Per Interest-Point Local Descriptors and Detecting People in Artwork

Westlake, Nicholas

Award date:
2019

Awarding institution:
University of Bath

Per Interest-Point Local Descriptors and Detecting People in Artwork

Nicholas Westlake

A thesis submitted for the degree of Doctor of Philosophy

University of Bath

Department of Computer Science

February 2018

Abstract

Matching points between images is difficult because of the many possible variations between images of the same scene or 3D feature. This includes different viewpoints, changing lighting conditions, occlusion and noise. In this work, I demonstrate that the use of a local descriptor which is individual to each interest-point improves matching performance over using one globally.

I propose two different approaches. The first is to use a pooling operation, based on geometric blur, which is individual to each interest-point. This is achieved by estimating how each interest-point will appear in other images through generating synthetic warps. The second is to learn an optimal combination of base-descriptors for each interest-point so as to obtain an optimal descriptor for each interest-point. This is achieved through supervised learning based on having multiple instances of the same interest-point.

Another difficult problem is detecting people in images of artwork, because of the huge variation in the ways people are depicted. This included the media used (e.g. pencil, paint and sculpture) and the range of poses and projections, including Cubism at the extreme. I demonstrate that state-of-the-art CNN based methods yield improved performance if the models are fine tuned on artwork, however their performance on photos is subsequently reduced. This shows that these approaches cannot simultaneously generalise over and perform well on both photos and artwork.

The underlying theme of this dissertation is the proposition that these and other algorithms lack generalisation because their invariance is at too low a level, lacking high level semantic information. For further work, I suggest the incorporation of high level semantic information into interest-point matching and better modelling of the structure of people which is shown to be essential for detecting people across different depictive styles.

Contents

List of Figures

List of Tables

Figure Copyright Attributions

1.1 Left: "Eiffel Tower from Champ-de-Mars, Paris" (https://commons.wikimedia.org/wiki/File:Eiffel_Tower_from_Champ-de-Mars,_14_June_2014.jpg) by Connie Ma is licensed under CC BY-SA 2.0 ;
middle: "Eiffel Tower at night, Paris" (https://commons.wikimedia.org/wiki/File:Eiffel_Tower_at_night,_Paris_20_April_2013.jpg) by Alexander Kachkaev is licensed under CC BY 2.0 ;
right: "Tour Eiffel" (https://commons.wikimedia.org/wiki/File:Tour_Eiffel_(2519307688).jpg) by edwin.11 is licensed under CC BY 2.0 . . . . . .

1.2 Left: "St Paul's" (https://www.flickr.com/photos/kouroff/13134006665/) by Tim Kouroff is licensed under CC BY-SA 2.0 ;
middle: "St. Paul's Cathedral in London with Millennium Bridge in the foreground" (https://commons.wikimedia.org/wiki/File:London_St_Pauls_Cathedral_with_Millennium_Bridge.jpg) by Robert Bauer is licensed under CC BY-SA 3.0 ;
right: "St Paul's in the snow" (https://commons.wikimedia.org/wiki/File:St_Pauls_in_the_snow_-_geograph.org.uk_-_1145221.jpg) by John Lord is licensed under CC BY-SA 2.0 . . . . . .

1.3 Left: Public Domain (http://www.wikiart.org/en/canaletto/the-river-thames-with-st-paul-s-cathedral-on-lord-mayor-s-day)
right: "Sarcophagus of Prince Thutmose's cat" (https://commons.wikimedia.org/wiki/File:St_Pauls_in_the_snow_-_geograph.org.uk_-_1145221.jpg) by Larazoni is licensed under CC BY 2.0 . . . . . .

Figure Copyright Attributions

1.4 Left: Public Domain (https://www.wikiart.org/en/georges-seurat/the-eiffel-tower-1889) ; middle: ©Ivan Generalic (https://www.wikiart.org/en/ivan-generalic/eiffel-tower-1972), used under fair dealing ; right: Public Domain (https://www.wikiart.org/en/pierre-bonnard/eiffel-tower-and-the-seine) . . . . . .

1.5 Images from PASCAL Visual Object Classes (VOC) 2012 dataset: original authors unknown, used under fair dealing . . . . . .

1.6 Images from PASCAL VOC 2007 dataset: original authors unknown, used under fair dealing . . . . . .

1.7 "Bath Abbey" (https://commons.wikimedia.org/wiki/Category:Bath_Abbey#/media/File:Bath_Abbey_02.jpg) by Chilli Head is licensed under CC BY 2.0 . . .

1.8 Left most photos: by myself; right most photos from the PASCAL VOC 2007 dataset, original authors unknown, used under fair dealing; top-left artwork: Public Domain (https://en.wikipedia.org/wiki/The_Creation_of_Adam#/media/File:Creaci%C3%B3n_de_Ad%C3%A1n_(Miguel_%C3%81ngel).jpg) ; bottom-left artwork: Public Domain (https://en.wikipedia.org/wiki/Mona_Lisa#/media/File:Mona_Lisa,_by_Leonardo_da_Vinci,_from_C2RMF_retouched.jpg) ; top-right artwork: Public Domain (https://en.wikipedia.org/wiki/The_Starry_Night#/media/File:Van_Gogh_-_Starry_Night_-_Google_Art_Project.jpg) ; bottom-right artwork: Public Domain (https://en.wikipedia.org/wiki/The_Storm_on_the_Sea_of_Galilee#/media/File:Rembrandt_Christ_in_the_Storm_on_the_Lake_of_Galilee.jpg) . . . .

1.9 Top and bottom left: by myself; top-middle: Public Domain (https://en.wikipedia.org/wiki/Alfred_Stieglitz#/media/File:Stieglitz-Hand.jpg) ; bottom-middle: ©Philippe Halsman, used under fair dealing; top-right: Public Domain (https://en.wikipedia.org/wiki/The_Starry_Night#/media/File:Van_Gogh_-_Starry_Night_-_Google_Art_Project.jpg) ; bottom-right: Public Domain (https://en.wikipedia.org/wiki/The_Storm_on_the_Sea_of_Galilee#/media/File:Rembrandt_Christ_in_the_Storm_on_the_Lake_of_Galilee.jpg) ; random noise: no copyright; synthetic images: ©Anh Nguyen, Jason Yosinski and Jeff Clune (http://www.evolvingai.org/fooling), used under fair dealing . . . . . .

1.10 ©René Magritte (https://en.wikipedia.org/wiki/The_Treachery_of_Images#/media/File:MagrittePipe.jpg), used under fair dealing . . . . . .

1.11 Images from the Photo-Art-50 dataset: original authors unknown, used under fair dealing . . . . . .

1.12 Left: Public Domain (https://www.wikiart.org/en/claude-monet/houses-of-parliament) ; right: "Palace of Westminster at dusk" (https://en.wikipedia.org/wiki/Palace_of_Westminster#/media/File:Palace_of_Westminster,_London_-_Feb_2007.jpg) by David Iliff is licensed under CC BY-SA 2.5 . . . . . .

1.13 ©Google, Getmapping plc, used under fair dealing . .

1.14 ©Google, Getmapping plc, used under fair dealing . .

1.15 ©Google and others, used under fair dealing . . . . .

1.16 "Bath - Roman Baths" (https://commons.wikimedia.org/wiki/Category:Roman_Baths_(Bath)#/media/File:Bath_-_Roman_Baths_(17087314158).jpg) by Paul Stephenson is licensed under CC BY 2.0 . . . . . .

1.17 "Bath - Roman Baths" (https://commons.wikimedia.org/wiki/Category:Roman_Baths_(Bath)#/media/File:Bath_-_Roman_Baths_(17087314158).jpg) by Paul Stephenson is licensed under CC BY 2.0 . . . . . .

1.18 From left to right: Public Domain (https://en.wikipedia.org/wiki/File:William_wilberforce.jpg) ; Public Domain (https://commons.wikimedia.org/wiki/File:Smile2.svg) ; ©Pablo Picasso (https://www.pablopicasso.org/seated-woman-with-green-shawl.jsp), used under fair dealing ; "Bora, Labrador Retriever" (https://en.wikipedia.org/wiki/Labrador_Retriever#/media/File:BoraDK20050331.JPG) by Michael Schreck is licensed under CC BY-SA 3.0 . . . . . .

1.19 Open Government Licence v1.0 . . . . . .

1.20 ©Salvador Dalí (https://www.wikiart.org/en/salvador-dali/the-metamorphosis-of-narcissus), used under fair dealing . . . . . .

1.21 Left: "Seeley Library from the southeast" (https://en.wikipedia.org/wiki/Seeley_Historical_Library#/media/File:History_Faculty_University_of_Cambridge.jpg) by Andrew Dunn is licensed under CC BY-SA 2.0 ; right: "Sydney Operate House" (https://en.wikipedia.org/wiki/Sydney_Opera_House#/media/File:Sydney_Opera_House,_botanic_gardens_1.jpg) by Adam J. W. C. is licensed under CC BY-SA 2.5 . . . . . .

1.22 "Logitech Unifying Receiver" (https://commons.wikimedia.org/wiki/Category:Pareidolias#/media/File:Logitech_Unifying_Receiver,_Logitech_Wireless_Mouse_M235.jpg) by MK2010 is licensed under CC BY-SA 4.0 . . .

1.23 "Bath - Roman Baths" (https://commons.wikimedia.org/wiki/Category:Roman_Baths_(Bath)#/media/File:Bath_-_Roman_Baths_(17087314158).jpg) by Paul Stephenson is licensed under CC BY 2.0 . . . . . .

1.24 Left: "A DeLorean DMC-12 from the front with the gull-wing doors open" (https://en.wikipedia.org/wiki/DeLorean_DMC-12#/media/File:DeLorean_DMC-12_with_doors_open.jpg) by Kevin Abato is licensed under CC BY-SA 3.0 ; right: "1909 Touring - Second-oldest Ford Model T still in existence" (https://en.wikipedia.org/wiki/Ford_Model_T#/media/File:Ford_Model_T_-_Serial_No._220,_Built_December_1908.jpg) by Jackdude101 is licensed under CC BY-SA 4.0 . . . . .

1.25 Left: "Duck" (https://www.flickr.com/photos/ptc24/3309603761/) by Peter Corbett is licensed under CC BY 2.0 ; right: Public Domain (https://commons.wikimedia.org/wiki/File:Setting_sun_at_the_lake.jpg) . . . . . .

1.26 Top left: "Dunsfold Wheels and Wings 2007" (https://commons.wikimedia.org/wiki/Category:Morris_Minor_Traveller#/media/File:Morris_Traveller_(1241914442).jpg) by Allen Watkin is licensed under CC BY-SA 2.0 ; top right: "smart Fortwo cabriolet" (https://en.wikipedia.org/wiki/Smart_(marque)#/media/File:Smart_fortwo_52_mhd_cabrio_-_Flickr_-_David_Villarreal_Fern%C3%A1ndez_(16).jpg) by Hohum is licensed under CC BY-SA 2.0 ; bottom left: "British Rolls Royce 1920 Mk1 Armoured Car at Bovington Tank Museum" (https://en.wikipedia.org/wiki/Rolls-Royce_Armoured_Car#/media/File:Rolls_Royce_1920_Mk1_1_Bovington.jpg) by Hohum is licensed under CC BY 3.0 ; bottom right: "f1" (https://en.wikipedia.org/wiki/McLaren_MCL32#/media/File:2017_British_Grand_Prix_(35127471353).jpg) by Stephen Grimes is licensed under CC BY-SA 2.0 . . . . . .

1.27 Top left: "Women's high jump" (https://commons.wikimedia.org/wiki/Category:High_jump#/media/File:Womens_high_jump_3.jpg) by ScottRay is licensed under CC BY 2.0 ; top right: "A horse free jumping." (https://en.wikipedia.org/wiki/Free_jumping#/media/File:Dirkhan.jpg) by Artur Baboev is licensed under CC BY-SA 3.0 ; bottom left: Public Domain (https://en.wikipedia.org/wiki/Jumping#/media/File:Tursiops_truncatus_01.jpg) ; bottom right: "An Emperor Penguin (Aptenodytes forsteri) in Antarctica jumping out of the water." (https://commons.wikimedia.org/wiki/File:Penguin_in_Antarctica_jumping_out_of_the_water.jpg) by Christopher Michel is licensed under CC BY 2.0 . . . . . .

Figure Copyright Attributions

3.3 Image from K. Mikolajczyk's "Feature Detector Evaluation Sequences", used under fair dealing . . . . . .
3.4 Image from K. Mikolajczyk's "Feature Detector Evaluation Sequences", used under fair dealing . . . . . .
3.5 Image from K. Mikolajczyk's "Feature Detector Evaluation Sequences", used under fair dealing . . . . . .
3.6 "Bath Abbey, City of Bath, England" (https://commons.wikimedia.org/wiki/File:Bath_Abbey,_City_of_Bath,_England..jpg) by JKMMX is licensed under CC BY-SA 3.0 . . . . . .
3.7 Created by myself . . . . . .
3.8 Created by myself . . . . . .
3.9 Created by myself . . . . . .
3.10 Created by myself . . . . . .
3.11 Created by myself . . . . . .
3.12 Diagram created by myself, image Public Domain . . .
3.13 Created by myself . . . . . .
3.14 Created by myself . . . . . .
3.15 Created by myself . . . . . .
3.16 Created by myself . . . . . .
3.17 Created by myself . . . . . .
3.18 Created by myself . . . . . .
3.19 Created by myself . . . . . .
3.20 Created by myself . . . . . .

4.1 Images from the DTU Robot Image dataset, used under fair dealing . . . . . .
4.2 Created by myself . . . . . .
4.3 Created by myself . . . . . .
4.4 Created by myself . . . . . .
4.5 Created by myself . . . . . .
4.6 Created by myself . . . . . .
4.7 Created by myself . . . . . .
4.8 Created by myself . . . . . .
4.9 Images from DTU dataset, used fair dealing . . . . . .
4.10 Images from DTU dataset, used fair dealing . . . . . .
4.11 Created by myself . . . . . .
4.12 Images from Oxford Buildings dataset, used under fair dealing . . . . . .

5.23 In reading order, Public Domain (https://en.wikipedia.org/wiki/Bayeux_Tapestry#/media/File:Odo_bayeux_tapestry.png) ; Public Domain (https://commons.wikimedia.org/wiki/File:Canterbury_Cathedral_020_Poor_Mans_Bbible_Window_01_adj.JPG) ; Public Domain (https://commons.wikimedia.org/wiki/File:Gudea_of_Lagash_Girsu.jpg) ; "Statuette Mambia Nigéria" (https://commons.wikimedia.org/wiki/File:Statuette_Mambia_Nig%C3%A9ria.jpg) by Siren-Com is licensed under CC BY-SA 3.0 ; M C Escher 1953, used under fair dealing; Public Domain (https://commons.wikimedia.org/wiki/File:William_Hogarth_-_Absurd_perspectives.png) ; Public Domain (https://commons.wikimedia.org/wiki/File:Korean_art-Donggwoldo-Changdeokgung_and_Changgyeonggung-Dong-A_University-01.jpg) ; Public Domain (https://en.wikipedia.org/wiki/Reverse_perspective#/media/File:Italo-Byzantinischer_Maler_des_13._Jahrhunderts_001.jpg) . . . . . .

List of Abbreviations

AdaBoost Adaptive Boosting

AP average precision

ROC-area area under an ROC curve

BoW bag-of-words

CKN convolutional kernel network

CNN convolutional neural network

COCO the "common objects in context" dataset (Lin et al., 2014)

DAG directed acyclic graph

DPM Deformable Part-based Model

EM expectation–maximisation

EMD Earth Mover's Distance

FPR false positive rate

FS Fixed sigma

FSR Fixed sigma ratio

GPU graphics processing unit

HOG Histograms of Oriented Gradients

HOS heuristic over-segmentation

HSL hue, saturation and luminance

HSV hue, saturation and value

ILSVRC the ImageNet Large Scale Visual Recognition Challenge (Russakovsky et al., 2015)

IoU intersection over union

LCN local contrast normalisation

LRN local response normalisation

LSVM latent-SVM

LUV CIE 1976 (L* u* v*) colour space

MAP maximum a posteriori

MLE maximum likelihood estimation

the MNIST database the *MNIST database of handwritten digits* (LeCun, Bottou, Bengio and Haffner, 1998)

NFL "no free lunch"

NORB the NYU Object Recognition Benchmark dataset (LeCun, F.J. Huang and Bottou, 2004)

OTS off-training set

PCA principal component analysis

PIP per interest-point

PMK Pyramid Match Kernel

PR Precision-Recall

RANSAC random sample consensus

RBF radial basis function

R-CNN "regions with CNN features"

RDA regularised dual averaging

ReLU rectified linear unit

RGB red, green and blue

RNN recurrent neural network

ROC receiver operating characteristic

ROI region of interest

RPN region proposal network

SGD stochastic gradient descent

SLAM simultaneous localisation and mapping

SPP spatial pyramid pooling

SSD sum of squared differences

SSVM structural SVM

SVM support vector machine

tf-idf "term frequency-inverse document frequency"

TPS thin plane spline

VOC Visual Object Classes

1 Introduction

Sometimes the same thing looks very different. Consider standing in the same place and looking at the same building: perhaps the base of the Eiffel Tower or the entrance to St Paul's Cathedral. The appearance will change over time. This may be due to changes in the weather, changes in the position of sun or the presence of other light sources. This may also be due to the interaction of other items, whether visible in the scene or casting a shadow on the scene. Despite these changes, the identity of the building remains the same. However, its description might change: perhaps one might add the label "at night" or "at dawn".

As one walks around and views the building from a different position or orientation, the appearance changes further still. Some parts of the building appear to be located in different places relative to us. Other parts of the building disappear while new parts of the building become visible. In spite of this, however, the identity of the building remains the same. However, one might use a different description for different

Figure 1.1: Left and middle: the same view of the Eiffel Tower at different times of the day; right: a different view of the Eiffel Tower

parts of the building: the front, the rear, the inside, the base, the top, the nave, the transepts.

Matching is an important tool for Computer Vision problems. The aim of matching is to discern whether two instances of an object or feature have the same identity: they are different instances of the same thing. The ability to match locations between 2D images enables the building of a 3D reconstruction and enables tracking changes through time, whether caused by movement or change in appearance.

However, matching is difficult because of the large amount of variation in direct visual appearance. The two instances being matched may be far from identical. Perhaps the two instances are the same scene, photographed at two different angles. A successful matching algorithm must allow for differences which do not prevent two instances being the "same" for the given task. However, in doing so it must reject differences which results in two non-matching instances, otherwise false matches will result. This makes matching a difficult Computer Vision problem in its own right.

Another task is classification, the aim of which is discern to which class, or classes, an images belongs. Yet another task is object detection, which requires not only classifying objects but finding where they are within an image. The variation within a same class can be huge. Consider four classes: cars, cats, buses and dogs. A successful classification algorithm must be able to handle the vast variation within these classes: the many different breeds of cats and dogs and the many different models of buses and cars. Yet it must be able to discern between a car and bus, a cat and a dog. Though the differences between them may not seem subtle to human observers, who excel at such classification, this task remains difficult to implement

Figure 1.2: Three photos of St Paul's Cathedral in different weather conditions; left: in sun; middle: in rain; right: in snow


Figure 1.3: Left: "The River Thames With St. Paul's Cathedral" by Canaletto; Right: Egyptian sarcophagus of a cat

with computers.

In all these problems, there are some properties which do not affect the identity and some which do. These properties vary with the particular matching or classification task: the presence of rain does not change the identity of a building, though it could change the description of the weather conditions. For optimal performance, Computer Vision algorithms for these tasks must be *invariant* to properties which do not change the identity for given the task. Yet, they must remain *discriminant* to properties which *do* change the identity.

On top of all the properties mentioned, there is another property: depiction. As well as natural images, formed by taking photographs of natural scenes, there are other ways objects can be depicted, such as paintings and line drawings. Yet the depiction does not change the identity. Canaletto's painting of St Paul's Cathedral is not a natural image, however it is still identifiable as St Paul's cathedral; a sarcophagus of a cat is not a natural image but it is still a cat. Likewise many artists have produced different paintings of the Eiffel Tower, many of which differ greatly from natural images, even exchanging straight lines for curves. However these paintings all share the same identify: the Eiffel Tower.

1.1 Theses

In this dissertation, I show a portfolio of research work which supports the following proposition:

> The invariance of state-of-the-art algorithms is at too low a level for generalised interest-point matching and generalised object detection.

Chapters 3 and 4 proves the following thesis:

Figure 1.4: Left: "The Eiffel Tower" by Georges Seurat; middle: "Eiffel Tower" by Ivan Generalic; right: "Eiffel Tower And The Seine" by Pierre Bonnard

> A local descriptor can be optimised per interest-point rather than globally to improve matching accuracy.

Chapter 5 proves the following thesis:

> It is not yet possible to train or tune state-of-the-art object detectors to simultaneously perform optimally on artwork and photos.

Some of the terms require further definition and are outlined in the sections identified in the following table:

Term	Section	Page
generalised interest-point matching	1.3	37
generalised object detection	1.3	37
invariance	1.5	47
low level	1.4	40
interest-point matching	1.2.3	33
local descriptor	1.2.3.2	36
object detector	1.2.2	32
artwork and photos	1.3	37

1.2 Image Classification, Object Detection and Interest-Point Matching

I begin by describing the Computer Vision tasks, image classification, object detection and interest-point matching. For each of these tasks, I assume that optimal performance occurs when the computer matches the opinion of one or more or a consensus of humans supervisors on new or previously unseen examples. The objective of maximising performance on new or unseen examples, exclusively, matches the literature for supervised learning (Bishop, 2006; Valiant, 1984; Weiss and Kulikowski, 1991). As noted in Ginosar et al. (2014), human supervisors can often disagree on annotations; however the consensus of the majority forms the target for optimal performance.

1.2.1 Image Classification

Image classification is the act of assigning a class or label to an image. Consider a digital image, as a function, $I(\mathbf{x})$,

$$I \colon \mathbb{P} \to \mathbb{N}_0^N, \tag{1.1}$$

which maps 2D coordinates, as a variable, $\mathbf{x} = \begin{bmatrix} x & y \end{bmatrix}^\top$ to pixels composed of tuples, length N, e.g. a single intensity value ($N = 1$), red, green and blue (RGB) ($N = 3$) or another tuple of values. Since images have a finite width, w, and height, h, $I(\mathbf{x})$ is defined only for a subset of heights and widths, all natural numbers,

$$\mathbb{P} \subset \mathbb{N}^2 \colon 0 < x \leq w, 0 < y \leq h. \tag{1.2}$$

Consider an image classifier as an operator,

$$C\{I(\mathbf{x})\} = \theta, \tag{1.3}$$

Figure 1.5: Images from the PASCAL VOC classification challenge

which operates on an image, $I(\mathbf{x})$, and outputs a class, $\theta \in \Theta$. Alternatively consider the classifier as a function or mapping from the set of all possible images, $\mathcal{I}$, to the set of possible classes, Θ:

$$C \colon \mathcal{I} \to \Theta \tag{1.4}$$

The class could represent the dominant object in an image, e.g. person, car, cat or dog. Alternatively, a class could represent a more abstract label, such as "landscape" or "building".

Figure 1.5 shows some examples of the PASCAL Visual Object Classes (VOC) classification challenge (Everingham, Van Gool et al., 2012), by way of example. In this challenge, the task is to identify the presence or lack of each class in an image. Therefore each image can have multiple classes. In the above framework, this could correspond to having many classifiers, $C\{I(\mathbf{x})\}$, which output either positive or negative for each class of object in the challenge. Alternatively, one could modify (1.3) to return a set of classes, $\{\theta_1, \theta_2, \ldots\} \subset \Theta$.

1.2.2 Object Detection

Object detection involves not just identifying the presence of objects, as in classification. Instead, object detection involves localising each object instance. An object detector may have different outputs including the following:

1. The centre of the object
2. A rectangular box, the bounding box, whose sides are vertical and horizontal and which covers the entire object with minimal area (see Figure 1.6 for examples)
3. A binary image mask, $\mathcal{Z}_2^{h \times w} : \mathcal{Z}_2 = \{0, 1\}$, where h and w are the height and width, respectively, of the input image: the mask indicates which pixels form the object; however, this would typically be considered segmentation on top of detection (Romera-Paredes and Torr, 2016).

Consider an object detector, which outputs a rectangular bounding box (item 2 above) as an operator,

$$D\{I(\mathbf{x})\} = (\theta, \mathbf{x}_{\mathrm{TL}}, \mathbf{x}_{\mathrm{BR}}), \tag{1.5}$$

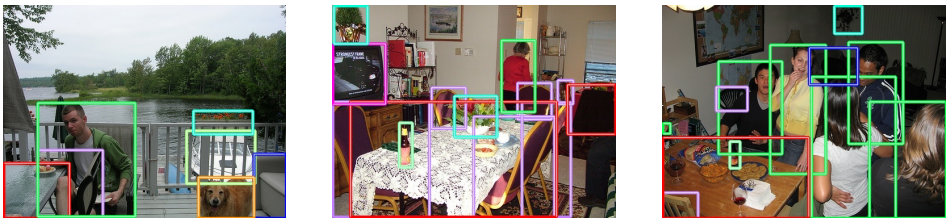

Figure 1.6: Images from the PASCAL VOC detection challenge with bounding box labellings; each class has a different colour

which operates on an image, $I(\mathbf{x})$, and outputs a tuple, $(\theta, \mathbf{x}_{\mathrm{TL}}, \mathbf{x}_{\mathrm{BR}})$, containing a class, $\theta \in \Theta$, and the coordinates, $\mathbf{x}_{\mathrm{TL}} \in \mathbb{P}$ and $\mathbf{x}_{\mathrm{BR}} \in \mathbb{P}$. These coordinates represent the top-left and bottom-right coordinates, respectively, of the detection bounding box, whose sides are horizontal and vertical. The coordinates lie within the image, i.e. the space, $\mathbb{P}$ (see (1.2)). More generally, the object detector may return a set of tuples,

$$\left\{(\theta_1, \mathbf{x}_{\mathrm{TL}1}, \mathbf{x}_{\mathrm{BR}1}), (\theta_2, \mathbf{x}_{\mathrm{TL}2}, \mathbf{x}_{\mathrm{BR}2}), \ldots, (\theta_N, \mathbf{x}_{\mathrm{TL}N}, \mathbf{x}_{\mathrm{BR}N})\right\}, \quad (1.6)$$

corresponding to multiple object instances.

Figure 1.6 shows some examples of the PASCAL Visual Object Classes (VOC) detection challenge (Everingham, Van Gool et al., 2012). In this challenge, the task is to identify and locate all instances of a set of classes within an image. An ideal object detector should return bounding boxes which match those in Figure 1.6 and with the correct class labels (corresponding to colours in the figure).

1.2.3 Interest-Point Matching

Many Computer Vision tasks require a set of matching points between images, for example 3D reconstruction (Agarwal et al., 2009; Brown and Lowe, 2002), image stitching (Brown and Lowe, 2007), image search (Philbin et al., 2007) and object detection (Brown and Lowe, 2005; Lowe, 1999). Though it would be possible, in theory, to match every pixel with every other pixel across images, this is not common practice as a first step due to inefficiency.

Likewise, it would, in theory, be possible to use the outputs of an object detector, however this would require a sufficient number of object instances for a given class to exist in each image, to generate a sufficient number of matching points between images. Moreover,

Figure 1.7: ORB interest-points extracted from an image of Bath Abbey: the interest-points are drawn so as to show their relative size and orientation.

this task usually requires matching of identical instances rather than identical objects. Hence an object detector would not be appropriate. Finally, many objects often move around in the 3D world, e.g. people and vehicles, while background elements such as buildings and landscape do not. The latter elements are more useful for matching between images as they provide a constant reference but these may not be identified by object detectors because they form a continuum rather than a single object instance.

One alternative is to extract a subset of pixel locations in each image which are expected to be useful for matching between images. These locations are known as *interest-points* and are detected using an interest-point extractor. (Interest-points can also be referred to as "features". However due to the potential for confusion, I always refer to these as interest-points.) Figure 1.7 shows an example of interest-points extracted from an image. The use of interest-points represents a technique for data reduction, reducing the huge amount of data contained in a single image into a much smaller amount of pertinent

information for further processing (e.g. image registration).

1.2.3.1 Interest-Points and Interest-Point Extractors

An interest-point extractor processes an image and returns a number of interest-points, consisting of pertinent locations (and possibly other parameters such as rotation and scale) in the image. Consider an interest-point extractor as an operator,

$$D_{IP}\{I(\mathbf{x})\} = \{\mathbf{p_1}, \mathbf{p_2}, \ldots, \mathbf{p_N}\}, \tag{1.7}$$

which operates on an image, $I(\mathbf{x})$, and outputs a set of interest-points, $\mathbf{p_n} \in \mathcal{P}$. The use of the term *extractor* rather than *detector* follows from the argument that interest-points do not exist *a priori* in the image (Tuytelaars and Mikolajczyk, 2008). Hence no *detection* takes place and there are no missed detections; instead, they are extracted from the image based on an algorithm. This is in contrast to objects which *do* exist *a priori* and, therefore, are either detected in an image or result in missed detections.

An interest-point must be well defined, often mathematically, however the definition varies depending on the extractor. At the minimum, each interest-point, $\mathbf{p_n}$, must be parametrised by a pixel location within the image, $\mathbf{x} \in \mathbf{p_n} \cap \mathbb{P}$. Since the purpose of interest-points is for matching the same points between difference images, the definition of an interest-point must be independent of location within an image: this is known as *location covariance*. (However, the interest-point extractor may not be able to extract interest-points too close to an edge of an image, as a practical limitation.)

In addition, an interest-point may also parameterise a region or neighbourhood (Tuytelaars and Mikolajczyk, 2008) surrounding $\mathbf{x}$. As an example, each interest-point could contain a scale parameter, s, which indicates the size of a region centred on the location, $\mathbf{x}$. The reasons for this will become apparent later.

An interest-point should have the following attributes (Farinella, Battiato and Cipolla, 2013; Tuytelaars and Mikolajczyk, 2008):

distinctiveness In order to permit the matching of interest-points between images, each interest-point should lie on or represent a distinct part of the image. As an example, areas of sky tend not to be distinct between photographs.

ubiquity Interest-points should be extractable, in sufficient quantities, from any image used for any task requiring interest-points.

repeatability Where different images contain an identical region, perhaps photographs of the same points in 3D photographed from a different viewing angle, the same interest-point should be extracted.

In addition, the interest-point extractor should ideally be efficient.

1.2.3.2 Local Descriptors

Matching interest-points between images can be achieved by matching the neighbourhood of the image local to or parameterised by the interest-point. This region, used to calculate the local descriptor, is known as the *support region* (Mikolajczyk and Schmid, 2005; Mikolajczyk, Tuytelaars et al., 2005). Consider a local descriptor extractor as an operator,

$$\mathbf{E}_{LD}\{I(\mathbf{x}), \mathbf{p}\} = \mathbf{d}, \tag{1.8}$$

which operates on an image, $I(\mathbf{x})$, and outputs a local descriptor, $\mathbf{d} \in \mathbb{R}^N$, a vector of length, N, for a given interest-point, $\mathbf{p} \in \mathcal{P}$. In general, the local descriptors exist in the same normed vector space, $\left(\mathbb{R}^N, \|\cdot\|\right)$, where the norm, $\|\cdot\|$, allows comparison between two local descriptors, by calculating the distance between two descriptors, $\mathbf{d}_1, \mathbf{d}_2 \in \mathbb{R}^N$ as

$$d(\mathbf{d}_1, \mathbf{d}_2) = \|\mathbf{d}_1 - \mathbf{d}_2\|. \tag{1.9}$$

In Chapter 4, however, I demonstrate the use of different distance functions for each interest-point.

There are many possible local descriptor extractors. Perhaps the simplest is to extract a square region centred on the location of an interest-point, $\mathbf{x_n} \in \mathbf{p_n}$, with horizontal and vertical sides of length, l. The intensity values of each pixel within the square could then be rearranged to form a vector for use as a local descriptor. Alternatively, if the interest-point also includes a scale parameter, $s_n \in \mathbf{p_n}$, a square region of size, $s_n l$, could be used. As a result, scaling of the image would have no effect on the descriptor: such a descriptor is *scale invariant* providing that the interest-point extractor is *scale covariant*.

In practice, neither method alone is robust enough for general matching tasks due to insufficient invariance to the differences between images, which are explored in greater detail in Section 1.5. In addition to

invariance, the local descriptor must maintain its discriminative power so as not to result in the matching of interest-points which should not match. This makes interest-point matching a challenging problem.

1.3 Generalised Object Detection and Generalised Interest-point Matching

The majority of existing Computer Vision algorithms typically perform well for a limited subset of problems. In the case of object detection, existing algorithms typically perform well on photos and other, photo-realistic, images. In the case of interest-point matching, existing algorithms typically also perform well on photographs, and for changes in viewpoint or viewing conditions rather than changes over time, e.g. building construction progress. However, the universe of possible images includes more than photographs. One interpretation is to consider the universe of possible images as including natural images (photos) and non-natural images (artwork), as in Figure 1.8.

Westlake, Cai and Hall (2016) argue that, though linguistically convenient, this is a false dichotomy. A more accurate interpretation is to allow photos and artwork to intersect, as in Figure 1.9, to cover fine art photos such as Alfred Stieglitz's "The Hand of Man" and Philippe Halsman "Marilyn Monroe". It is worth noting that the boundary for artwork is subjective and based on the observer's frame of reference.

In addition to photos and artwork, and the intersection between these two groups, there are images which fall into neither. This includes random noise images and images which are specifically designed to fool state-of-the-art object detectors (Nguyen, Yosinski and Clune, 2015).

While photographs of actual objects may be more abundant than other images, they form only one depictive style. Other styles include paintings, drawings, sculptures and symbols. A famous example of this is "The Treachery of Images" by René Magritte (see Figure 1.10), an oil panting which represents a pipe. The painting can therefore be recognised as a pipe, even if it can not be smoked as a pipe. In addition, Figure 1.11 shows five images of bicycles: only one is a photograph of a bicycle however the rest are clearly identifiable as bicycles.

Generalised object detection refers to the ability to detect objects

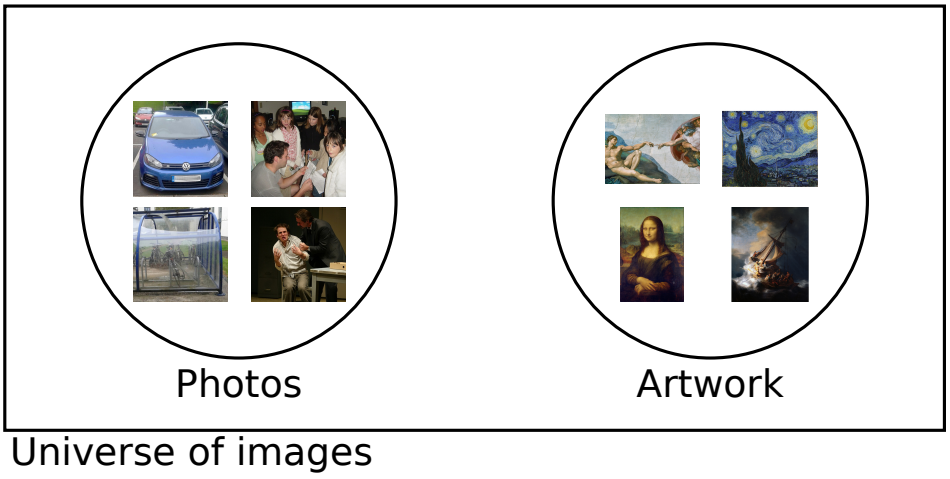

Figure 1.8: In this interpretation, the "Universe of Images" includes two non-overlapping subsets: photos and artwork.

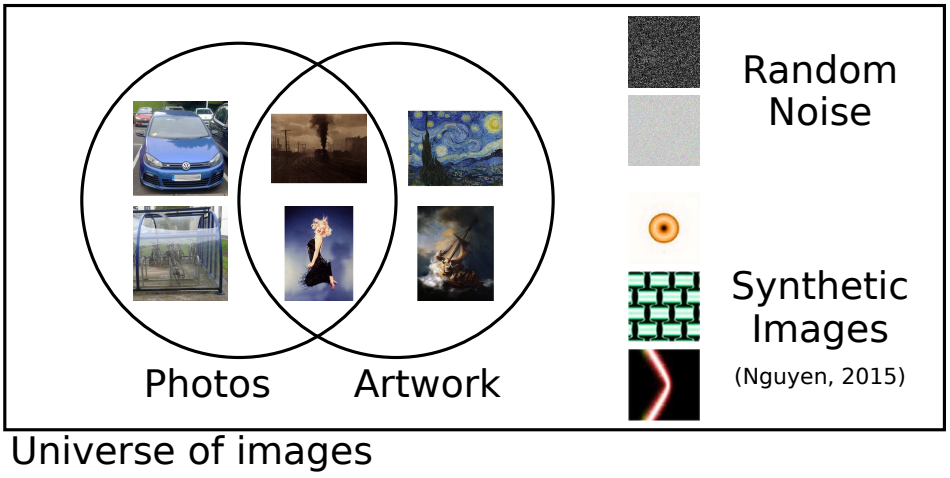

Figure 1.9: In this interpretation, the "Universe of Images" includes two overlapping subsets: photos and artwork, as well as images which do not fall into either.

Figure 1.10: "The Treachery of Images" by René Magritte: The text says "this is not a pipe"—indeed it is an oil paining which represents a pipe and can be recognised as a pipe, even if it can not be smoked as a pipe.

Figure 1.11: Five images of a bicycle: though only the leftmost is a photo, the rest are still identifiable as bicycles

Figure 1.12: Left: Claude Monet's "Parliament in London—Stormy day, 1904"; right: photo of the river front of the Palace of Westminster at dusk; although one is an impressionist painting and the other a photo, it is possible to match points between the two images

no matter how they are depicted. Similarly, generalised interest-point detection refers to the ability to extract and match interest-points between two images, even if they have different depictions. These are instances of the *cross-depiction problem* (Hall et al., 2015). Figure 1.12 shows an example of this: points between the impressionist painting and the photograph have been matched by hand between the two images, a task possible for a human annotator in spite of the different depictive types.

1.4 From Low to High Level

There is no universal definition of low and high level in Computer Vision, or of what constitutes different level of vision. A useful analogy might be viewing the ground from above, from a low level to a high level. At a low level, one can observe minute details such as individual trees and buildings (see Figure 1.13). At a higher level, one can only observe areas of earth with less detail, e.g. fields and built-up areas (see Figure 1.14). At an even higher level, one would only be able to observe different terrain, e.g. mountains, sea and plains (see Figure 1.15).

1.4.1 Closeness to Pixels

The basic building block of Computer Vision is the pixel, with images formed from combinations of them in a regular grid. As a result, pixels form the lowest level of vision. A linear sum or convolution over pixels in the image results in a new image, containing low level features (see Section 1.6.1), for example images showing the oriented gradients of the image as in Figure 1.17. Although this operation still remains close to the pixels, and hence is still low level, the result is at a higher level then the pixels; multiple images could produce the same gradient images.

Another example of low level features includes appearance-based edges, such as those generated using the Canny edge detector (Canny, 1986). Figure 1.16 shows edges extracted using the Canny detector for the same image. Since this resulted in further processing of the gradient images, it is at a higher level than them.

The mid level includes details about geometry and motion such as object depths, surface-normals, motion trajectories and primitive

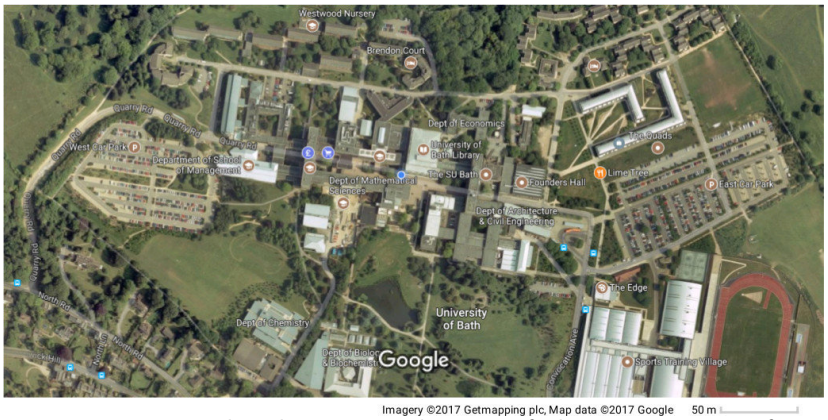

Figure 1.13: Low-level: an aerial view of the University of Bath

Figure 1.14: Higher level: an aerial view of Bath

Figure 1.15: Even higher level: an aerial view of UK

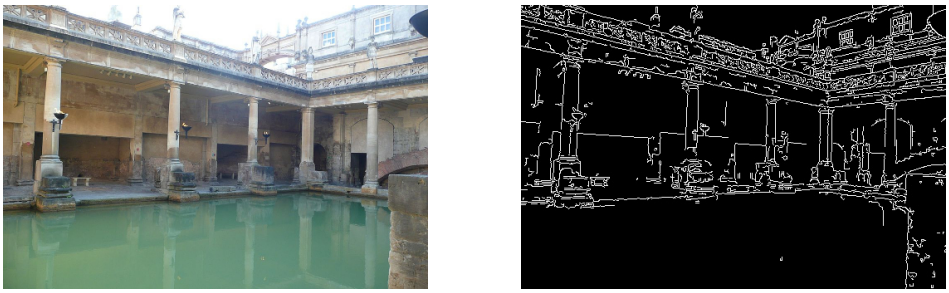

Figure 1.16: Left: A photo of the Roman Baths, Bath; right: the result of running the Canny edge detector on the photo

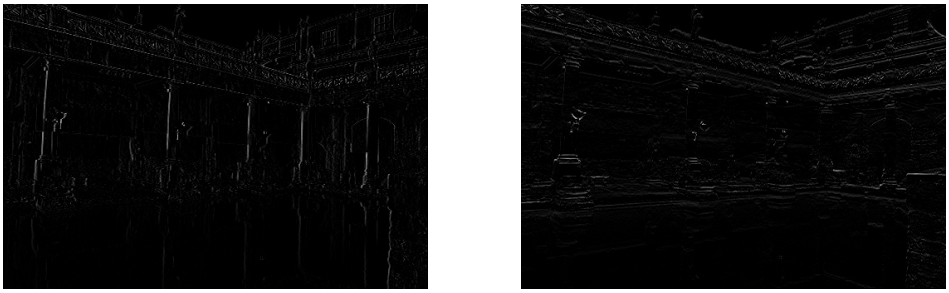

Figure 1.17: The result of converting to greyscale the photograph in Figure 1.16 followed by convolution with a kernel: left convolution with $\begin{bmatrix}-1 & 0 & 1\end{bmatrix}$ as the kernel; right: convolution with $\begin{bmatrix}-1 & 0 & 1\end{bmatrix}^\top$ as the kernel

object structures such as wheels and faces. These are further removed from the pixels, requiring more processing such as successive rounds of non-linear operations and convolution of low level features. The mid level results in more interpretation of the image, however falls short of a semantic interpretation. Figure 1.18 demonstrates an example of this: at low sensitivity (first row) the face detector only detects a single face; at high sensitivity, the detector also detects a second face, with many false positives. Even at low sensitivity, the detector fails to identify the "smiley" face and the dog face, suggesting it falls short of semantic interpretation.

The high level involves a semantic interpretation. This would includes object detectors, which apply a semantic label to part of an image, such as "a person". However, it would also include other semantic details including "looking to the right", "sitting down", "angry", etc. High levels of Computer Vision would also include association

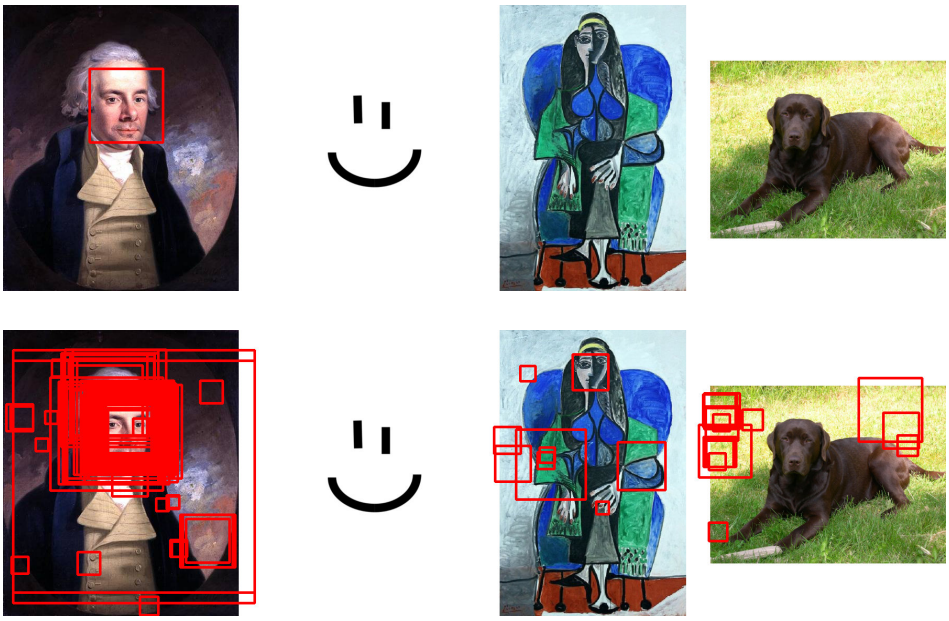

Figure 1.18: A face detector forming an example of mid level vision: the face detections are shown as red rectangles. The top row shows detectors when run with high sensitivity and the bottom row shows detections when run with low sensitivity. The images are as follows from left to right: "Portrait of William Wilberforce" by Karl Anton Hickel; a "smiley", "Seated Woman with Green Shawl" by Pablo Picasso and a photograph of a Chocolate Labrador

between symbols and real-life objects: an example of this would be the semantic interpretation of road signs (see Figure 1.19).

The highest level includes cases in which an object resembles another, for example a cloud resembling a person. Though the class of the object is clear, it resembles another object. The resemblance may be intentional or a case of pareidolia—perceiving a pattern in randomness. One famous example is "The Metamorphosis of Narcissus" by Salvador Dalí (see Figure 1.20), in which the head of Narcissus also appears as an egg from which a flower has sprouted. Figure 1.21 shows two examples in buildings: the Seeley Historical Library (with the appearance of a book) and the Sydney Opera House (with the appearance of sails). Figure 1.22 shows an example of pareidolia in a product.

There is a clear trend between the lower and higher levels of Com-

Figure 1.19: Although all the objects in road signs are black and white silhouettes, the semantic meaning is clear.

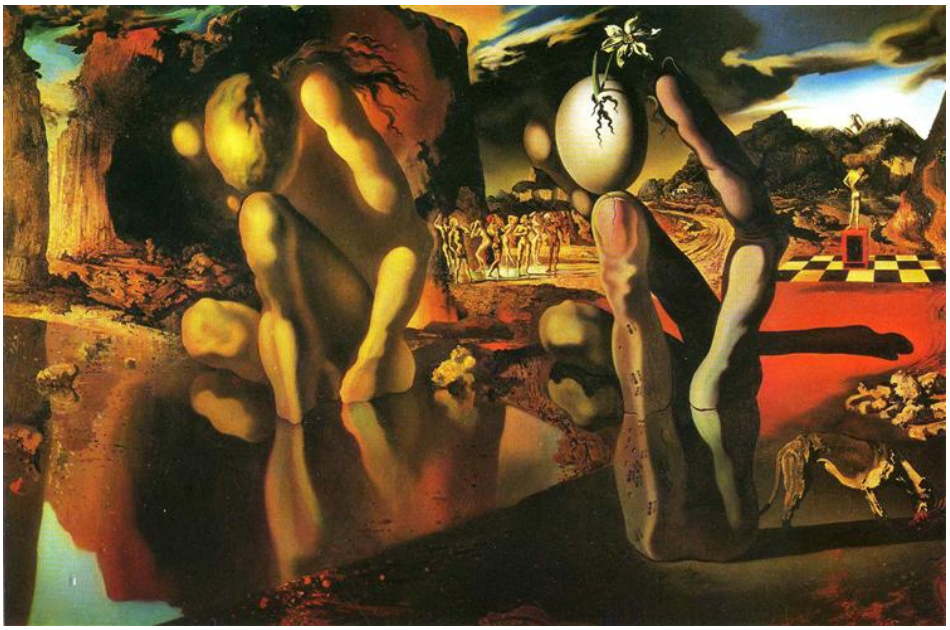

Figure 1.20: "The Metamorphosis of Narcissus" by Salvador Dalí

puter Vision. However, the boundaries between levels of Computer Vision, from low level to high level are subjective. Importantly, the level of a feature ought to be reflect practice rather than theory. An edge which corresponds to a true object boundary, therefore resulting in a semantic interpretation of the image, ought to be considered high level, while an edge based on appearance which cannot distinguish between a shadow and an object boundary ought to be considered low level. Figure 1.23 shows a high level segmentation performed using Badrinarayanan, Kendall and Cipolla (2017). The algorithm is highly accurate but confuses parts of the reflection in the pool for building, suggesting that it is not sufficiently high level in practice to identify the distinction between the actual building and mere reflection.

Similarly, an object detector which can only detect cars from a

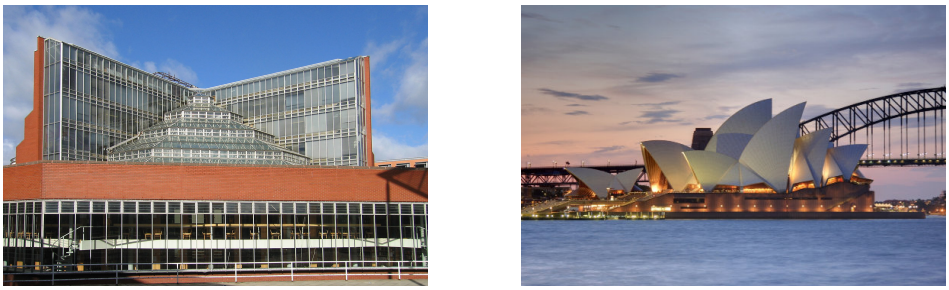

Figure 1.21: Left: Seeley Historical Library, Cambridge; right: Sydney Opera House

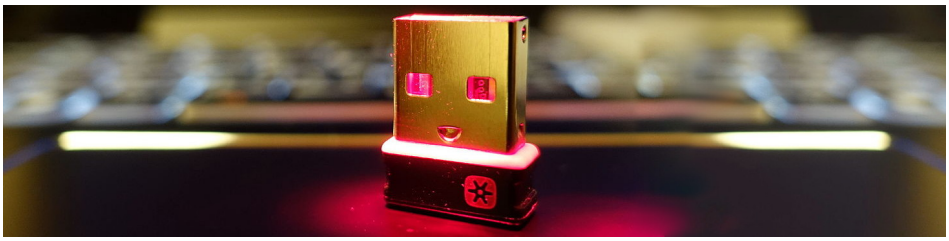

Figure 1.22: A face is apparent in the Logitech Unifying Receiver

Figure 1.23: Left: The previously seen photograph of the Roman Baths, Bath; right: the result of a semantic segmentation of the photo using Badrinarayanan, Kendall and Cipolla (2017) (blue corresponds to sky, red corresponds to building and yellow corresponds to pavement)

Figure 1.24: Left: a DeLorean DMC-12; right: a 1909 Ford Model T

frontal view ought to be considered lower level than one which can identify cars from all angles. Furthermore, an object detector which can detect both a DeLorean DMC-12 and Ford Model T (see Figure 1.24) ought to be considered higher level than one which can only detect a smaller range of cars, perhaps modern, conventional cars.

1.4.2 Vision as "inverse graphics"

Another principle is to consider Computer Vision as "inverse Graphics". Under this principle, high level Computer Vision corresponds to the initial (application) stage of the Computer Graphics pipeline (Shirley and Marschner, 2009): beginning with the configuring of objects based on their parameters, which could relate to high level semantics such as a "person sitting down" showing a "happy face".

The next stage, involves the transformation of the geometry of these objects, and the primitives which form them, relative to each other within the "world", which then includes the calculation of depth and surface normals. This stage correspond to the mid level of computer-vision: it no longer concerns the semantics but is not yet concerned with intensity values at each pixels. It involves the configuration of lighting and texture, which can be controlled independently of the geometry.

The final stage, rasterisation, results in fragments which corresponds to the low level Computer Vision. The fragments are close to forming pixels, missing only the fragment processing and blending steps to result in actual pixels. This stage includes information about the colour and texture.

1.5 Invariance

I will use the term *invariance* in both the mathematical sense and the more general sense. The mathematical sense is defined as follows (Oxford English Dictionary, 2018):

> The character of remaining unaltered after a linear transformation; the essential property of an invariant. Hence applied to a similar property with respect to any transformation or operation.

The general sense is defined as follows (Oxford English Dictionary, 2018):

> The property of remaining unaltered or of being the same in different circumstances; an instance of this, an invariant.

1.5.1 Translation Invariance

Invariance can be expressed in both mathematical and general terms: here, I demonstrate using *translation invariance* as an example. I begin with a demonstration using mathematical terms.

As an extension to consideration of a digital image as a function, $I(\mathbf{x})$ (1.1), consider a digital image, $J(\mathbf{x}, \mathbf{o})$, which contains an object centred on $\mathbf{o}$ as a function,

$$J \colon \mathbb{P}, \mathbb{P} \to \mathbb{N}_0^N, \tag{1.10}$$

which maps the image coordinates as a variable, $\mathbf{x}$, and the object centre as a parameter,

$$\mathbf{o} = \begin{bmatrix} o_x & o_y \end{bmatrix}^\top, \tag{1.11}$$

to pixel intensity values.

If a classifier, whose task is to classify the object centred on $\mathbf{o}$, is *translation invariant*, that is, invariant to a translation by a vector $\mathbf{t}$,

$$\mathbf{T} \colon \mathbb{P} \to \mathbb{P} \tag{1.12}$$

$$\begin{bmatrix} o_x \\ o_y \end{bmatrix} \mapsto \begin{bmatrix} o_x + t_x \\ o_y + t_y \end{bmatrix}, \tag{1.13}$$

applied to the object at $\mathbf{o}$, then

$$C\{J(\mathbf{x}, \mathbf{T}(\mathbf{o}))\} = C\{J(\mathbf{x}, \mathbf{o})\} = \theta; \tag{1.14}$$

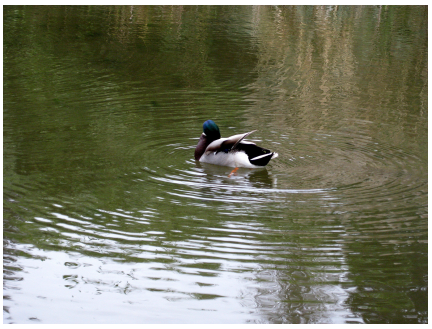
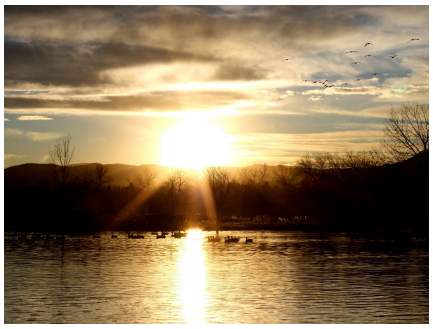

Figure 1.25: Left: a photo of a duck in a lake; right: a photo of the sun setting over a lake

the class of the dominant object, output by the classifier, remains *invariant* under translation of the dominant object.

Alternatively, *translation invariance* can be expressed in more general terms: the class of the image is the same even if the dominant object is located in a different place within the image. In the context of a 3D scene in front of a camera: moving an object relative to the camera such that it lies in a different position in the image does not change its identity or class.

However, translation invariance is not universal, as the two photos in Figure 1.25 demonstrate. In the left photo, the identity of the photo would remain a duck on the lake, irrespective of the location of the duck in the image. Assuming that the duck is the dominant object, a classifier which outputs the dominant object should classify the image as "a duck" irrespective of where the duck is located on the water's surface. Hence the classifier should have translation invariance across the whole image.

In the right photo, it is difficult to select a dominant object, so imagine a classifier which should output one of four classes,

$$\theta \in \{\text{sun}, \text{reflection}, \text{sun and reflection}, \text{none}\}\,, \tag{1.15}$$

indicating whether the sun, a reflection of the sun are present, or both, or none. Due to over-exposure, both the sun and the reflection are indistinguishable in the image, except for their surroundings. Instead the location(s) of the sun with respect to the horizon determine(s) whether or not it is the sun or a reflection of the sun. Hence a classifier should not have translation invariance across the whole image.

1.5.2 Other Examples of Invariance

There are many examples of cases in which invariance is desirable, that is, where variance of a property has no bearing on the class. I categorise invariance into two categories: intrinsic invariance and extrinsic invariance. The distinction is based on whether the variance for which the class remains invariant is intrinsic or extrinsic to the objects, actions or phenomenon which are members of the same class. The final category, depiction invariance, deserves special consideration as it may involve both intrinsic and extrinsic variance.

1.5.2.1 Intrinsic Invariance

A given class may represent a class of objects, e.g. people, cars, tables, chairs or cats. It may also represent an action, such as sitting, walking, running, jumping, working, studying and seeing; it may also represent phenomena such as darkness, rain, shadow or even something as abstract as beauty or inelegance. Within this same class, there is variation which is not related to the viewing or observation of class instance: it belongs to the class itself. I refer to this as intrinsic invariance.

Intra-class variation There can be a huge variance between instances of a single class. Consider the number of different cars made over the years and for different purposes. All of them, if in working order, can carry one or more people under their own power, and have wheels (though this may change in the future). Yet there is substantial variation between them, as Figure 1.26 shows.

A similarly large level of variation applies to actions. As Figure 1.27 shows, there is a huge variation of possibles images which could be classified as "jumping".

Pose Invariance Another example of variance is pose or configuration invariance. This refers to the arrangement or state of the object itself, hence being an intrinsic property. There is a link between pose and viewpoint in that both pose and viewpoint impact upon which parts of the object are visible, however, pose invariance specifically refers to the intrinsic state of the object.

In many cases pose has no bearing on class, for example, a person is a person whether sitting, standing or kneeling (see Figure 1.28).

Figure 1.26: Top left: a Morris Minor Traveller; top right: a Smart car; bottom left: a Rolls Royce armoured car; bottom right: a McLaren MCL32 Formula One racing car; all of these are cars, but are built for different use cases and there is a substantial variation between them

Figure 1.27: These four images all show an object jumping: there is a huge variation between the objects and images though this does not change the action, demonstrated by the objects momentarily having no contact with anything except from the air.

Figure 1.28: These images all contain people, however each person has a different pose.

Likewise, as a runner progresses, he or she will occupy many poses as Figure 1.29 demonstrates. A successful classifier for person or "running" must be invariant to these different poses. On the other hand, the pose itself my dictate the class: a person who is standing is not sitting. Clearly, pose invariance is task dependent.

1.5.2.2 Extrinsic Invariance

When observing an instance of a class, there will be variation which is not intrinsic the class, but rather the circumstances of the observation. I refer to this as extrinsic invariance. This includes the lighting conditions, the viewpoint from which the scene or object is observed (extrinsic camera parameters) and intrinsic camera parameters.

Illumination Invariance Vision is the interpretation of light and therefore the lighting conditions or illumination of an object or scene result in different images. This includes the placement, spectrum and intensity of light sources and the interaction of other objects, e.g. causing shadows by blocking a light source as in Figure 1.30. It may also include parameters intrinsic to the camera, such as exposure time or sensitivity to the light, which is the probable cause of the different images of Figure 1.31.

Many interpretations of the light are invariant to the lighting conditions. The presence of an object is unaffected by changes in its lighting condition, and therefore algorithms must be invariant to these changes. On the other hand, some interpretations, such as "day" v. "night", rely on exact illumination (in the absence of other visual cues

Figure 1.29: A sequence of photographs of a person running, captured by Eadweard Muybridge: the many different poses still correspond to the same activity—running.

such as a visible night sky) and are not illumination invariant.

Scale Invariance Objects and scenes appear at different scales in images. As well as viewing from an object or scene from closer or farther away, this could also be caused by lenses, such as a zoom lens or magnifying glass. Figure 1.33 shows an example of the same scene taken from the same viewpoint but with different levels of zoom (as well as camera orientation). In addition, images can be scaled through resizing.

Often the class of an object or scene is invariant to scale, for example all the top images of Figure 1.33 are of the same scene containing a boat and all the bottom images of Figure 1.33 of the same scene containing bark. This is in spite of them being photographed with the camera at different zoom levels. It is also possible to match parts of the scene between images, for example the funnel of the boat. On the other hand, the scale of an object, relative to other objects, can alter the class in other circumstances. Figure 1.32 shows a model railway layout. Here, the scale of the objects, compared to the bricks of the building housing it, is indicative of the objects being scale models of the original objects.

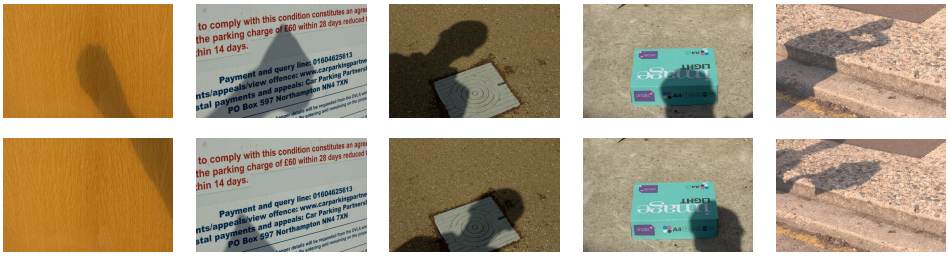

Figure 1.30: Shadows, caused by the interaction of other objects, result in changes in illumination conditions. This figure shows five different scenes; each scene has two images with different shadows.

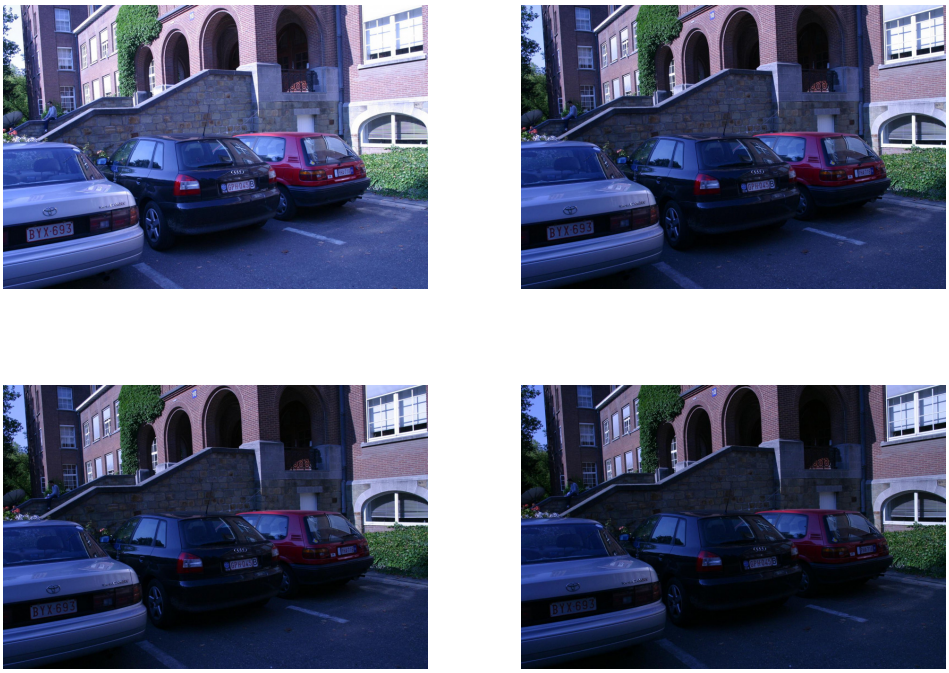

Figure 1.31: The same scene with a different illumination, perhaps cause by changes to the camera's exposure time rather than changes to the lighting of the scene itself.

Figure 1.32: 1930s model railway layout at Brighton Toy and Model Museum: objects are still recognisable in this images despite being at a small scale, though are clearly recognisable as models.

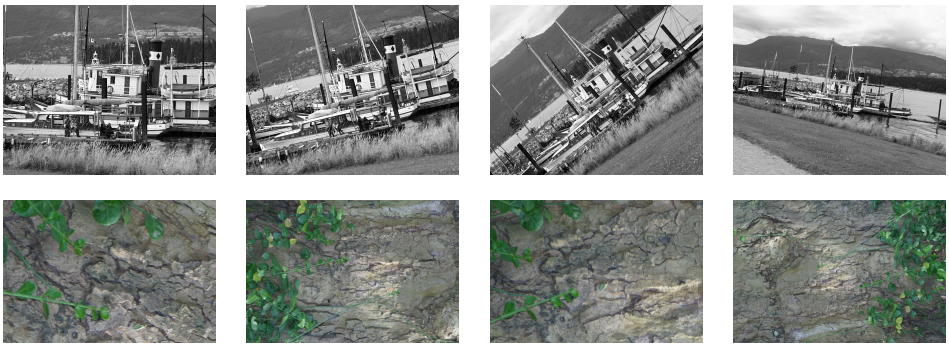

Figure 1.33: Two scenes taken with the camera at different zoom levels and orientations

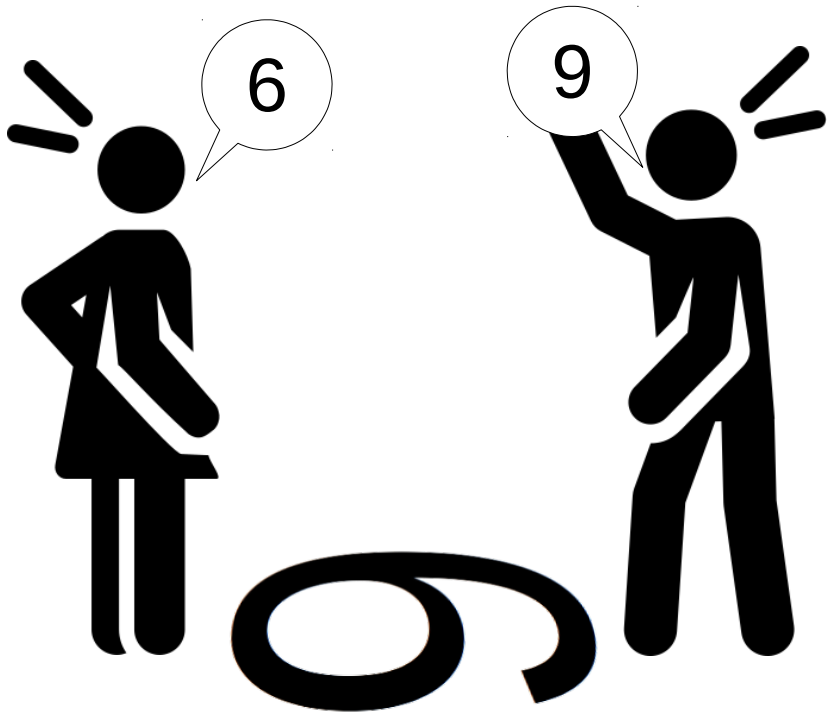

Figure 1.34: The class of a "6" or a "9" is not rotation invariant over a 180° rotation.

