Reconstruction loss: the last two terms in RHS and the first term in Eq. (3.6).

Algorithms of CETL

In this section, we will show that CETL is a unified framework that can be adapted into different tasks of transfer learning, knowledge distillation and domain adaptation. Moreover, CETL outperforms these instantiations by incorporating GFI. For better illustration, we first give the pseudo code of CETL for transfer learning in Algorithm 2, and then we show its variants.

The rationale for the three-stage training in Algorithm 2 is given below. In the coupled loss (Eq. 3.6), there are three loss terms. According to [74], catastrophic forgetting happens in multi-task training. If we optimize \( L_{coupled} \) directly using SGD, weights learned by certain tasks can be overridden by others, leading to the failure of convergence on these tasks.
Algorithm 2 Algorithm of CETL

1: **procedure** Stage 1
2:   
3: **top:**
4: **Input** $x_{tgt}$
5:   
6:   
7: **feature encoding** ← $S$
8: **feature decoding** ← $T_1$
9:   
10:  $loss_1$, $recons_1$ ← reconstruct $x_{tgt}$
11:  
12: **update** $T_1$ ← $loss_1$ w/ DWA2
13: 
14: **GFI1** ← $recons_1$
15: 
16: **procedure** Stage 2
17: 
18: **Input** $x_{tgt}$
19:   
20: **feature encoding** ← $T_2$
21: **feature decoding** ← $T_1$
22:   
23:  $loss_2$, $recons_2$ ← reconstruct $recons_1$
24:  
25: **update** $T_1 \circ T_2$ ← $loss_2$ w/ DWA1
26: 
27: **GFI2, GFI3** ← $recons_2$
28: 
29: **procedure** Stage 3
30: 
31: **Input** $x_{tgt}$
32:   
33:  classification $loss_3$ ← $T_2$
34:  
35: **update** $T_2$ ← $loss_3$ w/ DWA3
36: 
37: if $loss_1$, $loss_2$, $loss_3$ not converged
38:   
39: **goto** top
40: 
41: **end if**
Thus, we carefully designed an iterative three-stage training, in which GFI is introduced to indicate the importance of weights learned in the previous task, and DWA loss is applied as a regularization to remember the important ones during updating.

The main drawback of FI in [74] is that if most of the weights are considered important by the previous task, the model’s ability to learn a new task will be dramatically weakened. Differently, GFI uses hyperparameters to determine if the new task learning should be affected by FI. We define DWA loss using GFI to allow a dynamic allocation of shared and private weights for all the tasks.

**Transfer Learning** When we consider a traditional transfer learning problem, $T_2$ has an architecture similar to $S$, and $T_1$ has the one with reversed layers. As shown in Algorithm 2, we have three learning stages in total. For the first stage, with $S$ pre-trained on the source data, we reconstruct the target data with $T_1 \circ S$, in which the weights in $S$ are frozen while $T_1$’s are updated by the reconstruction loss to simulate $S$ in the reversed order. After the reconstruction by $T_1 \circ S$, we can obtain $GFI_1$, the generalized Fisher information for the weights in $T_1$ with respect to the reconstructed output $\text{recons}_1$.

During the second stage, we transfer the information in $T_1$ to $T_2$ while incorporating $DWA_1$. Specifically, by passing the target data through $T_1 \circ T_2$, we get the reconstruction $\text{loss}_2$. We use $\text{loss}_2$ to update $T_2$ and $\text{loss}_2$ with $DWA_1$ to update $T_1$. In this way, we keep weights unchanged if they were considered important by $GFI_1$ in $T_1 \circ S$, and update the other weights for the reconstruction in $T_1 \circ T_2$. At the end of this stage, we can obtain $GFI_2$ and $GFI_3$, which quantify the gradients of outputs with respect to weights in $T_1$ and $T_2$, respectively. Later, $DWA_2$ will be incorporated with $\text{loss}_1$ to update $T_1 \circ S$.

In the third stage, we have the classification loss on the target data given by $T_2$, and we update $T_2$ with this loss using $DWA_3$ as the regularization. Thus, part of the weights in $T_2$ will be updated for reconstruction while the rest would be for classification. The three stages are repeated iteratively until all losses are converged.
Knowledge Distillation  In knowledge distillation [65], teacher (source) and student (target) networks are generally assumed to share the same dataset. To adapt CETL for knowledge distillation, we simply need to let $T_1$ be a much more concise architecture comparing with $S$ and let $T_2$ have the reversed layers of $T_1$. Furthermore, with CETL, we can also handle the situation when source and target have different datasets. Actually, we don’t need the source data (usually a large dataset) for (expensive) training as long as we can utilize the weights from $S$ to take advantage of the soft targets.

As an improvement, FitNets was proposed later to utilize not only the soft targets but also the feature maps from the middle layer of the $S$ network [124]. CETL can be similarly modified for FitNets and we will not repeat it here due to space limit.

Domain Adaptation  The major issue we need to resolve when using CETL for domain adaptation is regarding the amount of labeled data in the target domain. We consider the following two scenarios: 1) When we have limited training labels for the target domain, we can still use them to compute the classification loss in the third stage of learning. As for the reconstruction losses, we can incorporate the reconstruction of testing samples in the target domain, similar to other domain adaptation methods [45], to improve the performance. 2) For the extreme case when no training labels are available, based on prior work in [45], we will have to use the training data from source domain to update the networks with the classification loss. In this way, the features in $T_2$ are considered invariant for both source and target domains, and thus the classification performance on target domain can be improved. Specifically, Algorithm 2 will be modified as follows: we will use source data $x_{src}$ as the input in line 17 instead if $x_{tgt}$.

Advantages of CETL  The advantages of CETL over existing transfer learning models can be summarized as follows:

- Comparing with directly fine-tuning on $S$, CETL can handle the situation when target data are not sufficient to update a deep/big source network.
• By incorporating GFI, CETL keeps the useful weights for reconstruction while updating the others. This leads to higher efficiency and better performance.

• From a practical perspective, CETL provides a very high level of flexibility on the selection of source networks. Regardless of the source architecture, source data availability, and the choice of computing platform, CETL can always leverage the pre-train source network for performance gain as long as the source output can be obtained with a forward pass. No re-training or fine-tuning is required. This unique nature makes CETL highly practical in solving various real-world problems.

We show these advantages through extensive experiments in the next section.

Table 3.8: The properties of the benchmark datasets adopted in the experiments.

<table>
<thead>
<tr>
<th></th>
<th>CIFAR-10</th>
<th>STL-10</th>
<th>CIFAR-100</th>
<th>ImageNet</th>
<th>MNIST</th>
<th>USPS</th>
<th>SVHN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classes</td>
<td>10</td>
<td>10</td>
<td>100</td>
<td>1000</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Purpose</td>
<td>image classification</td>
<td>image classification</td>
<td>image classification</td>
<td>digit recognition</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Training</td>
<td>50000</td>
<td>50000</td>
<td>50000</td>
<td>1.2 million</td>
<td>60000</td>
<td>7291</td>
<td>73257</td>
</tr>
<tr>
<td>Testing</td>
<td>10000</td>
<td>8000</td>
<td>10000</td>
<td>100000</td>
<td>10000</td>
<td>2007</td>
<td>26032</td>
</tr>
<tr>
<td>Type</td>
<td>color</td>
<td>color</td>
<td>color</td>
<td>color</td>
<td>grayscale</td>
<td>grayscale</td>
<td>grayscale</td>
</tr>
<tr>
<td>Size</td>
<td>32×32</td>
<td>96×96</td>
<td>32×32</td>
<td>256×256 (resized)</td>
<td>28×28</td>
<td>16×16</td>
<td>32×32</td>
</tr>
</tbody>
</table>

Experiments

In this section, we conduct experiments from three aspects to show the superior performance and flexibility of CETL. First, for general transfer learning tasks, we demonstrate the functionality of the components in the CETL algorithm and validate the configuration of CETL, followed by the performance analysis of the preferred architecture on various scenarios. Then, we compare CETL with other models on the performance of knowledge distillation task, and explicitly explain the rationale of using GFI. Last, we compare CETL with other state-of-the-art models on domain adaptation experiments.

