Self-Supervised Learning and Domain Adaptation for Visual Analysis

Kevin Lin

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2020

Reading Committee:

Ming-Ting Sun, Chair

Linda G. Shapiro

Zicheng Liu

Program Authorized to Offer Degree:
Electrical and Computer Engineering

University of Washington

Abstract

Self-Supervised Learning and Domain Adaptation for Visual Analysis

Kevin Lin

Chair of the Supervisory Committee:
Professor Ming-Ting Sun
Department of Electrical and Computer Engineering

Supervised training with deep Convolutional Neural Networks (CNNs) have achieved great success in various visual recognition tasks. However, supervised training with deep CNNs requires large amount of well-annotated data. Data labeling, especially for large-scale image dataset, is very expensive. How to learn an effective model without the need of training data labeling has become an important problem for many applications. A promising solution is to create a learning protocol for the neural networks, so that the neural networks can learn to teach itself without manual labels. This technique is referred as the self-supervised learning, which has recently drawn an increasing attention for improving the learning performance.

In this thesis, we first present our work on learning binary descriptors for fast image retrieval without manual labeling. We observe that images with the same category should have similar visual textures, and these similar textures are usually invariant to shift, scale and rotation. Thus, we could generate similar texture patch pairs automatically for training CNNs by shifting, scaling, and rotating image patches. Based on the observation, we design a training protocol for deep CNNs, which automatically generates pair-wise pseudo labels describing the similarity between the given two images. The proposed method performs more favorably than the baselines on different tasks including patch matching, image retrieval, and object recognition.

In the second part of this thesis, we turn our focus to the task of human-centric analy-

sis applications, and present our work on learning multi-person part segmentation without human labeling. Our proposed complementary learning technique learns a neural network model for multi-person part segmentation using a synthetic dataset and a real dataset. We observe that real and synthetic humans share a common skeleton structure. During learning, the proposed model extracts human skeletons which effectively bridges the synthetic and real domains. Without using human-annotated part segmentation labels, the resultant model works well on real world images. Our method outperforms the state-of-the-art approaches on multiple public datasets.

Then, we discuss our work on accelerating multi-person pose estimation using a proposed concatenated pyramid network. We observe that each image may contain an unknown number of people that can occur at any scale or position. This makes fast multi-person pose estimation very challenging. Different from the earlier deep learning approaches that extract image features by using a series of convolutions, our proposed method extracts image features from each convolution layer in parallel, which better captures image features in different scales and improve the performance of human pose estimation. Our proposed method eliminates the need of multi-scale inference and multi-stage detection, and the proposed method is many times faster than the state-of-the-art approaches, while achieving better accuracy on the public datasets.

Next, we present our work on 3D human mesh construction from a single image. We propose a novel approach to learn the human mesh representation without any ground truth mesh. This is made possible by introducing two new terms into the loss function of a graph convolutional neural network (Graph CNN). The first term is the Laplacian prior that acts as a regularizer on the mesh construction. The second term is the part segmentation loss that forces the projected region of the constructed mesh to match the part segmentation. Experimental results on multiple public datasets show that without using 3D ground truth meshes, the proposed approach outperforms the previous state-of-the-art approaches that

require 3D ground truth meshes for training.

Finally, we summarize our completed works and discuss the future research directions.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

This thesis would not have been possible without the support and guidance from many people around me during my PhD study.

First of all, I would like to express my sincere gratitude and deep appreciation to my thesis advisor, Ming-Ting Sun, for his support and mentoring of my research. I have learned a lot from his insightful advice on improving my research, teaching, writing, and presentation. Moreover, he is very supportive, letting me pursue my research interests in different areas. His patience, guidance, and supervision enabled me to grow and become a better person.

I would also like to express my deep gratitude to my committee members, including Linda Shapiro, Tim Althoff, and Zicheng Liu, for their invaluable guidance, enthusiastic encouragement and constructive critiques of my research works. I am truly thankful to Linda Shapiro for her insightful suggestions and for sharing her tremendous experience in computer vision. I am grateful to Tim Althoff for serving as a Graduate School Representative, and also for his insightful suggestions that make my research intact. I would also like to thank Zicheng Liu at Microsoft Research. I truly appreciate Zicheng Liu for his guidance and support in various research projects, for providing me numerous research insights, and for being my co-advisor and friend.

I would like to thank all my labmates and friends for the collaborations and discussions on various research projects. Thanks to all my friends for encouraging me and inspiring me to work harder to keep up with their wonderful talents.

During my study, I have been fortunate to have the opporunity to work with

many excellent researchers in various research institutes. I would like to thank my internship mentors and my collaborators at Microsoft Research, NVIDIA Research, and eBay Research Labs for giving me new perspectives on research, for offering me opportunity to explore new research problems, and for inspiring and guiding me to be a researcher.

I would like to acknowledge the support I have received from the University of Washington and Microsoft for funding my research. I was honored to receive the Rushmer Innovator Fellowship from the ECE Department for supporting my PhD study.

Finally, I would like to express my sincere gratitude to my wife, my parents and my family for their unconditioned love and support.

Chapter 1

INTRODUCTION

1.1 Introduction

Learning an effective image representation is important for many visual recognition tasks. For example, content-based image retrieval [174] aims at searching for similar images through the analysis of image content. Along this research direction, one of the most challenging issues is associating the pixel-level information to the high-level semantics from human perception [196]. Despite several hand-crafted feature descriptors [119, 36] have been proposed to represent the images, the performance of these feature descriptors is limited until the breakthrough of deep learning. Krizhevsky *et al.* [96] demonstrated outstanding performance of deep Convolutional Neural Networks (CNNs) for extracting discriminative image representation. Since then, numerous deep learning approaches have been proposed to improve the state-of-the-art of different visual recognition tasks, including image classification [70], object detection [56], and semantic segmentation [117].

The success of deep learning is mainly driven by the supervised training from large amount of well-annotated data (e.g., ImageNet [164] and COCO [113]). However, given a new visual recognition task with vast numbers of unlabeled data, a typical solution for the state-of-the-art models is still to perform manual data labeling to facilitate the training. Data labeling, especially for associating pixel-level information to high-level semantics, is labor intensive, and performing label acquisition in large-scale is prohibitively expensive. How to learn effective models without the needs of manual labeling becomes a critical problem.

Recent studies [41, 43, 5] show that the self-supervised learning is promising to relieve the labeling difficulty. Self-supervised learning aims at creating an auxiliary learning protocol for the neural network model, so that the model can learn to teach itself a supervised task

without needing the expensive manual labeling. In this thesis, we will describe our works on training deep CNN models without manual data labeling. The goal of our studies is to design, develop, and evaluate methods that can possibly eliminate or reduce the manual labeling requirement while achieving comparable or better performance than the fully-supervised models.

In the first part of this thesis, we propose self-supervised approach, named DeepBit, for learning binary descriptors for fast image retrieval [109]. We observe that images with the same category should have similar visual textures. These textures are usually shift, scale, and rotation invariant. Thus, we could generate similar texture patch pairs automatically for training CNNs by shifting, scaling, and rotating image pairs. Based on the observation, we design an auxiliary training protocol for deep CNNs, which automatically generates pair-wise pseudo labels describing the similarity between the given two images. During training, the proposed model learns the pair-wise image similarity, and also teaches itself to learn important properties of the binary descriptors. The proposed method performs more favorably than the baselines on different tasks including patch matching, image retrieval, and object recognition.

In the second part of this thesis, we turn our focus to the task of human body analysis, which is important to many applications [65, 67, 78]. We propose a cross-domain complementary learning approach [111] for learning multi-person part segmentation without human labeling. We observe that real and synthetically synthesized humans both have a skeleton (pose) representation. We propose to learn an auxiliary task of human pose estimation, which helps the proposed model to bridge the domain gap between real and synthetic domains. The proposed model learns part segmentation from graphics simulation, and works well on real images. Our method outperforms the state-of-the-art approaches on multiple public datasets.

Then, we discuss our work on accelerating multi-person pose estimation using a proposed concatenated pyramid network. We observe that each image may contain an unknown number of people that can occur at any scale or position. This makes fast multi-person pose

estimation very challenging. Different from the earlier deep learning approaches that extract image features by using a series of convolutions, our proposed method extracts image features from each convolution layer in parallel, which better captures image features in different scales and improve the performance of human pose estimation. Our proposed method eliminates the need of multi-scale inference and multi-stage detection, and the proposed method is many times faster than the state-of-the-art approaches, while achieving better accuracy on the public datasets.

Next, we propose a novel approach for learning 3D human representation without 3D ground truth mesh labels [110]. In the Chapter 2, we show that we can train a neural network to estimate the part segmentation of the human body. By projecting the 3D mesh into 2D and minimizing the error between the projection and the part segmentation, we could learn the human mesh without a ground truth mesh. Also, for different body shapes, the internal vertex distribution could be modeled by a Gaussian Mixture Model (GMM). By adding a loss term using Laplacian prior, we show that we could learn the internal vertex distribution effectively. Our technique learns human mesh construction without any human-annotated mesh labels and achieves performance comparable to several state-of-the-art approaches which require 3D ground truth mesh labeling.

In the end of this thesis, we summarize our completed works and describe our future research directions.

1.2 Background

1.2.1 Deep Learning for Computer Vision

Computer vision research aims at generating high-level understanding of the image through the analysis of the image content; hence image representation is critical in computer vision. Deep learning [102] has become one of the most successfully approaches for computer vision tasks due to its power of learning good representations. Deep learning is a branch of machine learning that proposes to learn discriminative representations using many processing layers

in the neural network models. One of the most widely used models for computer vision tasks is the deep Convolutional Neural Network (CNN). Deep CNN performs a series of convolutions, and learns to extract image representations from low-level pixel information to high-level semantics. Thus, the resultant image representations are discriminative for image content analysis. For example, Krizhevsky *et al*. [96] firstly proposed to train deep CNNs for image classification. Their proposed model performs classification based on the learned image representations, and significantly outperforms the previous state-of-the-art approaches.

1.2.2 Unsupervised Learning

Large-scale data has driven a revolution to many computer vision tasks thanks to the high-capacity learning models such as deep CNNs. However, supervised learning has become a bottleneck due to the difficulty of data collection. Many researchers have been exploring to train the models without data labeling. One of the most representative approaches is the autoencoder, which is proposed to learn the image representations by reconstructing the input image [71]. Auoencoder is constituted by two modules: an encoder that maps the input image into the image representations, and a decoder that maps the image representations to a reconstruction of the original input image. The idea of autoencoder is general for learning representations. However, it works well under an assumption that the training and test data are drawn from the same feature space and the same distribution [137]. When the distribution changes, most of the models need to be re-trained from scratch using the newly collected training data. However, data collection and model re-training are very expensive.

1.2.3 Transfer Learning

Transfer learning has been proposed to accelerate the process of re-training. Transfer learning [137] aims at improving the learning on a new task, e.g., target task, by leveraging the prior knowledge learned from a relevant task, e.g., source task. One of the most successful examples is to firstly pre-train the deep CNN on ImageNet dataset for learning discrimina-

tive image representations, and then fine-tune the deep CNN on a new task [169]. Since the deep CNN is initialized with the pre-trained weights learned from ImageNet, fine-tuning on a target task can be very fast. Though effective, it still requires data labeling for the target task.

1.2.4 Adversarial Learning

To make the knowledge transferable accross the source and target domains, recent studies [62, 157, 187, 189] proposed to train the neural networks using adversarial training for matching the feature distributions of two domains. They proposed to train a discriminator for distinguishing the images from two domains, and a generator for extracting the domain-invariant features that can fool the discriminator. However, the adversarial training may not converge due to the fact that it is difficult to maintain a balanced training between the generator and the discriminator. Thus, the performance improvement is still limited comparing to the supervised training with manual labeling.

1.2.5 Self-Supervised Learning

Self-supervised learning has opened an interesting new avenue into unsupervised learning [125, 41, 43, 5]. Self-supervised learning methods aim at creating a learning protocol, so that the computer can learn to teach itself a supervised task. Self-supervised learning is attractive as it not only reduces the need of data labeling, but also improves the learning performance. For instance, Doersch *et al.* [41] proposed to train deep CNN to learn the arrangement of patches in an image. By learning the relative position of these patches, deep CNN is forced to also learn the spatial relationship and semantics that underlie the image. Although their model is trained to learn patch positions, they applied the learned image representations to the target task of object detection, and achieve better performance than other transfer learning approaches. Self-supervised learning has been rapidly evolving in many areas including computer vision [125, 94, 157], natural language processing [217, 39],

and robotics [88, 81], and the performance of these methods is creeping closer to the fully-supervised approaches.

1.3 Organization of the Thesis

The rest of this thesis is organized as follows:

We start by focusing on the conventional visual recognition problem with self-supervised learning technique. We address the problem of learning binary descriptors for fast image retrieval and object recognition. In Chapter 2, we present the proposed learning approach for learning binary descriptors, and detailed experiments on multiple datasets. We employ the proposed approach on different visual analysis tasks including patch matching, image retrieval, as well as fine-grain object recognition. Experimental results on several public benchmark datasets demonstrate the effectiveness of the proposed approach.

In the following chapters, we turn our focus to the problem of human activity analysis and understanding. In Chapter 3, we present the proposed cross-domain complementary learning approach for multi-person part segmentation. The proposed method removes the data labeling requirement, and successfully transfer the knowledge from synthetic data to real data. Through experiments, we show that without any human-annotated part segmentation label, our method performs comparably with several state-of-the-art approaches which require human labeling on Pascal-PersonParts and COCO-DensePose datasets. On the other hand, if parts labels are also available in real images during training, our method outperforms the supervised state-of-the-art methods by a large margin.

In Chapter 4, we present our work on accelerating multi-person pose estimation. We propose a Concatenated Pyramid Network (CCPN) to learn different resolutions of the image representations using concatenation in a deep pyramid network. The proposed model eliminates the need for multi-stage and multi-scale detection, and accelerate pose estimation to many times faster than the state-of-the-art approaches.

In Chapter 5, we present our proposed self-supervised learning algorithm for 3D human mesh construction from a single image. We represent human mesh in a form of *graph*, and use

a graph convolutional neural network (Graph CNN) [93] to learn human mesh construction. We observe that we could use 2D part segmentation (as presented in Chapter 2) for learning 3D mesh construction. By projecting the 3D mesh into 2D and minimizing the error between the projection and the part segmentation, we could learn the human mesh without a ground truth mesh. Also, for different body shapes, the internal vertex distribution could be modeled by a Gaussian Mixture Model (GMM). In the experiments, we demonstrate that our method is able to achieve a comparable or better results than the fully supervised methods.

In the end of this thesis, we summarize the works we have conducted and discuss our future research directions.

Chapter 2

UNSUPERVISED LEARNING OF BINARY DESCRIPTOR FOR IMAGE RETRIEVAL AND OBJECT RECOGNITION

2.1 Introduction

Feature descriptors have been widely used in numerous computer vision tasks [124, 119, 63, 20] such as object recognition, image classification and panorama stitching. A feature descriptor is desired to capture important and distinctive information in an image [124, 119] and also to be robust to various image transformations such as rotation and scaling [123, 119]. On the other hand, a highly efficient descriptor enables fast retrieval of images from a large corpus [174].

Over the past decade, high quality descriptors such as the rich features learned from the deep Convolutional Neural Networks (CNN) [96, 154], and the representative SIFT descriptor [119], have been widely explored. These descriptors demonstrate superior discriminability, and bridge the gap between low-level pixels and high-level semantic information [196]. However, they are high-dimensional real-valued descriptors, and usually require high computational cost to search for matched images. In order to reduce the computational complexity in matching, several lightweight binary descriptors have been proposed such as BRIEF [22], ORB [162], BRISK [103], and FREAK [6]. These binary descriptors are highly efficient for storing and matching. Given compact binary descriptors, one can rapidly measure the similarity of the images by computing the Hamming distance between binary descriptors via XOR bitwise operations. Since these early binary descriptors are computed by hand-crafted sampling patterns or simple intensity comparisons, they are usually unstable and sensitive to scales, rotations, and noises. Some previous works [185, 47, 222, 226] im-

proved the binary descriptors by encoding the similarity information into binary codes with learning algorithms. These approaches formulate the problem as a learning-to-match optimization by encouraging similar patches to have similar binary descriptors. These methods learn similarity relationship within the image pairs by taking the advantage of supervised learning and pair-wised labels (e.g. matching and non-matching labels). With the supervised learning strategy, the learned binary descriptors are highly distinctive and partially invariant to lighting conditions and camera viewpoints. However, the success of these methods is mainly attributed to similarity labels. In other words, these methods are unfavorable in the case when training data do not have label annotations.

In this chapter, we propose an unsupervised learning approach, dubbed DeepBit, for learning compact binary descriptors using a deep neural network. We observe that images with the same category should have similar visual textures. These textures are usually shift, scale, and rotation invariant. Thus, we could generate similar texture patch pairs automatically for training CNNs by shifting, scaling, and rotating image pairs. Based on the observation, we design an auxiliary training protocol for deep CNNs, which automatically generates pair-wise pseudo labels describing the similarity between the given two images. During training, the proposed model learns the pair-wise image similarity, and also teaches itself to learn important properties of the binary descriptors. The proposed method performs more favorably than the baselines on different tasks including patch matching, image retrieval, and object recognition.

2.2 Related Works

Binary Descriptors: Earlier works related to binary descriptors can be traced back to BRIEF [22], ORB [162], BRISK [103], and FREAK [6]. These binary descriptors are built upon hand-crafted sampling patterns, and a set of pairwise intensity comparisons. While these descriptors are efficient, their performance is limited because pairwise intensity comparison is sensitive to the scale and geometric transformation. To address these limitations, several supervised approaches [186, 184, 176, 47, 222, 15, 211] have been proposed to learn

binary descriptors. D-BRIEF [186] encodes the desired similarity relationships and learns a project matrix to compute discriminative binary features. On the other hand, Local Difference Binary (LDB) [215] applies Adaboost to select optimal sampling pairs. Linear Discriminat Analysis (LDA) is also applied to learn binary descriptors [176]. Recently proposed BinBoost [184] learns a set of projection matrix using the boosting algorithm, and achieves state-of-the-art performance on patches matching. In addition, Supervised Discrete Hashing (SDH) [170] learns binary codes with the minimal classification loss of a linear classifier. Supervised Incremental Hashing (SIH) [136] jointly optimizes the classification error and the binary codes learning in a supervised manner. While these approaches have achieved impressive performance, their success is mainly attributed to pair-wise learning with similarity labels, and is unfavorable for the case when transferring the binary descriptor to a new task.

Unsupervised hashing algorithms learn compact binary descriptors whose distance is correlated to the similarity relationship of the original input data [8, 61, 166, 202]. Locality Sensitive Hashing (LSH) [8] applies random projections to map original data into a low-dimensional feature space, and then performs a binarization. Semantic hashing (SH) [166] builds a multi-layers Restricted Boltzmann Machines (RBM) to learn compact binary codes for text and documents. Spectral hashing (SpeH) [202] generates efficient binary codes by spectral graph partitioning. Iterative qauntization (ITQ) [61] uses iterative optimization strategy to find projections with minimal binarization loss. Even if these approaches have been proved effective, the binary codes are still not as accurate as the real-valued equivalents.

Deep Learning: Deep learning has drawn increasing attention in visual analysis since Krizhevsky*et al.* [96] demonstrated the outstanding performance of the deep CNN on the $1,000$ class image classification. Their success is attributed to training a deep CNN to learn rich mid-level image representations on millions of images. Recent studies [219, 68, 221, 98] show that training a Siamese deep network with contrastive loss based on pair-wised or triplet image pairs achieves superior performance to SIFT for the task of local patch matching or retrieval. Dosovitskiy *et al.* [43] showed that it is possible to learn discriminative real-valued descriptors with the surrogate patch labels. Paulin *et al.* [139] and Balntas *et al.* [14] showed

that the success of the deep descriptors learning is mainly attributed to the convolution operation that extract local information of the patches. However, the descriptors learned by the above mentioned approaches are generally high-dimensional real valued descriptors, which require relatively high computational cost image matching. In contrast, the proposed binary descriptor not only reduces the computational complexity, but also enable efficient image and object matching.

Recent studies [197] have shown that deep learning is effective for binary codes learning. For example, CNNH [209], DNNH [100] and DSH [116] employed deep CNN to learn a set of hash functions. However, their methods require pair-wised similarity labels or triplets training data. VDSH [225] proposed to learn binary codes via layer-wise local updates, which potentially avoids the gradient vanishing problem. Instead of using the sign function, BDNN [40] directly generated binary codes with the designed constraints on the output layer. Huang *et al.* [72] proposed to jointly learn feature representations with unsupervised discriminative clustering and weakly-supervised hashing. Deep Hashing (DH) [114] builded a three-layer hierarchical neural network to learn the discriminative projection matrix, but their method does not take the advantage of deep transfer learning, thus makes the binary codes less effective. In contrast, the proposed DeepBit not only transfers the mid-level image representations pre-trained from ImageNet to the target domain, but also learns compact yet discriminative binary descriptor. We will show that our method achieves better or comparable performance than state-of-the-art descriptors on several public benchmark datasets.

2.3 Proposed Approach

Fig. 2.1 shows the learning framework of our proposed method. We introduce an unsupervised deep learning approach, called DeepBit, to learn compact yet discriminative binary descriptors. Unlike previous works [185, 47, 186, 184] that optimize the projection matrix with hand-crafted features and label information from datasets, DeepBit learns a set of non-linear projection functions to compute compact binary descriptors in an unsupervised manner. We enforce three important criteria on the binary descriptors, and optimize the

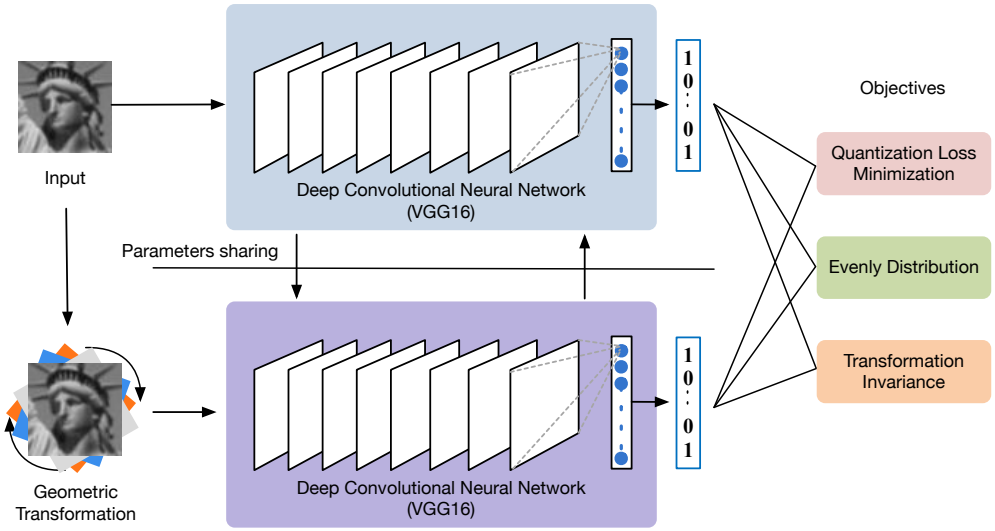


Figure 2.1: An overview of the proposed unsupervised deep learning framework for computing binary descriptors. We create a new fully-connected layer on the top of the existing deep neural network, and employ the proposed objective functions to learn compact yet discriminative binary descriptor. The proposed approach takes two inputs from the original image and the geometric transformed one into two towers of our network, respectively. Both of the networks share and update the same parameters. Then, we learn binary descriptors through the optimization of the proposed objective functions. Note that the binary descriptor is computed by applying only one tower of our network in the test phase.

parameters of the proposed network with the stochastic gradient descent technique. In this section, we first give an overview of our approach, and then describe the proposed learning objectives in the following sections.

2.3.1 Overall Learning Objectives

The proposed DeepBit aims to learn K projection functions that map each image x_n into the binary vector $b_n = [b_{n1}, b_{n2}, \ldots, b_{nK}] \in \{0,1\}^{1\times K}$. $\mathcal{F}_k(\cdot)$ and W_k represent the k-th projection function and its associated parameter set. The projection function $\mathcal{F}_k(x_n; W_k)$ is a composition of a number of non-linear projections which can be written as:

$$\mathcal{F}_k(x_n; W_k) = f_{kL}(\cdots f_{k2}(f_{k1}(x_n; w_{k1}); w_{k2})\cdots; w_{kL}), \tag{2.1}$$

where L is the total number of layers in the deep neural network, and $W_k = [w_{k1}, w_{k2}, \ldots, w_{kL}]$. Note that w_{kl} represents the projection parameter learnt for the l-th layer of the deep neural network. Sigmoid activation function is applied on the last layer, thus $\mathcal{F}_k(\cdot) \in \mathbb{R}$ and $0 \leq \mathcal{F}_k(\cdot) \leq 1$. The k-th bit of the binary descriptor b_{nk} is computed by applying the projection function $\mathcal{F}_k(\cdot)$ to the input image x_n and binarizing the results:

$$b_{nk} = 0.5 \times (\text{sign}(\mathcal{F}_k(x_n; W_k) - 0.5) + 1), \tag{2.2}$$

where $\text{sign}(v) = 1$ if $v > 0$ and -1 otherwise.

Let $W = [W_1, W_2, \ldots, W_K]$, which consists of K parameters for computing the K-bit binary descriptor. To learn the K-bit binary descriptor, we define three important criteria to optimize W. First, the learned binary descriptor should preserve the information of the feature extracted from the last layer of the deep CNN. The quantization loss should be as small as possible after projection. Second, we encourage the binary descriptor to be evenly distributed, so that each bit of the binary descriptor will convey information as much as possible. The third is to make the descriptor invariant to geometry transformations.

To achieve these objectives, we formulate the following optimization problem to learn W:

$$\begin{aligned} \min_W E(W) &= \alpha E_1(W) + \beta E_2(W) + \gamma E_3(W) \\ &= \alpha \sum_{k=1}^{K} \sum_{n=1}^{N} ||b_{nk} - \mathcal{F}_k(x_n; W_k)||^2 \\ &+ \beta \sum_{k=1}^{K} ||\mu_k - 0.5||^2 \\ &+ \gamma \sum_{k=1}^{K} \sum_{n=1}^{N} \left\{(y)d + (1-y)(K-d)\right\}, \end{aligned} \tag{2.3}$$

where N is the number of training data in a mini-batch. K is the bit length of the binary descriptor. μ_k is the mean of the k-th bit binary descriptor of the training data. d denotes the distance between the compared binary descriptors b_p and b_q. Note that b_p and b_q are computed from input image x_p and x_q. y denotes the relationship between the compared binary descriptors. $y = 1$ if $x_q = T(x_p)$, and $y = 0$ if $x_q \neq T(x_p)$. Note that $T(\cdot)$ computes

the geometric transformation. Moreover, α, β, and γ are the hyper-parameters to balance different objectives.

To give a better understanding of the proposed objectives, we describe the physical meaning of (5.13) as below. First, E_1 minimizes the quantization loss between the binary descriptor and the input feature vector. Then, E_2 encourages the binary descriptor to be evenly distributed to increase the information capacity of the binary descriptor. Finally, E_3 tolerates the geometry transformations by minimizing the Hamming distance between the descriptors that describe the reference image and the transformed ones. We elaborate the details of each proposed objective as follows.

2.3.2 Quantization Loss Minimization

The proposed DeepBit seeks to learn the projections that map the input image into a binary string while preserving the discriminative information of the original input. We first rewrite (2.2) as follows:

$$b_{nk} - 0.5 = 0.5 \times \text{sign}(\mathcal{F}_k(x_n; W_k) - 0.5). \tag{2.4}$$

The soul idea to keep the binary descriptors informative is to minimize the quantization loss between the binary descriptor and the rich image representation. We minimize the quantization loss of the k-bit binary descriptor as below:

$$\begin{aligned} \min_{W_k} E_1(W_k) &= \sum_{n=1}^{N} ||0.5 \times \text{sign}(\mathcal{F}_k(x_n; W_k) - 0.5) \\ &\quad - (\mathcal{F}_k(x_n; W_k) - 0.5)||^2 \\ &= \sum_{n=1}^{N} ||b_{nk} - 0.5 - \mathcal{F}_k(x_n; W_k) + 0.5||^2 \\ &= \sum_{n=1}^{N} ||b_{nk} - \mathcal{F}_k(x_n; W_k)||^2. \end{aligned} \tag{2.5}$$

The smaller the quantization loss is, the better the binary descriptor will preserve the information of the original data. Different from the previous work [61] that addresses this

problem by iteratively updating W_k and b_{nk} with two alternating steps, we formulate this optimization problem as the neural networks training objective. The goal of the proposed deep neural network becomes learning W_k that minimizes the quantization loss between b_{nk} and $\mathcal{F}_k(x_n; W_k)$. To this end, we optimize $W = [W_1, W_2, \ldots, W_K]$ for the K-bit binary descriptor by minimizing the loss of the following objective function:

$$\min_{W} E_1(W) = \sum_{k=1}^{K} \sum_{n=1}^{N} ||b_{nk} - \mathcal{F}_k(x_n; W_k)||^2. \tag{2.6}$$

2.3.3 Learning Efficient Binary Descriptors

To increase the information capacity of the binary descriptors, we maximize the usage of each bin in the binary string. Considering the variance for each bin, the higher the entropy is, the more information the binary codes express. Accordingly, we enhance the binary descriptor by making each bit has 50% probability of being one or zero. In other words, there is no preference for each bit to be one or zero, and the resulting binary string will convey the information as much as possible. To achieve this goal, we minimize the loss computed by the forward pass of the network:

$$\min_{W} E_2(W) = \sum_{k=1}^{K} ||\mu_k - 0.5||^2, \tag{2.7}$$

where K is the bit length of the binary descriptor. For each bit, we compute mean μ_k using:

$$\mu_k = \frac{1}{N} \sum_{n=1}^{N} b_{nk}, \tag{2.8}$$

where N is the number of images in a mini-batch.