Rotation Invariance In addition, objects may appear at different orientations within different images as a result of the relative orientation between the camera and the object. Typically, this has no bearing on the class: the top scenes in Figure 1.33 all contain a boat in spite of the different orientations. However, one would probably identify the left-most image as being the "right way up" or canonical orientation. Likewise, the label "upside down" would be based on the orientation being the opposite of the canonical orientation. Clearly rotation invariance is also task dependent. Another example is the numbers "6" and "9", which are rotations of each other, and hence only rotation invariant to an extent (see Figure 1.34).

Viewpoint Invariance With a few exceptions, such as a table tennis ball or ball bearing, objects and scenes typically have a different ap-

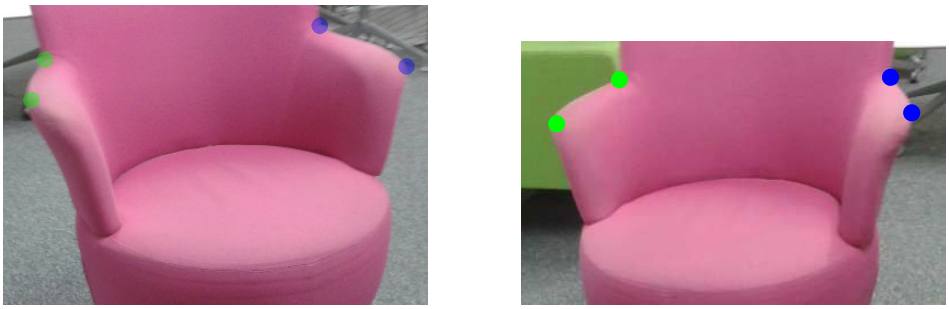

Figure 1.35: In these two images of the same chair, the same parts of the chair appear in different locations relative to each other between images.

pearance from different viewpoints. This also typically has no bearing on the class: a bicycle is still a bicycle whether it is observed from the front or side-on. Likewise for a scene, a kitchen remains a kitchen whether the cameras is focused on the oven or the refrigerator. On the other hand, the viewpoint itself, relative to the scene or object, can itself determine the class, e.g. the "front" or "back" of the object.

Another result of viewing objects or scenes from different viewpoints is that the same parts or features within appear in different locations relative to each other between images of the same object or scene. One example of this is *parallax* (see Figure 1.35).

1.5.2.3 Depiction Invariance

Depiction invariance deserves special consideration as the way a class is depicted in an image may arguably involve variation which is both intrinsic and extrinsic to the class. The choice of depicting a person for a portrait in watercolours, oil paint, acrylic paint, pencil, charcoal or another medium would appear to be extrinsic to the person class. On the other hand, a representation of a person which is more symbolic than lifelike may represent variation which is intrinsic to the class. For a human observer, a representation of a person goes beyond photographs and photo-realistic depictions: symbols, pictograms and sketches can also represent people in images and be recognised as such. Figure 1.36 shows examples of such depictions of people.

Figure 1.36: From left to right: a pictogram of a person, a portrait by a three-year-old child, a Chinese character in ancient oracle script representing a side view of a person, a light painting of a person in a chair

1.5.3 Sources of Invariance in Algorithms

The previous section identified examples of invariance within images. This section outlines the sources of invariance in Computer Vision algorithms. In my framework, there are three, which I present in the next subsections:

1. Implicit invariance
2. Invariance from *a priori* knowledge
3. Invariance from learning

1.5.3.1 Implicit Invariance

I consider implicit invariance to be invariance which cannot be attributed to knowledge, either *a priori* or *a posteriori*. An image classifier (see (1.3)) will always have invariance because a classifier is non-injective: it can only return one of a fixed number of outputs, $\theta \in \Theta$. Hence, for a given class, θ, there will be many images,

$$I(\mathbf{x}) \in \mathcal{I}_\theta : C\{I(\mathbf{x})\} = \theta, \tag{1.16}$$

which are classified the same. Similarly, for a given class, θ, and image, $I(\theta)$, there will be many transformations,

$$T : I(\mathbf{x}) \in \mathcal{I}_\theta \mapsto I'(\mathbf{x}) \in \mathcal{I}_\theta, \tag{1.17}$$

Figure 1.37: Images with the digit two from the the MNIST database: the 1-nearest neighbour classifier was able to correctly classify images in the top row but not the bottom row.

for which the classifier is invariant, i.e.

$$C\{T\left(I(\mathbf{x})\right)\} = C\{I(\mathbf{x})\} \tag{1.18}$$

As an example, consider a nearest neighbour image classifier, which operates on images of a fixed size, $w \times h$. The classifier first uses the image vectorisation operator (1.43), $\text{vec}(I(\mathbf{x}))$, to generate a vector, $\mathbf{v}$. Next, the image classifier operates as a nearest neighbour classifier using the generated vector. Prior to classification, the classifier requires a list of class labels and vectors, $\{(\mathbf{v_1}, \theta_1,), (\mathbf{v_2}, \theta_2,), \ldots\}$ from a set of images with known class labels, the training set, $\{I_1(\mathbf{x}), I_2(\mathbf{x})\}$. To return the class of an unseen image, $I_{\text{unseen}}(\mathbf{x})$, the classifier returns the class label of the nearest vector in the training set, using Euclidean distance, to the vector of the new image, $\mathbf{v}_{\text{unseen}}$, i.e.

$$\theta_i, i = \arg\min_i \|\mathbf{v}_{\text{unseen}} - \mathbf{v}_i\| \tag{1.19}$$

As a result, the training set image images result in decision boundaries in the vector space and the classifier is implicitly invariant within these decision boundaries.

An example of using such a classifier is to classify images in the MNIST database. All the images are the same size, 28×28 pixels. Each image contains a single digit from zero to nine. Hence, the classification task is to identify which digit (the class) is present in each image. There are 60 000 images in the training set and 10 000 in the test set. To perform well, the classify must be invariant to the different handwriting styles and slight differences in the orientation and position of each digit.

The nearest neighbour image classifier achieves an error rate of 3.1 %. Figure 1.37 shows ten images of the digit two which were correctly classified (top row) and ten which were not (bottom row). The

implicit invariance in the classifier yields some invariance to different styles of handwriting but is not optimal for the task.

1.5.3.2 Invariance from *A Priori* Knowledge

I consider "invariance from *a priori* knowledge" to be invariance which can be attributed to knowledge used in the design of the classifier, prior to the algorithm having access to the training data. This may involve a human observer applying their own prior knowledge of the world, including knowledge of what objects look like and how they can vary in appearance. It may also involve a human observer examining the dataset an making inferences about the invariance using his or her own vision, with the intention of creating a computer algorithm with a similar ability. Finally, it may also involve pragmatism, combining visual and object knowledge with that of how to implement computer algorithms in an efficient manner.

In the previous section, a Euclidean distance metric was chosen as a convenience. Other distance metrics could be used, for example Manhattan distance. However, the distance metric does not alter the vector space, which is formed by each pixel intensity as a result of the convenient use of the image vectorisation operator. This vector space is not ideal: it is not the value of individual pixels in themselves which cause an image to represent a digit but the relations between pixels. Therefore, no distance metric in this vector space will yield optimal performance.

Instead, with reference to the task, *handwritten digit recognition*, one can reason about what invariance the classifier should have. One way to do this is to consider what changes one could make to an image of a digit which do not change the identity of a digit (Simard, LeCun and Denker, 1993). Making the digit larger or smaller, i.e. scaling the image, does not change the identity unless the digit becomes so large as to not fit in the image or so small as to converge into too few pixels. Likewise, rotating the digit does not change its identity, with an exception of "6" and "9" which are rotated versions of each other. However, one could decide that a sensible cap on rotation would be 15° as one would not expect the digits to be rotated too much in this task. Finally, the digit could be translated within the image and as long as parts of it were not lost outside the image, the digit would still be recognisable.

One way to make the previous classifier invariant to these trans-

formations is to augment the test image with transformed versions at test time, including scaling rotation and translation. As a result, the Euclidean distance metric would measure the shortest distance not just from the test image but the rotated, scaled and translated versions: the classifier will be invariant to such transformations. I transform the image as follows:

1. a rotation by α, centred on the image,
2. a scaling of s, centred on the image, and
3. a translation of $\mathbf{t}$.

In order to generate a finite number of images from a test image, I randomly and uniformly sample from the infinite range of possible transformations; the range of transformations is the Cartesian product,

$$(\alpha, s, \mathbf{t}) = \mathcal{A} \times \mathcal{S} \times \mathcal{T} \tag{1.20}$$

$$\mathcal{A} = \{-5^\circ, -4^\circ, \ldots, 5^\circ, \} \tag{1.21}$$

$$\mathcal{S} = \{0.95, 1.00, 1.05\} \tag{1.22}$$

$$\mathcal{T} = \left\{ \begin{bmatrix} -2 \\ -2 \end{bmatrix}, \begin{bmatrix} -2 \\ 0 \end{bmatrix}, \begin{bmatrix} -2 \\ 2 \end{bmatrix}, \begin{bmatrix} 0 \\ -2 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 2 \end{bmatrix}, \begin{bmatrix} 2 \\ -2 \end{bmatrix}, \begin{bmatrix} 2 \\ 0 \end{bmatrix}, \begin{bmatrix} 2 \\ 2 \end{bmatrix} \right\}. \tag{1.23}$$

The new classifier performs with an error rate of 2.4 %, an improvement from the 3.1 % of the previous classifier. This suggests that augmenting the test image with many transformed versions results in a better invariance for the classifier. This is understandable: in the previous classifier, transformations such as a small rotation or scaling of the digit could result in a large distance between the original and transformed version. Under this classifier, the addition of transformed versions results in the distance between much closer to zero for such transformations. Figure 1.38 shows a similar figure to Figure 1.37 for this classifier, i.e. ten images of the digit two which were correctly and incorrectly classified respectively.

1.5.3.3 Invariance from Learning

I consider *invariance from learning* to be invariance resulting from machine learning, "a set of methods that can automatically detect patterns in data" (Murphy, 2012). The knowledge comes *a posteriori*:

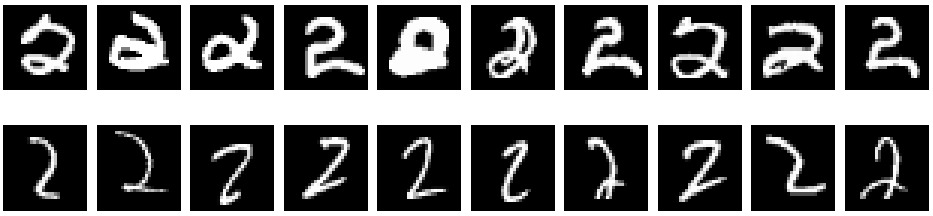

Figure 1.38: Images with the digit two from the the MNIST database: the 1-nearest neighbour classifier with augmented images at test time was able to correctly classify images in the top row but not the bottom row.

the learning algorithm itself learns from the training data, hopefully accurately, the required invariance for the task.

By learning patterns in the data, machine learning methods can automatically learn the required invariance for the task. In the earlier example, using a nearest neighbour image classifier to classify handwritten digits, the training set was stored in memory. Therefore, some learning took place, known as "lazy learning", since no generalisation took place until a new image was seen. With a large enough training set, the classifier would achieve the right level of invariance. However, if the right level of invariance could be instead be learnt from a small dataset, it would lead to improved performance.

One example of a learning algorithm is a convolutional neural network (CNN), a neural network with convolutional layers. Although the CNN involves a learning stage, the training, the design of the CNN is specified *a priori*. As with any learning algorithm, a CNN will operate no better than a random classifier over all possible classification problems (see Section 1.5.7). It is vital for the CNN or any learning algorithm to be designed with the problem in mind, in this case, relying on *a priori* knowledge.

As a specific instance, consider a feed-forward CNN with three convolutional layers, one fully connected layer and a final inner product layer. Each convolutional layer convolves the input of the previous layer with a number of kernels, producing a new images, with one channel per kernel. Between each pooling layer is a rectified linear unit (ReLU) layer, a normalisation layer and a pooling layer. The ReLU layers add non-linearity, the normalisation layer has an effect of "lateral inhibition" and the pooling (max-pooling) layers (see Section 1.6.2) takes the maximum response in a pixel region.

It should be noted that this is also an example of invariance from *a priori* knowledge. As Section 1.5.7 proves, the design of any learning algorithm for any task must either be based on *a priori* knowledge. Otherwise it has no guarantee of performing any better than random on another dataset. In this case, the design is based on the following *a priori* knowledge or estimates:

1. The digits are formed by a (generally) continuous line, so small convolutional kernels are appropriate: here the kernels are size 3×3.

2. Small translations locally have no significant bearing on the digits: the pooling layers achieve this.

3. The digits may have small local variations, such as the very bottom of the digit two, and larger variations across the whole digit, e.g. a slight slant: the choice of three layers allows a gradually increasing receptive field and for the invariance achieved by the pooling layers to double in size for each successive pooling layer.

4. After the final pooling layer, the output is a 3×3 pixel image with each pixel having a receptive field of 18×18 and 128 channels. There is a fully connected layer with 128 outputs, which ensures that the image is considered as a whole. The choice of 128 outputs is practical as a power of two, and allows just over ten channels per digit to allow for variations in how the digits are drawn.

5. After this layer, there is a fully connected layer with ten outputs, to correspond to each of the digits. This is followed by a Softmax layer, which applies the Softmax function (Bishop, 2006) to the ten outputs, such that the output is a valid discrete probability distribution over the ten possible digits.

The CNN learns weights for each of the convolution kernel using back-propagation (Goodfellow, Bengio and Courville, 2016; Rumelhart and Hinton, 1986) and stochastic gradient descent (SGD) on the training set. As well as the *a priori* invariance yielded from the design of the CNN, the CNN learns invariance through the learning of the weights. Therefore, this is an example of *invariance from learning*.

The resulting classifier has an error rate of under 0.37 %, smaller than the previous two classifiers. Figure 1.39 shows a similar figure

Figure 1.39: Images with the digit two from the the MNIST database: the CNN classifier was able to correctly classify images in the top row but not the bottom row

to Figures 1.37 and 1.38 for this classifier, i.e. ten images of the digit two which were correctly and incorrectly classified respectively.

1.5.4 Invariance from Covariance

One method of achieving invariance is to use an operator with covariance, followed by an additional operator. Both operators take an image as an input. The operator with covariance yields an output which is covariant to a transformation. The additional operator takes this output as an input, in addition to the to the image, and yields a result which is invariant to the transformation.

The operator may achieve covariance by its design, originating from *a priori* knowledge or reasoning. Alternatively the operator may be learnt. As such the source of invariance may be *a priori* or from learning, as outlined in the previous section. Here, I demonstrate the covariance of both object detectors and interest-point detectors. In addition, I show how to use covariance to yield invariance.

1.5.5 Translation Covariance in Object Detectors

Consider an image containing an object centred on $\mathbf{o}$, $J(\mathbf{x}, \mathbf{o})$. As with the classification case, consider translating the object's centre by a vector, $\mathbf{t}$,

$$\mathbf{T} \colon \mathbb{P} \to \mathbb{P} \tag{1.24}$$

$$\begin{bmatrix} o_x \\ o_y \end{bmatrix} \mapsto \begin{bmatrix} o_x + t_x \\ o_y + t_y \end{bmatrix}. \tag{1.25}$$

Assume that the object detector, Equation (1.5), yields a correct output,

$$D\{J(\mathbf{x}, \mathbf{T}(\mathbf{o}))\} = \left(\theta, \mathbf{T}(\mathbf{o}) - \frac{1}{2}\begin{bmatrix} w \\ h \end{bmatrix}, \mathbf{T}(\mathbf{o}) + \frac{1}{2}\begin{bmatrix} w \\ h \end{bmatrix}\right). \tag{1.26}$$

Here the class, θ, remains *invariant* under translation of the object $\mathbf{o}$. On the other hand, the coordinates of bounding box rectangle, $\mathbf{x}_{\text{TL}}$ and $\mathbf{x}_{\text{BR}}$, are *covariant* to the object centre, $\mathbf{o}$. This is an example of *translation covariance*.

Covariance may be used to yield invariance. Consider a rectangular cropping operator,

$$R(I(\mathbf{x}), \mathbf{x}_{\text{TL}}, \mathbf{x}_{\text{BR}}) = I_{\text{cropped}}(\mathbf{x}), \tag{1.27}$$

which outputs a cropped image, such that

$$I_{\text{cropped}}(\mathbf{x}) = I(\mathbf{x} + \mathbf{x}_{\text{TL}}). \tag{1.28}$$

In addition, the domain of $I_{\text{cropped}}(\mathbf{x})$, $\mathbb{P}_{\text{cropped}} \subset \mathbb{P} \subset \mathbb{N}^2$, is reduced such that the width and height of the new image match that of the bounding box,

$$w_{\text{cropped}} = x_{\text{BR}x} - x_{\text{TL}x}, h_{\text{cropped}} = x_{\text{BR}y} - x_{\text{TL}y}. \tag{1.29}$$

By applying $R(I(\mathbf{x}), \mathbf{x}_{\text{TL}}, \mathbf{x}_{\text{BR}})$ to the image containing an object, centred on $\mathbf{o}$ and translated by vector $\mathbf{t}$, $J(\mathbf{x}, \mathbf{T}(\mathbf{o}))$, and using the coordinates, output by the object detector $D\{J(\mathbf{x}, \mathbf{T}(\mathbf{o}))\}$, which are *covariant* to $\mathbf{o}$, as arguments, $\mathbf{x}_{\text{TL}}$ and $\mathbf{x}_{\text{BR}}$ for the cropping function, $R(I(\mathbf{x}), \mathbf{x}_{\text{TL}}, \mathbf{x}_{\text{BR}})$, the resulting image, $I_{\text{cropped}}(\mathbf{x})$, is invariant to the translation of the object, $\mathbf{o}$. Hence, the *translation covariance* of the detector bounding box output has been used to obtain a image containing the object, which is *translation invariant*. In more general terms, cropping an image to the bounding box of the object yielded by the object detector obtains the same image of the object, irrespective of the location of the object in the original image.

In practice, the object will typically not occupy the whole bounding box. Therefore only pixels in the image, $I_{\text{cropped}}(\mathbf{x})$, which are part of the object will be invariant. The other pixels will vary based on the background, as Figure 1.40 demonstrates.

Figure 1.40: Left: A silhouette of a cat has been placed in two difference places in a living room; right: the cropped images for both silhouettes

1.5.6 Covariance in Interest-Point Extractors

The same principle is used for obtaining local descriptors from interest-points (Section 1.2.3.2). Consider, for example, a local descriptor extractor which outputs a set of interest-points each containing a location,

$$\mathbf{p_n} \in \mathbb{P}. \tag{1.30}$$

Consider applying an interest-point extractor, D_{IP}, to an image $I(\mathbf{x})$,

$$D_{IP} \colon \mathbb{P} \to \mathbb{N}_0^N, \tag{1.31}$$

yielding interest-points,

$$\mathcal{P} = D_{IP}\{I(\mathbf{x})\}. \tag{1.32}$$

Consider also the image formed by translating this image by a vector $\mathbf{t}$, i.e. a translation,

$$\mathbf{T} \colon \mathbb{P} \to \mathbb{P} \tag{1.33}$$

$$\begin{bmatrix} x_x \\ x_y \end{bmatrix} \mapsto \begin{bmatrix} x_x + t_x \\ x_y + t_y \end{bmatrix}. \tag{1.34}$$

The translated image,

$$I_{\text{translated}}(\mathbf{x}) = I\big(\mathbf{T}^{-1}(\mathbf{x})\big), \tag{1.35}$$

$$\mathbf{T}^{-1}(\mathbf{x}) = \mathbf{x} - \mathbf{t}, \tag{1.36}$$

however not all pixels, $\mathbf{T}^{-1}(\mathbf{x})$, are defined, as they lie outside the original image. One solution is to set any pixel for which $\mathbf{T}^{-1}(\mathbf{x})$ lies outside the original image, i.e. the domain of $I(\mathbf{x})$, $\mathbb{P}$, to a fixed value, e.g. $\mathbf{0}$. Therefore let the translated image be as follows:

$$I_{\text{translated}}(\mathbf{x}) = \begin{cases} I(\mathbf{T}^{-1}(\mathbf{x})), & \text{if } \mathbf{T}^{-1}(\mathbf{x}) \in \mathbb{P}, \\ \mathbf{0}, & \text{otherwise.} \end{cases} \tag{1.37}$$

In addition, the width, w', and height, h', of the translated image will change as follows:

$$w' = w + t_x, h' = h + t_y. \tag{1.38}$$

Applying the interest-point detector to the translated image yields a new set of interest-points,

$$\mathcal{P}' = D_{IP}\{I_{\text{translated}}(\mathbf{x})\}. \tag{1.39}$$

Both $\mathcal{P}$ and $\mathcal{P}'$ are unsorted. If both interest-point sets were sorted, e.g. first by the first dimension then by the second dimension, it would, in theory, be possible to relate interest-points between the two sets as follows:

$$\mathbf{p_n} + \mathbf{t} = \mathbf{p'_n}. \tag{1.40}$$

In practice, there are two caveats:

1. When $t_x < 0$ or $t_y < 0$, some pixels in the original image, $I(\mathbf{x})$, do not appear after translation. Therefore, some interest-point in the original image, $I(\mathbf{x})$, may not exist in the translated image $I_{\text{translated}}(\mathbf{x})$, i.e. interest-points $\mathbf{p} \in \mathcal{P}$ for which $\mathbf{p} + \mathbf{t} \notin \mathcal{P}'$.

2. The introduction of new pixels, $\mathbf{x} : \mathbf{T}^{-1}(\mathbf{x}) \notin \mathbb{P}$, may result in additional interest-points in the translated image, $I_{\text{translated}}(\mathbf{x})(\mathbf{x})$, which were not in the original image, i.e. interest-points $\mathbf{p} \in \mathcal{P}'$ for which $\mathbf{p} - \mathbf{t} \notin \mathcal{P}$.

In addition, if the local descriptor is defined for interest-points, $\mathbf{p_m} \in \mathcal{P}'$, $\mathbf{p_n} \in \mathcal{P}$, i.e. both $\mathbf{E}_{LD}\{I(\mathbf{x}), \mathbf{p_n}\}$ and $\mathbf{E}_{LD}\{I_{\text{translated}}(\mathbf{x}), \mathbf{p_m}\}$ are defined and

$$\mathbf{p_m} = \mathbf{p_n} + \mathbf{t}. \tag{1.41}$$

then,

$$\mathbf{E}_{LD}\{I(\mathbf{x}), \mathbf{p_n}\} = \mathbf{E}_{LD}\{I_{\text{translated}}(\mathbf{x}), \mathbf{p_m}\}. \tag{1.42}$$

Hence, through the *translation covariance* of the interest-point extractor, the local descriptor is *translation invariant*. (The local descriptor operator operates on the region or neighbourhood or the interest-points. Therefore, the local descriptors may not be defined for a given interest-point if it lies too close to the edge of the image and the equation only holds if the local descriptor is defined at both locations.)

The result is that the local descriptor will be identical for the same point no matter what the location is. A similar principle applies for interest-point extractors with other covariance such as scale, rotation or affine. This allows interest-points to be matched between other images, through the use of the local descriptor, in spite of such transformations. Since the interest-point extractor is not covariant to all possible changes between images, e.g. viewpoint changes, the local descriptor must still be designed to allow for such variations in appearance, to be useful for matching the same 3D feature between two images (see Section 1.2.3.2).

1.5.7 Motivation for Considering Invariance

It is reasonable to ask whether it is necessary or useful to consider invariance for Computer Vision tasks. Consider classifying an image, $I(\mathbf{x})$, with fixed size, $w \times h$. Any such image can be turned into a fixed size vector using an image vectorisation operator,

$$\text{vec}(I(\mathbf{x})) = \begin{bmatrix} I\left(\begin{bmatrix}1 & 1\end{bmatrix}^\top\right) \\ \cdots \\ I\left(\begin{bmatrix}w & 1\end{bmatrix}^\top\right) \\ I\left(\begin{bmatrix}1 & 2\end{bmatrix}^\top\right) \\ \cdots \\ I\left(\begin{bmatrix}w & 2\end{bmatrix}^\top\right) \\ I\left(\begin{bmatrix}1 & h\end{bmatrix}^\top\right) \\ \cdots \\ I\left(\begin{bmatrix}w & h\end{bmatrix}^\top\right) \end{bmatrix} \tag{1.43}$$

As a short hand let $\mathbf{v} = \text{vec}(I(\mathbf{x}))$.

1.5.7.1 Assume the aim is to maximise accuracy

Assume that every classification is considered either correct or incorrect, i.e. there is the same penalty for a similar classification (cat v. dog, both being animals) as for a dissimilar one (car v. dog). In addition, assume that every class, $\theta \in \Theta$, and image $I(\mathbf{x})$, has equal weighting for a misclassification. This corresponds to using a "zero-one" loss,

$$L(\theta, \theta_H) = \begin{cases} 0, & \text{if } \theta = \theta_H, \\ 1, & \text{if } \theta \neq \theta_H, \end{cases} \tag{1.44}$$

where θ is the true class and θ_H is the classifier hypothesis. In addition, minimising this loss corresponds to maximising overall accuracy.

1.5.7.2 Optimal classification requires knowledge of the true distribution

The optimal classification, $C_{\mathbf{v}}$, for a vector, $\mathbf{v}$, is the one which minimises the expected loss (Bishop, 2006),

$$\mathbb{E}[L] = \int_{\mathbf{v}} \sum_{\theta \in \Theta} L(\theta, C_{\mathbf{v}}) P(\theta, \mathbf{v}) d\mathbf{v} \tag{1.45}$$

$$= \int_{\mathbf{v}} \sum_{\theta \in \Theta} L(\theta, C_{\mathbf{v}}) P(\theta|\mathbf{v}) P(\mathbf{v}) d\mathbf{v}. \tag{1.46}$$

Therefore, this is the classification, $C_{\mathbf{v}}$, which minimises $L(\theta, C_{\mathbf{v}})P(\theta|\mathbf{v})$ for a given $\mathbf{v}$. With a "zero-one" loss (1.44), the loss is unity except for a correct classification, where it is zero. Hence, optimal classification, $C_{\mathbf{v}}$, is the class, $\theta \in \Theta$, which maximises the posterior and therefore the joint probability,

$$\arg\max_{\theta \in \Theta} P(\theta|\mathbf{v}) = \arg\max_{\theta \in \Theta} P(\theta, \mathbf{v}). \tag{1.47}$$

Optimal classification, therefore, requires knowledge of either the true posterior distribution or the true joint distribution.

1.5.7.3 It is impossible to obtain the true distribution

If either the posterior distribution or joint probability distribution were known, it would be trivial to yield an optimal image classifier, without reference to invariance. This is consistent with the analysis

of X. Shi and Manduchi (2004), showing that the use of invariants is suboptimal from a Bayesian perspective.

However, digital images are the product of a vast range of inputs, including photographs of natural scenes, scans of paintings produced by artists and digital processes. Moreover, these processes and inputs evolve over time. Therefore, I assume that it is impossible to obtain the true probability distribution. Consequently, I assume that image classification relies on learning either an appropriate model of the distribution or a suitable decision function.

1.5.7.4 Training examples alone are insufficient

The "no free lunch" (NFL) theorems of Wolpert (1996) prove that simply using a set of training examples alone, for example 1000 images of each class, is insufficient to design an image classifier. Such an approach, specifically learning without any assumptions about the posterior, here $P(\theta|\mathbf{v})$, is known as *agnostic learning* (Kearns, Schapire and Sellie, 1994; Valiant, 1984). However, Wolpert (1996) demonstrates that such an approach, averaged across all possible targets, performs no better or worse than a random classifier.

Wolpert's second NFL theorem is as follows:

NFL Thorem Two. *Assuming off-training set (OTS) error, a vertical $P(d|f)$, and a homogeneous loss, the uniform average over all targets f, of $P(c|f,m)$ is $\frac{\Lambda(c)}{r}$, where $\Lambda(c)$ is a function independent of y_h and y_f (having been summed over y_f). This result is independent of the learning algorithm used.*

In Wolpert's terminology, the input space, $\mathbf{X}$, corresponds to the set of all possible images, $\mathcal{I}$, vectorised,

$$\mathbf{X} = \{\text{vec}(I(\mathbf{x}))|I(\mathbf{x}) \in \mathcal{I}\} \tag{1.48}$$

and the output space, $\mathbf{Y}$, is the set of possible classes, Θ. The training set, d, corresponds to the training examples, a set of m $\mathbf{X}$–$\mathbf{Y}$ pairs, which corresponds to a set of images and class label pairs,

$$\{(I_1(\mathbf{x}), \theta_1), (I_2(\mathbf{x}), \theta_2), \ldots, (I_m(\mathbf{x}), \theta_m)\}. \tag{1.49}$$

The target, $f(x \in \mathbf{X}, y \in \mathbf{Y}) = P(y|f,x)$, corresponds to the true posterior, $P(\theta|\mathbf{v})$, while the hypothesis, $h(x \in \mathbf{X}, y \in \mathbf{Y})$, shares the same form as f and corresponds to a particular hypothesis of the posterior,

$P(\theta|\mathbf{v})$. A given learning algorithm produces a hypothesis, h, based on the training data, d; since the algorithm may not be deterministic, it is expressed as a distribution, $P(h|d)$. A loss function is expressed as $c = L(y_F, y_H)$ where, y_F is the sample of the target f, for a given test point, q, and y_H is the sample of the hypothesis at the same test point. The NFL theorems hold for a homogeneous loss, a loss for which $\Sigma_{y_F}\delta[c, L(y_H, y_F)]$ is some function, $\Lambda(c)$, independent of y_H.

Each of the assumptions are justified as follows:

OTS error As noted in Section 1.2, I assume that the objective is to maximise the performance on unseen examples, hence minimising the OTS error.

Vertical $P(d|f)$ I assume that the same process is used to generate the test and training set, typically random assignment of all labelled images into the respective training, test and validation sets. $P(d|f)$ would not be vertical if a different process was used to generate the test set, e.g. the addition of noise not applied to the training set.

Homogeneous loss The zero-one loss (see Section 1.5.7.1) is homogeneous. See Section 1.5.7.9 for a discussion of other performance metrics.

The result of this theorem is that using an learning algorithm which places no assumption on the target, i.e. the true posterior, $P(\theta|\mathbf{v})$, and which considers only the training examples, on average performs no better or worse than a random classifier!

In addition, consider Wolpert's first NFL corollary:

NFL Corollary One. *Assuming OTS error, a vertical* $P(d|f)$*, a uniform* $P(f)$ *and a homogeneous loss,* $P(c|m) = \frac{\Lambda(c)}{r}$

This corollary demonstrates that, assuming a uniform prior for the target (i.e. no preference for one distribution, $P(\theta|\mathbf{v})$, over another) also yields performance which is no better or worse than random.

1.5.7.5 One can justify assumptions about the true distribution

The previous section argued that, without making assumptions about the true distribution (the target in Wolpert's terminology), a given learning algorithm will perform no better than a random one. Yet,

in spite of this, one observes that many Computer Vision learning algorithms perform significantly better than random. It follows that these algorithms must be predicated upon largely correct assumptions about the distribution or target, either by design or by selection of algorithms that perform well on previous tasks. Even though one cannot obtain the true distribution (see Section 1.5.7.3), one can use observations and interactions with the world to attain reasonable assumptions about the true distribution.

1.5.7.6 The proportion of sensical images is small

One observation is that the fraction of the set of all images, $\mathcal{I}$, which are sensical (e.g. those resemble a real-world class) is small (Goodfellow, Bengio and Courville, 2016). The empirical evidence for this is to sample images from a uniform distribution, as demonstrated in Figure 1.41. All these images are 400 by 400 pixels in size and each pixel intensity value is sampled from a uniform distribution spanning the full range of values; the sampling is independent for each RGB colour channel. As a result, every pixel and colour channel is uncorrelated.

Despite the images being very different from each other, the images appear indistinguishable from each other. Indeed, though one can identify such an image as coloured and uncorrelated noise, the images are otherwise non-sensical in terms of not representing classes encountered in the real-world. Since the sampling of these images could produce any 400 by 400 pixel image, it could produce an image visibly containing a person or a landscape. Yet even over a large number of samples, no sensical images are produced, suggesting that the majority of the images in $\mathcal{I}$ are non-sensical. Indeed, under the assumption of a uniform prior, $P(f)$, and N classes, two images from Figure 1.41 would be $N - 1$ times more likely not to belong to the same class than to not do so. This does not fit the perceived indistinguishability of the images. This demonstrates the invalidity of a uniform prior on targets for image classification.

1.5.7.7 One can observe class invariance under transformations

In addition to the observation that the space of sensical images is a small fraction of all possible images, another observation is *invariance*: there are many transformations under which the class remains invariant (see Section 1.5.2). These observations provide partial prior

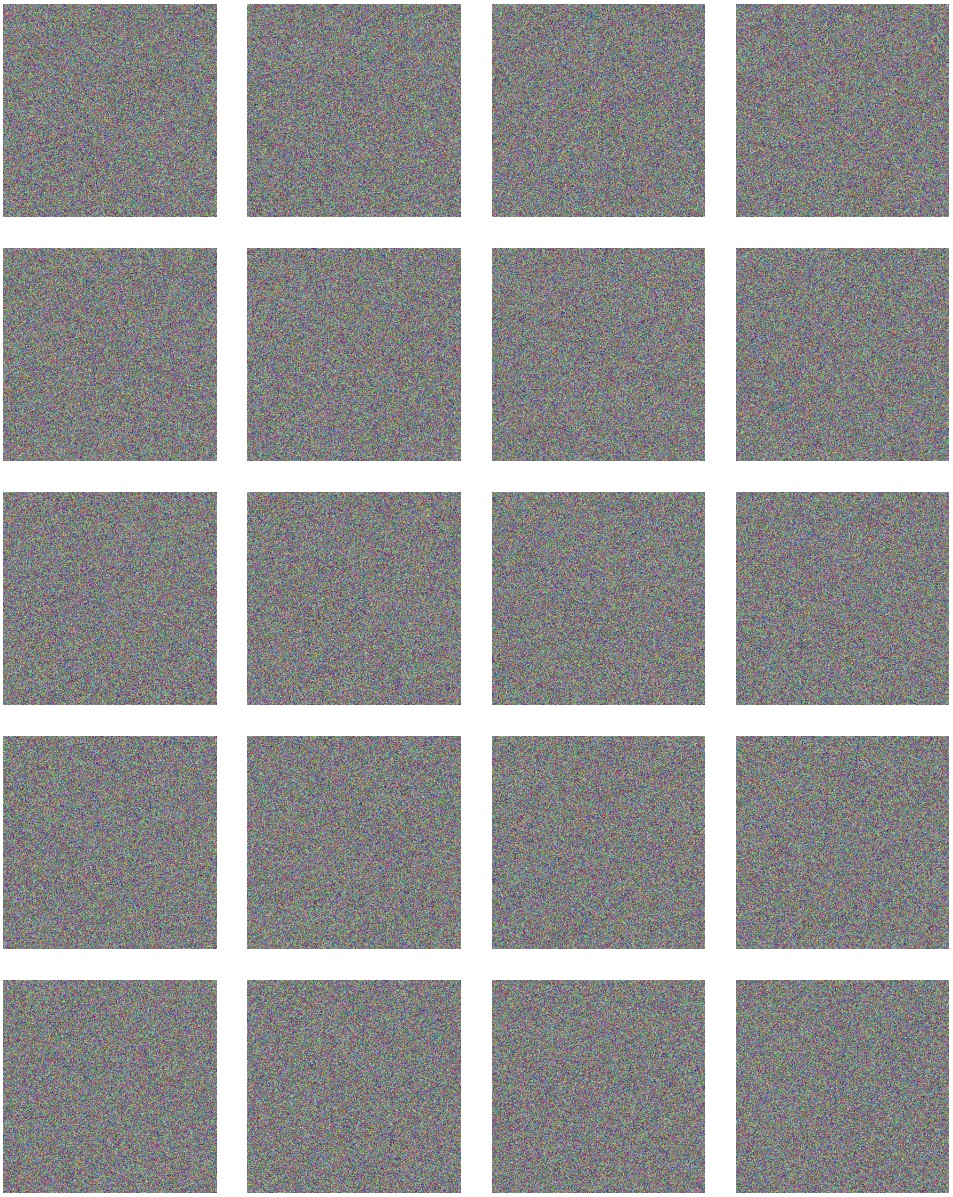

Figure 1.41: Twenty images sampled from a random uniform distribution (the RGB channels of every pixel are sampled independently); although it is possible for these to produce any 400 by 400 image, the images appear as random coloured noise and indistinguishable from each other

knowledge of the true distribution or target and enable one to design or select algorithms which perform better than random.

In conclusion, there is no universal classifier. Achieving better than random performance in classification requires forming correct assumptions about the true class posterior distribution; training example images alone are insufficient (see Section 1.5.7.4). While the true distribution cannot be obtained (see Section 1.5.7.3), invariance provides partial knowledge about the true class posterior distribution. Hence, it is useful and perhaps even necessary to consider invariance to achieve high performance for Computer Vision algorithms.

1.5.7.8 Beyond classification

In principle, it is possible to cast an object detector (1.5) to a classification problem. For every possible bounding box, specified by $(\mathbf{x}_{\mathrm{TL}}, \mathbf{x}_{\mathrm{BR}})$, the image can be cropped and resized to form a new image suitable for classification. As such, one might expect the NFL theorems to apply. In practice, detection involves classifying and localising individual object instances. Although there is often tolerance in terms of how close the predicted bounding box must be compared to the ground truth bounding box, duplicate detections of the same ground truth object instance typically result in a penalty.

If a perfect object localiser could be used, i.e. a detector which localises objects perfectly without outputting a class, requiring a separate classifier, the NFL theorem would still apply to the classification step. However, in the task of localisation, it is possible for one algorithm to have an *a priori* advantage over another. To understand this, note that there cannot be more objects than pixels within an image and in practice objects span multiple pixels. However the number of possible bounding boxes exceeds the number of pixels. As a result, an algorithm which favours a sparse number of object predictions will outperform one which does not. One such example of an *a priori* benefit in localisation is Lenc and Vedaldi (2015).

A similar principle applies for the ideal interest-point matching algorithm: this too can be cast to a classification problem. Two outputs of a local descriptor extractor (1.8), $\mathbf{d_1}$ and $\mathbf{d_2}$, can be concatenated to form a single vector,

$$\mathbf{v} = \begin{bmatrix} \mathbf{d_1} \\ \mathbf{d_2} \end{bmatrix}. \tag{1.50}$$

The vector, $\mathbf{v}$, can then be classified as a match or non-match. As a

result, the NFL theorems apply here too.

1.5.7.9 Performance metrics other than accuracy

The arguments so far, based on the NFL theorems, have assumed a homogeneous loss. One performance metric, accuracy, i.e. the ratio,

$$\frac{\text{True Negatives} + \text{True Positives}}{\text{Negatives} + \text{Positives}} \tag{1.51}$$

is equivalent to minimising the homogeneous zero-one loss, for which the NFL theorems are defined, i.e.

$$\sum_{y_H, y_F, q} \delta[y_H, y_F] P(y_H|q, d) P(y_F|q, f) P(q|d). \tag{1.52}$$

Hence the NFL theorem apply when using accuracy as a performance metric.

Other performance metrics include the area under an ROC curve (ROC-area) and average precision (AP), which is an approximation of the area under a Precision-Recall (PR) curve (Manning, Raghavan, Schütze et al., 2008). These metrics evaluate ranked retrieval tasks, however are often used to evaluate the performance of image classification, detection and matching. These metrics are used when the algorithm outputs a confidence value, possibly a predicted probability, for each input. This circumvents the need to specify a threshold (Szeliski, 2010).

In the PASCAL VOC classification challenge (Everingham, Van Gool et al., 2012), for instance, for every image, the presence or absence of each class must be predicted independently with a confidence value. This allows the images to be ranked independently for every class, allowing the calculation of the AP for each class.

The use of these metrics is to obviate the need to specify a threshold for evaluation purposes. However actual image classification, detection or matching requires a decision and hence a threshold to be specified for such algorithms. These performance metrics are not appropriate for measuring the ultimate goal which can be measured by accuracy or the zero-one loss. Therefore, the use of these performance metrics, in practice, does not undermine the arguments.

1.5.7.10 Conclusion

It follows from all the arguments that simply using training examples alone results in a performance no better than random for the classification part of object detection and for interest-point matching. Therefore, it is necessary to consider invariance in the design and advance of these Computer Vision algorithms.

1.6 Features and Spatial Pooling

Throughout this document, I consider a feature to be defined as follows (Farinella, Battiato and Cipolla, 2013):

> "a property by which real or abstract elements or objects can be distinguished"

1.6.1 Low Level Features

Just as there is no universal definition of low level and high level in Computer Vision, there is no universal definition for a "low level feature". However, I define low level features to be features meeting the following criteria:

1. Low level features must have the ability to be calculated throughout the image, with the possible exception of locations too close to the edge of the image. (At larger scales, the features may be calculated by down-sampling the image first, explicitly or in effect.)

2. Low level features are not as a result of a spatial pooling: this is to provide a more concrete distinction between low level features and those at other levels.

In practice, the second criteria is subjective and there is no mathematically expressed distinction between the "levels" of features.

1.6.1.1 Examples of low level features

There is a vast number of low level features used for various Computer Vision tasks (Farinella, Battiato and Cipolla, 2013; Rui, T.S. Huang and Chang, 1999). The simplest low level feature is the pixel intensity values for grayscale images, and the individual values of each pixel

channel for colour images. These have been used for template matching through normalised cross-correlation since before 1969 (Rosenfeld, 1969).

Multiple representations exist for describing colour, which can be used as low level features. The de facto RGB colour model dates from the work of Young (1802) on trichromatic colour vision in the 19th century. The hue, saturation and value (HSV) and hue, saturation and luminance (HSL) perceptual colour spaces were introduced to more correctly represent how humans perceive light (Joblove and Greenberg, 1978). These representations have been used to assist numerous Computer Vision tasks (Chebbout and Merouani, 2012; Ojala et al., 2001; Smith and Chang, 1995). Many other colour spaces have since been proposed such as normalised RGB (Cavallaro, Salvador and Ebrahimi, 2005), other perceptual colours spaces such as TSL (Terrillon, David and Akamatsu, 1998) and orthogonal colour spaces such as YUV (which aims to reduce the statistical dependence between colour channels) (Kakumanu, Makrogiannis and Bourbakis, 2007).

Many algorithms simply discard the chrominance (colour information), as it is difficult to use reliably for in many circumstances and often contains less useful information than the luminance. Such algorithms including the Viola-Jones framework (Jones and Viola, 2003), the original Histograms of Oriented Gradients (HOG) framework (Dalal and Triggs, 2005) and Scale-Invariant Feature Transform (SIFT) (Lowe, 2004).

Gradient (finite difference) operators have been used in Computer Vision for some time (Birk et al., 1979) to yield low level gradient features. They are commonly used for edge detection, e.g. the Canny edge detector (Canny, 1986) and are used in more recent algorithms, such as HOG (Dalal and Triggs, 2005) and local descriptors (Lowe, 2004; Simonyan, Vedaldi and Zisserman, 2012; Tola, Lepetit and Fua, 2010). These features have the advantage of being invariant to low frequency distortion, i.e. the level of illumination in part of an image—indeed, even if half the image had a global increase or decrease in the intensity values, only the features located at the boundary of the global change would be affected.

In addition, the small region used to calculate a single low level feature at one point in the image (the support region) of the gradient operator minimises the amount of high-frequency information lost in the process: in fact, Dalal and Triggs (2005) report that centred 1×3 gradient operators, and their transpose, outperform larger gradient

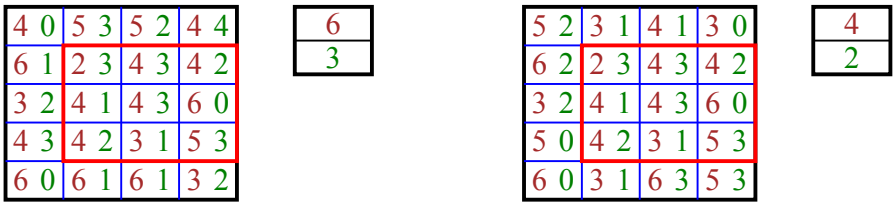

Figure 1.42: Max-pooling (left) and mean-pooling (right): the values have been pooled within the red area of the two channel image resulting in a two output values.

operators since more high-frequency information is retained; similarly the authors report that it is best not to include Gaussian smoothing.

1.6.1.2 Low Level Features from Descriptor Extractors

Under my definition, "low level features" are those produced after the smoothing and filtering operations (any of which may be omitted), as defined by the descriptor extraction framework proposed by Winder and Brown (2007). Features produced after the pooling step will be referred to as *pooled low level features*. There is some confusion in terminology between this definition and other literature: Boureau, F. Bach et al. (2010) refer to SIFT (Lowe, 2004) and HOG (Dalal and Triggs, 2005) as low level descriptors and propose that mid level features are those from a subsequent coding and additional spatial pooling step. Under my definition, the "low level descriptors" would be said to be composed of pooled low level features (with a subsequent normalisation step) as both descriptors encompass a rectangular spatial pooling operation.

1.6.2 Spatial Pooling

Spatial pooling refers to the collation of the values from multiple features in different locations within an image (the pooling region) and replacement with a summary statistic. This could be carried out by taking the maximum value (*max-pooling*), mean value (*mean-pooling*) or simply summing the values. Figure 1.42 shows an example on an image with two channels, one brown and one green. This results in spatial invariance: different inputs can produce the same output when translated slightly. The result of a small local pooling region is

to provide invariance to small translations, which is useful when such small translations do not affect the class.

The size of the pooling region influences the trade-off between invariance and the discriminative power from locational information. SIFT (Lowe, 2004) and HOG (Dalal and Triggs, 2005) use small pooling regions, achieving robustness to small misalignments in the location of a gradient feature. Bag-of-words models (see Section 2.2.1.3), on the other hand, pool medium or high level features across the whole image, resulting in the loss of locational information (for features at these levels) and spatial invariance across the whole image.

CNNs often use pooling layers and the order of layers at the convolutional stage is typically as follows:

1. A convolutional layer, followed by
2. An activation layer, for which a non-linear function, $\mathbb{R} \to \mathbb{R}$, operates on every pixel independently, followed by
3. A pooling layer

A pooling layer contains pooling regions, often square, which are applied across the image generating a new image. The pooling regions can overlap, tessellate or be sparse, creating different sized outputs. A pooling layer might have very small pooling regions, perhaps 2×2. A repeated sequence of these layers can result in a large amount of spatial invariance without too great a loss of discriminative ability, unlike a single layer with large pooling regions.

2 Literature Review

The theses cover both interest-point matching and object detection and so I present a review of the pertinent literature for each. Object detection is similar to image classification, but with localisation, and hence many developments in image classification have led to improvements in object detection. As a result, I additionally cover relevant literature for image classification which led to improvements in object detection.

2.1 Interest-Point Matching

The origin of interest-points is in corner detection. The earliest corner detector may have been that of Pingle and Thomas (1975). The authors first use a variance operator, operating on 16×16 pixel windows within the images. This operator estimates the variance in pixel intensity values within the window. Windows with an insufficient value are rejected, improving efficiency. Next the authors use a corner finder, on windows found to have sufficient variance. This applies a 3×3 vector gradient operator at every point in the window and builds a gradient histogram. At all points containing a sufficiently large gradient magnitude, the corner finder increments the component of a histogram which refers to the direction of the gradient, in terms of the half-quadrant the direction falls within.

The next step is to count the number of directions with a respected count greater than or equal to three. The corner finder retains windows for which between two and six directions have a sufficient count. Finally, the corner finder uses the points which led to these counts to estimate the location of line segments. The intersection of these lines, subject to merging and reliability tests, yield the centres and hence the locations of corners.

Moravec (1980) presents a more efficient version of Pingle and Thomas (1975)'s corner detector. Firstly, the detector uses a dir-

ectional variance detector. The detector operates on small square windows and calculates the sum of squared differences (SSD) of pixel intensity values between each pixel and the neighbouring pixel, separately over four different directions: horizontal, vertical and the two diagonals. The detector uses the minimum measurement over these four directions. The windows used to estimate the directional variance are spaced half a window length apart, and the algorithm retains only windows whose variance is a maximum over a 5×5 grid of windows centred on the window. These windows are deemed to be the locations of "corners".

Harris and Stephens (1988) present the most popular corner detector, known as the Harris and Stephens detector. The authors aimed to detect corners in addition to edges, to solve issues caused by tracking using edges alone. They identify shortcomings of the Moravec corner detector: the anisotropic response, noise caused by uniform weighting over each window and too high a response to edges. For their detector, the authors use approximated partial derivatives from the pixel intensity values yielding a 2×2 matrix at every point in this image. Corner locations are those with a large eigenvalue in both directions, which can be determined efficiently using the determinant and trace of the matrix, yielding a measure of "cornerness".

Many other corner detectors have since been produced: see Rosten, Porter and Drummond (2010) for an outline.

2.1.1 Introduction of scale space

While corner detectors provide a location, the corner, it is useful to be able to locate interest-points not just by position but by scale too, leading to the idea of scale space. This begins with Marr and Hildreth (1980), who present a theory of edge detection in human vision based on the ability to detect intensity changes over many scales. Since there exists no filter suitable for operating on all scales simultaneously, the authors present a two stage approach:

1. Blur the image to a desired resolution
2. Detect the changes in intensity at this resolution

The optimal filter for blurring the image is the one which is both band-limited in the spatial frequency domain, to pick out a particular

resolution, and spatially local, to pick out a particular location within the image. These considerations are conflicting, however the optimal filter is a Gaussian filter, whose Fourier Transform is also Gaussian.

The authors show that a large intensity change in an image results in a peak in the derivative of image intensity and a zero-crossing in the second derivative: the Laplacian, ∇^2, the orientation-independent second derivative operator. In addition, they show another important result: blurring the image with the Gaussian followed by applying the Laplacian is equivalent to convolving with the Laplacian of the Gaussian filter. The authors also propose to approximate this using the difference of two Gaussians.

The authors' approach is to turn lines of zero-crossings into small line segments and to extract an amplitude, calculated as the directional derivative perpendicular to the line segment. They then aim to combining these into a symbolic representation, a difficult task, in particular dealing with isolated edges, parallel edges, and term

Witkin (1984) identifies the need to obtain a compact and meaningful description of a signal rather than process raw numerical values directly. The author identifies the problem of scale, noting that processes which generate signals such as images often have events which occur over many scales. Instead of calculating descriptors at many scales, the author present the idea of "scale-space": convolving a signal with a Gaussian kernel leads to the scale-space image. The contours of the second partial derivatives of a 1D signal converge at a point in scale-space. The scale at the point of convergence provides a scale for the entire contour. The two other ends of a contour reach two locations in the x axis as the scale tends to zero, allowing localisation of the event. Together, these locations across all contours can be mapped as tessellating rectangles in scale-space. to provide a description.)

Koenderink (1984) also recognises the issue of scale within images and the desire to treat an image on all levels of resolution simultaneously. Analogous to the scale-space of Witkin (1984), the author proposes the idea of 3D scale space, based on the 2D image and a scale parameter. The requirements for the scale space are as follows:

causality a larger scale feature affects the finer scale, the reverse must not necessarily be true

homogeneity and isotropy the resolution depends only on the scale parameter, not the location within the image

This scale space is shown to be the solution of the diffusion equation. It is shown that the solution can be achieved by blurring an image with a Gaussian kernel.

Since the use of points of inflexion (Witkin, 1984) is not easily adapted for 2D, the author instead uses the family of equiluminance curves in scale space. As an image is successively blurred, saddle-points of these curves merge with extrema and are annihilated. An image is therefore structured in scale-space as "blobs" of light and dark which exist between the scale at which the saddle-point and extremum merge. This naturally justifies the use of "different of Gaussians" by Marr and Hildreth (1980), who overlooked the relation to the diffusion equation.

Lindeberg (1991) presented a formulation of scale-space for discrete signals, aiming for a solely bottom-up approach, with no need for *a priori* information. The approach for 1D signals begins with the following axioms:

1. Every representation should be a linear and shift-invariant transformation of the original signal, i.e. a convolution.

2. Increasing values of the scale parameter should produce signals with less structure.

3. All signals should be real and lie on the same grid.

In 1D, there exists such a kernel, which provides a true scale-space transformation, unlike a sampled Gaussian. In 2D, there is no way to transform the axioms of Koenderink (1984) into discrete space, however one method is to apply a separable convolution of the kernel used for 1D.

Lindeberg (1993a, 1993b, 1994, 1998) provides a methodology for automatically selecting locations in scale-space which are likely to correspond to "interesting structures" (interest-points), for use in higher level algorithms. The authors prove that, where one image is a scaled version of another, the m-th order derivatives correspond in scale space. Consequently, by localising minima or maxima of these

derivatives, interest-points which contain location and scale parameters can be identified between images of different scales.

This result does not specify which differential expressions to use to find interest-points in scale space, however the author proposes two differential expressions for blob interest-points, based on these expressing having been used in previous work:

1. the trace of the scale-space Hessian (also the Laplacian of Gaussian), and

2. the determinant of the scale-space Hessian

The author also analyses other differential expressions for corners, edges and ridges, as well as an approach for localising these over a finer detection scale.

2.1.2 Development of local descriptors

Z. Zhang et al. (1995) provide a framework for matching points between two images when the motion between the images is unknown. The authors improve on previous methods which are either based on global template matching (Chou and Chen, 1990; Goshtasby, Gage and Bartholic, 1984), which may suffer with occlusion boundaries, or feature matching (Shapiro and Haralick, 1981; Weng, Ahuja and T.S. Huang, 1992), which rely on reliable detection of features such as edges. The approach has three steps:

1. Establish matches between interest-points in the absence of epipolar geometry.

2. Estimate the epipolar geometry robustly.

3. Establish correspondences as in stereo matching.

The authors use Harris corners (Harris and Stephens, 1988) as interest-points, and perform matching by extracting a 15×15 pixel patch or sub-image centred on each corner, which in effect forms a local descriptor, perhaps the simplest "descriptor" for an interest-point. This patch is matched against interest-points in the other image through convolution with a much larger window (half the image in both dimensions). Locations with high enough correlation form potential matches.

To improve matching, the authors used a metric based on neighbouring matching pairs, relative distances and relative orientations. In addition, they use a heuristic to remove outliers, estimate the Fundamental matrix and establish correspondences using the epipolar constraint. The approach is shown to work in limiter circumstances, however it seems doubtful that it would work over changes in scale, or larger changes in orientation or viewpoint.

Schmid and Mohr (1997) were perhaps the first to introduce the concept of local descriptors, i.e. a replacement for matching with a sub-images or patches located at each interest-point. Their aim is to match a new (query) image with a set of existing images. At the time, previous approaches either relied on either geometric models, which were unsuitable for objects like trees, or luminance approaches which were calculated globally and suffered from partial visibility and background clutter. Their approach is instead to calculate a local descriptor at every interest-point in the image. The authors use the Harris & Stephens corner detector (Harris and Stephens, 1988), which was shown to be the most repeatable at the time (Schmid, 1996).

To form local descriptors, the authors extract a nine dimensional vector based on (Gaussian) differential invariants (rotation invariant) computed from the "local jet" at the interest-point (Koenderink and Doorn, 1987). These include the average local luminance, the square of the gradient magnitude and the Laplacian. Since this descriptor is not scale invariant, the descriptor is calculated at different scales (by changing size of the Gaussian), for the query image. Comparison between the descriptors occurs using the Mahalanobis distance (Mahalanobis, 1936).

Each descriptor in the query image votes on a database image set if its closest descriptor is from an image within that set, and the image set with the most votes is used as the label for the query image. The performance can be improved further by applying geometric constraints and removing votes which do not satisfy this. This approach is shown to be entirely rotation invariant, and successful over scale changes of up to an octave.

2.1.3 Arrival of SIFT

Lowe (1999) builds upon the work of Schmid and Mohr (1997), seeking to detect objects in scenes with background clutter and partial

occlusion, by introducing the famous Scale-Invariant Feature Transform (SIFT) keypoint. The main improvements over Schmid and Mohr (1997) are as follows:

1. Scale-invariance in the interest-point extractor (compared to the Harris & Stephens corner detector)

2. An orientation assignment process which can be used alongside an orientation-dependent descriptor to improve matching accuracy

3. A local descriptor which is more robust to illumination and viewpoint changes

2.1.3.1 Extracting interest-point

Instead of using a corner detector, SIFT extracts interest-points by locating maxima and minima in both position and scale, using a difference of two Gaussians function applied to image. This function is an approximation of the Laplacian of Gaussian (Marr and Hildreth, 1980) and includes the required normalisation i.e.

$$L(x, y, kt) - L(x, y, t) \approx (k - 1)t\nabla^2 L \tag{2.1}$$

The required outputs of this function over many scales can be produced efficiently through repeated separable Gaussian convolutions to produce many images, $L(x, y, t)$, for a given t an then by subtraction of each image from its adjacent to achieve the result of (2.1). For additional efficiency, images can be downsampled to half the width and height after each blur, resulting in an image pyramid. Extrema or "keypoints" are defined as pixels in each of the images for a given t) which are smaller or larger than all of the eight surrounding pixels, i.e. the result of *non-maximal suppression*. Each keypoint is also local maximum or minimum across scale which occurs by removing keypoints from any scale for which this does not apply. (The term keypoint and interest-point is interchangeable.)

This results in a number of keypoints which have a location and scale but no orientation. To assign an orientation to each keypoint, the gradients at pixels in the neighbourhood of the keypoint are accumulated into a 36-bin histogram, based on the orientation of the gradient at each pixel. The magnitude of the gradient at each pixel, weighted by a Gaussian distribution centred on the keypoint, is added

to the value for each bin; there is also a thresholding process on gradients to minimise the impact of illumination differences on the keypoint orientation. The orientation bin with the largest value provides the canonical orientation for the keypoint.

2.1.3.2 Calculating the local descriptor

The SIFT local descriptor is produced from the gradient magnitudes and orientations in a square neighbourhood of the keypoint. The principle is to bin the gradient magnitudes and orientations into eight orientation channels, relative to the canonical orientation. This occurs first by assigning the gradient for each pixel to the two closest orientation channels for each pixel using interpolation (see left in Figure 2.2) and then summing the gradient magnitudes for each channel across a 4×4 grid (see right in Figure 2.2). A similar approach occurs at a level of the pyramid one octave higher, using a 2×2 grid, which is not shown in Figures 2.1 and 2.2. The result is a vector of size $8 \times 4 \times 4 + 8 \times 2 \times 2 = 160$, which forms the local descriptor.

Matching between keypoints in different images occurs using the best-bin-first search method (Beis and Lowe, 1997) and searching the likely affine transformation between images occurs using a generalised Hough transform (Ballard, 1981). Matches located in the modal bin can then be used to obtain the optimal affine transform through the least-squares solution.

2.1.3.3 Later improvements

Lowe (2004) introduces a number of improvements to SIFT, closer resembling more commonly used version, including the following:

1. Instead of downscaling using bilinear interpolation after each blur, the images are downscaled (to half width and height) by taking every other pixel, forming octaves. In addition, repeated Gaussian blurring is used to achieve multiple scale intervals between octaves.

2. Images are initially doubled in size, followed by an immediate Gaussian blur, allowing interest-points to be extracted more reliably at the smallest scale.

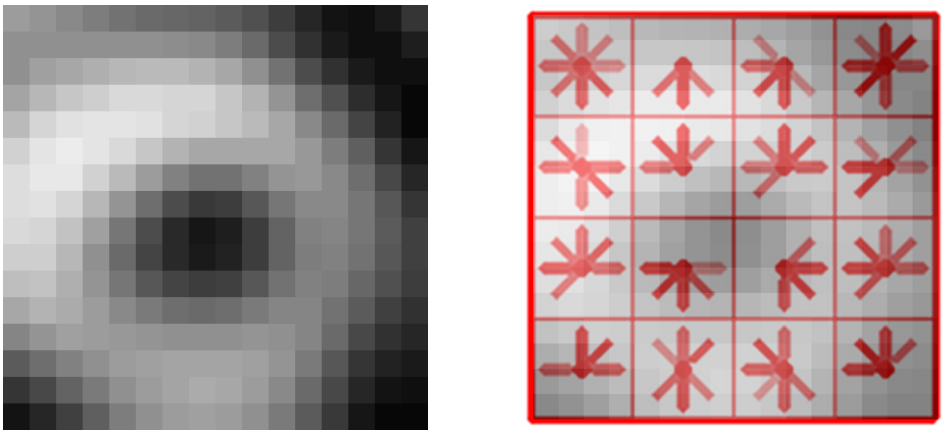

Figure 2.1: Left: the original patch used to calculate the SIFT descriptor; right: the patch with the histograms overlaid

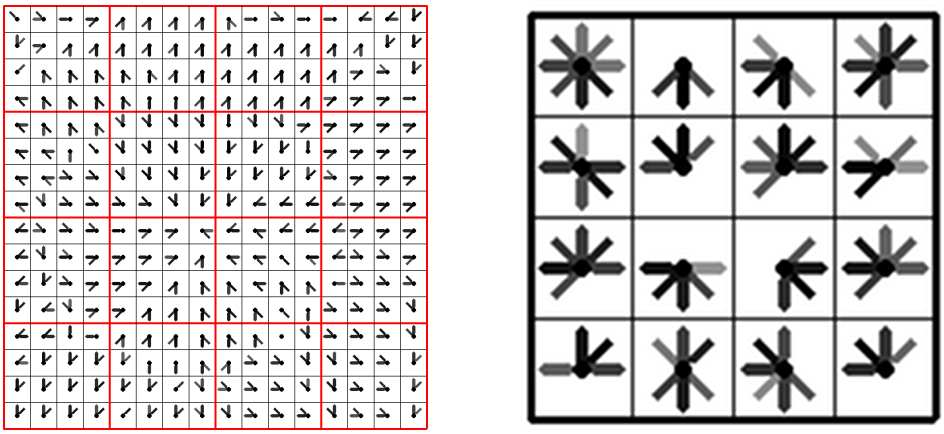

Figure 2.2: Left: the gradients orientations at each pixel interpolated between the two orientation bins; right: the histograms produced after sampling over the grid

3. The Hessian is used in scale-space both to better localise keypoints, as proposed by Brown and Lowe (2002), and to eliminate keypoints lying on edges, based on Harris and Stephens (1988).

4. Keypoints with a poor contrast are removed to improve reliability.

5. Keypoints are assigned a second orientation if the second highest bin is within 80% of the highest, essentially forming two keypoints for matching. This improve reliability.

6. Orientation assignment occurs by the fitting of a parabola to improve the precision.

7. Gradients used in the descriptor are weighted using a Gaussian over the patch.

8. The descriptor no longer uses binned gradients at a higher scale; the new descriptor size is $4 \times 4 \times 8 = 128$ dimensions.

2.1.4 Later interest-point extractors

Mikolajczyk and Schmid (2002, 2004) propose a new interest-point extractor based on the Harris & Stephens detector for scale space, using a scale-adapted version the original detector. The authors also replace the associated "cornerness" metric with the maxima of the Laplacian-of-Gaussian for scale-selection. Note that the maxima of the multi-scale Harris detector vary in location across scale, for the same corner.

This detector struggles with affine transforms, in which the scale change is greater in one direction than the other. This results in interest-points being detected at the wrong scale and hence in the wrong location. The authors propose an iterative solution which includes a *shape adaptation matrix*. This matrix applies an affine transform to the detector, with the largest eigenvalue set to unity to ensure correct scaling. Each iteration involves optimising the following parameters in turn:

shape adaptation matrix This is initially set to the identity matrix corresponding to a circular region.

integration scale This is the scale which maximises the Laplacian of Gaussian as before, though on the image transformed by the shape adaptation matrix.

differentiation scale This can simply be a fixed multiple of the integration scale, however a better value improves convergence.

spatial location This is determined as the local maximum of the Harris function as before, though in the image transformed by the shape adaptation matrix.

The authors show that their interest-point detector performs better than previous interest-point extractors, particularly for large viewpoint angle changes (approaching wide-baseline), however at the cost of computational complexity. However, Mikolajczyk, Tuytelaars et al. (2005) provides an in-depth comparison of affine interest-point extractors, showing that no single extractor performs best in all circumstances, and that a combination is preferable.

Bay, Tuytelaars and Van Gool (2006) introduce the Speeded Up Robust Features (SURF) interest-point extractor and local descriptor, which aims to perform comparably to or better than SIFT. Whereas SIFT uses an approximation of the Laplacian of Gaussian, SURF uses an approximation of the determinant of the Hessian. This is achieved by turning the discrete and truncated second order partial derivatives of the Gaussian function into box filters. The advantage is that convolutions involving box filters can be rapidly calculated using integral images (Viola and Jones, 2001). Convolution over multiple scales is achieved not by repeated convolution, as in SIFT, but by simply increasing the size of the box filters. The keypoints are generated as extrema in this scale-space, using a $3 \times 3 \times 3$ non-maximal suppression, and their location is improved using a similar approach to SIFT.

The orientation assignment occurs through using Haar wavelet responses in the vicinity and at the scale of the keypoint, which approximate gradient orientations as used in SIFT. These too can be calculated efficiently using integral images. (There is a version, "upright"-SURF, which skips this step.) SURF also includes a local descriptor, outlined in the next section.

2.1.5 Later hand-crafted local descriptors

Lazebnik, Schmid and Ponce (2005) present two new descriptors which are rotationally invariant, the *spin image* and the Rotation Invariant Feature Transform (RIFT) descriptor. The spin image is based on earlier work (Johnson and Hebert, 1999), and involves building a 2D histogram based on distance from the centre of the patch and the pixel intensity value. The RIFT descriptor is based on SIFT but using concentric circular pooling regions. During histogram binning, the gradient orientation is compared not with the canonical orientation but with the direction from the centre of the patch. This results in rotational invariance.

Mikolajczyk and Schmid (2005) present a performance evaluation of local descriptors for greyscale images, as in earlier work (Mikolajczyk and Schmid, 2003). In addition, the authors propose a new descriptor called the Gradient Location and Orientation Histogram descriptor (GLOH), which is an extension of SIFT. GLOH differs from SIFT in using a log-polar rather than rectangular grid for spatial pooling (gradient histogram binning) There are a total of 17 pooling regions compared to the 16 of SIFT, and there are 16 rather than 8 orientation bins for each pooling region, resulting a $17 \times 16 = 272$ dimension vector (compared to 128 for SIFT), which is reduced in size using principal component analysis (PCA). In the majority of experiments, GLOH outperforms SIFT; the exceptions are scale changes for which SIFT performs best and JPEG compression for which PCA-SIFT (Ke and Sukthankar, 2004) performs best.