In the experiments, we adopt widely used benchmark datasets to evaluate the performance of CETL, including CIFAR-10 (CI) and CIFAR-100 [77], STL-10 (ST) [26], ImageNet [29], MNIST (MN) [88], USPS (US) [68] and SVHN (SV) [111]. The descriptions of
these datasets are given in Table 3.8, and we explain how to use them in different tasks in the following sections.

**Transfer Learning**

As mentioned before, both knowledge distillation and domain adaptation can be considered as special cases of transfer learning. To avoid any confusion, in this section, we consider the scenario where both domains and tasks are different between the source and the target.

**Configurations of CETL** To start with, we consider the transfer between ImageNet and CIFAR-10 to decide the preferred configuration of CETL in Theano [141]. For ImageNet, we use the trained AlexNet model [78] provided by Caffe [71] as $S$. For CIFAR-10, as generally handled in transfer learning approaches, we randomly select only 20% of the original training data while keeping all the original testing data to form a subset of CIFAR-10 called CIFAR-10-s as the target dataset. Also, since the input image size in CIFAR-10 is much smaller than that in ImageNet, we adopt a reduced AlexNet and call it CI-CNN.

In CI-CNN, there are still five convolutional layers and three fully-connected layers, but the numbers of kernels in each layer are all reduced to about 1/2 to 1/4 of the ones in AlexNet. Also, the convolutional kernels are all set to be 3×3. As the reverse architecture of CI-CNN, there are three fully connected layers followed by alternative unpooling and deconvolution layers. For each reverse layer, the number of kernels is the same as the one in the corresponding layer in CI-CNN.

Specifically, we resize and pass the training data in CIFAR-10-s to the trained AlexNet to extract features before the last fully connected layer. Then, $T_1$ with the reverse architecture of CI-CNN reconstructs the feature from $S$. After that, the CI-CNN in $T_2$ carries out the multi-objective optimization to simultaneously reconstruct the CIFAR-10-s images with $T_1$ and classify them into ten image categories.
Table 3.9: Comparison of different configurations of CETL.

<table>
<thead>
<tr>
<th></th>
<th>Baseline</th>
<th>CETL_f</th>
<th>CETL_u</th>
<th>CETL_e</th>
<th>CETL_{fi}</th>
<th>CETL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>61.29%</td>
<td>62.41%</td>
<td>62.63%</td>
<td>63.97%</td>
<td>64.19%</td>
<td><strong>65.33%</strong></td>
</tr>
</tbody>
</table>

In Table 3.9, we compared different settings of CETL for the classification accuracy of testing samples in CIFAR-10-s. The baseline accuracy shows the result of directly training on CI-CNN. For CETL_{f}, we train \( T_1 \circ S \) until convergence and then train \( T_1 \circ T_2 \) with \( T_1 \) fixed. In this case, the reconstruction objective of \( T_2 \) can only be partially fulfilled since half of the weights in \( T_1 \circ T_2 \) are not updated. CETL_{u} takes a step further to update \( T_1 \circ T_2 \) after \( T_1 \circ S \) converged, but the problem is that \( T_1 \) could be tuned as a convolutional auto-decoder without maintaining the knowledge learned from \( S \). In CETL_{e}, we update \( T_1 \circ S \) and \( T_1 \circ T_2 \) iteratively until \( T_1 \) converges for both reconstruction objectives. However, without the control of GFI, all the weights in \( T_1 \) and \( T_2 \) are updated in the same way regardless of their importance for a given task.

For CETL_{fi}, the performance can be improved with FI, but the improvement is still limited. Finally, CETL with GFI dynamically allocates the weights in \( T_1 \) and \( T_2 \) to either shared or private, and updates them according to their importance for various tasks. Clearly, the best performance is achieved by CETL with GFI.

Notice that in this experiment, we neither update \( S \) which is more complicated than \( T_1 \), nor use the source dataset, ImageNet, which is dramatically larger than CIFAR-10-s. Instead, we take advantage of the trained AlexNet to improve the performance on CIFAR-10-s. In the following, we denote the selected configuration, CETL with GFI, as CETL, and use it in all the experiments. The architectures of \( S, T_1 \) and \( T_2 \) will be modified for different tasks.

**Different Source Networks** To show the flexibility of CETL, we perform the experiments with various combinations of source and target networks. In Table 3.10, all source architectures except for CI-CNN are pre-trained networks for the classification of ImageNet [71],
and then transferred to CIFAR-10-s and STL-10 respectively with CETL to improve their performance. CI-CNN was trained on CIFAR-100 from scratch and used as one of the source networks for STL-10. We used the same architecture CI-CNN for both CIFAR-10-s and STL-10 as the target network. As a comparison, we obtained the baseline accuracy by directly training on CI-CNN. In addition, we replaced the last fully-connected layer in VGG and fine-tuned it using the target datasets. The accuracy is reported as FT-VGG.

Table 3.10: Comparison of different source networks.

<table>
<thead>
<tr>
<th>Source Network</th>
<th>CIFAR-10-s+</th>
<th>STL-10+</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlexNet [78]</td>
<td>65.33%</td>
<td>62.98%</td>
</tr>
<tr>
<td>VGG [129]</td>
<td>65.57%</td>
<td>62.61%</td>
</tr>
<tr>
<td>GoogleNet [139]</td>
<td>65.14%</td>
<td>62.30%</td>
</tr>
<tr>
<td>ResNet-50 [62]</td>
<td>64.37%</td>
<td>62.77%</td>
</tr>
<tr>
<td>CI-CNN</td>
<td>-</td>
<td>65.49%</td>
</tr>
<tr>
<td>Baseline</td>
<td>61.29%</td>
<td>60.52%</td>
</tr>
<tr>
<td>FT-VGG</td>
<td>61.35%</td>
<td>61.17%</td>
</tr>
</tbody>
</table>

Apparently, fine-tuning is not as effective as CETL, providing little improvement. For CIFAR-10-s, highest accuracy is achieved when transferred from VGG. For STL-10, transferring from CI-CNN performs the best. The reason is that STL-10 dataset is more similar to CIFAR than to ImageNet. Also, it is clear that source data is a more important factor for the performance gain than the source architecture.

**Knowledge Distillation**

Knowledge distillation considers the problem when source and target data are the same while the student network is much smaller (thinner) than the teacher network. In this section, we compare CETL with other state-of-the-art knowledge distillation models on CIFAR-10 and CIFAR-100 datasets. To make a fair comparison, we follow some recent work [163] and choose ResNet-26 as the teacher network and CI-CNN as the student network with less than 10% parameters of AlexNet.
Specifically, for CIFAR-10, the teacher architectures are exactly the same. The student networks differ, but the initial accuracy (before distillation) are very close (87.91% in [163] and 87.55% in CETL). Same holds for CIFAR-100.