2.3.4 Learning Transformation Invariant Descriptors

Previous studies [124, 119, 123, 20] show that an effective local descriptor should not only be distinctive, but also invariant to geometry transformations. The local descriptors are desired to have three important properties including (1) rotation invariance, (2) translation

invariance, and (3) scaling invariance. To learn binary descriptors that are invariant to these transformations, we address the problem by minimizing the difference between the binary descriptors which are computed from the reference image and the transformed one. The transformed image is computed by applying a set of rotation, translation and scaling functions. In order to make the binary descriptor more distinctive, we further enhance the binary descriptors by increasing the distance between the descriptors computed from arbitrary images. Inspired by Dosovitskiy *et al.* [43] that incorporate surrogate labels (or pseudo labels) for unsupervised learning, we formulate the optimization problem as the loss function below:

$$\min_{W} E_3(W) = \sum_{n=1}^{N} \{(y)d + (1-y)(K-d)\}, \tag{2.9}$$

where $d = ||b_p - b_q||$. b_p and b_q are the binary descriptors computed from training data x_p and x_q, respectively. y reveals the transformation relationship between x_p and x_q. $y = 1$ if $x_q = T(x_p)$, and $y = 0$ if $x_q \neq T(x_p)$. $T(\cdot)$ computes the geometric transformation with random combination of rotation, scaling, and translation functions. During learning, we encode the important characteristics of local descriptors into the top layer of the deep neural network, including rotation, translation, and scaling invariance. Then, the binary descriptors are learned by optimizing the network parameters through back-propagation. Since y in (2.9) indicates the geometric transformation *on* or *off*, the objective function does not take into account the label annotations provided by the datasets.

2.3.5 Implementation Details

We implement our proposed approach using the open source Caffe [84]. The proposed approach includes two stages. First, we initialize our network with the parameters from the VGG16 [173], which is trained on the ImageNet dataset. Second, we replace the classification layer of the VGG16 with a new fully-connected layer following by the sigmoid activation function, and enforce the neurons in the new fully-connected layer to learn binary descriptor.

The proposed method learns binary descriptors based on the Siamese-like network architecture, which consists of two towers of our network. During learning, we optimize W using the proposed objective function (see (5.13)). Note that the two towers of the network share and update the same parameter W. To this end, we use the stochastic gradient descent (SGD) method and back-propagation to train our network. The bit-length of our binary descriptor can be customized from dataset to dataset by setting the number of neurons in the last fully-connected layer. We generate a training list with $2M$ pairs of x_p and x_q, where M is the total number of images in the dataset. We shuffle the training list, and feed to the network batch by batch. Other settings are listed below. Mini-batch size is 32. We train our model with a learning rate 0.0001, a momentum 0.9 and a weight decay 0.0005. μ_k in E_2 is replaced with the mean of the minibatch during training. Images are normalized to 256×256 and then randomly cropped to 224×224 for VGG16 as the network input. For the parameters of the geometric transformation function, the rotation range is from -10 to 10, shifting range is from -32 to 32 pixels, and the scaling factor is from 0.8 to 1.2. Note that we empirically set these parameters for simulating human perspective with small variations of viewpoints, and the performance of the binary descriptors could be further boosted by using optimal parameter settings via hyperparameter selections.

2.4 Experimental Results

We conduct experiments on several public datasets as well as a large-scale dataset with more than one million natural images. We provide extensive evaluations of the proposed binary descriptor, and demonstrate its performance on various visual recognition tasks. We start with introducing the datasets and then present the comparative evaluations with the state-of-the-art methods. Information of the datasets can be found in Table 2.1.

2.4.1 Datasets

Brown [19] is a standard dataset for evaluating local descriptors. It consists of three subsets, which are collected from the Photo Tourism reconstructions from three landmarks (Liberty,

Table 2.1: Statistics of datasets used in the experiments.

Dataset	Image type	Label	Training	Test
Brown	64×64 patches	Match or not	200,000	100,000
RomePatches	51×51 patches	Match or not	10,000	10,000
CIFAR-10	32×32 patches	10 class	50,000	10,000
Flower-17	Natural images	17 class	680	340
Flower-102	Natural images	102 class	1,020	6,149
ILSVRC2012	Natural images	1,000 class	$\sim$1.2 M	50,000
Oxford5k	Landscape images	N/A	N/A	55
Paris6k	Landscape images	N/A	N/A	55
Holidays	Natural images	N/A	N/A	500
UKB	Object images	2,550 class	N/A	10,200

Notre Dame, and Yosemite). Each of them contains different views of a given landmark. For each landmark, there are more than $400,000$ gray-scale patches, resulting in a total of $1,200,000$ patches. Each subset is split into training and test sets, with $200,000$ training pairs ($100,000$ matched and $100,000$ non-matched pairs) and $100,000$ test pairs ($50,000$ matched, and $50,000$ non-matched pairs), respectively.

CIFAR-10 [97] is one of the most widely used datasets for evaluating classification and retrieval. It contains 10 object categories and each class consists of $6,000$ images, resulting in a total of $60,000$ images. The dataset is split into training and test sets, with $50,000$ and $10,000$ images, respectively. We employ the training set for network training, and use the test set as the queries for retrieval evaluation.

RomePatches [139] is another dataset for descriptor evaluation. Different from Brown dataset, RomePatches consists of a series of local patches with color information. The dataset is collected from the 3D reconstruction of several landmarks in Rome. The patches

are obtained by projecting the 3D feature points back to the 2D original images. For both training and test sets, there are $1,000$ 3D feature points with 10 different views, resulting in a total of $10,000$ training patches and $10,000$ test patches.

ILSVRC2012 [164] is a standard benchmark for the ImageNet Large Scale Visual Recognition Challenge. It consists of $1,000$ object classes, and approximately 1.2 million training images, $50,000$ validation images and $100,000$ test images. Similar to the setting in CIFAR-10, we employ the training set to learn network parameters, and use the validation set as the queries in the evaluation.

Flower-17 [132] contains 17 categories and each class consists of 80 images, resulting in a total of $1,360$ images. The dataset is split into the training (40 images per class), validation (20 images per class), and test (20 images per class) sets.

Flower-102 [133] contains 102 categories and each class consists of $40 \sim 102$ images, resulting in a total of $8,189$ images. The dataset is split into the training (10 images per class), validation (10 images per class), and test (the rest $6,149$ images) sets.

Oxford [143] has $5,062$ images of 11 Oxford landmarks. Images are with different variations in viewpoints and scales, which pose practical challenges for image retrieval. In this dataset, 55 queries are used for the evaluation.

Paris [144] has $6,412$ images of Paris landmarks. Similar to Oxford, there are 55 queries for the retrieval evaluation.

INRIA Holidays [82] has $1,491$ images corresponding to 500 image groups. The images are with various rotations and scales, making the retrieval more challenging. The performance is evaluated on 500 queries.

UKB [134] contains $2,550$ object categories and each object has 4 images in different viewpoints, resulting in a total of $10,200$ object images.

2.4.2 Results on Brown Dataset

Comparison with unsupervised approaches: To evaluate the performance of local descriptors, we first compare our method with state-of-the-art unsupervised binary descriptors (DBD-MQ [45], BRIEF [22], BRISK [103], Boosted SSC [168]). Following the standard evaluation protocol [185], we conduct cross-validation on the three subsets, and compute the false positive rate at 95% recall rate (95%ERR). We train our model on one subset, and apply to the other two subsets. Thus, there are in a total of 6-round validations. The validations can be seen as the cross-scene evaluation since the three subsets are collected from different environments including natural scene, statue, and architecture, respectively. Fig. 2.2 shows the ROC curves for our proposed model and the compared methods, and Table 2.2 summarizes the 95% ERR for the Brown dataset. The overall performance of our model achieves 38.15% error rate when the recall rate is at 95%, which achieves lower error rates than the compared unsupervised binary descriptors over different training and testing configurations of the Brown dataset. In addition, we study the influence of the loss function designed for the transformation invariance objective. As shown in Table 2.2, our model learned with the transformation invariance objective achieves 38.15% error rate, which is better than 40.67% of our prior work [108] which does not consider learning transformation invariant descriptor. Since the proposed new objective function is more general for descriptor learning, our new model performs more favorably against the previous one.

Comparison with supervised approaches: We also compare the proposed unsupervised binary descriptors with the state-of-the-art supervised binary descriptors (Binary L2-Net [178], BinBoost [185], RFD [47] and others). In Table 2.2, supervised approaches employ similarity supervision (matched and non-matched labels) to optimize the model training, so that their binary descriptors are more effective for estimating the visual similarity given the pair-wised input patches. Since our method is unsupervised, where the learning process does not take advantage of the training labels provided by the dataset, our method performs less favorably than the supervised approaches.

Table 2.2: Comparison of the proposed binary descriptor to the state-of-the-art binary descriptors, in terms of 95% error rates (ERR) across all the splits of training and testing configurations. For reference, we also provide the results of real-valued descriptor SIFT [119]. The proposed method achieves better performance than the unsupervised binary descriptors in most cases, while remaining competitive to supervised approaches.

Method	Dimension	Type	Supervised	Yosemite Notre Dame	Yosemite Liberty	Notre Dame Yosemite	Notre Dame Liberty	Liberty Notre Dame	Liberty Yosemite	Average 95% ERR
SIFT [119]	128	Float	N	28.09	36.27	29.15	36.27	28.09	29.15	31.17
MatchNet [68]	4096	Float	Y	5.67	10.77	8.39	6.9	3.87	10.88	7.74
DeepCompare [219]	256	Float	Y	2.11	7.20	4.10	4.85	1.90	5.00	4.19
BRISK [103]	512	Binary	N	74.88	79.36	73.21	79.36	74.88	73.21	75.81
BRIEF [22]	256	Binary	N	54.57	59.15	54.96	59.15	54.57	54.96	56.23
ORB [162]	256	Binary	N	54.57	59.15	54.96	59.15	54.57	54.96	56.23
DeepBit [108]	256	Binary	N	29.60	34.41	63.68	32.06	26.66	57.61	40.67
DBD-MQ [45]	256	Binary	N	27.20	33.11	57.24	31.10	25.78	57.15	38.59
Boosted SSC [168]	128	Binary	Y	72.20	71.59	76.00	70.35	72.95	77.99	73.51
LDAHash [176]	128	Binary	Y	51.58	49.66	52.95	49.66	51.58	52.95	51.40
D-BRIEF [186]	32	Binary	Y	43.96	53.39	46.22	51.30	43.10	47.29	47.54
BinBoost [185]	64	Binary	Y	14.54	21.67	18.96	20.49	16.90	22.88	19.24
RFD [47]	293-598	Binary	Y	11.68	19.40	14.50	19.35	13.23	16.99	15.86
Binary L2-Net [178]	256	Binary	Y	2.51	6.65	4.04	4.01	1.90	5.61	4.12
Ours	256	Binary	N	28.49	34.64	54.63	33.83	20.66	56.69	38.15

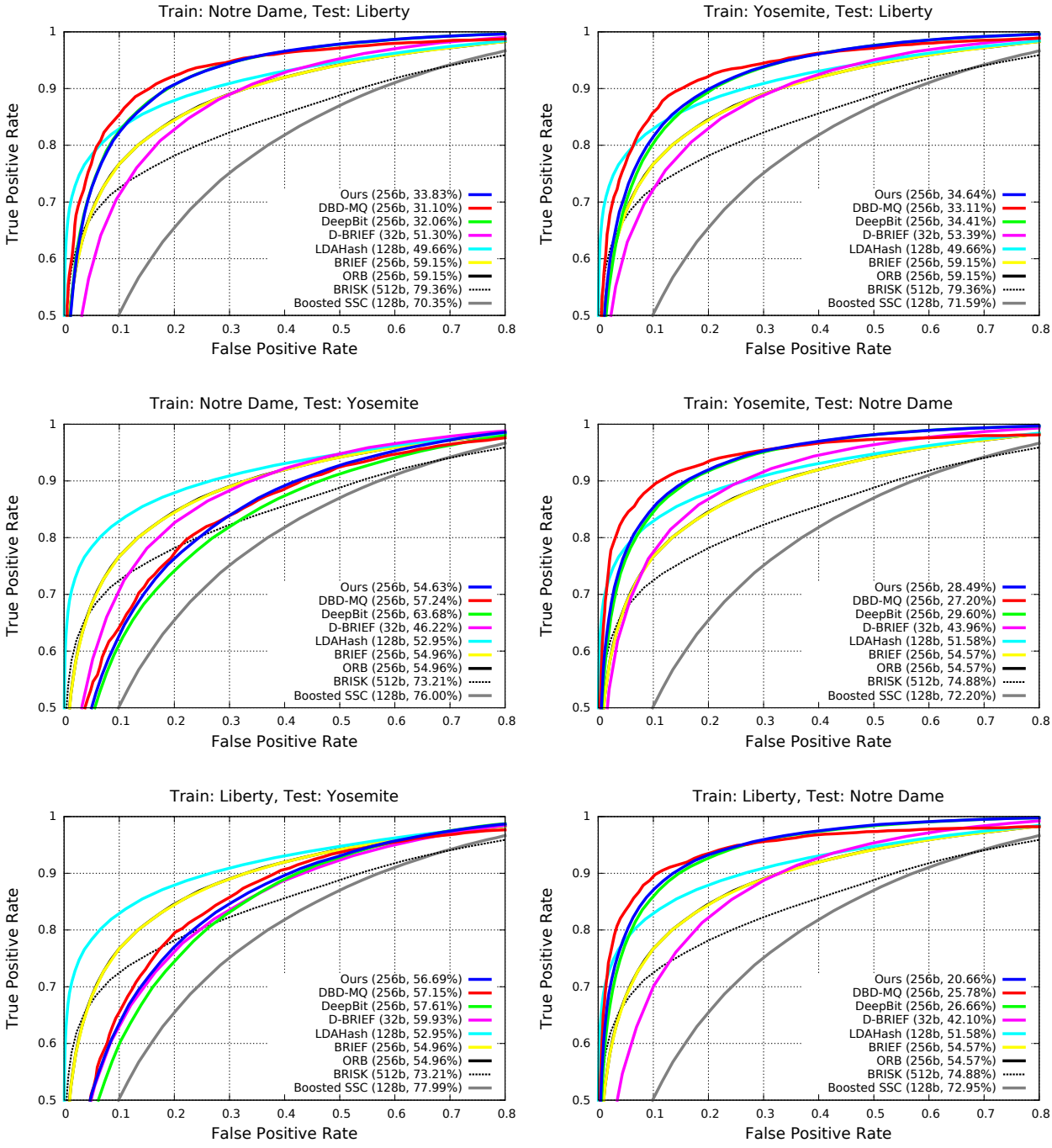


Figure 2.2: ROC curves of the proposed DeepBit descriptors and the compared binary descriptors, across all the splits of training and testing configurations on the Brown datasets. In parentheses: the bit length of the binary descriptor *(b)*, and the 95% error rates.

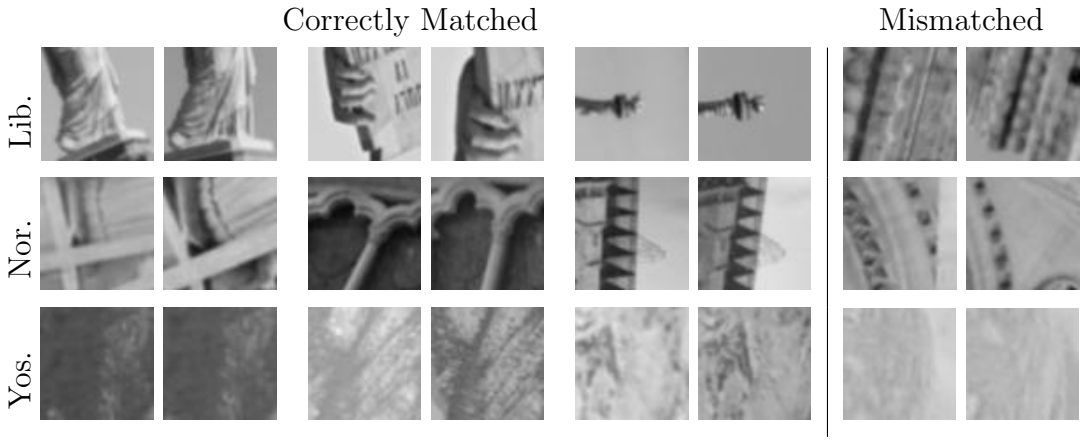
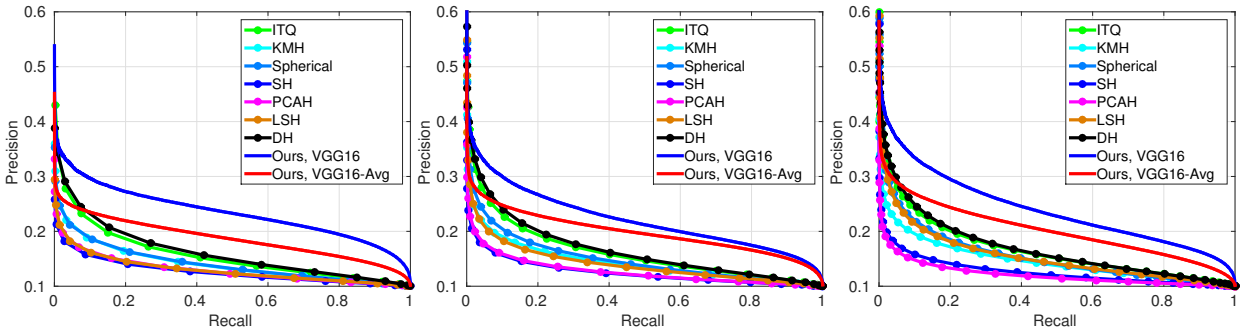


Figure 2.3: Correctly matched patches and mismatched ones from the Brown dataset. Top row shows the patches from Liberty classified as matched pairs; the first three are correctly classified, but the fourth is mismatched, which describes different architectures. Middle row shows the image pairs from Notredame classified as the matched pairs; the fourth is mismatched although both of them share similar pattern. Bottom row shows the patches from Yosemite classified as matched pairs; the last one is mismatched, which are visually similar but belong to different locations.

Visualization of patch matching: We further visualize the image matching results on the Brown dataset, and Fig. 2.3 shows the visualization results. As can be seen, our model successfully matches pairs of patches when they are visually similar, as shown in the first three columns of Fig. 2.3. The proposed method could also mismatch some patches as shown in the fourth column of Fig. 2.3. It is worth noting that the mismatched patches are still visually similar although they are from different scenes or locations. More specifically, the patches from Liberty and Notre Dame describe the local structure of the statue and architecture, where the visual similarity between different patches is usually weak. The proposed method achieves more favorable performance in these two datasets. However, the patches from Yosemite depict the surface of a mountain. Different local patches (such as snow and forest) could generate visually similar patterns, making them difficult to be distinguished. This could be the reason why our method, which tends to match patterns that are visually similar, performs less favorable than some methods for the Yosemite dataset.



(a) 16 bits (b) 32 bits (c) 64 bits

Figure 2.4: Precision/Recall curves of different unsupervised hashing methods on the CIFAR-10 dataset with respect to 16, 32 and 64 bits, respectively.

2.4.3 Results on CIFAR-10 Dataset

To evaluate the discriminability of the proposed binary descriptor, we further test our method on the task of image retrieval. We compare our method with several unsupervised hashing methods, including conventional hashing [8, 61, 202], and deep learning based approaches [114, 72, 40, 45] on the CIFAR-10 dataset. Following [114], we train the conventional approaches (such as ITQ) on the GIST descriptors. In addition, we also train PCA-ITQ and LSH on the ℓ_2-normalized feature extracted from the $fc7$ layer of the VGG16 network. These can be seen as the deep learning baselines.

Performance comparison: Following the settings in [114], Table 2.3 shows the retrieval results on CIFAR-10 in terms of the mean Average Precision (mAP) of all returned images of different methods. Fig. 2.4 shows the Precision/Recall curves of different unsupervised hashing methods with 16, 32, 64 bits, respectively. Our method achieves comparable or better mAPs than these compared methods with respect to different bit lengths. This suggests that the proposed objective function encourages each bit of our binary descriptor to be activated, and makes the binary descriptors more efficient. Specifically, the VGG16 + PCA-ITQ approach computes PCA to obtain the feature embedding, and performs ITQ for better

Table 2.3: Performance comparison (mAP, %) of different unsupervised hashing algorithms on the CIFAR-10 dataset. This table shows the mean Average Precision (mAP) of all returned images with respect to different number of hash bits.

Method	16 bit	32 bit	64 bit
GIST + SpeH [202]	12.55	12.42	12.56
GIST + SH [166]	12.95	14.09	13.89
GIST + PCAH [198]	12.91	12.60	12.10
GIST + LSH [8]	12.55	13.76	15.07
GIST + PCA-ITQ [61]	15.67	16.20	16.64
VGG16 + LSH	10.67	10.57	10.03
VGG16 + PCA-ITQ	20.97	21.74	22.32
DH [114]	16.17	16.62	16.96
Huang *et al.* [72]	16.82	17.01	17.21
UH-BDNN [40]	17.83	18.52	-
Ours	**21.70**	**20.64**	**23.07**

feature binarization. As discussed in the literature [61, 25], PCA is very effective to preserve semantic consistency for the small code sizes. Since our method learns the projection function without semantic-preserving supervision, our method performs less favorably than VGG16 + PCA-ITQ when the code size is small.

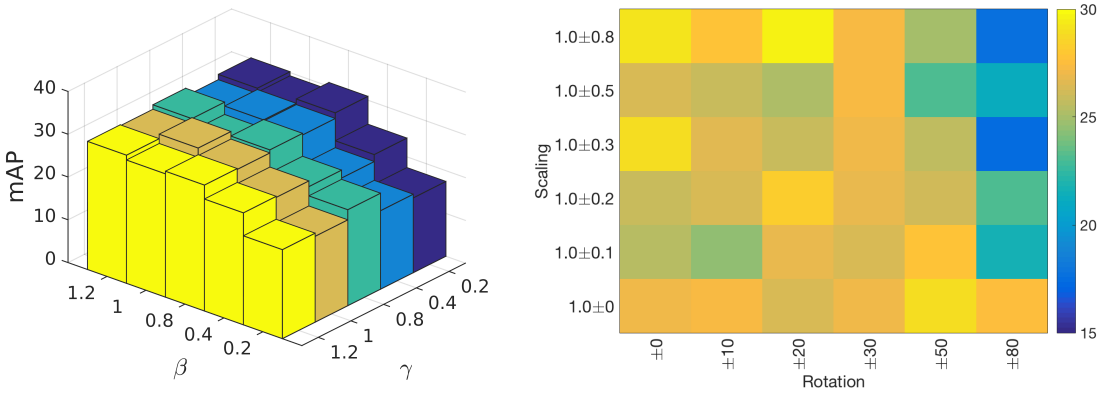
Following [108, 45], we report the mean Average Precision (mAP) of the top $1,000$ returned images of the recent state-of-the-art methods in Table 2.4. The results are consistent to previous finding. Our method achieves comparable or higher mAPs than the state-of-the-art deep learning approaches. Particularly, our method and DBD-MQ are initialized with the pre-trained parameters from ImageNet. The results show that network initialized with pre-trained parameters potentially avoids false local minima, and is useful to improve unsupervised learning of binary codes. Comparing to DBD-MQ which optimizes the binary descriptors with fine-grained multi-quantization, our proposed method is relatively robust against overfitting since our model produces binary codes with the standard sign function. It is worth noting that our method does not restrict the deep neural network architecture. We train our method on the VGG16-Avg, which is a simplified VGG16, and report the retrieval performance in Table 2.4. The retrieval performance is comparable to the counterparts. This suggests our proposed method can be applied on the simplified network architectures for efficient computation. We refer readers to Sec 2.4.7 for more detailed configurations of the simplified networks. Moreover, we study how our model retain the discriminative ability of the real valued representations. In Table 2.4, we show the performance comparison of our binary codes and its real-valued representations before quantization. The loss of binarization is relatively small when training on the VGG16-Avg network. Particularly, the VGG16-Avg is a simplified network with less parameters compared to the original VGG16 network so that it is easier for our model training. When training with the VGG16 network, E_1 mitigates the quantization loss especially when the code length is longer.

Effectiveness of the learning objectives: We study the influence of individual terms of the proposed objective function (in Eq. (5.13)). The objective function consists of three

Table 2.4: Performance comparison (mAP, %) of different unsupervised hashing algorithms on the CIFAR-10 dataset. This table shows the mean Average Precision (mAP) of top $1,000$ returned images with respect to different number of hash bits.

Method	16 bit	32 bit	64 bit
VGG16 + LSH	10.55	11.39	10.80
VGG16 + PCA-ITQ	27.69	28.46	30.63
DeepBit [108]	19.43	24.86	27.73
DBD-MQ [45]	21.53	26.50	31.85
Ours	**26.36**	**27.92**	**34.05**

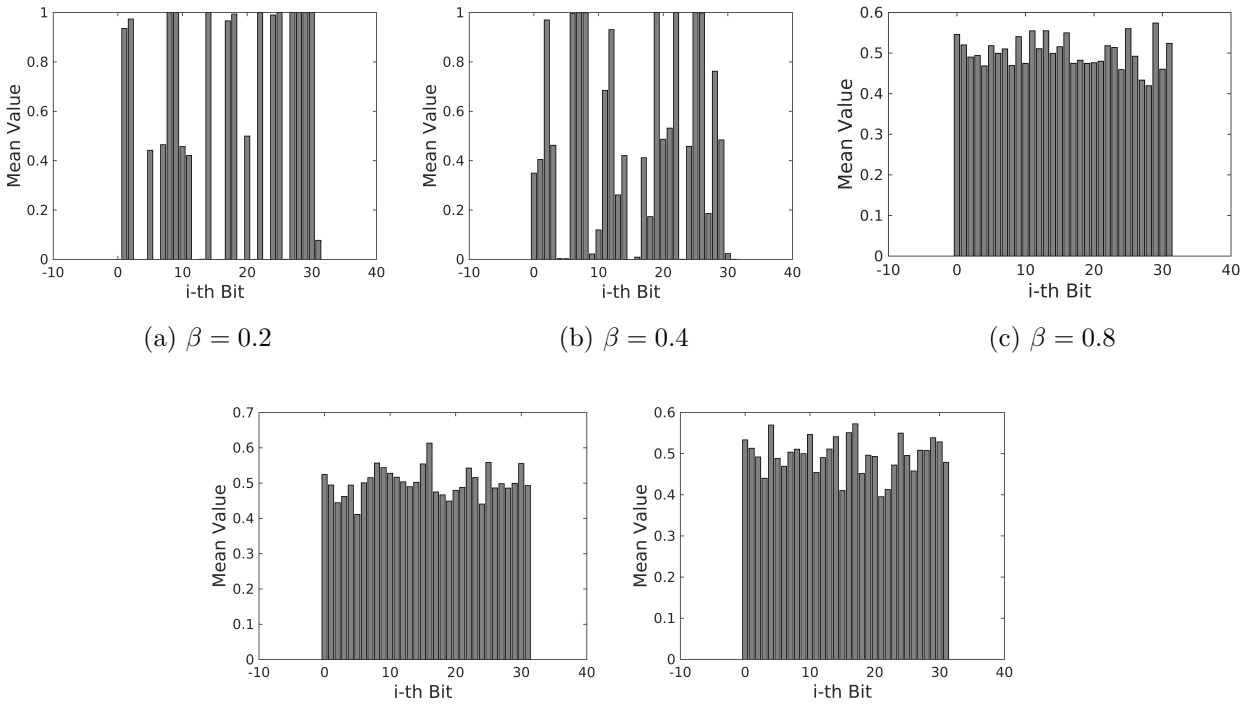
Method	16 bit	32 bit	64 bit
Ours, VGG16-Avg, before binarization	25.64	28.71	30.82
Ours, VGG16-Avg, after binarization	23.45	26.38	28.48
Ours, VGG16, before binarization	31.94	34.47	36.70
Ours, VGG16, after binarization	**26.36**	**27.92**	**34.05**



(a) (b)

Figure 2.5: Performance comparison (mAP, %) of DeepBit with different combinations of hyperparameters. The bit length is 32. (a) Different parameters of β and γ while α is set to 1, (b) Different scaling factors and rotation ranges.

terms that minimizing the binarization error, making codes evenly distributed, and learning transformation invariant bits. Since the quantization term E_1 in Eq. (5.13) is required to generate binary descriptors, we study how the other two objectives affect the performance. In particular, we fix α to 1.0, and iterate through all combinations of setting the parameter β and γ to 0.2, 0.4, 0.8, 1.0 and 1.2 on CIFAR-10 dataset. Fig. 2.5(a) shows the mAPs of the proposed method with different parameters. We see that E_2 is critical for learning effective binary codes since the retrieval performance is initially proportional to β, and converges to similar mAPs when β is higher than 0.8. We also see that our model has better mAPs given a higher γ, which indicates that E_3 is helpful to enhance the binary descriptors. Our model achieves the highest mAP when $\{\alpha, \beta, \gamma\} = \{1.0, 1.0, 1.0\}$. Hence, combining all these terms together is beneficial to achieve better performance. In Fig. 2.5(b), we study the effect of different combinations of the rotation and scaling parameters. Particularly, we observe that our model achieves the highest mAP with ± 20 rotation degrees and the scaling factors from 0.2 to 1.8. It is worth noting that training with larger rotation ranges and larger scaling factors may introduce larger variation of the training images. This may



(a) $\beta = 0.2$ (b) $\beta = 0.4$ (c) $\beta = 0.8$

(d) $\beta = 1.0$ (e) $\beta = 1.2$

Figure 2.6: Distribution of the binary codes when training with different weights of the objective function E_2. The generated binary codes are evenly distributed when the mean value for each bit is close to 0.5. The results show that our proposed objective function is effective for learning balanced binary codes.

reduce the visual similarity between the training images, and make the learning process of the proposed network more challenging. This is one of the key reasons that our model performs worse when training with large rotation ranges and large scaling factors. In Table 2.4, we compare our method with our prior work without learning transformation invariance [108]. The proposed model achieves more favorably mAPs than our prior work, which means that our new model retrieves more relevant samples from the database. The results show that the transformation invariance objective is more general than the rotation invariant objective. Not only the learned feature space is discriminative to different images, but also the binary descriptors computed from the transformed images are enforced to be centralized.

Analysis of the generated binary codes: To further understand the learned binary feature space, we show the distribution of the generated binary codes on the test set of CIFAR-10 with different settings. We extract the binary codes where the bit length is 32, and Fig. 2.6 shows the mean value for each bit when training with different β. The results show that the binary codes are better activated when applying a larger β. On the other hand, it is worth noting that we train our model with mini-batches, and the parameter N in Eq. (2.8) is 32 in the experiments. Particularly, all the training images are used for updating the network parameters after a number of iterations, and the information in the training set are learned by the model. This indicates that E_2 is still effective to estimate the feature distribution for learning balance binary codes during training with mini-batches.

2.4.4 Results on RomePatches Dataset

RomePatches consists of a series of color patches for descriptor evaluation. For both training and test set, each of them has $1,000$ feature points that are viewed in 10 different orientations, resulting in $10,000$ local patches. Following the evaluation protocol in RomePatches dataset [139], we select $1,000$ feature points as the queries, and the $9,000$ remaining feature points as the targets. Then, we compute mAP for evaluation. As suggested in the CIFAR-10 experiments, we simply choose the hyper-parameters $\{\alpha, \beta, \gamma\} = \{1, 1, 1\}$ in Eq. (5.13) for evaluation.