Bay, Tuytelaars and Van Gool (2006) (as well as introducing the SURF interest-point extractor) also introduce a new local descriptor. Whereas SIFT bins gradient orientations over 4×4 sub-regions of a square grid, SURF uses Haar wavelet responses, as in the orientation assignment stage, i.e. a horizontal and vertical wavelet. Instead of assigning them into a histogram based on orientation, SURF simply sums the horizontal and vertical wavelet responses and absolute versions of these two responses, over each sub-region, resulting in a 4D vector for each sub-region. This yields a descriptor with 64 dimensions. The authors show that SURF performs comparably or better than SIFT, yet runs about three times quicker.

Tola, Lepetit and Fua (2010) develop a descriptor which builds upon the robustness of GLOH and SIFT, designed to improve the performance of wide-baseline matching. The descriptor is named "DAISY" due to the appearance of its pooling regions. All pooling regions are isotropic Gaussian distributions, which permits efficient computation through separable convolution, and through repeated convolution to cover a range of standard deviations. The descriptor shape is close to the learnt descriptor of Winder and Brown (2007) (see next section), with identically sized Gaussian pooling regions lying on concentric circles, however hand-crafted for computational efficiency rather than learnt for matching performance. The descriptor is also strongly connected to geometric blur (Berg and Malik, 2001) (see Section 3.0.1), as the pooling regions increase in size with distance from the centre.

The authors demonstrate that the "DAISY" descriptor performs better than SIFT and SURF on two datasets. In addition to developing a descriptor, the authors use occlusion masks estimated by expectation–maximisation (EM) to improve the depth map estimation using the graph-cut based reconstruction algorithm due to Boykov, Veksler and Zabih (2001). The authors demonstrate that this shows visually similar results to a leading algorithm (Strecha, Fransens and Van Gool, 2006), even when using lower resolution images.

Calonder, Lepetit, Strecha et al. (2010) produce a method for generating a binary descriptor for an interest-point, a descriptor composed of a collection of bits rather than floating point numbers. The authors' descriptor is known as the Binary Robust Independent Elementary Features (BRIEF) descriptor. This allows faster comparison through the Hamming distance (Hamming, 1950), rather than metrics such as the L^2 norm which are slower to calculate.

Differing from other binary descriptor methods which take an existing descriptor such as SIFT, SURF or "DAISY" and then compact it into a binary descriptor (Brown, Hua and Winder, 2011; Calonder, Lepetit, Fua et al., 2009; Tuytelaars and Schmid, 2007), the authors aim to build a binary descriptor directly, by comparing the pixel intensity values between pairs of points (as used by Ozuysal et al. (2010) for corner detection). The authors show that the resulting descriptor outperforms SIFT and SURF in image sequences for which rotation invariance is not required (since BRIEF is orientation sensitive), and

can be calculated and matched many times faster.

Leutenegger, Chli and Siegwart (2011) introduce the Binary Robust Invariant Scalable Keypoints (BRISK) binary descriptor. The overall pipeline first uses the Adaptive and Generic Accelerated Segment Test (AGAST) corner detector (Mair et al., 2010), which is an improvement upon the Features from Accelerated Segment Test (FAST) corner detector. Whereas the FAST corner detector uses a ternary tree, the AGAST corner detector uses a binary tree, with an enhanced configuration space and the ability to use adaptive tree switching. In addition, the optimisation includes a cost based on CPU access times (register vs cache vs main memory) for each node. The authors show the AGAST corner detector to be more efficient than FAST.

The BRISK interest-point extractor is based on AGAST but also provides non-maximal suppression over scale, to provide scale invariant interest-points. As with the BRIEF descriptor, the authors use simple pixel intensity value comparisons. These intensity values are used both to estimate the orientation of the interest-point, through estimating the dominant gradient orientation, and to provide the binary descriptor. The authors show that BRISK performs competitively to SIFT and SURF, while running over three times faster.

Arandjelović and Zisserman (2012) propose a slight variation on SIFT, RootSIFT. Since the vectors forming the SIFT descriptor have a unit L^2 norm, comparison between descriptors is dominated by larger histogram values. To address this, the authors propose to replace use the Hellinger Kernel,

$$H(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{n} \sqrt{x_i y_i} \tag{2.2}$$

This can be achieved by simply using the element-wise square root of L^1 normalised SIFT descriptors as drop-in replacement descriptors, in any framework which already uses the Euclidean distance. The authors show that results in a performance boost over the original descriptors.

2.1.6 Learning local descriptors

Ke and Sukthankar (2004) modify the final stage of SIFT, the calculation of the local descriptor, to produce PCA-SIFT. The local descriptor is formed by lifting horizontal and vertical gradients over a 39×39 grid over the image region specified by the interest-point. This results in a vector of length $2 \times 39 \times 39 = 3042$, which is normalised to a magnitude of unity to reduce the effects of varying illumination. Finally, the descriptor is formed by applying PCA to reduce the vector to a much smaller size e.g. 20 dimensions. The eigenspace for PCA is calculated from patches across a wide range of images, therefore the descriptor is learnt on an unsupervised basis. PCA-SIFT shows better matching accuracy than SIFT in many circumstances, though is less robust to scale and orientation assignment errors.

Rublee et al. (2011) improve upon the BRIEF descriptor, producing a binary descriptor known as the Oriented FAST and Rotated BRIEF (ORB) descriptor. The pipeline detects corners using the FAST corner detector (Rosten and Drummond, 2006; Rosten, Porter and Drummond, 2010), which uses machine learning to produce a faster detector than previous work (Rosten and Drummond, 2005; Rosten, Reitmayr and Drummond, 2005), building a ternary decision tree. Since the FAST corner detector does not produce a measure of "cornerness", the authors use the Harris corner measure (Harris and Stephens, 1988) to filter the corner detections. In addition, since FAST does not offer scale detection, the authors run the detector on many levels of an image pyramid. Furthermore, to generate an orientation the authors use the method due to Rosin (1999). ORB is an orientation invariant version of the BRIEF descriptor: the pixel locations are in effect rotated based on the orientation, using a precalculated lookup table.

Whereas the pixel pairs used for the binary tests in the original BRIEF descriptor were randomly sampled, ORB benefits from the use of learning. The authors show that ORB performs better than or comparably to SIFT and SURF, while running much faster.

Brown, Hua and Winder (2011) and Winder and Brown (2007) build upon the previous machine learning techniques and aim to learn optimal low level transformations for interest-point matching. In contrast to previous approaches, the authors use a labelled dataset of 3D

patches, centred on interest-points, for the learning task. In order to learn parametric descriptors, the authors minimise the area under an ROC curve (ROC-area) using Powell's method (Press et al., 1992).

For non-parametric descriptors, the authors use different objective functions, which they minimise over a vector of weights. The minimum can be found through solving a generalised eigenvalue problem, though requires regularisation due to the high dimensionality of the vector. The authors find that the best parametric pooling regions resembles those of the "DAISY" descriptor (Tola, Lepetit and Fua, 2008, 2010). The best non-parametric embedding results in a slightly inferior performance, however fewer dimensions. The overall best performances comes from combining the optimal parametric spatial pooling with learnt dimensionality reduction, in this case either GLOH or "DAISY-like" pooling regions followed by PCA.

Trzcinski and Lepetit (2012) aim to improve the performance of binary descriptors such as BRIEF, BRISK and ORB, producing a new binary descriptor known as Discriminative BRIEF (D-Brief). The authors calculated each binary bit from projections of intensity values of a patch. The projection is limited to a dictionary of projections which can be applied to the image patch efficiency, e.g. through box rectangle and Gaussian filters, through integral images and convolution.

The authors optimise the descriptor on sets of training data so as to encourage the binary bits to be equal for matching patches and non-equal for non-matching patches, with a sparsity inducing L^1 norm term to minimise the number of projections used. They show that descriptor outperforms other binary descriptors, while having a very short length of only 32 bits.

Trzcinski, Christoudias, Fua et al. (2013), Trzcinski, Christoudias and Lepetit (2015) and Trzcinski, Christoudias, Lepetit and Fua (2012) present a new binary descriptor, BinBoost, learnt using Boosting (Schapire, 1990). Boosting is an algorithm which combines many "weak learners" (efficient but inaccurate classifiers) into a strong classifier based on training data. In this case, the weak learners are pixel intensity differences as in BRIEF, ORB and BRISK, as well as oriented gradient based learners based on Ali et al. (2012)'s object detector. These are close to those used in SIFT but can be calculated rapidly using integral images.

The authors using a modified form of boosting (compared to Adaptive Boosting (AdaBoost) (Freund and Schapire, 1995)) which is more suited to minimise correlation between the weak learners and hence to learn an effective binary descriptor. They find gradient-based learners to be the most discriminative and show BinBoost to be the best performing binary descriptor, an that it also outperforms SIFT.

Simonyan, Vedaldi and Zisserman (2012, 2014) use convex optimisation to learn local descriptors, specifically regularised dual averaging (RDA) (L. Xiao, 2010). This is in contract to earlier approaches (Brown, Hua and Winder, 2011; Trzcinski, Christoudias, Fua et al., 2013; Trzcinski, Christoudias and Lepetit, 2015; Trzcinski, Christoudias, Lepetit and Fua, 2012) which are not guaranteed to reach a global optimum. Once a patch, covariant with an interest-point, has been lifted, the descriptor is calculated as follows:

1. Apply a Gaussian blur, as used by Brown, Hua and Winder (2011)

2. Extract local intensity gradients at each pixel, following by a binning into eight orientation channels, as first used in SIFT

3. Normalise gradient magnitudes across the entire patch extracted from an interest-point support region (patch), based on the authors' novel quartile statistic

4. Spatially pool the gradient magnitudes, independently for each orientation channel, using isotropic Gaussian pooling regions, as in Tola, Lepetit and Fua (2010) and Brown, Hua and Winder (2011): each pooling region forms an eight-dimensional feature set, i.e. one for each orientation channel.

5. Apply a linear weighting to each pooling regions, or a matrix for dimensionality reduction

The Gaussian pooling regions are selected from a large number of candidates, which are reflection-symmetric. They also lie on rings, as in the "DAISY" descriptor, but are not necessarily rotation symmetric. The first objective, that used to learn a weights set of weights, $\mathbf{w}$, over

pooling region candidate rings, is

$$\underset{w_n \geq 0 \forall n}{\arg\min} \left\{ \sum_{\substack{(\mathbf{x},\mathbf{y}) \in \mathcal{P} \\ (\mathbf{u},\mathbf{v}) \in \mathcal{N}}} \ell\big(\mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x},\mathbf{y}) - \mathbf{w}^\top \boldsymbol{\phi}(\mathbf{u},\mathbf{v})\big) + \mu \|\mathbf{w}\|_1 \right\} \tag{2.3}$$

where $\mathcal{P}$ is the set of positive matches, with $(\mathbf{x}, \mathbf{y})$ being two matching patch pairs, $\mathcal{N}$ is the set of negative matches, with $(\mathbf{u}, \mathbf{v})$ being two non-matching patch pairs,

$$\ell(z) = \max(0, z + 1), \tag{2.4}$$

is a hinge loss function aiming to force the distance between two non-patching pairs to be at least unity distance greater than between two matching pairs and $\boldsymbol{\phi}(\mathbf{x}, \mathbf{y})$ returns a vector of squared distances between the spatially pooled gradient magnitudes, one for each ring of pooling regions. The final term, $\mu\|\mathbf{w}\|_1$, is an L^1 regularisation term, designed to induce a large fraction of zero values in $\mathbf{w}$. In effect, this selects a small subset of pooling region rings from a large candidate set.

The authors apply a similar optimisation on the subset of pooling region rings to learn a matrix for dimensionality reduction, using the nuclear norm as a convex surrogate. The authors show that the local descriptor outperforms all previous local descriptors in terms of accuracy.

Han et al. (2015) introduce a convolutional neural network (CNN)-based approach for interest-point or patch matching, MatchNet, which utilises a two-tower structure. This comprises of two AlexNet like CNNs (Krizhevsky, Sutskever and Hinton, 2012) known as "feature networks", with tied weights, each of which has as an input a separate patch. The output of both is input to a single "metric Network" which outputs a probability that the two patches match and is composed of three fully connected layers and a softmax output. The whole network, consisting of the features networks and metric network, is trained using stochastic gradient descent (SGD) and cross-entropy loss.

At run-time (for prediction), the feature network can be used in isolation to generate a local descriptor for each interest-point or patch, with the metric network then used to match the pre-calculated local descriptors. The network outperforms previous local descriptors in terms of matching performance.

Zagoruyko and Komodakis (2015) test multiple CNN based approaches including an architecture similar to Han et al. (2015). For training, the authors use a hinge-based loss rather than cross-entropy loss and introduce an L^2 regularisation term. The authors find the best network to be a CNN without separate "feature networks" but simply with two channels, one for each patch of the two patch input, or alternatively a two-tower structure in which both towers have both patches as in input, but the towers differ in the scale of the patches. These networks also outperform previous local descriptors.

Simo-Serra et al. (2015) differ from Han et al. (2015) and Zagoruyko and Komodakis (2015) in their use of a CNN to produce descriptors and Euclidean distance for matching, rather than a non-linear network. This is important for facilitating the use of existing efficient methods for local descriptor matching that use (approximate) Euclidean distance. Instead the authors use Euclidean distance trained with a hinge embedding (or contrastive) loss (Mobahi, Collobert and Weston, 2009). In addition to a different loss function, the authors use hyperbolic tangent units rather than rectified linear units (ReLUs) to achieve non-linearity.

The authors train using SGD, however at each epoch, the authors use "mining" of difficult patch pairs to improve training, inspired by the "hard negative mining" approach of Felzenszwalb, Girshick et al. (2010). As with the networks of Han et al. (2015) and Zagoruyko and Komodakis (2015), the network outperforms all previous non-CNN local descriptors, in terms of matching performance. However this network has the benefit of providing a drop-in replacement for descriptors using Euclidean distance matching.

Paulin et al. (2015) introduces a local descriptor based on a convolutional kernel network (CKN) (Mairal et al., 2014). In contrast to a CNN in which the convolutional layers use filters composed of learnt weights for the convolution, the CKN layers use kernels for the convolution. Since such kernels is intractable, the authors do not use this kernels directly. Instead they learn a CNN embedding which approximates this kernel, using SGD, which enables two separate networks to operate on the two patches independently.

The patches used in training need not be labelled and hence this is a form of unsupervised learning. The output of a layer implementing

the approximation can either be vectorised to form a descriptor or input into another layer to produce a multi-layer CKN, with the final output producing a descriptor. The authors experiment with different inputs: raw red, green and blue (RGB) values, pre-processed RGB values and oriented gradients. The authors show comparable or better performance compared to supervised CNNs.

Balntas, Riba et al. (2016) introduce a triplet loss for training a CNN to produce local descriptors, in contrast to the pair based loss functions in earlier work. This is inspired by Hoffer and Ailon (2015), who shows that a CNN can learn more effectively using triplets rather than pairs. The triplet loss involves three patches, an anchor patch, a positive patch which matches the anchor patch, and a negative patch which does not match the anchor patch. There are two such triplet losses, a margin ranking loss and a ratio loss.

The authors use a simple CNN with only two convolutional layers, improving efficiency over earlier CNNs used to produce local descriptors. Nevertheless, the authors show improved performance over all previous local descriptors in all but one case using both loss functions, and in spite of running at least 10 times faster than the previous CNNs, and at a comparable speed to BRIEF. Training with a margin based loss generally results in better performance than the ratio loss for nearest neighbour matching, while the ratio loss performs better for match/non-match classification.

Yi et al. (2016) introduce the Learnt Invariant Feature Transform (LIFT). Compared to previous work which uses CNNs for descriptor calculation alone, or in one case (Moo Yi et al., 2016), for orientation assignment, LIFT provides a back-to-back CNN pipeline consisting of detection of interest-points, orientation assignment and descriptor calculation, while still allowing back-propagation for end-to-end training. The network is trained as a four-branch siamese network, with four patch inputs: two matching patches, one non-matching patch and one patch without an interest-point (for training the detector stage only). The branches include a detector network which generates orientations of interest-points, an orientation estimator which predicts their orientations and a descriptor network, which outputs a descriptor. The authors show that their overall pipeline outperforms all previous approaches in terms of matching score.

2.1.7 A new benchmark and a note on previous benchmarks

Balntas, Lenc et al. (2017) highlight the issues of previous local image descriptor benchmarks, which are as follows:

1. Lack of diversity in the images and patches
2. Problems of reproducibility and fairness, e.g. due to different choices of interest-point extractors and patch extraction scales which often results in conflicting results on the same data sets
3. Lack of the diversity in the tasks used for benchmarking, i.e. classifying patch pairs as matching or non-matching, matching images and patch retrieval

The authors create a new benchmark using image sequences either taken from a camera or from previous datasets. They generate patches from each reference image using multiple interest-point extractors, and map them to other images in the image sequences using the ground truth homography. To simulate the noise of interest-point extraction process, the authors perturb the mapping i.e. they apply a random transformation including rotation, anisotropic scaling, and translation.

2.1.8 Per interest-point learning

In addition to the approaches in the previous section, which seek to use learning globally to improve performance, there are other approaches which seek to use learning per interest-point. Furthermore, this is the approached used my work in Chapters 3 and 4.

Lepetit and Fua (2006) formulate interest-point matching as a classification problem. Their aim is to achieve a multi-class classifier which is efficient at run-time, through effective training. The authors use randomised trees (Amit and Geman, 1997) for this purpose because of their suitability for handling multi-class classification efficiently. The purpose is to classify interest-points and hence find a number of interest-points at test-time associated with a given object, which can then be used to identify that object's pose.

The authors use a small number of images of the given object, augmented by a number of synthesised views, for training the classifier. In

addition, they only train and classify the most reliable interest-points. The authors train many trees and the of the output of the many trees is averaged at test-time, with the maximum average output providing the classification. The tests used for each node of a tree are based on binary tests involving intensity values (as used in, but pre-dating, BRIEF, BRISK and ORB) following Gaussian smoothing of the patch. The authors show that this approach is suitable for identifying the pose of an object in real-time, in a manner which is computationally more efficient than storing a multitude of SIFT descriptors and using efficient methods based on the Euclidean distance.

Ozuysal et al. (2010) build upon this earlier work, replacing trees with flat structures referred to as "ferns". Whereas in the earlier work, the probabilities were based on the average over trees, the authors here use a Semi-naive Bayesian approach (Zheng and Webb, 2005): the dependencies between binary tests are modelled within a fern, but assumed to be independent between ferns. Each fern is composed of a number of binary tests, which are all evaluated (unlike randomised trees where the tests performed depend on the path through each tree) to provide a single binary string. This string is used to provide a probability of a given class based on the classical probability yielded from interest-points and their associated binary strings at train time (the likelihood), with a uniform Dirichlet prior for regularisation. A similar approach to the earlier work is used to provide a synthetic training set.

The authors show comparable performance to SIFT for detecting planar and 3D objects, while being able to run faster. In addition, the authors show the suitability of their system for simultaneous localisation and mapping (SLAM).

Gupta and Mittal (2008) recognise that the SIFT description is robust to misalignment (localisation error) and to linear changes in pixel intensity, but is not robust to non-linear changes in intensity. They aim to provide a descriptor which is robust to these, while still remaining robust to misalignment. The approach begins by discovering regions which are sufficiently light or dark in the patch, known as extremal regions. Next the authors calculate the distance transform (Fabbri et al., 2008) from the region boundaries and find point pairs between light and dark extremal regions which lie sufficiently far

from a boundary, or existing points, using a greedy algorithm. These point pairs are used at run-time to match interest-points. Using this per interest-point approach, the authors show a performance improvement over SIFT, particularly under illumination changes, image blur or compression.

Balntas, Tang and Mikolajczyk (2015) proposes a method for generating a binary descriptor which is optimised for each image patch independently and online. The authors demonstrate that performance can be improved by using a different binary descriptor for each image patch, however storing multiple descriptors for each patch is impractical. Instead the authors proposed to store for a given patch, a binary descriptor (based on BRIEF) (as in earlier work) and a bit mask of equal length. Matching is then performed using a modified and symmetric Hamming distance.

The authors combine a global offline optimisation of the binary descriptor, with run-time learning of the binary mask. The offline optimisation of the descriptor involves taking many possible binary tests, as in BRIEF, ranking each test based on the inter-class variance (variance between non-matching patch pairs), and finding an uncorrelated set o highly ranked tests. At run-time, the learning occurs for each patch using descriptors calculated from geometrically transformed versions of the patch: this can be achieved efficiency using a fixed number of affine transformations and lookup tables. The bit mask is then set to include only binary tests which remain invariant under all the geometric transformations.

The authors show performance improvements over the BRIEF and ORB descriptors, while still remaining competitive in terms of efficiency. In addition, the performance is comparable with that of SIFT, despite the much faster runtime.

Uzyıldırım and Özuysal (2016) propose a two step approach for matching interest-points with binary descriptors. The first step is to find, for a given interest-point, the top N nearest neighbours based on the Hamming distance with other binary descriptors, instead of only the nearest neighbour. The second step is to use the random "ferns" approach of Ozuysal et al. (2010) using the existing binary tests, i.e. those that represent individual bits in the descriptor for the probability look-ups, which are learnt as part of the offline learning. Whereas

in the earlier work, the match score is based on the probability alone, here, the match score is the sum of the negative Hamming distance and the logarithm of the probability. The authors show that the use of the two-step approach improves matching performance, in terms of improving the number of correct matches between two images, with an overhead of around 5% above using the nearest neighbour alone.

2.2 Image Classification and Object Detection

The many different approaches to image classification and object detection can, practically but perhaps arbitrarily, be split into those that use deep learning and those that do not and are said to use "shallow learning". Deep learning is a form of machine learning involving the learning of a model or function which is hierarchical (Goodfellow, Bengio and Courville, 2016) and has many layers in the hierarchy.

The is no standard definition of "many", hence the border between shallow and deep learning is not well-defined. Within Computer vision, many authors make a distinction between the use of "handcrafted" features and those generated through end-to-end training, assigning the former features as belonging to "shallow learning" methods (Chatfield, Simonyan et al., 2014; LeCun, Bengio and Hinton, 2015; Szegedy, Toshev and Erhan, 2013). I follow that distinction within this section and commence with a review of shallow learning methods.

2.2.1 Shallow Learning Methods

Prior to 1963, algorithms focused on detecting 2D patterns. Roberts (1963), on the other hand presents an algorithm for detecting 3D objects composed of planar faces from line drawings. The output of the algorithm is a 3D descriptor of all the models, which can be visualised in 3D.

Line extraction from line drawings occurs by calculating differentials in the image and thresholding the result to generate a rough edge image. Through local regression of these points, the algorithm generates longer lines. The algorithm also uses heuristics to connect adjacent lines to create longer lines, splitting lines into sections and

to complete the line drawing. Next, the algorithm discovers the polygons within the image and identifies parts of the 3D model with the assumption that all parts either lie on the ground plane or in each other. The output of the algorithm can be visualised as a line drawing, and for this, the author includes a novel algorithm for projecting the objects and removing hidden lines, based on volume intersection tests. This algorithm relies on strong assumptions about the objects, e.g. all objects having planar polygon faces.

Fischler and Elschlager (1973) proposes a part-based embedding for objects such as faces, which has fewer assumptions. The embedding is based on two metrics:

1. a metric for evaluating a score for locating a part at a given location in an image (the local embedding cost), e.g. based on the pixel intensity values

2. a metric for evaluating a score for the relative location of two parts (the "spring" embedding cost), is akin to placing springs between the two parts which have an increasing cost as the spring is stretched or compressed.

In general, the computational time to determine the part locations which minimise the cost grows exponentially with the number of parts. However, the computational time grows linearly if all the parts are connected in an acyclic chain such that all parts are connected by only two springs each, with two ends of the chain being unconnected to each other. Other springs can be added to this acyclic chain to serve as heuristics, which improves the approximate solution but cannot yield the global optimal.

The authors demonstrate the algorithm using experiments in which human annotators label parts of a face and the spring embedding was set to be either infinite cost or zero cost based on the feasibility of positioning. This further reduces the computational expense as infinite distance edges can be removed from the graph as it is being built.

Simard, LeCun and Denker (1993) identify a weakness in using K-nearest neighbour classification with Euclidean distance, in a metric space based on pixel intensity values, for classifying handwritten digits

(as in Section 1.5.3.1). The Euclidean distance is highly sensitive to simple transformation such as translating or scaling the digits, or rotating by a small amount. Therefore, the authors propose a distance metric which is invariant to such transformations.

Transforming an image of a handwritten digit results in a new image which lies on an manifold in the metric space used for classification. The dimensionality of the manifold is equal the number of parameters which specify the transformation. When calculating the distance between two images, the authors propose the use the shortest distance between the two respective manifolds rather than Euclidean distance. When the transformations include rotation, scaling and translation, the new distance is rotation, translation and scale invariant.

Unfortunately, there is no analytical expression for these manifolds. Therefore, the authors approximate the manifold by using the tangent spaces of the manifolds. The shortest distance between the tangent spaces can be obtained by solving a least squares problem. This provides a distance metric, with an efficient analytical solution, which yields approximate invariance to the transformations. The tangent distance provides a better level of invariance for the task compared to the Euclidean distance and therefore improves accuracy.

2.2.1.1 Later approaches using statistical models

Many approaches for object detection rely on statistical models. Belongie, Malik and Puzicha (2001, 2002) recognise objects by matching shapes through co-ordinate transforms. This is inspired by D'Arcy Thompson's *On Grown and Form* (Thompson, 1942).

The algorithm first obtains uniformly sampled locations along a "shape" (for general images, this can be yielded through use of an edge detector), yielding a set of points. The authors then calculate a shape context descriptor for every point, by binning the relative location of every other point into a histogram using a log-polar grid. This forms a descriptor for matching two points and the authors use the χ^2 statistic as the distance metric. The total cost between shapes is the sum of all these distances for a permutation of point matches which minimises the total cost. To achieve scale invariance, the descriptor can be calculated based on distances normalised by the mean distance between all point pairs in the grid. Additionally, rotational invariance can be achieved by rotating the grid based on the tangent of the shape at each point.

To allow for some deformation, the authors use the thin plane spline (TPS) model (Duchon, 1977; Meinguet, 1979). Having identified matching point locations, the authors find a transform which minimises the *bending energy* of the transform. To match one shape against another, the authors use the shape context descriptor matches and TPS model for three iterations, using transformed point locations for the pair matching after the first iteration. The final cost is the weighted sum of the shape descriptor costs between the shapes (in both directions for symmetry), *after* the transformation, plus the bending energy and (optionally) a local appearance cost.

The authors use K nearest neighbour classification and show state-of-the-art performance on the *MNIST database of handwritten digits* (LeCun, Bottou, Bengio and Haffner, 1998) (the MNIST database) and a shape silhouette database. In addition, they show its use for 3D object recognition and trademark retrieval.

Fergus, Perona and Zisserman (2003) build upon earlier part-based models (Burl, Weber and Perona, 1998; Weber, Welling and Perona, 2000a, 2000b), additionally modelling the variability of appearance, which is learnt simultaneously with the relative positioning of parts. The benefit is that the model can handle objects with high variability of appearance compared to geometric arrangement, or vice versa. In addition, parts are discovered using an interest-point extractor (Kadir and Brady, 2001).

An object is modelled to have a number of parts, each with appearance, relative positioning (shape), relative scale and possible occlusion. The desired calculation is the ratio of the posterior of an object detection over a background detection for a given set of interest-points and their local descriptors. With equal priors, this collapses to likelihood ratio of a positive detection over a background "detection".

The probability terms involve various distributions including Gaussian for appearance and shape, uniform for scale and a Poisson distribution over interest-points for each part. The authors perform training using EM, with an efficient search method for handling the $O\left(N^P\right)$ possible hypotheses for N interest-points and P parts. Similar efficient search techniques are used for recognition. The authors show that their method is superior to previous methods for object detection, despite no tuning for a specific dataset.

Crandall, Felzenszwalb and Huttenlocher (2005) present a statistical model framework for part-based object recognition. An object is considered to have a number of parts lying at respective locations in an image, modelled by a probability distribution. The appearance model assumes independence between pixels and that every pixel is either an edge or not, based on the output of the Canny edge detector. The authors assign a fixed probability for an edge to lie in a background and a location and part specific probability for the object itself.

The authors split the parts into two groups for each object, reference and non-reference parts. The spatial prior assumes conditional independence of non-reference part locations given the location of the reference parts. All probabilities are modelled as Gaussian distributions, enabling efficient detection of objects through convolutions and distance transforms. The authors train the models using six parts of fixed size, hand labelled per object class, and show favourable results over Fergus, Perona and Zisserman (2003), at the time, the leading local descriptor based approach.

Felzenszwalb and Huttenlocher (2005) provides an efficient algorithm for solving the energy minimisation problem presented by Fischler and Elschlager (1973) in the following case:

1. The connections between vertices (parts) form an acyclic graph (i.e. a tree).

2. The cost between two parts is based on a normalised Euclidean distance from the ideal relative part locations.

In contrast to previous work (Burl and Perona, 1996; Burl, Weber and Perona, 1998), the authors' method finds a global minimum.

The original minimisation can be cast into an undirected graph or a maximum a posteriori (MAP) problem. Whereas solving the general problem would still be NP-hard, the tree structure means that the problem can be solved in polynomial time for the number of possible locations for each part, by tracing from each leaf node to the root. Furthermore, because the authors restrict the deformation costs to be the normalised Euclidean distance, the problem can be solved in linear time using distance transforms (Karzanov, 1992).

The authors also demonstrate how learning can be performed using training images with parts already labelled using maximum like-

lihood estimation (MLE). This includes learning the graph structure. Furthermore, other (non-minimal solutions) can be sampled from the posterior efficiently using Gaussian convolution to handle the normalised Euclidean distance in the (negative exponential) space of the MAP solution. The authors demonstrate efficient matching of faces, even when two parts of the face are occluded.

Amit and Trouvé (2007) introduce the concept of patchwork of parts (POP) models. The features used by the authors are binary oriented edge features (Amit and Geman, 1999) which are robust to changes in lighting conditions and small deformations. An instance of a deformable POP model, is composed of a number of parts which can be shifted relative to the object and combined using a "patchwork" like operation in which the parts are averaged together in areas close enough to the centre of the part.

The authors use a joint Gaussian prior for modelling the shifts and classification occurs by finding the class and shifts which maximises the posterior. Object detection occurs by looping over all possible locations and training of individual POP models occurs using EM, with a greedy algorithm for learning mixtures of POP models. The authors show state-of-the-art performance on the MNIST database and high performance for zip code recognition and face detection.

2.2.1.2 Local descriptor based approaches

Other approaches use local descriptors to assist object detection, some of which also use statistical models. A special class of local descriptor based approaches, known as bag-of-words methods, are outlined in Section 2.2.1.3.

Leibe, Leonardis and Schiele (2004, 2008) and Leibe and Schiele (2004) provide an interleaved object detection and segmentation framework. The object detector works as follows:

1. Extract interest-points and their local descriptors within the images

2. Match each local descriptor to a codebook of previously found interest-points and local descriptors by finding any cluster which is within a given threshold distance (the clusters are built using an efficient version of agglomerative clustering.)

3. Use every entry in the cluster to assign a vote for the object centre in 3D space (location and scale)

Each entry of the codebook, previously learnt from training examples, contains a relative location and scale between the interest-point an the centre of the object it was trained on. In the event of multiple clusters matching the training example with a threshold, these are both stored with the appropriate weighting. In a voting procedure is known as *Probabilistic Hough Voting*, each entry casts one or more weighted votes into a 3D voting space of location and scale and this is used to locate the likely location and scale of the object.

Once the most probable location has been found, the interest-points which resulted in a vote for this location can be used to determine a rough segmentation of the object, thus identifying the bounds of the object. In addition, if the patches in the codebook all contain segmentation masks, the segmentation masks can be overlaid on each interest-point to provide a pixel-wise segmentation.

The authors show that their algorithm performs well for object detection, surpassed by one or two other algorithms such as Mutch and Lowe (2006). The main contribution, however, is the use of segmentation to improve object detection and localisation.

Tombari, Franchi and Di Stefano (2013) introduce a new local descriptor, the Bunch of Lines Descriptor (BOLD), for detecting textureless objects. BOLD features are calculated from a collection of line segments, which have already been extracted from an image using the LSD algorithm (Von Gioi et al., 2010). The base local descriptor is calculated for a pair of nearby line segments based on the angles between each line segment, and a third line connecting their midpoints. Every line segment has a descriptor, which is the combination of the base descriptor the that line segment and a number of the closest line segments. This local descriptor is a 2D histogram calculated from binning the value of the angles across the line segments.

The authors use an object detection pipeline close to that of Lowe (2004). They show that their local descriptor, used in this pipeline, outperforms all others on their own textureless dataset and performs favourably on a 3D textureless object dataset.

2.2.1.3 Bag-of-words Methods

Bag-of-Words (bag-of-words (BoW)) methods in Computer Vision are analogous to bag-of-words models for text document classification (Joachims, 1998; McCallum, Nigam et al., 1998; Nigam, Lafferty and McCallum, 1999). Just as the presence of certain words in a document, regardless of their ordering or position, can indicate the topic of the document, so the presents of "visual words" in an image can identify the object present, without any reference to the spatial positioning of the visual word.

Sivic and Zisserman (2003, 2006) introduce the use of BoW methods for searching for objects in videos. The pipeline operates as follows:

1. Identify interest-points from video frames using both the "Shape Adapted" (SA) extractor of Mikolajczyk and Schmid (2002) and MSER (Matas et al., 2004)

2. Track interest-points between frames

3. Extract SIFT descriptors for each interest-point

4. Produce a histogram by assigning each descriptor based on the closest of a number of cluster cluster centres, formed using k-means clustering

5. Average the histogram over each interest-point track and apply "term frequency-inverse document frequency" (*tf-idf*) weighting to produce a descriptor

As in document classification, the authors remove the 5% most common and 10% least common visual words.

At run-time (searching), a user specifies a sub-part of the frame which is used to create a query descriptor. Ranking with the database tracks is based on the normalised scalar produce between query and database descriptors. In addition, the authors perform reranking while enforcing spatial consistency to remove spurious matches. The authors demonstrate efficient retrieval of objects within a video, using their methods.

Csurka et al. (2004) similarly introduce the use of BoW methods for classifying objects in images. The pipeline operate in similar fashion, as follows:

1. Identify interest-points in an image using the affine-invariant extractor of Mikolajczyk and Schmid (2002)

2. Extract SIFT descriptors for each interest-point

3. Produce a histogram by assigning each descriptor based on the closest of a number of cluster centres

4. Classify the histogram using a support vector machine (SVM) or a Naive Bayes classifier

The authors provide a new benchmark and show that the SVM outperforms the Naive Bayes classifier.

Grauman and Darrell (2005) provide an alternative approach to the use of k-means clustering, providing a kernel which can be used to match two unordered sets of interest-point descriptors, which may also differ in cardinality. The kernel, knows as the Pyramid Match Kernel (PMK), accounts for matching between individual features without an intractable matching step. The authors demonstrate the use of their kernel in an SVM for object recognition and result in performance comparable to the state-of-the-art, however much more computationally efficient.

Lazebnik, Schmid and Ponce (2006) also use the PMK and an SVM for classification, however they operate in the 2D image space rather than the descriptor space. The authors assign the descriptors to clusters, as common in previous work, and calculate the PMK in 2D space for each cluster based on the location in the image. The authors do not use interest-points, instead they use two feature types: oriented edge points (points where the gradient magnitude for a given orientation exceeds a threshold) and SIFT descriptors calculated densely over the image. The authors demonstrate state-of-the-art performance on scene category recognition and performs competitively to best performing object recognition methods.

Nister and Stewenius (2006) provide a more scalable version of Sivic and Zisserman (2003) through the use of a vocabulary tree. As in the earlier work, the authors use MSER interest-point and SIFT descriptors. However, the authors build a tree using hierarchical k-means clustering.

At run-time (searching), all descriptors traverse the tree and each provides a score based on the weighted number of descriptors which pass through each node. The weighting is set so as to result in the a *tf-idf*. The resulting vector is compared to those in the database based on the L^1 norm between the normalised query and database descriptor. The authors demonstrate real-time performance searching over 50 000 CD album covers.

J. Zhang et al. (2007) present a comprehensive study of object category classification using interest-point detectors, local descriptors, clustering and SVM classification. The authors achieve state-of-the-art performance on many object recognition datasets such as PASCAL Visual Object Classes (VOC) 2005 (Everingham, Zisserman et al., 2006) using both Harris-Laplace and Laplacian interest-point extractors, SIFT and spin image local descriptors and a one-vs-one SVM with an Earth Mover's Distance (EMD) kernel (Rubner, Tomasi and Guibas, 2000) for classification. They also discover that the use of affine-invariant interest-point extractors does not improve performance and that the use of background features does not improve classification using their approach however the use of varied backgrounds in the training set improves generalisation.

2.2.1.4 SVM based methods

The next set of approaches use SVMs for classification. A linear SVM seeks to learn a linear decision boundary which divides feature space into two so as to maximise the margin between the boundary and the closest training example of each class, the support vectors. In the event that this is not possible, the algorithm seeks to provide the best possible compromise. Extensions to the linear SVM allow it to handle classification of more than two classes and to create non-linear decision boundaries by using kernels to transform the feature space into a higher dimensionality.

Papageorgiou and Poggio (2000) develop a classier which uses filters based on Haar wavelets (Mallat, 1989). These filters are square and each pixel of a square is set to either -1 or 1 throughout. A combination of many such filters can provide a orthonormal basis for an image, however the authors do not limit the number of filters to those forming an orthonormal basis but also translated versions of them, resulting in an overcomplete dictionary. In addition, the authors remove filters which yield information at too coarse or too fine a scale, as these are uninformative. The authors use many of these filters to extract a feature vector for classification in an SVM, chosen because of its high performance on a small training set, forming an object detector. The kernel used is quadratic. To locate objects, the authors run the detector at many locations in the image and at multiple scales, i.e. it is a sliding-window detector. The detector performs well but takes 20 minutes per frame. To improve the speed, the authors propose to select only a subset of the wavelet features and to reduce the number of support vectors used in the classification.

Mohan, Papageorgiou and Poggio (2001) build upon Papageorgiou and Poggio (2000). They too use Haar wavelet features and SVMs for person detection, however they first use a number of individual component classifiers to independently locate parts, e.g. the head, legs, left arm and right arm, followed by a *combination classifier*. The component classifiers, SVMs with quadratic kernels, are independently trained on sub-images containing each respective part. The authors use an overall sliding window, specifying the position and scale of the possible human. The component filters are then only run at locations and scales relative this window for which a correct detection is feasible (based on the set of training images), which results in more efficient detection.

The maximum scores for each component, i.e. the distance from the decision boundary output for each SVM, form the vector for the combination classifier, a linear SVM. If a given part is not found, leading to a negative output from the SVM, a zero is inserted into the vector to make the classifier robust to occlusion of a part. This SVM is trained based on pre-processed positive examples and negative examples from applying the component classifiers over images containing no people. The authors compare the two-tier classifier to Papageorgiou and Poggio (2000) and show that it outperforms the

latter system.

Uijlings et al. (2013) develop a method, known as *selective search* for generating possible bounding box locations within an image. The aim is to achieve a high recall rate, as missing an object will prevent it from being detected in the image. On the other hand, false positives can be ruled out by a subsequent class-based classifier. The primary purpose is to reduce detection time, however selective search also improves object localisation and possibly improves training by narrowing down training inputs to the bounding box proposals.

Selective search has a hierarchical design because objects often exist as a hierarchy: as an example, wheels are objects but can be part of another object such as a car. To achieve this, the authors use bottom-up segmentation: they begin with small regions and recursively group smaller regions into larger regions, based on their similarity, until the whole image becomes a single region. The authors then use regions at every scale to produce rectangular bounding box proposals.

In order to support a large variety of objects and images, selective search uses a diverse set of strategies to assess the similarities between regions. The similarity measure between two regions is a weighted combination of four measures, which measure the following, respectively:

colour similarity of colours between the two regions

texture similarity of the texture using SIFT-like features between regions

size fraction of image left outside the two regions: this means that the similarity measure prefers to merge smaller rather than larger regions to avoid one region dominating

fill how well the regions fit into each other: if two regions wrap around each other (or indeed, if one is fully surrounded by another) they will both predict a similar (rectangular) object bounding box; this similar measure favours merging such regions

Regions are produced from a combination of different strategies, i.e. the algorithm is run multiple times with different weightings over measures and using different colour spaces, to provide robustness to different images.

As well as the selective search algorithm, the authors demonstrate the use of the selective search proposals for object recognition using an SVM based classifier. The authors use BoW features and the PMK as in Lazebnik, Schmid and Ponce (2006) but with a finer pyramid division and with colour-SIFT descriptors (Van De Sande, Gevers and Snoek, 2010), making use of the efficiency gains from selective search.

2.2.1.5 SVM based methods using HOG

Dalal and Triggs (2005) introduce the concept of Histograms of Oriented Gradients (HOG) for human detection. The HOG detector uses oriented histograms, similar to those of SIFT (see Section 2.1.3), to detect humans in images. However, whereas the SIFT framework provides a spare representation, the HOG descriptor is calculated over a dense grid composing the whole image. The detector operates on a sliding window composed of cells, with each cell corresponding to a sub-region of the SIFT descriptor. Within each cell, the values of local gradients are summed within a histogram, based on their orientation, as in SIFT. However, normalisation occurs over a collection of cells, known as a block, rather than throughout the window.

The authors use two arrangements for cells, rectangular (R-HOG) and circular (log-polar) cells (C-HOG). They use an SVM classifier to classify the descriptor for each window. The authors show that the HOG detectors outperform previous detectors.

Felzenszwalb, Girshick et al. (2010) and Felzenszwalb, McAllester and Ramanan (2008) build on earlier deformable part model work (Felzenszwalb and Huttenlocher, 2005), which is often outperformed by simpler models such as HOG and BoW models. The authors identify the success of HOG in terms of providing effective features and develop a part based model composed of a "root" HOG filter and a number of part HOG filters. The authors aim to perform training using only object (and not part) bounding boxes, hence the locations of parts form latent variables in the training. For this purpose, the authors use a latent-SVM (LSVM) which is equivalent to the SVM formation of Andrews, Tsochantaridis and Hofmann (2003) for multiple-instance learning.

To extract features at multiple scales, the authors use a feature pyramid, which is based on lifting HOG features from an image pyramid. In all cases, the part features are computed at twice the resolution

of the root features. The resulting model is known as a Deformable Part-based Model (DPM) The overall matching score for the DPM is the sum of the appearance cost, from comparing HOG features, and the (spring like) deformation costs between the root and part filters.

In order to perform matching, the authors calculate the score in a given image for many possible root filter locations. Given the location of the root filter, is possible to efficiently calculate the best placement for every part filter (and hence the matching score) using distance transforms as in the authors' earlier work (Felzenszwalb and Huttenlocher, 2005). In the case of a mixture model, an object is specified with multiple models (components), and their match score is calculated independently. This step is followed by bounding box prediction, using a linear least squares regression between the part locations and the overall bounding box.

The authors use their LSVM algorithm for training. This algorithm alternates between learning the parameters of the SVM given the latent variables (part locations) for a positive training instances and relabelling the positive training instances. This requires careful initialisation since it is no longer convex. To improve training efficiency the authors use data-mining: the algorithm retains a cache of hard examples to use for training and refreshes this periodically using the latest trained model. The authors test their performance on PASCAL VOC 2006–2008, achieving top performance in nine out of twenty categories and second best in eight.

Farhadi, Endres and Hoiem (2010) aim to provide an object detection system that can identify unseen objects based on their similarity to other seen objects, for example, an unseen four-legged animal based on having seen other four-legged animals. The authors train DPMs (Felzenszwalb, Girshick et al., 2010; Felzenszwalb, McAllester and Ramanan, 2008) for both parts (e.g. leg and wheel), categories (e.g. four-legged animal or vehicle part) and basic objects (e.g. dog or cat) and show a fair degree of generalisation to unseen objects based on the trained category detectors.

The authors then train two localisers: one for parts and one for animals. They do this by storing the scale and locational difference between each object, part or category detection and the ground truth, e.g. the scale and locational offset for both a "head" and "dog" detection compared to the ground truth bounding box for the dog.

At test time, all the stored offsets allow for probabilistic voting of the animal or vehicle bounding box. The authors combine the votes using a clustering-based algorithm and logistic regression. In addition to localising objects, the authors use a graphical model, learnt by EM, to infer attributes on objects based on the detections. The authors show improved performance for detection animals and vehicles (including unseen animals and vehicles) using their framework compared to simply using object detectors trained on basic objects.

2.2.1.6 Boosting based methods

Boosting (Schapire, 1990) is an algorithm which combines many "weak learners" (efficient but inaccurate classifiers) into a strong classifier based on training data. At each iteration, the algorithm seeks to find the best "weak learner" for all the training examples, adding it to the classifier. However, the weighting over training examples changes between each iteration based on those correctly and incorrectly classified by previous "weak learners", with the aim that each successive "weak learner" improves the overall classification performance.

Viola and Jones (2001) build upon the work of Papageorgiou and Poggio (2000), producing the Viola-Jones framework. They too use features inspired by Haar wavelets, but allow the set of all two-rectangle, three-rectangle and four-rectangle features in addition to Haar wavelet based features. To speed up calculation of these "Haar-like" features, the authors introduce integral images. The integral image makes it possible for two-rectangle, three-rectangle or four-rectangle features to be calculated at a given position using six, eight or nine image value lookups respectively, rather than by summation as in Papageorgiou and Poggio (2000).

In addition, the authors use AdaBoost (Freund and Schapire, 1995), modified so that each weak learner uses only a single rectangular features. Training, therefore, results in selecting one feature at a time and allows much for efficient training over a large set of possible features (180 000) compared to an SVM. In addition, it results in a classifier which requires the calculation of only a subset of the possible features.

Although this results in a very efficient classifier, it is not efficient to use it for object detection as it involves running the classifier over every possible position and scale in the image. The authors propose an alternative: a cascade of classifiers. The authors use AdaBoost

to learn a number of small and efficient classifiers with the threshold of each test modified to minimise the probability of a false negative. These resulting classifiers can be stacked in a cascade such that if any classifier in turn rejects a sub-window, no further processing occurs. This enables the majority of locations, which are negative windows, to be eliminated rapidly, while the remaining promising sub-windows are processed further with the slower but more accurate classifier.

The authors show that their detector operates fifteen times faster than the fastest known previous detector for face detection, while performing with similar accuracy. The detector often returns multiple detections of a given face, but this can be resolved by taking averages of the bounding box locations across all overlapping detections.

Dollár et al. (2009) add additional feature types to the Viola-Jones boosting framework. Their framework operates on many channels which are images computed from linear or non-linear transforms of the input image such that the output pixels corresponds to the feature located on each input pixel. The simplest channels are grayscale intensity value and different colour spaces, while more complex types include oriented Gabor filters (Malik and Perona, 1990), Canny edges and gradient features. Each weak learner is based on the sum of a rectangular region in a given channel, sampled randomly from the huge space of possibilities.

The channels are translation covariant, which allows for rapid lookup using integral images, as in the Viola-Jones framework, hence giving the name *integral channel features*. Histogram-based features such as HOG features can also be generated by generating a channel for each orientation and then using integral images to look up the histogram (Porikli, 2005). With a total of ten channels (which include the CIE 1976 (L* u* v*) colour space (LUV) and HOG based features), the algorithm outperforms previous algorithms for detecting pedestrians, except for DPM, while running much faster.

Benenson, Mathias, Tuytelaars et al. (2013) improve performance over Dollár et al. (2009) by optimising the design, including applying the multi-scale approach of Benenson, Mathias, Timofte et al. (2012). The authors discover that using all available features in the training (requiring a large computational expense at train time) and global normalisation of feature channels improves performance, while

other modifications show little or no performance boost. As a result of the two successful changes, the authors achieve a significant performance boost.

Wang et al. (2013) introduces a new feature pool for which to supply weak learners, known as regions and regionlets. As with other boosting approaches, these can operate on any integral channel feature. Each region is rectangular and contains one or more regionlets and supplies a single feature for which to use as a weak learner. The regionlets are also rectangular, and are identical in shape across a given region. The purpose of having multiple regionlets within the region are to allow a given feature, e.g. perhaps corresponding to a hand of a person, to be located in different places. In addition, the size of the regionlet allows for different levels of deformation for the feature. Instead of obtaining a feature from the rectangular region alone, the feature is the maximum obtained by any one regionlet.

Compared to previous boosting approaches, which form sliding window detectors, the authors use selective search (Uijlings et al., 2013), a class-independent algorithm which generates a large number of candidate windows, which hopefully cover all the required objects, for a given image. These are known as object bounding box proposals and the detector is run on each of these. To handle the varying aspect ratio of the bounding box proposals, the regions and hence the regionlets are always located relatively in position and scale to the bounding box.

The pool of available regions and their regionlets is huge, however the authors sample a subset of possible regions and then randomly generate only one set of regionlets for each region. These then form the candidate weak learners for training. The authors use RealBoost (Schapire and Singer, 1999) rather than AdaBoost for training.

The authors show that their regionlets based detector outperforms all other approaches on PASCAL VOC 2007, where it performs best on sixteen out of twenty categories, and on VOC 2010, where it performs best on fourteen out of twenty categories. The authors also show state-of-the-art performance on the ImageNet dataset (Russakovsky et al., 2015).

2.2.1.7 Self-similarity based methods

A special class of classifiers and detectors operate using internal self-similarity. Whereas other approaches assume that representations share similar properties such as colour or texture, self-similarity based approaches aim to identify classes with a similar geometric layout.

Shechtman and Irani (2007) propose a descriptor based on local self-similarity. This descriptor can be calculated on any pixel location and the local descriptor is based on the correlation with a patch centred on this location and a larger region around the patch. The result of the correlation is turned into a vector by pooling the maximum value over many pooling regions in a log-polar grid to achieve a 2D descriptor. An extension to the descriptor for matching videos is calculate the correlation through time as well, resulting in a 3D descriptor.

The authors do not use the descriptors in isolation, but produce an ensemble of them using the algorithm of Boiman and Irani (2007), yielding a "star-graph" model for every template. The authors extract descriptors over a dense grid, however remove descriptors with little or too much self-similarity, as these not informative. They also calculate descriptors over many scales. The authors show that their descriptor outperforms others such as SIFT and GLOH when detecting objects using a template such as an image of a flower. The authors also show the suitability of their approach for matching human poses from a sketch and for ballet turns in a video from a video template.

Chatfield, Philbin and Zisserman (2009) build upon Shechtman and Irani (2007) to produce a framework more suitable for image retrieval in large databases. They use the same descriptor concept for the pooling region grid) but normalise the descriptor and use the Probabilistic Hough Voting approach of Leibe, Leonardis and Schiele (2004) to detect objects from the descriptor instances. The authors also extract descriptors densely. However, they eliminate descriptors which are less descriptive by retaining only those whose nearest neighbour is sufficiently closer than their second nearest neighbour in descriptor space. In addition, they use the BoW architecture of Sivic and Zisserman (2003) to allow efficient matching of whole images, before using the Hough voting approach to localise the object. The authors show that the self-similarity descriptor outperforms

traditional local descriptors such as SIFT for some abstract classes, such as "bottles" and "swans" and propose that the descriptor be used alongside other descriptors to improve performance.

2.2.2 Deep Learning Methods

Deep learning perhaps began as early as 1943, however the first approaches were proposed but not realised. McCulloch and Pitts (1943) proposed a neural network based on theoretical neurophysiology. The network assumes that all neurons can be connected to other neurons so as to either to activate or inhibit them and yielding an "all-or-none" response. Each connection is directed, and the network may be cyclic if the connections form a cyclic graph, akin to a recurrent neural network (RNN). Such a network is shown to be realisable through the authors' framework. The authors note that network can only compute "such numbers as can a Turing machine".

Hebb (1949) bridged the gap between neurophysiology and psychology. Of particular interest is the attempts to understand neurological function based on the reaction to drawn figures, from humans and lab animals. The work also includes hypotheses about how learning takes place based on experiments. Of particular notability is *Hebb's Rule*, which can be summarised as "neurons wire together if they fire together", in effect meaning that if one neuron tends to result in exciting another neuron over time, the efficiency that this occurs is likely to increase. Hebb (1949) and McCulloch and Pitts (1943) did not attempt to construct their proposed neural network using learning but provided inspiration for later neural networks.

Rosenblatt (1958) proposed the *perceptron*, devised as a hypothetical nervous system, as an analogue to biological systems. Every neuron has an "all-or-nothing" response, activating at a fixed threshold. The perceptron design has four layers: the "retina", the projection layer, the association layer and the responses. Connections from the retina to the projection layer are distributed such that the number falls exponentially away from the centre, while the rest of the connections are random. The first three layers' connections are acyclic, while the connections between the association layer and the responses form a cyclic network.

The author proposes three different learning systems, each involving the increase in the "gain" of active neurons and possibly a change in the "gain" of inactive neurons when given stimuli are applied. The result is that there is more chance of the active neurons activating connected neurons in future, which follows Hebb's Rule. This alone can result in the perceptron converging to produce a stable output for two classes, and hence it forms an unsupervised learning algorithm. The author also considers spatial organisation rather than random assignment of neuron connections, to assist in contour detection.

Ivakhnenko (1968) and Ivakhnenko and Lapa (1965) train neural networks using the *Group Method of Data Handling* (Farlow, 1981). The whole neural networks produce a mapping of continuous data to a single output,

according to a high order polynomial. Each neuron calculates a polynomial of a fixed but low order, perhaps second order. Through building a hierarchical network of neurons, the neurons combine to model a much higher-order polynomial.

Each layer of the network is constructed by solving many least squares problems on small subsets of the training data. A number of solutions which provide the best estimate are retained and used to form the neurons in the layer. The same procedure takes place on subsequent layers, using the outputs of the previous layer. Consequently, the depth of the network increases with each iteration, until performance no longer improves on the validation data. Depending on the definition of "deep", this may be considered to be the first example of deep learning (Schmidhuber, 2015).

Kohonen (1972) considers the idea of "correlation matrix memories" in which a correlation matrix encodes the relations between data vector space and a key vector space. This corresponds to a network in which the key and data vector space form the inputs and every element of the correlation matrix forms an output. The matrix may be "incomplete", yielding a reduced number of outputs. If the key vectors are orthogonal, perfect recall is achievable, while otherwise there is cross-talk. The author proposes the use of such a matrix for supervised and unsupervised learning. In this respect, the incomplete matrix is similar to an "inner product" layer of an neural network.

Von der Malsburg (1973) refutes the proposal by Hubel and Wiesel (1963) that the circuitry of the primary visual cortex is genetically pre-determined. In during so, the author proposes a mechanism for self-organisation of visual cortex, simulated on a computer, consisting of 338 neurons. After training the model by exposing it to patterns of different orientations, individual neurons become sensitive only to one orientation, reflecting the behaviour shown by Hubel and Wiesel (1963). The connections between neurons are, for simplification, assigned a single "weight" parameter, the strength of a connection or synapse. The model increases the strength of a connection upon each activation, and then normalises over all the connections to the cell, resulting in selectivity and saturation. In this respect, the model performs unsupervised learning.

Fukushima and Miyake (1982) introduces the Neocognitron, perhaps the first convolutional neural network (CNN). The Neocognitron is a seven layer neural network, inspired by the work of Hubel and Wiesel (1962). The first layer is the input layer, and the remaining layers alternate between layers based on the "simple cells" (s-cells) of Hubel and Wiesel (1962) and layers based on the "complex cells" (c-cells). In practice, the s-cell layers act like convolutional layers, with inhibition. Each layer has a number of planes, corresponding to channels in modern CNN terminology.

The c-cell layers act like average or Gaussian pooling layers, also with inhibition. In effect, the layers smooth the input to each c-cell layer over a broader output, normalised based on the input across all planes (channels), which variables levels of saturation.

The authors learn the synapse weights for each position and plane by applying different stimuli, such as different letters to the Neocognitron, and updating the weights. This occurs by selecting the synapse (weight) which leads to the largest output for each plane and column and increasing the weight by a small amount. Since there is no supervision when the stimuli are applied, the Neocognitron is an unsupervised learning algorithm. Initially the weights in are set very small values and can only increase.

The authors design the network such that the size of each pair of layers decreases through the network, with the final layer only having one position. The authors show that, after showing many different numerical digits, the network results in only one output in the final

layer for each digit, even if the digit is slightly distorted.

2.2.2.1 Introduction of Backpropagation

Rumelhart and Hinton (1986) introduces a new method for the supervised learning of neural networks with multiple layers: backpropagation. This involves adjusting the weights of a neural network to minimise a loss function, e.g. the average sum squared difference between the desired output vector and the actual output vector, across many input vectors. Each unit (neuron) in the neural network, operates as a non-linear function applied to a weighted linear sum over inputs. Having calculated the partial derivative of the loss function with regard to the outputs units, the partial derivatives of the loss with respect to the weights can be calculated using the chain rule. Moreover, in the case of networks of many layers, the partial derivative of the loss with respect to any weight can be calculated through repeated use of the chain rule, or backpropagation.

Training the network involves a forward and backward pass through the network. A new input is applied to the network, and this propagates forward through the network, with the output of each unit stored for later calculation of partial derivatives. Once the loss has been calculated for the input, the partial derivatives with respect to the loss propagate back through the network. The network can then be improved through gradient descent: a simple method is to alter each weight by an amount proportional to the derivative. A more complicate approach is to use "momentum" and calculate the change for each time-step and a weighted combination of the partial derivative at the time step and the change from the previous time-step.

LeCun (1989) and LeCun, Boser et al. (1989) introduce a CNN for classifying handwritten digits. Input images are sub-sampled to 16×16 greyscale pixel images. The first two layers of the CNN are convolutional, each with 5×5 kernels. On the first convolutional layer, the stride is two pixels, resulting in downsampling the output width and height to half of that of the input. Twelve different kernels result in twelve channel $8 \times 8 \times 12$ output. The second convolutional layer also has a twelve channel output using convolution from twelve kernels which are convolved over eight of twelve possible input channels. The convolutional layers are followed by two fully connected layers. In every case, non-linearity occurs using tanh units.

The authors train the CNN using backpropagation, SGD and a mean squared error loss function. The authors find the rejection rate (based on the different between the top two outputs) required to yield 99% accuracy is 12%, thus achieving state-of-the-art performance on digit recognition.

LeCun, Boser et al. (1990) modify the previous network for improved performance with fewer learnt parameters. The first convolutional layer is similar to that before, but with only four different kernels. Instead of downsampling using a stride, this layer is followed by a layer which takes an average of across every non-overlapping 2×2 set of pixels. The next convolutional layer is similar to before but has only four kernels, and is followed by a similar downsampling layer and a single fully connected output layer. The authors show that, for 99% accuracy, the rejection rate required is 9%, lower than the previous network despite the reduction in the number of kernels.

LeCun, Bottou, Bengio and Haffner (1998) describe a CNN based system for learning to recognise whole strings of text, rather than just characters, using end-to-end training through backpropagation. This is in contrast to previous methods which rely on individually trained modules e.g. separate modules for segmenting characters and for recognising them. To allow end-to-end training, the entire system is a feed-forward neural network, and is based on modules connected together as a directed graph with multiple branches, which determines the order of the forward and backpropagation stages. The system allows for modules which are not themselves differentiable, e.g. a multiplexer with internal switching (based on the input rather than parameters), as a gradient can still be backpropagated after the forward pass. The system allows training based only on the labelling of a whole string of characters, without individual segmentation.

The authors initially demonstrate a neural network which inputs and outputs a graph, known as a Graph Transformer Network (LeCun, Bottou and Bengio, 1997), which starts with the output of a heuristic over-segmentation (HOS) algorithm (Breuel, 1994; Burges et al., 1992). Such an algorithm outputs many possible segments of handwritten text, based on applying cuts to the input text in different positions. The segments are connected as a directed acyclic graph (DAG) in which every path through the graph represents a feasible

segmentation. The authors input each possible segment into a CNN (e.g. as used in LeCun, Boser et al. (1990)), which generates a score for each class (digit). These scores are transformed into penalties for interpretation of each segment and combined with the penalties of the segmentation algorithm, producing an output graph. The string can then be determined by the lowest cost path through the graph.

The network is trained end to end through backpropagation and application of the *forward penalty* (Rabiner, 1989) as loss term. The authors demonstrate handwriting detection systems, as well as working system for reading the cash amount (in digits) on a cheque, with a correct recognition rate of 82%, a reject rate of 17% and an error rate of 1%. The system was in use at many banks from 1996. They also describe the use of their network for object detection, as presented in earlier work on face detection (Vaillant, Monrocq and LeCun, 1994).

Chopra, Hadsell and LeCun (2005) use a siamese CNN architecture (first used in Bromley et al. (1994)): two CNNs with shared weights. They use it face verification: matching a new face to a known face. The training is designed for the situation in which similar pairs of images are known, in this case faces of the same person, but the number of classes is large and a future input image might have a new class, i.e. the face of an imposter. The CNN structure is trained end-to-end on the using a *contrastive loss function*, and yields a multi-dimensional vector output (in the output space) for each input face. The effect of this loss function is to penalise distances in the output space of a network for similar images, e.g. faces of the same person, and to penalise two dissimilar inputs which are too close a distance in the output space.

The authors use SGD for the training. They demonstrate the system for face verification, in which a new face is compared to the mean output of many known faces of a subject, to verify the identity of the new face.

Hadsell, Chopra and LeCun (2006) use a similar architecture designed to produce a low dimensionality embedding, fulfilling the role of other dimensionality reduction techniques such as PCA, but capable of learning a wide range of non-linear operations. The authors show the use of the system for learning a shift-invariant low dimensional embedding which separates the digits 4 and 9, as well as an illumina-

tion invariant embedding for aeroplane images based on the azimuth and elevation of each plane (with no labelling except for similar and dissimilar pairs).

F.J. Huang and LeCun (2006) propose that CNNs, while excellent at learning features which are invariant, do not learn good decision surfaces. They also note that SVMs with kernels learn better decision surfaces but cannot themselves learn sufficient invariance for detecting objects at a high level of accuracy when operating on raw pixel intensity values. Consequently, the authors propose to combine the advantages of both CNNs and SVMs.

The authors first train a CNN independently, and then use the feature output of the final layer to learn an SVM with a Gaussian kernel. They show an error rate of 5.9% on a jittered version of the NYU Object Recognition Benchmark dataset (LeCun, F.J. Huang and Bottou, 2004) (NORB) when using both a CNN and an SVM compared to 43.3% when using just an SVM and 7.2% when using just a CNN. In addition, the dimensionality reduction of the CNN allows the SVM to operate quicker at both train-time and test-time.

Mrazova and Kukacka (2008) propose a "hybrid convolutional neural network" architecture in which radial basis function (RBF) neurons, rather than convolutional kernel neurons, form the "feature detection" layers. Their network is based on LeNet-5 (LeCun, Bottou, Bengio and Haffner, 1998), but uses Euclidean RBF neurons rather than a fully connected layer for the output layer. In this case, the main purpose is to produce an ASCII image output rather than a "1-of-N" coding output, for combination with a linguistic post-processor.

Another variation, the hybrid CNN, has a fully connected output layer while the "feature detection" layers uses RBF neurons. These RBF neurons reflect the distribution of the input data: in the ideal case, the input data forms clusters in the feature space of each layer and each RBF neuron lies in the centre of a each cluster. The main advantage of this is an increase in the rate of convergence and therefore faster training.

The "hybrid convolutional neural network" show worse performance than a (slightly modified) LeNet-5 on the MNIST database in all but one case (Gaussian Noise). However, the network can be trained in a fraction of the time.

Cireşan et al. (2010) note that previous benchmarks on the MNIST database show that multi-layer perceptrons perform worse than other methods including CNNs and neural networks combined with SVMs. The authors set out to test whether the poor performance is simply due to the lack of training of the multi-layer perceptrons as a result of the large training time requirement. They train a multi-layer perceptron on a graphics processing unit (GPU), obtaining a factor of forty increase in training speed. The resulting perceptron outperforms all other approaches, showing that the simple multi-layer perceptron can outperform more complex methods with sufficient training.

Ciresan et al. (2011) present a GPU implementation for CNNs. Differing from previous work which only allowed the forward stage to occur on the GPU, their implementation allows backpropagation to occur on the GPU, permitting training to occur wholly on the GPU.

The CNN architecture is similar to that of LeCun, Bottou, Bengio and Haffner (1998), but with a contrast-extracting layer (Fukushima, 2003) for NORB and max-pooling layers (Boureau, Ponce and LeCun, 2010; Scherer, Müller and Behnke, 2010): layers which return the maximum value over an area rather than the mean value. The authors show that the CNN architecture is able to achieve the same performance on the MNIST database as Cireşan et al. (2010) as well as state-of-art performance on NORB and CIFAR10.

2.2.2.2 Deep Learning Post AlexNet

Krizhevsky, Sutskever and Hinton (2012), showed state-of-the-art performance on ImageNet (Russakovsky et al., 2015) image classification and led to resurgence of interest in deep learning over shallow learning methods. The authors use a deep network, later known as AlexNet, consisting of five convolutional layers and three fully connected layers. The CNN network was designed to be trained on two GPUs, taking advantage of the increased memory. As a result, the network is split in two at the first layer, with interconnections after the second and fifth convolutional layers.

To increase training speed, the authors use ReLU units to achieve non-linearity rather than other functions such as tanh . In addition, the authors normalise the outputs of the ReLU units based on the

response of N adjacent layers, giving an output for each channel, i,

$$o_i(\mathbf{x}) = \frac{i_i(\mathbf{x})}{\left(k + \alpha \sum_{j=i-n/2}^{i+n/2} i_j(\mathbf{x})^2\right)^{\beta}}, \tag{2.5}$$

where i_j is the input of the jth channel and k, n, α, and β, are hyper-parameters. This is known as local response normalisation (LRN).

The authors use overlapping pooling: rather than pooling over non-overlapping regions, the regions are permitted to overlap. The width and height reduces through progressive layers due to the pooling: the size and overlap is such that the number of output units remains fewer than the number of input units. Finally, the authors use dropout (Srivastava et al., 2014) to reduce overfitting: each hidden neuron is set to zero 50% of the time which results in it not contributing to the forward or backwards passes.

The final layer has 1000 outputs with a softmax function corresponding to the 1000 ImageNet classes, and uses a multinomial logistic regression loss for training. The network is trained using SGD with momentum. The authors train the network using ImageNet with data augmentation: the training set is augmented by using many possible sub-patches within each image, as well as reflected versions and colour varied versions of the same image using PCA. The authors show state-of-the-art performance on the ImageNet Large Scale Visual Recognition Challenge (Russakovsky et al., 2015) (ILSVRC), outperforming the SIFT based approach.