Table 3.11: Comparison on knowledge distillation.

<table>
<thead>
<tr>
<th></th>
<th>CIFAR-10</th>
<th>CIFAR-100</th>
</tr>
</thead>
<tbody>
<tr>
<td>Teacher</td>
<td>91.86%</td>
<td>65.23%</td>
</tr>
<tr>
<td>Student</td>
<td>87.55%</td>
<td>60.71%</td>
</tr>
<tr>
<td>FitNets [124]</td>
<td>88.57%</td>
<td>61.28%</td>
</tr>
<tr>
<td>Soft-targets [65]</td>
<td>88.45%</td>
<td>61.03%</td>
</tr>
<tr>
<td>FSP DNN [163]</td>
<td>88.70%</td>
<td>63.33%</td>
</tr>
<tr>
<td>CETL</td>
<td><strong>89.11%</strong></td>
<td><strong>64.83%</strong></td>
</tr>
</tbody>
</table>

As shown in Table 3.11, CETL outperforms other knowledge distillation models on both CIFAR-10 and CIFAR-100 datasets. As more training samples are available for each category in CIFAR-10, the improvement is marginal through knowledge distillation. However, classification accuracy is significantly increased in the case of CIFAR-100, close to the teacher performance. This mainly attributes to the low number of samples per category in CIFAR-100.

To further demonstrate the rationale of the DWA loss, we trace the changes of classification loss of $T_2$ and normalized DWA losses by the solid lines in Fig. 3.2 for the CIFAR-10 classification task. Clearly, the classification loss decreases with the epochs as usual. How-

Table 3.12: Comparison on domain adaptation.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SA [36]</td>
<td>85.55%</td>
<td>65.77%</td>
<td>62.33%</td>
<td>25.95%</td>
<td>54.17%</td>
<td>63.61%</td>
</tr>
<tr>
<td>ReverseGrad [40]</td>
<td>85.89%</td>
<td>51.54%</td>
<td>63.17%</td>
<td>28.52%</td>
<td>54.04%</td>
<td>62.88%</td>
</tr>
<tr>
<td>DRCN [45]</td>
<td>91.11%</td>
<td>74.01%</td>
<td>73.91%</td>
<td>35.67%</td>
<td>56.91%</td>
<td>66.12%</td>
</tr>
<tr>
<td>ADDA [143]</td>
<td>91.80%</td>
<td>88.67%</td>
<td>81.97%</td>
<td>40.05%</td>
<td>58.86%</td>
<td>66.37%</td>
</tr>
<tr>
<td>WDA [157]</td>
<td>89.40%</td>
<td>90.10%</td>
<td>76.00%</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CETL</td>
<td><strong>92.96%</strong></td>
<td><strong>90.89%</strong></td>
<td><strong>83.33%</strong></td>
<td><strong>45.27%</strong></td>
<td><strong>60.11%</strong></td>
<td><strong>66.39%</strong></td>
</tr>
</tbody>
</table>
ever, note that the DWA1 loss first increases to a peak value before decreasing and getting converged. This is because at the very beginning of training, $T_1$ is randomly initialized and for the first a few epochs, most of the weights in $T_1$ are not important for $T_1 \circ S$ and thus the loss is small. Around epoch 32, the weights in $T_1$ becomes more important for $T_1 \circ S$, leading to a larger DWA1 loss, after which the DWA1 loss decreases as the changes of weights decrease until converged. The trends of DWA2 and DWA3 losses follow a similar pattern.

**Domain Adaptation**

For the last task, we compare CETL with current state-of-the-arts on domain adaptation where target data does not have labels but has same categories as the source data. In this case, similar to other models, the multi-objective in CETL is to use target data for reconstruction and source data for classification. Following the same settings used in some recent work [45], we directly compare CETL with the reported performance in the literature in Table 3.12. ADDA and WDA results are directly obtained from [143] and [157], and a dash in the table means that the result is not reported by the corresponding model on the given dataset.

Clearly, CETL significantly improved from the prior arts and achieved the best performance on all domain adaption combinations. In particular, CETL with coupled loss and
GFI can overcome the catastrophic forgetting in multi-tasks and outperforms models (e.g., DRCN) that consider the tasks (i.e., reconstruction and classification) separately.

To validate Remark 7 from the experimental perspective, we demonstrate the normalized values of RHS and LHS in equation (3.24) w.r.t epochs for “SV-MN”. As shown by the dashed and dotted lines in Fig. 3.2, LHS is bounded by RHS and their difference is decreasing during training procedure.

Furthermore, we compared CETL with some most recent associative domain adaptation models [58, 59]. Results show that CETL is very competitive with these prior arts on domain adaptation while having the flexibility of also performing knowledge distillation and transfer learning. For example, for “SVHN-MNIST” (one of the best results as mentioned in [58]), [58] improved from 69.29% (before domain adaptation) to 97.6% (after), and [59] from 81.44% to 99.49%, while CETL goes from 62.33% to 83.33%.

**Conclusion**

In this chapter, we proposed a novel CETL framework for image classification. A novel loss function, the coupled loss, established base on the learning bound of CETL, was introduced for CETL training. In addition, GFI was integrated to improve the multi-task optimization in CETL. Experimentally, we extensively compared CETL with other state-of-the-art models for various tasks on benchmark datasets and achieved superior performance.
CHAPTER 4 Person Re-Identification with Style Transfer

Although great successes have been achieved recently in person re-identification (re-ID), there are still two major obstacles restricting its real-world performance: large variety of camera styles and a limited number of samples for each identity. In this chapter, we propose an efficient and scalable framework for cross-dataset one-shot re-ID tasks. Single-model arbitrary style transfer and pairwise comparison are seamlessly integrated in our framework through adversarial training. Moreover, we propose a novel identity-preserving loss and mathematically show that its minimization guarantees that the generated images have identical conditional distributions (conditioned on identity) as the real ones, which is critical for person re-ID. Our model achieved state-of-the-art results in challenging and realistic cross-dataset one-shot re-ID tasks.

Introduction

Person re-identification (re-ID) is a retrieval problem under the cross-camera setting. Specifically, with a query image of a person, it aims to re-identify the same person with different camera views in various background environment. One of the major obstacles encountered in person re-ID is the various camera styles [178]. Neural Style transfer, which aims at generating a content image with a given style, provides a promising solution for this problem. In [67], it was shown that the adaptive instance normalization (AdaIN) can help achieve arbitrary style transfers. The main idea is to embed the activation distribution of a content image into the one of a style image based on their summary statistics (i.e., mean and standard deviation). Actually, neural style transfer can be considered as a special case of domain adaptation [96].

Another major issue faced by person re-ID is that images for a given identity can be very limited. To this end, few-shot learning provides a viable framework, in which one tries to accomplish recognition tasks with very limited samples or descriptions. Approaches such as metric learning can learn a representation space in which paired samples are easy
to be separated. Recently, a two-branch relation network [137] was proposed for learning to compare for few-shot recognition tasks.