We compare DeepBit with several local descriptors including real-valued descriptors (SIFT [119], CKN [139], and convolutional features [96]), and binary descriptors (ORB [162], FREAK [6], BRISK [103]). Following the evaluation protocol in [139], Table 2.5 shows the performance comparison. Real-valued descriptors demonstrate higher mAPs than the binary ones in most cases. Since convolutional feature (AlexNet-conv5) is only translation invariance, it may not be able to match relevant patches with different orientations. Comparing the binary descriptors when the bit length is 256 (32 Bytes), our method produces more

Table 2.5: Performance comparison (mAP, %) of different binary descriptors on the RomePatches dataset. This table shows the mean Average Precision (mAP) of top $1,000$ returned patches.

Descriptor	Feature type	Bytes	mAP
CKN-grad [139]	Real-valued	1024	88.10
SIFT [119]	Real-valued	128	87.90
AlexNet-conv5 [96]	Real-valued	256	49.60
FREAK [6]	Binary	64	23.26
BRISK [103]	Binary	64	31.95
Ours	Binary	64	**53.04**
ORB [162]	Binary	32	43.83
Ours	Binary	32	**46.97**

favorably mAP than ORB. The result shows that our designed objective is helpful to enhance the binary descriptors. We find similar results when comparing with other 512 bits (64 Bytes) binary descriptors. DeepBit is capable of learning effective binary descriptors with different bit-lengths, which is consistent to the finding in the CIFAR-10 experiments.

2.4.5 Results on ILSVRC2012

ILSVRC2012 comprises more than 1.2 millions of images and $1,000$ object categories, which is more challenging then the above datasets compared. Different from the above experiments that the comparisons are made mostly with traditional learning approaches, we compare DeepBit with deep learning baselines in this experiment. Since our approach learns a set of non-linear projections that quantize the deep features to binary descriptors, we hence compare DeepBit with the combination of deep features + PCA-ITQ [61], which can be seen as the baseline in this experiment. PCA-ITQ [61] is one of the most representative

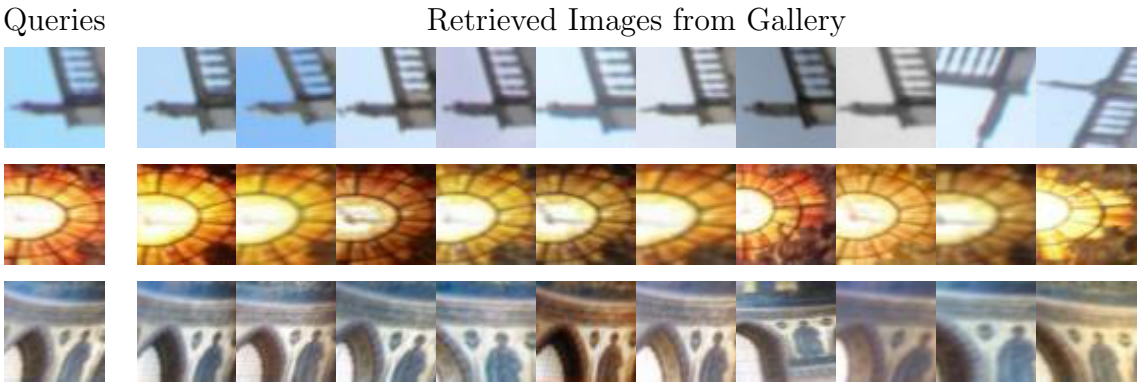


Figure 2.7: Query patches and the corresponding top ranked 10 patches retrieved from RomePatches dataset. The code length of our binary descriptor is 512.

Figure 2.8: Query images and their corresponding top 10 ranked images retrieved from ILSVRC2012 dataset. Upper two rows are the correct retrieval results. The bottom row shows our failure case. Images with red boarder are the false positives.

quantization approaches for binary codes learning. Since the deep CNN models are pre-trained on ILSVRC2012, we directly extract the 4096 dimensional features from the last fully-connected layer of the networks, and apply PCA-ITQ for binary codes quantization. To compare with the baseline approach which does not take into account the geometric transformation, we train our binary descriptors by minimize the loss in Eq. (5.13) with the first two terms for fair comparison.

In the experiment, we randomly select $1,000$ images from the validation set as the queries, and use all the images in the training set (around 1.2 millions of images) to form a large-scale database. Then, we compute the mAP based on the top $1,000$ returned images. As can be seen in Table 2.6, DeepBit achieves higher mAPs than the compared methods. The results suggest our method is able to learn discriminative binary descriptors in a large corpus. In addition, DeepBit produces consistently better performance than the baselines according to the precision at top n samples. This suggests that our approach better centralizes the relevant images in the feature space. Comparing to the baseline method, our proposed approach which learns deep features and binary descriptors in a single framework can perform more favorably against the compared multi-stage approaches (deep features + PCA-ITQ). Moreover, we report the performance of our method when training on AlexNet in Table 2.6. AlexNet is another well-known deep neural network which consists of 5 convolutional layers following by two fully connected layers. The results are consistent to that of the VGG16. This indicates our approach can be realized with different deep network architectures.

Fig. 2.8 shows the retrieved top 10 images from the ILSVRC2012 dataset with our 512 bits binary descriptors. As can be seen, our approach is able to retrieve relevant images which are visually similar. Our method also centralizes binary descriptor of the relevant images when they are in different viewpoints. However, we notice some failure cases such as the bottom row in Fig. 2.8. The category of the query is Coucal and the returned images with red boarder are Black Grouse. It may be difficult for the binary descriptors to distinguish some object categories which have visually similar patterns.

Table 2.6: The mAP at top 1,000 returned images and precision at n samples of methods on the ILSVRC2012 validation set. The code size is 512.

Method	mAP (%)	prec. (%) at n samples					
		5	10	15	20	25	30
AlexNet + PCA-ITQ	31.21	43.01	41.56	40.65	39.96	39.37	38.88
VGG16 + PCA-ITQ	47.07	57.82	56.73	55.99	55.40	54.97	54.63
Ours, AlexNet	31.68	47.84	45.84	44.48	43.55	42.80	42.07
Ours, VGG16-Avg	46.28	70.78	67.16	63.90	60.43	56.62	52.86
Ours, VGG16	49.75	66.68	64.87	63.80	62.95	62.32	61.56

2.4.6 Results on Instances Retrieval

To evaluate the robustness of the proposed method, we conduct experiments on the Oxford [143], Paris [144], INRIA Holidays [82], and UKB datasets [134] for instances retrieval applications, which include various scene types and the images are in different rotations, viewpoint and illumination changes. Since the object of interest may appear in different scales and viewpoints, these datasets present a challenging instance-level retrieval task. To apply our model on the instance-level retrieval, we follow the spatial search [154] approach, which divides the images into multiple local patches and measure the patch-level similarity. Specifically, the similarity between a query sub-patch and a gallery image is defined as the minimum among the Hamming distances of the query sub-patch and the gallery sub-patches. Then, the similarity between a query and a gallery image is defined as the average Hamming distance of every query sub-patches to the gallery image.

Following [13], we train our method on the Landmark dataset [13] for learning binary codes, and test our model on Paris, Oxford, and Holidays datasets. Table 2.7 shows the performance comparison with the state-of-the-art approaches. Among these compared ap-

proaches, ITQ [60] and Neural codes [13] compute binary codes for retrieval, and the rest approaches are based on real-valued features with 4096 [154, 218] or higher dimensions [83]. In Table 2.7, we show that deep learning approaches perform better than SIFT-based methods. Our method performs more favorably than the approaches based on other binary codes [13, 60] with the same code length. We also see that our method performs more favorably against CNN+aug+ss [154] in the Paris dataset. These results show that the proposed transformation invariant objective is effective for learning binary codes. Moreover, we conduct performance comparison on the UKB dataset [134]. Following the standard evaluation protocol in [134], Table 2.8 reports the precision at the top 4 returned images of different methods. The performance of the proposed method is comparable or slightly better than the counterparts, which indicates that the proposed method is relatively insensitive to the transformation variations of the input images.

2.4.7 Results on Fine-grained Recognition

Unlike previous binary descriptors that require matched/non-matched labels during training, the proposed DeepBit learns compact binary descriptors in an unsupervised manner; thus, DeepBit is flexible for various applications. In this section, we extend the evaluation to fine-grain recognition, which focuses on recognizing the subclasses of the same object category, e.g., recognizing Cowslip and Buttercup, which are both belong to flower category. Flower recognition is a classic visual analysis task, and it is challenging due to the variation of shapes, color distributions, and pose deformations. Besides, the computational cost and memory requirement become demanding while one wants to recognize the flowers in the wild using mobile devices. We show that the proposed binary descriptor performs more favorably against some basic real-valued descriptors such as HOG [36], and SIFT [119]. We further demonstrate that our approach can be realized on the compressed or simplified deep networks, and still maintains comparable recognition rate.

Table 2.7: Performance comparison of instances retrieval performance (mAP, %) of different methods on Paris, Oxford, and Holidays datasets.

Method	Paris	Oxford	Holidays
SIFT-based methods			
BOW-200k-D [144]	46.0	36.4	54.0
IFV [83]	-	41.8	62.6
CNN-based methods			
ITQ, 256 bits [155, 60]	66.3	48.9	67.1
ITQ, 512 bits [155, 60]	66.8	50.8	73.9
Neural codes + PCA, 256 bits [13]	-	55.7	78.9
Neural codes + PCA, 512 bits [13]	-	55.7	78.9
CNN + aug + ss [154]	79.5	68.0	84.3
Ng *et al.* [218]	69.4	64.9	83.8
DF.FC1 + SL [196]	86.8	46.5	-
ReDSL.FC1 [196]	94.7	78.3	-
Ours+ss, 256 bits	82.5	60.3	81.8
Ours+ss, 512 bits	**82.9**	**62.7**	**82.7**

Fine-grain recognition evaluation

To get more insight about our binary descriptors, we train the multi-class SVM classifiers with different features such as Colour, SIFT, HOG and our binary descriptor. Then, we evaluate the quality of the feature representations according to the classification accuracy. In this experiment, we directly use our pre-trained DeepBit model presented in Sec.**??** to compute the binary descriptors. Note that we use one descriptor per image for SVM training. Table 2.9 compares the classification accuracy of the 17 categories flowers using different descriptors

Table 2.8: Performance comparison (Precision, %) of the top 4 returned images of different methods on UKB dataset.

Method	Feature Type	Dimension	Precision
VGG16	Real-valued	4096	84.40
VGG16 + LSH	Binary	512	82.35
VGG16 + PCA-ITQ	Binary	512	82.25
Neural codes + PCA [13]	Binary	512	82.50
Ours	Binary	512	**82.55**

proposed in [132, 133], including low-level (Colour, Shape, Texture), and high level (SIFT, and HOG) features. The results show that DeepBit is possible to achieve comparable or better fine-grain recognition rate than other existing features. We owe this to the fact that our approach optimizes the projection function, which maps the features extracted from the deep neural network to the binary string. In other words, our binary descriptor can be seen as the abstract of the rich feature representations computed from the deep neural network. Fig. 2.9a shows the confusion matrix results.

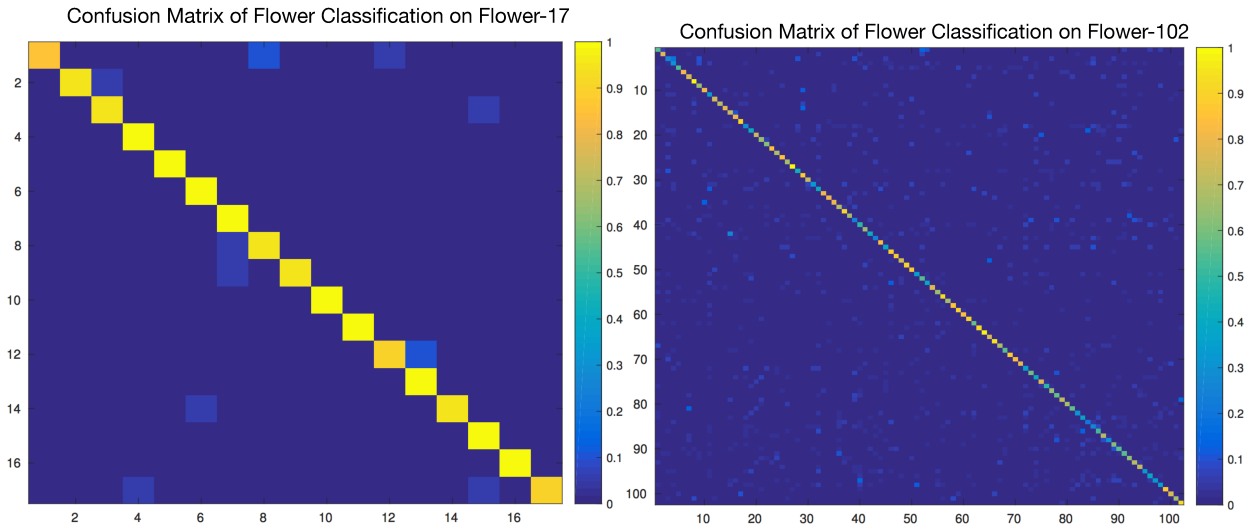
We also test our binary descriptor on a large flower dataset with 102 categories. Table 2.10 reports the accuracy of the SVM classifier when training with different feature descriptors, and Fig. 2.9b shows the confusion matrix of the 102 flower classes. Our binary descriptor (computed based on AlexNet) achieves 61.9% accuracy, which is comparable to many descriptors but worse than CNN-SVM [154] and BOW [51]. However, [154] and [51] employ high-dimensional real-valued features for classifier training. CNN-SVM [154] takes the original 4096 dimensional deep features computed from AlexNet. BOW [51] involves a series of pre-processing, including image segmentation to remove the background regions, and enhancing the descriptors by combining multiple features resulting in 4000 dimensional floating-valued descriptors. In contrast, our approach is more efficient because our descrip-

Table 2.9: The categorization accuracy (mean%) for different features on the Flower-17 dataset.

Descriptors	Accuracy
Colour [132]	60.9
Shape [132]	70.2
Texture [132]	63.7
HOG [36]	58.5
HSV [133]	61.3
SIFT-Boundary [133]	59.4
SIFT-Internal [133]	70.6
DeepBit, 512 bits, AlexNet	**89.2**
DeepBit, 512 bits, VGG16	87.5

Table 2.10: The categorization accuracy (mean%) for different features on the Flower-102 dataset.

Descriptors	Accuracy
HSV [133]	43.0
SIFT-Internal [133]	55.1
SIFT-Boundary [133]	32.0
HOG [36]	49.6
BOW [51]	65.5
CNN+SVM [154]	74.7
DeepBit, 512 bits, AlexNet	**61.9**
DeepBit, 512 bits, VGG16	58.6



(a) (b)

Figure 2.9: Confusion matrix of flower classification using the proposed binary descriptor. (a) Flower-17 dataset. (b) Flower-102 dataset.

tors are only of 512-bit binary codes.

Learning with simplified deep neural networks

Since the ultimate goal of binary descriptors is efficient matching, we investigate the possibility of learning binary codes with the simplified deep neural network. We adopt two more network architectures, SqueezeNet [76] and VGG16-Avg (of our own design [213]), for fine-grain flower recognition. Table 2.11 summarizes the required storage size and the number of parameters of different network designs. SqueezeNet [76] decreases the parameters of AlexNet by replacing 3×3 filters with 1×1 filters, and maintains the accuracy with large activation map by max pooling with stride of 2 in the deeper layers. VGG16-Avg [213] reduces the parameters in VGG16 by replacing the fully-connected layers with an average polling layer. Since the average polling layer preserves the spatial information from the previous convolutional layers, the reduced features are still effective for visual recognition.

To learn our binary descriptors, we replace the last softmax layer of the SqueezeNet with

Table 2.11: Parameters and storage size of different network models implemented with CAFFE.

	DeepBit-512			
	AlexNet	VGG16	VGG16-Avg	SqueezeNet
# parameters	57 M	134 M	15 M	0.9 M
required storage	228 MB	537 MB	49 MB	6MB

Table 2.12: Recognition accuracy (mean%) of our method with different deep neural networks on Flower-17 and Flower-102 datasets. We also report the recognition accuracy of SIFT for reference.

Descriptors	Flower-17	Flower-102
SIFT-Boundary [133]	59.4	32.0
SIFT-Internal [133]	70.6	55.1
DeepBit, 512 bits, SqueezeNet	76.8	33.7
DeepBit, 512 bits, AlexNet	89.2	61.9
DeepBit, 512 bits, VGG16	87.5	58.6
DeepBit, 512 bits, VGG16-Avg	**90.7**	**62.8**

a fully-connected layer. On the other hand, we add a new fully-connected layer after the average polling layer in VGG16-Avg. Both fully-connected layers have 512 neurons, and their parameters are initialized with random Gaussian. Then, we enforce the proposed criteria on these neurons to learn binary descriptors, and optimized the network parameters though back-propagation. Table 2.12 shows the fine-grain recognition rates using different networks. Overall, DeepBit achieves better performance than SIFT in most cases when using different networks. This indicates it is feasible to realize DeepBit with the relatively cheaper networks, and maintains comparable performance. Note that DeepBit + SqueezeNet produces comparable accuracy to SIFT, but worse than that of DeepBit + AlexNet. Since the compression process usually requires strongly supervision to maintain accuracy, the results suggest our approach, which belongs to unsupervised learning, may slightly drop the performance during deep network compression. However, the recognition rate of our binary descriptor is still comparable to some floating-valued descriptors such as SIFT-Boundary. Moreover, our study shows that DeepBit + VGG16-Avg not only reduces the model size, but achieves more favorably accuracy against DeepBit + the others. The main idea of VGG16-Avg is reducing the model parameters by down-sampling the feature maps, and further decreasing the number of parameters in the following fully-connected layers. Since the average polling layer does not require parameter learning, it is easier for our unsupervised approach to learn binary descriptors comparing to other compression approaches such as SqueezeNet. Our study shows that the proposed approach can be realized on a small or light-weighted deep network, and enable efficient visual recognition.

2.4.8 Computational Cost

Encoding time: We randomly select $1,000$ test images from CIFAR-10 dataset, and compute the average computational time. We report the encoding time of our approach on CPU and GPU, and the feature extraction time for both conventional binary descriptors and hashing methods. The experiments are carried out on a machine with an Intel Xeon 3.70 GHz CPU of 8 cores, and an NVIDIA Titan-X with CUDA-7.0. Fig. 2.10 shows the encoding

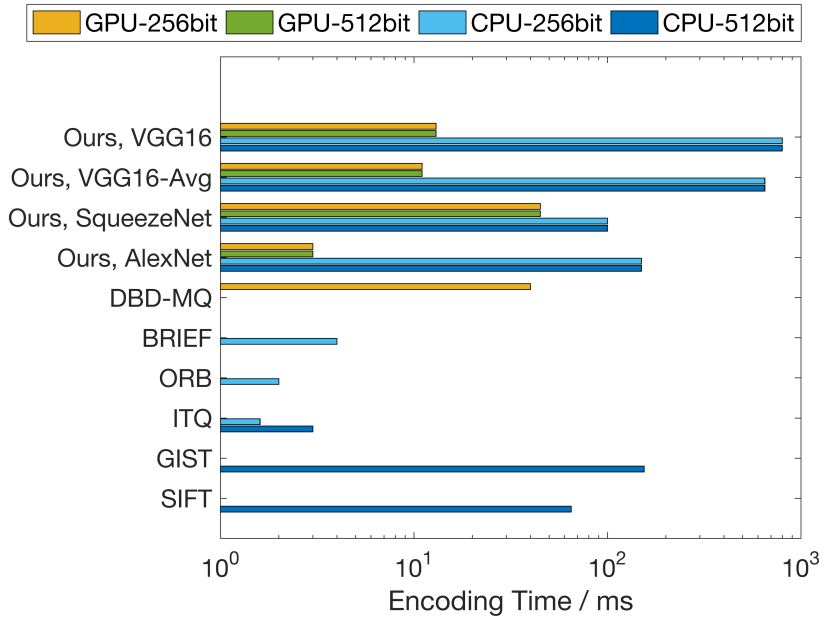


Figure 2.10: Computational cost for encoding an input image on CIFAR-10.

time (in milliseconds) of different methods. Overall, considering only the encoding time, the conventional methods (such as ITQ) are much faster than deep learning approaches. However, when taking feature extraction time into consideration (such as SIFT+ITQ), the computational time of the conventional methods is comparable or slower than some of our models. In addition, as discussed in [116], conventional methods may require additional processes (such as BOW or multiple features) to reach comparable retrieval performance. On one hand, since the major computational time of our method lies in the forward-pass of the network layers, "Ours, AlexNet" (7 layers) is faster than the very deep "Ours, VGG16" network (16 layers). In the near future, our method could be faster when employing the compressed deep networks. However, this is outside the major scope of this work. On the other hand, it is worth noting that the computational time of our model is independent of the code lengths. When using the same network backbone, the encoding time of our 256-bit and 512-bit models remain the same. This is an advantage comparing to conventional methods

such as ITQ.

Time complexity for image retrieval: The pipeline of image retrieval can be divided into (1) offline extracting binary codes of the images, and (2) online querying using the binary codes. Given the database consisting of n images, the offline process computes a single forward-pass for each image in the database, which takes time $O(n)$. Given a novel query, we firstly extract the binary codes of the query, and then perform linear search that computes the hamming distance between the query and each item in the database resulting in the time complexity $O(n)$. Though the complexity is linear to the number of images in the database, computing the hamming distance via XOR operation is very fast. It is worth noting that we primarily focus on learning discriminative binary codes for improving the retrieval performance, and the time complexity could be further reduced to sub-linear by using advanced implementations such as multiple hash tables or exploring the Hamming ball volume around the query [64, 199].

2.5 Conclusion

In this chapter, we have presented an unsupervised learning framework to learn compact binary descriptors. Our approach does not require label annotations provided by the dataset, and is more practical to real-world applications compared to supervised binary descriptors. Experiments on several public benchmark databases demonstrate that our method achieves better performance than the state-of-the-art feature descriptors in most cases. How to simultaneously learn binary descriptor and its optimal code size seems to be an interesting future work.

Chapter 3

CROSS-DOMAIN COMPLEMENTARY LEARNING USING POSE FOR MULTI-PERSON PART SEGMENTATION

3.1 Introduction

Human body part segmentation [27, 28, 29, 206] aims at partitioning persons in the image to multiple semantically consistent regions (*e.g.*, head, arms, legs), which is important to many human analysis applications [65, 67, 78]. Supervised training with deep Convolutional Neural Networks (CNNs) significantly improves the performance of various visual recognition tasks including the human body part segmentation [28, 117, 227]. However, it requires large amount of training data. Data labeling, especially at pixel level, is labor intensive and the acquisition of such annotations in large scale is prohibitively expensive.

A promising solution to address this problem is to take advantage of the graphics simulator to generate synthetic images with ground truths automatically [121, 158, 161]. For example, previous study [191] proposed to learn single-person part segmentation by directly training the neural networks using synthetic images. However, their method usually produces false alarms in the background, and it does not work well for real-world images consisting of multi-person with interactions and occlusions. Also, recent studies [179, 183, 192] show that the discrepancy of the pixel value statistics between real and synthetic data, so called the domain gap, makes it challenging to transfer knowledge from synthetic data to real data.

To address the discrepancy of the pixel value statistics between the two domains, recent studies [157, 187, 189] proposed to train the neural networks using adversarial training for matching the feature distributions of the real and synthetic data. They proposed to train a discriminator for distinguishing the real and synthetic images, and a generator for extracting the domain-invariant features that can fool the discriminator. However, the adversarial

training may not converge due to the fact that it is difficult to maintain a balanced training between the generator and the discriminator. Previous approaches also suffer from the issue of mode collapse, where the generator may only capture a part of the real data distribution. Thus, the performances of previous approaches are much worse than the supervised training on real data with pixel-wise manual labeling.

In this chapter, we observe that real and synthetic humans both have a skeleton (pose) representation and show that the skeletons can effectively bridge the synthetic and real domains during the training. With our proposed approach, we can take advantage of the complementary nature of the real and synthetic data, i.e., rich and realistic variations of the real data and the easily obtainable labels of the synthetic data, effectively. Our technique learns multi-person part segmentation on real images without any human-annotated labels and achieves performance comparable to several state-of-the-art approaches which require human labeling. On the other hand, if part labels are also available in the real-images during training, our method outperforms the supervised state-of-the-art methods by a large margin.

As shown in Figure 5.3, we have part segmentation labels from synthetic data, but do not have part segmentation labels from real data. It should be noted that the real and synthetic images in the figure are unrelated. We observe that real and synthetic humans both have a skeleton representation. By learning the skeleton representation of the real and synthetic humans, our proposed model learns a shared feature space for both real and synthetic domains. Different from previous works that try to minimize the discrepancy of the pixel value statistics between the domains, we propose to perform human pose estimation to extract skeletons from the real and synthetic images, and minimize the discrepancy of the feature spaces between the two domains by using the semantic prior of the human body structure. The automatically extracted skeletons capture the structural body information and can effectively bridge the real and synthetic data domains, so that both real and synthetic data can be used in the training effectively without needing the expensive manual human part labeling for the real images.

It is worth noting that the learning of human pose estimation requires training labels.

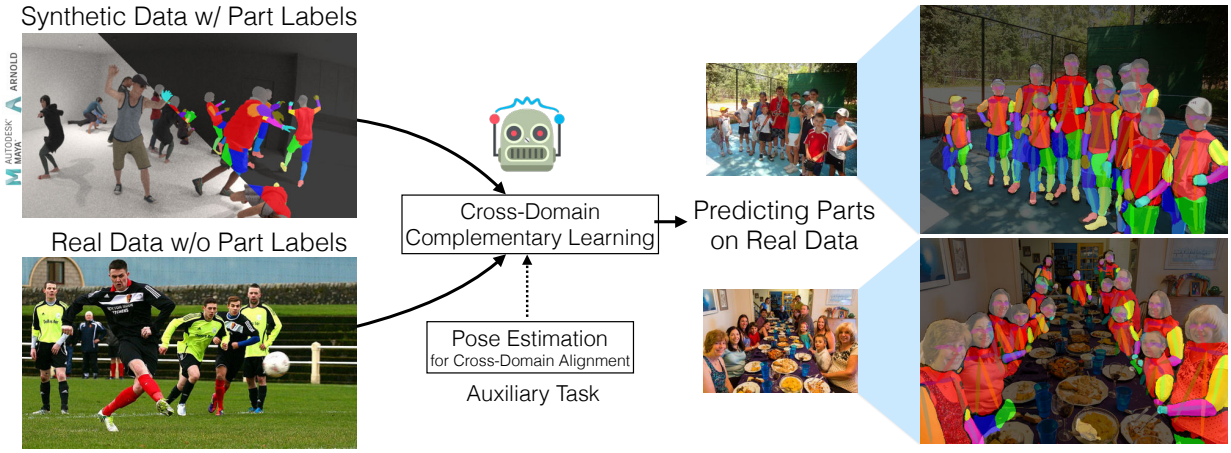


Figure 3.1: We address the problem of learning multi-person part segmentation without human labeling. Our proposed complementary learning technique learns a neural network model for multi-person part segmentation using a synthetic dataset and a real dataset. We observe that real and synthetic humans share a common skeleton structure. During learning, the proposed model extracts human skeletons which effectively bridges the synthetic and real domains. Without using human-annotated part segmentation labels, the resultant model works well on real world images.

However, the pose labels are readily available on several public large-scale datasets like COCO Keypoint dataset [113] and are easy to obtain than part segmentation labels. Thus, the proposed method has the advantage of saving labeling efforts in practice.

We also show that we can extend our method to label keypoints for real images. For example, to label keypoints on hands and feet, we just need to generate synthetic images with hands and feet labels, and the knowledge will transfer from the synthetic domain to the real domain using our proposed approach.

In summary, the main contributions of this work include:

- We discover that human pose is very effective to bridge the real and synthetic domains for human-centric analysis applications.

- We introduce an effective framework, called cross-domain complementary learning with pose, to leverage information in both real and the synthetic images for multi-person

part segmentation.

- Through experiments, we show that without any human-annotated part segmentation label, our method performs comparably with several state-of-the-art approaches which require human labeling on Pascal-PersonParts and COCO-DensePose datasets. On the other hand, if parts labels are also available in real images during training, our method outperforms the supervised state-of-the-art methods by a large margin.

- We show that our method can be extended to generate labels for keypoints such as those on hands and feet in real images without human labeling.

3.2 Related Works

3.2.1 Synthetic data for computer vision tasks

There has been a long-standing history of exploring the use of 3D synthetic data for computer vision problems [107, 128, 171]. Recent studies use 3D CAD models for visual recognition tasks, such as 3D model repository [26, 204], object recognition [142, 148, 183], human analysis applications [78, 121, 191, 192], and semantic segmentation for urban scenes [161]. Among the literature, Varol *et al*. [191] proposed to render a single-person avatar on top of a static background image, and generate ground truths for training deep CNNs. However, their method does not leverage real data, which contains realistic variations and the real world scenarios. Thus, their method only works for the well-controlled environment and the single-person scenario in an image. In contrast, we address a more challenging and more general scenario, where multiple people with interactions and occlusions are considered. Different from training the deep CNNs using synthetic data only [191], we propose to leverage the complementary natural of the real and synthetic data with human pose estimation. In the experiments, we show that our method, which learns to bridge the reality gap, performs more favorably against those proposed in previous studies [191]. In addition, as demonstrated in the experiments, our technique reduces the requirement on the synthetic data quality because

it effectively learns from real data.

3.2.2 Domain adaptation

Domain adaptation is a special case of transfer learning [137] that aims to learn a single task from a source domain, so that it performs well on a target domain. Many approaches have been proposed to address the *visual* dataset bias [181] for domain adaptation, including active learning with human-in-the-loop [192], training deep CNNs with reverse gradient [53], learning with auxiliary tasks to reduce domain variations [18, 217], and matching feature distributions of two domains by adversarial training [157, 187, 189]. In particular, Ren and Lee [157] proposed to learn image classifiers and object detectors using synthetic images with adversarial training. Instead of adversarial training, our approach uses an auxiliary task of human pose estimation to bridge synthetic and real domains which is shown to be more effective from our experiments.