Le (2013) uses a deep autoencoder-based network to learn features using unsupervised learning. The neural network contains three stages (layers) each with a local filtering sublayer, pooling sublayer and a local contrast normalisation (LCN) sublayer. The neurons of the local filter sublayers have connections limited to a local region as in a convolutional layer but the weights are not shared over all locations. The pooling sublayers calculate the L^2 norm rather than the maximum or mean value and the LCN sublayers normalise by subtracting the mean and dividing by the maximum value over a Gaussian local window.

Since there is no supervision, there is no label based loss. Instead, each local filter sublayer is trained with both encoding and decoding weights, based on the reconstruction error, with an additional term used to encourage sparsity. Training occurs using SGD.

The author shows that, in spite of the unsupervised learning, the best performing neuron is able to detect faces with 81.7% accuracy. The author also show that supervised learning of the same network achieves state-of-art performance on ImageNet.

Szegedy, Toshev and Erhan (2013) modify the architecture of AlexNet for localisation by replacing the final layer of the CNN with a regression layer. This layer produces a binary mask indicating whether a given pixel lies within the bounding box of an object. The mask is significantly smaller than the input image, as in the dimensionality of the later layers in AlexNet.

The network is initially trained for image classification as in AlexNet, with the weights kept before training for the new task. The loss function is a weighted L^2 distance between the output mask and a ground truth, and the authors used a variation of SGD (Duchi, Hazan and Singer, 2011) to minimise this.

The simplest scheme is to set the ground truth to either 0 or 1. A more complicated scheme is to train five separate networks corresponding to the full bounding box, and the left, top, bottom and right halves, respectively of the bounding box, and to set the values in the interval $[0, 1]$ based on how much the bounding box covers the receptive field of each pixel of the mask. In order to obtain more accurate bounding boxes from the output at test time, the authors' algorithm searches all possible bounding boxes and selects the one which best matches the five mask outputs.

The authors run the networks on three different scales of the image: the full image and two smaller scales. In addition, the authors use a number of sub-windows for the smaller scales and combine the output masks. After yielding the bounding boxes, the authors apply a refinement stage: running the networks on a window corresponding to these bounding boxes to refine them into more accurate bounding boxes. In addition, they use a network trained for image classification as in AlexNet, to ensure the class score is high enough based on the bounding box, and to apply non-maximal suppression as in DPM (Felzenszwalb, Girshick et al., 2010). The authors show state-of-the art performance on PASCAL VOC 2007, outperforming DPM.

Sermanet et al. (2014) present OverFeat, a CNN algorithm which combines classification, localisation and detection. The authors use a

network similar to that of AlexNet. However, contrary to AlexNet, in which max-pooling is performed over the whole of the final convolutional layer, the authors perform max-pooling over a window corresponding to different scales and locations within the input image.

Having trained a network for classification, the authors add a regression network to perform object localisation. This network shares the output of the convolutional layers used by the classifier layers and consists of two fully connected layers followed by four fully separate connected output layers corresponding to the four coordinates for the bounding-box edges. Only these later layers are altered during regression training, and an L^2 loss is used between output coordinates and the ground truth coordinates, for each output bounding box which covers at least half of a ground truth bounding box.

At test-time, the authors use a greedy strategy to merge bounding boxes together, based on averaging coordinates in the cases that two bounding boxes are sufficiently close in centre and overlap to each other. This differs from the traditional non-maximal suppression. The authors show state-of-the-art performance for object detection on ILSVRC 2013.

Chatfield, Simonyan et al. (2014) perform an evaluation of techniques for CNN performance and conclude the following:

1. Data augmentation improves the training of CNNs (and shallow methods): using horizontally flipped and cropped versions of the images improves classification performance.

2. Using a colour image input results in better performance than a greyscale input.

3. Fine-tuning a network (pre-training it on a larger dataset and then further training it on a smaller one using the weights learnt from pre-training for initialisation) improves performance, even when using a much smaller dataset for fine-tuning.

The authors also analyse three different CNN structures. The "medium" speed network, based on Zeiler and Fergus (2014), has become a standard network known as "VGG 1024".

Girshick et al. (2014) also use a CNN based on AlexNet, pre-training it for image classification and then fine-tuning it by training

it on bounding boxes containing objects. Their algorithm is known as "regions with CNN features" (R-CNN). The authors use selective search (Uijlings et al., 2013), a hierarchical segmentation method, to generate bounding box proposals: possible object locations within an image. At test-time, the input image is cropped to each bounding box proposal and then warped to fit the input size of the network. The output of the CNN, followed by a linear SVM, provides a score for each class and bounding box. Non-maximal suppression follows.

At train-time, the authors also use selective search to generate bounding box proposals, selecting positive and negatives proposals for fine-tuning based on the intersection over union (IoU) with the ground truth bounding box. After fine-tuning the network using the same loss as in AlexNet, the authors replace the final fully connected layer with a linear SVM trained on the same input.

The authors show state-of-the art performance on PASCAL VOC 2007 and 2010. As an extension, the authors also implement a regression model to improve the object bounding box prediction from the bounding box proposal. Inspired by DPM, they use linear regression, using the output of the final convolutional layer. This results in a further performance improvement.

He et al. (2015) improve the run-time performance of CNNs for object detection by running the convolutional layers on the whole image as in LeCun, Bottou and Bengio (1997) and Vaillant, Monrocq and LeCun (1994). They introduce a new network, the spatial pyramid pooling network (SPPNet). To handle the transition between the variable size output of the final convolutional layer and the fixed input size of the first fully connected layer, the authors introduce a new spatial pyramid pooling (SPP) layer based on Grauman and Darrell (2005) and Lazebnik, Schmid and Ponce (2006). This layer produces a fixed size output regardless of the input size. This results both in efficiency, as the convolutional layers need only be processed once, and better scale-invariance, as training can occur with the objects at different scales. The authors show a slight improvement in accuracy over R-CNN while running over twenty times faster. Alternatively the single scale version performs slightly worse in accuracy but sixty-four times faster.

Lenc and Vedaldi (2015) make R-CNN and SPPNet even faster and less dependent on selective search. They use a static set of region proposals, independent of the input image. This relies on the suitability of the CNN, specifically the final convolutional layer output, to perform accurate bounding box regression as in R-CNN. In addition, the authors also streamline the design by replacing the SVM with a softmax output and operating the SPP layer at a single scale. These both result in efficiency gains at the cost of a small drop in the CNN's performance. The authors demonstrate a sixteen times speed-up compared to SPPNet, mainly due to not using selective search.

Szegedy, Liu et al. (2015) introduces a new network structure, GoogLeNet inspired by the belief of Arora et al. (2014) the a network with sparse connections is more desirable than a dense one. Since it is computationally expensive to evaluate a sparse network compared to a dense network, the authors aim to approximate it locally using dense components. The authors introduce the Inception module in which different layers operate in parallel rather than connected to each other, with concatenation of the outputs into a single module output. The belief is that many neurons tend to fire together, and can therefore be joined together before reaching the next layer.

Since convolutions using larger kernels are expensive in terms of the number of calculations and weights, it is desirable to precede it with use dimensionality reduction, using a 1×1 convolution, i.e. convolving over the channels alone.

The authors use ReLU units to provide non-linearity and the overall network is twenty-two layers deep. To assist in the training of the deep network, the authors put additional classifier outputs mid-network, which are not used at test-time. For object detection, the network operates like R-CNN, i.e. using selective search for proposals. The authors show state-of-the-art performance on the ILSVRC 2014 detection challenge.

Simonyan and Zisserman (2015) introduce deeper networks than the original AlexNet and VGG 1024. The new networks rely on back-to-back 3×3 convolution layers rather than convolution layers with larger kernels. This has the result of reducing the number of parameters compared to larger kernel sizes. The new networks do not have any LRN, as it is shown to be counter-productive in this network

structure.

The new networks significantly outperform earlier networks for classification on ILSVRC-2014. The use of these networks can also lead to improved object detection, shown in later work.

Girshick (2015) introduces the "Fast Region-based Convolutional Network" method (Fast R-CNN) based on R-CNN and SPPNet. The author uses a region of interest (ROI) pooling layer, which is equivalent to the SPP layer of He et al. (2015) with a single scale: i.e. it performs max-pooling over a single $W \times H$ grid. The author also initialises the weights of the network to those pre-trained on ImageNet, but additionally fine-tunes the convolutional layers.

The author fine tune the network using a multi-task loss: as well as an output layer which yields a probability distribution for every possible class, using a softmax function, an additional output layer outputs four offsets, $\mathbf{t} = (x, y, w, h)$, specifying the offsets between the centre position, the width and the height of the object location with reference to the bounding box proposal. At test time, the outputs are used to refine the bounding box location.

The fine-tuning is performed by using ground truth bounding boxes with sufficient IoU with an anchor as positives, and those with a sufficiently small IoU as negatives; those whose IoU is neither sufficiently small or large are not used in training. While the loss over classes is the multinomial logistic regression loss, there is an additional loss placed over the offsets, and a hyper-parameter specifies the relative weighting of each losses. If $\mathbf{t}$ is the target offsets and $\mathbf{p}$ is the bounding box proposal, the loss, known as the smooth L^1 loss is

$$\sum_i \begin{cases} 0.5(t_i - p_i)^2 & \text{if} |t_i - p_i| < 1, \\ |t_i - p_i| - 0.5 & \text{otherwise.} \end{cases} \tag{2.6}$$

The author show state-of-the-art performance on PASCAL VOC 2007, 2010 and 2012. He shows that the bounding box regression is an important factor in the performance improvement, as is the ability to fine-tune the convolutional layers. Finally, he notes that for VGG16, it is better not to train an SVM on the final layer and to simply use the softmax output.

Ren et al. (2015) replace the bounding box proposal algorithm (selective search or an alternative) with a CNN called a region proposal

network (RPN). The resulting algorithm is known as Faster-RCNN. This is a CNN which, just like a bounding box proposal algorithm, takes an image input and outputs a number of region proposals.

The classification (object or not) output is trained with a multinomial logistic regression loss while the regression output uses the same smooth L^1 loss as Fast R-CNN. The detection network is identical to Fast R-CNN, using the output of the RPN instead of selective search. For efficiency, the two networks share the same convolutional layers.

The authors show state-of-the-art performance on PASCAL VOC 2007 and 2012, as well as running much faster than Fast R-CNN due to the removal of the selective search bottleneck. In addition, they show that that, although the RPN generates an order of magnitude fewer proposals than selective search can, these achieve a higher recall rate than selective search.

Liang and Hu (2015) use a recurrent CNN for object recognition (classification). The authors use recurrent connections for each convolutional layer such that each convolutional layer is a function of both the input and the previous time step output. In practice, the authors use unfolding: duplicating the convolutional layers over the number of time steps and introducing bypass connections. This turns the recurrent CNN into a feed-forward CNN with the same functionality, and allows it to be trained as a feed-forward CNN.

The authors show state-of-the-art performance on CIFAR100, the MNIST database and Google street view house numbers datasets. The performance on object detection datasets in unclear, and it is possible that with unfolding, the network would be too large for larger image sizes.

Redmon et al. (2016) present the "You Only Look Once" (YOLO) algorithm. This approach differs to earlier approaches (Girshick, 2015; He et al., 2015; Ren et al., 2015; Sermanet et al., 2014) in that it uses a single network to output both object bounding box proposals and class probabilities simultaneously. An input image is divided into a grid of cells, and each cell is responsible for predicting a number of bounding boxes and a confidence score that an object centre lies within the cell. The confidence score is the product of the probability of there being an object and the IoU. In addition, each grid cell predicts the probability of a given class conditional on there being an object.

The CNN is based on GoogLeNet and consists of twenty-four convolutional layers followed by two fully connected layers. The first twenty layers are pre-trained on ImageNet with a mean-pooling layer and a fully connected layer. Rather than ReLU the authors use a "leaky ReLU", which always yields an output even if the input is negative.

The loss function is a weighted sum of the following L^2 losses:

1. The bounding box location and the square root of width and height (to avoid favouring smaller bounding boxes) compared to the ground truth
2. The probability of an object for each bounding box
3. The class probabilities, conditioned on the object probability, for each class and bounding box

The authors network does not achieve the same accuracy as previous networks, but runs much faster and is suitable for real-time processing.

Szegedy, Vanhoucke et al. (2016) re-examine the Inception architecture used in GoogLeNet. The authors describe a number of design principles, to be observed in a new design, as follows:

1. Bottlenecks in the network in terms of dimensionality, a proxy for information flow, should be avoided; instead, it is best to gradually reduce the dimensionality through the network.
2. Increasing the number of activations per location yields faster training.
3. Reducing the dimensionality before convolution over a large patch can improve efficiency without information loss.
4. The width and depth of the network should be balanced.

The authors also suggest taking advantage of factorisation, e.g. replacing a convolutional layer with a large kernel with successive convolutional layers with smaller kernels (as in VGG16). In addition, they suggest aiming to double the number of filters while downsampling the width and height by half as a rule of thumb.

Following these principles, the authors produce a new network which is forty-two layers deep with a computational cost of only two-and-a-half times that of GoogLeNet. The new network shows state-of-the-art performance on the classification task of ILSVRC 2012.

He et al. (2016) show that the reason for the performance plateau associated with increasing the depth (number of layers) in a CNN is not due to overfitting, as previously believed, because the training error increases rather than decreases in this instance. Nevertheless, they state that depth alone could not be the issue, as identity layers would not alter the input and allow infinite (though meaningless) depth. Following this, they propose to introduce a residual building block which outputs the sum of the input and a collection of other layers. If the latter layers output zero, the block models the identify function.

The authors use the new building block design, to build networks with as many as 152 layers. The new network architecture achieves state-of-the-art performance on classification on ILSVRC and on object detection on PASCAL VOC 2007 and 2012, as well as the "common objects in context" dataset (Lin et al., 2014) (COCO).

Kong et al. (2016) alter the framework of Faster-RCNN. The authors note from previous work (Ghodrati et al., 2015) that earlier convolutional layers in a network are better suited for localising objects, while later layers achieve higher recall than earlier convolutional layers. The authors therefore aggregate the convolutional outputs at multiple layers rather than just using the final convolutional layer for both region proposal generation and classification. Similar to Faster-RCNN, the authors use an RPN to generate region proposals, in their case on the multiple layer output "Hyper Feature" space.

The authors use a similar network structure as in Fast R-CNN and Faster-RCNN for classification, i.e. two fully connected layers with dropout, however this is applied to the "Hyper Feature" space. In addition, this network also includes a bounding box region output, as in Fast R-CNN, to improve upon the regions generated by the RPN. The authors achieve state-of-the-art performance on PASCAL VOC 2007 and 2012 with a running time on par with Faster-RCNN.

Yang et al. (2016) identify a weakness in previous approaches e.g. R-CNN and its derivatives. After region proposal generation, there is a single classifier network (class-1) responsible for determining whether the proposal corresponds to an object or not has only one class output for "background", i.e. not an object. This means that it is difficult for the single "background" class to effectively cover close the whole range

of possibilities for background detections.

The authors instead add an additional network (class-2) after the convolutional layers which is used to classify all non-background objects classified by class-1 and to remove false-positives. This network is trained based on positive and negative classifications in the training data from the class-1 network and is trained to provide a one-vs-rest output for every object class. At test time, RPN proposals which are classified as not background by class-1 are then classified by class-2 to remove any false positives.

The authors show that this cascade design, using an additional one-vs-rest classifier network, improves detection performance. They too achieve state-of-the-art performance on PASCAL VOC 2007 and 2012.

3 Interest-Point Specific Geometric Blur Descriptors

This chapter involves the task of matching interest-points between images (see Section 1.2.3). This is achieved by calculating local descriptors for each interest-point and matching them using a distance metric such as Euclidean distance (see Section 1.2.3.2). The aim in this chapter is to improve the local descriptors when matching using the same distance metric.

Although interest-point extractors (see Section 1.2.3.1) such as Scale-Invariant Feature Transform (SIFT) offer a high level of repeatability (Mikolajczyk and Schmid, 2004), the position of interest-points extracted in other images (projected to the 2D space of the image) varies relative to the actual 3D feature. In other words, the position is only approximately covariant with the projected location of the 3D feature, as there is some misalignment (Lowe, 2004). The same is true of the scale and orientation estimated by the interest-point extractor.

Assume the existence of the following:

1. The presence of many source images
2. A method to transform an interest-point between any source image and a destination image
3. A criteria for two interest-points to be considered matching when transformed to a destination image

Interest-points extracted from the many source images (point 1) transformed to the destination image (point 2) form samples of a probability distribution over the destination image, for example Figure 3.2.

By matching these samples to interest-point extracted from the destination image using the criteria (point 3), these samples form a distribution specific to each interest-point extracted from the destination

Figure 3.1: Photograph of Bath Abbey

Figure 3.2: The distribution of interest-points when transformed from other images is indicated by the hue of each pixel

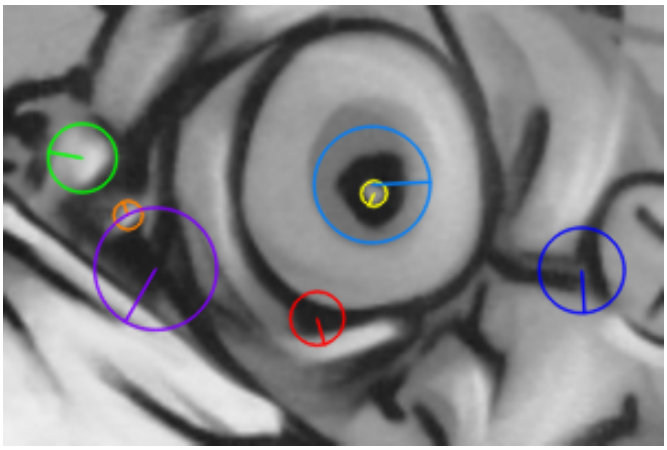

Figure 3.3: A set of interest-points extracted from an image

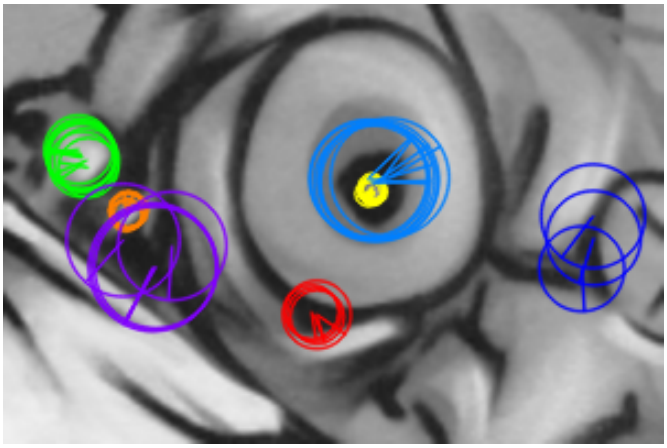

Figure 3.4: Sets of samples of matching interest-points from other images transformed to the image

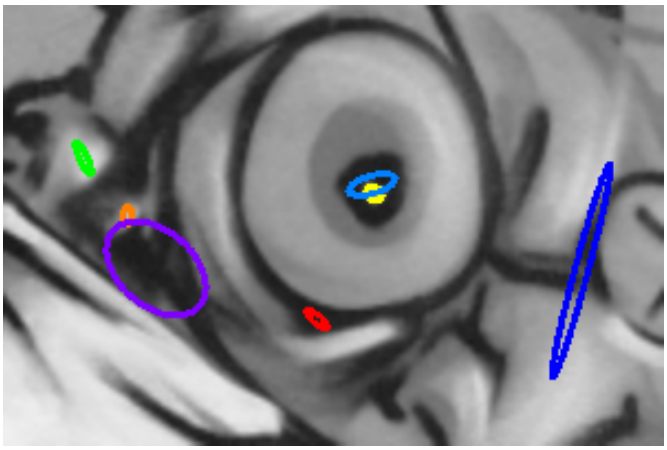

Figure 3.5: 99% confidence interval ellipses for the locational uncertainty distribution (position only) modelled as a 2D Gaussian distribution

image. I define this as the *locational uncertainty distribution* of the interest-point. In the case of SIFT, this distribution is over four dimensions: 2D location (x and y), orientation and scale.

Each interest-point has a unique locational uncertainty distribution, as Figures 3.3–3.5 show. This is an issue for the calculation of local descriptors (see Section 1.2.3.2), which must be robust in spite of the distribution. Many local descriptors use spatial pooling (see Section 1.6.2) to increase robustness (Bay, Tuytelaars and Van Gool, 2006; Lowe, 2004; Mikolajczyk and Schmid, 2005; Simonyan, Vedaldi and Zisserman, 2012; Tola, Lepetit and Fua, 2010; Winder, Hua and Brown, 2009). This includes rectangular or circular pooling regions, over each of which the algorithms calculate a weighted sum or weighted average of low level features (see Section 1.6.1).

All the aforementioned methods use an identical pooling strategy for every interest-point, which means the pooling is not optimal given the different locational uncertainty distributions between interest-points. The method which I present uses a different pooling strategy for each interest-point and is designed to improved performance by allowing for each interest-point's individual locational uncertainty, i.e. to account for the variation which can be seen in Figures 3.3–3.5. My method aims to estimate the locational uncertainty distribution for each interest-point in an image, from that single image alone, through the use of many synthetically warped versions of the image, in a manner similar to that used to enlarge the test set in Winder and Brown (2007). The transformations used for the synthetic warping are samples from a prior distribution, therefore this is an example of invariance from *a priori* knowledge (see Section 1.5.3.2).

Having estimated the locational uncertainty distribution for each interest-point, I use a Monte Carlo method to pool the low level features. This pooling is based on geometric blur, which I outline in the next section. I demonstrate the technique's performance against the descriptor used in SIFT for comparison purposes, however my method is suitable for any interest-point extractor and any low level feature.

3.0.1 Geometric Blur

Berg and Malik (2001) introduce the concept of *geometric blur* for template matching. The authors point out that the then standard strategy for making templates robust to *geometric distortion*, applying a uniform Gaussian blur throughout, is not the "right" thing to

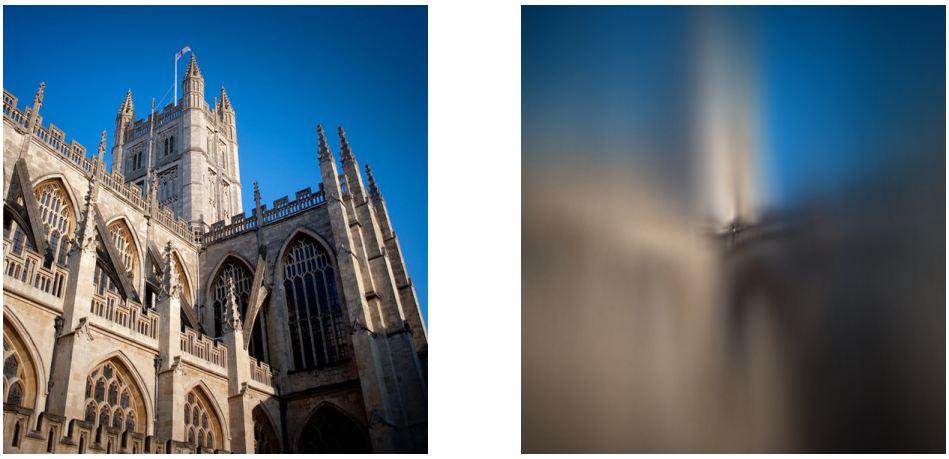

Figure 3.6: Left: original image of Bath Abbey; right: the image produced by averaging the colour values at each pixel location when the original image has been warped by a number of affine transformations centred on the image centre. This results in locations further from the centre of the image being more blurred.

do if the distortion is induced through observing the scene at different viewpoints. This is because a Gaussian blur corresponds to a uniform positional uncertainty throughout the image, whereas the positional uncertainty increases for points further from the centre of a window on the image. Figure 3.6 demonstrates this: to approximate the effect of observing the scene from multiple viewpoints, the original image has been warped by a number of affine transformations centred on the image centre (x-axis and y-axis scales independently sampled from a uniform distribution between 0.5 and 2.0 and rotation sampled from a uniform distribution between -5.0° and 5.0°). Berg and Malik (2001) propose an alternative blur: applying a blur which increases with distance from the centre of the window, which they refer to as a *geometric blur*. The authors demonstrate that this enables more accurate template matching compared to using a Gaussian blur.

Figures 3.7–3.9 demonstrate the principle of geometric blur applied to a 1D signal. Figure 3.7 demonstrates three signals which are versions of each other scaled around the x-axis. Consider the task of matching distorted versions of the sparse signal at the top of Figure 3.7 using normalised cross-correlation (Szeliski, 2010) with a template. Figures 3.8 and 3.9 shows two possible templates for matching this sig-

nal, which are versions of the signal blurred along the x-axis. The first uses a blur which is uniform along the x-axis and the second uses a geometric blur, which increases with distance from the origin. The geometric blurred template is better for matching distorted versions of the signal if the distortion is caused by scaling along the x-axis rather than translation.

Berg and Malik (2001) show that geometric blur is most effective when applied to sparse signals. In general, blurring a template results in a loss of high-frequency information and reduces its discriminative ability. However, a sparse signal, a signal which is zero-valued in most locations and whose non-zero locations are far apart from each other, allows blurring to be carried out to a much larger extent with little or no loss of information. This is demonstrated in Figures 3.10 and 3.11: the sparse signal can be reconstructed from its blurred version (with the assumption that it was sparse) while the dense signal can not. A similar principle applies to the discriminative ability when using normalised cross-correlation: if sparse candidate signals are blurred prior to attempting to match them, there is little or no loss of discriminative ability. However, the matching between signals is much more robust to errors in alignment and stretching and scaling of the signal.

3.0.2 Sparse Signals in Local Descriptors

Berg and Malik (2001) propose the use of oriented edge filters to generate sparse signals from images. Similarly, local descriptor extractors such as the one used in SIFT (Lowe, 2004) use gradient magnitudes binned into eight different orientations. In effect, this involves pooling eight different sparse 2D signals: the gradient orientation at each pixel is split into eight orientation channels; since each gradient is interpolated into the two closest orientations of eight, three-quarters (or more) of locations in each orientation channel will be zero. The use of pooling over sparse signals (the orientation channels) is crucial to the robustness of the SIFT local descriptor, compared to using pooled pixel intensity values.

The principles of geometric blur have already been applied to local descriptor extractors. The "DAISY" descriptor (Tola, Lepetit and Fua, 2010) uses larger Gaussian kernels for positions further away from the centre of the descriptor, based on geometric blur. I propose a different use of geometric blur, which in the case of SIFT spans four dimensions (2D location, scale and orientation), which I outline next.

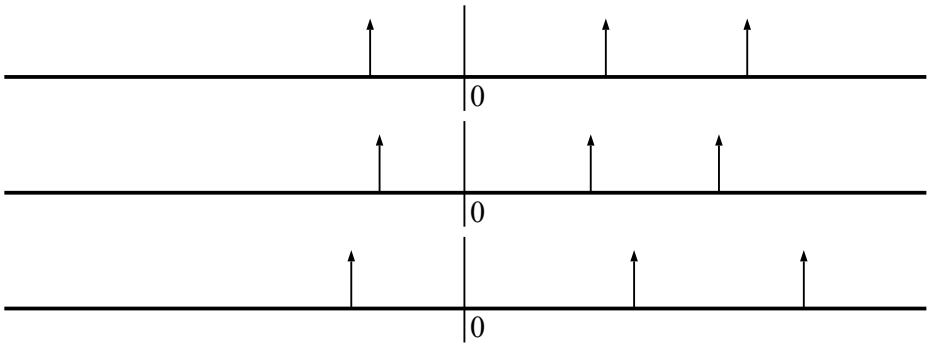

Figure 3.7: A sparse 1D signal (top) and two x-axis scaled versions of the same signal

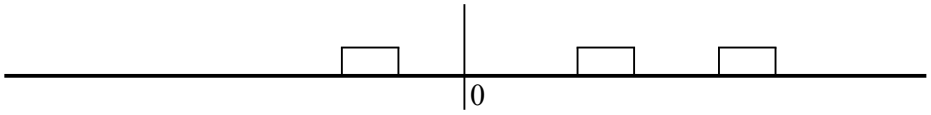

Figure 3.8: A template for matching the sparse signal using a uniform blur

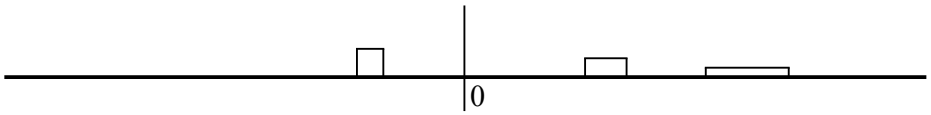

Figure 3.9: A template for matching the sparse signal using a geometric blur

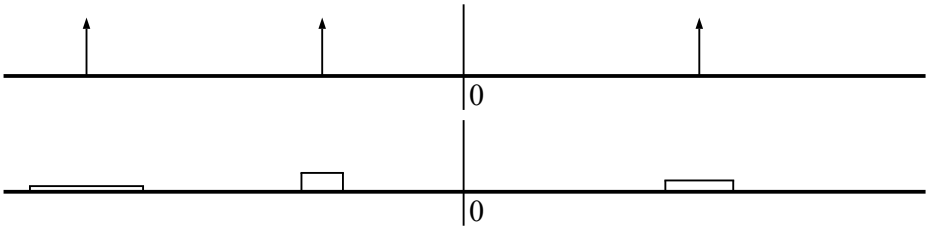

Figure 3.10: Top: a sparse 1D signal; bottom: a template formed by a geometric blur of the signal

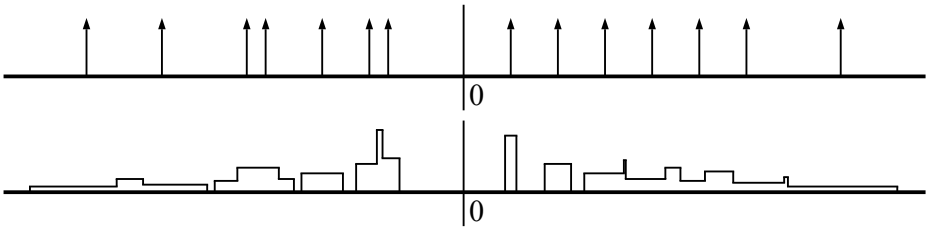

Figure 3.11: Top: a dense 1D signal; bottom: a template formed by a geometric blur of the signal

3.0.3 An Interest-Point Specific Geometric Blur

Berg and Malik (2001) propose the use of geometric blur to provide robustness to geometric distortion when matching templates across an entire image. In their approach, there is no locational uncertainty at the centre of the template, since the template is matched densely across the image. The greatest locational uncertainty is at points furthest away from the centre of template.

However, in an interest-point based framework, matching is performed between local descriptors calculated only at interest-points and from a region of neighbouring pixels (the support region). As shown earlier, interest-point parameters (e.g. location, scale and orientation) corresponding to the same feature in 3D space are known to vary between multiple images relative to the 3D feature. Therefore, the locational uncertainty of a pixel lying at the centre of the support region, is equal to the uncertainty over the interest-point parameters. This differs from in dense template matching, where there was no locational uncertainty at the centre. Further away from the centre, the locational uncertainty of a pixel increases due to the effects of uncertainty in the scale and orientation measurement of the interest-point

extraction.

I, therefore, use a modified version of geometric blur to provide spatial pooling which reflects this, with the goal of achieving more robust descriptors. In contract to Berg and Malik (2001), I apply the geometric blur over all interest-point parameters, including scale and orientation where appropriate, resulting in a 4D rather than a 2D geometric blur.

3.0.4 Comparison to Other Work

Other work has been undertaken to characterise the locational uncertainty of corners or interest-points. J. Shi and Tomasi (1994) proposed a feature selection criteria which selects corners or interest-points that can be precisely positioned within an image, based on their local gradient structure. The purpose is to ensure that only reliable corners or interest-points are used for tracking. Triggs (2004) generalised the corner detection approaches by Förstner (1994), Förstner and Gülch (1987) and Harris and Stephens (1988), used to select corners which offer stability of position, to select those which additionally are stable in scale and orientation. These approaches seek to exclude corners with a large locational uncertainty, rather than characterise and use the locational uncertainty in the matching process.

Other approaches seek obtain a locational uncertainty distribution of a given corner or interest-point. Orguner and Gustafsson (2007) demonstrated a method for calculating the bias and covariance of the location of corners detected with the Harris & Stephens corner detector. The method involves calculating, with the assumption that the image is formed of a "true image" plus independent and identically distributed additive Gaussian noise, the probability that the corner in the true image lies in any of the pixels close to the detected location. Work has also been undertaken to characterise the locational uncertainty distribution of scale-invariant interest-points: Zeisl et al. (2009) demonstrated the use of the Hessian matrix generated from the Taylor series expansion of the *interest-point detector operator* for determining the locational uncertainty of a SIFT of SURF interest-point. They demonstrated that the use of the distributions can improve the performance of 3D reconstruction. In contrast, my work in this chapter, seeks to characterise the local uncertainty of interest-point in order to generate more robust local descriptors.

In addition, there is other work (see Section 2.1.8) related to using

learning per interest-point e.g. Balntas, Tang and Mikolajczyk (2015), Gupta and Mittal (2008), Lepetit and Fua (2006) and Ozuysal et al. (2010). Gupta and Mittal (2008), Lepetit and Fua (2006) and Ozuysal et al. (2010) require training a classifier rather than simply comparing local descriptors e.g. by using Euclidean distance. Balntas, Tang and Mikolajczyk (2015) requires learning a binary mask for each interest-point which is then used in a modified Hamming distance calculation. In contrast, the work in this chapter calculates local descriptors which can be compared using conventional approaches, e.g. Euclidean distance and fast methods which approximate Euclidean distance.

3.1 Methodology

My framework is not dependant on a particular interest-point extractor nor a set of low level image features used to produce the descriptor. I demonstrate it by using the interest-point extractor of SIFT (Lowe, 2004) because of its popularity. The SIFT interest-point extractor uses maxima and minima of the difference of two Gaussian functions (an approximation of the Laplacian of Gaussian function), applied to the image, across many scales, to extract interest-points (see Section 2.1.3). To demonstrate that the approach can be used with any low level features, I show the use of both Histograms of Oriented Gradients (HOG) features, as used in SIFT, and pixel intensity values.

I calculate local descriptors for each interest-point, $\boldsymbol{\theta}_i$, in an image as follows:

1. I model the *locational uncertainty distribution* over interest-point parameters, $p(\boldsymbol{\theta}'_i)$, for each interest-point, $\boldsymbol{\theta}_i$, using a multivariate Gaussian distribution, centred on the interest-point extraction, $\boldsymbol{\theta}_i$,

$$p(\boldsymbol{\theta}'_i) \sim \mathcal{N}(\boldsymbol{\theta}_\mathbf{i}, \boldsymbol{\Sigma}_i) \tag{3.1}$$

   (e.g. for SIFT interest-points, $\boldsymbol{\theta}_i = \left\{t_{xi}, t_{y_i}, \log(\sigma_i), \phi_i\right\}$). I estimate this distribution by using warped versions of the image and redetecting the interest-points in the warped images.

2. I use a Gaussian distribution,

$$q(\boldsymbol{\theta}'_i) \sim \mathcal{N}(\boldsymbol{\theta}_i, s\boldsymbol{\Sigma}_i) \tag{3.2}$$

for the geometric blur. This results in a covariance-based parameterisation for the geometric blur. There are two parameters:

$\boldsymbol{\Sigma}_i$, the four by four covariance matrix which is the estimated covariance of the interest-point's locational uncertainty distribution, and,

s, the *blur moderation parameter*, a scalar, described in Section 3.1.2.

3. I use the Gaussian kernel, $q(\boldsymbol{\theta}'_i)$, to calculate the geometrically blurred descriptor,

$$\boldsymbol{\phi}_{\boldsymbol{i}}^{(gb)} = \int_{\boldsymbol{\theta}'_i} \mathbf{F}(\boldsymbol{\theta}'_i) q(\boldsymbol{\theta}'_i) \, \mathrm{d}\boldsymbol{\theta}'_i, \tag{3.3}$$

where $\mathbf{F}(\boldsymbol{\theta}_i)$ is a set of low level features (see Section 1.6.1) lifted from the support region defined by the interest-point parameters, $\boldsymbol{\theta}_i$. This geometric blur is specific to each interest-point and replaces the spatial pooling used in other descriptors such as SIFT which is uniform across all interest-points.

4. Finally, I normalise the descriptor using the L^2 norm to provide robustness to changes to lighting conditions.

Figure 3.12 shows an overview of my approach.

Figure 3.14 shows a geometrically blurred image. This is only one possible geometric blur: each interest-point will have its own individual blur which would correspond to different patterns of dots (see Figure 3.17). Note that the dots further from the centre show a greater blur than those in the centre, which is the result of blurring over scale and orientation of an interest-point.

Figure 3.16 shows a geometrically blurred HOG descriptor (see Section 2.1.3 for explanation). Each square (pixel) contains contributions from many adjacent squares. As with the synthetic dots pattern, squares further out from the centre show a greater blur. However, there is still some amount of blur in the centre and no square has fewer than three orientation bins (directions) occupied, compared to before the blur (Figure 3.15) in which at most two orientation bins are occupied.

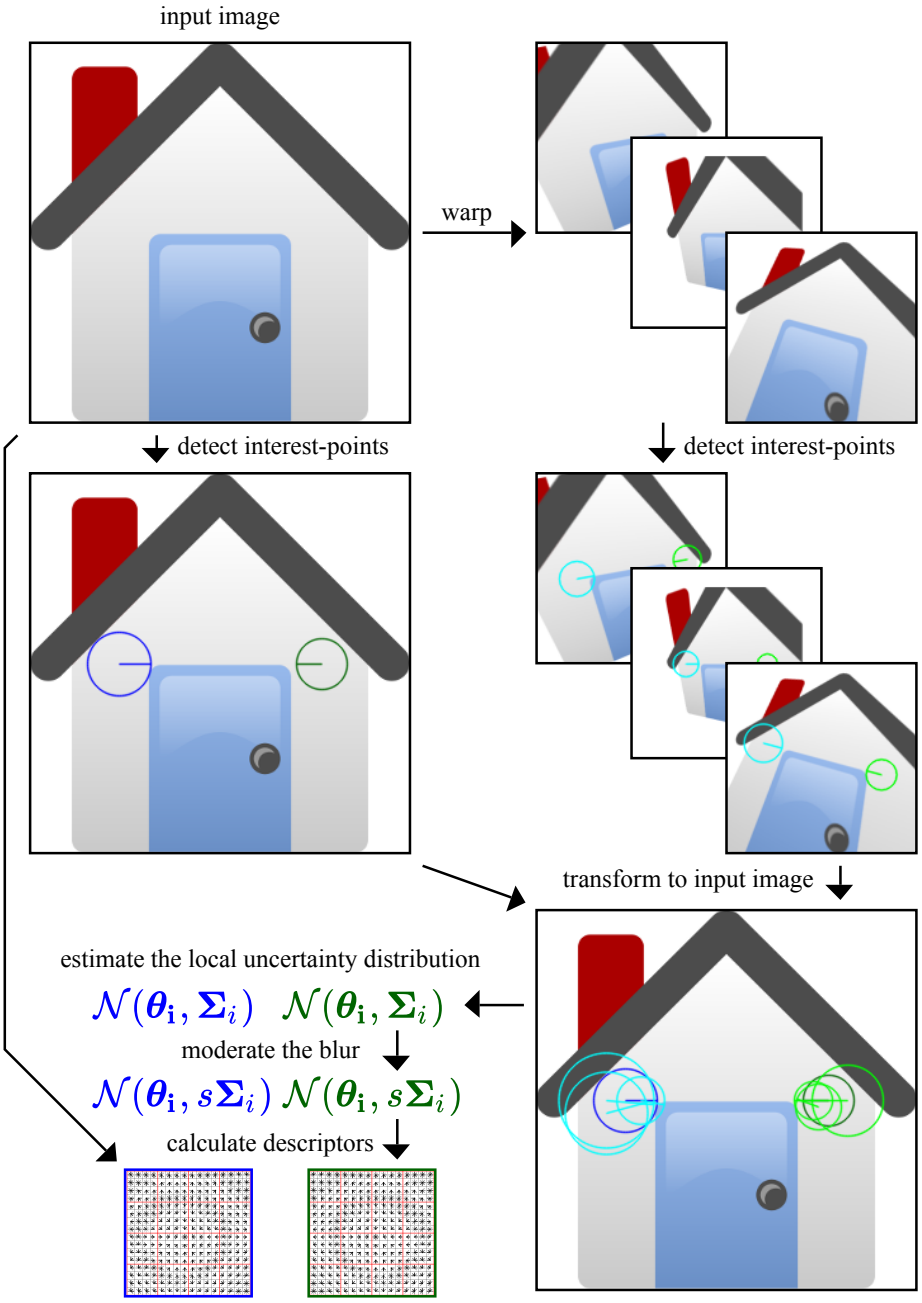

Figure 3.12: An overall diagram of my approach for calculating local descriptors for each interest-point in an image

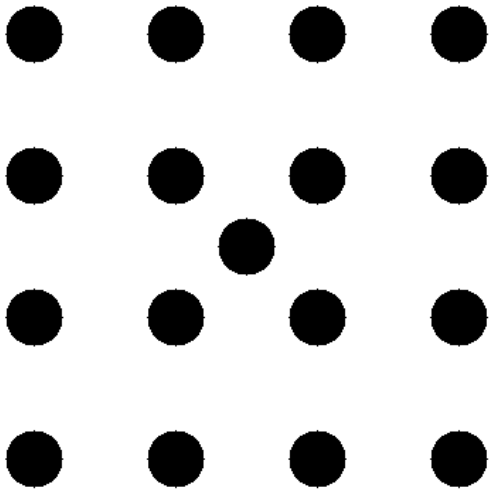

Figure 3.13: A patch containing dots before applying a geometric blur

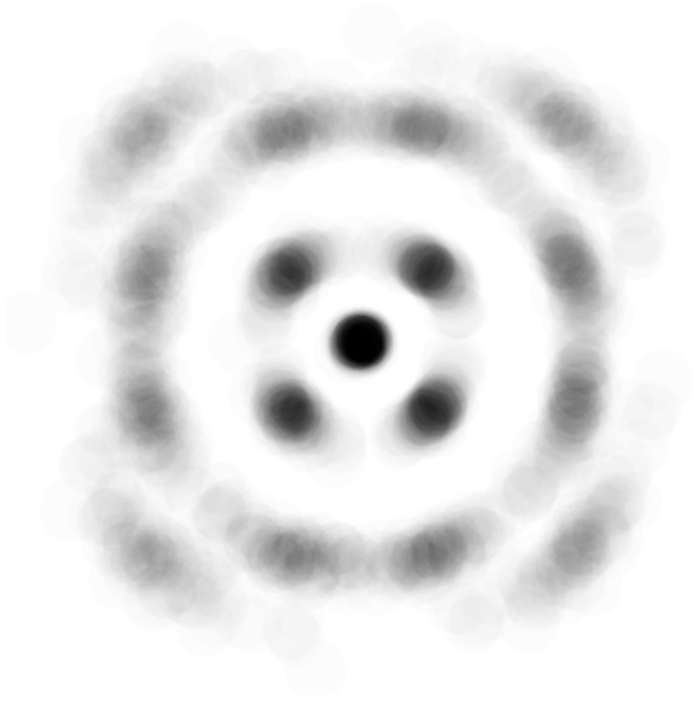

Figure 3.14: A patch containing dots after applying a geometric blur

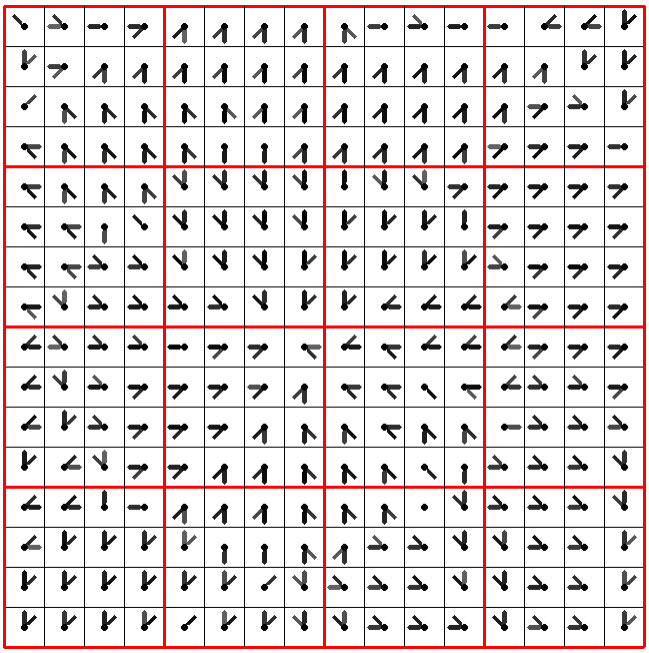

Figure 3.15: HOG features before applying a geometric blur

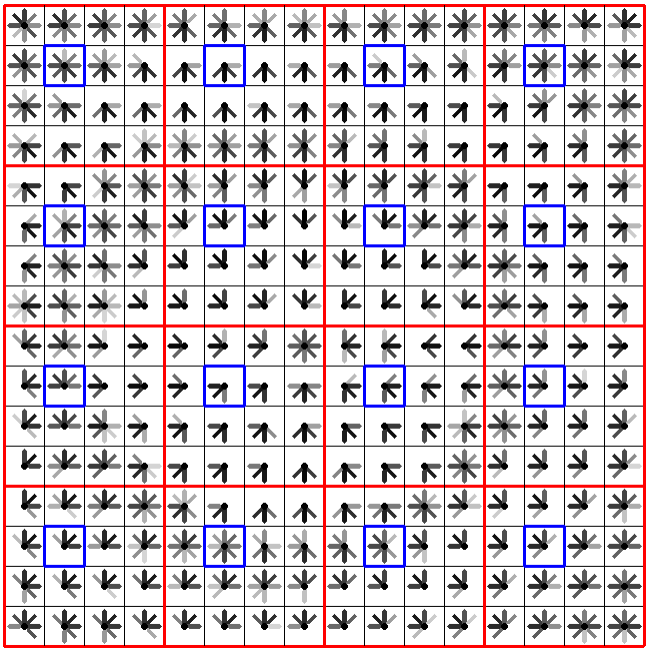

Figure 3.16: HOG features after applying a geometric blur

3.1.1 Criteria for Corresponding Interest-Points

The aim of my approach is to improve local descriptors used to match interest-points. However, the approach I use requires a means for matching interest-point between and image and a warped version of the image, before the local descriptor has been calculated and in order to assist its calculation.

There is no universal definition of *corresponding* when referring to two interest-points between two images: it is application-specific and depends on other factors such desired tolerances and knowledge of the scene geometry. My approach, however, requires a concrete criteria for the following two reasons:

1. Enabling the calculation of the locational uncertainty distribution, $p(\boldsymbol{\theta}'_i)$ for each interest-point through the method described in the next section

2. Providing a ground truth for optimisation and evaluation

My approach for determining whether two interest-points in two different images correspond has two steps:

1. I transform the interest-point in the source image into the destination image.

2. I compare the location, orientation and scale of this transformed interest-point with the interest-point extracted from the destination image to determine whether the two interest-points correspond.

I assume that the transformation of a single pixel location, $\mathbf{x} = \begin{bmatrix} x & y \end{bmatrix}^\top$, between a pair of images can be expressed as a homography, $\mathbf{H}$, such that the point in the destination image,

$$\mathbf{x}' = \mathcal{H}^{-1}(\mathbf{H}\mathcal{H}(\mathbf{x})) \tag{3.4}$$

where $\mathcal{H}$ is a transformation between Cartesian and homogeneous image coordinates. Note that $\mathcal{H}$, if unconstrained, is non-surjective, as a Cartesian co-ordinate, $\begin{bmatrix} u & v \end{bmatrix}^\top$, corresponds to a range of homogeneous image coordinates,

$$\begin{bmatrix} wu \\ wv \\ w \end{bmatrix} \forall w \neq 0. \tag{3.5}$$

Therefore, $\mathcal{H}$ is defined by constraining w to unity. $\mathcal{H}^{-1}$ is a left inverse as $\mathcal{H}^{-1}(\mathcal{H}(\mathbf{x})) = \mathbf{x}$.

The homography defines a transformation for individual positions within an image but does not define a transformation for the location, scale and orientation of an interest-point. Therefore, I use the following method to transform an interest-point between images:

1. I sample uniformly a number of locations, $\mathbf{x}_i$, in the support region defined by the interest-point's parameters and transform these according to the homography, producing an equal number of locations,
$$\mathbf{x'}_i = \mathcal{H}^{-1}(\mathbf{H}\mathcal{H}(\mathbf{x_i})). \tag{3.6}$$
For the SIFT interest-point extractor, I use the sixteen points located at the centres of each of the pooling regions usually used for calculating the SIFT descriptor.

2. I then find the translation, scale and orientation change between these point pairs, $\{\mathbf{x}_i, \mathbf{x'}_i\}$. which solves the least-squares problem:
$$\arg\min_{\mathbf{S}} \sum_i \left|\mathcal{H}^{-1}\left(\mathbf{S}\mathcal{H}(\mathbf{x}_i)\right) - \mathcal{H}^{-1}\left(\mathbf{H}\mathcal{H}(\mathbf{x}_i)\right)\right|^2, \tag{3.7}$$
where
$$\mathbf{S} = \begin{bmatrix} a\begin{bmatrix} \cos(\alpha) & \sin(\alpha) \\ -\sin(\alpha) & \cos(\alpha) \end{bmatrix} & \begin{matrix} t_1 \\ t_2 \end{matrix} \\ \begin{matrix} \mathbf{0}^\top \end{matrix} & 1 \end{bmatrix}. \tag{3.8}$$
This amounts to a local approximation of the homography, $\mathbf{H}$, with a similarity transform, $\mathbf{S}$, in the range of the descriptor footprint. This similarity transform has four parameters corresponding to the SIFT interest-point and therefore allows a SIFT interest-point to be transformed.

Transforming an interest-point from the source image to the destination image, using the similarity transform, $\mathbf{S}$, allows it to be matched to interest-points in the destination image. The transformed interest-point matches an interest-point in the destination image if the following criteria apply:

1. The transformed interest-point has the following, with respect to the interest-point in the second image (the same criteria as used in Winder, Hua and Brown (2009)):

a) a 2D location within a five pixels radius relative to its scale,

b) an orientation within $\frac{\pi}{8}$ radians, and

c) a $\log_2$ scale within 0.25 of the $\log_2$ scale

2. The interest-point in the destination image is closer to the transformed interest-point than any other interest-point in the destination image.

3.1.2 Calculating the Geometric Blur Parameters

Following the logic of geometric blur, I believe the optimal amount of blur for a given interest-point relates to two factors:

1. The locational uncertainty distribution of the interest-point

2. The sparsity of the low level features from which the descriptor is calculated

Consider the case of comparing the descriptors calculated from two matching interest-points between two images. If both interest-points are transformed to the same image, there will (hopefully) be an extent to which corresponding pooling regions overlap. This will be related to the locational uncertainty of each interest-point. A large overlap means that the descriptors are more likely to be identical despite the misalignment between the interest-points. As the proportion of overlap increases, one would expect the *true positive rate* to increase for a given threshold.

Although an increase in blur would be expected to result in an increase in the *true positive rate* as it increasing the overlap of the pooling regions, it will also increase the *false positive rate*. This is because an increase in blur results in larger pooling regions and the descriptor begins to describe more of the whole image and less of the locality. The extreme case, an infinite blur, would result in every descriptor containing the average value of the low level features across the whole image. The extent to which the blur can be increased without diminishing returns in terms of too large a *false positive rate* would be related to the sparsity of the signal.

I use a geometric blur to account for these two factors. The geometric blur is always a multivariate Gaussian distribution,

$$q(\boldsymbol{\theta}'_i) \sim \mathcal{N}(\boldsymbol{\theta}_i, s\boldsymbol{\Sigma}_i), \tag{3.9}$$

centred on the original interest-point detection, $\boldsymbol{\theta}_i$, where $\boldsymbol{\Sigma}_i$ is an estimate of the covariance of the interest-point parameters, calculated using the methodology in the next section. The *blur moderation parameter*, s, is a scalar which enables the amount of blur to be moderated relative to the estimated covariance. Note that for $s = 1$, the descriptor becomes the expected low level features over the locational uncertainty distribution, i.e. $\mathbf{D}^{(\mathbf{gb})}{}_i = \mathrm{E}[\mathbf{F}(\boldsymbol{\theta}'_i)]$, so the technique can be viewed as a *probabilistic* version of Berg and Malik (2001)'s geometric blur.

3.1.2.1 Estimating the locational uncertainty distribution

I observe that the locational uncertainty of an interest-point is dependent upon multiple factors, including the following:

1. The different locations from which the 3D feature is observed, which results in *geometric distortion*
2. Differences in lighting conditions, specifically those which result in a non-uniform changes to pixel intensity values, such as shadows
3. Differences in the camera and configuration used to attain the image
4. Differences in any post-processing of the image
5. Occlusion of the 3D feature by any other object(s) or phenomena
6. Other sources of noise, such as electrical noise within the camera

Based on my observation, I assume that the dominant factor is the different viewpoints from which a 3D feature which results in an interest-point detection is observed.

For every image, it would be ideal to model the 3D scene which resulted in the image and to generate samples from the locational uncertainty by projecting this model to 2D from different viewpoints. However, in Computer Vision problems for which interest-point detection and matching is used, the 3D scene is usually not known *a priori*—indeed interest-point matching is often used to generate matches between images for the purpose of 3D reconstruction. Therefore, I use a method which relies upon only a single image (without depth or 3D

information) to obtain the locational uncertainty of each interest-point within that image.

Let $p(\mathbf{H})$ be a prior on homographies. My method operates as follows:

1. I generate N multiple warped versions,

$$\mathbf{I}\big(\mathcal{H}^{-1}(\mathbf{H_n}\mathcal{H}(\mathbf{x}))\big), n = 1, 2, \ldots, N, \tag{3.10}$$

   of the original image, $\mathbf{I}(\mathbf{x})$, using homographies sampled from a prior distribution, $p(\mathbf{H})$. This is intended to approximately model the different viewpoints from which one expects to view the same interest-point.

2. I run the interest-point extractor on the N transformed images, $\mathbf{I}(\mathcal{H}^{-1}(\mathbf{H_n}\mathcal{H}(\mathbf{x})))$, and match all the interest-points against the interest-points in the original image, $\mathbf{I}(\mathbf{x})$, using the method outlined in Section 3.1.1.

3. Finally, I transform each matching interest-point from the transformed images into the original image, $\mathbf{I}(\mathbf{x})$. These interest-points are considered be samples from a distribution approximating the locational uncertainty distribution of the original interest-point,

$$p(\boldsymbol{\theta}_i') \sim \mathcal{N}(\boldsymbol{\theta}_\mathbf{i}, \boldsymbol{\Sigma}_i). \tag{3.11}$$

I use the covariance of these samples as an estimate (the maximum likelihood estimate) for $\boldsymbol{\Sigma}_i$.

I believe that the approximation is correct when the following conditions are met:

1. Only geometric distortion contributes to the locational uncertainty.

2. The interest-point and its support region lie on a plane.

3. The interest-point is being observed from a direction perpendicular to the plane.

4. The prior distribution on homographies correctly models the distribution of viewpoints of the feature, relative to the viewpoint of the source image.

In practice, these conditions, particularly number 2, will not, in general be true. However, since non-affine invariant features extractors rely on conditions 2 and 3 being approximately true (i.e. narrow-baseline matching), this does not overly affect the performance of the method.

3.1.2.2 Prior on homographies

The prior on the homographies used to generate synthetic warps,

$$p(\mathbf{H}) = p(t_x)p(t_y)p(\alpha)p(s_x)p(s_y)p(s_s)p(p_x)p(p_y) \tag{3.12}$$

where

$$\mathbf{H} = \begin{bmatrix} \mathbf{RS} & \mathbf{t} \\ \mathbf{p} & 1 \end{bmatrix}, \tag{3.13}$$

and

$$\mathbf{R} = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix}, \mathbf{S} = \begin{bmatrix} s_x & s_s \\ 0 & s_y \end{bmatrix}, \mathbf{t} = \begin{bmatrix} t_x \\ t_y \end{bmatrix}, \mathbf{p} = \begin{bmatrix} p_x & p_y \end{bmatrix}. \tag{3.14}$$

The factors have the following distributions: $p(t_x) \sim \mathcal{U}(-0.5, 0.5)$, $p(t_y) \sim \mathcal{U}(-0.5, 0.5)$, $p(s_x) \sim \mathcal{U}(0.5, 2)$, $p(s_y) \sim \mathcal{U}(0.5, 2)$, $p(s_s) \sim \mathcal{U}(-0.5, 0.5)$, $p(\alpha) \sim \mathcal{U}\left(\frac{-\pi}{4}, \frac{\pi}{4}\right)$ radians, $p(p_x) \sim \mathcal{U}(-0.0001, 0.0001)$ and $p(p_y) \sim \mathcal{U}(-0.0001, 0.0001)$.

This prior is chosen to reflect possible viewing positions of the same scene, excluding those which would require wide-baseline matching. The distribution over translations, $\mathcal{U}(-0.5, 0.5)$, covers all possible translations since the interest-point extractor is covariant to translation over whole pixels. Likewise, the distribution over scalings, $\mathcal{U}(-0.5, 0.5)$, covers all scales since the interest-point extractor repeatedly downscales the image to half width and half height. Similarly, the distributions over rotations, $\mathcal{U}\left(\frac{-\pi}{4}, \frac{\pi}{4}\right)$, covers all rotations since the interest-point extractor relies on gradient operators which are the transpose of each other, tantamount to a rotation of 90° ($\frac{-\pi}{2}$ radians).

The hyper-parameters, p_x and p_y, are important as they allow for the simulation of perspective transformations (with the planar assumption). If their magnitude be too small, the warping would fail to capture the likely appearances of interest-point following perspective transforms. However, too large a magnitude would mean that the warping would likely captures appearances of interest-point which

would require wide-baseline matching. The choice of 0.0001 for both parameters represents a compromise which was verified through visually examining the range of warped images over different values.

3.1.2.3 Value of the blur moderation parameter

I fix the blur moderation parameter, s, for all images and interest-points. I determined the optimal value experimentally through a brute force search (see experiment in Section 3.2.2) and always use $s = 1.5$.

3.1.3 Applying geometric blur to calculate descriptors

I apply a geometric blur as in Equation (3.3), parameterised by the covariance estimated in the previous section, $\mathbf{\Sigma}_i$, and the blur moderation parameter, s, to the low level features, $\mathbf{F}(\boldsymbol{\theta}_i)$. The result is a geometrically blurred descriptor, $\boldsymbol{\phi_i}$. Equation 3.3 cannot be calculated analytically, so I use a Monte Carlo method. I approximate the blur by taking a finite number of samples, N_j in total, $\boldsymbol{\theta}'_{ij}, j = 1, 2, \ldots, N_j$, from the distribution, $q(\boldsymbol{\theta}'_i) \sim \mathcal{N}(\boldsymbol{\theta}_i, s\mathbf{\Sigma}_i)$ and calculate the blurred descriptor as

$$\boldsymbol{\phi_i} = \int_{\boldsymbol{\theta}'_i} \mathbf{F}(\boldsymbol{\theta}'_i) q(\boldsymbol{\theta}'_i) \, \mathrm{d}\boldsymbol{\theta}'_i \approx \frac{1}{N_j} \sum_{j=1}^{N_j} \mathbf{F}\Big(\boldsymbol{\theta}'_{ij}\Big). \tag{3.15}$$

In all cases, I use $N = 100$.

Figure 3.13 shows an image patch, a synthetic pattern of black dots, prior to a geometric blur. Here the low level features, $\mathbf{F}(\boldsymbol{\theta}_i)$, would simply be the pixel intensity values. Figure 3.14 is the result of applying a geometric blur and reconstructing the pixel intensity values into an image. This is just one possible geometric blur: Figure 3.17 shows six different geometric blurs applied to the synthetic dots image.

3.1.3.1 Geometrically Blurred HOG Descriptors

To apply geometric blur to HOG low level features, as used in SIFT (Lowe, 2004), I represent the low level image feature lifting operation, $\mathbf{F}(\boldsymbol{\theta}_i)$, as a function which returns a either a $16 \times 16 \times 8$ dimensional vector or a $4 \times 4 \times 8$ dimensional vector. Both are calculated on the

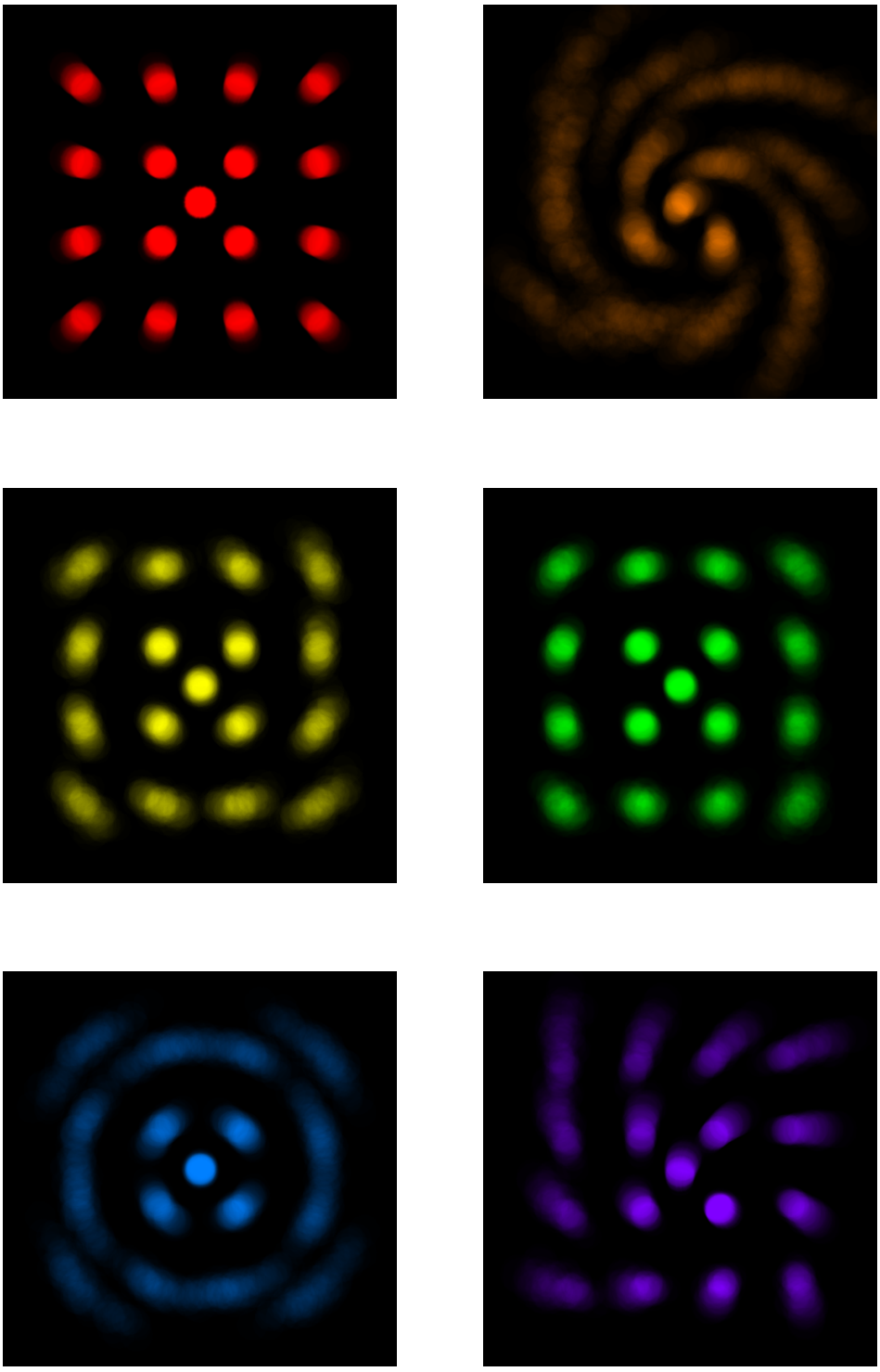

Figure 3.17: Different geometric blurs applied to a pattern of dots (Figure 3.13); the colour corresponds to the interest-point which yielded that blur in Figure 3.3.

same 16×16 pixel support region used by the SIFT interest-point (see Figure 3.15).

The $16 \times 16 \times 8$ version produces an 8D gradient orientation histogram for every pixel whereas the $4 \times 4 \times 8$ version returns an 8D orientation histogram for one in sixteen pixels: it returns the histogram for the top-left of the four centre pixels of each 4×4 pixel sub-cell (marked blue in Figure 3.16). The SIFT descriptor bins orientated gradients into a histogram for every such 4×4 cell, i.e. the regions bounded by the red lines of Figure 3.16. Therefore, carrying out spatially pooling during the binning operation. However, no pooling occurs as part of the low-level image feature lifting operation under my framework because the next step, applying a geometric blur, results in spatial pooling.

The spatial pooling of the SIFT descriptor can be expressed in my geometric blur framework, by casting the pooling to blurring using a local uncertainty distribution which has equal values for each of the locations,

$$\begin{aligned}
&(-1.5, -1.5), (-1.5, -0.5), (-1.5, 0.5), (-1.5, 1.5), \\
&(-0.5, -1.5), (-0.5, -0.5), (-0.5, 0.5), (-0.5, 1.5), \\
&(0.5, -1.5), (0.5, -0.5), (0.5, 0.5), (0.5, 1.5), \\
&(1.5, -1.5), (1.5, -0.5), (1.5, 0.5) \text{ and } (1.5, 1.5),
\end{aligned}$$

and zero else-where. With δ being the Dirac delta function, this is equal to a geometric blur with the following locational uncertainty distribution:

$$\begin{aligned}
&p\left(\{t_x\, t_y,\, \log(\sigma),\, \phi\}\right) = \\
&\frac{1}{16}\left(\delta(t_x - 1.5) + \delta(t_x - 0.5) + \delta(t_x + 0.5) + \delta(t_x + 1.5)\right)\left(\delta(t_y - 1.5) + \right. \\
&\quad \left. \delta(t_y - 0.5) + \delta(t_y + 0.5) + \delta(t_y + 1.5)\right)\delta(\sigma - \sigma_i)\delta(\phi - \phi_i), \quad (3.16)
\end{aligned}$$

and with the *blur moderation parameter*, s, set to unity.

3.2 Experiments

I test the performance of different configurations of my local descriptor (calculated using the methodology in Section 3.1) against the local descriptor used in SIFT (see Section 2.1.3.2). I test each descriptor by

matching between SIFT interest-points found in the image sequences of Mikolajczyk's dataset (Mikolajczyk and Schmid, 2005). Table 3.1 indicates the transformations or distortions applied (e.g. by movement of the camera or changing the camera focus) between images in the sequence.