Motivated by these works, in this chapter we propose an efficient and scalable model for one-shot person re-ID to address the aforementioned challenges. Our framework contains three main modules: Style Transfer (ST), Feature Encoding (FE) and Relation Comparison (RC). In ST, we adopt AdaIN [67] to achieve a fast single-model style transfer between any pair of cameras. The FE module, commonly a deep convolutional network, functions to simply extract image representations in a supervised manner. Finally, the RC module formulate the person re-ID problem into pair-wise ranking, which compared with classification-based approaches is better suited for re-ID. ST, FE and RC are seamlessly integrated through adversarial training.

Conceptually, the essence of our model is illustrated in Fig. 4.1. We start with the pairwise adversarial training on source datasets (e.g., Market and Duke) to obtain an ST module for arbitrary camera style transfer. Then, we transfer labeled source samples to the unlabeled target domain (e.g., VIPeR) with unseen camera styles (cam a and b) using ST. Finally, FE+RC is trained using these style-transferred images for cross-dataset one-shot re-ID task. The major contribution of our work is summarized as follows:
We propose a novel model for one-shot person re-ID. Compared with recent work based on pairwise camera style adaptation [178], our single-model style transfer dramatically reduces the complexity during training and significantly increases the generalization to unseen camera styles, providing a more efficient and scalable solution to real-world person re-ID tasks. In addition, the RC module is introduced to deal with few-shot re-ID by learning an embedding space with better class separation when samples in each identity are sparse.

- ST, FE and RC are seamlessly integrated by identity-preserving regularization through adversarial training. Theoretically, we show that images generated by ST will have identical conditional distributions as the real ones through the optimization of the proposed loss function.

- Experimentally, we get very competitive results on benchmark datasets for supervised re-ID tasks. More importantly, our model really shines in challenging and realistic cross-dataset one-shot re-ID tasks - state-of-the-art results are obtained.

Related Work

Person Re-ID

Pioneering work in person re-ID highly relies on handcrafted discriminative features. Recently, deep learning models become a great success in person re-ID tasks. Siamese structure is popular due to its ability in measuring similarity between pairs of images. Yi et al. [162] adopt a Siamese convolutional neural network (CNN) based on the features extracted from three horizontal parts of a pedestrian image. By adding a new patch matching layer, Ahmed et al. [2] compare the activation of two images in neighboring pixels. Varior et al. [146] propose a gated function in Siamese framework to adaptively select fine local discriminative features. In [24], two Siamese networks are combined to learn from both spatial and temporal information.
Classification is an alternative method to pairwise comparison. Zheng et al. [173] propose a discriminative CNN (IDE) fine-tuned from ImageNet. Sun et al. [136] apply Singular Value Decomposition (SVD) on fully connected features to reduce feature correlation. Data augmentation is also employed in the literature to boost the re-ID performance of CNN-based methods. In [109], background and linear transformations have been applied to generate various new samples. In [174], generative adversarial network (GAN) [49] is used to generate fake samples, and a label smoothing regularization (LSR) is applied to classify fake images as none of the existing classes. Significant performance gain is recently achieved by Zhong et al. [178], in which IDE is adopted as a backbone, and CycleGAN [179] is used to generate high quality fake images. In addition, LSR is applied to give a much higher probability for the identity labels associated with real images.

**Neural Style Transfer**

Style transfer origins in non-photorealistic rendering [83] and is closely related to texture synthesis [34]. Following the seminal work of [41], in which the transfer is guided by the separation of content and style losses, numeral deep neural network (DNN)-based methods were proposed for style transfer. Examples include per-style-per-model [144, 145], multiple-style-per-model [15], and arbitrary-style-per-model fast transfer [22, 52, 67]. In particular, arbitrary-style-per-model transfer provides a unified model for efficient style adaptation.

In [41], the network is updated iteratively to minimize both the content and the style losses, which is time consuming. Feed-forward networks with the same loss function [93, 144] are designed to improve the optimization process. Improvements on image diversity and quality are also introduced recently [145]. However, these models are only capable of transferring with a fixed style. Multiple-style transfer model was discussed in [97], which works for a limited sets of styles. Arbitrary style transfer is proposed in [22] with a swap layer that replaces content features with the closest-matching style features. However, its long computation time greatly limits the usage in real-time applications. Later, Huang et
al. [67] achieved real-time arbitrary style transfer with a novel AdaIN layer that simply adjusts the summary statistics (i.e., mean and variance) of the content image to match those of the style image.

**Metric Learning**

The main intuition of metric learning is to pick a query image and a batch of gallery images from the target domain to learn projection functions that provide the similarity between each image pair [7, 130, 147]. An embedding space is learned through parameterizations of the weights of a set of mapping functions, in which paired images are easy to be separated by simple nearest neighbor or linear classifiers. For example, initial parameters [37], optimization algorithms [120] and embedding spaces [130] can be learned to facilitate the training for few-shot learning tasks.

Recently, the relation network, a CNN-based relation classifier, was proposed [137]. Instead of metric learning with a linear classifier [7, 130], the relation network is designed to learn a metric with a non-linear classifier, leading to a more flexible model with better generalization.

**Person re-ID with Identity Preserving Style Transfer and Relation Comparison**

![Figure 4.2: Optimization pipeline of our model.](image-url)
In this section, we describe in details our one-shot person re-ID framework with identity-preserving style transfer and relation comparison. We first explain the main modules in our framework and then mathematically show that images generated by ST will have identical conditional distributions (conditioned on identity) as the real ones through the optimization of our identity-preserving loss.

**General Framework**

Our main idea is to provide a versatile framework for one-shot person re-ID that combines a single-model identity-preserving style transfer and few-shot learning through adversarial training. The architecture of our framework is motivated by GAN [49]. Basically, there are three modules in our framework: ST, FE and RC. The ST module consists of its own encoder, an AdaIN layer and a trainable decoder, and given a pair of images (content and style), it will transfer the content image to the given style. The FE module, commonly a deep convolutional network, functions to simply extract image representations in a supervised manner. Finally, the RC module formulate the person re-ID problem into pair-wise ranking, which, in general, is better suited for re-ID when compared with classification-based approaches. ST, FE and RC are seamlessly integrated through adversarial training. By adding an identity-preserving regularization term in the ST module, our model insures that the transferred images keep its original identity. In the following, we describe each module in details.

**ST module.** The ST module, acting as the generator $G$, is designed for single-model camera style transfer for cross-camera person re-ID. Similar to other work in neural style transfer, in ST, we propose to tweak a content image’s style by matching its feature statistics (encoded by a VGG-19 pre-trained on ImageNet) with that of a style image. We also adopt the AdaIN layer to shift the distribution of the content image to be close to that of the style image without parameter tuning. So, ST can perform fast arbitrary style transfer for person re-ID. By decoding back to the image space, the generated image carries both the
content and style information from the inputs. More specifically, we train the decoder of ST using cross-camera images, with the content loss $L_c$ and the style loss $L_s$ as defined in neural style transfer [41]. The images generated by $G$ are then fed into FE+RC, a feature encoder followed by a relation network as the discriminator $D$, which computes the similarity score between a query and gallery images. ST and FE+RC are trained in an adversarial manner.

Different from existing style transfer models, we further introduce an identity-preserving regularization term $L_r$ to ST in addition to the style and content losses. By minimizing $L_r$, we encourage the style-transferred image to keep its original identity, which is critical for person re-ID tasks. So, the entire objective function of ST is given as follows:

$$L = L_c + \lambda_1 L_s + \lambda_2 L_r.$$  \hspace{1cm} (4.1)

where $\lambda_1$ and $\lambda_2$ are weights for style transfer and identity-preserving, respectively. $L_r$ is discussed in details in Section.