3.2.3 Multi-task learning

Prior works [44, 57, 69, 104, 137, 147, 231] have shown that multi-task learning is effective for many vision problems. Given multiple different tasks, where a subset of these tasks are related, multi-task learning aims to improve the learning of the original task by using knowledge from all or some of the other tasks [163, 224]. However, most of the previous studies assume that, for all the tasks, the labeled data have to be available for training [137]. Different from previous works, our method learns without human-annotated segmentation labels in a cross-domain scenario, and learns to bridge the domain gap between real and synthetic data.

3.2.4 Supervised and semi-supervised part segmentation

Recent studies [42, 59, 99, 207, 208] proposed to jointly train human part segmentation and human pose estimation for improving the performance of part segmentation. However, the

successes of the previous studies are mainly attributed to the supervised training with the pixel-wised manual labeling. Different from the strongly supervised approaches, we propose to remove the labeling requirement by learning with synthetic data, which is more applicable to real-world applications. On the other hand, Fang *et al*. [48] proposed a semi-supervised approach that aims to augment training samples by transferring the human-labeled part segmentation from an existing dataset to another unlabeled dataset. Our method differs from theirs in that our method does not require any human-labeled part segmentation dataset at all.

3.3 Synthetic Data

It is a common belief that high-quality synthetic data should be created as similar as possible to the real-world scenarios. For example, in generating single-person synthetic data [191], the authors composed their synthetically generated human images with a variety of real world background images. An advantage of our technique is that we reduce the requirement on the synthetic data quality. In particular, we use a simple empty room as the background for all of our synthetic data. The reason why our technique works well even with such a simple synthetic background is that our technique learns about the background from the real data.

We have 20 3D human models with different body shapes and clothing. These avatars are randomly placed at different positions in the virtual room, and they are animated to perform a variety of actions such as walking, jumping, crawling, etc. To create realistic human motions, we retarget the motion capture data from CMU MoCap database [4] to the avatars. We use a ray-tracing based rendering engine [2, 3] to render the scene.

Multiple virtual cameras are set up at different positions in the environment to capture the scene from a variety of viewpoints. Figure 3.3 shows the layout of our simulation environment. The virtual camera model we used is a pinhole camera with a 90 degree FoV. The exposure of the camera is 1/30-th of a second. The focal length is 35 mm.

Figure 3.2 shows the examples of our synthetic data and the ground truths. Our graphics simulator generates different types of per-pixel ground truth labels for the animations.

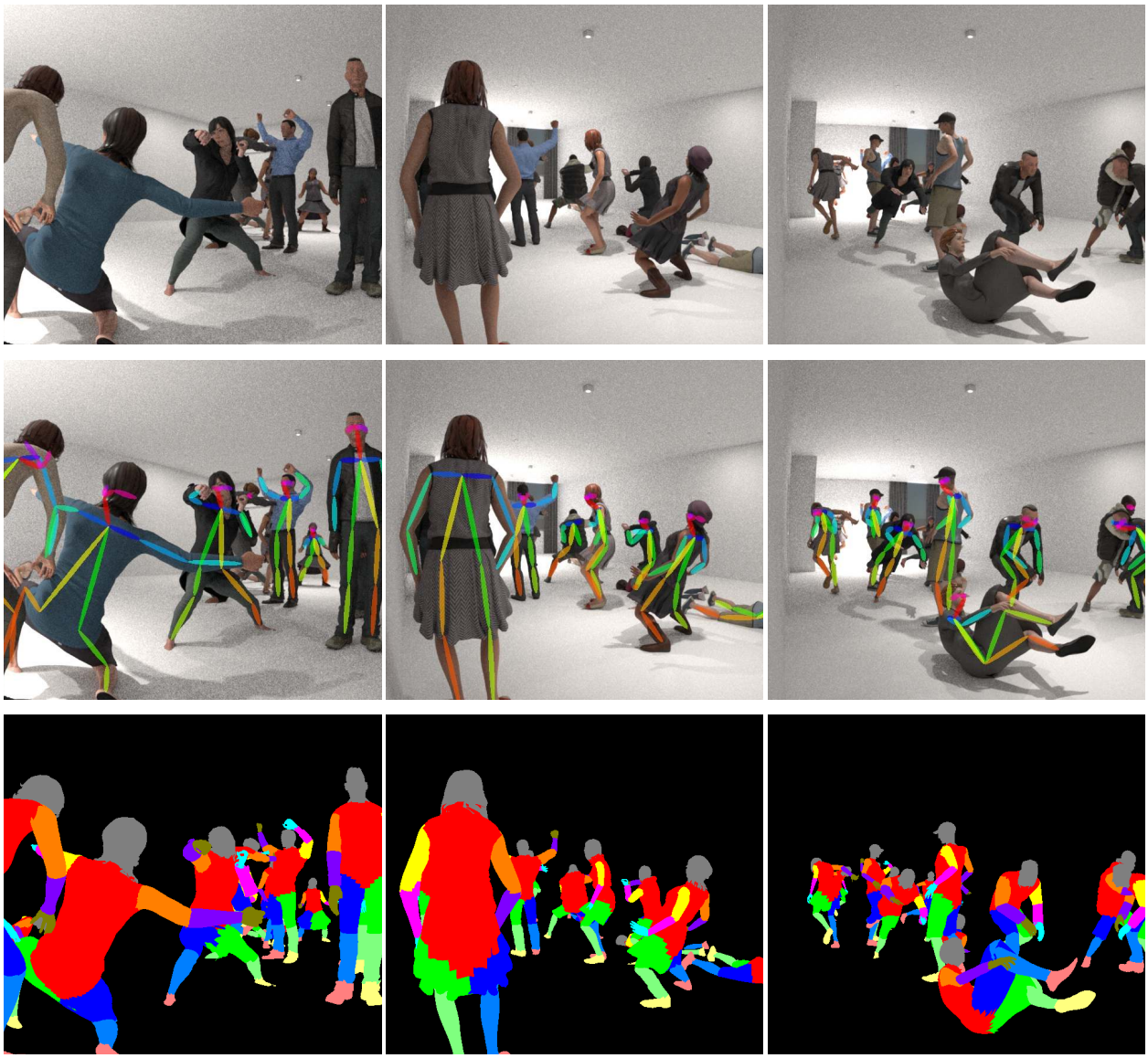


Figure 3.2: Samples of our synthetic data. Our synthetic data contain multiple persons performing various actions in a 3D room. Top row: the synthetic RGB images. Middle row: the synthetic pose labels. Bottom row: the synthetic part labels.

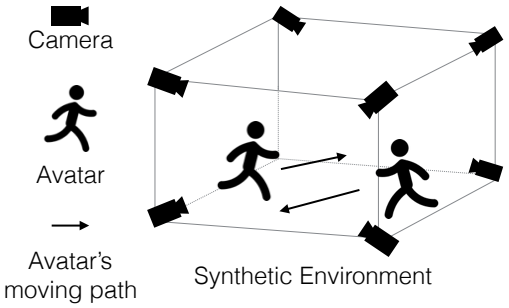


Figure 3.3: The layout of our synthetic environment. We render multiple avatars performing different actions in a 3D room, and capture the animations from multiple different viewpoints.

Following the common definitions of body parts and human pose [67, 113], we generate 14 categories of body part ground truth labels, and 17 types of keypoint ground truth labels. It is worth noting that the labels for the synthetic data can be freely extended depending on user preferences, and are more flexible than those in the conventional real datasets. For example, as shown in Section 3.5.6, we generate a new set of keypoints including hands and feet from synthetic data thus allowing our model to predict new keypoints.

Another advantage of the graphics simulation is that we can easily generate large amount of data. In this work, we generate a total of $17,211$ frames and their corresponding ground truths for model training.

3.4 Proposed Approach

Assume we have a set of synthetic data with human part segmentation labels. We would like to learn a function that performs human part segmentation on real world data. If we directly train a neural network with synthetic data labels, it does not generalize well to real data due to the reality gap. Unlike existing methods [157] that try to transform the synthetic

data to real data domain to make them look similar to each other, we use a complementary learning strategy that effectively leverages the rich variation of the real data and the part segmentation labels of the synthetic data. To make sure the synthetic data and real data are aligned in a common latent space, we use an auxiliary task, pose estimation, to bridge the two domains. In summary, our training data consist of part segmentation labels and pose labels from synthetic data, and pose labels from real data. We learn a part segmentation function without any part segmentation labels from real data.

3.4.1 Learning objective

Assume we have a real dataset with pose labels D_r^{pose}, a synthetic dataset with pose labels D_s^{pose}, and a synthetic dataset with part segmentation labels D_s^{part}, we formulate the cross-domain complementary learning (CDCL) as the following optimization problem:

$$L = L_{pose}(D_r^{pose}) + L_{pose}(D_s^{pose}) + L_{part}(D_s^{part}), \tag{3.1}$$

where L_{pose} is the loss function for pose estimation, and L_{part} is the loss function for part segmentation. The first two terms together form the objective function for learning the auxiliary task of pose estimation from both real and synthetic data. The third term learns part segmentation from synthetic data.

Following the common definition of pose labels [23, 113], we use the annotations of keypoints and Part Affinity Fields (PAFs) [23] for learning pose estimation. In particular, let $D_r^{pose} = \{I_r^i, \bar{K}_r^i, \bar{P}_r^i\}_{i=1}^M$, where M is the total number of real images, $I_r \in R^{w \times h \times 3}$ denotes a real RGB image, $\bar{K}_r \in R^{w \times h \times J}$ denotes a real keypoint ground truth, which has J different maps, one per keypoint, $\bar{P}_r \in R^{w \times h \times C}$ denotes a real part affinity ground truth, which has C affinity vector fields. Also, we have a synthetic dataset with pose labels $D_s^{pose} = \{I_s^i, \bar{K}_s^i, \bar{P}_s^i\}_{i=1}^N$, where N is the total number of images in the synthetic data. Furthermore, we have a synthetic dataset with part segmentation labels $D_s^{part} = \{I_s^i, \bar{B}_s^i\}_{i=1}^N$, where $\bar{B}_s \in R^{w \times h \times Z}$ is the synthetic body part segmentation ground truth and Z is the total number of body part categories. Note that it is convenient to assume D_s^{pose} and D_s^{part} share the same

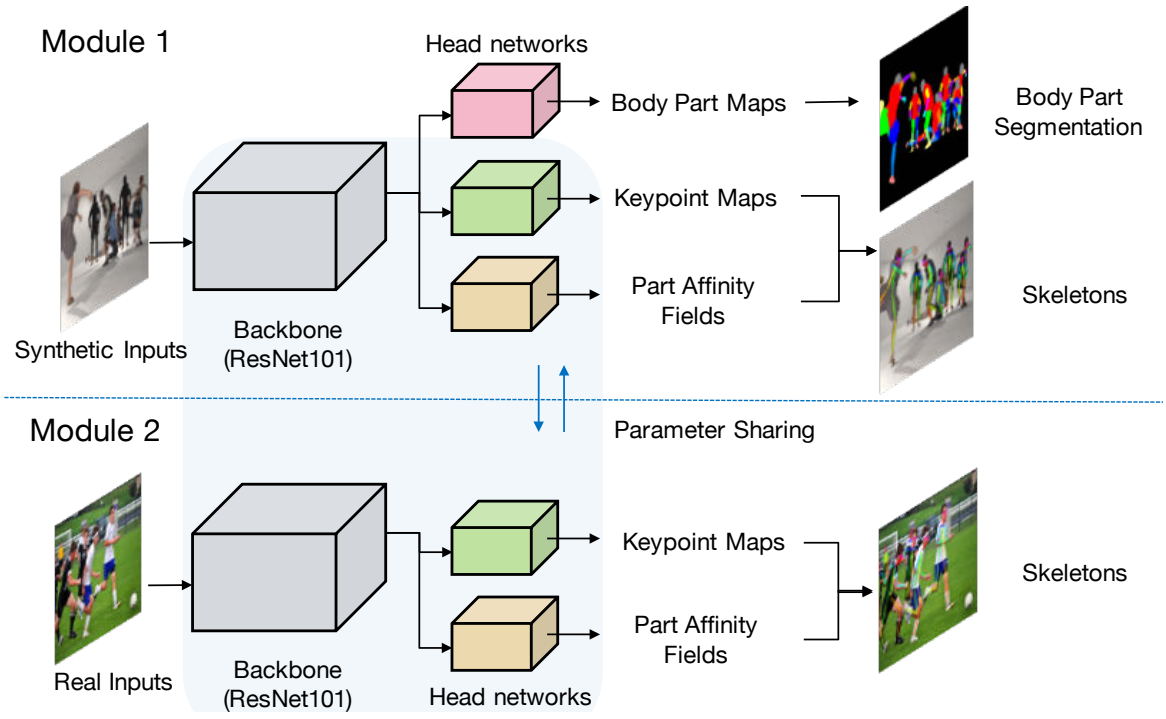


Figure 3.4: An overview of the proposed framework. Our framework consists of two main components. The first is the synthetic input training to learn body parts and human poses on the synthetic domain. In the second component for real input training, we share the network parameters of the backbone, keypoint map head, and part affinity field head with the first component. During learning, we train our network using two modules within a mini-batch, and optimize the network using back-propagation.

set of images. In this work, we use COCO Keypoint dataset [113] as D_r^{pose}.

In the following, we omit the subscript r and s and use D^{pose} to represent either real or synthetic data. The loss function we use for learning pose estimation is $L_{pose}(D^{pose}) = L_{kpts}(I, K, \bar{K}) + L_{paf}(I, P, \bar{P})$ where $L_{kpts}(\cdot)$ and $L_{paf}(\cdot)$ are the Euclidean loss functions minimizing the differences between the predictions and the ground truths, and they are defined below:

$$L_{kpts}(I, K, \bar{K}) = \sum_{j=1}^{J} \sum_{\theta} \mathcal{M}(\theta) ||K(\theta) - \bar{K}(\theta)||_2^2, \tag{3.2}$$

$$L_{paf}(I, P, \bar{P}) = \sum_{c=1}^{C} \sum_{\theta} \mathcal{M}(\theta) ||P(\theta) - \bar{P}(\theta)||_2^2, \tag{3.3}$$

where K and P denote the predicted keypoint confident map and the predicted part affinity field, respectively, and $\bar{K}$ and $\bar{P}$ denote the ground truths. $\mathcal{M}$ is a binary mask, where $\mathcal{M}(\theta) = 0$ if the ground truth is missing at the location θ of the image. The mask is used to avoid penalizing the correct predictions as discussed in the literature [23].

The loss function of learning part segmentation is denoted as $L_{part}(D^{part}) = L_{part}(I, B, \bar{B})$ which is defined to be the categorical cross entropy loss for classifying pixels to different human parts, that is:

$$L_{part}(I, B, \bar{B}) = -\sum_{z=1}^{Z} \sum_{\theta} \mathcal{M}(\theta) \bar{B}(\theta) \log(B(\theta)), \tag{3.4}$$

where B denotes the predicted body part maps, $\bar{B}$ denotes the synthetic part segmentation ground truths.

In summary, the overall objective function is

$$\begin{aligned} L = & L_{kpts}(I_r, K, \bar{K}_r) + L_{paf}(I_r, P, \bar{P}_r) \\ & + L_{kpts}(I_s, K, \bar{K}_s) + L_{paf}(I_s, P, \bar{P}_s) \\ & + L_{part}(I_s, B, \bar{B}_s). \end{aligned} \tag{3.5}$$

3.4.2 Network architecture

Figure 4.1 illustrates the proposed network. Our network takes an image of arbitrary size as input, and predicts three different outputs including (1) a set of body part segmentation maps B, (2) a set of confidence keypoint maps K, and (3) a set of Part Affinity Fields (PAFs) P [23]. For clarity, we describe our network in two components: *backbone* and *head* networks.

Backbone network

The backbone network is for extracting feature maps from the input. In this chapter, all the results are obtained by using ResNet101 [70] with pyramid connections [112] as our backbone network. We denote f as our backbone network, and the output of our backbone is $F = f(I)$, where I is an input image.

Head network

We detect multi-person body parts and human poses in a bottom-up strategy, which is in spirit similar to OpenPose [23]. Our network predicts three target outputs in parallel, which are B, K, and P. Each head network is a fully convolutional network. Note that this is different from prior studies [23, 201] that have a cascaded multi-stage head architecture. Our head networks do not have such a cascaded design, and can be seen as a single-stage network compared to prior works. Finally, we denote the three head networks as α, β, and γ, respectively. The body part segmentation maps B are computed by $B = \alpha(F)$, where F is the output of our backbone. The confidence keypoint maps K are computed by $K = \beta(F)$, and the Part Affinity Fields [23] P are computed by $P = \gamma(F)$.

3.4.3 Training

During training, we randomly pick an equal number of real and synthetic images to form a mini-batch, and feed it to the network. Then, we compute the loss using Eq(5.13), and

update the network parameters via Adam optimizer. The training batch size is set to 10.

3.4.4 Inference

During testing, we only predict the part segmentation. Our model predicts 14 body part score maps and one background score map. Following DeepLab [28], we run multi-scale inference and perform max-pooling to obtain the final part score maps. The part segmentation is derived by using the argmax value from the final part score maps.

3.5 Experimental Results

We trained our model with COCO Keypoint dataset [113] and our synthetic dataset. We then evaluated the performance of the resulting model on two public benchmarks, the Pascal-Person-Parts [31], and the COCO-DensePose [67].

3.5.1 Evaluation benchmarks

Pascal-Person-Parts [31] is a challenging dataset for multi-person body part segmentation. It consists of $1,716$ training and $1,817$ test images, where the human body is split into 6 different parts including head, torso, upper and lower arms, as well as upper and lower legs.

COCO-DensePose [67] is a manually annotated dataset with the body part annotations. We evaluate multi-person body part segmentation on its body part annotations. The dataset contains $26,151$ training images, and the *minival* has $1,508$ validation images.

3.5.2 Main results

We compare our technique with several state-of-the-art supervised approaches, including HAZN [206], Attention [29], LG-LSTM [106], LIP [59], Graph LSTM [105], DeepLab [28, 27], and WSHP [48]. Note that all these approaches use Pascal-Person-Parts dataset including the part segmentation labels as the training data while our network does not need to use any of the data from Pascal-Person-Parts at all. Following the settings of Pascal-Person-Parts [31],

Figure 3.5: Qualitative results of the proposed method *CDCL* on Pascal-Person-Parts and COCO validation images.

we predict 6 body parts and measure the prediction results using the mean Intersection of Union (mIOU) [46].

Table 3.1 shows the performance comparison with different state-of-the-art methods, and Figure 3.5 shows our prediction results. Without the segmentation training data provided by Pascal-Person-Parts, the proposed method *CDCL* achieves 65.02% mIOU, which is comparable to or better than several state-of-the-art supervised approaches, such as DeepLab v2 [28] and Graph LSTM [105].

We further compare our method with the state-of-the-art approach [48] on COCO-DensePose. For a fair comparison, we follow the body part settings of WSHP [48], and measure mIOU for the 6 different body parts and background. As shown on the second row *CDCL* of Table 3.2, our result is slightly better than WSHP [48] which used real segmentation training data from both Pascal-Person-Parts and AIC [1].

Method	Real Seg. GT	Syn Seg. GT	Head	Torso	U-arms	L-arms	U-legs	L-legs	Bkg	Avg
DeepLab-LFOV [27]	✓	✗	78.09	54.02	37.29	36.85	33.73	29.61	92.85	51.78
HAZN [206]	✓	✗	80.79	59.11	43.05	42.76	38.99	34.46	93.59	56.11
Attention [29]	✓	✗	81.47	59.06	44.15	42.50	38.28	35.62	93.65	56.39
LG-LSTM [106]	✓	✗	82.72	60.99	45.40	47.76	42.33	37.96	88.63	57.97
LIP [59]	✓	✗	83.26	62.40	47.80	45.58	42.32	39.48	94.68	59.36
Graph LSTM [105]	✓	✗	82.69	62.68	46.88	47.71	45.66	40.93	94.59	60.16
DeepLab v2 [28]	✓	✗	-	-	-	-	-	-	-	64.94
WSHP [48]	✓+	✗	87.15	72.28	57.07	56.21	52.43	50.36	97.72	67.60
CDCL	✗	✓	**75.53**	**66.26**	**63.28**	**57.14**	**47.75**	**51.45**	**93.72**	**65.02**
CDCL+Pascal	✓	✓	**86.39**	**74.70**	**68.32**	**65.98**	**59.86**	**58.70**	**95.79**	**72.82**

Table 3.1: Performance comparison of human body part segmentation (mIOU, %) on Pascal-Person-Parts dataset [31]. Note that WSHP [48] used an additional real dataset with human-annotated segmentation labels.

Method	Real Seg. GT	Syn Seg. GT	Head	Torso	U-arms	L-arms	U-legs	L-legs	Bkg	Avg
WSHP [48]	✓+	✗	67.33	62.22	51.50	55.66	54.22	53.11	76.81	60.12
CDCL	✗	✓	68.45	66.21	59.96	51.72	50.71	50.57	75.55	60.45
CDCL+Pascal	✓	✓	66.16	64.80	60.33	61.19	55.97	54.96	92.03	65.06
CDCL+COCO	✓	✓	**73.15**	**68.74**	**63.79**	**67.66**	**63.39**	**60.62**	**93.55**	**70.13**

Table 3.2: Performance comparison of human body part segmentation (mIOU, %) on COCO-DensePose human body masks [67]. Note that WSHP [48] used additional real dataset with human-annotated segmentation labels.

Method	Pascal-Person-Parts	COCO-DensePose
SYN	10.18	10.12
ADV	16.42	19.24
CDCL	**65.02**	**60.45**

Table 3.3: Performance comparison of human body part segmentation (mIOU, %) of different methods.

3.5.3 Adding real data with part segmentation labels

To obtain the performance *upper bound* of our technique, we evaluate our method when real data with part segmentation labels are used during training. The bottom row *CDCL+Pascal* of Table 3.1 shows the result on Pascal-Person-Parts where Pascal-Person-Parts training data is used. Our method outperforms WSHP by a large margin.

The same model is evaluated on the COCO-DensePose test data and the result is shown on the third row *CDCL+Pascal* of Table 3.2. Again it outperforms WSHP by a large margin.

If we use COCO-DensePose training data instead, and evaluate on COCO-DensePose test data, we obtain an additional gain and the result is shown on the fourth row *CDCL+COCO* of Table 3.2.

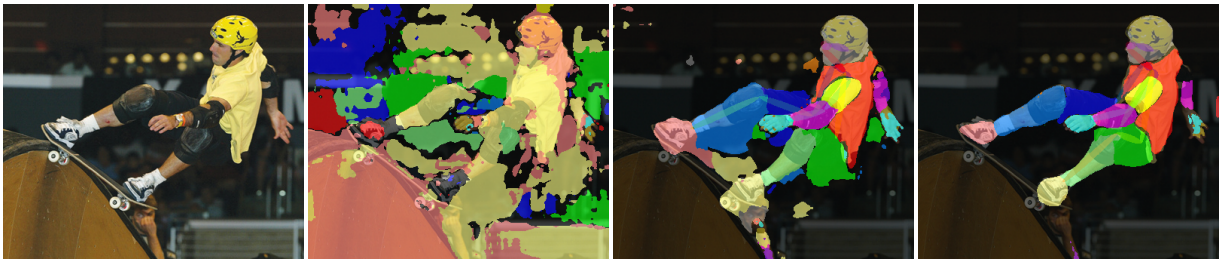
3.5.4 Comparison with adversarial learning

Recent studies [157, 187] used adversarial training to align the feature spaces of the synthetic and real images. Thus, we compare the performance of our method with the adversarial training strategy. Since the model presented in [157] cannot be directly used for part segmentation, we implemented our own network similar to [157]. Our network has a backbone (ResNet101) and two head networks, one for the part segmentation head and the other for the discriminator.

Table 3.3 shows the performance comparison on two datasets. We can see that adversarial

Method	Syn. Parts	Syn. Poses	Real Poses	Pascal mIOU	COCO mIOU
SYN	✓	✓	✗	10.18	10.12
NO-SP	✓	✗	✓	49.71	50.66
CDCL	✓	✓	✓	**65.02**	**60.45**

Table 3.4: Ablations of training with different types of data.



Input image SYN NO-SP CDCL

Figure 3.6: Qualitative comparison of the proposed method with different training strategies.

training (ADV) achieves better performance than that of training with synthetic data only without adversarial training (SYN), but it does not perform as well as our complementary learning technique.

3.5.5 Ablation study

Synthetic pose labels

Since our approach uses both synthetic poses and real poses, one interesting question is whether the synthetic pose is useful. To answer this question, we have trained our network without the synthetic poses (i.e. with synthetic parts and real poses). This configuration is denoted as *NO-SP*, and the results on Pascal-Person-Parts and COCO-DensePose are shown in Table 3.4. For completeness, we also show the results of SYN (synthetic parts + synthetic

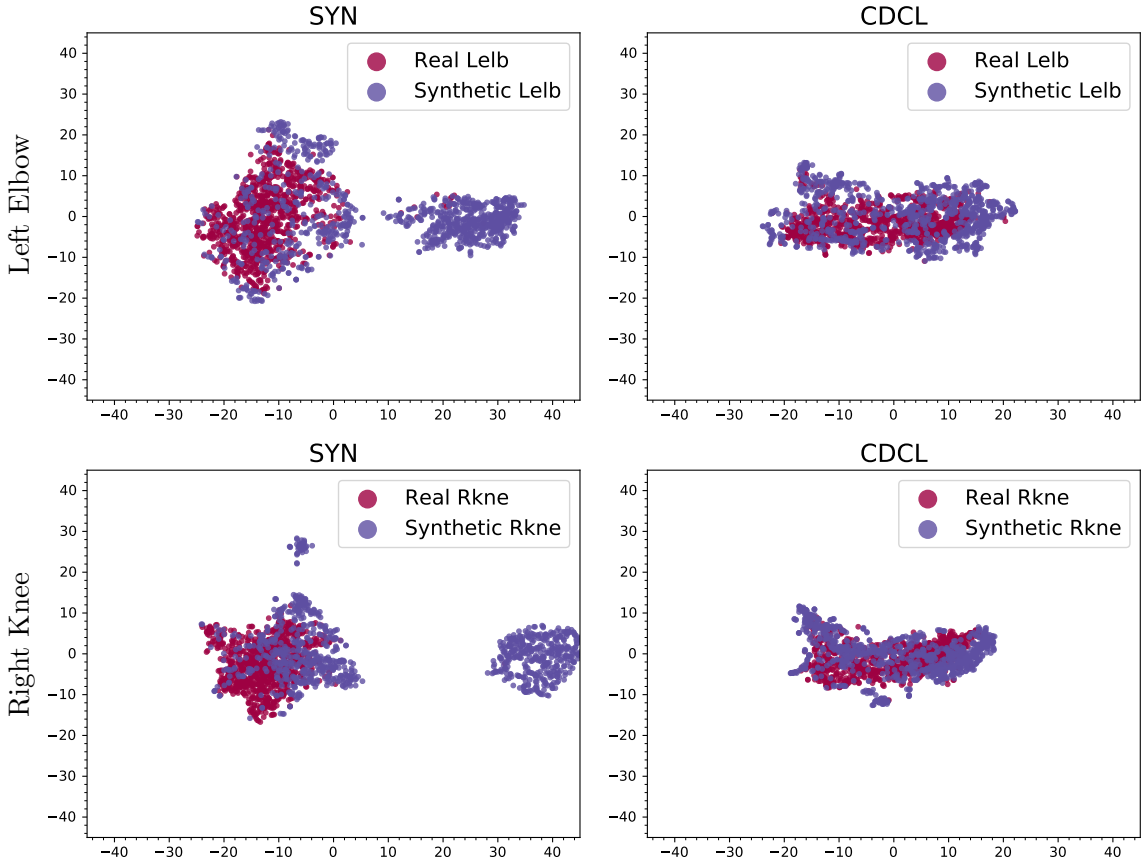


Figure 3.7: t-SNE visualization [120] of the feature spaces of the real and synthetic body parts.

poses), and *CDCL* (synthetic parts + synthetic poses + real poses). We can see that *NO-SP* outperforms *SYN* by a large margin thanks to the knowledge learned from the real data, and adding synthetic poses further boosts the performance. Figure 3.6 shows a qualitative comparison of the three configurations. *SYN* has trouble handling the background, *NO-SP* performs much better, and *CDCL* further improves upon *NO-SP*.

Feature space visualization

We visualize the features of two different models (*SYN* and *CDCL*) from the real and synthetic images using the t-SNE visualization technique [120]. In Figure 3.7, the left column

Original No Background Gray-scale Binary Mask

Figure 3.8: Different configurations of the synthetic data.

shows the features extracted with the model *SYN* (trained with synthetic data only), and the right column are from the model *CDCL*. The first row shows the features extracted at the left elbow position, and the second row shows the features extracted at the right knee position. In each plot, the red dots indicate the real data while the purple dots indicate the synthetic data. We can see that the red and purple dots in the right column are aligned very well, but they do not align well in the left column. This indicates that our complementary learning technique is effective at aligning the feature space of the real data with that of the synthetic data.

Synthetic training data analysis

Since our method learns part segmentation from synthetic data, one may wonder what elements of the synthetic data are essential to be rendered. To answer the question, we ablate our synthetic training data by gradually removing the background, colors, and the human texture, and train our model with these configurations, respectively.

Figure 3.8 shows the examples of different configurations of the synthetic training data, and Table 3.5 shows the performance comparison. Firstly, we observe that removing the background from the synthetic data causes only a small drop on the segmentation performance. This is an indication that our framework is learning the background from the real data. Secondly, after we further remove the color of the synthetic data (Gray-scale), we again

	Original	No Background	Gray-scale	Binary Mask
mIOU	65.02	63.96	62.78	43.38

Table 3.5: Performance comparison of our method on Pascal-Person-Parts using different synthetic training data.

only see a small drop on the performance. Finally, when we degrade our synthetic data to the extreme by just using binary masks, our framework still works reasonably well. These studies indicate that our framework mainly requires the pose variations in the foreground data and the rendering quality is not as critical compared to the conventional approach of directly training from synthetic data.

3.5.6 Novel keypoint detection

Since our approach can easily create arbitrary annotations on synthetic data and transfer the knowledge to real domain, our method is highly scalable and flexible to users needs. For example, suppose we want to predict a new set of keypoints including hands and feet, it would be difficult to re-label the entire COCO dataset. With our technique, we can simply generate new labels on the synthetic data. We have performed an experiment to demonstrate this capability.

We create 30 novel keypoints for each avatar in the graphics simulator, and use the proposed method to learn the new set of keypoints. Figure 3.10 shows the definition of the novel keypoints. To enable our existing network to learn such a new task, we add two additional head networks in our framework to learn the newly created 30 keypoints and their Part Affinity Fields, resulting in a total of 5 head networks in our network architecture.