I test the ability of each local descriptor to match interest-points between two images in the dataset through calculating the Euclidean distance between all positive interest-point matching pairs and an equal number of randomly sampled negative matches across each image pair. For the purposes of generating a ground truth, I use the same criteria as used in 3.1.1 to determine a positive match:

1. The transformed interest-point has the following, with respect to the interest-point in the second image (the same criteria as used in Winder, Hua and Brown (2009)):
   a) a 2D location within a five pixels radius relative to its scale,
   b) an orientation within $\frac{\pi}{8}$ radians, and
   c) a $\log_2$ scale within 0.25 of the $\log_2$ scale
2. The interest-point in the destination image is closer to the transformed interest-point than any other interest-point in the destination image.

A negative match requires the that interest-point, transformed from the first image to the second image, lies outside an exclusion zone formed by doubling the value of each parameter in the criteria. Matches lying within the exclusion zone are considered ambiguous and discarded.

For the purposes of testing the descriptor, I consider a match to be positive if the Euclidean distance between the two matching points is less than a threshold. However, rather than specifying a fixed threshold, I allow the threshold to vary and generate receiver operating characteristic (ROC) curves based on varying the threshold. I report the area under an ROC curve (ROC-area) as a performance metric (higher is better).

3.2.1 Performance

Figure 3.18 and Table 3.2 demonstrate the performance of my approach against SIFT. In all experiments, the *blur moderation parameter* (see Section 3.1.2), s, was set to 1.5 (other than the experiments

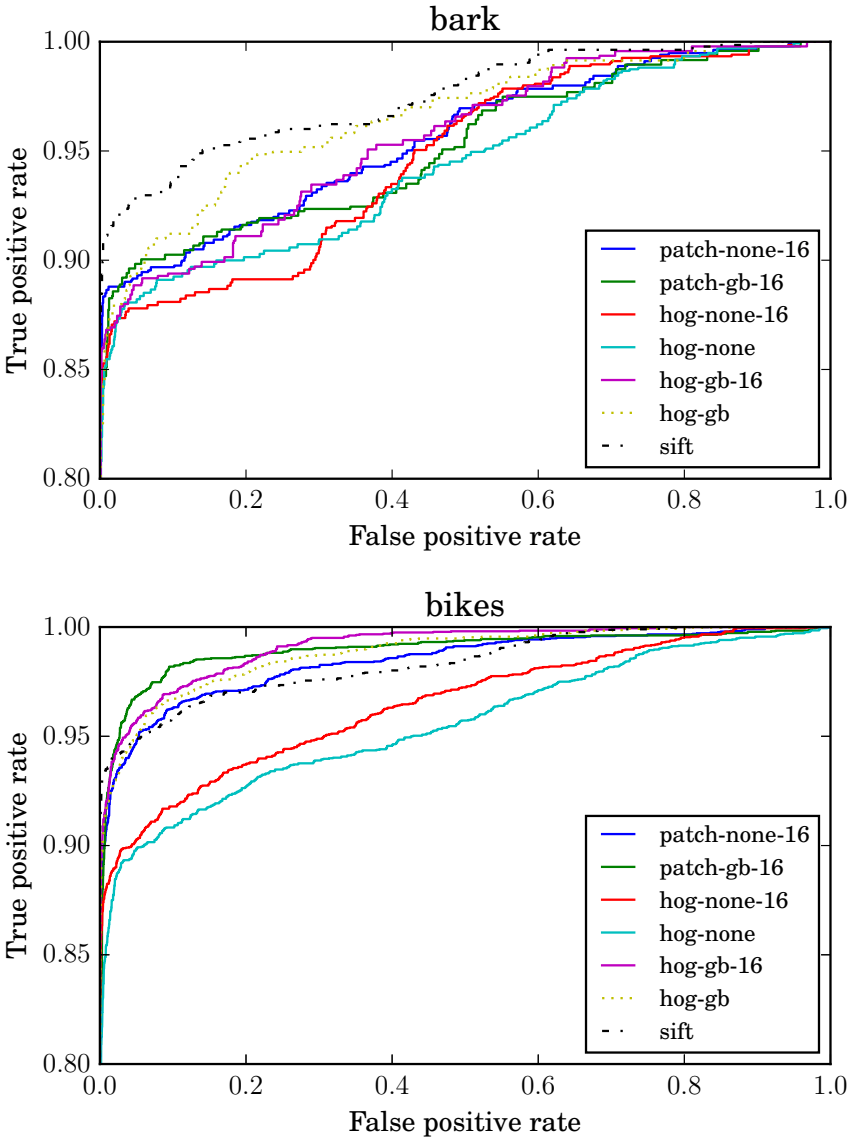

Figure 3.18: ROC curves for matching with different descriptors

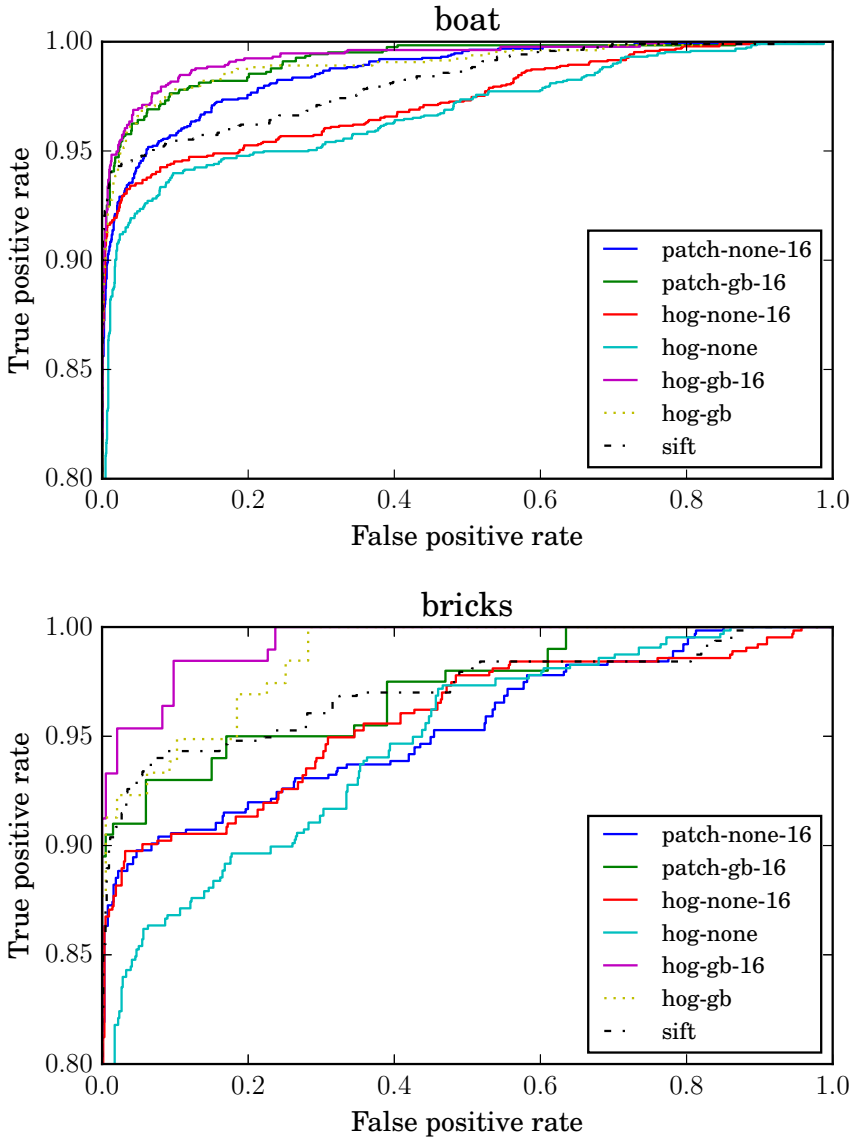

Figure 3.18: ROC curves for matching with different descriptors

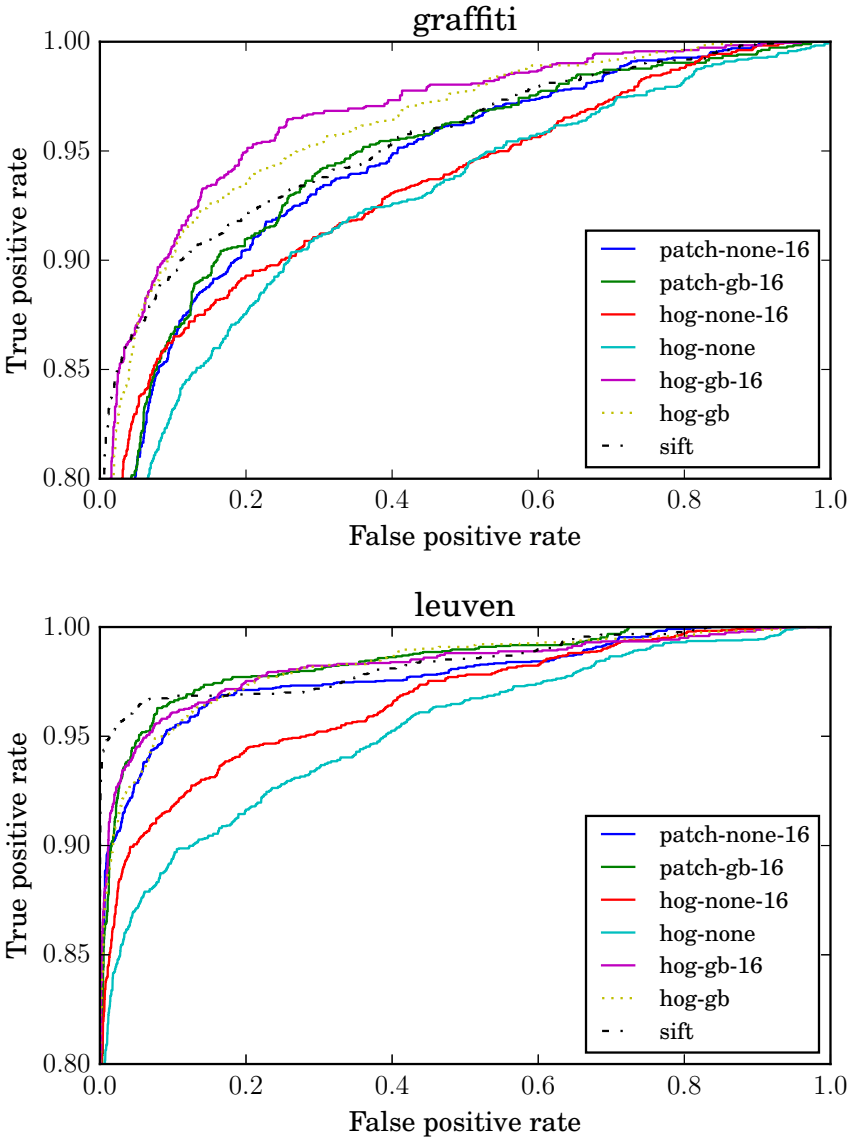

Figure 3.18: ROC curves for matching with different descriptors

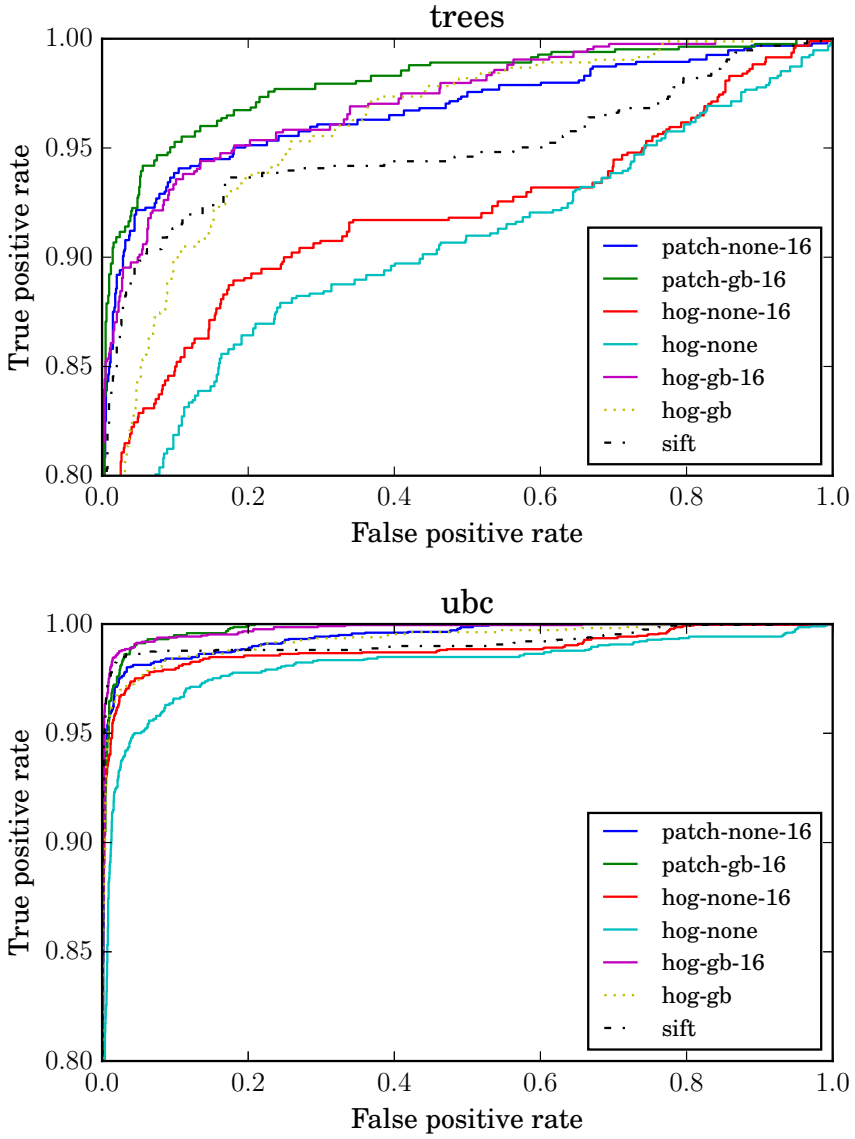

Figure 3.18: ROC curves for matching with different descriptors

Image sequence	Distortion or Transformation
bikes	blur
trees	blur
graffiti	viewpoint
bricks	viewpoint
bark	zoom and rotation
boat	zoom and rotation
leuven	light
ubc	JPEG compression

Table 3.1: The distortions or transformations applied to images in Mikolajczyk's dataset

	Image Sequence							
Descriptor	bark	bikes	boat	bricks	graffiti	leuven	trees	ubc
patch-none-16	0.955	0.984	0.986	0.953	0.943	0.978	0.967	0.994
patch-gb-16	0.951	0.989	0.991	0.969	0.944	**0.984**	**0.979**	**0.997**
hog-none-16	0.947	0.964	0.973	0.957	0.933	0.966	0.918	0.988
hog-none	0.944	0.955	0.967	0.944	0.924	0.952	0.900	0.981
hog-gb-16	0.955	**0.991**	**0.993**	**0.990**	**0.966**	0.982	0.971	0.997
hog-gb	0.965	0.988	0.989	0.983	0.961	0.981	0.958	0.993
SIFT	**0.974**	0.983	0.981	0.970	0.954	0.983	0.949	0.992

Table 3.2: The area under the ROC curve produced using each descriptor (Figure 3.18); the best result is in bold

to determine the value in Section 3.2.2), thirty transformed versions of each image were generated to estimate the covariance matrix, $\boldsymbol{\Sigma_i}$, for each interest-point (see Section 3.2.3) and one hundred samples were generated from the multivariate Gaussian distribution for each interest-point, to perform the blur using the approximation in (3.15).

Each descriptor (other than SIFT) is described by the abbreviated format, *(lifting)-(pooling)[-(size)]*, with the following options:

(lifting) Either **patch**, the pixel intensity values of the 16×16 support region covariant with the SIFT interest-point, or **hog**, for HOG low level features (see Section 3.1.3.1) obtained from this support region

(pooling) The operation used to pool the descriptor: either **none** (i.e. $\boldsymbol{\phi_i} = \mathbf{F}(\boldsymbol{\theta}_i)$) or **gb** for my approach, an interest-point specific geometric blur (3.3).

(size) A **-16** suffix, for the **hog** lifting operation, indicates that the low-level features, $\mathbf{F}(\boldsymbol{\theta}_i)$, returns a $16 \times 16 \times 8$ dimensional vector, instead of the $4 \times 4 \times 8$ dimensional vector of the other version. The **patch** lifting operation is always formed from $16 \times 16 \times 1$ pixel intensity values and hence is always suffixed.

In six of the image sequences, "bikes", "boat", "bricks", "graffiti", "trees" and "ubc", the geometric blur based pooling (hog-gb) outperformed SIFT. It is not surprising that the performance is particularly better on the image sequences involving viewpoint changes ("bricks" and "graffiti") given the synthetic warps aim to mimic a viewpoint change but more surprising is the improvement in performance on those not involving a viewpoint change. Perhaps, in the cases of "bikes", "boat", "trees" and "ubc", the rectangular pooling of SIFT results in the pooling regions being larger than optimal, whereas the geometric blur results in smaller pooling regions more appropriate for the transformations forming these image sequences.

The geometric blur based pooling (hog-gb) performs particularly well compared to SIFT for mid-range values of *false positive rate* (between 0.2 and 0.7). I hypothesise that this is because SIFT has in insufficient level of blur for interest-points with a high locational uncertainty.

3.2.2 Optimal Level of blur

Figure 3.19 shows the effect of changing the extent of the geometric blur, through different values of the *blur moderation parameter*, s (see Section 3.1.2). The optimal level of blur varies depending on the image sequence, but the best overall performance is achieved around $s = 1.5$. This is to some extent what one would expect for the following reasons, assuming the predicted locational uncertainty distribution is approximately correct:

1. $s < 1.0$ would result in the descriptor not being blurred over the whole locational uncertainty distribution.

2. $s > 1.0$ would provide increased robustness by increasing the intersection between pooling regions when there is misalignment.

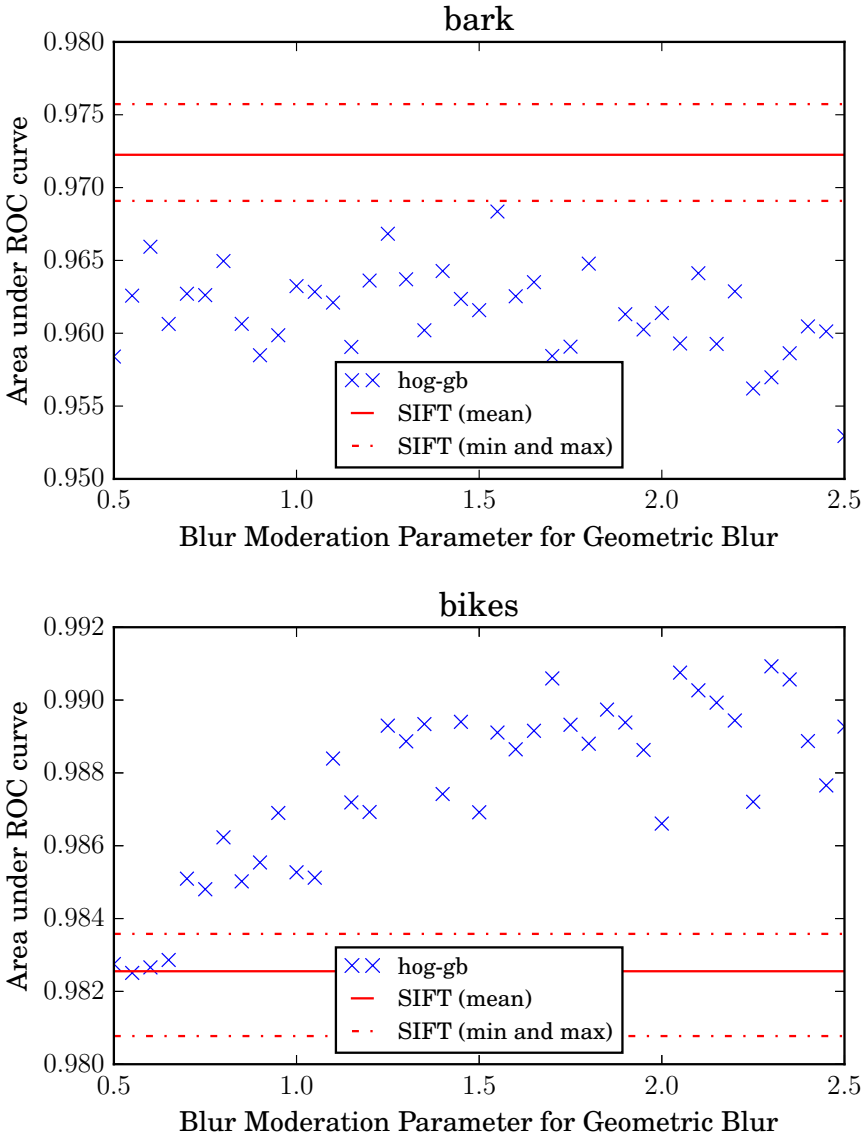

Figure 3.19: The effect of changing the blur moderation parameter, s, on the ROC-area for "hog-gb"; the results for the SIFT descriptor are shown too: the solid line indicates the mean and the dashed lines the maximum and minimum

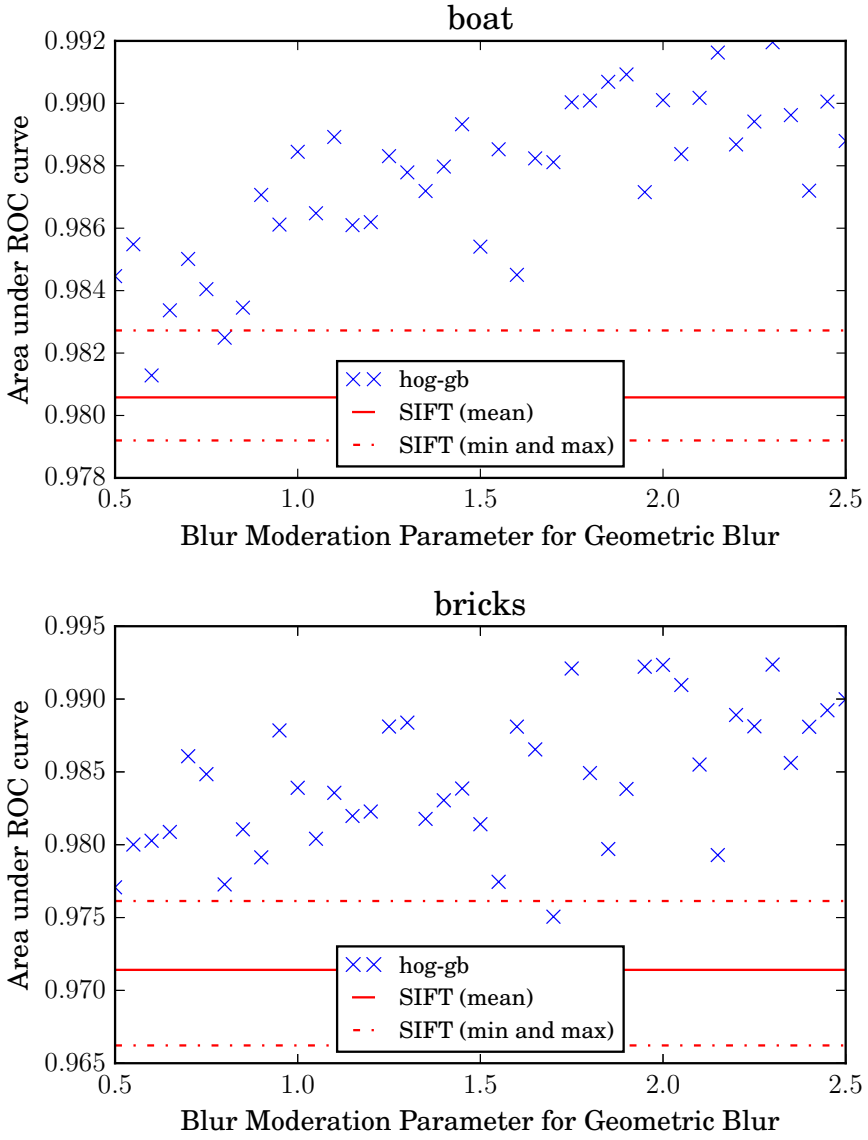

Figure 3.19: The effect of changing the blur moderation parameter, s, on the ROC-area for "hog-gb"; the results for the SIFT descriptor are shown too: the solid line indicates the mean and the dashed lines the maximum and minimum

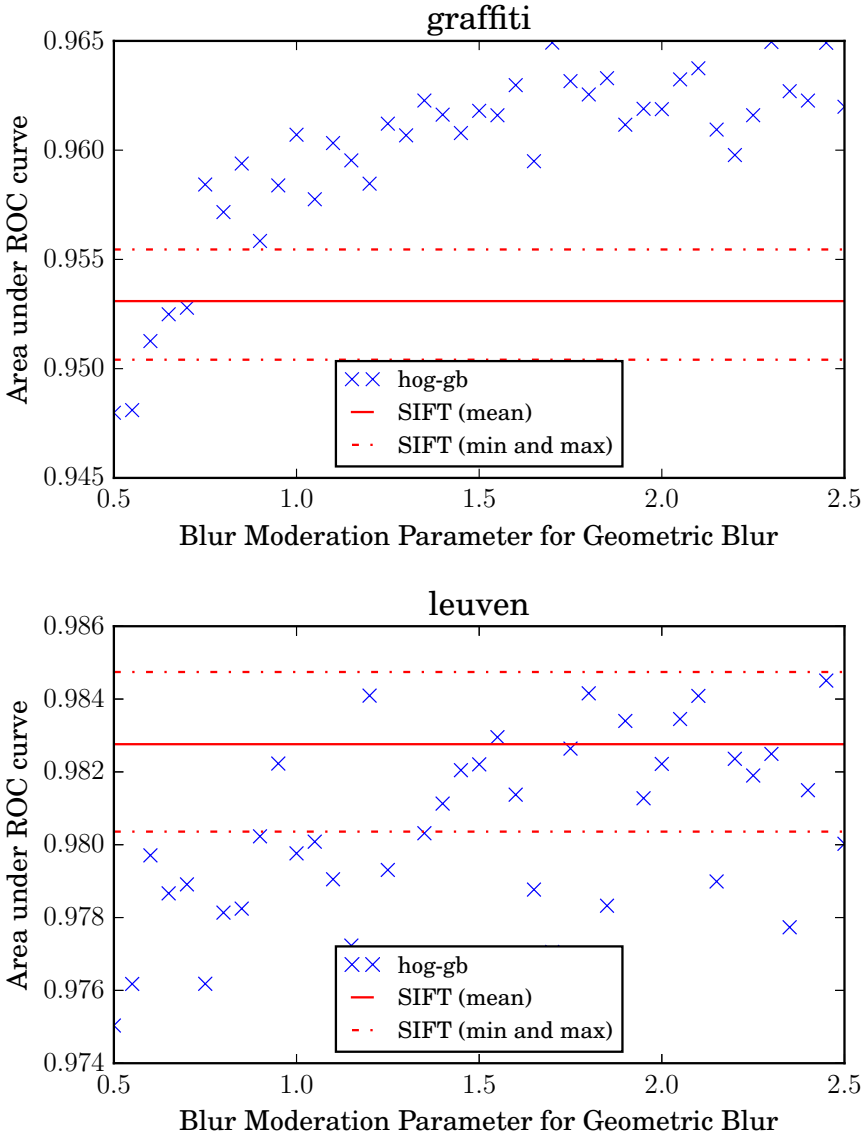

Figure 3.19: The effect of changing the blur moderation parameter, s, on the ROC-area for "hog-gb"; the results for the SIFT descriptor are shown too: the solid line indicates the mean and the dashed lines the maximum and minimum

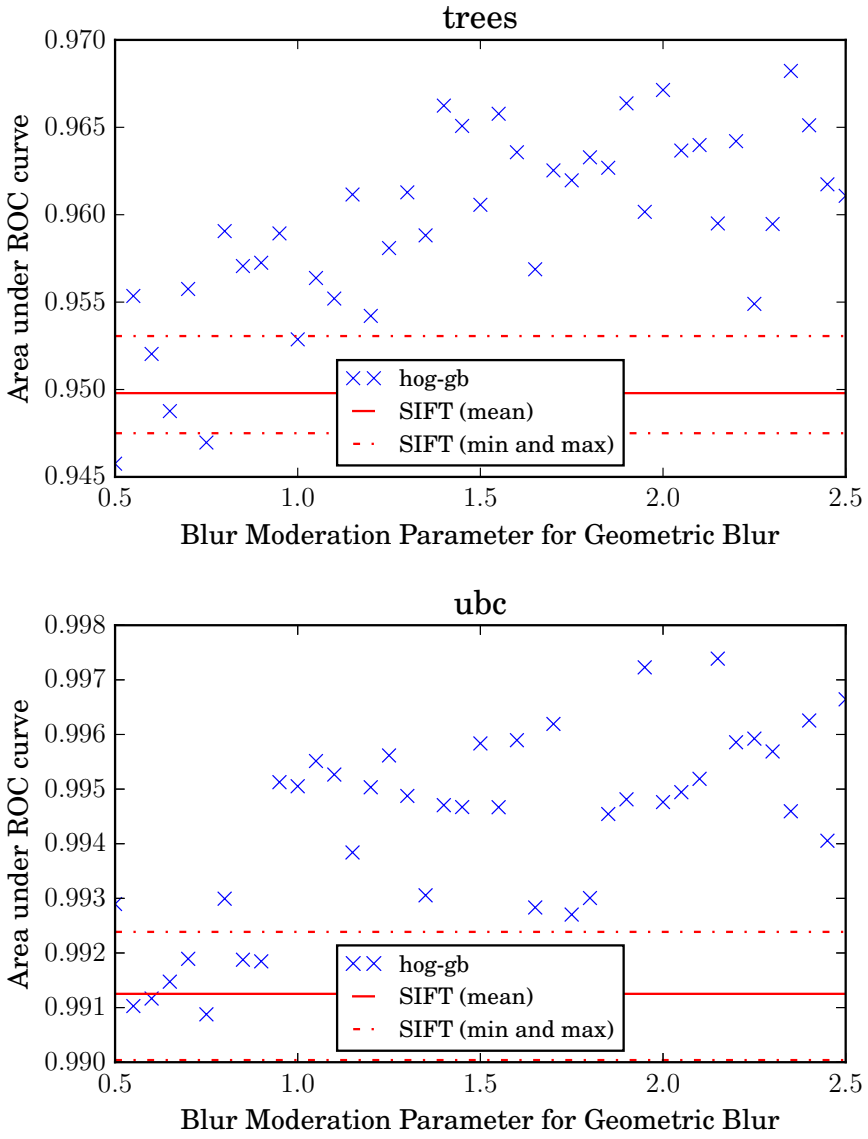

Figure 3.19: The effect of changing the blur moderation parameter, s, on the ROC-area for "hog-gb"; the results for the SIFT descriptor are shown too: the solid line indicates the mean and the dashed lines the maximum and minimum

3. $s > 2.0$ would result in a blur more than twice as much as the predicted locational uncertainty and it is reasonable to expect diminishing returns as the spatial pooling regions become too large.

However, in six out of eight image sequences, "bikes", "boat", "bricks", "graffiti", "trees" and "ubc", the performance remains constant or actually increases for higher values of s, suggesting that sparsity of the oriented gradient features in the vicinity rather than locational uncertainty of the interest-point may play a more important factor in optimising the descriptor.

3.2.3 Number of transformed images used to predict the locational uncertainty distribution

Figure 3.20 indicates the effect of altering the number of transformed images, N, used to predict the locational uncertainty distribution, $p(\boldsymbol{\theta}'_i)$, for calculating the "hog-gb" descriptor. There is a positive correlation between the number of transformed images, N, and the descriptor performance, up to around twenty to forty images, after which the performance plateaus. I conclude that convergence of the predicted locational uncertainty distribution, $p(\boldsymbol{\theta}'_i)$, occurs around this point.

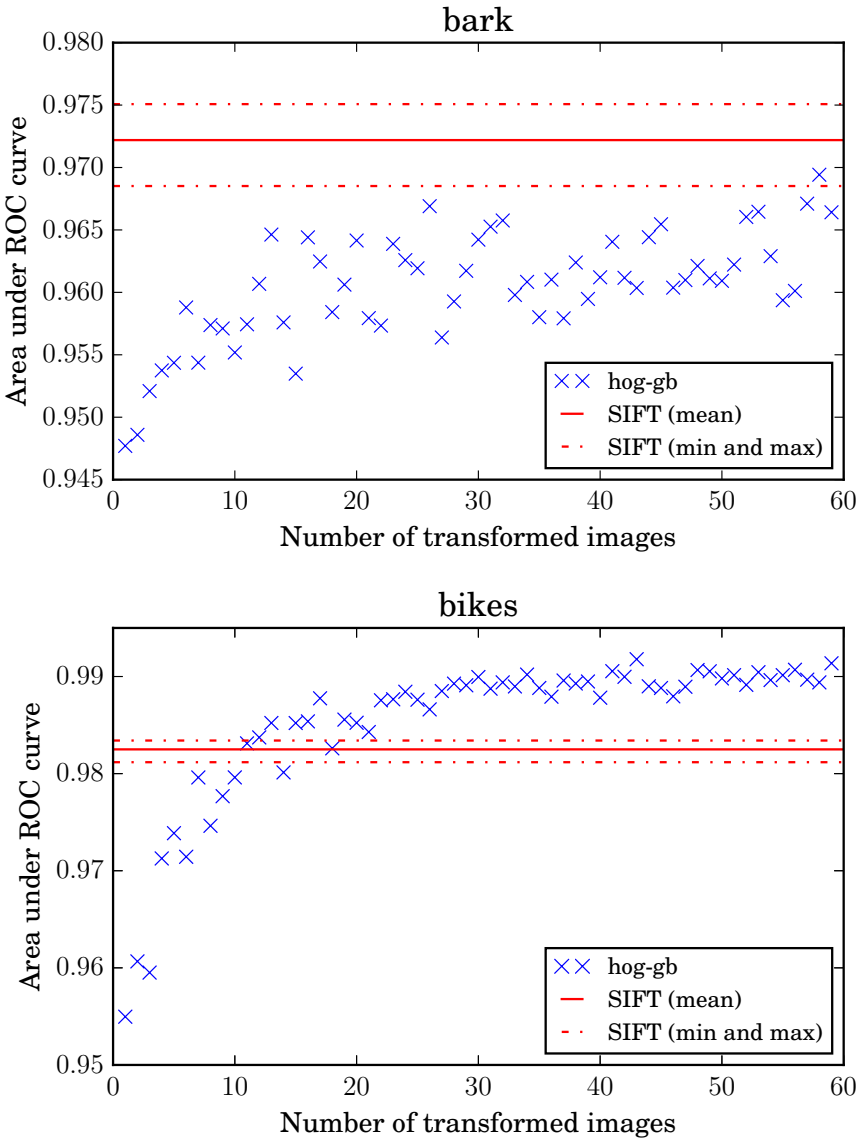

Figure 3.20: The effect of changing the number of transformed images used to predict the locational uncertainty description for "hog-gb"; the results for the SIFT descriptor are shown too: the solid line indicates the mean and the dashed lines the maximum and minimum

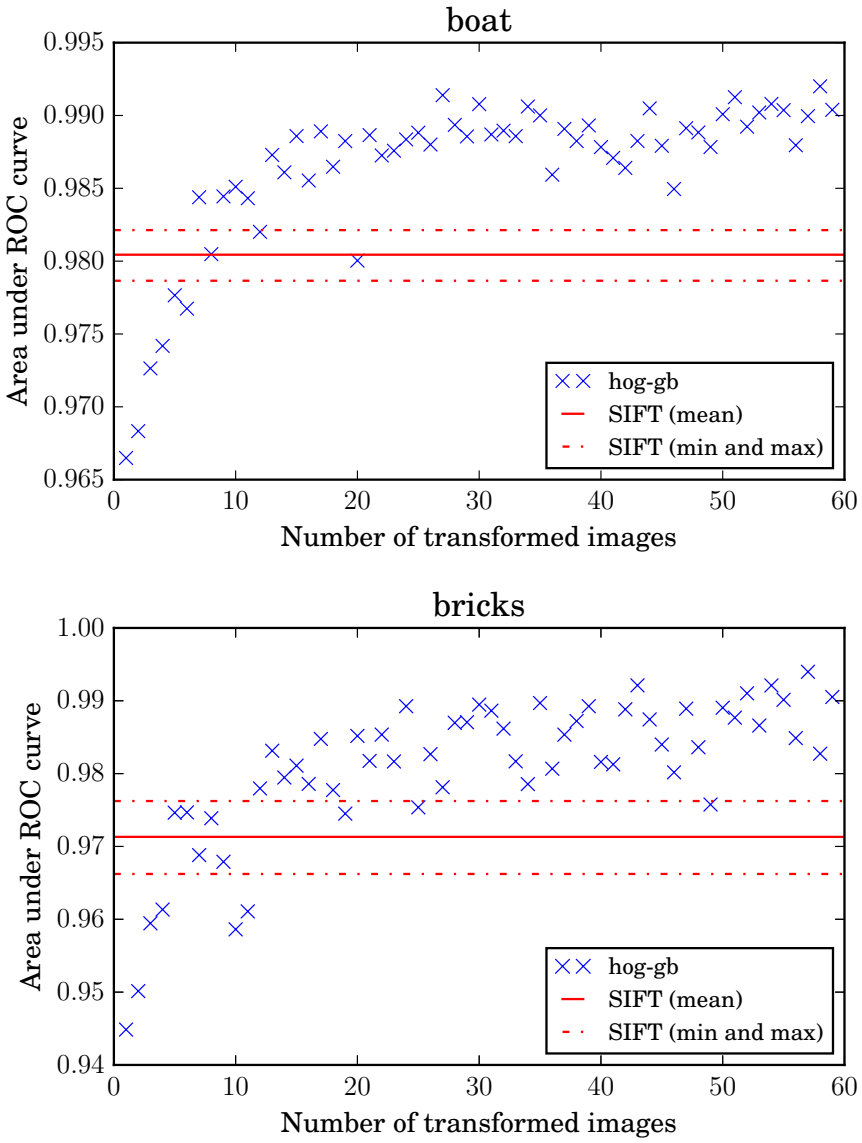

Figure 3.20: The effect of changing the number of transformed images used to predict the locational uncertainty description for "hog-gb"; the results for the SIFT descriptor are shown too: the solid line indicates the mean and the dashed lines the maximum and minimum

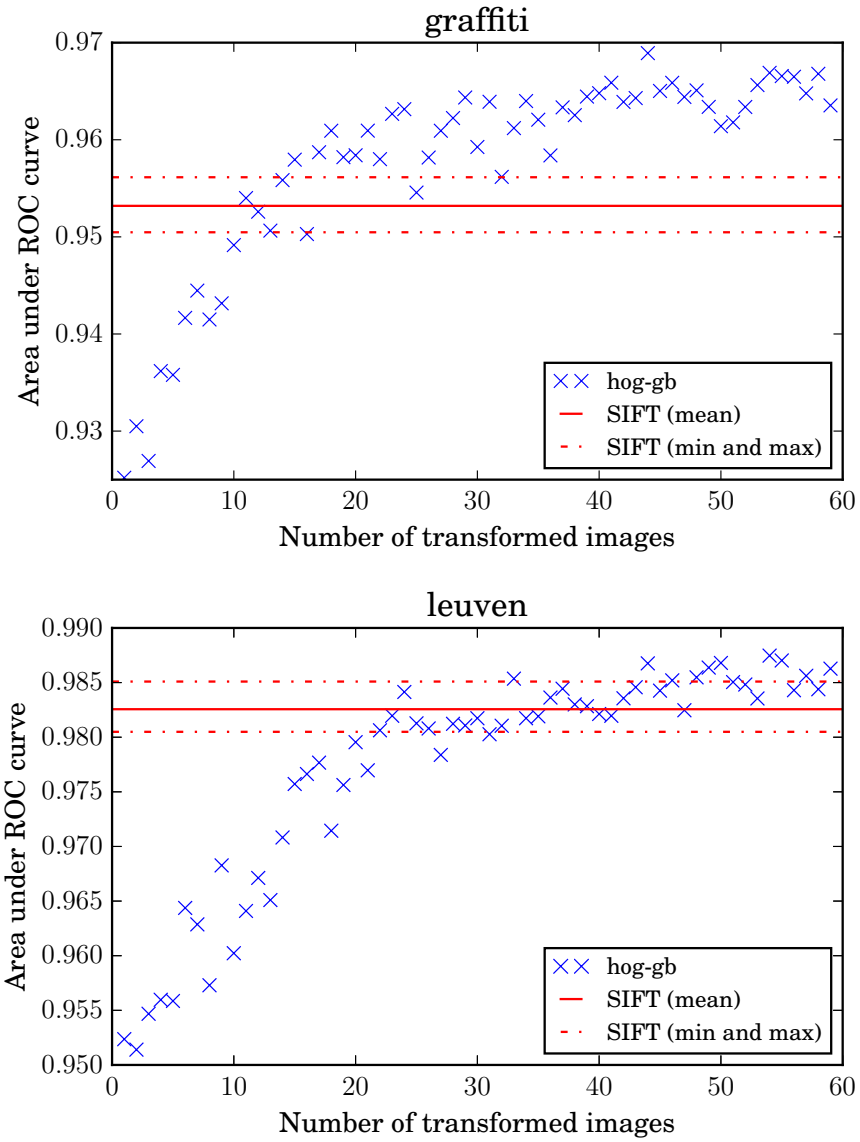

Figure 3.20: The effect of changing the number of transformed images used to predict the locational uncertainty description for "hog-gb"; the results for the SIFT descriptor are shown too: the solid line indicates the mean and the dashed lines the maximum and minimum

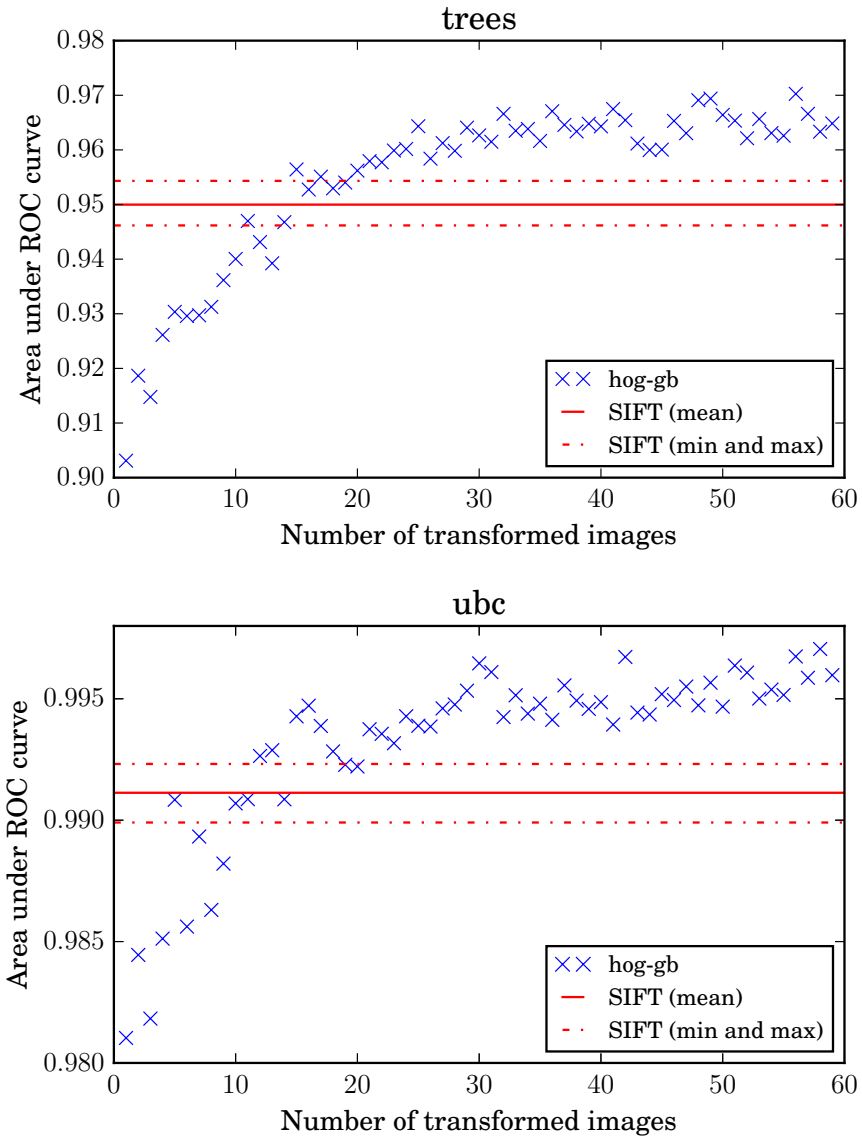

Figure 3.20: The effect of changing the number of transformed images used to predict the locational uncertainty description for "hog-gb"; the results for the SIFT descriptor are shown too: the solid line indicates the mean and the dashed lines the maximum and minimum

3.3 Limitations

Although the technique results in a matching performance improvement in some circumstances, it does not result in a universal improvement. Moreover, it suffers from the following limitations:

Descriptor extraction asymmetry The local descriptor extraction is based on a geometric blur which is specific to each interest-point. Consider matching an interest-point located in one image to one located in another. The descriptors for both interest-points would be calculated using a different geometric blur for each image. If the locational uncertainty estimate is reliable for both images, the geometric blur the technique will perform well, in spite of the asymmetry. However, if the estimate is unreliable for one or both of the images, the technique could perform worse than an identical spatial pooling across all interest-points.

Runtime speed Both the locational uncertainty estimation (Section 3.1.2.1) and the application of geometric blur (Section 3.1.3) rely on carrying out many image transforms, making them time consuming steps. Although, this can be mitigated through the use of one or more graphics processing units (GPUs), the technique is unlikely to compete on processing speed with other local descriptors.

Too strong a prior The locational uncertainty estimation procedure makes strong assumptions about the cause of the locational uncertainty (Section 3.1.2.1). When these strong assumptions do not hold even approximately, the technique is likely to perform poorly.

3.4 Conclusion

I presented a new technique for replacing the uniform low level spatial pooling operation, common to many interest-point local descriptor algorithms, with an interest-point specific low level pooling operation. This pooling operation is based on geometric blur, applied to all parameters of the interest-point and uses many synthetically warped versions of the image to estimate the individual uncertainty in the parameters of each interest-point. The technique is not specific to a single interest-point detector nor descriptor.

In some cases, the approach outperforms the rectangular pooling of the SIFT descriptor, however the performance improvement is marginal. In addition, there are other limitations, such as a slow running time, asymmetric comparison and reliance on too strong a prior. In the next chapter, I seek to address these issues.

4 Interest-Point Specific Learnt Descriptors

The work in this chapter involves the same task of matching interest-points between images (see Section 1.2.3) as the previous chapter. However, it aims to address the limitations of technique used in the previous chapter. In addition, unlike the technique used in the previous chapter, it aims to take advantage of machine learning. In this chapter, I introduce a new technique for calculating per interest-point local descriptors, which is suitable for large scale matching and image search. The resulting descriptors are known as *per interest-point (PIP) descriptors*.

Every PIP descriptor is composed of a subset of globally learnt descriptors, which I refer to as *base-descriptors*. These base-descriptors are based on those learnt using the work of Simonyan, Vedaldi and Zisserman (2014). The authors' descriptor is composed of a number of rings of pooling regions. The reason for having pooling regions is to provide spatial pooling (see Section 1.6.2), as used in other algorithms. However, the authors' contribution is to learn which pooling regions to use using a convex optimisation algorithm. (Although there is no need to maintain the ring structure, the authors find that it improves performance.)

Whereas the authors' ultimately to learn a "one size fits all" descriptor, my approach is to learn a number of base-descriptors, which can be combined in a different matter for every interest-point, producing a PIP descriptor. Each base-descriptor is designed to have a different discriminative power to invariance trade-off (Varma and Ray, 2007). I use regularised dual averaging (RDA) (L. Xiao, 2010) to select the subset of base-descriptors for each individual interest-point, as well as learning a linear weighting over the subset used for the distance function. This forms an optimal descriptor for each interest-point.

Figure 4.1 shows the pooling region rings and their weighting (by colour), of the PIP descriptor, overlaying the patch associated with each interest-point. These are learnt using the algorithm and with

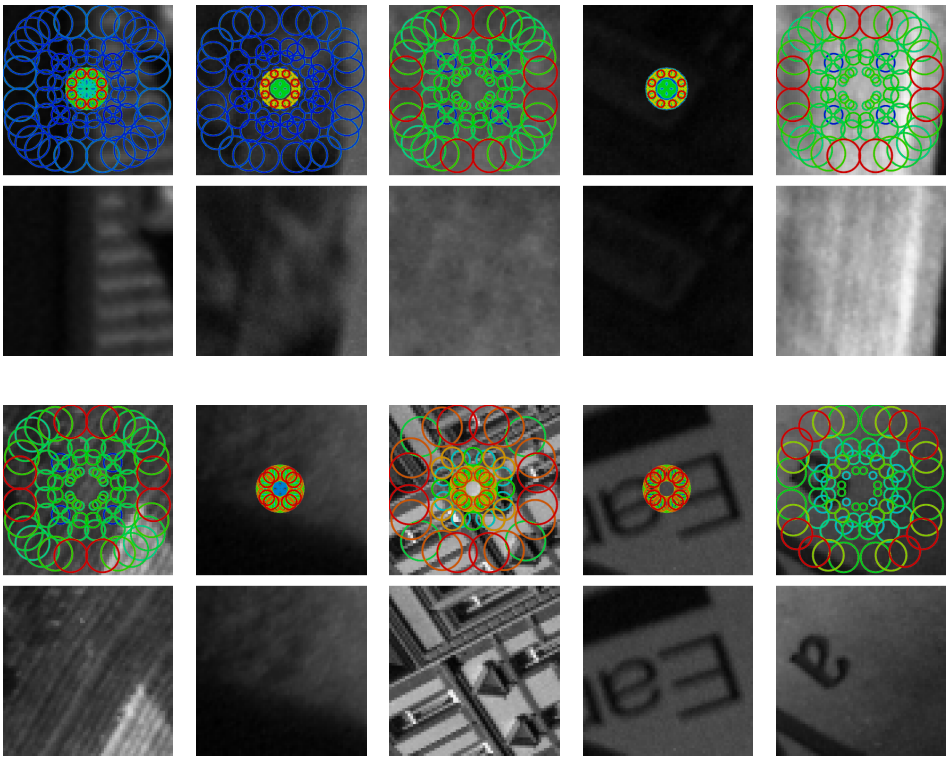

Figure 4.1: My PIP descriptors are different for each interest-point: here I show my descriptors' pooling regions and their learnt weightings (from blue, low, through green and yellow to red, high) on a number of support regions

reference to other patches of the same 3D feature viewed at different angles. However, one can observe a general trend, that more interesting and varied parts of the patch tend to have larger weighting.

My aim is to increase the accuracy of interest-point matching over large numbers of interest-points. Through the use of a small number of base-descriptors and weights, my approach scales in terms of memory storage, and is suitable for large scale image search and 3D reconstruction.

4.0.1 Comparison to Other Work

The work in this chapter uses learning to combined a number of base-descriptors, each of which are learnt using the work of Simonyan, Vedaldi and Zisserman (2012, 2014). However, in comparison to their

approach, I use different subsets of pooling region rings for each base-descriptor, rather than a single large set.

Combining base-descriptors is related to the work of Varma and Ray (2007), which itself built upon earlier approaches for combining descriptors (Bosch, Zisserman and Munoz, 2007; Lazebnik, Schmid and Ponce, 2005; Nilsback and Zisserman, 2006; J. Zhang et al., 2007) by introducing learning. The authors used a support vector machine (SVM) to learn weights over a number of descriptors and associated distance functions. Each base-descriptor was designed to yield a different *discriminative power-invariance trade-off* and an ℓ_1 regulariser was used to induce sparsity. Their approach is related to *multiple kernel learning* (F.R. Bach, Lanckriet and Jordan, 2004; Varma and Babu, 2009) however each kernel and hence weight is specific to a single descriptor and its associated features. Whereas, Varma and Ray (2007) used their approach for image classification, the work in this chapter provides an approach suitable for interest-point matching: instead of learning a classifier, I learn a weight over base-descriptors allowing comparison in a weighted Euclidean distance framework in which the weights depend on each interest-point.

The work in this chapter is closely related to other work (see Section 2.1.8) on using per interest-point learning e.g. Balntas, Tang and Mikolajczyk (2015), Gupta and Mittal (2008), Lepetit and Fua (2006) and Ozuysal et al. (2010). Lepetit and Fua (2006) and Ozuysal et al. (2010) learnt individual classifiers for each interest-point. These perform well in real-time applications with a small number of interest points (a few thousand). Due to the large storage requirements of leaf nodes, these approaches do not scale to large numbers of interest-points, i.e. 10^6 or more, which I focus on (see section 4.2.2).

Gupta and Mittal (2008) learnt a set of binary tests based on intensity values for each interest-point. This approach requires the storage of a number of 2D point pairs and associated stability factors, for each interest-point. This is likely to scale much better than the aforementioned approaches. However, the requirement to perform a set a binary tests for matching local descriptors means that it is not suitable for rapid matching of interest-point and perhaps only suitable for a verification stage. In comparison, the work in this chapter uses a weighted Euclidean distance which is more suitable for large scale matching.

Balntas, Tang and Mikolajczyk (2015) learnt a binary mask for each interest-point which is then used as part of a per interest-point

Hamming distance calculation. This method is most similar to that of this chapter and can be seen as a binary per interest-point descriptor.

Whereas the work in this chapter involves learning weights over base-descriptors, Balntas, Tang and Mikolajczyk (2015) learn a bit map, effectively zero or one weighting, over a set of binary tests. As with binary local descriptors compared to their floating point counterparts, Balntas, Tang and Mikolajczyk (2015) will run more efficiently in both computational and memory expense than the work in this chapter, at the cost of matching accuracy.

4.0.2 Regularised Dual Averaging

Regularised dual averaging (RDA) is a class of online learning algorithms which can better exploit the regularisation structure of an optimising function compared to other methods such as stochastic gradient descent (SGD). In particular, under ℓ_1 regularisation, which I use in this chapter, RDA is better suited than SGD for achieving sparsity, which is desirable for the PIP descriptor. I use RDA (L. Xiao, 2010) both for the global learning and the per interest-point learning. The use of RDA in the global learning case is identical to the use in Simonyan, Vedaldi and Zisserman (2014), except for a change in the set of candidate pooling regions and hence the dimensionality of the optimisation variable (a set of weights). Both cases involve optimising $\mathbf{w}$ in the following loss function:

$$\arg\min_{\mathbf{w}}\{\eta(\mathbf{w})\} \tag{4.1}$$

$$= \arg\min_{\mathbf{w}}\{E_{\mathbf{d}\in D}[f(\mathbf{w})] + \mu\|\mathbf{w}\|_1\}, \tag{4.2}$$

$$= \arg\min_{\mathbf{w}}\big\{E_{\mathbf{d}\in D}\big[\max\big(0, 1 + \mathbf{w}^\top\mathbf{d}\big)\big] + \mu\|\mathbf{w}\|_1\big\}, \tag{4.3}$$

where $\mathbf{w}$ is a vector of weights and $\mu\|\mathbf{w}\|_1$ is an ℓ_1 regularisation term.

For the purposes of explaining the use of RDA, consider $\mathbf{d}$ to be a sample of D, an independent and identically distributed random variable. In this chapter, however, it will be a vector of distances between the appearance of local image patches as calculated using different descriptors or pooling regions. Some descriptors and associated distance metrics will yield better results than others, and this motivates using RDA to optimise $\mathbf{w}$, which will result in selection and weighting the descriptors. As RDA is an online method, optimisation will occur

through processing values of $\mathbf{d}$ sequentially and updating $\mathbf{w}$ at each step.

The loss function (4.3) is convex, however RDA requires a strongly convex loss function, i.e. there exists an $\iota > 0$ such that

$$\Phi(\alpha b + (1-\alpha)\, c) \leq \alpha\Phi(b) + (1-\alpha)\,\Phi(c) - \tfrac{\iota}{2}\alpha\,(1-\alpha)\,\|b-c\|^2, \forall b, c \in \operatorname{dom}\Phi. \tag{4.4}$$

This is not the case for (4.3), however this can be resolved by adding the strongly convex term,

$$\frac{\beta_{(t)}}{2t}\|\mathbf{w}\|_2^2, \tag{4.5}$$

where, to ensure convergence, $\beta_{(t)}$ is a nonnegative and nondecreasing sequence for time steps, $t = 1, 2, 3, \ldots$. In all cases, I use the "enhanced ℓ_1 RDA method", for which

$$\beta_{(t)} = \gamma\sqrt{t}, \tag{4.6}$$

where γ is a hyper-parameter.

At each time step, t, the "enhanced ℓ_1 RDA method" for the above loss function operates as follows:

1. Calculate the subgradient,

$$\mathbf{g}_{(t)} = \frac{\partial f_{(t)}}{\partial \mathbf{w}} = \begin{cases} \mathbf{d} & \text{if } \mathbf{w}_{(t)}^\top \mathbf{d} > -1 \\ 0 & \text{otherwise.} \end{cases} \tag{4.7}$$

   where d is randomly sampled from D.

2. Update the dual average (so called because the subgradients live in the dual space of $\mathbf{w}$),

$$\bar{\mathbf{g}}_{(t)} = \frac{t-1}{t}\bar{\mathbf{g}}_{(t-1)} + \frac{1}{t}\mathbf{g}_{(t)} \tag{4.8}$$

3. Update the weights,

$$\mathbf{w}_{(t+1)} = \arg\min_{\mathbf{w}}\left\{\mathbf{w}_{(t+1)}^\top \bar{\mathbf{g}}_{(t)} + \mu\|\mathbf{w}\|_1 + \frac{\gamma}{2\sqrt{t}}\|\mathbf{w}\|_2^2\right\}, \tag{4.9}$$

   which (through differentiation to find the minimum) yields the closed-form solution,

$$\mathbf{w}_{(t+1)} = \max\left[0, \frac{-\sqrt{t}}{\gamma}\left(\bar{\mathbf{g}}_{(t)} + \mu\right)\right]. \tag{4.10}$$

Initially,

$$\bar{\mathbf{g}}_{(0)} = 0, \mathbf{w}_{(0)} = \mathbf{0}. \tag{4.11}$$

With $\beta_{(t)} = \gamma\sqrt{t}$, it can be shown that the algorithm has the following convergence rate on average weights, $\bar{\mathbf{w}}$ compared to the optimal solution η^*,

$$\eta\big(\bar{\mathbf{w}}_{(t)}\big) - \eta^* \leq O\bigg(\frac{G}{\sqrt{t}}\bigg), \tag{4.12}$$

where G is a uniform upper bound on the norms of the subgradient $\mathbf{g}_{(t)}$. The proof is long and can be found in Sections 3–4 of L. Xiao (2010).

4.1 Methodology

I build upon the descriptor computation pipeline used by Simonyan, Vedaldi and Zisserman (2014), which is a refined version of the pipelines used by Brown, Hua and Winder (2011) and Lowe (2004). This operates on a patch extracted from an interest-point support region (patch) of a greyscale image as follows:

1. Apply a Gaussian blur, as used by Brown, Hua and Winder (2011)

2. Extract local intensity gradients at each pixel, and bin them into eight orientation channels,as first used in SIFT (Lowe, 2004), to yield a eight-channel image, containing low level features (see Section 1.6.1)

3. Normalise gradient magnitudes across the entire patch based on the quartile statistic; this is novel to Simonyan, Vedaldi and Zisserman (2014)

4. Spatially pool the gradient magnitudes, independently for each orientation channel, using isotropic Gaussian pooling regions, as used by Tola, Lepetit and Fua (2010) and Brown, Hua and Winder (2011): each pooling region forms an eight-dimensional feature set, i.e. one for each orientation channel

5. Apply a linear weighting to each of the pooling regions: each weight is shared across a ring of either four or eight rotationally symmetric pooling regions

The selection and weighting of these pooling region rings is the product of machine learning. The learning algorithm selects from a large but finite set of candidate pooling region rings, optimising performance when matching descriptors over a large dataset. This results in a single descriptor.

My approach, on the other hand, is to learn multiple descriptors, each using smaller and non-overlapping sets of candidate pooling region rings, as outlined in the next section. I use the same approach as Simonyan, Vedaldi and Zisserman (2014), as described in Section 4.1.2, to carry out this learning. I then perform additional learning at run-time, per interest-point, to combine these base-descriptors into a PIP descriptor, as described in Section 4.1.3.

4.1.1 Base-Descriptors

Simonyan, Vedaldi and Zisserman (2014) uses five configurations of pooling region rings, all of which are symmetrical along the horizontal, vertical and both diagonal centre lines. The five configurations are specified based on the angular offset, α. This is the angle from the centre of the ring, between the x axis and the first isotropic Gaussian pooling region. it is located at one of the following angular offsets from the x axis:

$$\alpha = \left\{0, \frac{\pi}{16}, \frac{\pi}{8}, \frac{3\pi}{16}, \frac{\pi}{4}\right\}. \tag{4.13}$$

By symmetry, this yields two ring configurations which contain four pooling regions, and three which contain eight pooling regions.

In addition to the angular offset, α, each pooling region ring has two additional parameters:

r, the radius of the ring on which all the centres of the pooling regions lie

σ, the standard deviation of the isotropic Gaussian pooling regions

In my approach, I learn ten separate base-descriptors in place of the single descriptor of Simonyan, Vedaldi and Zisserman (2014), which I then combine using linear weighting learnt at run-time to form PIP descriptors. The base-descriptors differ from the single descriptor in two ways:

1. Each base-descriptor has only one value for the standard deviation, σ, for each radius, r

2. Each base-descriptor covers only a limited range of radii.

The base-descriptors can be interpreted as non-intersecting subsets of the original descriptor. The smaller training data in per interest-point descriptor learning problems, perhaps as few as four exemplars, necessitates this succinct set of ten base-descriptors.

There are two categories of base-descriptor:

Fixed sigma (FS) descriptors use isotropic Gaussian pooling regions lying close to the centre of the support region with a fixed standard deviation, σ, such as those in Figure 4.2: this allows for varying amounts of misalignment of the centre of interest-points between images.

Fixed sigma ratio (FSR) descriptors which use isotropic Gaussian pooling regions far from the centre of the support region with a fixed ratio, $\sigma = kr$, such as those in Figure 4.3: as noted by Berg and Malik (2001), the geometric error is larger further from the descriptor; the FSR base-descriptors allow for this.

There are five of each, yielding ten base-descriptors in total. Figures 4.2 and 4.3 show examples of pooling ring candidates. Table 4.1 contains details of candidate pooling region rings used to form each base-descriptor.

base-descriptor	**radii** (r)	**stdev** (σ)	**angular offset** (α)
Simonyan et al. (2014)	$\{0, 1, \ldots, 31\}$	$\{0.5,1,\ldots,16\}$	$\left\{0, \frac{\pi}{16}, \frac{\pi}{8}, \frac{3\pi}{16}, \frac{\pi}{4}\right\}$
FS0	$\{1, 2, \ldots, 6\}$	1.0	$\left\{0, \frac{\pi}{16}, \frac{\pi}{8}, \frac{3\pi}{16}, \frac{\pi}{4}\right\}$
FS1	$\{1, 2, \ldots, 6\}$	1.5	$\left\{0, \frac{\pi}{16}, \frac{\pi}{8}, \frac{3\pi}{16}, \frac{\pi}{4}\right\}$
FS2	$\{1, 2, \ldots, 6\}$	2.0	$\left\{0, \frac{\pi}{16}, \frac{\pi}{8}, \frac{3\pi}{16}, \frac{\pi}{4}\right\}$
FS3	$\{1, 2, \ldots, 6\}$	2.5	$\left\{0, \frac{\pi}{16}, \frac{\pi}{8}, \frac{3\pi}{16}, \frac{\pi}{4}\right\}$
FS4	$\{1, 2, \ldots, 6\}$	3.0	$\left\{0, \frac{\pi}{16}, \frac{\pi}{8}, \frac{3\pi}{16}, \frac{\pi}{4}\right\}$
FSR0	$\{7, 8, \ldots, 31\}$	$0.2r$	$\left\{0, \frac{\pi}{16}, \frac{\pi}{8}, \frac{3\pi}{16}, \frac{\pi}{4}\right\}$
FSR1	$\{7, 8, \ldots, 31\}$	$0.3r$	$\left\{0, \frac{\pi}{16}, \frac{\pi}{8}, \frac{3\pi}{16}, \frac{\pi}{4}\right\}$
FSR2	$\{7, 8, \ldots, 31\}$	$0.4r$	$\left\{0, \frac{\pi}{16}, \frac{\pi}{8}, \frac{3\pi}{16}, \frac{\pi}{4}\right\}$
FSR3	$\{7, 8, \ldots, 31\}$	$0.5r$	$\left\{0, \frac{\pi}{16}, \frac{\pi}{8}, \frac{3\pi}{16}, \frac{\pi}{4}\right\}$
FSR4	$\{7, 8, \ldots, 31\}$	$0.6r$	$\left\{0, \frac{\pi}{16}, \frac{\pi}{8}, \frac{3\pi}{16}, \frac{\pi}{4}\right\}$

Table 4.1: Candidate pooling region rings for PIP's base-descriptors and for those of Simonyan, Vedaldi and Zisserman (2014)– the latter's candidate pooling regions are the Cartesian product of a range of radii and standard deviations, while PIP pooling regions have a fixed expression or constant value for the standard deviation.

The PIP base-descriptor design reflects the following two assumptions:

1. The main requirement for spatial pooling in the centre of the support region is to compensate misalignment errors caused by the interest-point detector.

2. Away from the centre of the support region, differences in viewing angle become the dominant purpose for pooling, roughly proportional to do the distance from the centre of the support region: this is supported by *geometric blur* (Berg and Malik, 2001), the "DAISY" descriptor (Tola, Lepetit and Fua, 2010) and the pooling regions selected by Simonyan, Vedaldi and Zisserman (2014).