**FE module.** As a common practice, we adopt ResNet-50 as our FE module similar to IDE [173]. Differently, to keep the flexibility of our framework, we only use the layers before the last pooling layer in ResNet-50 during training, and feed the feature maps to RC module for further computation.

**RC module.** The RC module starts with convolutional layers for feature extraction followed by the concatenation of feature maps from the query and gallery images. RC decides the relationship between pairs of input images instead of considering input images individually in conventional classification models. We introduce the concept of “learning to compare” in RC. Specifically, we encode the feature maps of the image pair with one convolutional layer followed by two fully-connected layers. The feature vectors of the paired images are concatenated so that their relationship can be determined and learned during training.

We use the RC module in two stages. First, with the discriminator loss (i.e., the comparison loss) from RC, we update ST and FE+RC through adversarial training. Then,
the training dataset is augmented by the images generated by ST, and FE+RC goes through further supervised training.

**Optimization pipeline.** An example is given in Fig. 4.2 to illustrate the pipeline of our network optimization. First, we feed random pairs of images (ID1 cam1 and ID2 cam2) from different cameras (i.e., content and style images) as the input to ST and compute the content and style losses. Then, after pairing the real (ID1 cam1) and generated images (ID1 cam2) with other real images, from either a different camera or a different identity (e.g., ID1 cam3 and ID3 cam4), we feed them into FE+RC and compute the comparison loss, which is also used to compute the identity-preserving term. ST and FE+RC are updated to minimize these losses through adversarial training.

After ST is converged, FE+RC is further trained with both real and generated images for better performance in single dataset, supervised re-ID. For cross-dataset one-shot re-ID, we also transfer source samples into the styles of the target dataset and fine-tune FE+RC so that the relationship between query and gallery images can be better learned in the target domain.

**Identity Preserving Regularization**

In the ST module, we transfer camera style by shifting the statistics of a content image in the feature space to that of a style image. However, information other than the camera style in the style image, such as person identity-related information, can also be encoded in the feature vector and then conveyed into the generated images. It would be a confounding factor when training the FE+RC modules and ultimately impair the model's performance.
In this work, we design a novel regularization term $L_r$, which imposes the generated images to approach the distribution of real images with the same identity.

$$L_r = \min_G \sum_k E_{x \sim p_d} \log D_k(x|k) + E_{x \sim p_g} \epsilon \log D_k(G(x|k))$$

$$+ (1 - \epsilon)/(n - 1) \sum_{i \neq k} E_{x \sim p_g} \log D_i(G(x|k))$$

(4.2)

where $D_i$ is the similarity score from the relation network $D$ w.r.t. identity $i = 1 \cdots n$. Given the identity label $k$, the terms containing $D_k$ represents the likelihood function of a image belonging to the identity $k$. Terms with $D_i, i \neq k$ represent the likelihood functions of being other identities. $\epsilon$ and $n$ are applied to replace identity labels with smooth values for the relation network, which is used to reduce the vulnerability of $D$ to adversarial examples.

FE+RC as the discriminator $D$ and ST as the generator $G$ are adversarially trained - $D$ tries to differentiate generated images from real ones while $G$ tries to keep the identity information when generating. Specifically, the objective function of adversarial training is given as follows:

$$\max_D \min_G \sum_k E_{x \sim p_d} \log D_k(x|k) + E_{x \sim p_g} \epsilon \log D_k(G(x|k))$$

$$+ (1 - \epsilon)/(n - 1) \sum_{i \neq k} E_{x \sim p_g} \log D_i(G(x|k))$$

$$= \sum_k \min_G \max_D E_{x \sim p_{d|k}} \log D_k(x) + E_{x \sim p_{g|k}} \epsilon \log D_k(x)$$

$$+ (1 - \epsilon)/(n - 1) \sum_{i \neq k} E_{x \sim p_{g|k}} \log D_i(x),$$

(4.3)

where $p_{d|k}$ is the real image distribution given identity $k$, and $p_{g|k}$ the generated image distribution conditional on identity $k$. Different from a typical GAN model, in ST, only the term $L_r$ in Eq. (4.1) is used in adversarial training. An AdaIN layer with $L_c$ and $L_s$ is consolidated in ST to achieve arbitrary style transfer simultaneously with identity-preserving.
Theoretical analysis

In the following, we mathematically establish the optimality of identity-preserving in our model, which is critical for one-shot person re-ID. That is, we show that optimizing the loss function in Eq. (4.3) guarantees that images generated by ST will have identical conditional distributions (conditioned on identity) as the real ones. Note that in our framework, the discriminator $D$ is a relation network, which represents a set of projection functions conditioned on person identities $k = 1, \cdots, n$. This is different from a classifier used in a regular GAN. Thus, in our proof, each projection function is analyzed independently. Our approach also sheds lights on other GAN models employing pairwise comparison as D.

**Theorem 8.** For $G$, $p_{d|k}$ fixed, the optimal $D$ satisfies:

\[
D_k^* = \frac{p_{d|k} + \epsilon p_{g|k}}{p_{d|k} + p_{g|k}}, \quad D_i^* = \frac{(1-\epsilon)p_{g|k}/(n-1)}{(p_{d|k} + p_{g|k})}(4.4)
\]

**Proof** With fixed $G$ and $p_{d|k}$, we aim to find a set of projection functions $D_i, i = 1, \cdots, n$, as to maximize the objective function:

\[
E_{x \sim p_{d|k}} \log D_k(x) + E_{x \sim p_{g|k}} \epsilon \log D_k(x) + (1 - \epsilon)/(n-1) \sum_{i \neq k} E_{x \sim p_{g|k}} \log D_i(x)
\]

where $\sum_{i=1}^n D_i(x) = 1, D_i > 0$. It is equal to maximize:

\[
\int_x p_{d|k} \log D_k(x) + \epsilon p_{g|k} \log D_k(x) + (1 - \epsilon)/(n-1)p_{g|k} \sum_{i \neq k} D_i(x) dx
\]

\[
(4.6)
\]
Since for any \( y_i > 0, \) \( i = 1, \ldots, n, \) \( x_i > 0, \) \( y_i^* = \frac{x_i}{\sum_i x_i} \) is the maximizer of the objective function:

\[
\max_y \sum_{i=1}^{n} x_i \log y_i, \quad \text{subject to} \quad \sum_{i=1}^{n} y_i = 1.
\]

Then let \( x_k = p_{d|k} + \epsilon p_{g|k} \), and \( x_i = (1 - \epsilon)p_{g|i}/(n - 1) \), for \( i \neq k \), the desired result follows.

**Theorem 9.** Given optimal \( D^* \), the global minimum of objective function is achieved as:

\[
(1 + \epsilon)\log\left(\frac{1 + \epsilon}{2}\right) + (1 - \epsilon)\log\left(\frac{1 - \epsilon}{2(n - 1)}\right)
\]

with minimizer \( p_{g|i} = p_{d|i} \). Then, the generator approaches the real distribution by simulating its conditional distributions \( p_{d|i} k = 1, \ldots, n \).