Figure 3.9 shows the qualitative results of our novel keypoint detection. With small modifications of the existing network, our method learns the novel skeleton representations from the synthetic data and transfers the knowledge to the real domain. It eliminates the

Figure 3.9: Novel keypoint detection results on COCO validation images. Without any human labeling effort, our method learns to predict a new set of keypoints including those on the hands and feet.

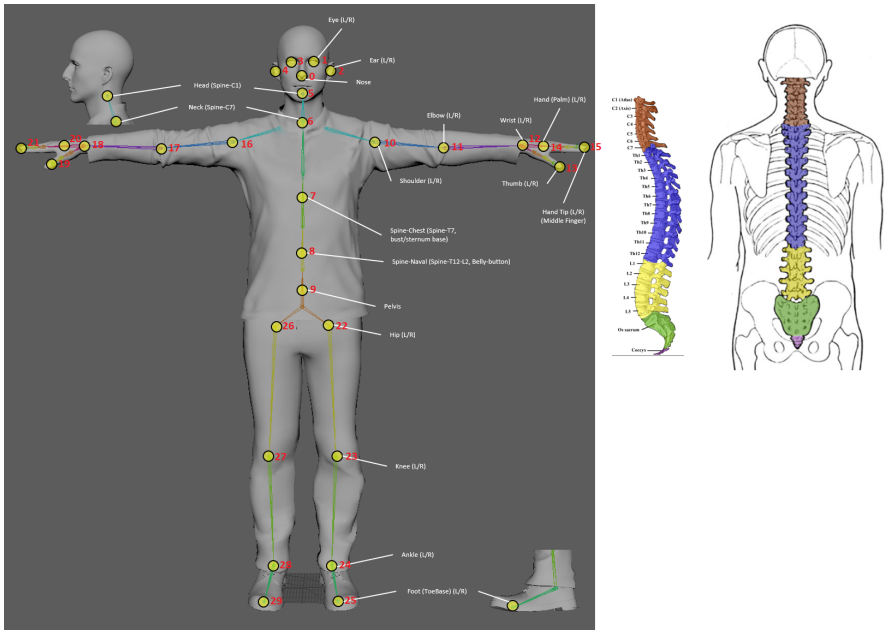
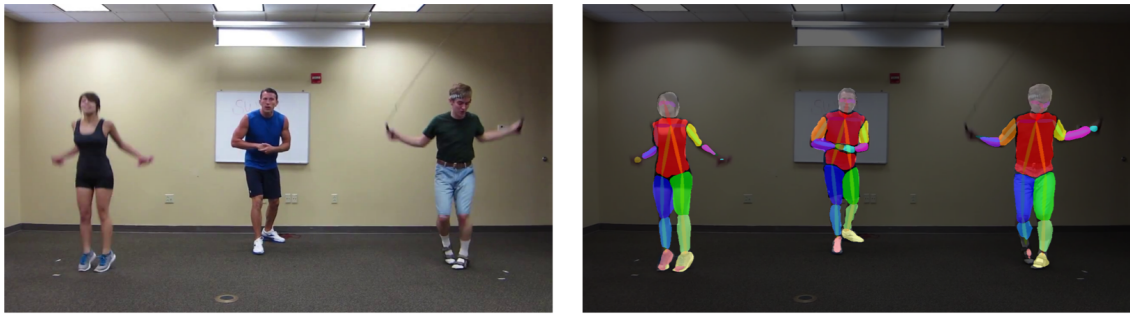
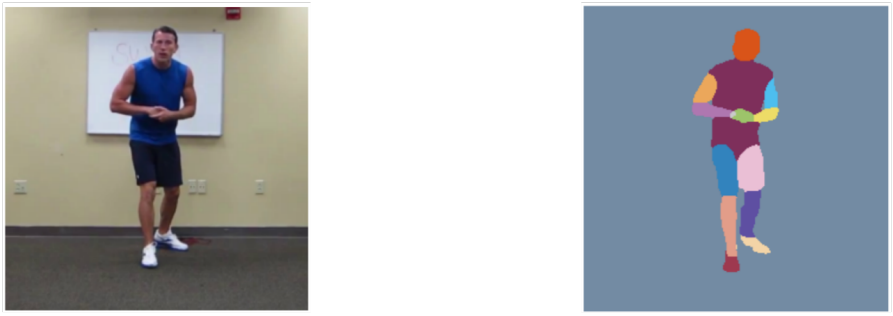


Figure 3.10: The definition of our created novel keypoints. There is a total of 30 keypoints and 29 part associations for constructing fine-grained human skeleton.



Original Test Image | Our Result



Center-cropped Test Image [191] | Result from [191]

Figure 3.11: There are three people in the image. Our method successfully detected multi-person body parts in the image. Previous method required additional preprocessing, and only detected single person body parts.

needs of ground truth labeling of the additional joints on the real data.

3.6 Conclusion

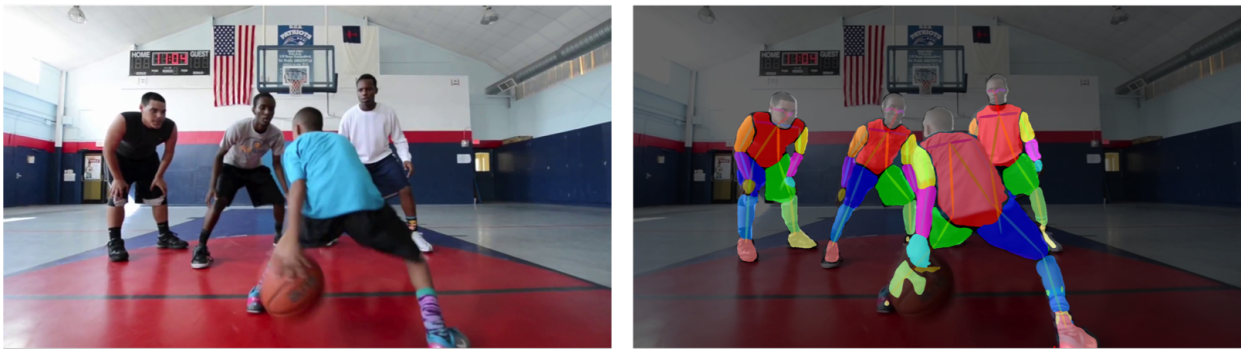
We presented a cross-domain complementary learning framework for multi-person part segmentation. Without using any real data part segmentation labels, our method is able to achieve a comparable or better performance than several state-of-the-art techniques that use real part segmentation data for training. We further demonstrated that our technique can also be used to learn novel keypoint detection from synthetic data.

3.7 Supplementary Material

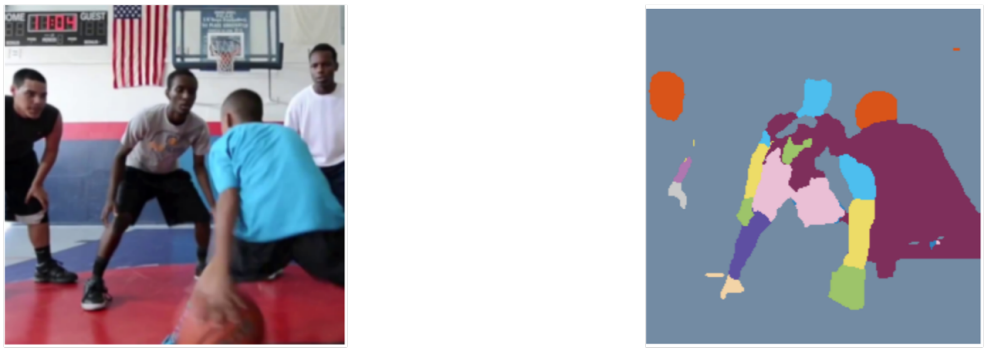
We have presented the cross-domain complementary learning for multi-person part segmentation. Without using any real part segmentation labels for training, the proposed method achieves comparable or better performance than the supervised state-of-the-art approaches on Pascal-Person-Parts and COCO-DensePose datasets. Our method outperforms the supervised state-of-the-art methods by a large margin when real part segmentation labels are also available for training. In this supplementary material, we further provide qualitative comparisons with the recent study [191] on MPII dataset.

Qualitative comparisons on MPII

Recent study [191] proposed to estimate body part segmentation by learning with synthetic data, which is closely related to our method. Since MPII dataset [9] does not have part segmentation labels for quantitative evaluation, [191] showed qualitative results on selected images from MPII. Given a test image, [191] used additional preprocessing to normalize the input. From their results on MPII dataset with multiple people, it appears that they cropped each image centered at a specific person before feeding to their network. In contrast, our method does not require such preprocessing. Furthermore, our method produces better results as shown in Figure 3.11, 3.12, 3.13 and 3.14. For each example, we show the original image from MPII dataset [9], our part segmentation result on the original image, the cropped version which was used as the network input in [191], and the part segmentation result of [191]. It is worth noting that our model does not use any real part segmentation labels for training in this experiment.



Original Test Image | Our Result



Center-cropped Test Image [191] | Result from [191]

Figure 3.12: There are four people playing basketball in the image. Previous method may not work well for multi-person scenario. In contrast, our method successfully detected the body parts of the four people in this image.

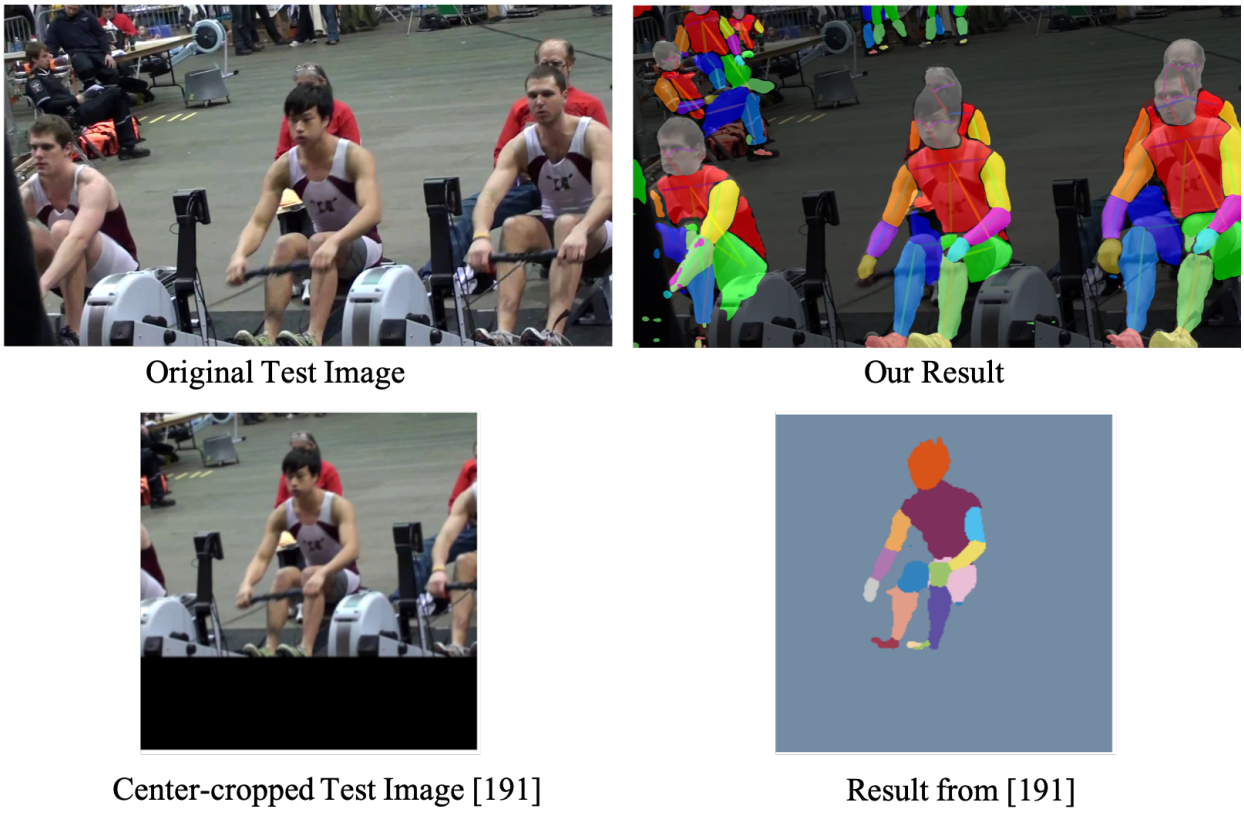


Figure 3.13: There are three people wearing white shirts in the image. In addition, there are two people wearing red clothing behind the white-shirt people. There are also many small people in the background. Our method generated correct part segmentation for all the people in the image even though some of them are heavily occluded by others. In contrast, previous method only detected a single person in the image.

Original Test Image

Our Result

Center-cropped Test Image [191]

Result from [191]

Figure 3.14: There are two people fencing in the image. Our method detected the body parts of both people in the image.

Chapter 4

FAST MULTI-PERSON POSE ESTIMATION USING A CONCATENATED PYRAMID NETWORK

4.1 Introduction

2D Human pose estimation is a fundamental challenge for computer vision. In practice, recognizing the pose of multiple persons in the wild is a lot more challenging than recognizing the pose of a single person in an image [11, 12, 180, 33]. A common approach [138, 35, 131] is to employ a person detector and perform single-person pose estimation for each detection. These top-down approaches directly leverage existing techniques for person detection and single-person pose estimation [180, 33, 138, 35, 131, 69, 49, 74] and have achieved the best accuracy on the COCO keypoint challenge [35, 69]. However, the runtime of these top-down approaches is proportional to the number of people: for each detection, a single-person pose estimator is run, and the more people there are, the greater the computational cost. In contrast, bottom-up approaches are attractive as they have the potential to decouple runtime complexity from the number of people in the image.

Unlike single-person pose estimation in top-down approaches, directly inferring the poses of multiple people in an image presents a lot more challenges. First, each image may contain an unknown number of people that can occur at any position or scale. Second, interactions between people induce complex spatial interference, due to contact, occlusion, and limb articulations, making association of parts difficult. These challenges require much more powerful network and sophisticated learning approach.

OpenPose [23] is the most representative bottom-up approach in multi-person pose estimation. It presented the first bottom-up representation of association scores via Part Affinity Fields (PAFs). It demonstrated that by simultaneously inferring the bottom-up representa-

tions of parts and their associations they can encode the global context sufficiently well to allow a greedy parse to achieve high-quality results. More importantly, OpenPose achieves near real-time runtime performance on frame by frame pose estimation without tracking. However, to achieve near real-time performance, OpenPose has to sacrifice its best precision achieved in multi-scale (which is 4x slower), and instead uses a single scale with much lower precision. The main problem that limits their runtime performance is that their network includes multi-stage sequential refinement and is sensitive to scale variation, thus highly relying on multi-scale detection to uphold its precision.

Primarily, OpenPose is a sequential prediction framework, where the detection confidence maps computed by the previous stage are used as the input of the next stage. By performing the multi-stage prediction, OpenPose iteratively refines the detection confidence maps, and improves the performance of the pose estimation. However, the multi-stage network increases the computational cost and the network parameters. Specifically, in their network design, OpenPose requires 6 stages of successive predictions for quality detection. Thus, the multi-stage prediction is one of the critical bottlenecks for fast detection. In addition, since OpenPose imposes intermediate supervision at each stage, the training cost of their method is proportional to the number of stages. On the other hand, to achieve higher accuracy, OpenPose sacristies their speed and performs multi-scale detection to achieve the state-of-the-art performance on the COCO 2016 keypoint challenge. Multi-scale detection is the scheme that conducts detection on resized images, which is equivalent to the detection at different scales [50, 16]. However, multi-scale detection introduces additional overhead. More specifically, during testing, one may need to rescale the image multiple times, and recompute the prediction multiple times.

In this chapter, we propose a novel Concatenated Pyramid Network (CCPN) as the backbone network to learn feature map representations. The proposed concatenated pyramid network learns different levels of representations using concatenation in a deep pyramid network. Different from the lateral connection in the Feature Pyramid Network [112], our network leverages both low-level and high-level semantics using concatenation for learning

multi-scale feature representations so that it improves the robustness to scale variations of input instances. Also, as the feature map is largely enhanced, the head network for prediction is much relieved such that a single stage network can achieve quality detection. Thus, the newly proposed network, with just a single-stage and single-scale, achieves comparable or even higher accuracy than the multi-stage OpenPose in multi-scale. Experimental results clearly demonstrate that our proposed network improves both speed and detection quality over OpenPose.

The main contributions of this chapter are summarized as below.

- We present an efficient Concatenated Pyramid Network (CCPN) for accelerating multi-person pose estimation.

- Our network eliminates the need for the multi-stage and multi-scale detection.

- Our network is faster than the state-of-the-art OpenPose while achieving higher accuracy than OpenPose.

4.2 Related Works

Human pose estimation [11, 21, 12, 151, 145, 152, 216, 24, 182, 58, 79] is a fundamental research problem of localizing and associating the human body parts in the images. Classical approaches [11, 12, 180, 33] address human pose estimation by using pictorial structures or graphical models. More recently, Convolutional Neural Networks (CNN) have been widely applied, and achieved large improvements on pose estimation [35, 69, 138, 131, 201, 21, 182, 146]. Furthermore, multi-person pose estimation becomes increasingly important due to the demand for practical applications. Previous works can be divided into top-down and bottom-up approaches.

Top-down approaches.

Top-down approaches [69, 49, 35, 138, 131, 74, 180] employ a person detector to firstly localize the possible regions in the image, and then reduce the problem to single-person pose estimation. Given the detected bounding box, previous methods apply different models to detect the body parts. Papandreou *et al.* [138] detect body part locations by predicting the heatmaps and the offsets of the keypoints. MaskRCNN [69] predicts human body parts using a fully convolution network head on the top of a backbone architecture. Cascaded Pyramid Network [35] detects human body parts using deep pyramid network and online hard keypoints mining. It is worth noting that the top-down approaches are based on the assumption that there is only one person appeared in the bounding box. With this strong prior knowledge, top-down approaches simplify the task of multi-person pose estimation, and make the learning process more efficient. That is one of the reasons top-down approach achieves better performance than other approaches. However, top-down approaches are highly dependent on the person detection. If the person detection fails, there will be no resource to recover in the subsequent stages. Secondly, top-down approaches are computationally expensive for multi-person pose estimation. The computational cost of top-down approaches is proportional to the number of bounding boxes detected by the person detector.

Bottom-up approach.

Bottom-up approaches [130, 146, 77] firstly detect human body parts, and then associate the parts to obtain the full skeletons. Bottom-up approaches are attractive as they do not require person detector and bounding boxes. Despite several bottom-up methods have been proposed, previous works may not be efficient due the costly parsing until the recent breakthrough of OpenPose [23]. OpenPose demonstrates its impressive performance on multi-person pose estimation, and speeds up the parsing via Part Affinity Fields (PAFs). They proposed to learn the part-to-part association and the orientation using convolutional neural networks. Particularly, PAFs are learned from the visual appearance, and are highly

efficient for guiding the part association during parsing. However, it requires multi-stage refinement to obtain quality results. To be specific, OpenPose performs 6 stages of successive predictions for refining the pose estimation. The multi-stage refinement increases the network complexity and slows down the inference time. Moreover, to retain the precision, OpenPose sacristies its speed and conducts multi-scale detection to achieve the state-of-the-art performance. OpenPose firstly resizes the image to 4 different scales. Then, it takes each image as the network input, and average the detection confidence maps and PAFs for subsequent greedy parsing. Such multi-scale detection introduces large computational costs. How to eliminate multi-stage and multi-scale detection without affecting the detection is the motivation of this work.

4.3 Proposed Appraoch

4.3.1 Overview

Figure 4.1 illustrates the overview of the proposed approach. Specifically, the proposed approach consists of the Concatenated Pyramid Network (CCPN) and the Skeleton Grouping Network (SGN) for rapid multi-person pose estimation.

Different from previous works that need multi-stage refinement [23, 131, 35, 201], the proposed network detects human skeletons in a single stage. Our network learns effective representations using the deep pyramid network architecture. Our pyramid network leverages the low-level and high-level semantics with the concatenation. Secondly, the proposed SGN learns the skeleton structure and the spatial relationship between the body parts for refining the segmented skeletons. Given the imperfect parsing results, the proposed SGN is able to group and complement the segmented skeletons. We elaborate the details of the proposed networks as below.

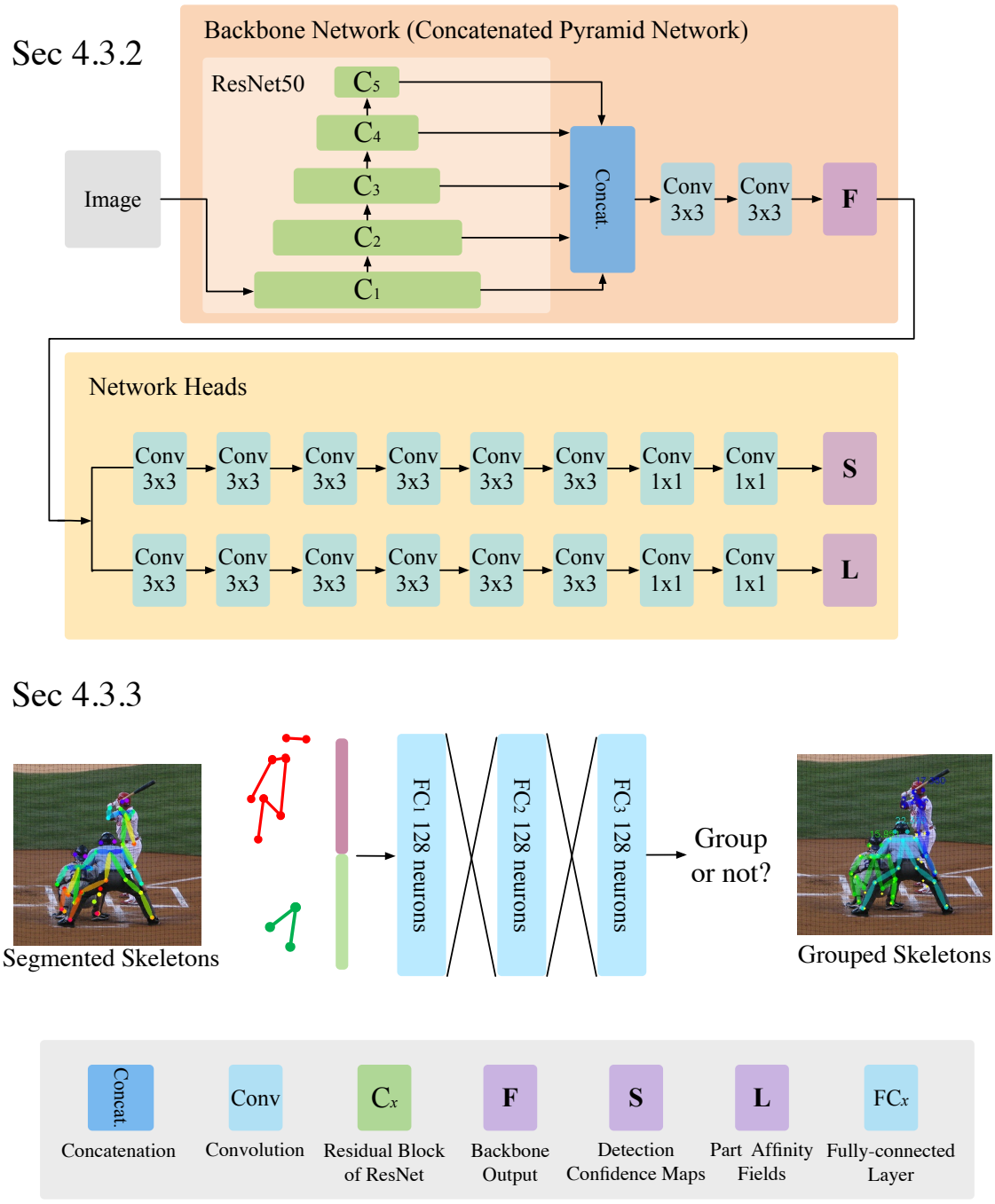


Figure 4.1: An overview of the proposed Concatenated Pyramid Network (CCPN) and Skeleton Grouping Network (SGN). In Sec 4.3.2, we present the proposed Concatenated Pyramid Network (CCPN), which learns rich representations and multi-scale information in a single framework for rapid multi-person pose estimation. In Sec 4.3.3, we desgin an efficient neural network for refining the segmented skeletons in the occlusion case.

4.3.2 Concatenated Pyramid Network

Previous studies [112, 35] show that Feature Pyramid Network (FPN) and its variants are effective for learning representations for visual recognition tasks. However, previous pyramid networks are designed mainly for region-based object detectors, such as MaskRCNN [69] using Region Proposal Network (RPN) [156]. Recently, the pyramid network has also been applied to multi-person pose estimation [35] and achieved the state-of-the-art performance on COCO 2017 keypoint challenge. However, their method belongs to top-down approaches, which restrict the input of the network to the person bounding box or region proposals. These top-down approaches greatly increase the computational costs due to the large number of person bounding boxes. Therefore, previous pyramid-based pose estimator [35] performs slower than OpenPose.

In this paper, we propose an efficient pyramid network to learn rich representations for detecting the human parts and their associations. As shown in Figure 4.1, the proposed network takes an image in arbitrary size as input, and outputs a set of detection confidence maps **S** and a set of part affinity fields **L** for part localizations and associations, respectively. For clarity, we describe our network in two parts: *backbone* and *head* architectures. The backbone network is for extracting useful feature maps from the input image, and the head network is for predicting the target outputs, which are **S** and **L** in our network, from the feature maps.

Backbone architecture. As discussed in DenseNet [73], creating short path from early layers to later layers with element-wise summation may not maximize the information flows between the layers. FPN learns multi-scale representations with the lateral connection, where the semantic information at different scales are directly accumulated. The accumulated feature representations are less discriminative since it losses the property of the representations at different levels. Instead of using the lateral connection, we propose to concatenate the intermediate layers in a pyramid network to preserve the information flows between different scales and different levels of semantics. We normalize the size of different feature maps by

adopting the deconvolution operations, where the deconvolution kernels can be optimized during learning. Then, the upsampled feature maps are concatenated along the channel axis. Finally, we feed the concatenated feature map into the subsequent convolutional layers, and distill the multi-scale feature representations $\mathbf{F}$. In contrast to applying the element-wise summation, concatenating the multi-scale features simultaneously preserves the semantic information at different scales and enriches the variation of the input of the subsequent head network. This also improves the learning efficiency. This is the major difference between our network and the previous feature pyramid networks [35, 112]. Comparing to FPN, our training is more efficient than previous pyramid networks since each layer in our network can directly receive the gradients back-propagated from our head networks.

Our approach is independent of the choice of the backbone convolutional networks, and in this paper we present our results on ResNet50 [70]. Particularly, we denote the outputs of the intermediate layers in ResNet50 as $\{C1, C2, C3, C4, C5\}$ for $conv1$, $conv2$, $conv3$, $conv4$, and $conv5$. We normalize the size of the feature maps $\{C1 - C5\}$ to a fixed size as $\{C_n1 - C_n5\}$ via deconvolution. We then concatenate the feature maps $\{C_n1 - C_n5\}$ as the input of the subsequent convolution layers. We denote the output of the subsequent convolution layers as $\mathbf{F}$.

Head architecture. Following [23], we train our network to learn human part localization and association. We attach two network heads on the top of our backbone network. Each network head is a fully convolution network consisting of 8 convolution layers. We denote two network heads as ρ and ϕ, respectively. The detection confidence maps $\mathbf{S}$ are computed by $\mathbf{S} = \rho(\mathbf{F})$, and the Part Affinity Fields [23] $\mathbf{L}$ are computed by $\mathbf{L} = \phi(\mathbf{F})$.

Training. The proposed network is trained in the end-to-end fashion. We initialize our backbone network using the parameters pre-trained on ILSVRC12 [164], while the other parameters are randomly initialized. The learning rate for every layer in our network is 0.0002. Our loss function is similar to that of OpenPose [23]. The difference is that we train our network for only a single stage, but OpenPose is trained for multiple stages. Note

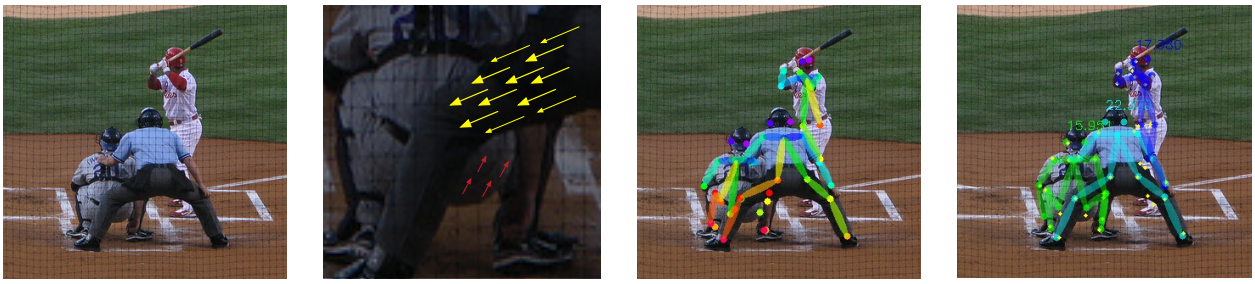


Figure 4.2: The Part Affinity Fields (PAFs) are mainly dependent on visual appearance, so that the part affinity score tends to be lower if the visual evidence is insufficient. In this example, previous methods may not correctly infer the part affinity of the catcher's lower body due to the heavily occlusion resulting in fragmented skeletons. In contrast, our approach is able to group the segments and construct the correct skeletons. Yellow arrows indicate strong affinity fields, and red arrows are relatively weak affinity fields. See Section 4.3.3 for details.

that our network could be extended to multi-stage network for higher accuracy by cascading additional network heads, and in this paper we focus on single-stage network for accelerating multi-person pose estimation.

4.3.3 Skeleton Grouping Network

The Part Affinity Fields (PAFs) [23] has been proven efficient for parts assemble. However, a successful detection of PAFs is mainly relying on the clear visual appearance. As shown in Figure 4.2, previous bottom-up approaches tend to produce segmented skeletons if some of the body parts are missing, invisible, or occluded by other objects. Previous studies [146, 77] proposed to associate the detected parts using sophisticated algorithms. But their average running time for each image is roughly 500 seconds [77]. Therefore, previous methods may not meet the requirement of real-time pose estimation.