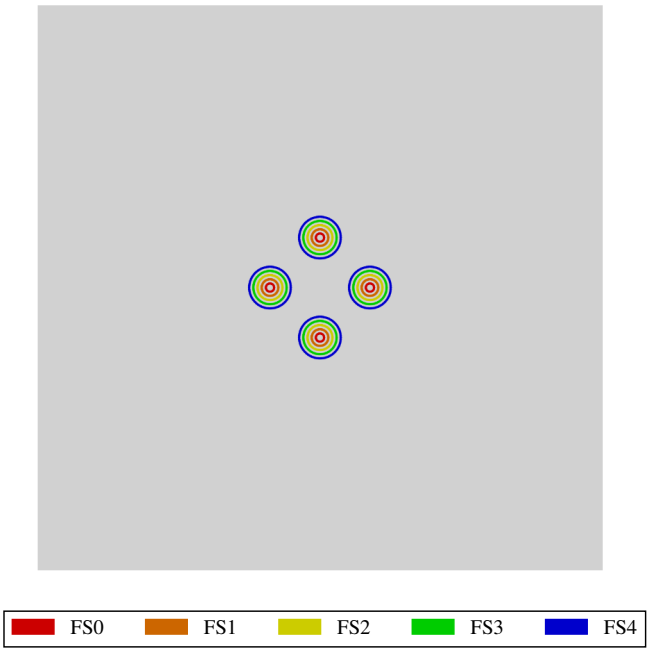

Figure 4.2: The pooling candidate ring at $r{=}6,\alpha{=}0$ for each FS base-descriptor

4.1.2 Global Learning

Following Simonyan, Vedaldi and Zisserman (2014), I use RDA (L. Xiao, 2010) (see Section 4.0.2) to carry out global learning, learning weights over and selecting pooling region candidate rings. Each base-descriptor is learnt independently, using a set of matching and non-matching patch pairs, split into training and validation sets. (Every patch is located at an interest-point support region.) The operation is as follows:

1. I learn a sparse set of weights over the pooling region candidate rings using RDA, as described in Section 4.1.2.1.

2. I learn a linear projection matrix, also using RDA, to achieve dimensionality reduction, over the pooling regions with non-zero weighting, as described in Section 4.1.2.2

3. I use the pooling regions with a non-zero weights and the linear projection matrix, learnt in the previous steps, to form the base-descriptor.

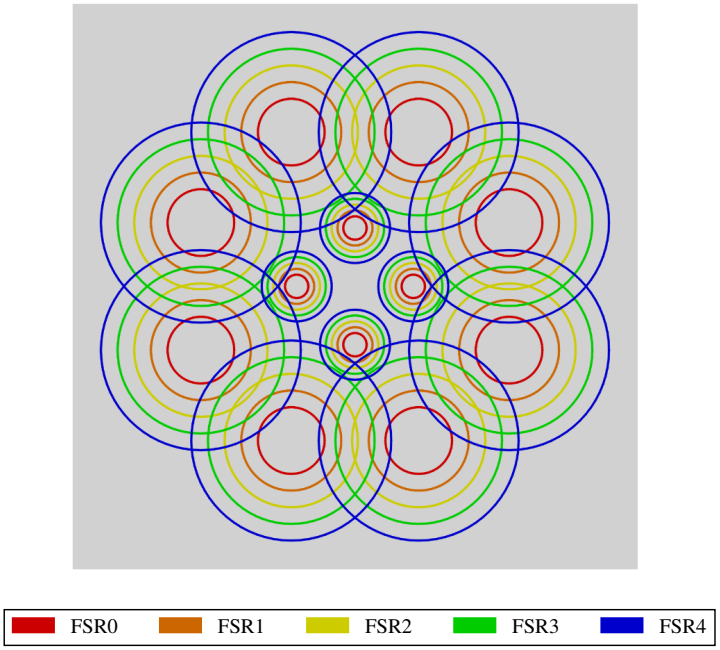

Figure 4.3: Two pooling candidate rings at r=7,α=0 and r=20,$\alpha=\frac{\pi}{8}$ respectively, for each FSR base-descriptor

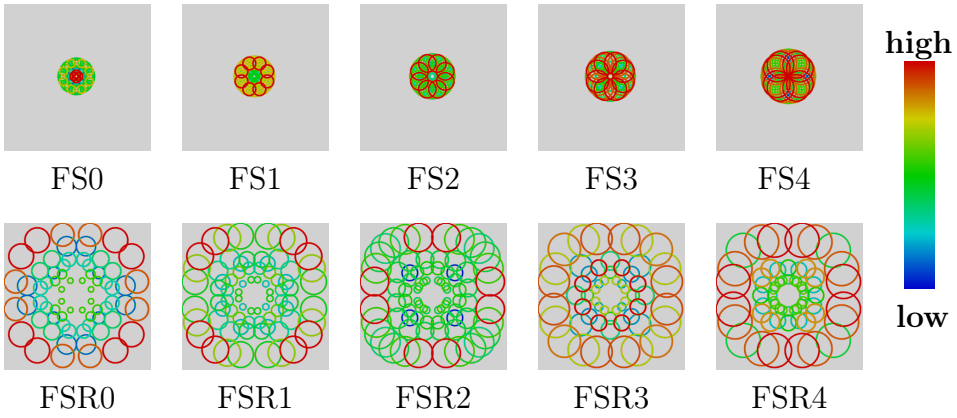

Figure 4.4: The PIP base-descriptors formed by global training on the "liberty" dataset: the circles indicate isotropic Gaussian pooling regions with radii proportional to the standard deviation and colour to represent the weighting; for clarity, the radii are doubled for the FS descriptors

4.1.2.1 Learning the weights

The first step involves selecting and learning weights for the pooling candidate regions for each base-descriptor. The method for calculating a descriptor using Simonyan, Vedaldi and Zisserman (2012, 2014) is outlined in Section 2.1.6. Starting with a patch, applying a Gaussian Blur, extracting local intensity gradients, binning the gradients into 8 orientations and normalising them results in an unpooled descriptor. Following Simonyan, Vedaldi and Zisserman (2012, 2014), a pooled descriptor is $\Phi_{i,j,c}(\mathbf{x})$ and indexed over pooling region ring i, pooling region index within a ring, j and channel (orientation bin), c.

The set of pooling regions is Ω_i, where i again indexes over pooling region rings. Differing from Simonyan, Vedaldi and Zisserman (2012, 2014), which has one set of pooling region rings, Ω, I have N sets, $\Omega_{h,i}$, where $h = 1, 2, \cdots, N$ indexes over N base-descriptors ($N = 10$) corresponding to my ten base-descriptors throughout my experiments), leading to N learnt base-descriptors, $\rho_{h,i,j,c}(\mathbf{x})$. However, as each base-descriptor is learnt independently and identically, I will drop the index h for simplicity until I describe the per interest-point learning.

Each learned descriptor,

$$\rho_{i,j,c}(\mathbf{x}) = \sqrt{w_i}\Phi_{i,j,c}(\mathbf{x}), \tag{4.14}$$

is a weighted version of the "full" pooled descriptor, $\Phi_{i,j,c}(\mathbf{x})$. The weights, $\mathbf{w} = \begin{bmatrix} w_1 & w_2 & \cdots \end{bmatrix}^\top$ are all nonnegative (and zero for pooling region rings that are not selected). There is a single weight, w_i, for each ring of pooling regions. Hence, the number of weights corresponds to the number of candidate pooling region rings, i.e. the size of the Cartesian product of a single row in Table 4.1.

Comparison between descriptors occurs using the squared ℓ_2 distance,

$$\begin{aligned} d(\mathbf{x}, \mathbf{y}) &= \|\rho(\mathbf{x}) - \rho(\mathbf{y})\|_2^2 && (4.15)\\ &= \sum_{i,j,c} \left(\sqrt{w_i}\Phi_{i,j,c}(\mathbf{x}) - \sqrt{w_i}\Phi_{i,j,c}(\mathbf{y})\right)^2 && (4.16)\\ &= \sum_i w_i \sum_{j,c} \left(\Phi_{i,j,c}(\mathbf{x}) - \Phi_{i,j,c}(\mathbf{y})\right)^2 && (4.17)\\ &= \mathbf{w}^\top \psi(\mathbf{x}, \mathbf{y}), && (4.18) \end{aligned}$$

where $\psi(\mathbf{x}, \mathbf{y})$ is a vector containing the squared distances corresponding to each pooling region ring Ω_i, so that the ith component of the

vector,

$$\psi_i = \sum_{j,c} \left(\Phi_{i,j,c}(\mathbf{x}) - \Phi_{i,j,c}(\mathbf{y})\right)^2. \tag{4.19}$$

The objective function used to learn the weights is as follows:

$$\arg\min_{w \geq 0} \left\{ \sum_{\substack{(\mathbf{x},\mathbf{y}) \in \mathcal{P} \\ (\mathbf{u},\mathbf{v}) \in \mathcal{N}}} \ell\big(\mathbf{w}^\top \left(\psi(\mathbf{x},\mathbf{y}) - \psi(\mathbf{u},\mathbf{v})\right)\big) + \mu \|\mathbf{w}\|_1 \right\}, \tag{4.20}$$

where $\mathcal{P}$ and $\mathcal{N}$ are the training sets of positive (matching) and negative (non-matching) patch pairs, $\ell(z) = \max(0, z+1)$, is a hinge loss function and $\mu\|\mathbf{w}\|_1$ is an ℓ_1 regularisation term. This objective encourages every negative patch pair to be further apart in the descriptor space than every positive patch pair by at least a distance of unity, while simultaneously encouraging sparsity in the weights, $\mathbf{w}$.

As in Simonyan, Vedaldi and Zisserman (2012, 2014), I solve the objective using RDA, following the approach outlined in Section 4.0.2. I maximise the objective on a given training set for many combinations of μ and γ and then select the values which maximise the false positive rate (FPR) at 95% recall on the associated validation set. The values used are the Cartesian product of

$$\mu = \{0.05, 0.1, 0.15, \ldots, 1.0\} \tag{4.21}$$

and

$$\gamma = \left\{2^{-1}, 1, 2, 2^2, 2^3, 2^4\right\}. \tag{4.22}$$

4.1.2.2 Learning a linear projection matrix

The next step involves dimensionality reduction applying a matrix, $\mathbf{W} \in \mathcal{R}^{m \times n}$, where n is the number of non-zero elements in $\mathbf{w}$ and $m < n$. The matrix is applied only to the elements of the descriptor, $\Phi_{i,j,c}(\mathbf{x})$ for which the associated weight, w_i is non-zero, indicated by $\Phi'_{i,j,c}(\mathbf{x})$ resulting in a descriptor,

$$\boldsymbol{\phi}(\mathbf{x}) = \mathbf{W}\Phi'(\mathbf{x}). \tag{4.23}$$

The squared ℓ_2 distance between two descriptors,

$$d_A(\mathbf{x},\mathbf{y}) = \|\mathbf{W}\Phi'(\mathbf{x}) - \mathbf{W}\Phi'(\mathbf{y})\|_2^2 = \theta(\mathbf{x},\mathbf{y})^\top \mathbf{A}\theta(\mathbf{x},\mathbf{y}), \tag{4.24}$$

where $\theta(\mathbf{x}, \mathbf{y}) = \Phi'(\mathbf{x}) - \Phi'(\mathbf{y})$ and $\mathbf{A} = \mathbf{W}^\top \mathbf{W}$. The optimising of $\mathbf{W}$ occurs by the convex optimisation of $\mathbf{A}$, with the following optimisation function:

$$\arg\min_{\mathbf{A} \succeq 0} \left\{ \sum_{\substack{(\mathbf{x},\mathbf{y}) \in \mathcal{P} \\ (\mathbf{u},\mathbf{v}) \in \mathcal{N}}} \ell\Big(\theta(\mathbf{x}, \mathbf{y})^\top \mathbf{A} \theta(\mathbf{x}, \mathbf{y}) - \theta(\mathbf{u}, \mathbf{v})^\top \mathbf{A} \theta(\mathbf{u}, \mathbf{v})\Big) + \mu_* \|\mathbf{W}\|_* \right\}, \tag{4.25}$$

where $\mathbf{A} \succeq 0$ means that $\mathbf{A}$ is positive semidefinite and $\| \dots \|_*$ is the matrix norm. The matrix norm is a convex surrogate of the rank of the matrix and hence the term encourages $\mathbf{W}$ to form a lower dimensional embedding. Hence, μ_* is a hyper-parameter which trades off the hinge loss term the rank of the matrix.

This objective is also solved using RDA. However, because of the use of a matrix rather than a vector, the update steps differ from Section 4.0.2. The subgradient calculation, in place of (4.7), is as follows:

$$\mathbf{g}_{(t)} = \begin{cases} \theta(\mathbf{x}, \mathbf{y}) \theta(\mathbf{x}, \mathbf{y})^\top - \theta(\mathbf{u}, \mathbf{v}) \theta(\mathbf{u}, \mathbf{v})^\top, & \text{if } d_A(\mathbf{x}, \mathbf{y}) + 1 > d_A(\mathbf{u}, \mathbf{v}) \\ 0, & \text{otherwise,} \end{cases} \tag{4.26}$$

where $(\mathbf{x}, \mathbf{y}) \in \mathcal{P}$ and $(\mathbf{u}, \mathbf{v}) \in \mathcal{N}$ are randomly sampled at each time step. Similarly, the update step for $\mathbf{A}$, in place of (4.9), is as follows:

$$A_{t+1} = \Pi\left(-\frac{\sqrt{t}}{\gamma} \big(\bar{\mathbf{g}}_{(t)} + \mu_* \mathbf{I} \big) \right), \tag{4.27}$$

where $\mathbf{I}$ is the identity matrix and $\Pi \colon \mathcal{R}^{n \times n} \to \mathcal{R}^{n \times n}_{\text{PSD}}$ is a projection from $\mathcal{R}^{n \times n}$ onto the cone of positive semidefinite matrices which occurs by eigenvalue decomposition, setting all negative eigenvalues to zero, and recomposing the matrix.)

If the eigenvalue decomposition of $\mathbf{A}$ is $\mathbf{Q} \mathbf{\Lambda} \mathbf{Q}^{-1}$ then, since $\mathbf{A} = \mathbf{W}^\top \mathbf{W}$, $\mathbf{W}$ can be obtained by calculating. $\sqrt{\mathbf{\Lambda}} \mathbf{Q}^{-1}$, and removing the rows which are equal to $\mathbf{0}^\top$. Hence, m is determined by the number of non-zero eigenvalues of $\mathbf{A}$ and the resulting value will depend on the value of hyper-parameter, μ_*.

I maximise the objective on the same training set as used for learning the weights using many values for μ_* and γ and then select the values maximise the FPR at 95% recall on the associated validation set. The values used are the Cartesian product of

$$\mu_* = \{0.001, 0.0015, 0.002, 0.0025, 0.003\} \tag{4.28}$$

and

$$\gamma = \left\{ 2^{-4}, 2^{-3}, 2^{-2}, 2^{-1}, 1, 2 \right\}. \tag{4.29}$$

Figure 4.4 shows the base-descriptors learnt on the "liberty" dataset (see Section 4.2.1.1), prior to dimensionality reduction. After dimensionality reduction, FS base-descriptors have 46 dimensions on average and FSR base-descriptors have 66 dimensions on average. The original descriptors (Simonyan, Vedaldi and Zisserman, 2014) have 74 dimensions on average.

4.1.3 Per Interest-Point Learning

The aim is to learn a descriptor for each interest-point which is optimal for that interest-point. This is equivalent to learning a descriptor for each patch associated with the interest-point. Here, it is assumed that multiple positive patches exist for a given interest-point in an image, perhaps obtained from a prior reconstruction or produced through applying synthetic jitter to the original image. In addition, the availability of negative patches from non-matching interest-points is also assumed, e.g. other locations in the same image, which in the absence of reflections are guaranteed to be negative.

4.1.3.1 Combining the Base-Descriptors

The aim is to find an optimal descriptor for a given patch, $\mathbf{O}$ (the original patch); if this patch is the support region for interest-point, $\boldsymbol{\theta}_o$, this also corresponds to finding an optimal descriptor for $\boldsymbol{\theta}_o$. Let $\mathbf{X}$ and $\mathbf{Y}$ be patches; let

$$\boldsymbol{\phi_1}(\mathbf{X}), \boldsymbol{\phi_2}(\mathbf{X}), \cdots, \boldsymbol{\phi_N}(\mathbf{X}) \tag{4.30}$$

be the set of N base-descriptors. These are the formed from base-descriptors learnt in the previous section, i.e. (4.23) with the learnt dimensionality reduction. In addition, let

$$d_1(\mathbf{X}, \mathbf{Y}), d_2(\mathbf{X}, \mathbf{Y}), \cdots, d_N(\mathbf{X}, \mathbf{Y}) \tag{4.31}$$

be the set of N associated distance functions. Though many distance functions could be used, I always use the Euclidean distance (ℓ_2 norm), therefore

$$d_n(\mathbf{X}, \mathbf{Y}) = \|\phi_n(\mathbf{X}) - \phi_n(\mathbf{Y})\|_2 \tag{4.32}$$

in all cases.

I aim to learn a set of nonnegative weights, $\mathbf{w} = \begin{bmatrix} w_1 & w_2 & \cdots \end{bmatrix}^\top$, over the vector of distances for each base-descriptor between $\mathbf{O}$ and another patch, $\mathbf{Z}$,

$$\mathbf{d}(\mathbf{O}, \mathbf{Z}) = \begin{bmatrix} d_1(\mathbf{O}, \mathbf{Z}) \\ \cdots \\ d_N(\mathbf{O}, \mathbf{Z}) \end{bmatrix}. \tag{4.33}$$

The results in an optimal PIP descriptor, $\boldsymbol{\phi}_{(\text{opt})}$, for the patch $\mathbf{O}$, with distance function,

$$d_{(\text{opt})}(\mathbf{O}, \mathbf{Z}, \mathbf{w}) = \sum_{n=1}^{N} w_n d_n(\mathbf{O}, \mathbf{Z}) = \mathbf{w}^\top \mathbf{d}(\mathbf{O}, \mathbf{Z}). \tag{4.34}$$

Zero weighting of the nth base-descriptor, $w_n = 0$, implies that it does not feature in the PIP descriptor, resulting in selection of a few base-descriptors or perhaps one single base-descriptor.

4.1.3.2 Learning the weights

Let $\mathcal{P}$ be a set of positive (matching) patches and let $\mathcal{N}$ be a set of negative (non-matching) patches, i.e. those located at interest-points matching or not matching the original interest-point, $\boldsymbol{\theta}_o$, respectively. Following the approach used by Simonyan, Vedaldi and Zisserman (2014), inspired by Weinberger and Saul (2009), I place a hinge loss function between every pair of positive and negative patches. As with the global learning, there is an ℓ_1 regularisation term to induce a sparse solution, i.e. one in which $w_n = 0$ for many n. This, in effect, selects a few or a single base-descriptor(s), reducing the dimensionality of each PIP descriptor and preventing over-fitting.

As a shorthand, let the *difference of distances* function,

$$\mathbf{d}(\mathbf{P}, \mathbf{N}, \mathbf{O}) = \mathbf{d}(\mathbf{P}, \mathbf{O}) - \mathbf{d}(\mathbf{N}, \mathbf{O}) \tag{4.35}$$

$$= \begin{bmatrix} d_1(\mathbf{P}, \mathbf{O}) \\ \cdots \\ d_N(\mathbf{P}, \mathbf{O}) \end{bmatrix} - \begin{bmatrix} d_1(\mathbf{N}, \mathbf{O}) \\ \cdots \\ d_N(\mathbf{N}, \mathbf{O}). \end{bmatrix} \tag{4.36}$$

The objective is to learn a set of weights, $\mathbf{w}$, which satisfy

$$\arg\min_{w_n \geq 0} \left\{ \sum_{\substack{\mathbf{P} \in \mathcal{P} \\ \mathbf{N} \in \mathcal{N}}} \ell\big(\mathbf{w}^\top [\mathbf{d}(\mathbf{P}, \mathbf{N}, \mathbf{O})]\big) + \mu \|\mathbf{w}\|_1 \right\} \tag{4.37}$$

where, as before,

$$\ell(z) = \max(0, z + 1), \tag{4.38}$$

is a hinge loss function and $\mu\|\mathbf{w}\|_1$ is an ℓ_1 regularisation term. As before, this objective encourages every negative patch, $\mathbf{N}$, to be further from the original patch, $\mathbf{O}$, than every positive patch, $\mathbf{P}$, by at least a distance of unity, while simultaneously encouraging sparsity in the weights, $\mathbf{w}$. The hyper-parameter μ allows control over the trade-off between the two terms.

This is identical to the hinge loss term of the global learning objective (4.26), with $\mathbf{x} = \mathbf{P}$ and $\mathbf{u} = \mathbf{N}$ except that $\mathbf{y} = \mathbf{v} = \mathbf{O}$ because the objective is to learn weights for positive and negatives with respect to the same patch, $\mathbf{O}$, rather than between many sets of matching and non-matching patch pairs.

I use the same RDA learning process to learn the weights as for the global learning in Section 4.1.2.1. For each iteration, I select a random positive and negative patch pair, $\mathbf{P} \in \mathcal{P}$ and $\mathbf{N} \in \mathcal{N}$, respectively, and proceeds as follows:

1. Update the average gradient of the objective with respect to the weights, w, $\bar{\mathbf{g}}$, at time t as follows:

$$\bar{\mathbf{g}}_{(t)} = \begin{cases} \frac{(t-1)\bar{\mathbf{g}}_{(t-1)} + \mathbf{d}(\mathbf{P},\mathbf{N},\mathbf{O})}{n}, & \text{if } \mathbf{w}_{(t-1)}^{\top}\mathbf{d}(\mathbf{P},\mathbf{N},\mathbf{O}) > -1 \\ \frac{t-1}{t}\bar{\mathbf{g}}_{(t-1)}, & \text{otherwise.} \end{cases} \tag{4.39}$$

2. Calculate the new weights as follows:

$$\mathbf{w}_{(t)} = \max\left[0, \frac{-\sqrt{t}}{\gamma}\left(\bar{\mathbf{g}}_{(t)} + \mu\right)\right], \tag{4.40}$$

   where γ is the index of the auxiliary strongly convex term (see Section 4.0.2)

I train a number of descriptors in parallel, with a range of values for the hyper-parameters, μ and γ, known to perform well, each for 100 000 iterations, and then select the descriptor which yields optimum performance on the set of positive and negatives, in terms of maximising the area under an ROC curve (ROC-area).

4.1.3.3 Selecting the Threshold

Since the descriptor and distance function differ for each patch, the optimal threshold for classifying a match v. non-match from the resulting distance, will differ. This is resolved by setting the threshold, t'_i, for each patch, i, as

$$t'_i = st_i \tag{4.41}$$

where t_i is the threshold which yields 95% recall (using linear interpolation) for the patch on the training positives and negatives, and s is a global threshold scaling factor which allows the matching performance across all patches, $\mathbf{O}_i \in \mathcal{O}$, to be varied along the receiver operating characteristic (ROC) curve.

4.2 Experiments

I experimented using both patch datasets (to form a benchmark) and image sequences (to demonstrate a performance improvement in an application). I use FPR at 95% recall as the performance metric rather than ROC-area which is used in the previous chapter. This is to follow the approach of Simonyan, Vedaldi and Zisserman (2014), who use FPR at 95% recall to evaluate descriptors, rather than Brown, Hua and Winder (2011), who use area under an ROC curve.

4.2.1 Experiments with Patch Datasets

I compare performance on two datasets: the Learning Local Image Descriptors dataset (Winder, Hua and Brown, 2009) and the DTU Robot dataset (Aanæs, Dahl and Pedersen, 2012) with the following three approaches:

Simonyan (SIM) Descriptors trained according to Simonyan, Vedaldi and Zisserman (2014) and matched using the Euclidean distance and a fixed threshold, as a state-of-the-art baseline

Simonyan with optimal threshold (SIM-OT) The previous, except that I vary the threshold per interest-point (Section 4.1.3.3), to show the contribution from this element alone

PIP I learn weights over the ten base-descriptors, FS0–4 and FSR0–4 (see Table 4.1) using the learning approach outlined in Section 4.1.3

4.2.1.1 Learning Local Image Descriptors Data Set

I tested the per interest-point learning on the *Learning Local Image Descriptors* dataset (Winder, Hua and Brown, 2009), which consists of sets of patches centred on interest-point detections. The global learning for the base-descriptors which PIP and the descriptor for the SIM baseline is carried out on an entire unmodified scene (i.e. *liberty*, *notredame* or *yosemite*) as in previous work (Brown, Hua and Winder, 2011; Simonyan, Vedaldi and Zisserman, 2014). I use the same training and validation set split as in the previous work, which, for each scene, corresponds to 500 000 labelled (matching or non-matching) patch pairs for training, and 100 000 labelled patch pairs for validation.

The patches in the dataset are divided by 3D feature, i.e. corresponding to same feature in the 3D reconstruction, as per the dataset's ground truth. Many features in this dataset have few patches and so the original dataset, though suitable for global training, is not suitable for testing the PIP descriptor. I therefore exclude all features with fewer than eight patches; for each remaining feature, I select only one patch to be the original, $\mathbf{O}$, and one to be a single positive test patch, $\mathbf{P}_{(\text{test})}$; the remaining patches are used as the positive training set, $\mathcal{P}$. To increase the number of patch combinations used for testing, I perform this selection on a exhaustive basis, testing all possible permutations of one original and one test patch for the 3D feature (see Table 4.2 for an example of the permutations). I select an equal number of negative training patches from other features in the dataset and select an additional patch from one of these features to be a negative test patch, $\mathbf{N}_{(\text{test})}$.

The number of unique original/test patches is as follows: 26 896 for *notredame*, 10 991 for *liberty* and 2529 for *yosemite*. The number of exhaustive combinations is much larger, as follows: 202 906 for *notredame*, 82 882 for *liberty* and 19 034 for *yosemite*.

Table 4.3 details the FPR at 95% recall for each approach. In all cases, SIM-OT achieves better results than SIM, showing that allowing the threshold to vary per patch (see Section 4.1.3.3) is beneficial. In addition, PIP yields a further performance improvement in all cases, showing the benefit of learning a combination of the base-descriptors, per interest-point.

Figure 4.5 shows Precision-Recall (PR) curves for each of the global training and per-IP learning combinations. In all cases, PIP outper-

forms SIM and SIM-OT for recall values above 90%. Interestingly, PIP sometimes performs worse than SIM for lower recall values. This may be because the training regime is not optimal for lower values of recall and a different training could be carried out if a better performance at lower recall is desired.

Table 4.4 shows the average sparsity (percentage of non-zero base-descriptor weights) of PIP descriptors across all patches. This amounts to an average of between one-and-a-half and two base-descriptors for each PIP descriptor, effectively around 100 dimensions, showing that the use of RDA results in the selection of a small number of base-descriptors. This makes the approach scalable in terms of memory storage: even allowing for weights, PIP will require, at most, double the storage requirements of SIM. In addition, calculating the distance between two PIP descriptors through linear weighing will take less than double the number of operations, on average, compared to SIM, due to the sparsity induced by RDA.

Figure 4.6 shows the distribution of base descriptors selected. The percentage corresponds to the percentage of patches which have a non-zero weight for each given base-descriptor. (As the average sparsity is 15–20%, the sum for a given combination will be 150–200%.) The figures shows a that the distribution depends on the global training set used. In particular, the distribution is most uniform when globally trained on *notredame*, and least uniform when trained on *yosemite*. While the distribution does not appear to have a great an impact in performance, it highlights the importance of ensuring that the global training data set is suitable for the desired range of target images.

Figure 4.7 and Table 4.5 show the number of the different combinations of base-descriptors with non-zero weights. In the majority of cases, only one or two base-descriptors are used. The most common configuration is a single FSR base-descriptor, followed by one FSR and one FS base-descriptor. The former suggests that, for many interest-points, the centre of the patch does not offer significant discriminative information—the PIP descriptor exploits this by only selecting FSR base descriptors. The latter suggests that many interest-points require two pooling strategies: one for the centre to compensate for misalignment areas, which is offered by one of the FS descriptors; another for locations far from the centre to compensate for differences in viewing angle, offered by one of the FSR descriptors.

Figure 4.8 shows a number of patches from *liberty* categorised by their base-descriptors combination. Only those with at most one FS

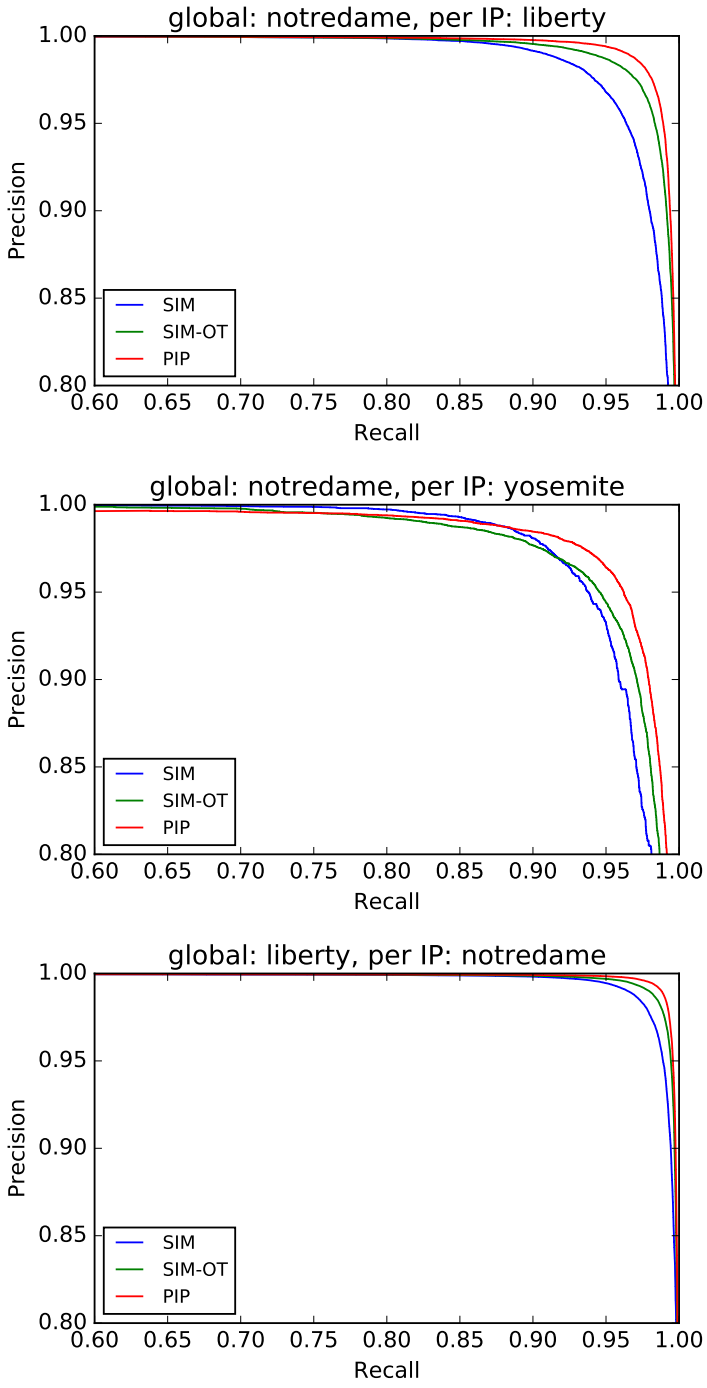

Figure 4.5: PR curves for different global training and per-IP learning combinations

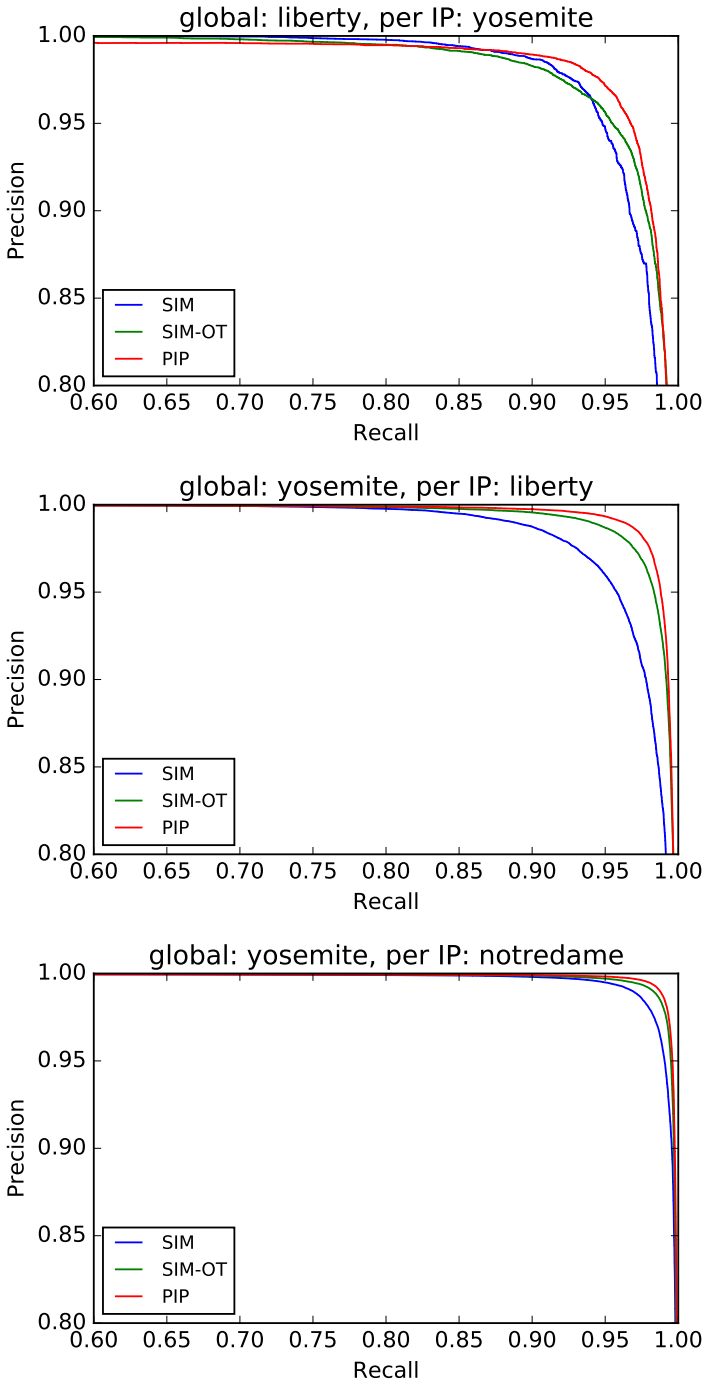

Figure 4.5: PR curves for different global training and per-IP learning combinations

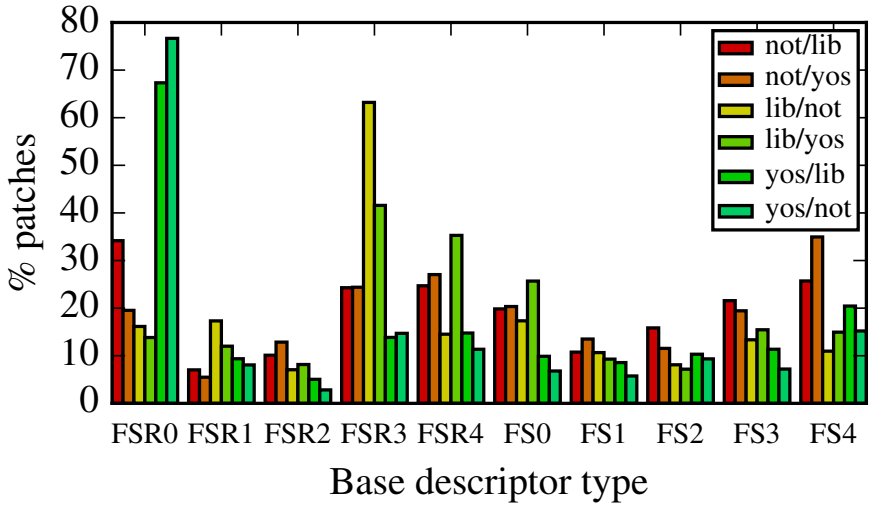

Figure 4.6: The distribution of base-descriptors selected across patches key: global training dataset / per interest-point learning and testing dataset

and at most one FSR base-descriptor are shown, to allow this information to be displayed on a 2D grid—this corresponds to the grey cells in Table 4.5. Where possible, I show multiple patches from the same 3D feature, to give a better idea of the variation between patches which led to the base-descriptor selection.

The weighting and selection of base-descriptors per interest-point is a function of both the positive and negative patches in the training so one would not expect to see absolute trends in Figure 4.8. However, I observe the following:

1. Regions not matched with an FS base-descriptor appear to suffer from centre misalignment between regions.

2. Regions matched with the FS0 base-descriptor tend to have less centre misalignment than those matched with the FS4 base-descriptor.

3. Regions with no FSR base-descriptor appear to suffer from occlusion in around the outside of the regions or a large variation in viewing angle between regions.

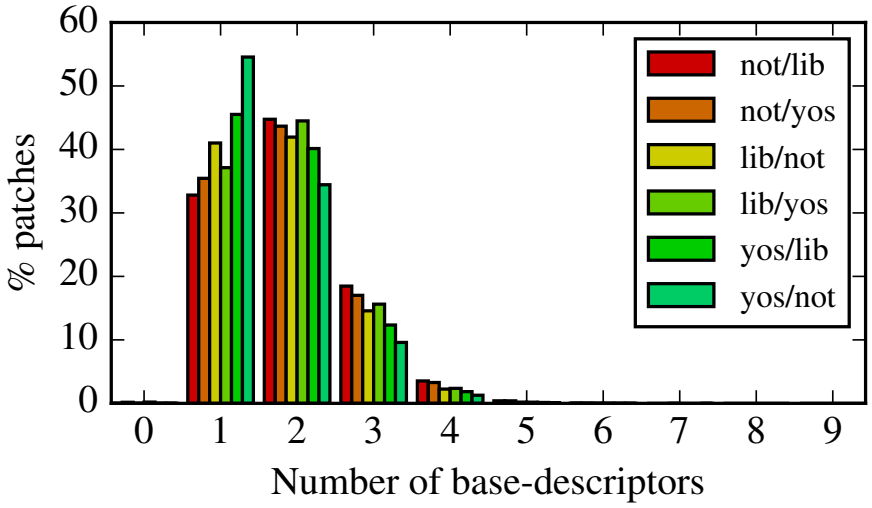

Figure 4.7: The distribution of patches categorised by the number of base-descriptors selected; key: global training dataset / per interest-point learning and testing dataset

4. Regions with the FSR4 base-descriptor tend to show a larger variation in viewing angle than those in FSR1.

Using RDA, a very small fraction of patches are matched using no base-descriptors and other patches are always classified as negative matching, e.g. the 23 patches in the top-left cell of Table 4.5. One solution would be to avoid matching any patches or associated interest-point for which this occurs.

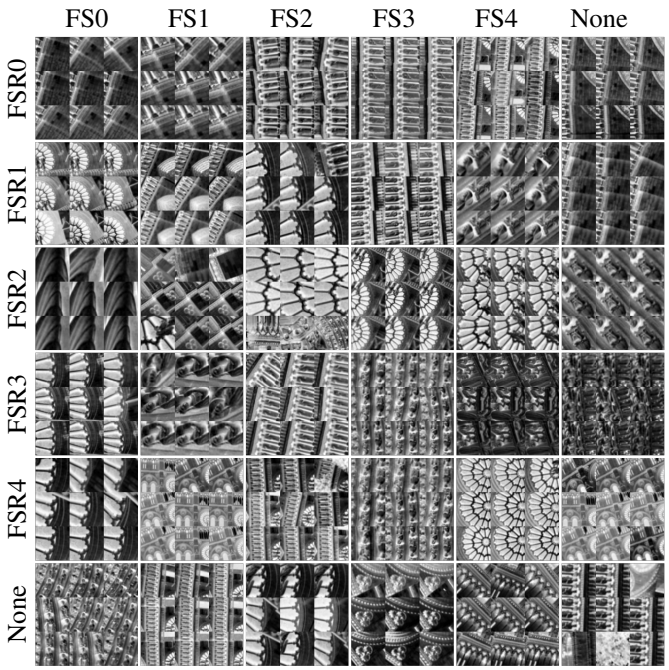

Figure 4.8: Patches from *liberty* categorised by the constituent base-descriptors

orig	test	+	+	+	+	+	+
orig	+	test	+	+	+	+	+
orig	+	+	test	+	+	+	+
orig	+	+	+	test	+	+	+
orig	+	+	+	+	test	+	+
orig	+	+	+	+	+	test	+
orig	+	+	+	+	+	+	test
test	orig	+	+	+	+	+	+
+	orig	test	+	+	+	+	+
+	orig	+	test	+	+	+	+
+	orig	+	+	test	+	+	+
+	orig	+	+	+	test	+	+
+	orig	+	+	+	+	test	+
+	orig	+	+	+	+	+	test
...	...	...	...	...	...	...	...
test	+	+	+	+	+	+	orig
+	test	+	+	+	+	+	orig
+	+	test	+	+	+	+	orig
+	+	+	test	+	+	+	orig
+	+	+	+	test	+	+	orig
+	+	+	+	+	test	+	orig
+	+	+	+	+	+	test	orig

Table 4.2: To increase the number of patch combinations used for testing PIP, I select patches to be original and test patches on an exhaustive basis. Each row corresponds to one combination of original patch (orig), test patch (test) and positive training patches (+).

scene		descriptors		
global training	**per IP learning**	SIM	SIM-OT	PIP
notredame	liberty	3.16%	1.26%	**0.58%**
notredame	yosemite	6.87%	5.63%	**3.54%**
liberty	notredame	0.53%	0.29%	**0.15%**
liberty	yosemite	5.28%	4.36%	**2.77%**
yosemite	liberty	3.97%	1.26%	**0.64%**
yosemite	notredame	0.49%	0.28%	**0.16%**

Table 4.3: The FPR at 95% recall achieved on each combination of datasets: PIP descriptors yield the lowest error in each case

global training	**per IP learning**	**sparsity**
notredame	liberty	19.40%
notredame	yosemite	18.91%
liberty	notredame	17.87%
liberty	yosemite	18.34%
yosemite	liberty	17.09%
yosemite	notredame	15.79%

Table 4.4: The average sparsity (percentage of non-zero weights over base-descriptors) across all patches

	number of FS base-descriptors					
num. FSR	**0**	**1**	**2**	**3**	**4**	**5**
0	23	17368	8796	932	36	1
1	61317	52711	10562	717	27	0
2	29440	13781	1796	99	0	0
3	3791	1191	112	6	0	0
4	166	31	2	0	0	0
5	0	1	0	0	0	0

Table 4.5: The number of patches in *notredame* categorised by the number of base-descriptors (globally trained on liberty) selected (non-zero weighted), split between the FS and FSR base-descriptor sets

4.2.1.2 DTU Robot Image Point Feature Data Set

I also tested PIP using a patch dataset formed of patches from interest-points from scenes of the DTU Robot Image dataset (Aanæs, Dahl and Pedersen, 2012), which was originally created for comparing interest-point extractors. This dataset contains a number of model scenes which are photographed from a range of viewpoints and under different lighting conditions, as shown in Figure 4.9. The authors use a robot and structured light scanner to ensure a precise 3D reconstruction. This set-up is similar to the previous experiment but with a larger number of patches for each set of matching interest-points. I also believe that this dataset is more challenging than the previous, e.g. due to the larger range of lighting variations.

The dataset does not contain a list of corresponding interest-points, or associated patches. I therefore use the dataset to obtain sets of patches. For every model scene, I selected a single image to be the key frame and extract all Scale-Invariant Feature Transform (SIFT) interest-points (Lowe, 2004). For every interest-point in this scene, I carry out the following:

1. I use the 3D data to transform the interest-point into every other image in the model scene.

2. I find any interest-points which match the criteria used by Winder, Hua and Brown (2009) and as used in the previous chapter (see Section 3.1.1).

3. I discard any interest-points whose support region extends beyond the border of the image.

4. Once a total of forty-nine suitable interest-points have been found, I extract patches from the support regions for these and the original interest-point. This results in a set of fifty patches: see Figure 4.10 for examples.

I use the following method to order to transform an interest-point from one image to another:

1. I obtain a rectangular grid of eight by eight pixels from the first interest-point footprint (this grid occupies a quarter of the overall support region size)

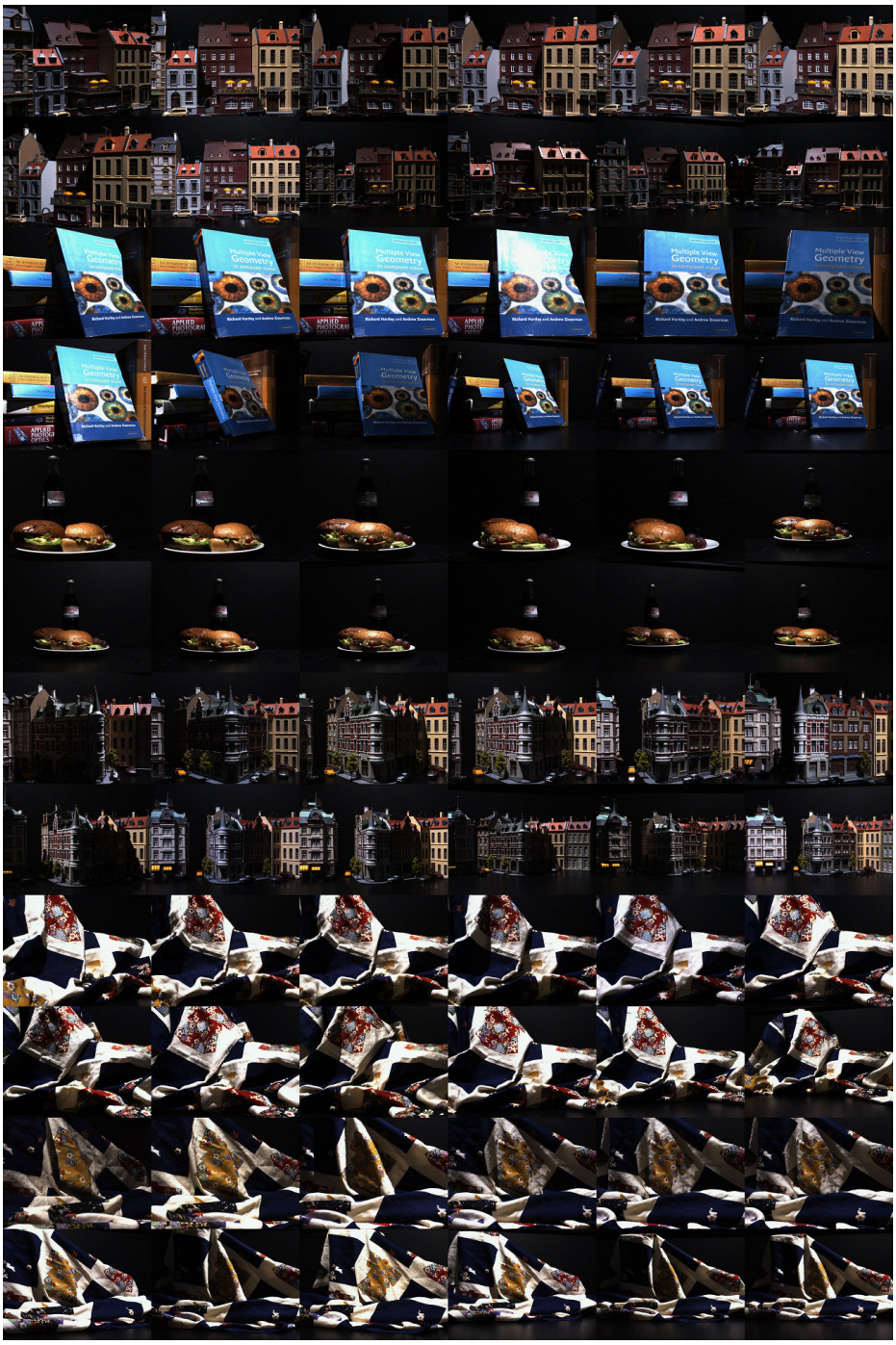

Figure 4.9: Images from different scenes in the DTU Robot Image Dataset: note the variation in viewing angle and lighting conditions

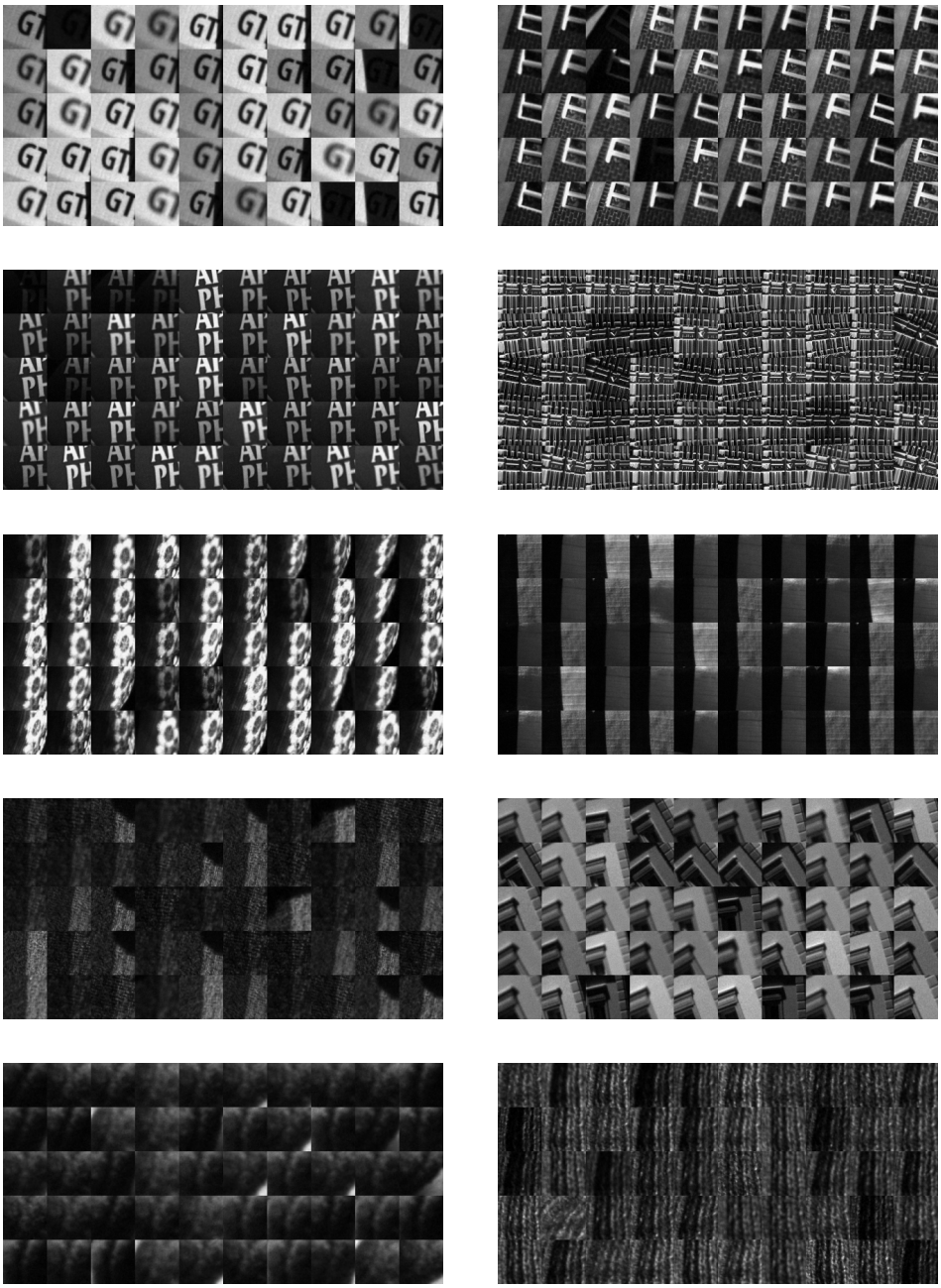

Figure 4.10: Sets of patches corresponding to matching interest-points between different images in the DTU dataset; note the differences in position and lighting conditions

scenes		Patches	
global training	**test**	**per interest-point training**	**test**
1–6	7–12	2–25	26–50
7–12	1–6	2–25	26–50

Table 4.6: The division of test and training sets, both for the global learning and per interest-point learning

2. I use the provided 3D reconstruction and camera parameters to transform these pixels to the second image, through ray-casting and projection.

3. I calculate the best similarity transform between these pixel pairs, using the method of least squares, and use this to transform the interest-point.

The twelve scenes are split into two configurations for test and training as shown on the left of Table 4.6. The global training occurs using 500 000 labelled (matching or non-matching) patch pairs for training, and 100 000 labelled patch pairs for validation, which are randomly selected from the six scenes for each of the two scene splits. This is for compatibility with the configuration of the *Learning Local Image Descriptors* dataset.

In addition, the sets of patches are split into test and training positives as shown on the right of Table 4.6. The per interest-point training set is used for per interest-point training (PIP) and threshold selection (PIP and SIM-OT); it is not used for SIM. An equal number of negatives are selected randomly from the per interest-point training sets and test sets of other patches. The total number of unique test patches for each scene can be found in Table 4.7, which is 25 times the number of matching interest-point.

Table 4.7 details the FPR at 95% recall for each approach. In nine out of twelve cases, PIP outperforms SIM. In the three cases where PIP performed poorly, the FPR for SIM was below 3%; I believe that the learning algorithm is attempting to optimise the base-descriptor for a small set of difficult features and adversely affecting performance on the rest. SIM-OT outperformed SIM in only seven out of the twelve cases, showing that varying the threshold alone is not sufficient

		descriptors		
scene	**test patches**	SIM	SIM-OT	PIP
1	25 300	8.05%	7.00%	**5.73%**
2	52 650	**0.72%**	1.46%	1.26%
3	9600	24.89%	17.77%	**15.41%**
4	3650	10.19%	9.01%	**8.30%**
5	16 600	7.49%	6.11%	**4.89%**
6	30 700	6.90%	7.20%	**5.56%**
7	21 325	32.74%	18.06%	**15.05%**
8	19 200	3.14%	3.54%	**2.77%**
9	28 700	17.90%	14.82%	**11.13%**
10	36 225	26.65%	14.90%	**12.01%**
11	29 500	**0.70%**	1.19%	0.96%
12	28 675	**2.57%**	3.64%	2.71%

Table 4.7: The FPR at 95% recall achieved on the DTU dataset; the best is in bold.

for better performance on this dataset. All descriptors show a large variation in their performance between difference scenes: I believe that this is because some scenes are more subject to the effects of camera translation and rotation or changes in lighting conditions than others.

4.2.2 Experiments with Photo Collections

The previous experiments tested the performance of the PIP on patch datasets, but did not demonstrate an application for improved interest-point matching. I therefore test the use of PIP for image matching/search using the Oxford Buildings dataset (Philbin et al., 2007). The dataset consists of 5062 photos of Oxford buildings, obtained from Flickr, divided into scenes based on landmark and view.

I consider a scenario where N images of a scene have already been matched, and it is desirable to match a new (query) image, I_Q, to a target image, I_T, in this scene. Initially, N starts at one, and matching begins using SIM alone, with matches verified using random sample consensus (RANSAC). As N grows, more interest-point matches become available to train PIP descriptors for interest-points in I_T. Typ-

ically N grows to around 200 once all images have been processed for a given landmark. In all cases, I extract interest-point using SIFT's interest-points extractor (Lowe, 2004) and use inlier and incorrect (outlier) matches of successful homographies (those obtained from at least sixteen RANSAC inliers) as positive and negative exemplars.

I repeat for 4280 pairs of target and query images, $\{I_T, I_Q\}$; in total there are 55 unique query images. For every pair, I match with two approaches:

SIM SIM only, as a baseline

SIM + PIP the union of matches from both descriptors

Ideally, one would like to use PIP alone, however using PIP requires that each interest-point has sufficient matches to learn an effective PIP descriptor. In the experiments, N is not sufficiently large for this to be the case as many interest-points appear sporadically between images.

A successful match of the query image is defined by having at least sixteen RANSAC inliers. Table 4.8 shows the percentage of successful matches for each approach. In every case, including matches yielded from PIP descriptors yielded an improved result over the use of SIM alone. Figure 4.11 shows the success rate per interest-point (probability of the feature being a RANSAC inlier) as a function of the number of training exemplars (matched interest-point in other images). The performance of PIP increases with the number of training exemplars available, being at par with SIM descriptors alone for two exemplars, and increasing to 20% more matches at ten training exemplars. Note that the performance of SIM also improves (to a lesser extent) for interest-point with many matches, perhaps due to these features being *a priori* more visually distinctive or reliable.

In addition to the numerical results, I demonstrate matching between individual image pairs using figures showing the matches using the colour code in Table 4.9. Blue matches were inliers in both cases and must have been a closest match using SIM (they may or may not be using PIP). Red matches must represent a failure case of SIM + PIP: even though it was closest match using SIM, it was lost with the introduction of PIP matches. Green, yellow and orange matches are those which were an inlier only when SIM + PIP was used. Green and orange matches were also matched using SIM, however required

	success rate	
global training	SIM	SIM + PIP
notredame	21.14%	22.41%
liberty	21.52%	22.34%
yosemite	21.54%	23.06%

Table 4.8: The percentage of query images correctly matched to the original; Using PIP in addition to SIM improves performance

the extra corroboration of the PIP matches to be a RANSAC inlier. Yellow matches were matched by PIP alone.

Figure 4.12 shows a case in which SIM + PIP succeeded, but SIM alone did not: the successful set of inliers included those from PIP alone (yellow), those common to both PIP and SIM (orange) and those from SIM alone (green). With SIM alone, only the blue matches were obtained by RANSAC. This shows that PIP achieved better results partly by finding additional matches and partly by strengthening existing SIM matches during RANSAC.

There are occasional failure cases where PIP adds no new correct matches. This effectively adds noise to the matches, reducing the inlier probability and causing occasional RANSAC failures due to the non-deterministic nature of the algorithm. Figure 4.13 shows a failure case. SIM alone found a large enough set of inliers (red). Despite these SIM matches being retained with the addition of matches from PIP fewer than sixteen inliers were obtained by RANSAC on the combination of matches. The non-deterministic nature of RANSAC means that the inclusion of PIP does not necessarily improve image matching performance in every case (due to possible selection of incorrect sample sets). However, Table 4.8 and Figure 4.11 clearly show that, on average, PIP performs better at image matching and offers improved performance per interest-point in terms of match probability.

Figure 4.14 shows other cases in which SIM + PIP succeeded, but SIM alone did not. Figure 4.15 shows other cases in which SIM alone succeeded but SIM + PIP failed to obtain a homography with RANSAC. Finally, Figure 4.16 shows cases in which neither approach succeeded.

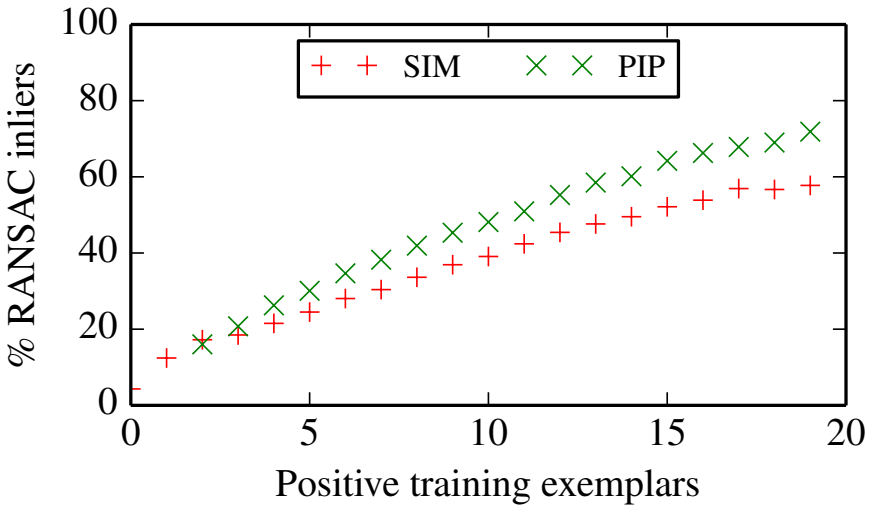

Figure 4.11: The percentage of RANSAC inliers for SIM and PIP in query images against the number of positive exemplars for each interest point found prior to PIP training

4.3 Conclusion

In this chapter, I presented a new approach for learning an interest-point specific descriptor (PIP) by learning weights over a number of base-descriptors. I globally train each base-descriptor over a large set of image regions, building on the descriptor of Simonyan, Vedaldi and Zisserman (2014) (SIM), and then select and learn weights over these base-descriptors to achieve an optimal descriptor for each interest-point. The approach outperforms the use of a single globally-learnt descriptor on the Learning Local Image Descriptors and DTU Robot datasets, even when a small number of patches are provided for training each interest-point. I also demonstrated the suitability of the approach for matching in photo collections. The memory requirement of the PIP descriptor is only slightly more than double for SIM making the approach scalable.