**Proof** We substitute the \( D^*_k \) and \( D^*_i \), \( i \neq k \) into Eq. (4.3), we got

\[
E_{x \sim p_{d|k}}\log\left(\frac{p_{d|k} + \epsilon p_{g|k}}{p_{d|k} + p_{g|k}}\right) + \epsilon E_{x \sim p_{g|k}}\log\left(\frac{p_{d|k} + \epsilon p_{g|k}}{p_{d|k} + p_{g|k}}\right) \times (1 - \epsilon)/(n - 1) \sum_{i \neq k} E_{x \sim p_{g|i}}\left(\frac{1 - \epsilon}{(p_{d|i} + p_{g|i})}\right)
\]

\[
= \int p_{d|k} \log\left(\frac{p_{d|k} + \epsilon p_{g|k}}{p_{d|k} + p_{g|k}}\right) + \epsilon p_{g|k} \log\left(\frac{p_{d|k} + \epsilon p_{g|k}}{p_{d|k} + p_{g|k}}\right) + (1 - \epsilon)p_{g|k} \log\left(\frac{1 - \epsilon}{p_{d|k} + p_{g|k}}\right)
\]

\[
= \int (1 + \epsilon)\frac{p_{d|k} + \epsilon p_{g|k}}{1 + \epsilon}[\log((1 + \epsilon)/2) + \log\left(\frac{(p_{d|k} + \epsilon p_{g|k})/(1 + \epsilon)}{(p_{d|k} + p_{g|k})/2}\right)]
\]

\[
+ \int (1 - \epsilon)p_{g|k}[\log\left(\frac{1 - \epsilon}{2(n - 1)}\right) + \log\left(\frac{p_{g|k}}{(p_{d|k} + p_{g|k})}\right)]
\]

\[
= (1 + \epsilon)KL\left(\frac{p_{d|k} + \epsilon p_{g|k}}{1 + \epsilon}\right)\left(\frac{1}{2(n - 1)}\right) + (1 - \epsilon)KL\left(p_{g|k}\right)\left(\frac{1}{2(n - 1)}\right)
\]

\[
+ (1 + \epsilon)\log\left(\frac{1 + \epsilon}{2}\right) + (1 - \epsilon)\log\left(\frac{1 - \epsilon}{2(n - 1)}\right)
\]

\[
(4.7)
\]

The equation above obtains its minimum if and only if:

\[
\frac{p_{d|k} + \epsilon p_{g|k}}{1 + \epsilon} = \left(p_{d|k} + p_{g|k}\right)/2
\]

\[
(4.8)
\]

and \( p_{g|k} = (p_{d|k} + p_{g|k})/2 \)
That is:

\[(1 - \epsilon)p_{d|k} = (1 - \epsilon)p_{g|k} \quad \text{and} \quad p_{d|k} = p_{g|k}. \quad (4.9)\]

When \(0 \leq \epsilon < 1\), it is equivalent to \(p_{g|k} = p_{d|k}\).

From theorem 9, by optimizing (4.3), \(G\) is capable of generating images that has identical conditional distributions given identity labels to that of real ones.

The objective function \(G\) can be interpret as:

\[
L_G = \min G \mathbb{E}_{x \sim p_g} \epsilon \log D_k(G(x|k)) + (1 - \epsilon)/(n - 1) \sum_{i \neq k} \mathbb{E}_{x \sim p_g} \log D_i(G(x|k))
\]  

\[
L_G = \max G \mathbb{E}_{x \sim p_g} - \epsilon \log D_k(G(x|k)) - (1 - \epsilon)/(n - 1) \sum_{i \neq k} \mathbb{E}_{x \sim p_g} \log D_i(G(x|k))
\]  

which takes its maximum value when \(D_k(G(x|k))\) approaches 1, which could be approximated by a minimization problem as:

\[
L_G = \min G \mathbb{E}_{x \sim p_g} - \log D_k(G(x|k))
\]  

As for the objective function \(D\), it can be further interpreted as:

\[
L_D = \sum_k \min D \mathbb{E}_{x \sim p_d|k} - \log D_k(x) + \mathbb{E}_{x \sim p_g|k} - \epsilon \log D_k(x)
\]

\[-(1 - \epsilon)/(n - 1) \sum_{i \neq k} \mathbb{E}_{x \sim p_g|k} \log D_i(x)\]  

\[
-(1 - \epsilon)/(n - 1) \sum_{i \neq k} \mathbb{E}_{x \sim p_g|k} \log D_i(x)\]
When $\epsilon = 0$, $D$ is trained adversarial to $G$. Given an image of certain identity, $G$ generates images with different camera styles that will be considered as the same identity by the relation network. But the relation network is trained such that it classifies the synthesis image as other identities. The relation between $G$ and $D$ resembles that of a traditional conditional GAN ($D_k$ as real, $D_i, i \neq k$ as fake). When $\epsilon = 1/n$, it means the generated image is classified as none of the existing classes, which implicitly define a new class labeled as $n+1$. Notice during adversarial training, the value of $\epsilon$ could not be too large. Otherwise, it would tweak the generated distribution erroneously.

Experiments

Datasets

We choose three widely-adopted person re-ID benchmark datasets for experimental evaluations. Following some recent work, we adopt the standard re-ID data split settings for supervised [178] and one-shot re-ID tasks [150].

In the Market-1501 dataset [171], there are 32,668 images of 1,501 identities. The images were captured at a university by six different cameras. The images were cropped by a pedestrian detector and thus makes it challenging for the re-ID task because of the variant background environment.

In the DukeMTMC-ReID dataset [174], 2 to 426 images from eight different cameras are available for each person. It was obtained from the tracking dataset DukeMTMC by manually labeling the bounding boxes.

In the VIPeR dataset [51], there are two images from different cameras for each of the 632 identities. This dataset is extremely challenging due to the limited number of samples and the one-shot per query setting.

During training, one advantage of our model is that we can restrict the relation pairs are chosen from different cameras. This is of course not applicable to classification-based approaches. More importantly, instead of training an ST module for each different style, we
can transfer the content to arbitrary camera styles with a single ST module. This, subsequently, allows us to have more training samples by combining various datasets. As shown in a later section, it helps us achieve significant performance improvement on challenging and realistic one-shot person re-ID tasks.

**Model Ablation**

We start our experiments with different configurations of our model to determine the parameters and learning strategies that are most suitable for our experiments. We use PyTorch [116] to implement our framework (will be available in Github soon), and we evaluate our results by Rank-1 accuracy and Mean Average Precision (MAP). In the ablation study, we used the simple FE+RC modules as the baseline, and empirically set $\lambda_1$, $\lambda_2$, and $\epsilon$ in Eqs. (4.1) and (4.3) to 3, 0.5, and 0.1 respectively.

**Number of Ways** The first thing we need to decide would be the number of ways (i.e., the number of pairs of training samples with the same identity) ($N$) [137] to train the RC module. As shown in Fig. 4.3, we experimented on the Market dataset with FE+RC and report the influence on Rank-1 and MAP by the number of ways $N$. Clearly, when the number is small, like 5 or 10, inadequate comparison information is provided for learning, leading to the poor performance. As the number increases, the performance increased dramatically but not monotonically. The best results are achieved around 100 ways, after which the performance would gradually go down.

The reason behind this trend is the trade-off between generality and specialty. That is, when the number is small, an easier relationship is learned but cannot generalize well to unseen pairs. When the number is too large, the learning is affected by the much larger batch size, and it is hard to identify the relationship patterns with limited training data and computing resources.