In this paper, we propose a simple yet effective neural network for learning to recover the segmented skeletons by finding their missing parts. As can be seen in Figure 4.2, given

the lower body skeleton and the other detected skeletons, we would like to determine which upper-body skeleton best fits the lower body skeleton. To this end, we formulate this as a pairwise matching problem. We denote S as the skeleton, and denote P as the location of the body part, respectively. In addition, there are in a total of j body parts for each skeleton, so that $S = \{P_1, P_2, \ldots, P_j\}$. Note that $P_i = 0$ if the i body part is missing. Our network takes a pair of skeletons $\{S_p, S_q\}$ as the input, and outputs the confidence score indicating how possible that $\{S_p, S_q\}$ belongs to the same person. Our network consists of three fully-connected layers followed by the binary classification layer. Each fully-connected layer has 128 neurons. Dropout operation is applied for each layer to stimulate the case of missing parts. To train our network, we generate the pairwise skeletons using COCO keypoint dataset. For generating the positive pairs, we take the groundtruth skeletons in COCO keypoint training set, and split each skeleton to a pair of segmented skeletons. To generate the negative pairs, we synthesize the fake skeleton by randomly merging two different skeletons appeared in an image. Then, we randomly split each fake skeleton to a pair of segmented ones to form a negative pair.

4.4 Experimental Results

In this section, we compare the performance of the proposed method with OpenPose on the COCO keypoint dataset [113]. Following the evaluation protocol in OpenPose [23], we train our network on COCO 2016 `train` set, and test on `val-1k` set. Note that `val-1k` is the validation subset provided by OpenPose[1]. Firstly, we study the effectiveness of different backbone architectures, and evaluate the head networks with different number of stages. Then, we compare the performance of different skeleton grouping algorithms. Finally, we present the comparative evaluation on COCO keypoint challenge.

---

[1] `val-1k` is publicly available at `https://github.com/CMU-Perceptual-Computing-Lab/caffe_rtpose/blob/master/image_info_val2014_1k.txt`

4.4.1 Backbone architectures

We compare the performance of different backbone networks including VGG19 [173], ResNet50 [70], FPN [112] as well as the proposed Concatenated Pyramid Network (CCPN). To evaluate the effectiveness of the backbone architectures, we fix all approaches to single-stage in this experiment. For VGG19 backbone, we follow Convolutional Pose Machines (CPM) [201] and use the first 10 layers as the backbone network. Following [112, 23], we take the residual blocks $\{C_1 - C_5\}$ in ResNet50 as the backbone. Table 4.1 shows the performance comparison of different backbone networks using single-stage detection. Replacing VGG19 with ResNet50 improves the Average Precision (AP) from 31.0% to 42.6% AP. This indicates deep network structure is effective for learning part localization and association. Next, we study the performance of the pyramid network and its variant. We design a pyramid network that connects five residual blocks in ResNet50 individually to the subsequent network heads. We denote this approach as FPN+intermediate supervision. We notice 2.6% AP increase comparing to ResNet50. Moreover, we also evaluate FPN [112] with different residual blocks. We observe that learning with more residual blocks is helpful for improving AP. Since FPN leverages multi-scale representation using the lateral connection, it achieves higher AP than ResNet50. Particularity, the proposed CCPN achieves the highest AP score than the compared backbone architectures. The results show that concatenating the multi-scale representations preserves the semantic information at different levels, and learns more discriminative representations. The results also show that CCPN performs more favorably than FPN for learning multi-scale representations. It is worth noting that the our single-stage and single-scale method achieves 50.3% AP, which is higher than 31.0% AP of the single-stage and single-scale OpenPose (denoted as VGG19 in Table 4.1).

Furthermore, we analyze the details of the pyramid network designs. FPN [112] proposes to use the lateral connection and element-wise summation for leveraging the information at different levels. To make it feasible, nearest neighbor upsampling is applied to fix the size of feature maps for summation. However, this approach introduces potential drawbacks. First,

Table 4.1: Performance comparison (Average Precision, %) of different backbone networks using single-stage and single-scale detection.

Method	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}
VGG19 [173]	31.0	62.9	26.0	32.3	30.6
ResNet50 [70]	42.6	71.1	42.9	37.4	52.6
FPN + intermediate supervision	45.2	72.8	45.8	39.7	55.7
FPN (C3+C4+C5) [112]	48.3	74.9	50.0	43.3	57.8
FPN (C2+C3+C4+C5) [112]	48.4	75.1	49.6	43.3	58.0
FPN (C1+C2+C3+C4+C5) [112]	48.7	75.9	50.5	43.9	58.0
Ours, CCPN (C1+C2+C3+C4+C5)	**50.3**	**76.9**	**52.6**	**45.0**	**60.5**

nearest neighbor upsamping scales the feature maps in coarse resolution. Secondly, element-wise summation blends the feature maps and losses the property of the representations at different levels. In contrast, the proposed CCPN removes the lateral connection and directly concatenates the feature maps learned from different layers. We eliminate the nearest neighbor upsampling, and apply deconvolution for learning the optimal upsampling in the network. Table 4.2 shows the performance comparison of different network designs. It is clear that our CCPN performs more favorably than FPN in terms of AP. Deconvolution and concatenation are both effective for learning representations in a pyramid network. The results show that our CCPN is an effective backbone network, and performs more favorably against the previous backbone architectures.

4.4.2 Head networks

We analyze the network heads in the section. Previous approaches propose to cascade multiple heads on the top of the backbone networks, and perform sequential predictions for refining the detection results. It can be seen in Table 4.3, multi-stage refinement is

Table 4.2: Performance comparison (Average Precision, %) of different pyramid networks using single-stage and single-scale detection

Method	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}
NN unsampling + Lateral connections [112]	48.3	74.9	50.0	43.3	57.8
Deconv. + Lateral connections	48.7	75.6	50.3	43.7	58.2
NN unsampling + Concatenate	49.0	75.8	50.5	44.2	58.1
Deconv. + Concatenate (Ours, CCPN)	**50.3**	**76.9**	**52.6**	**45.0**	**60.5**

critical for OpenPose to improve its performance. By increasing the number of stages from the single stage to 6 stages, OpenPose gains an overall 16.7% AP increase. Though effective, multi-stage detection greatly increases the computational cost, and slows down its runtime performance. In contrast, the proposed single-stage network performs more favorably against multi-stage OpenPose in terms of both precision and speed. Our network removes the need for multi-stage refinement, and thus accelerates the detection. The results also point out that, our network learns more discriminative representations using the concatenated pyramid network architecture, so achieves higher precision than OpenPose.

4.4.3 Skeleton grouping

We further evaluate the performance of different skeleton grouping approaches in this section. After associating the body parts, OpenPose removes the skeletons if the detected body parts are less than 3. Thus, it may not detect a person who are occluded by others, and potentially increases the false negative errors. In Table 4.4, the proposed Skeleton Grouping Network (SGN) improves the original parsing approach by 1.6% AP. The results show that, when the segmented skeletons are available, our neural network is able to group the segmented skeletons and further reduce the false negative errors. It also shows that our network can learn the spatial relationship between body parts without image inputs. Note that we also

Table 4.3: Performance comparison (Average Precision, %) of different methods using multi-stage refinement.

Method	Multi-scale	Stages	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}
OpenPose [23]	✗	1	31.0	62.9	26.0	32.3	30.6
OpenPose [23]	✗	2	41.2	71.5	39.7	38.9	46.7
OpenPose [23]	✗	3	44.8	73.8	44.5	41.0	52.7
OpenPose [23]	✗	4	46.2	74.7	46.7	42.0	54.9
OpenPose [23]	✗	5	47.3	74.8	48.6	42.6	56.6
OpenPose [23]	✗	6	47.7	74.7	49.5	43.0	57.0
Ours, CCPN	✗	1	**50.3**	**76.9**	**52.6**	**45.0**	**60.5**

Table 4.4: Performance comparison (Average Precision, %) of different skeleton grouping approaches using single-stage and single-scale detection.

Method	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}
Greedy parsing	50.3	76.9	52.6	45.0	60.5
Greedy parsing + Skeleton Grouping Network	**51.9**	**78.7**	**54.3**	**46.5**	**62.5**

Table 4.5: Performance comparison (Average Precision, %) of different methods on COCO 2016 keypoint `val-1k` set.

Method	Multi-scale	Stages	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}
OpenPose [23]	✓	6	58.4	81.5	62.6	54.4	65.1
Ours	✓	1	62.3	83.4	67.0	59.2	69.2

train the Siamese network to learn skeleton matching, but we observe worse performance. Note that the segmented skeletons can be seen as sparse vectors. The sparse inputs are challenging for training the Siamese network.

4.4.4 Results on COCO keypoint 2016 and 2017

Following OpenPose's evaluation protocol that applies multi-scale detection with four different scales, Table 4.5 shows the performance comparison between OpenPose and our method on the COCO 2016 `val-1k` set. Note that `val-1k` is a subset of the COCO validation set, which consists of $1,160$ images that are randomly selected and provided by OpenPose. Our method achieves 62.3% AP, and performs more favorably against OpenPose while multi-scale detection is adopted. Moreover, Table 4.6 shows the performance comparison of different methods when training with data augmentation. Benefiting from data augmentation, our network achieves 60.5% and 64.7% AP using single-scale and multi-scale detection, respectively. It can be seen that our single-stage and single-scale network performs more favorably against multi-stage and multi-scale OpenPose.

Following COCO 2017 keypoint challenge, we train our network on COCO 2017 `train` and `val` set, and evaluate the performance on the `test-dev` set. Table 4.7 shows the performance comparison with different methods. Our method performs more favorably than OpenPose in terms of speed and accuracy.

Note that the best performance of OpenPose published in COCO keypoint leaderboard is

Table 4.6: Performance comparison (Average Precision, %) of different methods on COCO 2016 keypoint `val-1k` set using training data augmentation.

Method	Multi-scale	Stages	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}
OpenPose + aug [23]	✓	6	59.8	81.5	64.6	56.6	67.3
Ours + aug	✗	1	60.5	84.1	65.1	59.3	63.5
Ours + aug	✓	1	**66.6**	**86.3**	**72.0**	**65.0**	**70.5**

Table 4.7: Performance comparison (Average Precision, %) of different methods on COCO 2017 keypoint `test-dev` set.

Method	Strategy	External data	Single model	Multi-scale	AP	fps
OpenPose [23]	Bottom-up	✗	✓	✓	55.5	18.9
Ours	Bottom-up	✗	✓	✓	**59.6**	**33.1**

61.8% AP. However, this is achieved by using additional person detection model and single-person pose estimation algorithms (clarified by [69]). For a fair comparison, we compare only the performance of multi-person pose estimation in the experiments.

4.4.5 Runtime analysis

We discuss the computational cost of different methods in this section. We randomly select $1,000$ test images from COCO `val-1k` dataset, and compute the average computational time. The experiments are carried out on a machine with an Intel Xeon E5 CPU, and an NVIDIA Titan XP with CUDA-8.0. Following [23], we fix all testing images to a scale 368x654. Figure 4.3 shows the comparison between speed and detection quality. Our method achieves better performance than the state-of-the-art OpenPose in terms of both speed and accuracy. Note that our skeleton grouping network does not influence the overall computational cost since its cost is two orders of magnitude less than the forward-pass of our Concatenated

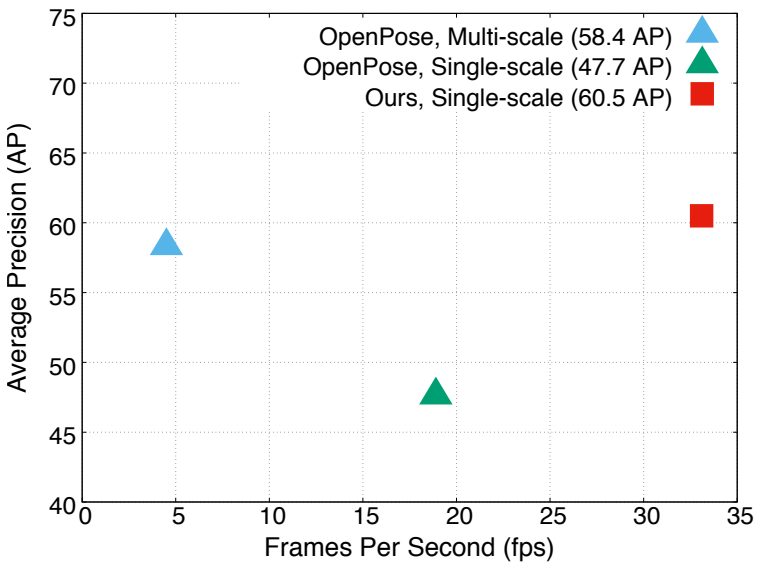


Figure 4.3: Average precision and speed on COCO 2016 keypoint `val-1k` dataset.

Pyramid Network.

4.5 Conclusion

We have presented a simple yet effective bottom-up approach for accelerating multi-person pose estimation. Experimental results demonstrate that our proposed network improves both speed and detection quality over the state-of-the-art OpenPose. Applying our method to other human analysis tasks such as human activity analysis can be an interesting future work.

Figure 4.4: Keypoint detection results on COCO `test-dev` using FastPose.

Chapter 5

NONPARAMETRIC HUMAN MESH CONSTRUCTION FROM A SINGLE IMAGE WITHOUT GROUND TRUTH MESH

5.1 Introduction

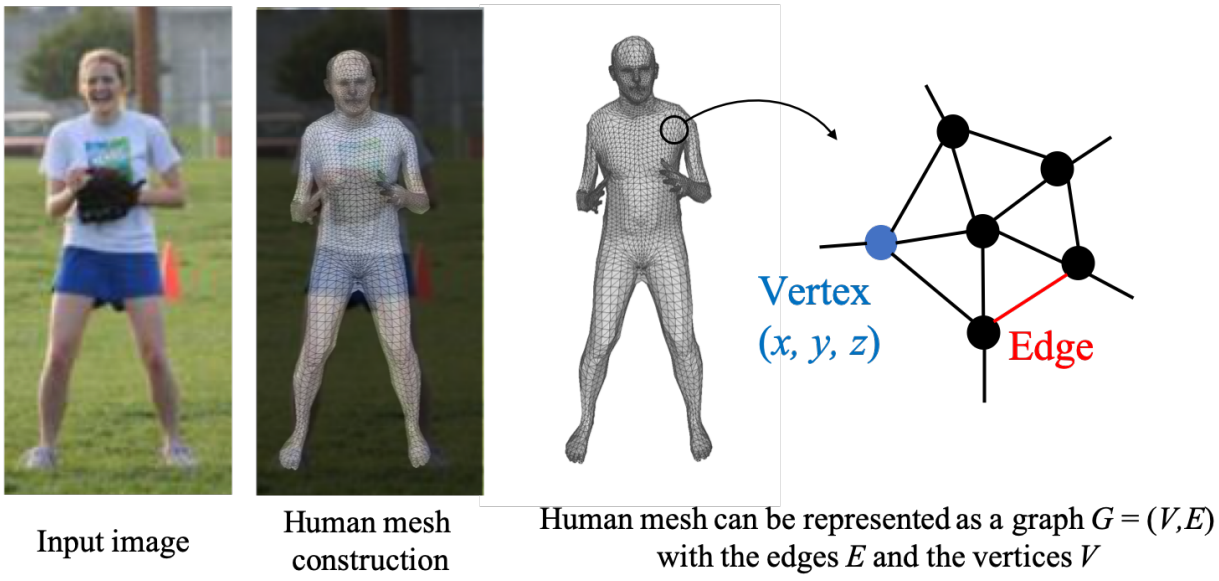


Figure 5.1: Illustration of the human mesh construction. Human mesh construction aims to estimate the 3D human mesh associated with the humans in the given input image.

A human in an image can be modeled by a 3D mesh. Let G denotes the 3D mesh of a generic human body, where G can be represented as a 3D graph $G = (V, E)$ with the edges E and the vertices V. As shown in Figure 5.1, 3D human mesh construction aims to estimate the 3D graph associated with the humans in the input image.

Estimating the 3D mesh of a human body is one of the fundamental challenges in computer vision. Human mesh construction [118, 203, 67, 78, 101, 141, 67, 95, 194, 91] has recently drawn an increasing attention as it plays an important role for a variety of applications

Input frame Our result Input frame Our result

Figure 5.2: Given a challenging video, our proposed approach constructs the human meshes for the two people. Our proposed method does not require expensive ground truth mesh labeling, and achieves comparable or better performance than the previous state-of-the-art methods which require ground truth mesh labels for training.

such as augmented reality, human-computer interaction, and activity analysis. While many studies have demonstrated effective 3D mesh construction using depth sensors [129, 172], inertial measurement units (IMUs) [228, 75, 194], and multiple cameras [126, 87, 140], people are exploring to use a monocular camera setting which is more convenient and efficient. However, it remains challenging to construct the human mesh from a single monocular image due to complex deformation of the human body, object occlusion, and limited 3D information.

Supervised training with deep convolutional neural networks has shown great progress on human mesh construction from a single image. However, many existing approaches [101, 141, 67, 95, 220] require 3D ground truth mesh labels for training. Since it is difficult and expensive to capture 3D ground truth meshes for a large variety of scenes, it is desirable to avoid the requirement on the ground truth meshes. To address the problem, recent studies [141, 89, 135, 66] propose to use a parametric human model such as skinned multi-person linear model (SMPL) [118] and regress the shape and pose coefficients of the model. However, parameter regression remains a very challenging task and it usually requires a large number of paired image-SMPL data for supervised training. Also, the parametric representation has limitations. Construction of the model like SMPL requires digitizing a

large number of people with different shapes and poses, and it is very time consuming and expensive. In practice, only a limited amount of shape and pose variations can be captured in a dataset. As a result, the resulting parameter space may not cover all the variations in the real world, which affects the performance of the algorithms.

In this chapter, we observe that 2D pose labels, 3D pose labels, and 2D part segmentation labels are readily available in many datasets [101, 78]. Using these labels we can train a neural network to estimate the part segmentation of the human body. By projecting the 3D mesh into 2D and minimizing the error between the projection and the part segmentation, we could learn the human mesh without a ground truth mesh. Also, for different body shapes, the internal vertex distribution could be modeled by a Gaussian Mixture Model (GMM). By adding a loss term using Laplacian prior [175, 127], we show that we could learn the internal vertex distribution effectively. Our technique learns human mesh construction without any human-annotated mesh labels and achieves performance comparable to several state-of-the-art approaches which require ground truth mesh labeling. On the other hand, if ground truth mesh labels are also available during training, our method outperforms the supervised state-of-the-art methods by a large margin.

Like the recent state-of-the-art approach [95], we represent human mesh in a form of *graph*, and use a graph convolutional neural network (Graph CNN) [93] to learn human mesh construction. Since we do not use ground truth meshes in training, we introduce two new terms in the loss function. The first term is the Laplacian prior that acts as a regularizer in the mesh construction to learn the correct internal vertex distribution for different body shapes. Laplacian prior has been used widely for geometric modeling and mesh editing [177, 175, 127], but we are the first to use it with Graph CNN to learn the mesh construction. The second term is the part segmentation loss that forces the projected region of the constructed mesh to match the part segmentation.

Since the existing datasets like UP-3D [101] and Human3.6M [78] do not contain many scenes with occlusions, the learned model usually does not handle occlusions very well. To address this problem, we propose to feed the 2D pose and part segmentation heatmaps to

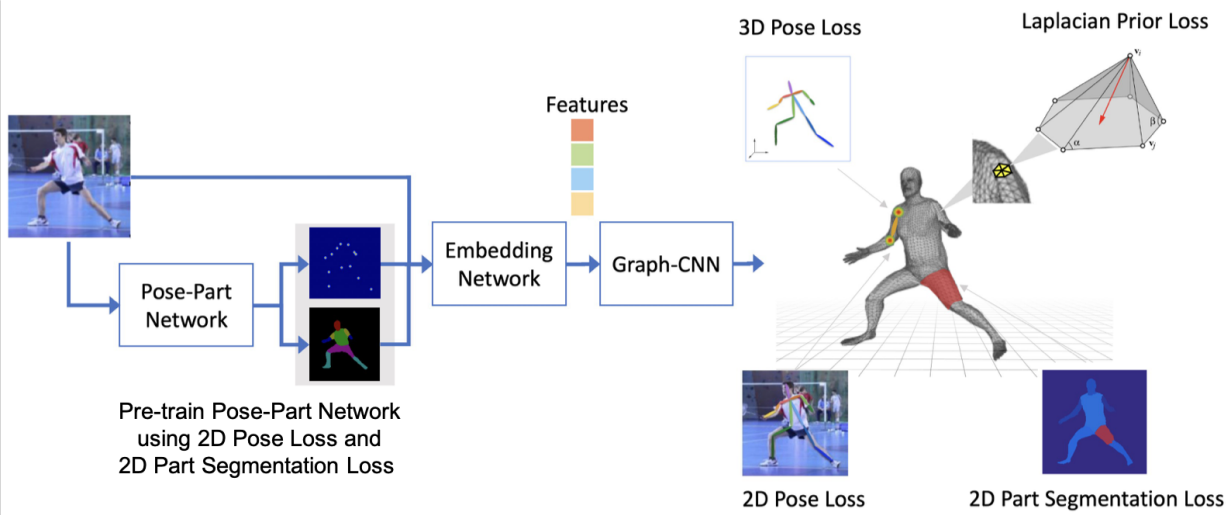


Figure 5.3: An overview of our mesh construction framework. It consists of three subnetworks: (1) Pose-Part Network that extracts the pose heatmaps and body part feature maps from the input image, (2) Feature Embedding Network that converts the feature maps to a feature embedding, and (3) Graph Convolutional Neural Network (Graph CNN) that takes as input the feature embedding and outputs the 3D coordinates of all the mesh vertices. We use four loss terms to optimize Graph CNN including Laplacian prior for regularizing the locations of the vertices, part segmentation loss for constructing correct body shape, 3D and 2D pose loss for optimizing the pose of the human mesh.

the feature embedding network. The 2D pose and part segmentation heatmaps are learned by leveraging the existing pose and part segmentation datasets like MSCOCO [113] and Pascal-Person-Parts [32] (both containing large number of images with occlusions). As a result, our method works much better in handling occlusions.

As shown in Figure 5.2, our proposed method constructs correct human meshes for the two people walking on the street in all the frames even though the two people have different pose interactions and variations.

The main contributions of this chapter include:

- We are the first to learn nonparametric body shape construction from a single image without mesh supervision.

- We introduce Laplacian prior and part segmentation loss terms into the Graph CNN

framework to learn the 3D human mesh model without the need of ground truth mesh labels.

- We show that by explicitly feeding 2D pose and part segmentation heatmaps into the feature embedding network, the robustness in occlusion scenarios can be significantly improved.

- Experimental results on multiple datasets show that our proposed method achieves comparable or better performance than the state-of-the-art methods which require ground truth meshes in training.

The remainder of the chapter is organized as follows. Section 5.2 provides a survey of the related works. Section 5.3 presents the proposed method. In Section 5.4, extensive experiments are conducted to validate the proposed approach. Finally, concluding remarks are given in Section 5.5.

5.2 Related Works

5.2.1 Parametric approach

Human mesh construction using parametric approaches has a long-standing history. Majority of the previous works adopt the SMPL parametric model [118], and propose to regress the shape and pose parameters. The regression can be done with the help of various 2D human body features such as human skeletons [101, 141], silhouettes [141], and body part segmentations [135]. Kanazawa *et al.* [89] proposed to integrate the differentiable SMPL model as a layer within a neural network, and estimate SMPL parameters from an input image using pose prior with an adversarial training framework. Tung *et al.* [188] proposed to learn SMPL parameters using a self-supervised strategy.

5.2.2 Nonparametric approach

Instead of using a parametric model, various works have been reported to directly estimate the body shape from an image including leveraging depth [172] or 2D-to-3D correspondences [67], and representing a 3D mesh into a volumetric space [190] or graph [95]. Specifically, Varol *et al.* [190] proposed to embed the 3D mesh into a volumetric space for learning human body shape. The volumetric representation is memory intensive resulting in a limited resolution. In addition, it requires ground truth meshes obtained from synthetic datasets. Güler *et al.* [67, 7] proposed to associate image pixels with part-based UV maps. However, manual label acquisition is very expensive, and model prediction does not explicitly provide semantic information of the 3D geometry. Kolotouros *et al.* [95] showed that the regression can be significantly easier than the conventional approaches by using graph convolutional neural networks (Graph CNNs), but it requires well-annotated ground truth meshes since its regression target for each vertex is its 3D ground truth location. Zhu *et al.* [232] proposed a multi-stage deformation refinement, and used depth information to find surface variations. But it needs to manually define the handles on the surface for controlling the mesh deformation. Natsume *et al.* [126] cast the problem as a multi-view silhouette-based construction, but rely heavily on multi-view segmentation synthesis. Saito *et al.* [165] proposed to learn a 3D occupancy field using depth information. Among the literature, the common theme of all these works is that they have focused on strongly supervised learning using labeled training data. However, the acquisition of large-scale 3D mesh labels, especially for human body shape, is very expensive. We propose to relieve the need for ground truth meshes by formulating a new learning objective function using Laplacian prior and part segmentation in a Graph CNN framework.

5.2.3 Graph convolutional neural networks for computer vision

While deep convolutional neural networks [102] are effective for extracting hidden patterns from data, there are many computer vision tasks where the data can be represented in a form

of graph [205]. By using the graph as the representation, it is shown to be more effective for high-level semantic analysis, such as scene graph generation [210, 214], image content generation [85], category-specific object modeling [200], 3D hand estimation [55, 223, 195], 3D face construction [153], human action recognition [80, 212], human-object interaction [149], semantic segmentation [150], and image classification [54]. Recently Graph CNN has been used [95, 115, 193] to estimate the 3D shape of a human body, however, these methods require the 3D ground truth locations for each vertex of the mesh as their regression target. These limitations have motivated us to develop a technique that does not require ground truth mesh supervision.

5.3 Proposed Method

Given a dataset D with 2D pose labels, 3D pose labels, and 2D part segmentation labels. Let $D = \{I^i, \bar{J}^i_{2D}, \bar{J}^i_{3D}, \bar{B}^i_{2D}\}^M_{i=1}$, where M is the total number of training images, $I \in R^{w \times h \times 3}$ denotes an image, $\bar{J}_{2D} \in R^{K \times 2}$ denotes the ground truth 2D coordinates of the joints and K is the number of joints on a person. Similarly, $\bar{J}_{3D} \in R^{K \times 3}$ denotes a 3D joint ground truth. $\bar{B}_{2D} \in R^{w \times h \times Z}$ is the body part segmentation ground truth and Z is the total number of body part categories.

Figure 5.3 is an overview of our proposed framework. Our proposed framework takes an image of size 224 x 224 as input, and predicts a set of mesh vertices Y. The proposed model consists of three subnetworks: *Pose-Part Network*, *Feature Embedding Network*, and *Graph CNN*. We first pre-train a Pose-Part Network for extracting the part segmentation and pose heatmap from a given input image. Then, we use a Feature Embedding Network to convert the input image and the extracted feature maps (i.e., the part segmentation and pose heatmap) to a feature embedding with a much lower dimension. Finally, we train a Graph Convolutional Neural Network (Graph CNN) which takes the feature embedding as input and output the vertex coordinates of the human mesh. We use four loss terms: Laplacian prior, 3D pose loss, 2D pose loss, and part segmentation loss to train the Graph CNN. The detail is described as follows.

5.3.1 Pose-Part Network

In the first part of our model, we train a Pose-Part Network similar to the multi-task networks [69, 111] to extract human-related feature maps from the input image. The human-related feature maps include the human pose heatmaps and the body part feature maps. These feature maps describe the high-level semantics including the human pose variations and the human shape information, which are very helpful for Graph CNN to learn human mesh construction. We denote P as our Pose-Part Network, and its outputs are $H_2D, B_2D = M(I)$, where I is an input image, H_{2D} denotes the pose estimation heatmaps, and B_{2D} denotes the body part feature maps. To train the Pose-Part Network, we minimize the differences between the prediction and the ground truth by using the Mean Square Error (MSE) loss functions proposed in the previous works [69, 111].

5.3.2 Feature embedding network

While the input image describes the appearance and the color information, the human-related feature maps extract rich semantics including the joint locations and the shape of different body parts. To leverage the information from both the input image and the human-related feature maps, we proposed to convert these information to a feature embedding (i.e., feature vector with a much reduced dimension). To achieve the goal, we concatenate the input image and the human-related feature maps, and use a CNN to extract the feature embedding. To be specific, the inputs to the Feature Embedding Network include the input image I, the pose heatmaps H_{2D}, and the body part feature maps B_{2D}. It outputs feature vector X as the input of the Graph CNN. In this work, we use a ResNet50 network [70], and extract a 2048-dimension feature vector. We denote E as our Feature Embedding Network, and its output is $X = E(I, H_{2D}, B_{2D})$.

5.3.3 Graph CNN

The Graph CNN in our proposed method constructs a human mesh by applying the projection matrix to the input feature vectors X and then compute the vertex coordinates:

$$Y = F(X; \bar{A}, W), \tag{5.1}$$

where $\bar{A} \in R^{N \times N}$ denotes the adjacency matrix of the human mesh. $X \in R^{N \times d}$ denotes a set of d-dimension feature vectors which are the output of the embedding network. $Y \in R^{N \times 3}$ is the estimated 3D coordinate for the mesh vertices. $F(X; \bar{A}, W)$ is a composition of a number of projections which can be written as:

$$F(X; \bar{A}, W) = f_T(\cdots f_2(f_1(X; \bar{A}, W_1); \bar{A}, W_2) \cdots ; \bar{A}, W_T), \tag{5.2}$$

where $f_t(\cdot)$ takes the input X_t, the adjacency matrix $\bar{A}$, and parameter W_t as inputs, and produces the projection result X_{t+1} by using:

$$X_{t+1} = f_t(X_t; \bar{A}, W_t) = \sigma(\bar{A} X_t W_t), \tag{5.3}$$

where $\sigma(\cdot)$ is the activation function introducing non-linearity to the network model. We use rectified linear unit (ReLU) in this work.

The proposed method aims to learn a series of Graph Convolution layers, which are T projection matrices $W = \{W_1, W_2, \cdots, W_T\}$ that map the input feature vectors X into the output vertex coordinates Y.

In summary, we estimate the 3D coordinates of the mesh vertices by using the Graph CNN. Our Graph CNN is in spirit similar to [95], but we do not have a SMPL regression network. Given the feature vector X extracted from our Feature Embedding Network, we attach X to the 3D coordinates of each vertex in the graph. Then, we perform a series of convolutions on the graph and output the mesh vertices Y. In addition, we use Graph CNN to predict the camera parameters $c_w = [s, t_x, t_y]$, where s, t_x, t_y indicate the scaling factor and translation of two directions, respectively. We denote R as our Graph CNN, and the outputs of our Graph CNN are $\{Y, c_w\} = \mathcal{R}(X)$.