Although the PIP descriptor, on average, outperforms SIM, it led to worse performance on a quarter of the scenes in the DTU dataset. I believe that this is because the descriptor design is built upon wrong set of assumptions, which prevents universal improvement, as follows:

	RANSAC inlier		
Colour	**SIM**	**SIM + PIP**	**Source**
blue	yes	yes	doesn't matter
red	yes	no	doesn't matter
green	no	yes	SIM only
yellow	no	yes	PIP only
orange	no	yes	SIM and PIP

Table 4.9: The colour code used for image pair matches

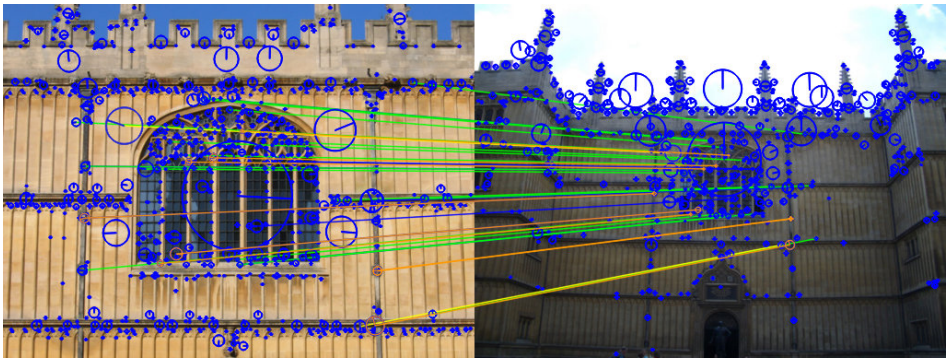

Figure 4.12: A success case using SIM + PIP

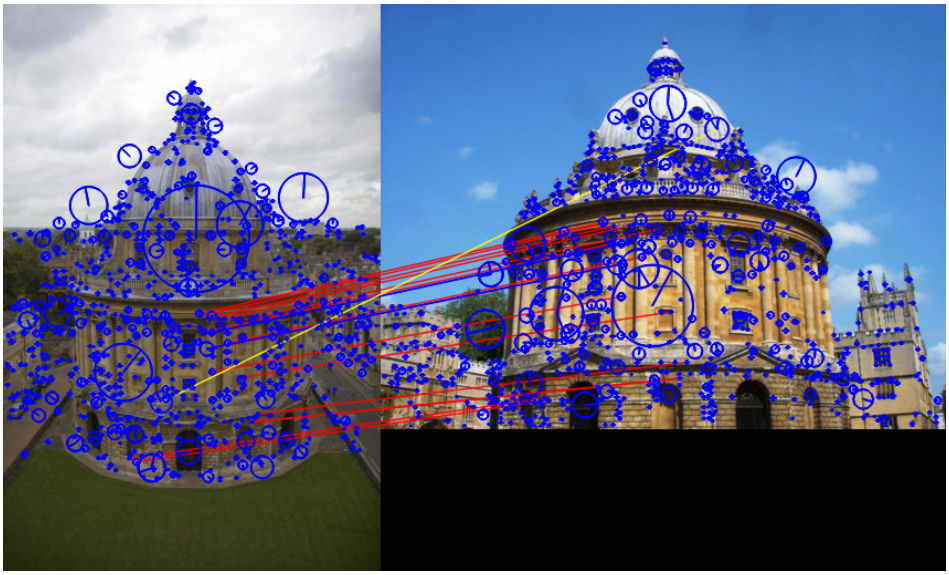

Figure 4.13: A failure case using SIM + PIP

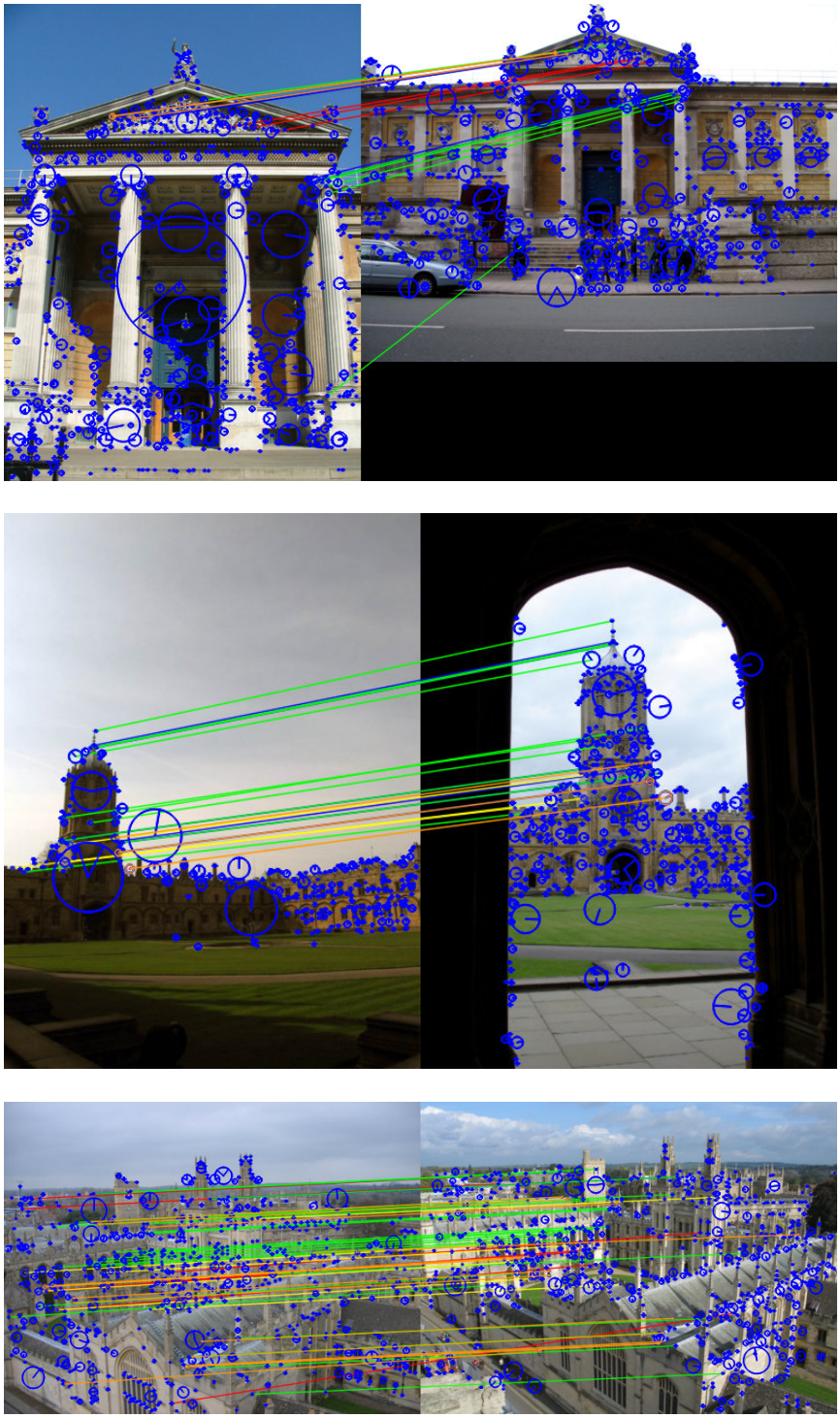

Figure 4.14: Other success cases with SIM + PIP

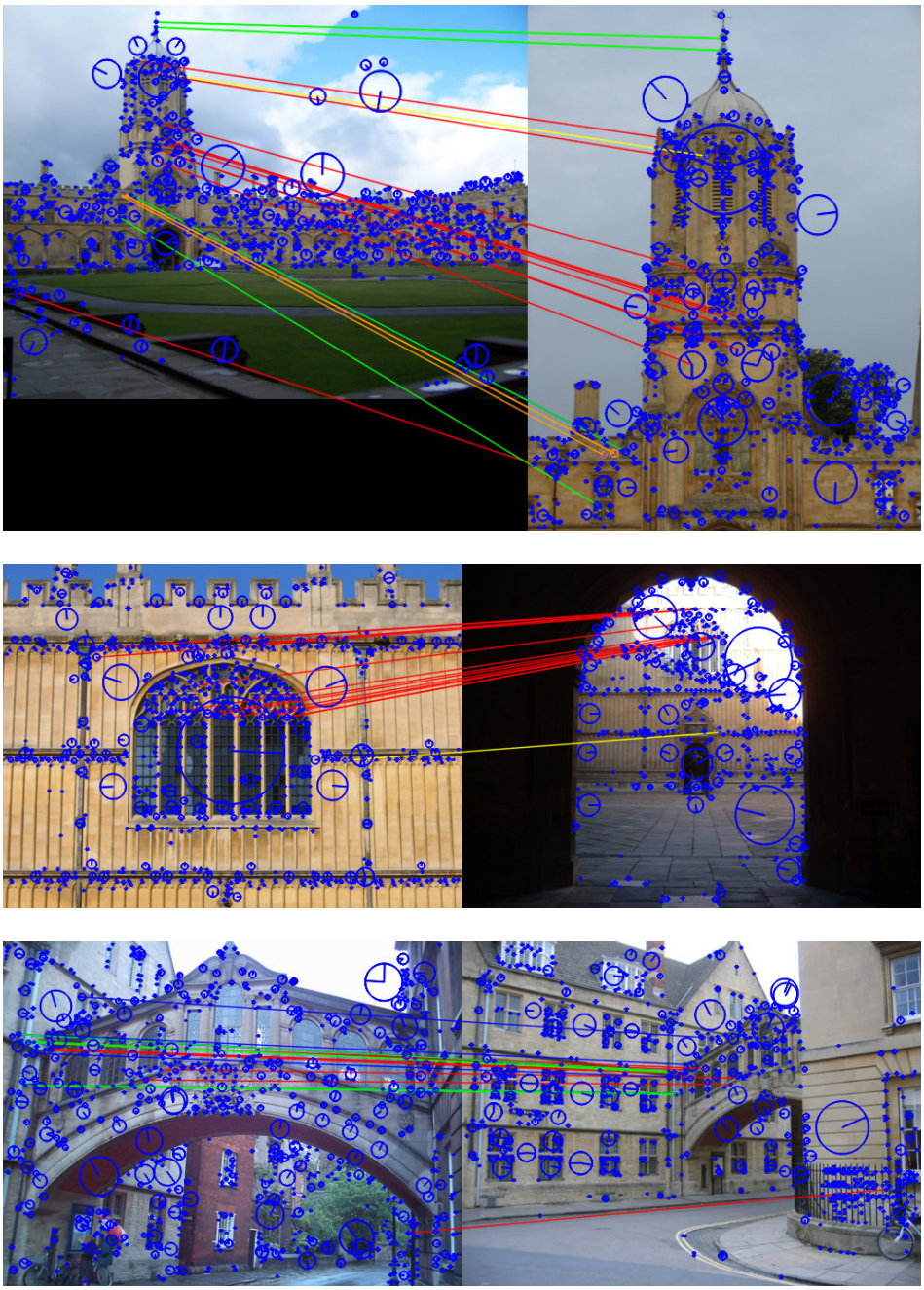

Figure 4.15: Other failure cases with SIM + PIP

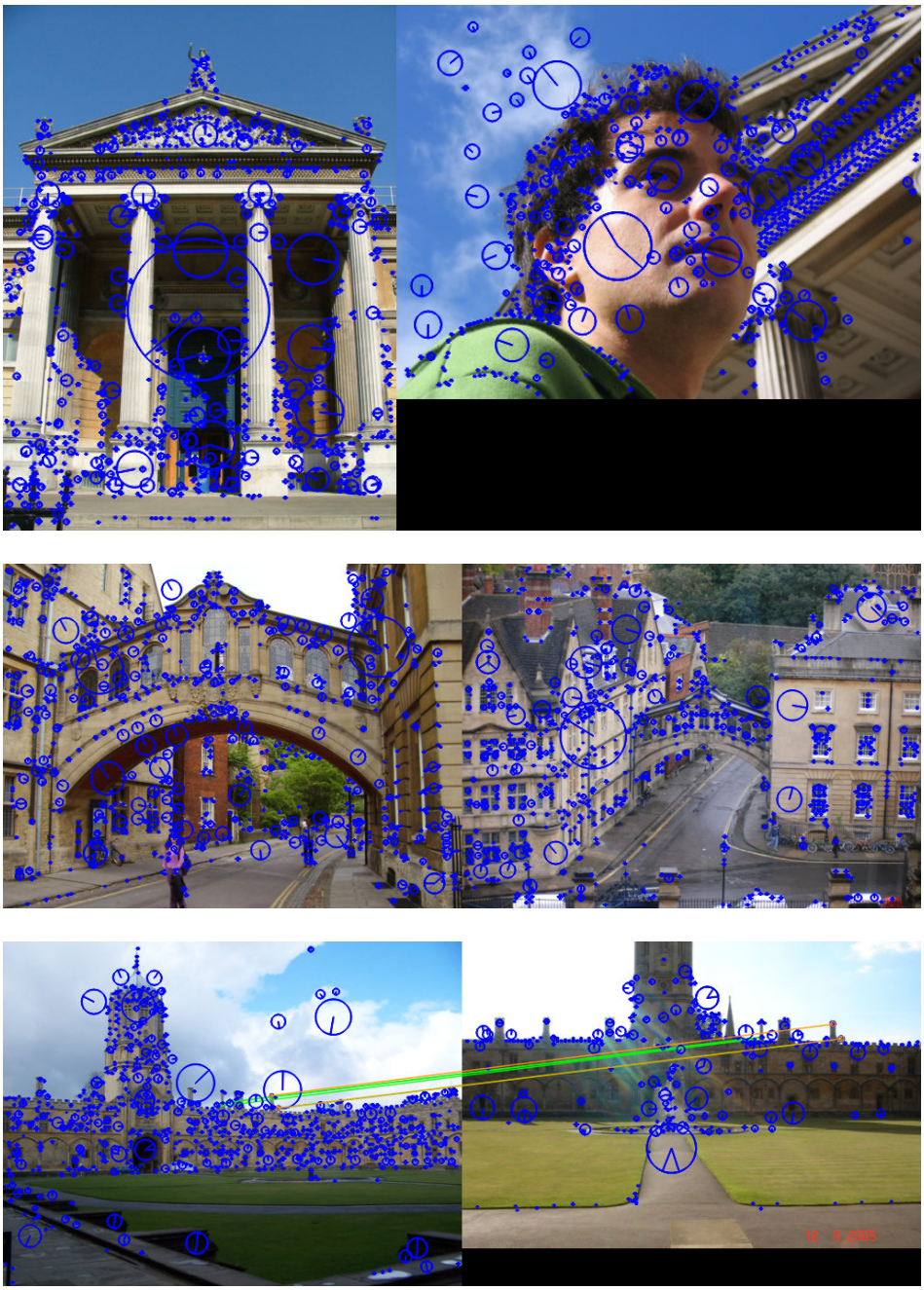

Figure 4.16: Cases in which neither SIM nor SIM + PIP succeeded

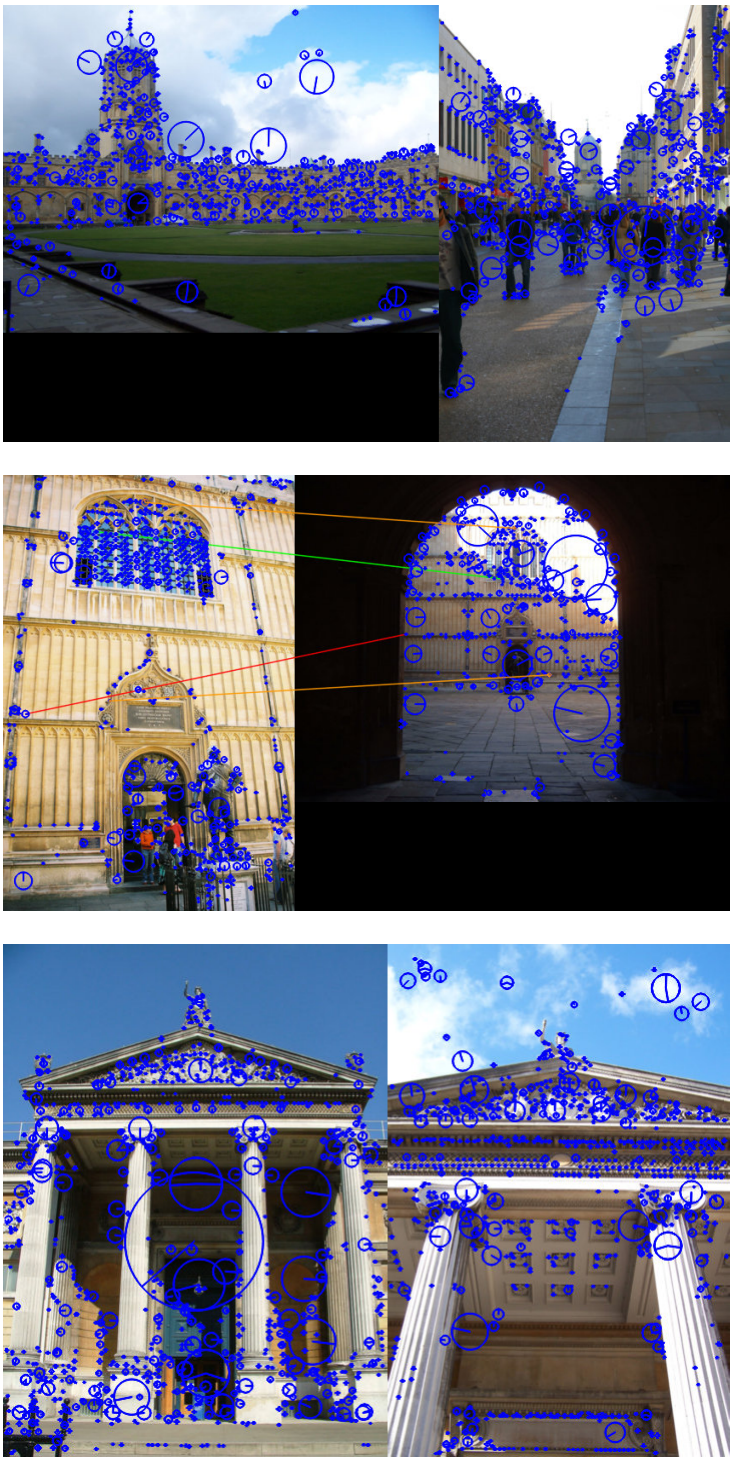

Figure 4.16: Cases in which neither SIM nor SIM + PIP succeeded

1. Oriented gradients, eight in total, are the right features.
2. All pooling regions should be Gaussian.
3. Pooling regions should lie over circular rings centred on the centre of the support region.
4. Weights should be shared over symmetric rings of four or eight regions.

I believe that the use of pooled features at such a low level limits the performance of either SIM or PIP in accordance with my proposition:

> The invariance of state-of-the-art algorithms is at too low a level for generalised interest-point matching and generalised object detection.

If there was invariance at a higher level, i.e. involving semantics, the matching could be universally better. One example is patches containing letters. As a human, it is easy to identify and match the letters between patches even where there is a large amount of variation in viewing conditions. Having confirmed that two letters match, it is then possible to determine whether the letters could be from the same 3D object, e.g. based on the colour or font style. On the other hand, the SIM and PIP descriptors simply lift, bin and pool low level Histograms of Oriented Gradients (HOG) features, and are unable to perform any semantic analysis.

Another example for which higher level invariance would result in better performance is in patches containing part of a recognisable object, such as a window or column. There are objects such as these present in the image-pairs in which both SIM and PIP failed, e.g. those in Figure 4.16. A human is unlikely to accidentally confuse a patch with a window with one not containing a window (though two similar but different windows might be confused) whereas the PIP descriptor is prone to such false positives. Moreover, once a human has matched based semantic objects, he or she can analyse the scene to determine whether the scenes match e.g. determining whether the windows or columns are consistent in spite of adverse such changes such as lighting conditions, partial occlusion or poor camera focus. This requires both a high level semantic understanding and the ability to then determine whether or not the lower level detail is consistent with the same object instance under different viewing conditions.

In addition, the PIP descriptor has only been shown to work on natural images (photos), and even then imperfectly. An even greater issue would occur when trying to match between photos and other depictive styles (see Section 1.3), e.g. matching between photos and non-natural images (artwork) of the Eiffel tower.

5 Detecting People in Artwork with CNNs

Whereas the previous two chapters involved interest-point matching, this chapter involves the task of object detection. Object detection has improved significantly in recent years, especially as a result of the resurgence of convolutional neural networks (CNNs) and the increase in performance and memory of graphics processing units (GPUs) (see Section 2.2.2.2). However, in spite of the successes in photo-based recognition and detection, research into recognition within styles of images other than natural images (photos) remains limited (Hall et al., 2015; Westlake, Cai and Hall, 2016).

This is an instance of the *cross-depiction problem*: detecting objects regardless of how they are depicted (photographed, painted, drawn, etc.). The motivation for considering the cross-depiction problem is two fold (Hall et al., 2015):

foundational Because we, as human observer's, are able to recognise objects across a plethora of depictive styles, there must be something fundamental about the object which is invariant across depictions. Analysis into the cross-depiction problem forces one to test all assumptions about object detection, many of which are premised on photos alone, including the very foundations of Computer Vision.

practical Since the world contains images in a plethora of depiction styles, any Computer Vision task which must operate on all images must overcome the cross-depiction problem. An example is image search: if a method relies on the assumption that all images to be photos or photo-like non-natural images (artwork), it is unlikely to generalise across all depictive styles. Therefore, its results will either be limited to photos and similar artwork or, worse, include false positives results other depictive styles.

This chapter focuses on one class: people. As Westlake, Cai and

Figure 5.1: Detecting people across different depictive styles is a challenge: here are some successful detections

Hall (2016) observe, people occur far more frequently than other classes across the spectrum of depictive styles perhaps, for at least two reasons:

1. As human observer's, we are interested in other humans and this is reflected in the choice of photos and artwork.

2. Many classes only existed after a particular time period, e.g. cars, computers and aeroplanes.

As a result, focusing on people increase the range of depictive styles represented in artwork and may even provide the most difficult object to detect across different depictions.

5.0.1 Limitations of Earlier Work

This chapter builds upon the limitations of earlier work addressing the cross-depiction problem, which are set out here. These limitations include both the methods and the datasets used.

Algorithm	Precision	Recall	F-measure
Human	80 %	86 %	0.829
DPM (Felzenszwalb, Girshick et al., 2010)	44 %	46 %	0.458
Poselets (Bourdev et al., 2010)	31 %	24 %	0.271
RCNN (Girshick et al., 2014)	32 %	18 %	0.226
HOG (Dalal and Triggs, 2005)	3 %	49 %	0.051

Table 5.1: Performance of computer algorithms against humans on the Picasso dataset: the figures are based on the point on the precision-recall curve which maximises the F-measure

5.0.1.1 Detecting People In Cubist Art

Ginosar et al. (2014) identify that evaluations of Computer Vision algorithms are often limited to photos, which means the algorithm only needs to be robust to natural poses. Humans, on the other hand, are able to recognise people in abstract representations, such as in the Cubist paintings of Picasso, even without prior training. The authors created a dataset using such paintings (the Picasso Dataset) to determine whether test object detection algorithms are also able to recognise people in such paintings.

One interesting result was that there was disagreement between different human observers as to whether or not a person was present and if so, what the bounding box for the person was. Nevertheless, there was, on the whole, a consensus with 80 % precision and 86 % recall. By taking the median location of the bounding box corners as labelled by humans, the authors were able to produce a "ground truth" for training and evaluating Computer Vision algorithms.

The results (Table 5.1) showed that all Computer Vision algorithms perform worse than humans. One limitation of this study is that the algorithms were trained on photos of people rather than artwork containing people. As a result, they almost certainly overfitted. However, as the authors point out, humans are able to identify people in Cubist art without prior training and so it is not unreasonable to expect or to seek this ability in Computer Vision. Another is that the dataset only contains one depictive style, and indeed one artist: Picasso. The best performing algorithm was DPM, suggesting that the use of a parts allows better generalisation.

5.0.2 In Search of Art

Crowley and Zisserman (2014) evaluate the performance of CNNs learnt on photos for classifying objects in oil paintings, showing strong performance in spite of the difference in domain. The CNNs used for classification allow paintings to be searched for based on the object they contain.

The authors evaluate the algorithms using the BBC "Your Paintings" dataset which contains 210 000 oil paintings. A subset of these images have been tagged with the objects they contain, and so are used for classification. One limitation, however, is that many paintings include people without being tagged as such. As a result, the author's evaluation excludes people as a class.

To allow the CNNs to be used to search for objects which are not part of the (off-line) training process, the authors download a number of images from Google to form positive training examples, and use a set of other images as negative training examples. Next, they train a Linear-SVM classifier using the output of the penultimate layer of the CNN. One interesting result is that this approach performs best when using only photos from Google and not clip-art. This appears to be because the paintings tend to be closer to photographs than clip-art, suggesting the dataset is limited to photo-like artwork.

5.0.3 Learning Graphs to Model Visual Objects Across Different Depictive Styles

Wu, Cai and Hall (2014) improved DPM (Felzenszwalb, Girshick et al., 2010) to perform cross-depiction matching across different depictive styles. Instead of using root and part-based filters and a latent-SVM (LSVM), the authors learn a fully connected and bidirected graph to better model object structure between depictions, using the structural SVM (SSVM) formulation of M. Cho, Alahari and Ponce (2013). Moreover, instead of single set of attributes (weights and biases over HOG features) for each node, the authors learn two sets of attributes: e.g. one for photos and the other for artwork though the set assignment occurs at train-time in an unsupervised manner through k-means clustering. At test-time, the algorithm calculates the score of a detection, based on the location of all the nodes, by summing the similarity score of the nodes and edges against the model. The similarity score for the each node is the maximum score across each at-

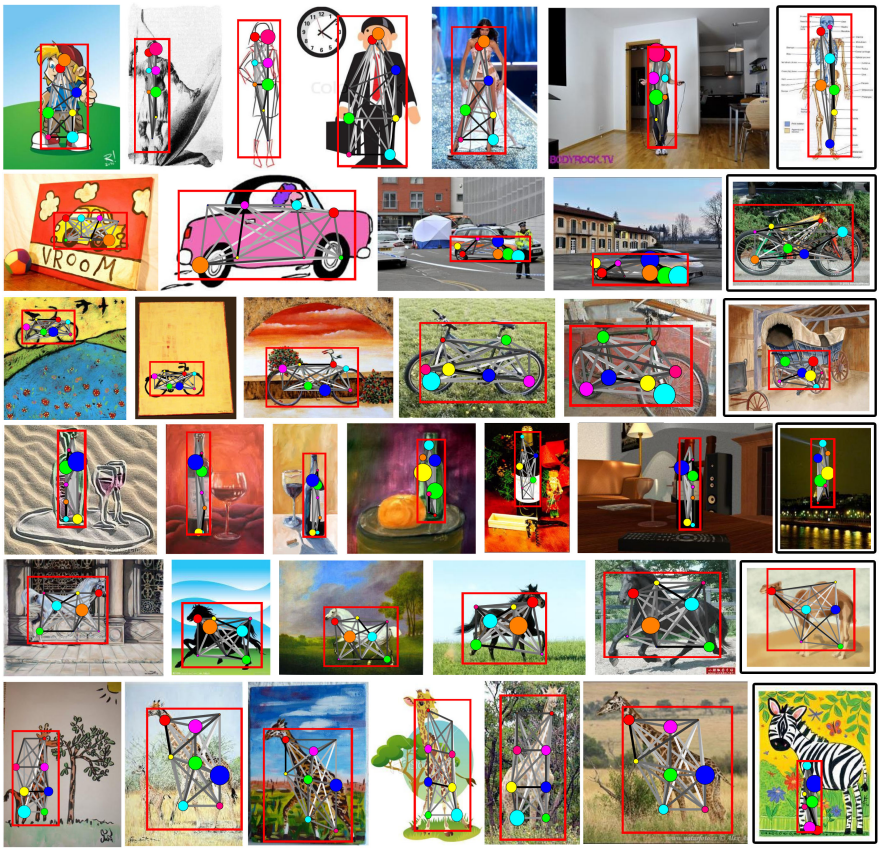

Figure 5.2: Detections using Wu, Cai and Hall (2014)'s algorithm, showing the fully connected graph. The size of the nodes and darkness of the edges indicates the relative contribution towards the detection score.

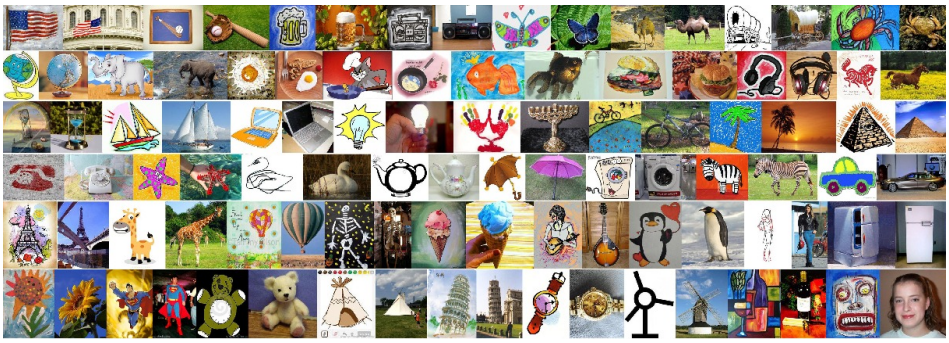

Figure 5.3: Images from Wu, Cai and Hall (2014)'s Photo-Art-50: there is one artwork image and one photo for each of the fifty categories

tribute set. The edge parameters are shared between the sets and the features used are as in M. Cho, Alahari and Ponce (2013). Figure 5.2 shows examples of detections with the relevant graph.

One limitation of the model is the need to cluster the images by depiction style (only two are used by authors) which means that generalisation across photos and artwork effectively occurs by learning one model per depiction style. This does not apply to the edges, which are shared between depiction styles, and the use of a fully connected graph rather than a "hub-and-spoke" model may improve generalisation across depictive styles.

Another limitation is the use of DPM to initialise the parts and to bootstrap the model. This means it suffers from the same limitation as DPM, the heuristic used to select parts is not ideal and may not be reliable. This can be observed in Figure 5.2: in particular, note the seven parts of a bottle and the preference for parts to lie on the torso of a human, perhaps due to its consistency, over parts such as the hands and elbows.

Wu, Cai and Hall (2014) introduce a new dataset, *Photo-art-50*, to test their model. This comprises of photos and artwork, with ground truth bounding boxes, for fifty object classes. Figure 5.3 shows some examples. Although the dataset includes photos, paintings, drawings and cartoons, the number of depictive styles represented remains small compared to the many possible in the world. In addition, the images tends to show objects in their entirety. As most or all parts are visible, this makes the images suitable for a part-based model. However, it does not represent the true challenge of the cross-depiction problem.

5.0.4 The People-Art Dataset

The experiments in this chapter involve a more recent dataset, *People-Art* (Westlake, Cai and Hall, 2016). The dataset can be found online at https://github.com/BathVisArtData/PeopleArt. Hall et al. (2015) used an earlier version of the dataset in their work.

The dataset contains images from 41 different artwork movements, increasing the number of depictive styles compared to the *Photo-Art-50* dataset. It also contains photos and cartoons making a total of 43 depictive styles. Whereas, that dataset had fifty classes, the People-Art dataset has one single class, people, focusing on a diversity of depictive styles rather than classes. This dataset presents a challenge because of the sheer diversity in the way artists have depicted humans,

and overcomes the limited range of depictive styles in earlier datasets. Prior to the experiments in this chapter, the best performance on the dataset was 45% average precision (AP), from a CNN that was neither trained nor fine-tuned for this task.

The images in the People-Art dataset come from the following sources:

photos PASCAL Visual Object Classes (VOC) 2012 (Everingham, Van Gool et al., 2012)

cartoons Google searches

other images *WikiArt.org*

Figure 5.4 shows one image for every depictive style represented in the *People-Art* dataset. In total, there are 4778 images, of which 1644 images contain one or more people.

The 41 depictive styles from *WikiArt.org* are categorised based on art movements, e.g. Art Deco, Cubism, impressionism and photorealism. These depictive styles cover the full range of projective and denotational styles, as defined by Willats (1997). In addition, Westlake, Cai and Hall (2016) propose that these styles cover many poses, a factor which Willats did not consider. They identify the following challenges presented by the dataset.

range of denotational styles The denotational style is the style with which primitive marks in the image are made (brush strokes, pencil lines, etc.) (Willats, 1997), with photos of natural scenes rather than artwork being a denotational style in its own right. Figure 5.5 shows a range of images with different denotational styles.

range of projective style The projective style includes linear camera projection, orthogonal projection, inverse perspective, and a range of ad-hoc projections (Willats, 1997). An extreme protective styles is shown in Cubism, in which it is common for an object to drawn or painted as if seen from many different viewpoints into the 2D canvas (Ginosar et al., 2014). Figure 5.6 shows a range of images with different projective styles.

range of poses Though pose is handled by previous Computer Vision algorithms (Felzenszwalb, Girshick et al., 2010), Westlake,

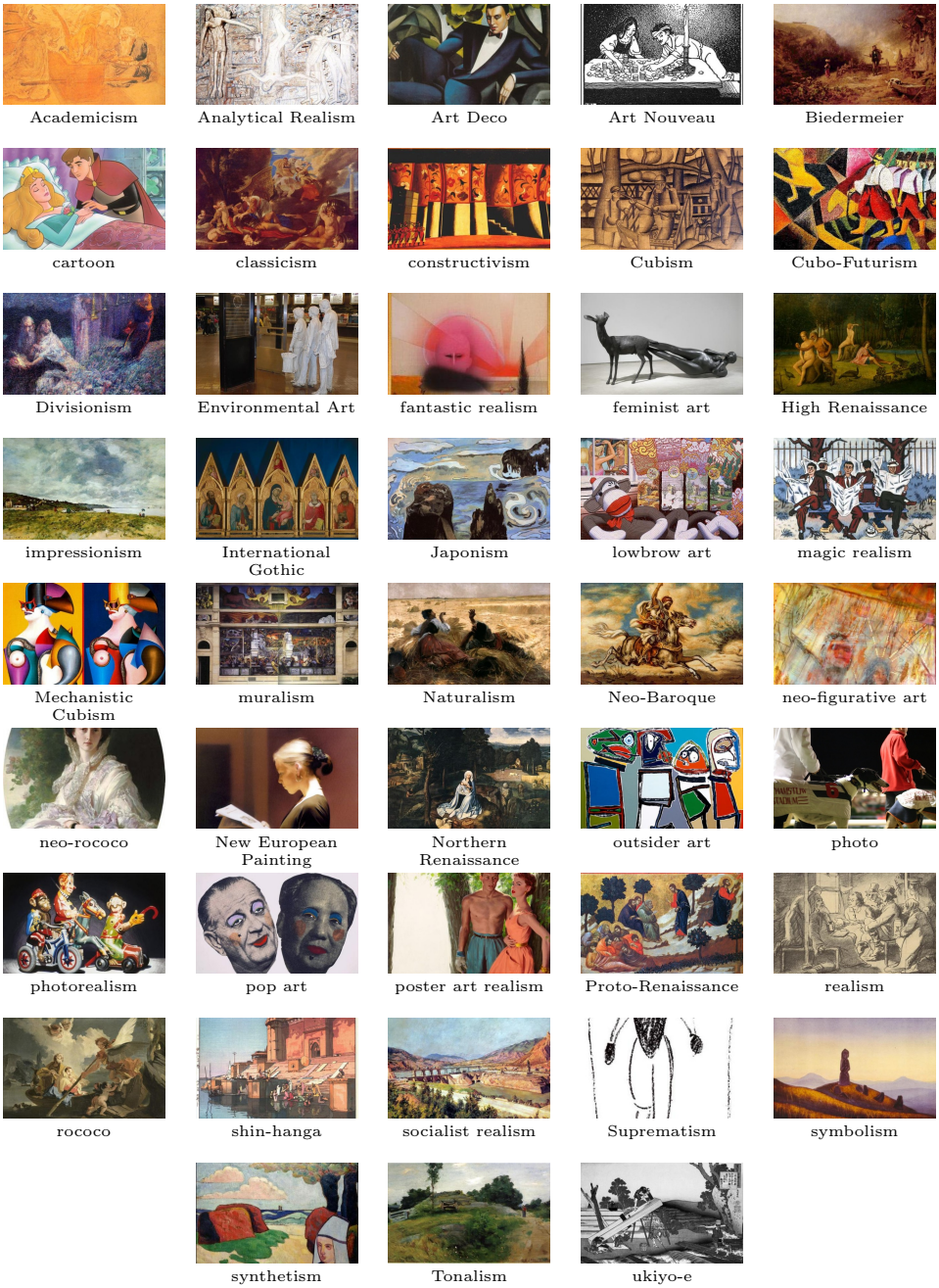

Figure 5.4: The *People-Art* dataset (Westlake, Cai and Hall, 2016) contains images from 43 different depictive styles: here is one example for every depictive style.

Cai and Hall (2016) have observed that artwork, in general, exhibits a wider variety of poses than photos. Figure 5.7 shows examples of images with different poses.

overlapping, occluded and truncated people This occurs in artwork as in photos, and perhaps to a greater extent. Figure 5.8 shows some examples.

5.1 Methodology

I use the same architecture as the "Fast Region-based Convolutional Network" method (Fast R-CNN) (Girshick, 2015), which represented a state-of-the-art object detector in 2015. Fast R-CNN is built upon a modified version of the Caffe library (Jia et al., 2014). A CNN using this architecture has two inputs: an image and a set of class-agnostic rectangular region proposals. The region proposals are a large number of rectangular regions, all with horizontal and vertical sides, which are the output of another algorithm.

There are many algorithms for generating region proposals. All experiments in this chapter use selective search (Uijlings et al., 2013) with the default configuration. The number of regions produced for a single image is large, however much smaller than the number of all possible regions for an image. Figure 5.9 shows an example of the many regions generated by selective search on an image: only 1 in 10 are shown for clarity.

The first stage of the CNN consists of convolutional layers, rectified linear units (ReLUs) (Krizhevsky, Sutskever and Hinton, 2012; Nair and Hinton, 2010), max-pooling layers, and, in some cases, local response normalisation (LRN) layers (Krizhevsky, Sutskever and Hinton, 2012) (see (2.5)). This stage operates on the entire image (having been resized to a more consistent size while preserving aspect ratio). The region proposals enter the CNN only at the final pooling layer of this stage. This final layer is a region of interest (ROI) pooling layer which is novel to Fast R-CNN: as well as the input from the previous convolutional or ReLU layer, this layer receives another input, a region proposal or ROI; the output is a fixed-length feature vector formed by max-pooling of the convolutional stage features. (See Figure 5.22 for a comparison of the ROI pooling layer compared to an ordinary max-pooling layer.)

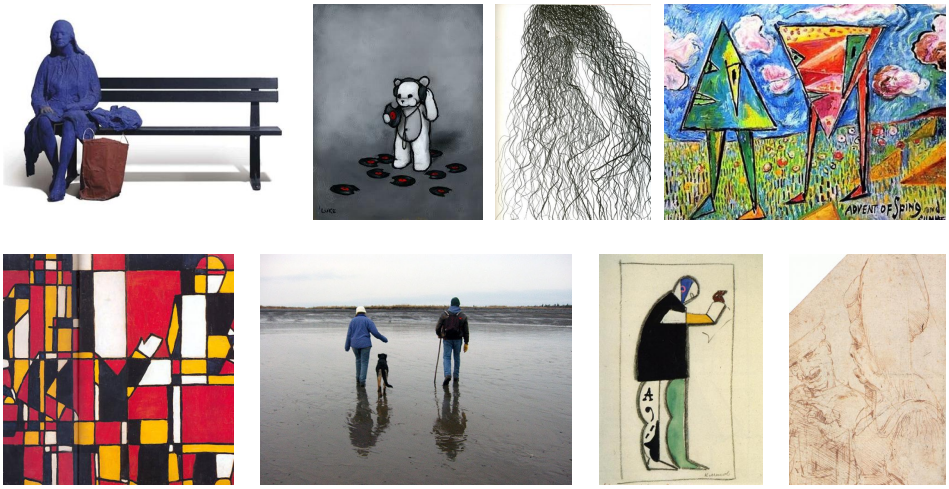

Figure 5.5: The *People-Art* dataset contains a range of denotational styles, including acrylic paint, oil paint, ink, pencil and sculpture.

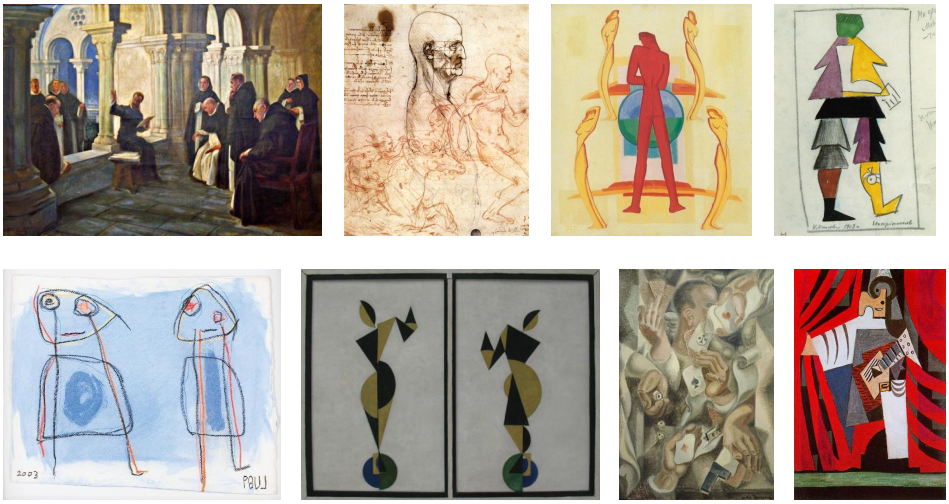

Figure 5.6: The *People-Art* dataset contains a range of projective styles including linear camera projection, orthogonal projection, Cubism and child-like art.

Figure 5.7: The *People-Art* dataset contains a range of poses, perhaps a wider variety compared to photos.

Figure 5.8: The *People-Art* dataset contains images with overlapping, occluded or truncated people

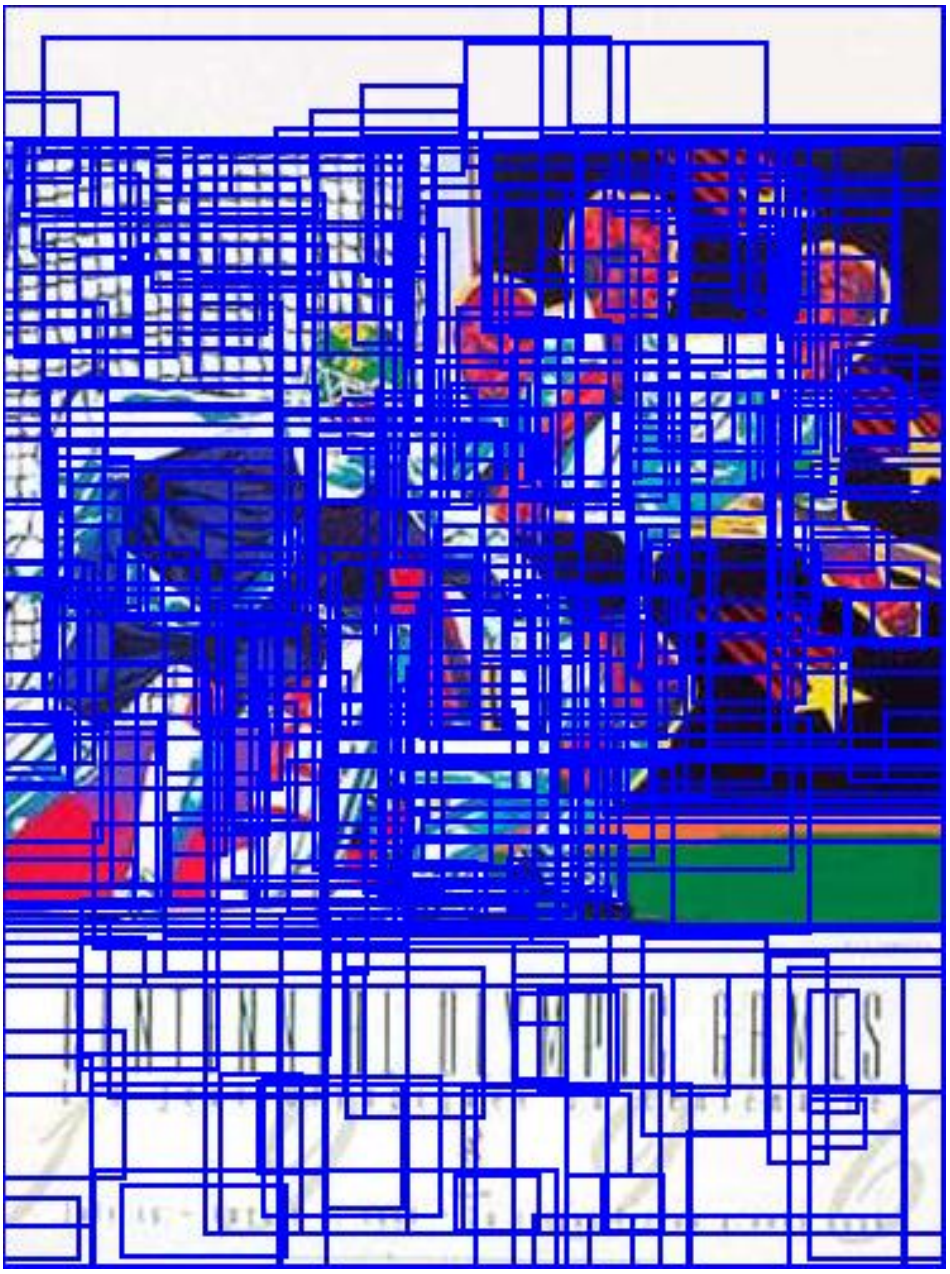

Figure 5.9: Region proposals (blue) generated by selective search (Uijlings et al., 2013) on "Hockey" by Hiro Yamagata

In order to preserve information about the global structure of the ROI, i.e. at a scale within an order of magnitude of the ROI size, the max-pooling happens over a uniformly spaced rectangular grid, size $H \times W$. As a result, the layer outputs feature vector with CHW dimensions where C is the number of channels of the previous convolutional layer.

The input of the second stage of the CNN, which is fully connected, is the CHW–dimensional feature vector. This stage consists of inner product and ReLU layers, as well as dropout layers (training only) which are aimed at preventing overfitting (Srivastava et al., 2014). The output for each class is a score and a set of four co-ordinates which indicate the bounding box coordinates relative to the ROI, providing an adjustment to the bounding box proposal. In the original CNN architecture, the final layer outputs a score over many classes, as well as a bounding box prediction. Hence, for the experiments of this chapter, I modify the final layer of any imported CNN architecture to

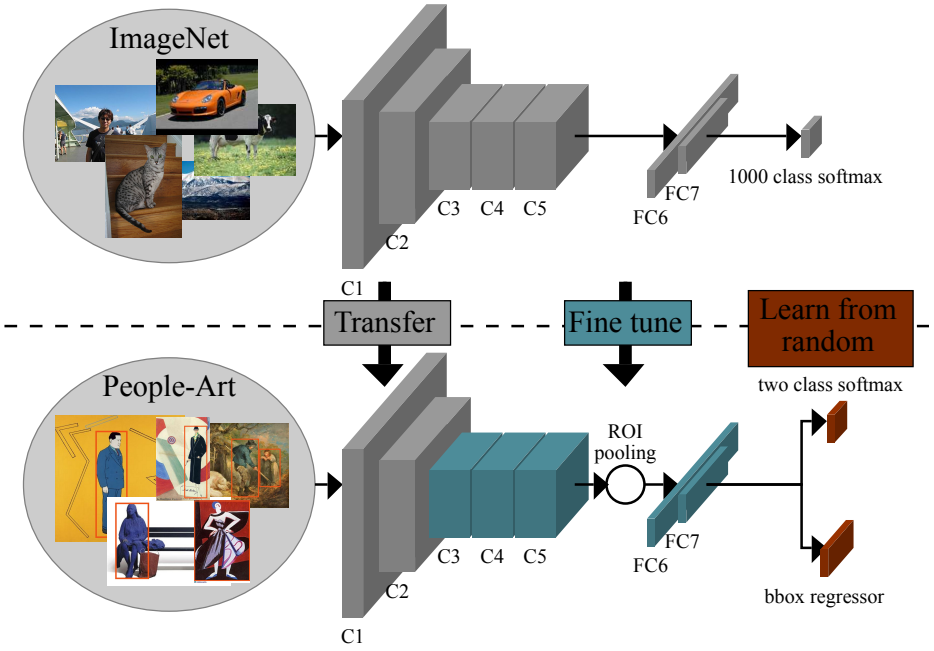

Figure 5.10: The CNNs use a network pre-trained on ImageNet and fine-tuned on the *People-Art* dataset (training and validation sets): I fix the weights for the first F layers, selected by validation.

output a score and bounding box prediction for only one class: person.

The approach used for training the CNN is identical to that of Fast R-CNN, save where expressly stated. This involves the use of stochastic gradient descent (SGD) with momentum (Krizhevsky, Sutskever and Hinton, 2012) and uses networks initialised with weights from the pre-trained models. All experiments in this chapter use CNN models pre-trained on ImageNet (Deng et al., 2009; Krizhevsky, Sutskever and Hinton, 2012).

Both CaffeNet and VGG1024 have five convolutional layers and LRN layers and vary slightly: in particular VGG1024 has more weights and channels. VGG16 is much a larger network, with thirteen convolutional layers and no LRN layers. Except for the number of dimensions, all three networks have the same ROI pooling layer and fully connected network structure: each CNN's fully connected network structure consists of two inner product layers, each followed by ReLU and dropout layers (training only). Table 5.2 summarises the networks, as well indicating the number of filters or outputs in each level.

I fine-tune the models (pre-trained on ImageNet) using the *People-Art* dataset (training and validation sets). I test three different models:

CaffeNet a reproduction of AlexNet (Krizhevsky, Sutskever and Hinton, 2012) with some minor changes

VGG1024 Oxford Visual Geometry Group (VGG)'s "CNN M 1024" (VGG1024) (Chatfield, Simonyan et al., 2014)

VGG16 Oxford VGG's "Net D" (Simonyan and Zisserman, 2015)

All CNN models are unchanged from Girshick 2015, save for changing the number of class output to one plus the background class, and where expressly stated in the next section or for an individual experiments. The original implementation and models pre-trained on ImageNet can be found online at https://github.com/rbgirshick/fast-rcnn. The implementation of selective search is not bundled and can be found online at http://disi.unitn.it/~uijlings/MyHomepage/index.php. I use the default settings for selective search in all cases.

5.1.1 Fine turning the CNNs

In order to fine-tune the CNN, the experiments involve fixing the weights of the first F convolutional layers to those in the pre-trained

CNN	num. layers (num. filters/outputs for each)		LRN
	convolutional	**fully connected**	
AlexNet	5 (96, 256, 384, 384, 256)	2 (4096, 4096)	Yes
VGG1024	5 (96, 256, 512, 512, 512)	2 (4096, 1024)	Yes
VGG16	13 (64, 64, 128, 128, 256, 256, 256, 512, 512, 512, 512, 512, 512)	2 (4096, 4096)	No

Table 5.2: The structure of the CNNs including the number of convolutional and fully connected layers, the number of filters or outputs each layer has and whether or not the network uses LRN

ImageNet model; this parameter is selected by validation. This is achieved by setting the learning rate of these convolutional layers to zero, such that the SGD does not change the weights. Figure 5.10 shows the network architecture in detail.

Before fine-tuning the parameters (weights and biases) of the network are set to those in the models pre-trained on ImageNet, for all layers except the final inner product layers. Since the final inner product layer has a different size output as the experiments involve the detection of only one class, people, I initialise the weights for this layer using random Gaussian initialisation with zero mean and a standard deviation of 0.01 (person vs not person classification) or 0.001 (bounding box prediction); I initialise the all biases to zero. These values are identical to those of Fast R-CNN. In addition, the experiments involve different criteria for the region proposals to use as training ROI in the fine-tuning, as detailed in Section 5.2.2.

In Fast R-CNN, the authors train for 30 000 iterations with a learning rate of 0.001, reduce the learning rate to 0.0001, and then train for another 10 000 iterations. I do the same for the fine-tuning, except that I train for 20 000 iterations, which was found to yield a slight performance improvement on the validation set.

5.2 Experiments

I test the performance of the approach on two different datasets, the People-Art dataset and the Picasso Dataset, with fine-tuning and validation performed on People-Art. The benchmark for both valida-

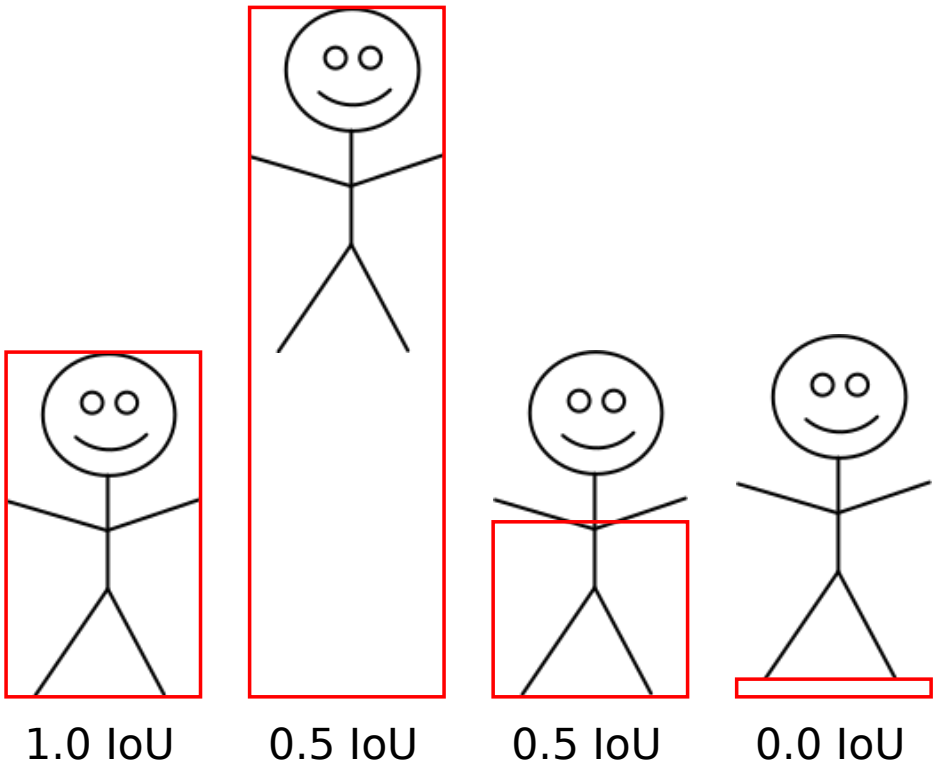

Figure 5.11: Examples of intersection over union values: the red bounding box represents the bounding box produced by an object detector; the ground truth bounding box is equal to the bounding box of the left most stick figure

tion and testing is average precision (AP), calculated using the same method as the PASCAL VOC detection task (Everingham and Winn, 2007). A positive detection is one whose intersection over union (IoU) overlap with a ground truth bounding box is greater than 50%; duplicate detections are considered false positives. Annotations marked as difficult are excluded.

5.2.1 Intersection over Union (IOU)

The intersection over union (IoU), in Computer vision, is a measure of how close an object detection, with a bounding box, reflects the

		ROI IoU			
CNN	**configuration**	**neg**	**pos**	**F**	**AP**
CaffeNet	default	$[0.1, 0.5)$	≥ 0.5	2	33.7%
CaffeNet	gap	$[0.1, 0.4)$	≥ 0.6	2	33.5%
CaffeNet	**all-neg**	$[\mathbf{0.0}, \mathbf{0.5})$	$\geq \mathbf{0.5}$	**0**	**42.5%**
CaffeNet	gap + all-neg	$[0.0, 0.4)$	≥ 0.6	1	42.2%
VGG1024	default	$[0.1, 0.5)$	≥ 0.5	1	38.4%
VGG1024	gap	$[0.1, 0.4)$	≥ 0.6	3	35.8%
VGG1024	**all-neg**	$[\mathbf{0.0}, \mathbf{0.5})$	$\geq \mathbf{0.5}$	**1**	**42.6%**
VGG1024	gap + all-neg	$[0.0, 0.4)$	≥ 0.6	1	42.0%
VGG16	default	$[0.1, 0.5)$	≥ 0.5	1	43.9%
VGG16	gap	$[0.1, 0.4)$	≥ 0.6	2	39.0%
VGG16	all-neg	$[0.0, 0.5)$	≥ 0.5	3	50.0%
VGG16	**gap + all-neg**	$[\mathbf{0.0}, \mathbf{0.4})$	$\geq \mathbf{0.6}$	**3**	**50.1%**

Table 5.3: Validation performance using different criteria for positive and negative ROI using CNNs pre-trained on ImageNet, fine-tuned on the People-Art training set and then evaluated on the People-Art validation set; the best configuration for each CNN is in bold.

ground truth (GT) and is defined as follows:

$$\text{IoU} = \frac{\text{Intersection of detection and GT bounding box area}}{\text{Union of detection and GT bounding box area}} \quad (5.1)$$

Figure 5.11 shows examples of different IoU values.

5.2.2 Hyper-parameter Selection

I optimised performance on the People-Art validation set by modifying the hyper-parameters used for selecting which ROIs to use in training the CNN and the number of layers to fix, F.

Although the experiments rely on the default selective search settings to generate region proposals, I experimented with different criteria to specify which region proposals to use in training. The default configuration of Fast-RCNN (Girshick, 2015) defines positive ROIs to be region proposals whose IoU overlap with a ground truth bounding

box is at least 0.5, and defines negative ROI to be those whose overlap lies in the interval $[0.1, 0.5)$. The cutoff between positive and negative ROI matches the definition of positive detection according the VOC detection task (Everingham and Winn, 2007). Girshick (2015) states that the lower cut-off (0.1) for negative ROI appears to act as a heuristic to mine hard examples, inspired by the approach used in DPM (Felzenszwalb, Girshick et al., 2010).

I experimented with two alternative configurations for selecting ROIs:

gap Discard ROIs whose IoU overlap with a ground truth bounding box lies in the interval, $[0.4, 0.6)$: this follows the hypothesis that ROIs lying in this interval are ambiguous and impede training performance.

all-neg Remove the lower bound for negative ROI used in training: this is based on the hypothesis that this improves performance for two reasons:

1. This results in the inclusion of ROIs containing classes which appear similar to people, for example animals with faces, to serve as negative examples.
2. This permits the inclusion of more artwork examples, for example images without any people present. This may ameliorate the CNN's ability to discern between people and features resulting from a particular depiction style.

The validation procedure optimises performance for each ROI configuration by selecting, F, the number of convolutional layers to fix to the weights obtained from the ImageNet pre-training. Table 5.3 shows the validation performance for the different criteria, i.e. performance on the validation set after fine-tuning on the *People-Art* training set. Removing the lower bound on negative ROI (all-neg) results in a significant increase in performance, the highest increase to AP being around nine percentage points. Therefore, it appears that what is *not* a person is as important as what *is* a person for fine-tuning. Discarding ROI with an IoU overlap in the interval $[0.4, 0.6)$ (gap) yields mixed results: it was marginally beneficial in one case and detrimental in all others.

The optimal number of convolutional layers for which to fix weights to the pre-trained model, F, varies across the different training configurations, even for the same CNN. The variation in performance could

method	datasets		AP
	pre-train	**fine tuning**	
Fast R-CNN (CaffeNet)	ImageNet	People-Art (train+val)	46%
Fast R-CNN (VGG1024)	ImageNet	People-Art (train+val)	51%
Fast R-CNN (VGG16)	ImageNet	People-Art (train+val)	59%
Fast R-CNN (CaffeNet)	ImageNet	VOC 2007	36%
Fast R-CNN (VGG1024)	ImageNet	VOC 2007	36%
Fast R-CNN (VGG16)	ImageNet	VOC 2007	43%
DPM	People-Art	N/A	33%
YOLO	ImageNet	VOC 2010	45%

Table 5.4: Performance of different methods on the test set of the *People-Art* dataset: the best performance is achieved using a CNN (Fast R-CNN) fine-tuned on *People-Art*

be explained by stochastic variation caused by the use of SGD. The performance falls rapidly for $F \geq 5$; leading to the conclusion that the first three or four convolutional layers transfer well from photos to artwork. Fine-tuning these layers yields no significant improvement nor detriment in performance. In this respect, the experiments show similar results to Yosinski et al. (2014) for this task: i.e. the first three or four convolutional layers are more transferable than later layers, in this case from photos to artwork.

All later experiments, including the performance benchmarks, use the configuration which maximises performance on the *validation set* (bold in Table 5.3) and use CNNs re-trained (fine-tuned) on the combination of the training and the validation set.

5.2.3 Performance Benchmarks on the People-Art Dataset

Table 5.4 shows how each CNN model and other methods perform on the *People-Art test set*. The best performing CNN, VGG16, scores 59% AP, an improvement of 14 percentage points on the best previous result of 45% with "You Only Look Once" (YOLO) (Redmon et al., 2016). It is difficult to conceptualise average precision as a metric, as it is an average value of precision over different recall values (the full

range from 0% to 100%). However, an increase in precision from 45% to 59% at 50% recall (544 bounding boxes out of 1088 bounding boxes detected) would correspond to a reduction in false positive bounding boxes from 665 to 377: a significant reduction.

The results demonstrate the benefits of fine-tuning the CNN on (the *training and validation sets* of) *People-Art* for the task. It is clear that training and fine-tuning a CNN on photos yields a model which overfits and does not generalise to detection in images of other depictive styles.

As noted in Section 5.1, Fast R-CNN (unlike YOLO) relies on an external algorithm, here selective search (Uijlings et al., 2013), to generate region proposals. The experiments here used the default settings, which are tuned to photos. Selective search achieves a recall rate of 98% on the *People-Art test set*. Consequently, the use of selective search does not appear to be a limiting factor for the performance.

I attempted to fine-tune YOLO (Redmon et al., 2016) on *People-Art*. The default configuration results in an exploding gradient, perhaps due to the sparsity of regions containing objects (only people in this case) compared to other datasets. I expect that a brute-force search over the parameters or heuristic may solve this problem.

5.2.4 Detection Performance on People-Art

I used the tools of Hoiem, Chodpathumwan and Dai (2012) to analyse the detection performance of the best performing CNN. Since there is only a single class (person), detections are classed into three types based on their IoU with a ground truth labelling:

Cor correct (true positives) i.e. $IoU \geq 0.5$

Loc false positive caused by poor localisation, $0.1 \leq IoU < 0.5$

BG false positive caused by a background region, $IoU < 0.1$

Figure 5.12 shows the detection trend: the proportion of detection types as the number of detections increases, i.e. from reducing the threshold. At higher thresholds, the majority of incorrect detections are caused by poor localisation; at lower thresholds, background regions dominate. In total, there are 1088 people labelled in the test set which are not labelled difficult. The graph in Figure 5.12 shows a grey dashed line corresponding to this number of detections and a separate

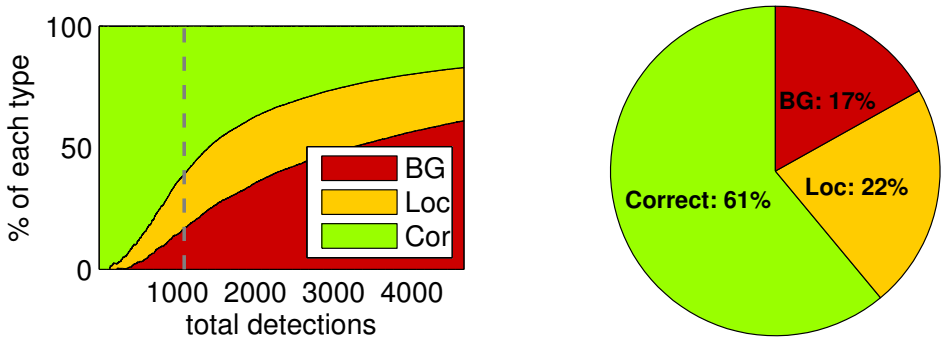

Figure 5.12: Left: The proportion of detections by type as the threshold decreases: either correct, a background region (BG) or poor localisation (LOC); right: the proportion of detections by types for 1088 detections (the correct number of people) marked as a grey dashed line on the left plot

pie chart for this threshold. This threshold is significant: with perfect detection, there would be no false positives or false negatives at this point. This shows that poor localisation is the bigger cause of false positives, though only slightly more so than background regions.

Figures 5.13 and 5.14 show some of the correct positive detections. Figure 5.15 shows some of the false positives caused by background regions. Some are caused by mammals which is understandable given these, like people, have faces and bodies. Others detections have less clear causes. Figure 5.16 shows some of the false positives caused by poor localisation. In some of the cases, the poor localisation is caused by the presence of more than one person, which leads to the bounding box covering multiple people. In other cases, the bounding box does not cover the full extent of the person, i.e. it truncates limbs or the lower torso. I believe that this shows the extent to which the range of poses makes detecting people in artwork a challenging problem.

5.2.5 Performance on the Picasso Dataset

In addition to the results on *People-Art*, I show results on the *Picasso Dataset* (Ginosar et al., 2014). Table 5.5 shows how each CNN and other methods perform. Figures 5.17–5.20 show some of the successful detections and false positive detections on this dataset. While the CNN performs well on a range of complicated artwork, it suffers from

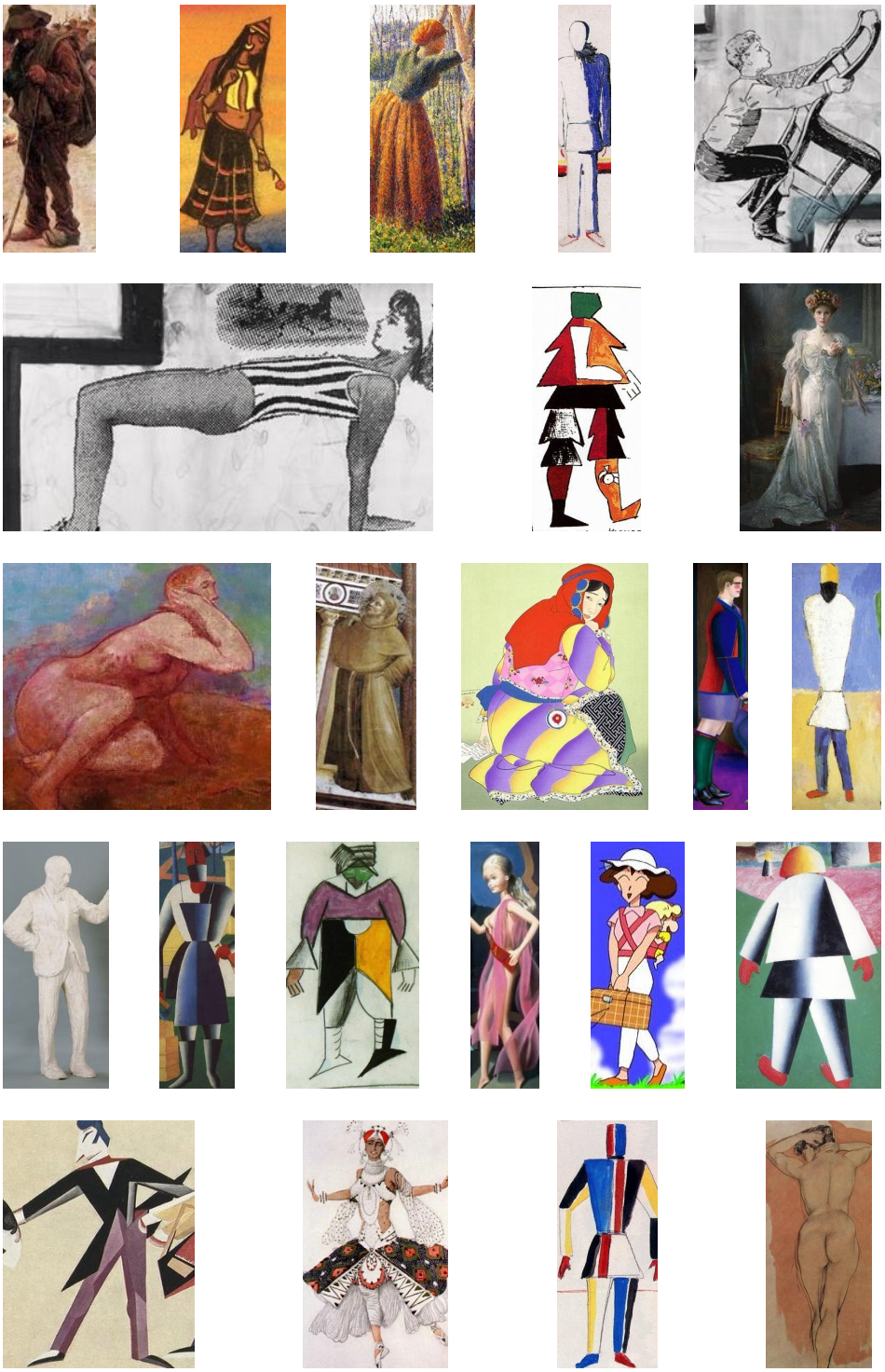

Figure 5.13: Successful detections (*People-Art* dataset) showing the cropped detections, when using the best performing CNN

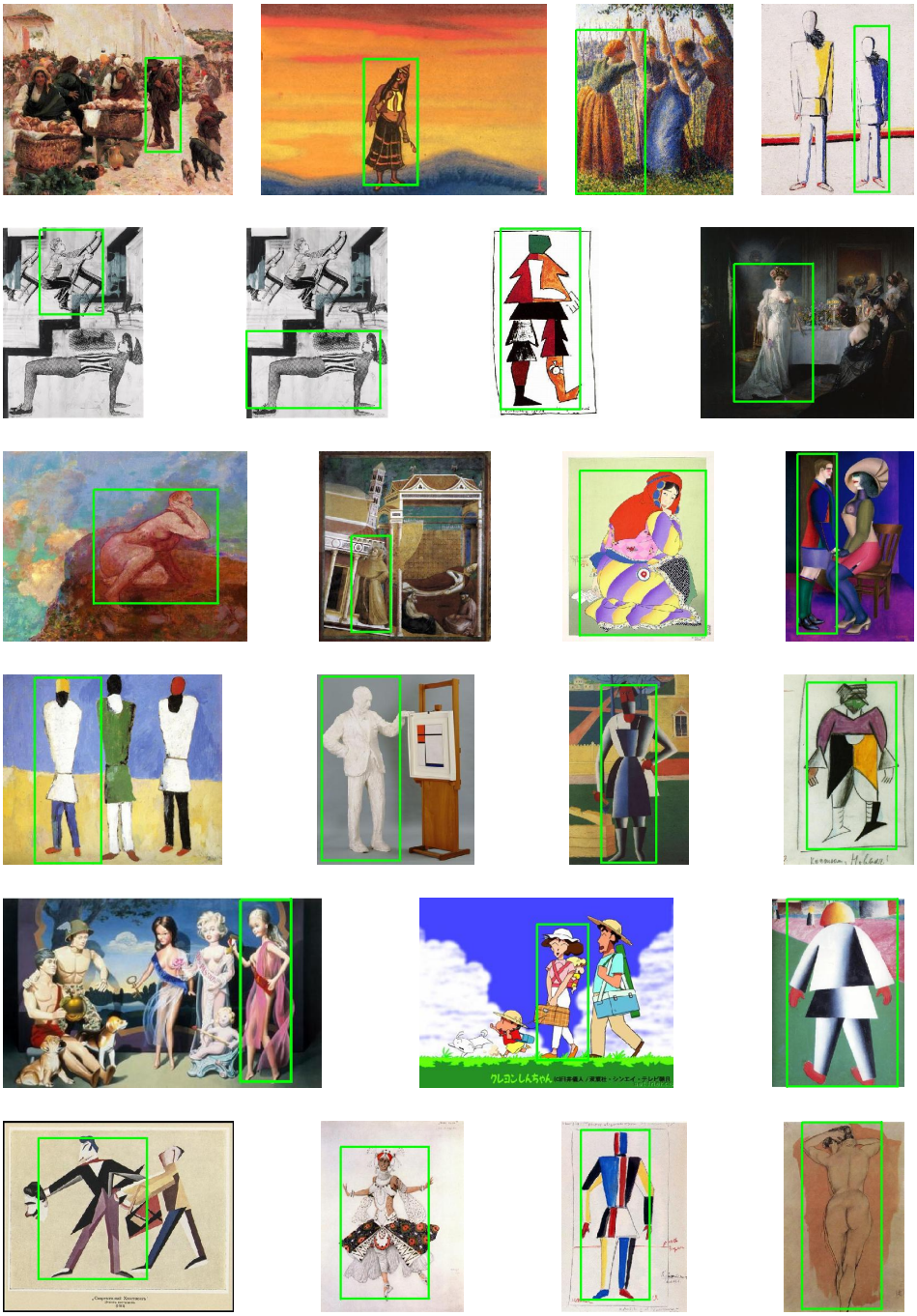

Figure 5.14: Successful detections (*People-Art* dataset) showing the images with bounding box superimposed, when using the best performing CNN

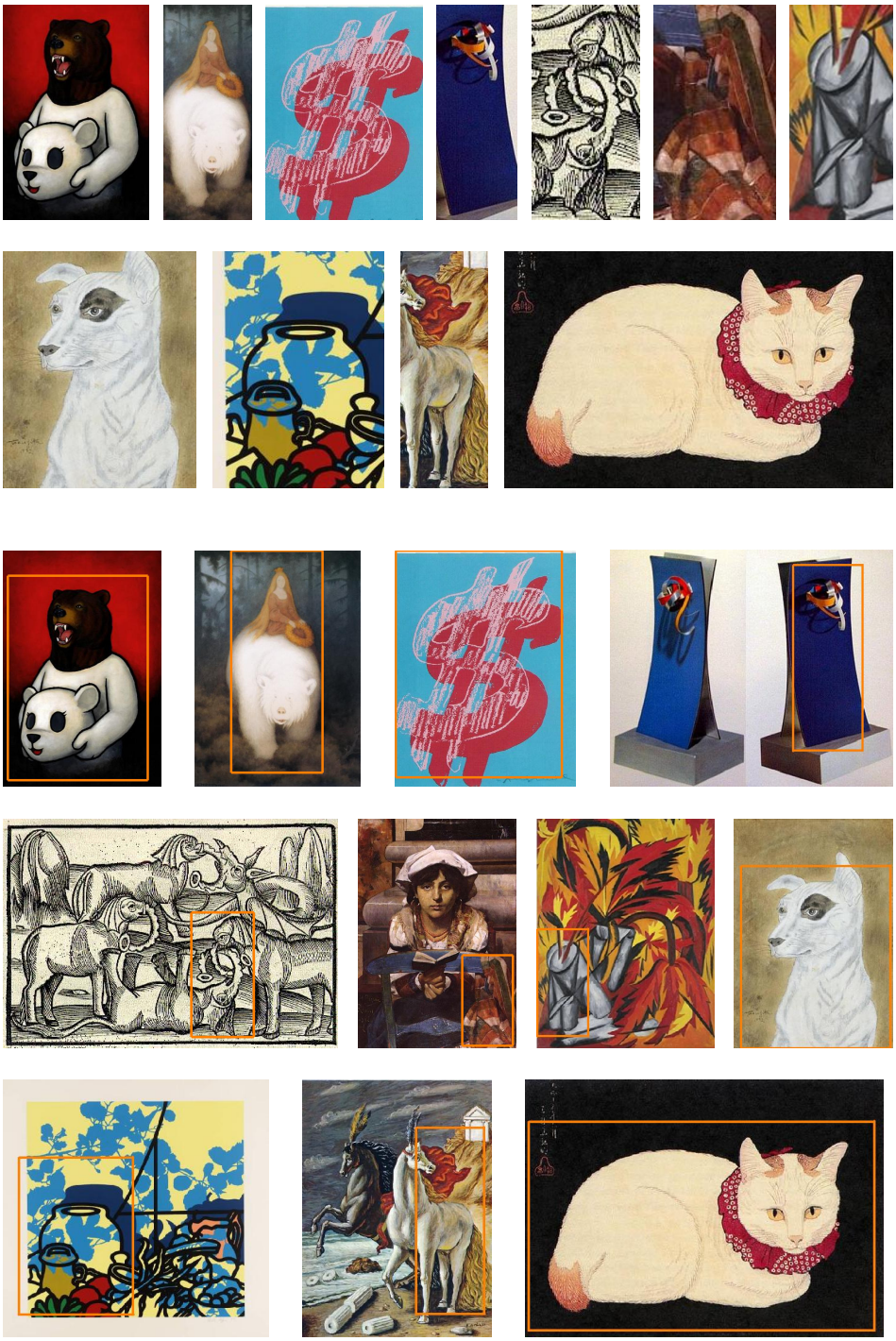

Figure 5.15: False positive detections on background regions (*People-Art* dataset) from the best performing CNN; top two rows: cropped detections; bottom three rows: original images with bounding box superimposed

Figure 5.16: False positive detections due to poor localisation (*People-Art* dataset) from the best performing CNN

method	**training**	**fine tuning**	**AP**
Fast R-CNN (CaffeNet)	ImageNet	People-Art	45%
Fast R-CNN (VGG1024)	ImageNet	People-Art	44%
Fast R-CNN (VGG16)	ImageNet	People-Art	44%
Fast R-CNN (CaffeNet)	ImageNet	VOC 2007	29%
Fast R-CNN (VGG1024)	ImageNet	VOC 2007	37%
Fast R-CNN (VGG16)	ImageNet	VOC 2007	33%
DPM	VOC 2007	N/A	38%
YOLO	ImageNet	VOC 2012	53%

Table 5.5: Performance of different methods on the *Picasso* dataset

similar issues as before, in particular truncating people or mistaking many people for a single person.