**Ratio of Generated and Real Images** With the addition of the ST module during training, we expect the training samples to be more abundant and balanced across camera
styles. As mentioned earlier, ST and FE+RC are learned through adversarial training, the batch size of which is empirically chosen at 16. After convergence, we keep the ST module unchanged and further train FE+RC with both real and generated images in 100 ways.

With the help of the AdaIN layer in ST, we can actually transfer the real images to arbitrary style. That is, ST can generate an infinite number of images. Thus, the ratio of generated vs. real images becomes another hyper-parameter to be determined empirically. As shown in Fig. 4.4, as the ratio of generated and real images increases, the performance would always increase. However, the slope would decrease and get flat when the ratio is close to 12. Thus, we choose the ratio to be 12 in our following experiments.

**Person re-ID Results**

**Single Dataset Supervised** In this scenario, we compare our model with state-of-the-art methods on both the Market and Duke datasets. Following some recent work [102, 122, 127, 131, 152, 178], we use ST+FE+RC to carry out the supervised re-ID task.

Specifically, we first train the ST module with pairs of images from the training set of the single re-ID dataset. We update ST in the adversarial manner until convergence, and later use it to transfer the content from one camera to the style of another camera. By generating 12 times more training samples to different camera styles, we form the new
training dataset. Finally, by feeding the new training data in 100-way to FE+RC, we update until convergence and use it to predict the score between query and gallery images.

As shown in Tables 4.13 and 4.14, our model is able to achieve very competitive results compared with the state-of-the-art models. Specifically, we can achieve the best Rank-1 results in both datasets. Our MAP is close to the best result. As our model is designed to focus on the one-shot query setting, it is not really a surprise that it does not perform the best on MAP. In the next section, we will show that the cross-dataset one-shot setting is the place where our model really succeeds.
Table 4.13: Comparison on the Market-1501 dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>Rank-1 Accuracy</th>
<th>MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOW [171]</td>
<td>34.40</td>
<td>14.09</td>
</tr>
<tr>
<td>LOMO+XQDA [98]</td>
<td>43.79</td>
<td>22.22</td>
</tr>
<tr>
<td>DNS [166]</td>
<td>61.02</td>
<td>35.68</td>
</tr>
<tr>
<td>IDE [172]</td>
<td>72.54</td>
<td>46.00</td>
</tr>
<tr>
<td>Re-rank [176]</td>
<td>77.11</td>
<td>63.63</td>
</tr>
<tr>
<td>DLCE [173]</td>
<td>79.50</td>
<td>59.90</td>
</tr>
<tr>
<td>MSCAN [94]</td>
<td>80.31</td>
<td>57.53</td>
</tr>
<tr>
<td>DF [169]</td>
<td>81.00</td>
<td>63.40</td>
</tr>
<tr>
<td>SSM [5]</td>
<td>82.21</td>
<td>68.80</td>
</tr>
<tr>
<td>SVDNet [136]</td>
<td>82.30</td>
<td>62.10</td>
</tr>
<tr>
<td>GAN [174]</td>
<td>83.97</td>
<td>66.07</td>
</tr>
<tr>
<td>PDF [133]</td>
<td>84.14</td>
<td>63.41</td>
</tr>
<tr>
<td>TriNet [64]</td>
<td>84.92</td>
<td>69.14</td>
</tr>
<tr>
<td>DJL [95]</td>
<td>85.10</td>
<td>65.50</td>
</tr>
<tr>
<td>MGCAM [131]</td>
<td>83.79</td>
<td>74.33</td>
</tr>
<tr>
<td>BraidNet-CS+SRL [152]</td>
<td>83.70</td>
<td>69.48</td>
</tr>
<tr>
<td>Pose-transfer [102]</td>
<td>87.65</td>
<td>68.92</td>
</tr>
<tr>
<td>AWTL [122]</td>
<td>89.46</td>
<td>75.67</td>
</tr>
<tr>
<td>KPM+RSA+HG [127]</td>
<td>90.10</td>
<td>75.30</td>
</tr>
<tr>
<td>IDE+CamStyle [178]</td>
<td>88.12</td>
<td>68.72</td>
</tr>
<tr>
<td>IDE+CamStyle+RE [177]</td>
<td>89.49</td>
<td>71.55</td>
</tr>
<tr>
<td>ST+FE+RC</td>
<td>90.23</td>
<td>73.17</td>
</tr>
</tbody>
</table>

Cross-dataset One-shot For a more challenging and realistic situation of person re-ID [117, 134, 150, 165], we address the re-ID problem in cross-dataset one-shot scenario. The biggest advantage of our model is that it can generalize well in this most strict re-ID task. That is, the ST module can handle arbitrary camera styles, and the RC module can determine the intrinsic relation between query and gallery image pairs.

Following the experimental protocol set in [150], we randomly split the whole population into two sets, one for unsupervised fine-tuning, and the other for testing. We repeat for 10 times and report the average results. In each split, with 632 samples from two camera styles $S_1$ and $S_2$, we choose the same amount of paired identities from the source dataset. That is, we randomly select and transfer 316 source samples to style $S_1$ and transfer the
Table 4.14: Comparison on the DukeMTMC dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>Rank-1 Accuracy</th>
<th>MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOW [171]</td>
<td>25.13</td>
<td>12.17</td>
</tr>
<tr>
<td>LOMO+XQDA [98]</td>
<td>30.75</td>
<td>17.04</td>
</tr>
<tr>
<td>IDE [172]</td>
<td>65.22</td>
<td>44.99</td>
</tr>
<tr>
<td>GAN [174]</td>
<td>67.68</td>
<td>47.13</td>
</tr>
<tr>
<td>OIM [155]</td>
<td>68.10</td>
<td>47.40</td>
</tr>
<tr>
<td>APR [99]</td>
<td>70.69</td>
<td>51.88</td>
</tr>
<tr>
<td>PAN [175]</td>
<td>71.59</td>
<td>51.51</td>
</tr>
<tr>
<td>TriNet [64]</td>
<td>72.44</td>
<td>53.50</td>
</tr>
<tr>
<td>SVDNet [136]</td>
<td>76.70</td>
<td>56.80</td>
</tr>
<tr>
<td>BraidNet-CS+SRL [152]</td>
<td>76.44</td>
<td>59.49</td>
</tr>
<tr>
<td>Pose-transfer [102]</td>
<td>78.52</td>
<td>56.91</td>
</tr>
<tr>
<td>AWTL [122]</td>
<td>79.80</td>
<td>63.40</td>
</tr>
<tr>
<td>KPM+RSA+HG [127]</td>
<td>80.30</td>
<td>63.20</td>
</tr>
<tr>
<td>IDE+CamStyle [178]</td>
<td>75.27</td>
<td>53.48</td>
</tr>
<tr>
<td>IDE+CamStyle+RE [177]</td>
<td>78.32</td>
<td>57.61</td>
</tr>
<tr>
<td>ST+FE+RC</td>
<td><strong>80.51</strong></td>
<td>59.07</td>
</tr>
</tbody>
</table>