In the following, we describe the details of the proposed loss functions for training our Graph CNN.

5.3.4 Laplacian prior

Laplacian prior has been commonly used for geometric modeling and mesh editing [175, 229, 127, 38]. In this work, we are the first to use it with Graph CNN to learn human mesh construction. Let G denote the 3D mesh of a generic human body, where G is represented as a graph $G = (V, E)$ with the edges E and the vertices V. We denote $V = [v_1, v_2, \cdots, v_n]$ and $v_i = [v_{ix}, v_{iy}, v_{iz}]$. Given a vertex v_i, the Laplacian of v_i can be written as:

$$\delta_i = \sum_{\{i,j\}\in E} w_{ij}(v_i - v_j) = v_i - \left[\sum_{\{i,j\}\in E} w_{ij} v_j\right], \tag{5.4}$$

where $\sum_{\{i,j\}\in E} w_{ij} = 1$. To compute the Laplacians for the human body mesh, assume we have n vertices in the mesh, which means $V = [v_1, v_2, \cdots, v_n]^T$. We can use an $n \times n$ Laplacian matrix:

$$L_{i,j} = \begin{cases} w_{ij} & \text{if } \{i,j\} \in E \\ -1 & \text{if } i = j \\ 0 & \text{otherwise,} \end{cases} \tag{5.5}$$

and compute the Laplacians $\Delta = [\delta_1, \delta_2, \cdot, \delta_n]^T$ using

$$\Delta = LV. \tag{5.6}$$

In the reminder of this chapter, we use the uniform Laplacian [38, 127] where the 1-ring vertex neighbors are equally weighted. The uniform Laplacian of v_i points to the centroid of its neighboring vertices, and has the nice property that its weights do not depend on the vertex positions. To obtain the Laplacian for the entire mesh, we compute the x, y and z coordinates of the Laplacian $\Delta_d = [\delta_{1d}, \delta_{2d}, ..., \delta_{nd}]^T, d \in \{x, y, z\}$, separately as

$$\Delta_d = LV_d, \tag{5.7}$$

where V_d is the position of the vertices in d coordinate, and $d \in \{x, y, z\}$. That is, $V_x = [v_{1x}, v_{2x}, \cdot, v_{nx}]^T$, $V_y = [v_{1y}, v_{2y}, \cdot, v_{ny}]^T$, and $V_z = [v_{1z}, v_{2z}, \cdot, v_{nz}]^T$.

Unlike a rigid object mesh where its Laplacian is a constant, the shape of the human body can be deformed in various ways depending on different poses and body movements. To learn the Laplacian prior, we randomly sample pose parameters and generate a large number of meshes with different poses for the average person in the SMPL database [118]. We model the density distribution of Laplacian under the framework of Gaussian Mixture Model (GMM). Given a batch of M mesh samples, we can estimate the parameters in GMM as follows.

$$P(\Delta_d) = \sum_{k=1}^{K} \hat{\phi}_{dk} \mathcal{N}(\Delta_d | \hat{\mu}_{dk}, \hat{\Sigma}_{dk}), \tag{5.8}$$

where $\hat{\phi}_{dk}, \hat{\mu}_{dk}, \hat{\Sigma}_{dk}$ are mixture probability, mean, co-variance for component k in GMM for Δ_d, $d \in \{x, y, z\}$ respectively. With the estimated parameters, the overall loss function for the Laplacian prior is written as:

$$\mathcal{L}_{Lap}(W) =$$

$$\sum_{d \in \{x,y,z\}} -\log \sum_{k=1}^{K} \hat{\phi}_{dk} \frac{\exp\left(-\frac{1}{2}(\Delta_d - \hat{\mu}_{dk})^T \hat{\Sigma}_{dk}^{-1} (\Delta_d - \hat{\mu}_{dk})\right)}{\sqrt{2\pi \hat{\Sigma}_{dk}}}. \tag{5.9}$$

In our experiments, we assume $\hat{\Sigma}_{dk}$ are diagonal matrices and estimate the GMM parameters using the EM algorithm [37]. We enforce the learning objective on the top layer of our graph convolutional network and learn the model parameters W with back-propagation.

5.3.5 3D Pose estimation

We optimize the 3D pose estimation, where the 3D pose is derived from the output mesh. Assume we have an output mesh, which is computed from the graph convolutional neural network. We regress the output mesh to the 3D pose, and minimize the error between the predicted 3D pose J_{3D} and the ground truth $\bar{J}_{3D}$. Similar to previous study [95], we apply

L1 loss function to achieve this objective:

$$\mathcal{L}_{3DPose}(W) = \frac{1}{K}\sum_{i=1}^{K}\left|\left|J_{3D} - \bar{J}_{3D}\right|\right|_1, \tag{5.10}$$

where K is the total number of joints.

5.3.6 2D Pose estimation

In addition to 3D pose estimation, we enhance the performance of pose estimation by projecting the 3D pose to the 2D pose using the weak-perspective projection with the predicted camera parameters. Following the previous works [95, 89], the camera parameters consist of a scaling factor and a 2D translation. The camera parameters are regressed using the graph convolutional neural network. We then minimize the prediction error between the predicted 2D pose J_{2D} and the ground truth $\bar{J}_{2D}$.

$$\mathcal{L}_{2DPose}(W) = \frac{1}{K}\sum_{i=1}^{K}\left|\left|J_{2D} - \bar{J}_{2D}\right|\right|_1. \tag{5.11}$$

5.3.7 2D Part segmentation

Inspired by previous studies [52, 167, 159, 92] that have shown the effectiveness of using silhouette information for 3D object modeling, we add the part segmentation in our loss function. Given the predicted camera parameters, we project the output mesh to the 2D part segmentation masks B_{2D}, and minimize the difference between the predicted part segmentation masks B_{2D} and the ground truth masks $\bar{B}_{2D}$. We apply Mean Square Error (MSE) loss function to obtain the objective:

$$\mathcal{L}_{2DPart}(W) = \frac{1}{Z}\sum_{i=1}^{Z}\left|\left|B_{2D} - \bar{B}_{2D}\right|\right|_2^2, \tag{5.12}$$

where Z is the number of body part categories. To achieve end-to-end training, we use a differentiable rendering model [92] to render the part segmentation masks, and approximate the gradients for back propagation.

5.3.8 Overall objective

To learn the neural network parameters W, the entire objective function is given as

$$\begin{aligned}\min_{W} \mathcal{L}(W) = \alpha\mathcal{L}_{Lap}(W) + \beta\mathcal{L}_{3DPose}(W) \\ + \gamma\mathcal{L}_{2DPose}(W) + \theta\mathcal{L}_{2DPart}(W),\end{aligned} \tag{5.13}$$

where $\mathcal{L}_{Lap}$ is the Laplacian prior term, $\mathcal{L}_{3DPose}$ is the 3D pose loss, $\mathcal{L}_{2DPose}$ is the 2D pose loss, and $\mathcal{L}_{2DPart}$ is the part segmentation loss. α, β, γ, θ are the hyperparamters to balance the loss terms.

5.3.9 Training

We train our model in an end-to-end fashion. We first train our Pose-Part Network using MSCOCO [113] and Pascal-Person-Parts [32] to extract human-related feature maps. Next, we train our Graph CNN using UP-3D [101] and Human3.6M [78] datasets to learn the mesh construction. Specifically, we train our Graph CNN by applying the proposed loss functions on the output of the Graph CNN. We use an Adam optimizer with a learning rate 3×10^{-4}, and the batch size is 32. Although some of the existing datasets have the 3D mesh annotations, we do not use the ground truth meshes for training. To have a fair performance comparison, we follow the previous studies [101, 135, 95, 160, 89, 141] and use the same human mesh topology as the SMPL model [118] in the experiments. To be specific, the human mesh topology consists of 6980 vertices. It is worth noting that our method does not have restrictions on the human mesh topology, and can be extended to other human mesh that does not have SMPL parameters.

In the experiment, we observe that the proposed part segmentation loss is usually larger than 2D and 3D pose estimation losses. We also observe the proposed Laplacian prior loss is usually smaller than 2D and 3D pose estimation losses. To balance the loss terms, we empirically set $\alpha = 10$, $\beta = 1.0$, $\gamma = 1.0$, $\theta = 0.1$ in our experiments.

Input GraphCMR Ours

Figure 5.4: Qualitative comparison with the state-of-the-art nonparametric approach on the UP-3D dataset. Light blue color indicates the results of the proposed method, and light pink color indicates the results of GraphCMR [95]. Without using ground truth meshes in the training, our method achieves comparable or better performance than the state-of-the-art method which requires ground truth meshes.

5.4 Experimental Results

In this section, we first describe the evaluation benchmarks, and present the experimental results and the comparisons with the state-of-the-art approaches using different evaluation metrics. Finally, we show the qualitative comparisons on a challenging video and demonstrate the robustness of the proposed approach.

5.4.1 Evaluation benchmarks

- **UP-3D [101]** is an outdoor-image dataset with rich annotations including 3D pose, 2D pose, part segmentation, and mesh ground truthes. The images are collected from 2D human pose benchmarks, such as MPII [10] and LSP [86]. The annotations are created by performing shape fitting on each human in the image. We train our model using UP-3D training data, and evaluate the performance using the metric of mean Per-Vertex-Error (mPVE) on the UP-3D test set.

- **Human3.6M [78]** is an indoor large-scale dataset with 3D pose annotations. Each image has a subject performing a different action. Following the common setting [95], we use the subjects S1, S5, S6, S7 and S8 for training, and use the subjects S9 and S11 for testing.

- **LSP [86]** is an outdoor-image dataset. We evaluate part segmentation performance on LSP test set, where the segmentation labels are provided by Lassner *et al.*[101].

- **3DPW [194]** is an outdoor large-scale dataset with mesh ground truth labels. We evaluate the robustness of our method with cross-dataset evaluation, i.e., trained on UP-3D dataset and applied to 3DPW dataset.

Table 5.1: Performance comparison of human mesh construction using metric mean Per-Vertex-Error (mPVE) on the UP-3D test set. The unit is millimetter (mm).

Method	mean Per-Vertex-Error
Lassner *et al.*[101]	169.8
NBF [135]	134.6
HMR [89]	149.2
DC [160]	137.5
Pavlakos *et al.*[141]	100.5
GraphCMR [95]	100.2
Ours	81.5
Ours + GT Inputs	**73.7**

Table 5.2: Performance comparison of human mesh construction using metric mean Per-Vertex-Error (mPVE) on 3DPW sequences. The unit is millimeter (mm).

Method	DTCrossStreets	DTRampAndStairs	DTRunForBus	DTWarmWelcome	OutdoorsFencing	CourtyardDancing
GraphCMR [95]	85.22	86.69	70.57	85.02	73.90	112.36
Ours	**83.47**	**83.99**	**68.87**	**83.92**	**70.88**	**71.04**

5.4.2 Main results

We compare the performance of our method with the state-of-the-art approaches which require either ground truth meshes or 2D-to-3D dense correspondence labels. We evaluate the performance of mesh construction by using the metric mean Per-Vertex-Error (mPVE) [141], where the unit is millimeter (mm). For each mesh vertex, we estimate the Euclidean distance between the ground-truth location and the predicted location. We average over all the vertices to provide the mPVE. Table 5.1 shows the performance comparison on UP-3D dataset. Our method outperforms the previous state-of-the-art approaches by a significant margin.

Since we use Pose-Part Network to extract human-related features (pose heatmaps and body part feature maps) from the input image, one may wonder what if we use the ground truth labels for both pose heatmaps and body part masks as the inputs of our Feature Embedding Network. To answer the questions, we have conducted this experiment and the result is shown in the bottom row of Table 5.1. The results show that pose estimation and part segmentation are useful for human mesh construction.

Figure 5.4 shows the qualitative comparisons with the state-of-the-art nonparametric approach (GraphCMR [95]) which also uses graph convolutional neural network but directly regresses the ground truth mesh vertices. The results show that, without using the ground truth meshes, our method is on par or even slightly better than the existing techniques which require ground truths.

We evaluate the robustness of the proposed method on the 3DPW dataset [194]. Table 5.2 shows the performance comparison with the state-of-the-art nonparametric approach [95]. Both models are trained using UP-3D and Human3.6M but without 3DPW dataset. Our method does not use any of the 3D ground truth meshes in either UP-3D or Human3.6M, while GraphCMR [95] used the ground truth meshes of both UP-3D and Human3.6M. For a fair comparison, we use ground truth bounding boxes to crop the persons as the inputs for the two methods. Our method performs comparably or better than GraphCMR [95].

We evaluate the 3D pose of the constructed mesh by comparing the performance of 3D pose estimation on Human3.6M dataset [78] using Protocol 2 Reconstruction Error metric [78, 230, 122], where the unit is millimeter (mm). In Table 5.3, the upper-rows show the state-of-the-art results that try to regress SMPL parameters for human mesh construction. The bottom two rows show the comparison of our method with the state-of-the-art nonparametric method that does not regress SMPL parameters. Our method does not use any of the ground truth meshes in training, and achieves comparable or even better performance than several baseline approaches that require Human3.6M ground truth meshes.

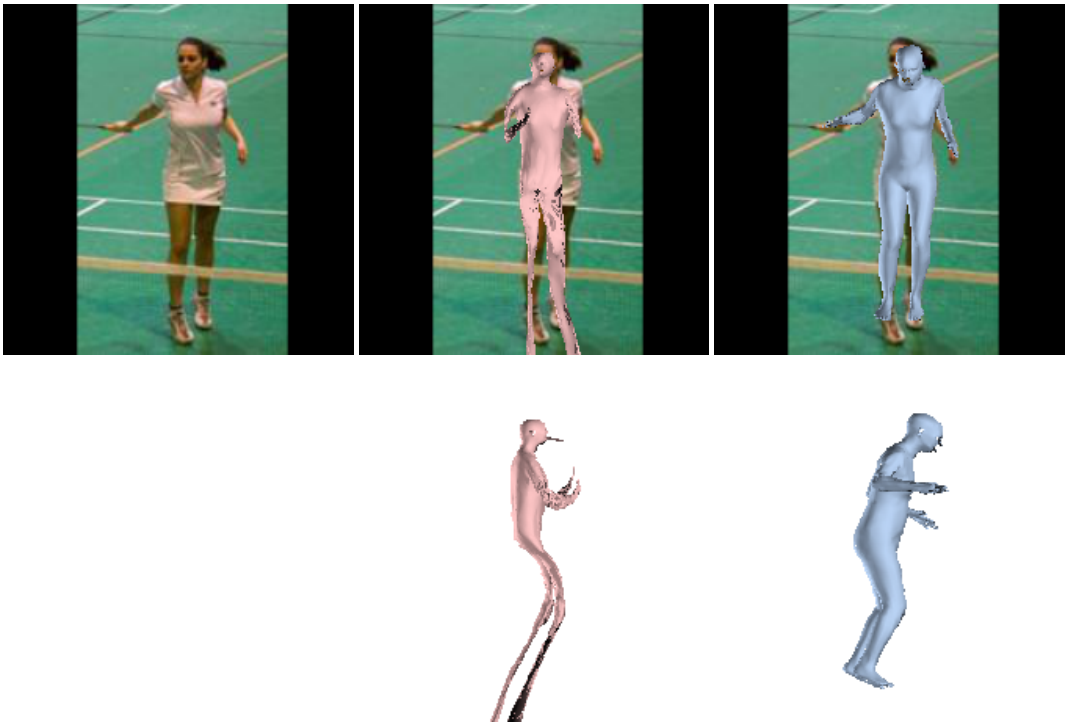
We also evaluate the 3D shape by comparing the performance of part segmentation on LSP test set. Following the common settings [89, 95], we report the segmentation accuracy

Table 5.3: Evaluation of 3D pose estimation on Human3.6M dataset using Protocol 2. The results are Reconstruction errors in millimeter (mm). Our approach is competitive with the state-of-the-art approaches.

Method	SMPL	Reconst. Error (mm)
Lassner *et al.*[101]	✓	93.9
SMPLify [17]	✓	82.3
Pavlakos *et al.* [141]	✓	75.9
HMR unpaired [89]	✓	66.5
NBF [135]	✓	59.9
HMR [89]	✓	56.8
GraphCMR+SMPL [95]	✓	50.1
GraphCMR [95]	✗	69.0
Ours	✗	58.5

Table 5.4: Performance comparison of segmentation on LSP test set. The numbers are accuracy scores and F1 scores. The top three rows show the approaches that perform some optimization (post)-processing. The bottom three rows show the comparison with the regression-based approaches. Without using ground truth meshes in training, our approach is competitive with the state-of-the-art methods which require ground truth mesh labels.

	FB Seg.		Part Seg.	
Method	Accuracy	F1	Accuracy	F1
SMPLify [17]	91.89	0.88	87.71	0.64
SMPLify on [141]	92.17	0.88	88.24	0.64
BodyNet [190]	92.75	0.84	–	–
HMR [89]	91.67	0.87	87.12	0.60
GraphCMR [95]	91.46	0.87	88.69	0.66
Ours	91.23	0.86	88.86	0.66



Input *w/o Laplacian* *w/ Laplacian*

Figure 5.5: Qualitative comparison of our method using different training configurations.

and the average F1 score for six body parts and the background in Table 5.4. We also report the results on foreground-background segmentation. Our method achieves comparable or better performance than the state-of-the-arts approaches that use ground truth meshes in training.

5.4.3 Ablation study

Laplacian prior. Since our approach learns with the Laplacian prior, one interesting question is whether the proposed learning objective is useful. To answer this question, we have trained our network without the Laplacian prior (i.e. with pose and segmentation losses only). This configuration is denoted as *w/o Laplacian*, and the qualitative comparisons are shown in Figure 5.5. We can see that Laplacian prior is critical to our learning objective for

Table 5.5: Ablation study of the proposed two loss terms, evaluated on UP-3D test set with mean Per-Vertex-Error. The unit is millimeter (mm).

Laplacian prior	Part Seg Loss	mean Per-Vertex-Error
✗	✓	240.3
✓	✗	91.3
✓	✓	81.5

Table 5.6: Ablation study of the Pose-Part Network, also evaluated on UP-3D test set with mean Per-Vertex-Error. The unit is millimeter (mm).

Method	mean Per-Vertex-Error
Ours w/o Pose-Part Net	110.0
Ours	81.5

human mesh construction.

Part segmentation loss. We also evaluate the effectiveness of the proposed part segmentation loss, and Table 5.5 shows the comparison. For completeness, we also show the results of training with the Laplacian prior. We can see that training only with part segmentation loss does not work well, and Laplacian prior further improves the results. Our model achieves the best performance when two proposed loss terms are used.

Pose-Part Network. Since our Pose-Part Network predicts pose heatmaps and part segmentation masks, one may wonder whether this is useful. To answer the question, we train our model without Pose-Part Network, and Table 5.6 shows the results. We can see that pose heatmaps and part segmentation masks significantly improve the learning.

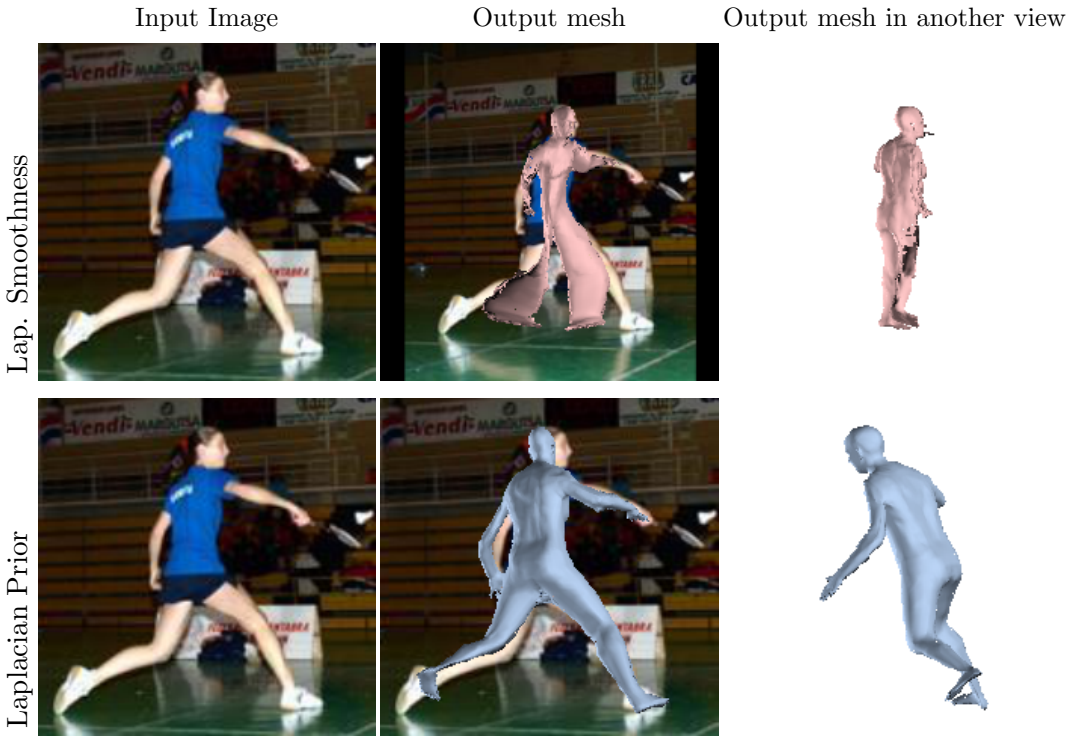


Figure 5.6: Comparison of Laplacian smoothness and the proposed Laplacian prior.

Analysis of different regularizers. Laplacian smoothness is commonly used in the literature [200, 90] as a regularizer to avoid self-intersections for 3D object modeling. One may wonder what if we replace the proposed Laplacian prior with the Laplacian smoothness. We have conducted this experiment and Figure 5.6 shows the qualitative comparison. We can see that training with Laplacian smoothness produces wrong results. This is because Laplacian smoothness term is not a strong enough regularizer, and as a result the training process usually gets stuck in a local minimum. The results show that it is very helpful to model the Laplacian prior with Gaussian Mixture Model (GMM) because it effectively models the distribution of the human mesh vertices.

Extension to supervised training. We study the *upper bound* performance of our method when ground truth meshes are available for training, and Table 5.7 shows the results. We add a vertex regression loss to our learning objective (Eq.(5.13)), and train our model with

Table 5.7: Ablation study of our method with and without using GT meshes in training, evaluated on UP-3D test set with mean Per-Vertex-Error. The unit is millimeter (mm).

Method	mean Per-Vertex-Error
Ours, without GT meshes	81.5
Ours, with GT meshes	65.1

the ground truth meshes provided in UP-3D training set. Our model improves the previous state-of-the-art performance to 65 mPVE on UP-3D test set.

5.4.4 Qualitative comparison

We conduct qualitative comparisons with the state-of-the-art methods [95, 89] on the challenging 3DPW dataset [194], and Figure 5.7 shows the results. We can see that previous state-of-the-art approaches [95, 89] had difficulties to construct the mesh of the person on the right due to occlusions, and they failed completely as the occlusions became more severe. Our method constructs correct human meshes for both people even though there are quite severe occlusions between them. By explicitly feeding 2D pose heatmaps and part segmentation masks into the Graph CNN feature embedding, the robustness of our method to occlusions has been significantly improved.

5.5 Conclusion

We presented a novel nonparametric approach to construct the 3D human mesh from a single image. Compared with the existing methods, our technique does not require any ground truth meshes during training. We introduced a Laplacian prior term and the part segmentation term in the loss function of the Graph CNN. In addition, we fed the pose estimation heatmaps and part segmentation masks to the feature embedding network to improve the robustness against occlusions. Experiments demonstrated that our technique is on par or outperforms existing techniques that use ground truth meshes in training.

Figure 5.7: Qualitative comparison with the previous state-of-the-art approaches [89, 95] on the challenging 3DPW dataset [194]. The top row shows two people embracing each other. The second row shows the results of a representative parametric approach HMR [89]. The third row shows the results of the previous state-of-the-art nonparametric approach GraphCMR [95]. The bottom row shows our results. Previous approaches failed to construct the mesh for the two persons due to occlusions. In contrast, our method constructs correct human meshes for both people in all the frames.

Chapter 6

SUMMARY AND FUTURE WORK

6.1 Summary

In this thesis, we have presented our works on self-supervised learning and domain adaptation for computer vision research problems. In Chapter 2, we proposed a self-supervised learning model, named DeepBit, for generating binary descriptors using deep CNNs [109]. We designed an auxiliary training protocol for deep CNNs, which automatically generates training labels describing the similarity between the given two images. During training, the proposed model learns the pair-wise image similarity through the synthetically generated labels, and is also forced to learn important properties of the binary descriptors. Experimental results show that our method achieves better performance than the relevant approaches on patch matching, image retrieval, and object recognition.

In Chapter 3, we proposed a cross-domain complementary learning for multi-person part segmentation [111]. We propose to learn an auxiliary task of human pose estimation, which helps the proposed model to bridge the domain gap between real and synthetic domains. As the result, the proposed model learns part segmentation from graphics simulation, and works well on real images. Experimental results show that our method outperforms the state-of-the-art domain adaptation approaches on multiple public datasets. Also, our method outperforms the supervised state-of-the-art methods if real part segmentation training labels are available for training. We also show that our method can be generalized to learn new keypoints such as those on the hand and feet without manual labeling.

In Chapter 4, we proposed a concatenated pyramid deep convolutional neural network for fast multi-person pose estimation. The proposed method learns multi-scale image representations using a concatenated pyramid network architecture, and thus removes the need

for the multi-stage and multi-scale detection. We conducted extensive experiments on the public datasets, ans show that our network is faster than the state-of-the-art OpenPose while achieving higher accuracy.

In Chapter 5, we proposed a novel learning algorithm for nonparametric human mesh construction without ground truth labels [110]. We propose to use the Laplacian prior with the Gaussian Mixture Model to model the vertex distributions. We propose to project the 3D mesh to 2D and minimize the difference between the projections and the 2D part segmentation map. Through experiments, we show that our method effectively constructs the 3D human mesh, and relieves the need of 3D mesh ground truth labeling.

6.2 Future Work

Self-supervised learning has become an emerging research direction. There are many new research papers being released on arXiv everyday. There are a few research topics that I feel promising for future works.

- **Visual representation learning:** Recently, learning image representations in an self-supervised manner has become a very active direction. The common theme of the recent works [30, 34] is to explore the self-similarity among the images, which is closely related to our work [109]. Research could be conducted to further understand this problem. It has great potentials for a variety of applications in computer vision, robotics, and natural language processing.

- **Automatic learning:** The core idea of this dissertation is to design an auxiliary training protocol for the neural networks, so that the neural networks can teach itself without manual labeling. However, the auxiliary training protocol is designed in a task-specific fashion, and it usually requires domain knowledge for protocol design. In future, it would be interesting to develop automatic learning algorithms that can help design the self-supervision algorithm.

BIBLIOGRAPHY

[1] AI Challenger. https://challenger.ai/.

[2] Arnold. https://www.arnoldrenderer.com/.

[3] Autodesk software. https://www.autodesk.com/.

[4] Carnegie Mellon University Motion Capture Database. http://mocap.cs.cmu.edu/.

[5] Pulkit Agrawal, Joao Carreira, and Jitendra Malik. Learning to see by moving. In *Proc. ICCV*, pages 37–45, 2015.

[6] Alexandre Alahi, Raphael Ortiz, and Pierre Vandergheynst. Freak: Fast retina keypoint. In *Proc. CVPR*, 2012.

[7] Riza Alp Guler, George Trigeorgis, Epameinondas Antonakos, Patrick Snape, Stefanos Zafeiriou, and Iasonas Kokkinos. Densereg: Fully convolutional dense shape regression in-the-wild. In *Proc. CVPR*, 2017.

[8] Alexandr Andoni and Piotr Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *Proc. FOCS*, 2006.

[9] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *Proc. CVPR*, 2014.

[10] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *Proc. CVPR*, 2014.

[11] Mykhaylo Andriluka, Stefan Roth, and Bernt Schiele. Pictorial structures revisited: People detection and articulated pose estimation. In *Proc. CVPR*, 2009.

[12] Mykhaylo Andriluka, Stefan Roth, and Bernt Schiele. Monocular 3d pose estimation and tracking by detection. In *Proc. CVPR*, 2010.

[13] Artem Babenko, Anton Slesarev, Alexandr Chigorin, and Victor Lempitsky. Neural codes for image retrieval. In *Proc. ECCV*, pages 584–599, 2014.

[14] Vassileios Balntas, Edward Johns, Lilian Tang, and Krystian Mikolajczyk. Pn-net: conjoined triple deep network for learning local image descriptors. *arXiv preprint arXiv:1601.05030*, 2016.

[15] Vassileios Balntas, Lilian Tang, and Krystian Mikolajczyk. Bold-binary online learned descriptor for efficient image matching. In *Proc. CVPR*, 2015.

[16] Rodrigo Benenson, Markus Mathias, Radu Timofte, and Luc Van Gool. Pedestrian detection at 100 frames per second. In *Proc. CVPR*, 2012.

[17] Federica Bogo, Angjoo Kanazawa, Christoph Lassner, Peter Gehler, Javier Romero, and Michael J Black. Keep it smpl: Automatic estimation of 3d human pose and shape from a single image. In *Proc. ECCV*, 2016.

[18] Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. Domain separation networks. In *Proc. NeurIPS*, 2016.

[19] Matthew Brown, Gang Hua, and Simon Winder. Discriminative learning of local image descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):43–57, 2011.

[20] Matthew Brown and David G Lowe. Automatic panoramic image stitching using invariant features. *IJCV*, 74(1):59–73, 2007.

[21] Adrian Bulat and Georgios Tzimiropoulos. Human pose estimation via convolutional part heatmap regression. In *Proc. ECCV*, 2016.

[22] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. Brief: Binary robust independent elementary features. In *Proc. ECCV*, 2010.

[23] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proc. CVPR*, 2017.

[24] Joao Carreira, Pulkit Agrawal, Katerina Fragkiadaki, and Jitendra Malik. Human pose estimation with iterative error feedback. In *Proc. CVPR*, 2016.

[25] Tsung-Han Chan, Kui Jia, Shenghua Gao, Jiwen Lu, Zinan Zeng, and Yi Ma. Pcanet: A simple deep learning baseline for image classification? *arXiv preprint arXiv:1404.3606*, 2014.

[26] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[27] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. In *Proc. ICLR*, 2015.

[28] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2018.

[29] Liang-Chieh Chen, Yi Yang, Jiang Wang, Wei Xu, and Alan L Yuille. Attention to scale: Scale-aware semantic image segmentation. In *Proc. CVPR*, 2016.

[30] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

[31] Xianjie Chen, Roozbeh Mottaghi, Xiaobai Liu, Sanja Fidler, Raquel Urtasun, and Alan Yuille. Detect what you can: Detecting and representing objects using holistic models and body parts. In *Proc. CVPR*, 2014.

[32] Xianjie Chen, Roozbeh Mottaghi, Xiaobai Liu, Sanja Fidler, Raquel Urtasun, and Alan Yuille. Detect what you can: Detecting and representing objects using holistic models and body parts. In *Proc. CVPR*, 2014.

[33] Xianjie Chen and Alan L Yuille. Articulated pose estimation by a graphical model with image dependent pairwise relations. In *Proc. NIPS*, 2014.

[34] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

[35] Yilun Chen, Zhicheng Wang, Yuxiang Peng, Zhiqiang Zhang, Gang Yu, and Jian Sun. Cascaded pyramid network for multi-person pose estimation. In *Proc. CVPR*, 2018.

[36] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Proc. CVPR*, 2005.

[37] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

[38] Mathieu Desbrun, Mark Meyer, Peter Schröder, and Alan H Barr. Implicit fairing of irregular meshes using diffusion and curvature flow. In *SIGGRAPH*, 1999.

[39] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[40] Thanh-Toan Do, Anh-Dzung Doan, and Ngai-Man Cheung. Learning to hash with binary deep neural network. In *Proc. ECCV*, 2016.

[41] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proc. ICCV*, pages 1422–1430, 2015.

[42] Jian Dong, Qiang Chen, Xiaohui Shen, Jianchao Yang, and Shuicheng Yan. Towards unified human parsing and pose estimation. In *Proc. CVPR*, 2014.

[43] Alexey Dosovitskiy, Philipp Fischer, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with exemplar convolutional neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(9):1734–1747, 2016.

[44] Wenbin Du, Yali Wang, and Yu Qiao. Rpan: An end-to-end recurrent pose-attention network for action recognition in videos. In *Proc. ICCV*, 2017.

[45] Yueqi Duan, Jiwen Lu, Ziwei Wang, Jianjiang Feng, and Jie Zhou. Learning deep binary descriptor with multi-quantization. In *Proc. CVPR*, 2017.

[46] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *Int. J. Comput. Vis*, 111(1):98–136, 2015.

[47] Bin Fan, Qingqun Kong, Tomasz Trzcinski, Zhiheng Wang, Chunhong Pan, and Pascal Fua. Receptive fields selection for binary feature description. *IEEE TIP*, 23(6):2583–2595, 2014.

[48] Hao-Shu Fang, Guansong Lu, Xiaolin Fang, Jianwen Xie, Yu-Wing Tai, and Cewu Lu. Weakly and semi supervised human body part parsing via pose-guided knowledge transfer. In *Proc. CVPR*, 2018.

[49] Hao-Shu Fang, Shuqin Xie, Yu-Wing Tai, and Cewu Lu. Rmpe: Regional multi-person pose estimation. In *Proc. CVPR*, 2017.

[50] Pedro F Felzenszwalb, Ross B Girshick, and David McAllester. Cascade object detection with deformable part models. In *Proc. CVPR*, 2010.

[51] Basura Fernando, Elisa Fromont, and Tinne Tuytelaars. Mining mid-level features for image classification. *IJCV*, 108(3):186–203, 2014.

[52] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8):1362–1376, 2009.

[53] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *Proc. ICML*, 2015.

[54] Victor Garcia and Joan Bruna. Few-shot learning with graph neural networks. *arXiv preprint arXiv:1711.04043*, 2017.

[55] Liuhao Ge, Zhou Ren, Yuncheng Li, Zehao Xue, Yingying Wang, Jianfei Cai, and Junsong Yuan. 3d hand shape and pose estimation from a single rgb image. In *Proc. CVPR*, 2019.

[56] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proc. CVPR*, 2014.

[57] Georgia Gkioxari, Bharath Hariharan, Ross Girshick, and Jitendra Malik. R-cnns for pose estimation and action detection. *arXiv preprint arXiv:1406.5212*, 2014.

[58] Georgia Gkioxari, Alexander Toshev, and Navdeep Jaitly. Chained predictions using convolutional neural networks. In *Proc. ECCV*, 2016.

[59] Ke Gong, Xiaodan Liang, Dongyu Zhang, Xiaohui Shen, and Liang Lin. Look into person: Self-supervised structure-sensitive learning and a new benchmark for human parsing. In *Proc. CVPR*, 2017.

[60] Yunchao Gong and Svetlana Lazebnik. Iterative quantization: A procrustean approach to learning binary codes. In *Proc. CVPR*, 2011.

[61] Yunchao Gong, Svetlana Lazebnik, Albert Gordo, and Florent Perronnin. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(12):2916–2929, 2013.

[62] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[63] Kristen Grauman and Trevor Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *Proc. ICCV*, 2005.

[64] Kristen Grauman and Rob Fergus. Learning binary hash codes for large-scale image search. In *Machine learning for computer vision*, pages 49–87. Springer, 2013.

[65] Chunhui Gu, Chen Sun, David Ross, Carl Vondrick, Caroline Pantofaru, Yeqing Li, Sudheendra Vijayanarasimhan, George Toderici, Susanna Ricco, Rahul Sukthankar, et al. Ava: A video dataset of spatio-temporally localized atomic visual actions. In *Proc. CVPR*, 2018.

[66] Riza Alp Guler and Iasonas Kokkinos. Holopose: Holistic 3d human reconstruction in-the-wild. In *Proc. CVPR*, 2019.

[67] Rıza Alp Güler, Natalia Neverova, and Iasonas Kokkinos. Densepose: Dense human pose estimation in the wild. In *Proc. CVPR*, 2018.

[68] Xufeng Han, Thomas Leung, Yangqing Jia, Rahul Sukthankar, and Alexander C Berg. Matchnet: Unifying feature and metric learning for patch-based matching. In *Proc. CVPR*, pages 3279–3286, 2015.

[69] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proc. ICCV*, 2017.

[70] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. CVPR*, 2016.

[71] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[72] Chen Huang, Chen Change Loy, and Xiaoou Tang. Unsupervised learning of discriminative attributes and visual representations. In *Proc. CVPR*, 2016.

[73] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proc. CVPR*, 2017.

[74] Shaoli Huang, Mingming Gong, and Dacheng Tao. A coarse-fine network for keypoint localization. In *Proc. ICCV*, 2017.

[75] Yinghao Huang, Manuel Kaufmann, Emre Aksan, Michael J. Black, Otmar Hilliges, and Gerard Pons-Moll. Deep inertial poser learning to reconstruct human pose from sparseinertial measurements in real time. *ACM Transactions on Graphics*, 37(6):185:1–185:15, November 2018.

[76] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

[77] Eldar Insafutdinov, Leonid Pishchulin, Bjoern Andres, Mykhaylo Andriluka, and Bernt Schiele. Deepercut: A deeper, stronger, and faster multi-person pose estimation model. In *Proc. ECCV*, 2016.

[78] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, 2014.

[79] Arjun Jain, Jonathan Tompson, Mykhaylo Andriluka, Graham W Taylor, and Christoph Bregler. Learning human pose estimation features with convolutional networks. *arXiv preprint arXiv:1312.7302*, 2013.

[80] Ashesh Jain, Amir R Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-rnn: Deep learning on spatio-temporal graphs. In *Proc. CVPR*, 2016.

[81] Eric Jang, Coline Devin, Vincent Vanhoucke, and Sergey Levine. Grasp2vec: Learning object representations from self-supervised grasping. *arXiv preprint arXiv:1811.06964*, 2018.

[82] Herve Jegou, Matthijs Douze, and Cordelia Schmid. Hamming embedding and weak geometric consistency for large scale image search. *Proc. ECCV*, 2008.

[83] Herve Jegou, Florent Perronnin, Matthijs Douze, Jorge Sánchez, Patrick Perez, and Cordelia Schmid. Aggregating local image descriptors into compact codes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1704–1716, 2012.

[84] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proc. ACM MM*, 2014.

[85] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image generation from scene graphs. In *Proc. CVPR*, 2018.

[86] Sam Johnson and Mark Everingham. Clustered pose and nonlinear appearance models for human pose estimation. In *Proc. BMVC*, 2010.

[87] Hanbyul Joo, Tomas Simon, and Yaser Sheikh. Total capture: A 3d deformation model for tracking faces, hands, and bodies. In *Proc. CVPR*, 2018.

[88] Gregory Kahn, Adam Villaflor, Bosen Ding, Pieter Abbeel, and Sergey Levine. Self-supervised deep reinforcement learning with generalized computation graphs for robot navigation. In *Proc. ICRA*, pages 1–8. IEEE, 2018.

[89] Angjoo Kanazawa, Michael J Black, David W Jacobs, and Jitendra Malik. End-to-end recovery of human shape and pose. In *Proc. CVPR*, pages 7122–7131, 2018.

[90] Angjoo Kanazawa, Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *Proc. ECCV*, 2018.

[91] Angjoo Kanazawa, Jason Y Zhang, Panna Felsen, and Jitendra Malik. Learning 3d human dynamics from video. In *Proc. CVPR*, 2019.

[92] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3d mesh renderer. In *Proc. CVPR*, 2018.

[93] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[94] Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. Revisiting self-supervised visual representation learning. *arXiv preprint arXiv:1901.09005*, 2019.

[95] Nikos Kolotouros, Georgios Pavlakos, and Kostas Daniilidis. Convolutional mesh regression for single-image human shape reconstruction. In *Proc. CVPR*, 2019.

[96] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *Proc. NeurIPS*, 2012.

[97] Alex Krizhevsky. Learning multiple layers of features from tiny images. *Computer Science Department, University of Toronto, Tech. Report*, 2009.

[98] BG Kumar, Gustavo Carneiro, Ian Reid, et al. Learning local image descriptors with deep siamese and triplet convolutional networks by minimising global loss functions. In *Proc. CVPR*, pages 5385–5394, 2016.

[99] Lubor Ladicky, Philip HS Torr, and Andrew Zisserman. Human pose estimation using a joint pixel-wise and part-wise formulation. In *Proc. CVPR*, 2013.

[100] Hanjiang Lai, Yan Pan, Ye Liu, and Shuicheng Yan. Simultaneous feature learning and hash coding with deep neural networks. In *Proc. CVPR*, 2015.

[101] Christoph Lassner, Javier Romero, Martin Kiefel, Federica Bogo, Michael J Black, and Peter V Gehler. Unite the people: Closing the loop between 3d and 2d human representations. In *Proc. CVPR*, pages 6050–6059, 2017.

[102] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[103] Stefan Leutenegger, Margarita Chli, and Roland Y Siegwart. Brisk: Binary robust invariant scalable keypoints. In *Proc. ICCV*, 2011.

[104] Sijin Li, Zhi-Qiang Liu, and Antoni B Chan. Heterogeneous multi-task learning for human pose estimation with deep convolutional neural network. In *Proc. CVPR Workshop*, 2014.

[105] Xiaodan Liang, Xiaohui Shen, Jiashi Feng, Liang Lin, and Shuicheng Yan. Semantic object parsing with graph lstm. In *Proc. ECCV*, 2016.

[106] Xiaodan Liang, Xiaohui Shen, Donglai Xiang, Jiashi Feng, Liang Lin, and Shuicheng Yan. Semantic object parsing with local-global long short-term memory. In *Proc. CVPR*, 2016.

[107] Joerg Liebelt and Cordelia Schmid. Multi-view object class detection with a 3d geometric model. In *Proc. CVPR*, 2010.

[108] Kevin Lin, Jiwen Lu, Chu-Song Chen, and Jie Zhou. Learning compact binary descriptors with unsupervised deep neural networks. In *Proc. CVPR*, 2016.

[109] Kevin Lin, Jiwen Lu, Chu-Song Chen, Jie Zhou, and Ming-Ting Sun. Unsupervised deep learning of compact binary descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(6):1501–1514, 2019.

[110] Kevin Lin, Lijuan Wang, Ying Jin, Zicheng Liu, and Ming-Ting Sun. Learning nonparametric human mesh reconstruction from a single image without ground truth meshes. *arXiv preprint arXiv:2003.00052*, 2020.

[111] Kevin Lin, Lijuan Wang, Kun Luo, Yinpeng Chen, Zicheng Liu, and Ming-Ting Sun. Cross-domain complementary learning with synthetic data for multi-person part segmentation. *arXiv preprint arXiv:1907.05193*, 2019.

[112] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proc. CVPR*, 2017.

[113] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Proc. ECCV*, 2014.

[114] Venice Erin Liong, Jiwen Lu, Gang Wang, Pierre Moulin, and Jie Zhou. Deep hashing for compact binary codes learning. In *Proc. CVPR*, 2015.

[115] Or Litany, Alex Bronstein, Michael Bronstein, and Ameesh Makadia. Deformable shape completion with graph convolutional autoencoders. In *Proc. CVPR*, 2018.

[116] Haomiao Liu, Ruiping Wang, Shiguang Shan, and Xilin Chen. Deep supervised hashing for fast image retrieval. In *Proc. CVPR*, 2016.

[117] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proc. CVPR*, 2015.

[118] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *ACM transactions on graphics (TOG)*, 34(6):248, 2015.

[119] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[120] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *J. Mach. Learn. Res.*, 9(Nov):2579–2605, 2008.

[121] Javier Marin, David Vázquez, David Gerónimo, and Antonio M López. Learning appearance in virtual scenarios for pedestrian detection. In *Proc. CVPR*, 2010.

[122] Julieta Martinez, Rayat Hossain, Javier Romero, and James J Little. A simple yet effective baseline for 3d human pose estimation. In *Proc. ICCV*, 2017.

[123] Krystian Mikolajczyk and Cordelia Schmid. Indexing based on scale invariant interest points. In *Proc. ICCV*, 2001.

[124] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, 2005.

[125] T Nathan Mundhenk, Daniel Ho, and Barry Y Chen. Improvements to context based self-supervised learning. In *Prc. CVPR*, pages 9339–9348, 2018.

[126] Ryota Natsume, Shunsuke Saito, Zeng Huang, Weikai Chen, Chongyang Ma, Hao Li, and Shigeo Morishima. Siclope: Silhouette-based clothed people. In *Proc. CVPR*, 2019.

[127] Andrew Nealen, Takeo Igarashi, Olga Sorkine, and Marc Alexa. Laplacian mesh optimization. In *ACM GRAPHITE*, 2006.

[128] Ramakant Nevatia and Thomas O Binford. Description and recognition of curved objects. *Artificial Intelligence*, 8(1):77–98, 1977.

[129] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohli, Jamie Shotton, Steve Hodges, and Andrew W Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Proc. ISMAR*, 2011.

[130] Alejandro Newell, Zhiao Huang, and Jia Deng. Associative embedding: End-to-end learning for joint detection and grouping. In *Proc. NIPS*, 2017.

[131] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *Proc. ECCV*, 2016.

[132] Maria-Elena Nilsback and Andrew Zisserman. A visual vocabulary for flower classification. In *Proc. CVPR*, 2006.

[133] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Proc. ICVGIP*, 2008.

[134] David Nister and Henrik Stewenius. Scalable recognition with a vocabulary tree. In *Proc. CVPR*, 2006.

[135] Mohamed Omran, Christoph Lassner, Gerard Pons-Moll, Peter Gehler, and Bernt Schiele. Neural body fitting: Unifying deep learning and model based human pose and shape estimation. In *Proc. 3DV*, 2018.

[136] Bahadir Ozdemir, Mahyar Najibi, and Larry S Davis. Supervised incremental hashing. *arXiv preprint arXiv:1604.07342*, 2016.

[137] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 10(22):1345–1359, 2010.

[138] George Papandreou, Tyler Zhu, Nori Kanazawa, Alexander Toshev, Jonathan Tompson, Chris Bregler, and Kevin Murphy. Towards accurate multi-person pose estimation in the wild. In *Proc. CVPR*, 2017.

[139] Mattis Paulin, Matthijs Douze, Zaid Harchaoui, Julien Mairal, Florent Perronin, and Cordelia Schmid. Local convolutional features with unsupervised training for image retrieval. In *Proc. ICCV*, 2015.

[140] Georgios Pavlakos, Nikos Kolotouros, and Kostas Daniilidis. Texturepose: Supervising human mesh estimation with texture consistency. In *Proc. ICCV*, 2019.

[141] Georgios Pavlakos, Luyang Zhu, Xiaowei Zhou, and Kostas Daniilidis. Learning to estimate 3d human pose and shape from a single color image. In *Proc. CVPR*, pages 459–468, 2018.

[142] Xingchao Peng, Baochen Sun, Karim Ali, and Kate Saenko. Learning deep object detectors from 3d models. In *Proc. ICCV*, 2015.

[143] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Proc. CVPR*, 2007.

[144] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *Proc. CVPR*, 2008.

[145] Leonid Pishchulin, Mykhaylo Andriluka, Peter Gehler, and Bernt Schiele. Poselet conditioned pictorial structures. In *Proc. CVPR*, 2013.

[146] Leonid Pishchulin, Eldar Insafutdinov, Siyu Tang, Bjoern Andres, Mykhaylo Andriluka, Peter V Gehler, and Bernt Schiele. Deepcut: Joint subset partition and labeling for multi person pose estimation. In *Proc. CVPR*, 2016.

[147] Alin-Ionut Popa, Mihai Zanfir, and Cristian Sminchisescu. Deep multitask architecture for integrated 2d and 3d human sensing. In *Proc. CVPR*, 2017.

[148] Charles R. Qi, Hao Su, Matthias Niessner, Angela Dai, Mengyuan Yan, and Leonidas J. Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *Proc. CVPR*, 2016.

[149] Siyuan Qi, Wenguan Wang, Baoxiong Jia, Jianbing Shen, and Song-Chun Zhu. Learning human-object interactions by graph parsing neural networks. In *Proc. ECCV*, 2018.

[150] Xiaojuan Qi, Renjie Liao, Jiaya Jia, Sanja Fidler, and Raquel Urtasun. 3d graph neural networks for rgbd semantic segmentation. In *Proc. CVPR*, 2017.

[151] Varun Ramakrishna, Daniel Munoz, Martial Hebert, James Andrew Bagnell, and Yaser Sheikh. Pose machines: Articulated pose estimation via inference machines. In *Proc. ECCV*, 2014.

[152] Deva Ramanan, David A Forsyth, and Andrew Zisserman. Strike a pose: Tracking people by finding stylized poses. In *Proc. CVPR*, 2005.

[153] Anurag Ranjan, Timo Bolkart, Soubhik Sanyal, and Michael J Black. Generating 3d faces using convolutional mesh autoencoders. In *Proc. ECCV*, 2018.

[154] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. CNN features off-the-shelf: An astounding baseline for recognition. In *Proc. CVPRW*, 2014.

[155] Konda Reddy Mopuri and R Venkatesh Babu. Object level deep feature pooling for compact image representation. In *Proc. CVPRW*, 2015.

[156] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Proc. NeurIPS*, 2015.

[157] Zhongzheng Ren and Yong Jae Lee. Cross-domain self-supervised multi-task feature learning using synthetic imagery. In *Proc. CVPR*, 2018.

[158] Stephan R Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *Proc. ECCV*, 2016.

[159] Alec Rivers, Frédo Durand, and Takeo Igarashi. 3d modeling with silhouettes. In *ACM SIGGRAPH*, 2010.

[160] Yu Rong, Ziwei Liu, Cheng Li, Kaidi Cao, and Chen Change Loy. Delving deep into hybrid annotations for 3d human recovery in the wild. In *Proc. ICCV*, 2019.

[161] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *Proc. CVPR*, 2016.

[162] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: an efficient alternative to sift or surf. In *Proc. ICCV*, 2011.

[163] Sebastian Ruder. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017.

[164] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 115(3):211–252, 2015.

[165] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. *arXiv preprint arXiv:1905.05172*, 2019.

[166] Ruslan Salakhutdinov and Geoffrey E. Hinton. Semantic hashing. *IJAR*, 50(7):969–978, 2009.

[167] Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Proc. CVPR*, 2006.

[168] Gregory Shakhnarovich. *Learning task-specific similarity*. PhD thesis, Massachusetts Institute of Technology, 2005.

[169] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Proc. CVPR workshops*, 2014.

[170] Fumin Shen, Chunhua Shen, Wei Liu, and Heng Tao Shen. Supervised discrete hashing. In *Proc. CVPR*, 2015.

[171] Philip Shilane, Patrick Min, Michael Kazhdan, and Thomas Funkhouser. The princeton shape benchmark. In *Proc. Shape Modeling Applications*, 2004.

[172] Daeyun Shin, Zhile Ren, Erik B Sudderth, and Charless C Fowlkes. 3d scene reconstruction with multi-layer depth and epipolar transformers. In *Proc. ICCV*, 2019.

[173] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proc. ICLR*, 2015.

[174] Arnold WM Smeulders, Marcel Worring, Simone Santini, Amarnath Gupta, and Ramesh Jain. Content-based image retrieval at the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12):1349–1380, 2000.

[175] Olga Sorkine, Daniel Cohen-Or, Yaron Lipman, Marc Alexa, Christian Rössl, and H-P Seidel. Laplacian surface editing. In *Eurographics/ACM SIGGRAPH symposium on Geometry processing*, 2004.

[176] Christoph Strecha, Alexander M Bronstein, Michael M Bronstein, and Pascal Fua. Ldahash: Improved matching with smaller descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(1):66–78, 2012.

[177] Gabriel Taubin. A signal processing approach to fair surface design. In *SIGGRAPH*, 1995.

[178] Bin Fan Yurun Tian. L2-net: Deep learning of discriminative patch descriptor in euclidean space. In *Proc. CVPR*, 2017.

[179] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *Proc. IROS*, 2017.

[180] Jonathan J Tompson, Arjun Jain, Yann LeCun, and Christoph Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *Proc. NIPS*, 2014.

[181] Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *Proc. CVPR*, 2011.

[182] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Proc. CVPR*, 2014.

[183] Jonathan Tremblay, Aayush Prakash, David Acuna, Mark Brophy, Varun Jampani, Cem Anil, Thang To, Eric Cameracci, Shaad Boochoon, and Stan Birchfield. Training deep networks with synthetic data: Bridging the reality gap by domain randomization. In *Proc. CVPR Workshops*, 2018.

[184] Tomasz Trzcinski, Mario Christoudias, Pascal Fua, and Vincent Lepetit. Boosting binary keypoint descriptors. In *Proc. CVPR*, 2013.

[185] Tomasz Trzcinski, Mario Christoudias, and Vincent Lepetit. Learning image descriptors with boosting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):597–610, 2015.

[186] Tomasz Trzcinski and Vincent Lepetit. Efficient discriminative projections for compact binary descriptors. In *Proc. ECCV*, 2012.

[187] Y.-H. Tsai, W.-C. Hung, S. Schulter, K. Sohn, M.-H. Yang, and M. Chandraker. Learning to adapt structured output space for semantic segmentation. In *Proc. CVPR*, 2018.

[188] Hsiao-Yu Tung, Hsiao-Wei Tung, Ersin Yumer, and Katerina Fragkiadaki. Self-supervised learning of motion capture. In *Proc. NeurIPS*, 2017.

[189] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proc. CVPR*, 2017.

[190] Gul Varol, Duygu Ceylan, Bryan Russell, Jimei Yang, Ersin Yumer, Ivan Laptev, and Cordelia Schmid. Bodynet: Volumetric inference of 3d human body shapes. In *Proc. ECCV*, 2018.

[191] Gül Varol, Javier Romero, Xavier Martin, Naureen Mahmood, Michael Black, Ivan Laptev, and Cordelia Schmid. Learning from synthetic humans. In *Proc. CVPR*, 2017.

[192] David Vazquez, Antonio M Lopez, Javier Marin, Daniel Ponsa, and David Geronimo. Virtual and real world adaptation for pedestrian detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(4):797–809, 2014.

[193] Nitika Verma, Edmond Boyer, and Jakob Verbeek. Feastnet: Feature-steered graph convolutions for 3d shape analysis. In *Proc. CVPR*, 2018.

[194] Timo von Marcard, Roberto Henschel, Michael Black, Bodo Rosenhahn, and Gerard Pons-Moll. Recovering accurate 3d human pose in the wild using imus and a moving camera. In *Proc. ECCV*, 2018.

[195] Chengde Wan, Thomas Probst, Luc Van Gool, and Angela Yao. Self-supervised 3d hand pose estimation through training by fitting. In *Proc. CVPR*, 2019.

[196] Ji Wan, Dayong Wang, Steven Chu Hong Hoi, Pengcheng Wu, Jianke Zhu, Yongdong Zhang, and Jintao Li. Deep learning for content-based image retrieval: A comprehensive study. In *Proc. ACM MM*, 2014.

[197] Jingdong Wang, Ting Zhang, Nicu Sebe, Heng Tao Shen, et al. A survey on learning to hash. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[198] Jun Wang, Sanjiv Kumar, and Shih-Fu Chang. Semi-supervised hashing for scalable image retrieval. In *Proc. CVPR*, 2010.

[199] Jun Wang, Wei Liu, Sanjiv Kumar, and Shih-Fu Chang. Learning to hash for indexing big dataa survey. *Proceedings of the IEEE*, 104(1):34–57, 2016.

[200] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proc. ECCV*, 2018.

[201] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *Proc. CVPR*, 2016.

[202] Yair Weiss, Antonio Torralba, and Robert Fergus. Spectral hashing. In *Proc. NIPS*, 2008.

[203] Chung-Yi Weng, Brian Curless, and Ira Kemelmacher-Shlizerman. Photo wake-up: 3d character animation from a single photo. In *Proc. CVPR*, 2019.

[204] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Proc. NeurIPS*, 2016.

[205] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S Yu. A comprehensive survey on graph neural networks. *arXiv preprint arXiv:1901.00596*, 2019.

[206] Fangting Xia, Peng Wang, Liang-Chieh Chen, and Alan L Yuille. Zoom better to see clearer: Human and object parsing with hierarchical auto-zoom net. In *Proc. ECCV*, 2016.

[207] Fangting Xia, Peng Wang, Xianjie Chen, and Alan L Yuille. Joint multi-person pose estimation and semantic part segmentation. In *Proc. CVPR*, 2017.

[208] Fangting Xia, Jun Zhu, Peng Wang, and Alan L Yuille. Pose-guided human parsing by an and/or graph using pose-context features. In *Proc. AAAI*, 2016.

[209] Rongkai Xia, Yan Pan, Hanjiang Lai, Cong Liu, and Shuicheng Yan. Supervised hashing for image retreieval via image representation learning. In *Proc. AAAI*, 2014.

[210] Danfei Xu, Yuke Zhu, Christopher B Choy, and Li Fei-Fei. Scene graph generation by iterative message passing. In *Proc. CVPR*, 2017.

[211] Xing Xu, Fumin Shen, Yang Yang, Heng Tao Shen, and Xuelong Li. Learning discriminative binary codes for large-scale cross-modal retrieval. *IEEE TIP*, pages 2494–2507, 2017.

[212] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Proc. AAAI*, 2018.

[213] Huei-Fang Yang, Kevin Lin, and Chu-Song Chen. Supervised learning of semantics-preserving hash via deep convolutional neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[214] Jianwei Yang, Jiasen Lu, Stefan Lee, Dhruv Batra, and Devi Parikh. Graph r-cnn for scene graph generation. In *Proc. ECCV*, 2018.

[215] Xin Yang and Kwang-Ting Tim Cheng. Local difference binary for ultrafast and distinctive feature description. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(1):188–194, 2013.

[216] Yi Yang and Deva Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *Proc. CVPR*, 2011.

[217] Jianfei Yu and Jing Jiang. Learning sentence embeddings with auxiliary tasks for cross-domain sentiment classification. In *Proc. EMNLP*, 2016.

[218] Joe Yue-Hei Ng, Fan Yang, and Larry S Davis. Exploiting local features from deep networks for image retrieval. In *Proc. CVPRW*, 2015.

[219] Sergey Zagoruyko and Nikos Komodakis. Learning to compare image patches via convolutional neural networks. In *Proc. CVPR*, pages 4353–4361, 2015.

[220] Andrei Zanfir, Elisabeta Marinoiu, and Cristian Sminchisescu. Monocular 3d pose and shape estimation of multiple people in natural scenes-the importance of multiple scene constraints. In *Proc. CVPR*, 2018.

[221] Jure Zbontar and Yann LeCun. Stereo matching by training a convolutional neural network to compare image patches. *JMLR*, 17(1-32):2, 2016.

[222] Shiliang Zhang, Qi Tian, Qingming Huang, Wen Gao, and Yong Rui. Usb: ultrashort binary descriptor for fast visual matching and retrieval. *IEEE TIP*, 23(8):3671–3683, 2014.

[223] Xiong Zhang, Qiang Li, Hong Mo, Wenbo Zhang, and Wen Zheng. End-to-end hand mesh recovery from a monocular rgb image. In *Proc. ICCV*, pages 2354–2364, 2019.

[224] Yu Zhang and Qiang Yang. A survey on multi-task learning. *arXiv preprint arXiv:1707.08114*, 2017.

[225] Ziming Zhang, Yuting Chen, and Venkatesh Saligrama. Efficient training of very deep neural networks for supervised hashing. In *Proc. CVPR*, 2016.

[226] Liang Zheng, Shengjin Wang, and Qi Tian. Coupled binary embedding for large-scale image retrieval. *IEEE TIP*, 23(8):3368–3380, 2014.

[227] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. Conditional random fields as recurrent neural networks. In *Proc. ICCV*, 2015.

[228] Zerong Zheng, Tao Yu, Hao Li, Kaiwen Guo, Qionghai Dai, Lu Fang, and Yebin Liu. Hybridfusion: real-time performance capture using a single depth sensor and sparse imus. In *Proc. ECCV*, 2018.

[229] Kun Zhou, Jin Huang, John Snyder, Xinguo Liu, Hujun Bao, Baining Guo, and Heung-Yeung Shum. Large mesh deformation using the volumetric graph laplacian. In *ACM SIGGRAPH*. 2005.

[230] Xiaowei Zhou, Menglong Zhu, Georgios Pavlakos, Spyridon Leonardos, Konstantinos G Derpanis, and Kostas Daniilidis. Monocap: Monocular human motion capture using a cnn coupled with a geometric prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[231] Xingyi Zhou, Qixing Huang, Xiao Sun, Xiangyang Xue, and Yichen Wei. Towards 3d human pose estimation in the wild: a weakly-supervised approach. In *Proc. ICCV*, 2017.

[232] Hao Zhu, Xinxin Zuo, Sen Wang, Xun Cao, and Ruigang Yang. Detailed human shape estimation from a single image by hierarchical mesh deformation. In *Proc. CVPR*, 2019.