Table 4.15: Comparison on the VIPeR dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>Rank-1 Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>DLLR [76]</td>
<td>29.6</td>
</tr>
<tr>
<td>GL [75]</td>
<td>33.5</td>
</tr>
<tr>
<td>SDC [170]</td>
<td>25.8</td>
</tr>
<tr>
<td>ISR [100]</td>
<td>27.0</td>
</tr>
<tr>
<td>Dic [76]</td>
<td>29.9</td>
</tr>
<tr>
<td>SAE [90]</td>
<td>20.7</td>
</tr>
<tr>
<td>CAMEL [165]</td>
<td>30.9</td>
</tr>
<tr>
<td>kLFDA_N [35]</td>
<td>15.9</td>
</tr>
<tr>
<td>UDML [117]</td>
<td>31.5</td>
</tr>
<tr>
<td>SSDAL [134]</td>
<td>37.9</td>
</tr>
<tr>
<td>TJ-AIDL+Duke [150]</td>
<td>35.1</td>
</tr>
<tr>
<td>TJ-AIDL+Market [150]</td>
<td>38.5</td>
</tr>
<tr>
<td>ST+FE+RC+Market</td>
<td>32.9</td>
</tr>
<tr>
<td>ST+FE+RC+Duke</td>
<td>30.4</td>
</tr>
<tr>
<td>ST+FE+RC+Market+Duke</td>
<td><strong>40.7</strong></td>
</tr>
</tbody>
</table>
remaining half of source samples to style $S_2$. Fig. 4.5 shows some representative examples of transferring source identities to the target styles. Clearly, these images maintain their original identity while fitting to new camera styles as identity-preserving is enforced in the ST module.

With ST module fixed, we fine-tune FE+RC modules by the comparison loss using all the style transferred images. The learning rate of RC is 10 times of FE. Note that the samples from the source dataset get different camera styles of their own, thus, after one epoch, we permute the source samples so they are paired with different images in each style. After 100 iterations of update, FE+RC would converge again and we consider now the model is ready for the identity comparison in the target domain.

Finally, we determine the relationship of the target testing samples with the fine-tuned FE+RC modules. The results are given in Table 4.15. Using VIPeR as the target domain and either Duke or Market as the source domain, we achieved the Rank-1 accuracy of 30.4% and 32.9%, respectively. Note that these results are obtained without using the attribute information in either the source or target domain, and they are already very close to the state-of-the-arts, obtained in [150] by using additional attribute information.

As we adopted the RC module for person re-ID in our framework, as long as we can have 100 ways of comparison, the pairs can come from different cameras as well as different datasets. This is the one of the main advantages of our model. In our experiments, we trained a new model by combining Duke and Market into a single dataset and using it as the source. Thus, we now have a training dataset with 14 camera styles and 1453 individuals, which makes the trained model significantly more robust. Note that even for this setting, we still use the same architecture of ST+FE+RC and a single-model ST model. Comparing with GAN-based style transfer models where 91 GANs would then be required, our model scales much better.

The training of ST+FE+RC remains the same as mentioned previously. Now, after fine-tuning on VIPeR, we achieve the Rank-1 accuracy of 40.7%, which outperforms all
existing methods. Compared with the methods where 7 source datasets were utilized [76, 90, 165], we only use two source datasets which is much more efficient. Also, compared with the methods using attribute information [150], we avoided the problem of different attribute description between the two source datasets. Thus, our model is more efficient and scalable for the real-world person re-ID tasks.

Finally, we removed the identity-preserving term in the objective function and employed our model for the same task. That is, ST is trained with content and style losses, and FE+RC is separately trained with the comparison loss. We found out that the Rank-1 accuracy drops to 35.6%, which clearly demonstrates the importance of identity-preserving regularization through adversarial training.

**Conclusion**

In this chapter, we proposed an efficient and scalable framework for cross-dataset one-shot person re-ID. Single-model arbitrary style transfer and pairwise comparison are seamlessly integrated in our framework by a novel identity-preserving regularization through adversarial training. Compared with current state-of-the-arts, our model achieved superior performance on challenging cross-dataset one-shot re-ID tasks.
CHAPTER 5 CONCLUSION

In this dissertation, we applied ranking algorithm, transfer learning and style transfer in deep learning framework. We developed deep learning algorithms beyond supervision, including ranking-CNN, CETL, and ST+FE+RC to deal with the cases when training data is limited.

Specifically, we first proposed a novel ranking-based Convolutional Neural Network architecture, which can take advantage of both ranking algorithms and features learned with CNN models. Instead of using labels in classification or regression, it can take ordinal information into consideration. Meanwhile, features learned in CNN-based models can significantly outperform engineered features to achieve superior performance.

Then, we proposed a transfer learning framework which can also fulfill the functions of knowledge distillation and domain adaptation. In this step, we solved the problem when inadequate or even no labels are available for a target domain by taking advantage of a source domain. Furthermore, our approach can utilize the information across platform and architecture as long as a forward pass of the source network is obtainable.

Last, we proposed an efficient and scalable model for cross-dataset one-shot person re-identification tasks. In this case, we addressed the problem to determine the relationship for a pair of query and gallery images from different camera styles. We adopted the concept from style transfer together with adversarial training to boost the performance and improve the robustness.
APPENDIX

Journal Publications


Conference Publications


5. “One-shot Person Re-Identification with Identity-preserving Style Transfer”, in submission.
REFERENCES


ABSTRACT

DEEP LEARNING BEYOND TRADITIONAL SUPERVISION

by

SHIXING CHEN

May 2019

Advisor: Dr. Ming Dong
Major: Computer Science
Degree: Doctor of Philosophy

With the rapid development of innovative models and huge success on various applications, the field of deep learning has attracted enormous attention in computer vision, machine learning, and artificial intelligence. Countless researches have validated the superior performance and unprecedented extensiveness of deep learning models, especially with the advantages of high performance computing by GPUs and parallel computation. Nonetheless, drawbacks including strong dependency on supervision (sufficient labeled data) and monotonous usage of categorized labels are negatively interfering the advancement of deep learning.

In this dissertation, we plan to expose and exploit some possibilities of deep learning without using data and labels in the traditional supervision way. Specifically, we propose a pipeline to fulfill this process in a three-step manner: ranking instead of classification and regression, transfer leaning including domain adaptation, and finally data synthesis without supervised labels.

First, we propose a novel ranking-based Convolutional Neural Network architecture. It can take advantage of both ranking algorithms and features learned with CNN models. Specifically, instead of using labels in classification or regression, it can take ordinal information into consideration. Meanwhile, features learned in CNN-based models can significantly outperform engineered features to achieve superior performance.
Then, we propose a transfer learning framework which can also fulfill the functions of knowledge distillation and domain adaptation. In this step, we propose to solve the problem when inadequate or even no labels are available for a target domain by taking advantage of a source domain. Furthermore, our approach can utilize the information across platform and architecture as long as a forward pass of the source network is obtainable.

Last, we propose an efficient and scalable model for cross-dataset one-shot person re-identification tasks. In this case, we address the problem to determine the relationship for a pair of query and gallery images from different camera styles. We adopt the concept from style transfer together with adversarial training to boost the performance and improve the robustness.
AUTOBIOGRAPHICAL STATEMENT

Shixing Chen received the BS degree from School of Telecommunications Engineering, Xidian University, P.R. China in 2013. He is currently a PhD candidate in the Machine Vision and Pattern Recognition Laboratory, Department of Computer Science, Wayne State University. His research interests include computer vision, machine learning, data mining, and multimedia analysis.