Figure 5.21 shows some patches centred on high scoring locations in the images, one for each of the thirty-five channels with the highest mean response prior to the ROI pooling layer. These are obtained using unsupervised part learning approach of Simon and Rodner (2015) using the best performing CNN, CaffeNet. Often the CNN detects part of a face or limb, while in other times the CNN confuses other patterns in the image.

As before, each CNN performed better if it was fine-tuned on *People-Art* rather than *VOC 2007*; moreover, DPM performs better than CNNs fine-tuned on *VOC 2007* but worse than those fine-tuned on

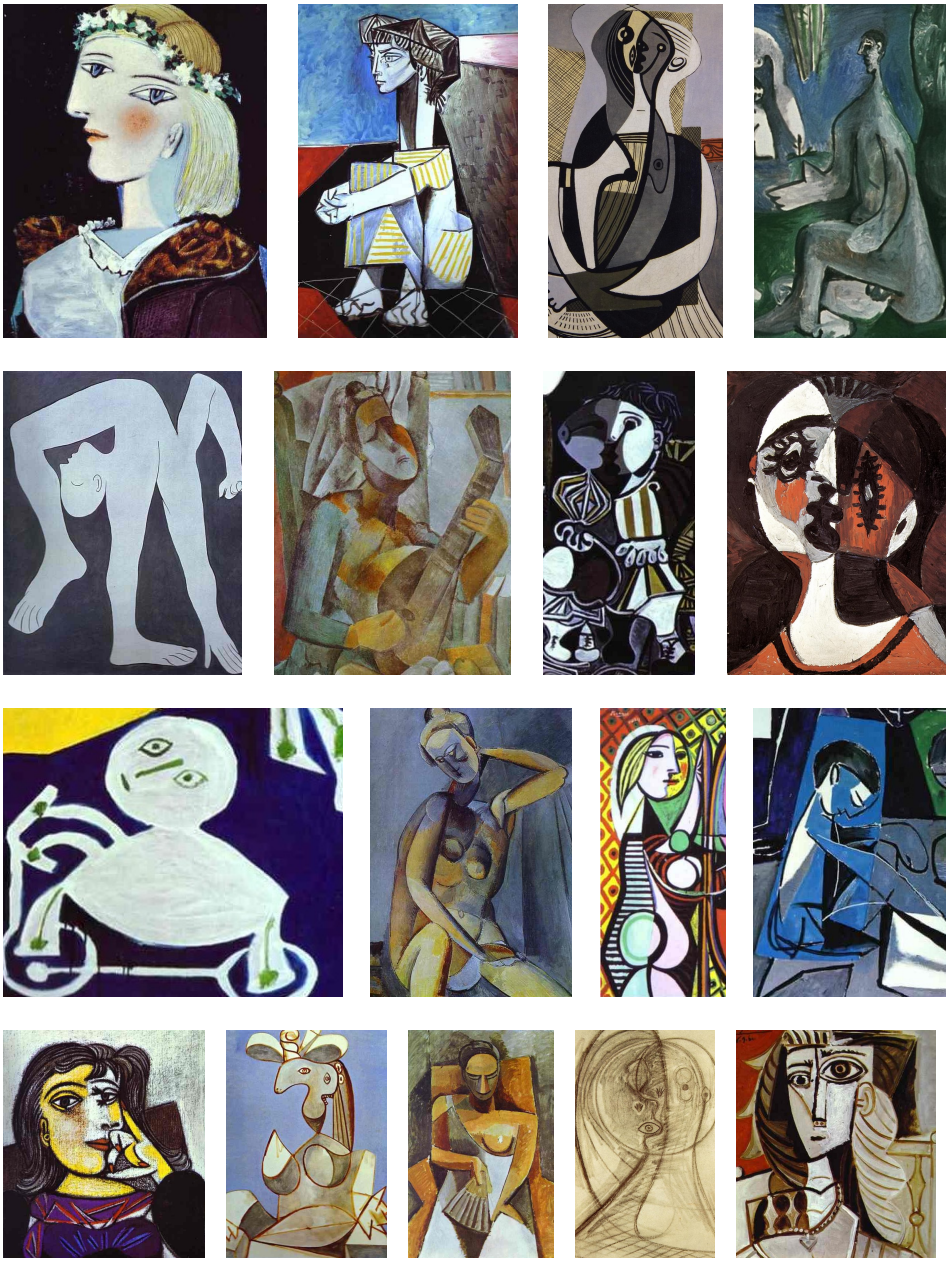

Figure 5.17: Successful detections (*Picasso* dataset) showing the cropped detections, when using the best performing CNN

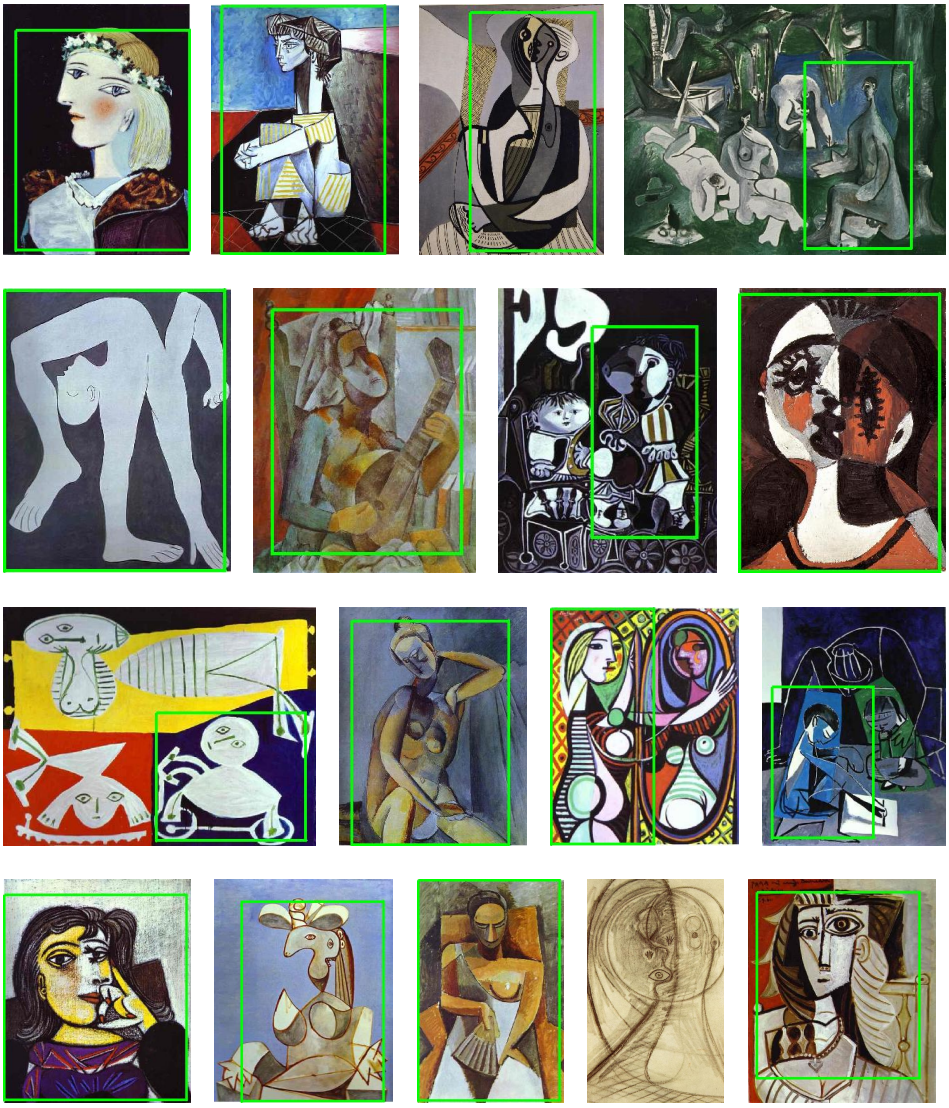

Figure 5.18: Successful detections (*Picasso* dataset) showing the images with bounding box superimposed, when using the best performing CNN

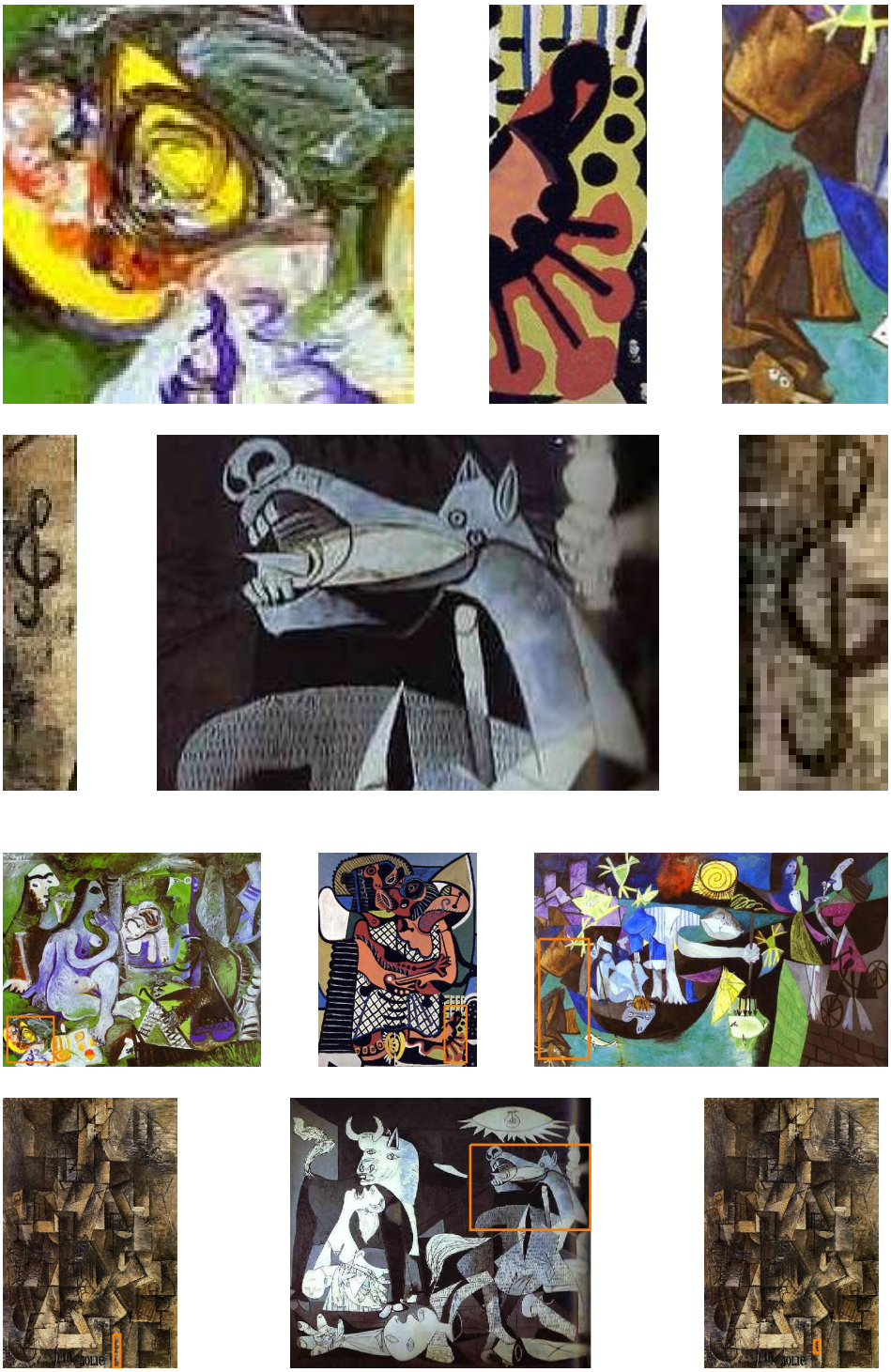

Figure 5.19: False positive detections on background regions (*Picasso* dataset) from the best performing CNN; top two rows: cropped detections; bottom two rows: original images with bounding box superimposed

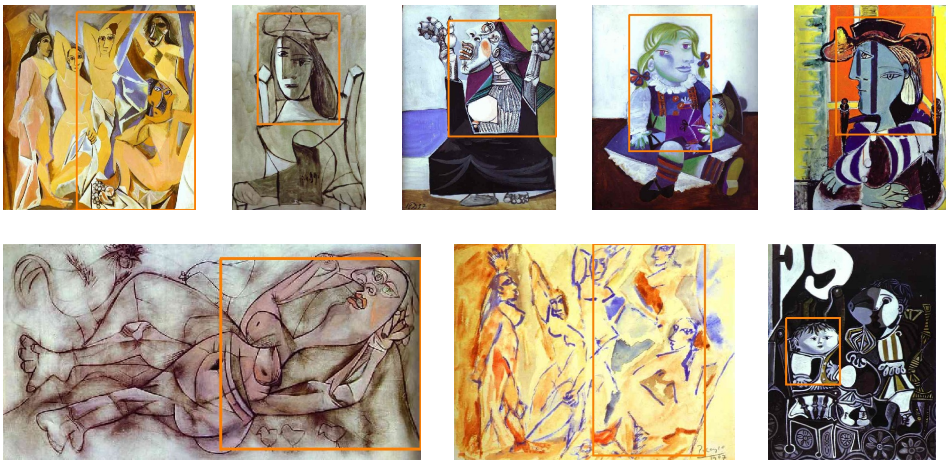

Figure 5.20: False positive detections due to poor localisation (*Picasso* dataset) from VGG16

People-Art. This confirms the earlier findings: CNNs fine-tuned on photos alone overfit to photos. In addition, this shows that fine-tuning on *People-Art* results in a model which is not just better for *People-Art* but additionally better for a dataset also containing artwork but on which the CNN was not fine-tuned.

Interestingly, the best performing CNN is the smallest (CaffeNet), suggesting that the CNNs may still be overfitting to less abstract artwork (from fine-tuning on People-Art, for which Cubist paintings are but a small subset) and a simpler CNN generalises better. Furthermore, the best performing method is YOLO despite being fine-tuned on photos (*VOC 2012*). Selective search achieved a recall rate of 99% on the *Picasso Dataset*, so this is unlikely to be the reason that Fast R-CNN performs worse than YOLO. I therefore believe that YOLO's design is more robust to abstract forms of art.

5.2.6 Importance of Global Structure

Earlier work (Wu, Cai and Hall, 2014; B. Xiao, Song et al., 2008; B. Xiao, Yi-Zhe and Hall, 2011) suggests that structure is invariant across depictive styles, and is therefore useful for cross-depiction detection. As described in Section 5.1, Fast R-CNN includes an ROI pooling layer, which carries out max-pooling over an $H \times W$ uniformly spaced rectangular grid. Therefore, the ROI pooling layer captures the global structure of the person, while earlier convolutional layers only pick up

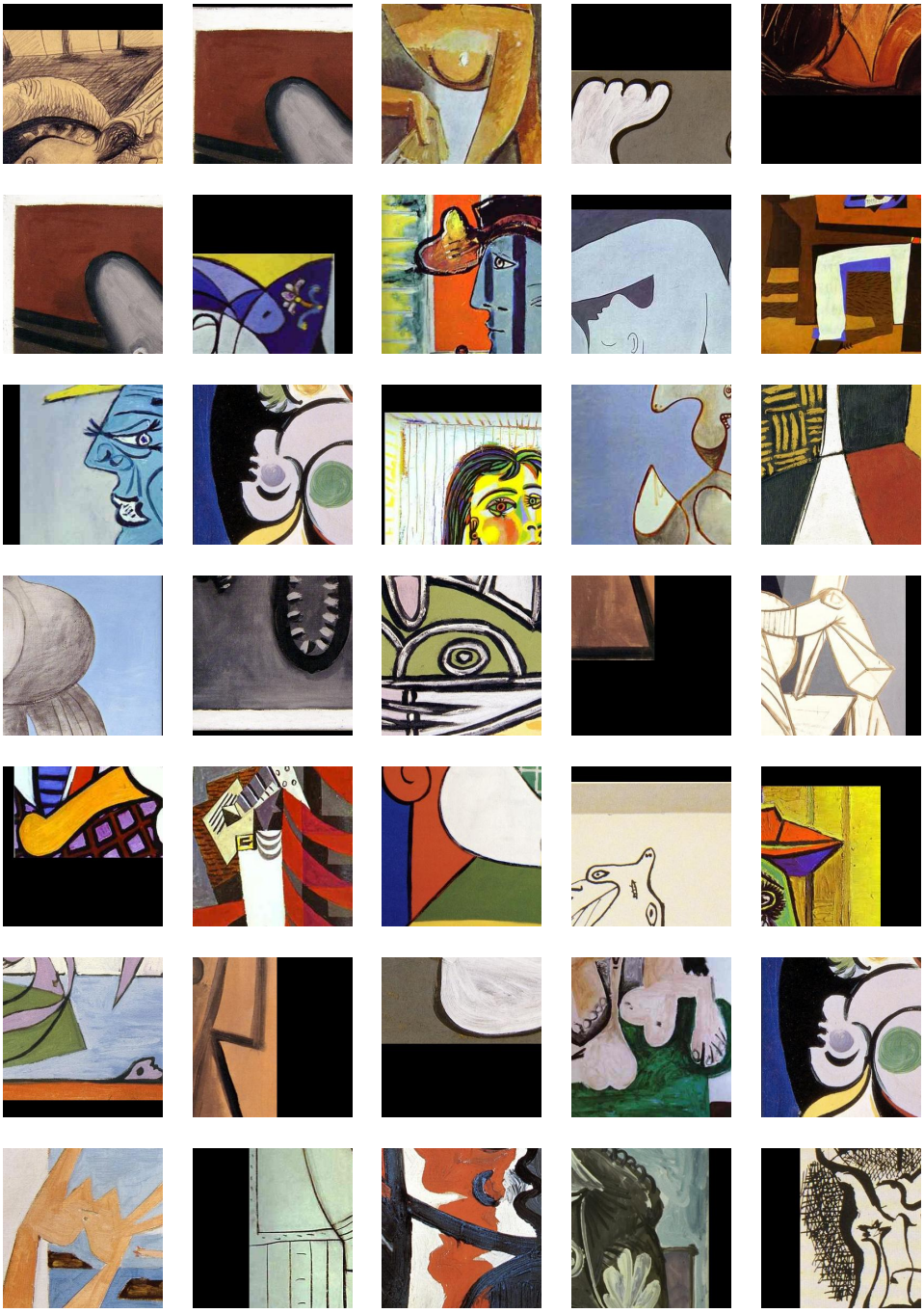

Figure 5.21: Patches at the higher scoring locations in the Picasso Dataset: one for each of the thirty-five channels with the highest mean response prior to the ROI pooling layer

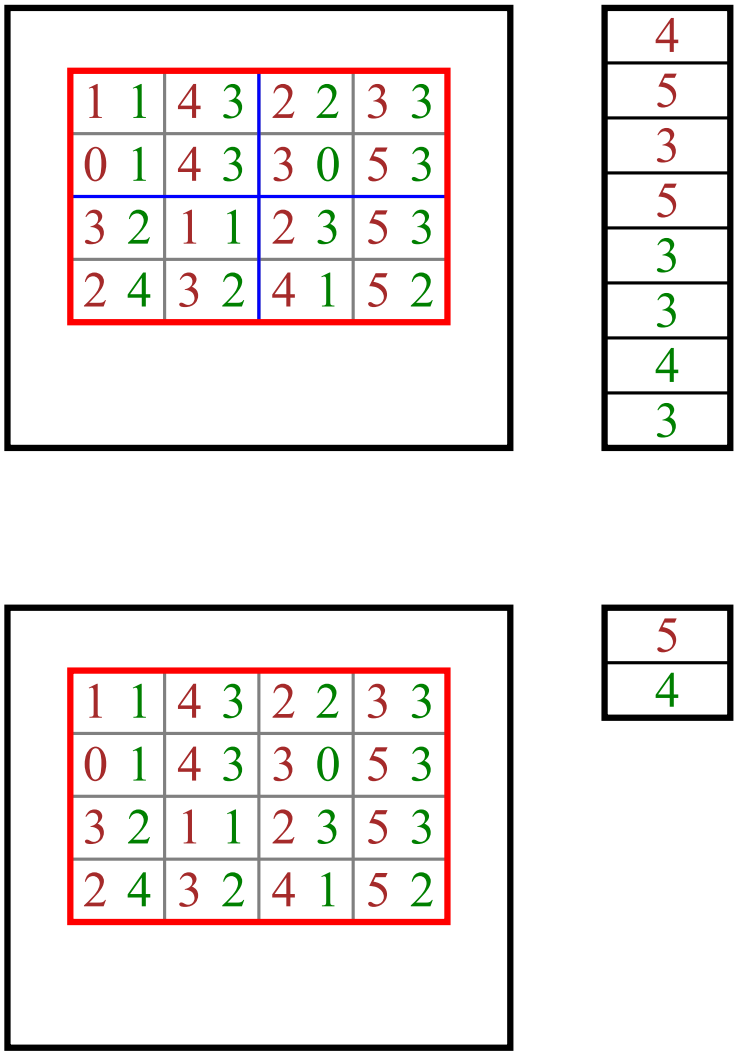

Figure 5.22: Two max-pooling layers and their resulting feature vectors from a two channel input image; top: an ROI pooling layer takes the maximum for each channel in each cell of an ROI (blue grid) resulting in an 8 dimensional vector; Right: a global max-pooling layer simply takes the maximum yielding a 2 dimensional vector

fine tuning	People-Art		VOC 2007
ROI pooling	default	single cell	default
CaffeNet	46%	34%	36%
VGG1024	51%	35%	36%
VGG16	59%	40%	43%

Table 5.6: Replacing the ROI pooling layer (default) with a single cell max-pooling layer yields a performance drop greater than not fine-tuning on *People-Art*

the local structure.

In order to examine what effect the ROI pooling layer and the *global structure* which it captures has on detection performance on artwork, I replaced the ROI pooling layer replaced with a single cell max-pooling layer. This is equivalent to setting $W = 1$ and $H = 1$ for the ROI pooling layer (see Figure 5.22). This is similar to bag-of-words (BoW) algorithms (see Section 2.2.1.3): with $W = H = 1$, the fully connected layers have no information about the location the previous layer's output. The CNN is fine-tuned as before.

Table 5.6 shows the results. In all cases, replacing the default ROI pooling layer with a single cell max-pooling layer results in worse performance. On top of this, the performance is worse than when fine-tuned on *VOC 2007* with the default configuration. This supports the claim of the earlier work, i.e. that structure is invariant across depictive styles and necessary for high performance.

5.2.7 Generalisation Performance

Section 5.2.3 demonstrated that fine-tuning on the People-Art dataset rather than PASCAL VOC 2007 resulted in better performance on the People-Art dataset. An interesting result is the reverse: fine-tuning on the People-Art data and then testing on PASCAL VOC 2007. Given People-Art contains photos (from PASCAL VOC 2012), this is not a domain adaptation task, but rather the ability of the CNNs to learn a model which performs better on artwork without performing worse on photos.

Table 5.7 contains the results of this task, as well as other results for

datasets		AP		
fine tuning	**test**	**CaffeNet**	**VGG1024**	**VGG16**
VOC2007	VOC2007	57%	60%	66%
People-art	VOC2007	42%	48%	52%
VOC2007	People-art	36%	36%	43%
People-art	People-art	46%	51%	59%

Table 5.7: Performance of the CNNs on the People-Art and VOC2007 datasets when fine-tuned on each

comparison. Just as there is a performance drop on People-Art when fine-tuned on PASCAL VOC 2007, so there is also a performance drop on PASCAL VOC 2007 when fine-tuned on the People-Art dataset. In the case of VGG16, the AP drops from 66% to 52%, a drop of 14 percentage points. This demonstrates that these CNNs are not able to generalise to both photos and artwork, even when fine-tuned on People-Art which includes both photos and artwork.

5.3 Conclusion

As demonstrated by the performance on the *People-Art* dataset, detecting people across many depictive styles is challenging. The results show that a CNN trained on photos alone overfits to photos, while fine-tuning on other depictive styles allows the CNN to better generalise to other depictive styles. These findings also apply to the *Picasso* dataset and so are not limited to a single dataset. In addition, the validation results demonstrate the importance of using negative exemplars from artwork with no overlap with a person bounding box, rather than only those with at least a small overlap.

Nevertheless, the performance on the *People-Art* dataset, though the best so far, is still less than 60% AP. The CNN often mistakes other mammals for people and makes other spurious detections. It often fails to localise people correctly, by either truncating them or mistaking multiple people for a single person. Further work is required to address these issues.

In addition, the People-Art dataset only covers a subset of possible images containing people. It does not include African, Babylonian,

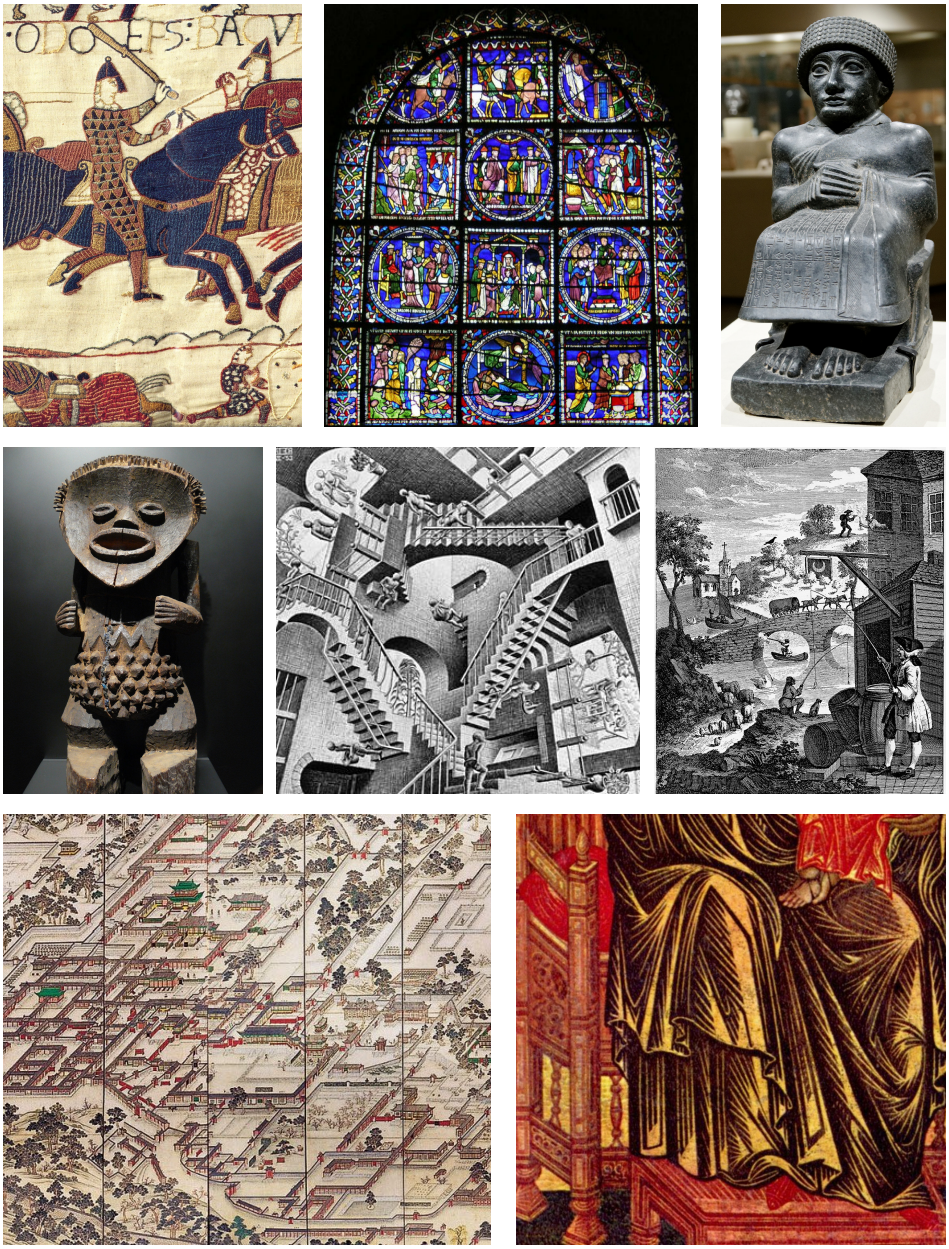

Figure 5.23: Examples of depictive styles not included in the People-Art dataset

Chinese or Egyptian art, the Bayeux Tapestry, stained glass windows, photos of sculptures and all kinds of other possibilities (see Figure 5.23 for some examples). Therefore, the experiments are only beginning to examine the cross-depiction problem, which provides a huge scope for further research.

Furthermore, just as every CNN fine tuned on the PASCAL VOC 2007 dataset performed worse on the People-Art dataset, so every CNN fine tuned on People-Art performed worse on PASCAL VOC 2007. While the first is perhaps understandable, the People-Art dataset includes a selection of photos from PASCAL VOC2012, therefore it is not unreasonable to expect the CNNs to maintain its performance on photos after fine-tuning. It is evident that even the best CNN model cannot simultaneously perform well on photos and artwork. This also suggests that invariance occurs at too low a level: the CNN is unable to learn an abstraction for people at a high enough level to handle the variety across artwork and achieve human levels of detection.

Finally, the results demonstrated the importance of global structure for object detection. However, even the best performing CNN often results in poor localisation, especially when more than one person is in close proximity. This suggests that the CNN has a weak and unreliable model of the structure of a person. As a result, it is often unable to correctly identify how many people are present and what their bounds are.

6 Conclusion

Throughout this dissertation, I have shown that the same thing often looks very different. I began with the task of interest-point matching: matching interest-points between different photos, based on them being the same feature in 3D. In this case, the issues were limited to changes in viewpoint, variations in lighting, occlusion and other artefacts such as blur and compression. Nevertheless, the task remained challenging.

In Chapters 3 and 4, I set out to prove the following thesis:

> A local descriptor can be optimised per interest-point rather than globally to improve matching accuracy.

In Chapter 3, I presented a technique for replacing the low level spatial pooling operation, applied globally across all interest-points, with an interest-point specific low level pooling operation, based on geometric blur. In Chapter 4, I presented a technique for learning an interest-point specific descriptor by learning weights over a number of base-descriptors for each interest-point. Both of these techniques resulted in an increase in matching accuracy, thereby proving the thesis.

Next, I presented the task of detecting people in artwork. There are huge differences between instances of people, even in photos alone, due to variations which include pose, body shape, body features, hair-style and clothes. This variation is aggravated and the challenge is even greater with artwork, as the number of depictive styles increases: people are painted or drawn using many different styles of brush stroke and media and in different projections, especially ad-hoc projections such as in Cubist art.

In Chapter 5, I set out to prove the following thesis:

> It is not yet possible to train or tune state-of-the-art object detectors to simultaneously perform optimally on artwork and photos.

I demonstrated that a convolutional neural network (CNN) trained on photos alone overfits to photos, while fine-tuning on artwork allows

the CNN to perform better at detecting people in artwork. However, performance on photos falls as a result, even though the images used for fine-tuning includes photos in addition to artwork. This proves the second thesis.

All the work that I have presented follows a similar theme: the need be invariant to variation which does not prevent two interest-points from matching or an object being a person, without losing discriminative ability and causing false positives. While the techniques I presented improved performance, the performance still falls short of human performance, suggesting that the techniques are far from reaching the "right" invariance.

Furthermore, none of the technique resulted in a universal benefit. In the case of Chapters 3 and 4, the new techniques were better on average but were not better on all image sequences: there was no universal improvement. In addition, fine-tuning the best performing CNN on a dataset containing both photos and artwork resulted in a performance drop on a dataset containing only photos. This supports my proposition:

> The invariance of state-of-the-art algorithms is at too low a level for generalised interest-point matching and generalised object detection.

This proposition provides scope for further work: developing algorithms with invariance at a higher level.

6.1 Further Work

The approach used in Chapter 3 involves using synthetic warps of the images to estimate the distribution over the appearance of corresponding interest-points in other images. However, the performance is limited because of the strong assumptions implied by the warping operation. The approach in Chapter 4 involves combining a number of base-descriptors into an optimal descriptor based on multiple positives and negative exemplars. However, the performance is limited because of the strong assumptions implied by the base-descriptors.

Both of these approaches involve applying different spatial pooling to low level features, modifying the spatial pooling for each interest-point, to achieve invariance. However, for each image in an image

sequence, a human is able to obtain a semantic understanding, identifying individual objects such as windows and columns, as well as estimating the 3D geometry of the scene. Using this information, if a Computer vision algorithm could obtain it, would provide a better indicator of how to match each interest-point individually.

Furthermore, if matching occurred at the semantic level, e.g. matching window to window and column to column, prior to determining whether the two scenes match, the matching performance would be improved. Likewise, the semantic information gives a clue as to likely future presence of parts of an image: in particular, the presence and appearance of buildings tends to remain the same for much longer than people and vehicles. I therefore propose, as further work, investigation into using state-of-the-art deep learning methods to obtain a semantic understanding of the scene and incorporating this into the interest-point matching framework.

Chapter 5 demonstrated that the best performing algorithm was unable to generalise so as to simultaneously perform optimally on photos and artwork. Other issues included poor localisation of people and spurious detections. These results and other work demonstrate the importance of structure for cross-depiction however the poor localisation suggests that the modelling of structure in the algorithms is unsatisfactory.

One solution, inspired by Wu, Cai and Hall (2014), might be to use deep learning with graph structures such as K. Cho et al. (2014), Chung et al. (2014), Li et al. (2015) and Scarselli et al. (2009). However, the critical issue appears to be the inability learn the actual structure, end-to-end, from only bounding boxes, which makes the algorithms in their current forms unsuitable. It appears that there is not yet a satisfactory way to learn parts reliably in an unsupervised matter, a limitation of Felzenszwalb, Girshick et al. (2010), Simon and Rodner (2015) and Wu, Cai and Hall (2014). Addressing this issue provides scope for further work.

An alternative may be to take advantage of body part labelling datasets such as the MPII Human Pose dataset (Andriluka et al., 2014) and seek to modify existing algorithms (Cao et al., 2017; Insafutdinov et al., 2016) to detect body parts in artwork without any additional part-based labelling. It would appear that humans are able to identify individual body parts in artwork, including Cubism and sketches from an early age without being specifically taught. They can use knowledge learnt from seeing the world with their own eyes to interpret

artwork. Achieving this ability with Computer Vision algorithms, on par with humans, would be a triumph of machine learning and artificial intelligence.

Another issue appears to be handling the lack of body parts in images: sometimes only the face or upper torso is present and sometimes parts of the body are occluded. A similar issue was handled for bag-of-word models, e.g. by the Pyramid Match Kernel (PMK) of Grauman and Darrell (2005). It is clear that the CNN model has a limited ability to overcome lack of body parts. However this ability is insufficient, resulting in the algorithm often truncating part of the body in detections. The presence or lack of certain parts would fit in naturally with better structural modelling, as proposed earlier for further work.

Finally, the presence of false positives caused by other mammals and miscellaneous objects on the People-Art dataset suggests that an algorithm would benefit from using knowledge of such objects learnt from photos. This is challenging if they are not labelled individually, however the use of an extra classifier stage, as in Yang et al. (2016) could help address this issue.

References

Aanæs, H., Dahl, A.L. and Pedersen, K.S. (2012). Interesting interest points. *IJCV*, 97(1), pp.18–35.

Agarwal, S., Snavely, N., Simon, I., Seitz, S. and Szeliski, R. (2009). Building rome in a day. *Iccv*. IEEE. 2009, pp.72–79.

Ali, K., Fleuret, F., Hasler, D. and Fua, P. (2012). A real-time deformable detector. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(2), pp.225–239.

Amit, Y. and Geman, D. (1997). Shape quantization and recognition with randomized trees. *Neural computation*, 9(7), pp.1545–1588.

Amit, Y. and Geman, D. (1999). A computational model for visual selection. *Neural computation*, 11(7), pp.1691–1715.

Amit, Y. and Trouvé, A. (2007). Pop: patchwork of parts models for object recognition. *International Journal of Computer Vision*, 75(2), p.267.

Andrews, S., Tsochantaridis, I. and Hofmann, T. (2003). Support vector machines for multiple-instance learning. *Advances in neural information processing systems*. 2003, pp.577–584.

Andriluka, M., Pishchulin, L., Gehler, P. and Schiele, B. (2014). 2d human pose estimation: new benchmark and state of the art analysis. *Proceedings of the ieee conference on computer vision and pattern recognition*. 2014, pp.3686–3693.

Arandjelović, R. and Zisserman, A. (2012). Three things everyone should know to improve object retrieval. *Computer vision and pattern recognition (cvpr), 2012 ieee conference on*. IEEE. 2012, pp.2911–2918.

Arora, S., Bhaskara, A., Ge, R. and Ma, T. (2014). Provable bounds for learning some deep representations. *International conference on machine learning*. 2014, pp.584–592.

Bach, F.R., Lanckriet, G.R. and Jordan, M.I. (2004). Multiple kernel learning, conic duality, and the SMO algorithm. *Acm*. ACM. 2004, p.6.

Badrinarayanan, V., Kendall, A. and Cipolla, R. (2017). Segnet: a deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Ballard, D.H. (1981). Generalizing the hough transform to detect arbitrary shapes. *Pattern recognition*, 13(2), pp.111–122.

Balntas, V., Lenc, K., Vedaldi, A. and Mikolajczyk, K. (2017). Hpatches: a benchmark and evaluation of handcrafted and learned local descriptors. *Proceedings of the ieee conference on computer vision and pattern recognition*. 2017.

Balntas, V., Riba, E., Ponsa, D. and Mikolajczyk, K. (2016). Learning local feature descriptors with triplets and shallow convolutional neural networks. *Bmvc*. Vol. 1, 2. 2016, p.3.

Balntas, V., Tang, L. and Mikolajczyk, K. (2015). Bold-binary online learned descriptor for efficient image matching. *Proceedings of the ieee conference on computer vision and pattern recognition*. 2015, pp.2367–2375.

Bay, H., Tuytelaars, T. and Van Gool, L. (2006). Surf: speeded up robust features. *Computer vision–ECCV 2006*, pp.404–417.

Beis, J.S. and Lowe, D. (1997). Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. *Computer vision and pattern recognition, 1997. proceedings., 1997 ieee computer society conference on*. IEEE. 1997, pp.1000–1006.

Belongie, S., Malik, J. and Puzicha, J. (2001). Matching shapes. *Computer vision, 2001. iccv 2001. proceedings. eighth ieee international conference on*. Vol. 1. IEEE. 2001, pp.454–461.

Belongie, S., Malik, J. and Puzicha, J. (2002). Shape matching and object recognition using shape contexts. *IEEE transactions on pattern analysis and machine intelligence*, 24(4), pp.509–522.

Benenson, R., Mathias, M., Timofte, R. and Van Gool, L. (2012). Pedestrian detection at 100 frames per second. *Computer vision and pattern recognition (cvpr), 2012 ieee conference on*. IEEE. 2012, pp.2903–2910.

Benenson, R., Mathias, M., Tuytelaars, T. and Van Gool, L. (2013). Seeking the strongest rigid detector. *Proceedings of the ieee conference on computer vision and pattern recognition*. 2013, pp.3666–3673.

Berg, A.C. and Malik, J. (2001). Geometric blur for template matching. *Cvpr*. Vol. 1. IEEE. 2001, pp.I–607.

Birk, J., Kelley, R., Chen, N. and Wilson, L. (1979). Image feature extraction using diameter-limited gradient direction histograms. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (2), pp.228–235.

Bishop, C.M. (2006). *Pattern recognition and machine learning*. Springer, 2006.

Boiman, O. and Irani, M. (2007). Detecting irregularities in images and in video. *International journal of computer vision*, 74(1), pp.17–31.

Bosch, A., Zisserman, A. and Munoz, X. (2007). Representing shape with a spatial pyramid kernel. *Acm*. ACM. 2007, pp.401–408.

Bourdev, L., Maji, S., Brox, T. and Malik, J. (2010). Detecting people using mutually consistent poselet activations. *Computer Vision–ECCV 2010*, pp.168–181.

Boureau, Y.-L., Bach, F., LeCun, Y. and Ponce, J. (2010). Learning mid-level features for recognition. *Computer vision and pattern recognition (cvpr), 2010 ieee conference on*. IEEE. 2010, pp.2559–2566.

Boureau, Y.-L., Ponce, J. and LeCun, Y. (2010). A theoretical analysis of feature pooling in visual recognition. *Proceedings of the 27th international conference on machine learning (icml-10)*. 2010, pp.111–118.

Boykov, Y., Veksler, O. and Zabih, R. (2001). Fast approximate energy minimization via graph cuts. *IEEE Transactions on pattern analysis and machine intelligence*, 23(11), pp.1222–1239.

Breuel, T.M. (1994). A system for the off-line recognition of handwritten text. *Pattern recognition, 1994. vol. 2-conference b: computer vision & image processing., proceedings of the 12th iapr international. conference on*. Vol. 2. IEEE. 1994, pp.129–134.

Bromley, J., Guyon, I., LeCun, Y., Säckinger, E. and Shah, R. (1994). Signature verification using a" siamese" time delay neural network. *Advances in neural information processing systems*. 1994, pp.737–744.

Brown, M., Hua, G. and Winder, S. (2011). Discriminative learning of local image descriptors. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(1), pp.43–57.

Brown, M. and Lowe, D. (2002). Invariant features from interest point groups. *Bmvc*. Vol. 4. 2002.

Brown, M. and Lowe, D. (2005). Unsupervised 3d object recognition and reconstruction in unordered datasets. *3-d digital imaging and modeling, 2005. 3dim 2005. fifth international conference on*. IEEE. 2005, pp.56–63.

Brown, M. and Lowe, D. (2007). Automatic panoramic image stitching using invariant features. *IJCV*, 74(1), pp.59–73.

Burges, C.J., Matan, O., LeCun, Y., Denker, J., Jackel, L., Stenard, C., Nohl, C. and Ben, J. (1992). Shortest path segmentation: a method for training a neural network to recognize character strings. *Neural networks, 1992. ijcnn., international joint conference on*. Vol. 3. IEEE. 1992, pp.165–172.

Burl, M.C. and Perona, P. (1996). Recognition of planar object classes. *Computer vision and pattern recognition, 1996. proceedings cvpr'96, 1996 ieee computer society conference on*. IEEE. 1996, pp.223–230.

Burl, M.C., Weber, M. and Perona, P. (1998). A probabilistic approach to object recognition using local photometry and global geometry. *European conference on computer vision*. Springer. 1998, pp.628–641.

Calonder, M., Lepetit, V., Fua, P., Konolige, K., Bowman, J. and Mihelich, P. (2009). Compact signatures for high-speed interest point description and matching. *Computer vision, 2009 ieee 12th international conference on*. IEEE. 2009, pp.357–364.

Calonder, M., Lepetit, V., Strecha, C. and Fua, P. (2010). Brief: binary robust independent elementary features. *Computer Vision–ECCV 2010*, pp.778–792.

Canny, J. (1986). A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (6), pp.679–698.

Cao, Z., Simon, T., Wei, S.-E. and Sheikh, Y. (2017). Realtime multi-person 2d pose estimation using part affinity fields. *Cvpr*. Vol. 1, 2. 2017, p.7.

Cavallaro, A., Salvador, E. and Ebrahimi, T. (2005). Shadow-aware object-based video processing. *IEE Proceedings-Vision, Image and Signal Processing*, 152(4), pp.398–406.

Chatfield, K., Philbin, J. and Zisserman, A. (2009). Efficient retrieval of deformable shape classes using local self-similarities. *Computer vision workshops (iccv workshops), 2009 ieee 12th international conference on*. IEEE. 2009, pp.264–271.

Chatfield, K., Simonyan, K., Vedaldi, A. and Zisserman, A. (2014). Return of the devil in the details: delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*.

Chebbout, S. and Merouani, H.F. (2012). Comparative study of clustering based colour image segmentation techniques. *Signal image technology and internet based systems (sitis), 2012 eighth international conference on*. IEEE. 2012, pp.839–844.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Cho, M., Alahari, K. and Ponce, J. (2013). Learning graphs to match. *Proceedings of the ieee international conference on computer vision*. 2013, pp.25–32.

Chopra, S., Hadsell, R. and LeCun, Y. (2005). Learning a similarity metric discriminatively, with application to face verification. *Computer vision and pattern recognition, 2005. cvpr 2005. ieee computer society conference on*. Vol. 1. IEEE. 2005, pp.539–546.

Chou, C.-H. and Chen, Y.-C. (1990). Moment-preserving pattern matching. *Pattern Recognition*, 23(5), pp.461–474.

Chung, J., Gulcehre, C., Cho, K. and Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Ciresan, D.C., Meier, U., Masci, J., Maria Gambardella, L. and Schmidhuber, J. (2011). Flexible, high performance convolutional neural networks for image classification. *Ijcai proceedings-international joint conference on artificial intelligence*. Vol. 22, 1. Barcelona, Spain. 2011, p.1237.

Cireşan, D.C., Meier, U., Gambardella, L.M. and Schmidhuber, J. (2010). Deep, big, simple neural nets for handwritten digit recognition. *Neural computation*, 22(12), pp.3207–3220.

Crandall, D., Felzenszwalb, P. and Huttenlocher, D. (2005). Spatial priors for part-based recognition using statistical models. *Computer vision and pattern recognition, 2005. cvpr 2005. ieee computer society conference on*. Vol. 1. IEEE. 2005, pp.10–17.

Crowley, E.J. and Zisserman, A. (2014). In search of art. *Workshop at the european conference on computer vision*. Springer. 2014, pp.54–70.

Csurka, G., Dance, C., Fan, L., Willamowski, J. and Bray, C. (2004). Visual categorization with bags of keypoints. *Workshop on statistical learning in computer vision, eccv*. Vol. 1, 1-22. Prague. 2004, pp.1–2.

Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. *Computer vision and pattern recognition, 2005. cvpr 2005. ieee computer society conference on*. Vol. 1. IEEE. 2005, pp.886–893.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. and Fei-Fei, L. (2009). Imagenet: a large-scale hierarchical image database. *Computer vision and pattern recognition, 2009. cvpr 2009. ieee conference on*. IEEE. 2009, pp.248–255.

Dollár, P., Tu, Z., Perona, P. and Belongie, S. (2009). Integral channel features.

Duchi, J., Hazan, E. and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul), pp.2121–2159.

Duchon, J. (1977). Splines minimizing rotation-invariant semi-norms in sobolev spaces. *Constructive theory of functions of several variables*, pp.85–100.

Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J. and Zisserman, A. (2012). *The PASCAL visual object classes challenge 2012 (VOC2012) results*. 2012.

Everingham, M. and Winn, J. (2007). The pascal visual object classes challenge 2007 (voc2007) development kit. *University of Leeds, Tech. Rep*.

Everingham, M., Zisserman, A., Williams, C.K., Van Gool, L., Allan, M., Bishop, C.M., Chapelle, O., Dalal, N., Deselaers, T., Dorkó, G. et al. (2006). The 2005 pascal visual object classes challenge. *Machine learning challenges. evaluating predictive uncertainty, visual object classification, and recognising tectual entailment*. Springer, 2006, pp.117–176.

Fabbri, R., Costa, L.D.F., Torelli, J.C. and Bruno, O.M. (2008). 2d euclidean distance transform algorithms: a comparative survey. *ACM Computing Surveys (CSUR)*, 40(1), p.2.

Farhadi, A., Endres, I. and Hoiem, D. (2010). Attribute-centric recognition for cross-category generalization. *Computer vision and pattern recognition (cvpr), 2010 ieee conference on*. IEEE. 2010, pp.2352–2359.

Farinella, G.M., Battiato, S. and Cipolla, R. (2013). *Advanced topics in computer vision*. Springer, 2013.

Farlow, S. (1981). The gmdh algorithm of ivakhnenko. *The American Statistician*, 35(4), pp.210–215.

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

Felzenszwalb, P. and Huttenlocher, D. (2005). Pictorial structures for object recognition. *International journal of computer vision*, 61(1), pp.55–79.

Felzenszwalb, P., McAllester, D. and Ramanan, D. (2008). A discriminatively trained, multiscale, deformable part model. *Computer vision and pattern recognition, 2008. cvpr 2008. ieee conference on*. IEEE. 2008, pp.1–8.

Fergus, R., Perona, P. and Zisserman, A. (2003). Object class recognition by unsupervised scale-invariant learning. *Computer vision and pattern recognition, 2003. proceedings. 2003 ieee computer society conference on*. Vol. 2. IEEE. 2003, pp.II–II.

Fischler, M.A. and Elschlager, R.A. (1973). The representation and matching of pictorial structures. *IEEE Transactions on computers*, 100(1), pp.67–92.

Förstner, W. (1994). A framework for low level feature extraction. *Computer vision-eccv'94*. Springer, 1994, pp.383–394.

Förstner, W. and Gülch, E. (1987). A fast operator for detection and precise location of distinct points, corners and centres of circular features. *Proc. isprs intercommission conference on fast processing of photogrammetric data*. 1987, pp.281–305.

Freund, Y. and Schapire, R.E. (1995). A desicion-theoretic generalization of on-line learning and an application to boosting. *European conference on computational learning theory*. Springer. 1995, pp.23–37.

Fukushima, K. (2003). Neocognitron for handwritten digit recognition. *Neurocomputing*, 51, pp.161–180.

Fukushima, K. and Miyake, S. (1982). Neocognitron: a self-organizing neural network model for a mechanism of visual pattern recognition. *Competition and cooperation in neural nets*. Springer, 1982, pp.267–285.

Ghodrati, A., Diba, A., Pedersoli, M., Tuytelaars, T. and Van Gool, L. (2015). 'deepproposal: hunting objects by cascading deep convolutional layers. *Proceedings iccv 2015*. 2015, pp.2578–2586.

Ginosar, S., Haas, D., Brown, T. and Malik, J. (2014). Detecting people in cubist art. *Computer vision-eccv 2014 workshops*. Springer. 2014, pp.101–116.

Girshick, R. (2015). Fast r-cnn. *Proceedings of the ieee international conference on computer vision*. 2015, pp.1440–1448.

Girshick, R., Donahue, J., Darrell, T. and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. *Proceedings of the ieee conference on computer vision and pattern recognition*. 2014, pp.580–587.

Goodfellow, I., Bengio, Y. and Courville, A. (2016). *Deep learning* **online**. Book in preparation for MIT Press. 2016. Available from: http://www.deeplearningbook.org.

Goshtasby, A., Gage, S.H. and Bartholic, J.F. (1984). A two-stage cross correlation approach to template matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (3), pp.374–378.

Grauman, K. and Darrell, T. (2005). The pyramid match kernel: discriminative classification with sets of image features. *Computer vision, 2005. iccv 2005. tenth ieee international conference on*. Vol. 2. IEEE. 2005, pp.1458–1465.

Gupta, R. and Mittal, A. (2008). Smd: a locally stable monotonic change invariant feature descriptor. *Computer Vision–ECCV 2008*, pp.265–277.

Hadsell, R., Chopra, S. and LeCun, Y. (2006). Dimensionality reduction by learning an invariant mapping. *Computer vision and pattern recognition, 2006 ieee computer society conference on*. Vol. 2. IEEE. 2006, pp.1735–1742.

Hall, P., Cai, H., Wu, Q. and Corradi, T. (2015). Cross-depiction problem: recognition and synthesis of photographs and artwork. *Computational Visual Media*, 1(2), pp.91–103.

Hamming, R.W. (1950). Error detecting and error correcting codes. *Bell Labs Technical Journal*, 29(2), pp.147–160.

Han, X., Leung, T., Jia, Y., Sukthankar, R. and Berg, A.C. (2015). Matchnet: unifying feature and metric learning for patch-based matching. *Proceedings of the ieee conference on computer vision and pattern recognition*. 2015, pp.3279–3286.

Harris, C. and Stephens, M. (1988). A combined corner and edge detector. *Alvey vision conference*. Vol. 15, 50. Manchester, UK. 1988, pp.10–5244.

He, K., Zhang, X., Ren, S. and Sun, J. (2015). Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(9), pp.1904–1916.

He, K., Zhang, X., Ren, S. and Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the ieee conference on computer vision and pattern recognition*. 2016, pp.770–778.

Hebb, D. (1949). *The organization of behavior: a neuropsychological theory*. Wiley & Sons, 1949.

Hoffer, E. and Ailon, N. (2015). Deep metric learning using triplet network. *International workshop on similarity-based pattern recognition*. Springer. 2015, pp.84–92.

Hoiem, D., Chodpathumwan, Y. and Dai, Q. (2012). Diagnosing error in object detectors. *European conference on computer vision*. Springer. 2012, pp.340–353.

Huang, F.J. and LeCun, Y. (2006). Large-scale learning with svm and convolutional for generic object categorization. *Computer vision and pattern recognition, 2006 ieee computer society conference on*. Vol. 1. IEEE. 2006, pp.284–291.

Hubel, D.H. and Wiesel, T.N. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1), p.106.

Hubel, D.H. and Wiesel, T.N. (1963). Receptive fields of cells in striate cortex of very young, visually inexperienced kittens. *J. neurophysiol*, 26(994), p.1002.

Insafutdinov, E., Pishchulin, L., Andres, B., Andriluka, M. and Schiele, B. (2016). Deepercut: a deeper, stronger, and faster multi-person pose estimation model. *European conference on computer vision*. Springer. 2016, pp.34–50.

Ivakhnenko, A.G. (1968). The group method of data handling—a rival of the method of stochastic approximation. *Soviet Automatic Control*, 13(3), pp.43–55.

Ivakhnenko, A.G. and Lapa, V.G. (1965). *Cybernetic predicting devices*. CCM Information Corporation, 1965.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S. and Darrell, T. (2014). Caffe: convolutional architecture for fast feature embedding. *Proceedings of the 22nd acm international conference on multimedia*. ACM. 2014, pp.675–678.

Joachims, T. (1998). Text categorization with support vector machines: learning with many relevant features. *Machine learning: ECML-98*, pp.137–142.

Joblove, G.H. and Greenberg, D. (1978). Color spaces for computer graphics. *Acm siggraph computer graphics*. Vol. 12, 3. ACM. 1978, pp.20–25.

Johnson, A.E. and Hebert, M. (1999). Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE Transactions on pattern analysis and machine intelligence*, 21(5), pp.433–449.

Jones, M. and Viola, P. (2003). Fast multi-view face detection. *Mitsubishi Electric Research Lab TR-20003-96*, 3, p.14.

Kadir, T. and Brady, M. (2001). Saliency, scale and image description. *International Journal of Computer Vision*, 45(2), pp.83–105.

Kakumanu, P., Makrogiannis, S. and Bourbakis, N. (2007). A survey of skin-color modeling and detection methods. *Pattern recognition*, 40(3), pp.1106–1122.

Karzanov, A. (1992). Quick algorithm for determining the distances from the points of the given subset of an integer lattice to the points of its complement. *Cybernetics and System Analysis*, pp.177–181.

Ke, Y. and Sukthankar, R. (2004). Pca-sift: a more distinctive representation for local image descriptors. *Computer vision and pattern recognition, 2004. cvpr 2004. proceedings of the 2004 ieee computer society conference on*. Vol. 2. IEEE. 2004, pp.II–II.

Kearns, M.J., Schapire, R.E. and Sellie, L.M. (1994). Toward efficient agnostic learning. *Machine Learning*, 17(2-3), pp.115–141.

Koenderink, J.J. (1984). The structure of images. *Biological cybernetics*, 50(5), pp.363–370.

Koenderink, J.J. and Doorn, A.J. van (1987). Representation of local geometry in the visual system. *Biological cybernetics*, 55(6), pp.367–375.

Kohonen, T. (1972). Correlation matrix memories. *IEEE transactions on computers*, 100(4), pp.353–359.

Kong, T., Yao, A., Chen, Y. and Sun, F. (2016). Hypernet: towards accurate region proposal generation and joint object detection. *Proceedings of the ieee conference on computer vision and pattern recognition*. 2016, pp.845–853.

Krizhevsky, A., Sutskever, I. and Hinton, G. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*. 2012, pp.1097–1105.

Lazebnik, S., Schmid, C. and Ponce, J. (2005). A sparse texture representation using local affine regions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8), pp.1265–1278.

Lazebnik, S., Schmid, C. and Ponce, J. (2006). Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. *Computer vision and pattern recognition, 2006 ieee computer society conference on*. Vol. 2. IEEE. 2006, pp.2169–2178.

Le, Q. (2013). Building high-level features using large scale unsupervised learning. *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*. IEEE. 2013, pp.8595–8598.

LeCun, Y. (1989). Generalization and network design strategies. *Connectionism in perspective*, pp.143–155.

LeCun, Y., Bengio, Y. and Hinton, G. (2015). Deep learning. *Nature*, 521(7553), pp.436–444.

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

LeCun, Y., Boser, B., Denker, J., Henderson, D., Howard, R., Hubbard, W. and Jackel, L. (1990). Handwritten digit recognition with a back-propagation network. *Advances in neural information processing systems*. 1990, pp.396–404.

LeCun, Y., Bottou, L. and Bengio, Y. (1997). Reading checks with multilayer graph transformer networks. *Acoustics, speech, and signal processing, 1997. icassp-97., 1997 ieee international conference on*. Vol. 1. IEEE. 1997, pp.151–154.

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

LeCun, Y., Huang, F.J. and Bottou, L. (2004). Learning methods for generic object recognition with invariance to pose and lighting. *Computer vision and pattern recognition, 2004. cvpr 2004. proceedings of the 2004 ieee computer society conference on*. Vol. 2. IEEE. 2004, pp.II–104.

Leibe, B., Leonardis, A. and Schiele, B. (2004). Combined object categorization and segmentation with an implicit shape model. *Workshop on statistical learning in computer vision, eccv*. Vol. 2, 5. 2004, p.7.

Leibe, B., Leonardis, A. and Schiele, B. (2008). Robust object detection with interleaved categorization and segmentation. *International journal of computer vision*, 77(1-3), pp.259–289.

Leibe, B. and Schiele, B. (2004). Scale-invariant object categorization using a scale-adaptive mean-shift search. *Dagm-symposium*. Springer. 2004, pp.145–153.

Lenc, K. and Vedaldi, A. (Sept. 2015). R-cnn minus r. **online** Sept. 2015, 5.1–5.12. Available from: http://dx.doi.org/10.5244/C.29.5.

Lepetit, V. and Fua, P. (2006). Keypoint recognition using randomized trees. *IEEE transactions on pattern analysis and machine intelligence*, 28(9), pp.1465–1479.

Leutenegger, S., Chli, M. and Siegwart, R.Y. (2011). Brisk: binary robust invariant scalable keypoints. *Computer vision (iccv), 2011 ieee international conference on*. IEEE. 2011, pp.2548–2555.

Li, Y., Tarlow, D., Brockschmidt, M. and Zemel, R. (2015). Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*.

Liang, M. and Hu, X. (2015). Recurrent convolutional neural network for object recognition. *Proceedings of the ieee conference on computer vision and pattern recognition*. 2015, pp.3367–3375.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P. and Zitnick, C.L. (2014). Microsoft coco: common objects in context. *European conference on computer vision*. Springer. 2014, pp.740–755.

Lindeberg, T. (1991). *Discrete scale-space theory and the scale-space primal sketch*. PhD thesis. KTH Royal Institute of Technology, 1991.

Lindeberg, T. (1993a). Detecting salient blob-like image structures and their scales with a scale-space primal sketch: a method for focus-of-attention. *International Journal of Computer Vision*, 11(3), pp.283–318.

Lindeberg, T. (1993b). On scale selection for differential operators. *Proc. 8th scandinavian conf. on image analysis*. Citeseer. 1993.

Lindeberg, T. (1994). Scale-space theory: a basic tool for analyzing structures at different scales. *Journal of applied statistics*, 21(1-2), pp.225–270.

Lindeberg, T. (1998). Feature detection with automatic scale selection. *International journal of computer vision*, 30(2), pp.79–116.

Lowe, D. (1999). Object recognition from local scale-invariant features. *Computer vision, 1999. the proceedings of the seventh ieee international conference on*. Vol. 2. Ieee. 1999, pp.1150–1157.

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

Mahalanobis, P.C. (1936). On the generalised distance in statistics. *Proceedings of the National Institute of Sciences of India, 1936*, pp.49–55.

Mair, E., Hager, G.D., Burschka, D., Suppa, M. and Hirzinger, G. (2010). Adaptive and generic corner detection based on the accelerated segment test. *European conference on computer vision*. Springer. 2010, pp.183–196.

Mairal, J., Koniusz, P., Harchaoui, Z. and Schmid, C. (2014). Convolutional kernel networks. *Advances in neural information processing systems*. 2014, pp.2627–2635.

Malik, J. and Perona, P. (1990). Preattentive texture discrimination with early vision mechanisms. *JOSA A*, 7(5), pp.923–932.

Mallat, S.G. (1989). A theory for multiresolution signal decomposition: the wavelet representation. *IEEE transactions on pattern analysis and machine intelligence*, 11(7), pp.674–693.

Manning, C.D., Raghavan, P., Schütze, H. et al. (2008). *Introduction to information retrieval*. Vol. 1, 1. Cambridge university press Cambridge, 2008.

Marr, D. and Hildreth, E. (1980). Theory of edge detection. *Proceedings of the Royal Society of London B: Biological Sciences*, 207(1167), pp.187–217.

Matas, J., Chum, O., Urban, M. and Pajdla, T. (2004). Robust wide-baseline stereo from maximally stable extremal regions. *Image and vision computing*, 22(10), pp.761–767.

McCallum, A., Nigam, K. et al. (1998). A comparison of event models for naive bayes text classification. *Aaai-98 workshop on learning for text categorization*. Vol. 752. Madison, WI. 1998, pp.41–48.

McCulloch, W.S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4), pp.115–133.

Meinguet, J. (1979). Multivariate interpolation at arbitrary points made simple. *Zeitschrift für angewandte Mathematik und Physik (ZAMP)*, 30(2), pp.292–304.

Mikolajczyk, K. and Schmid, C. (2003). A performance evaluation of local descriptors. *In proceedings of the conference on computer vision and pattern recognition*. Citeseer. 2003.

Mikolajczyk, K. and Schmid, C. (2002). An affine invariant interest point detector. *Computer Vision—ECCV 2002*, pp.128–142.

Mikolajczyk, K. and Schmid, C. (2004). Scale & affine invariant interest point detectors. *International journal of computer vision*, 60(1), pp.63–86.

Mikolajczyk, K. and Schmid, C. (2005). A performance evaluation of local descriptors. *IEEE transactions on pattern analysis and machine intelligence*, 27(10), pp.1615–1630.

Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T. and Van Gool, L. (2005). A comparison of affine region detectors. *International journal of computer vision*, 65(1-2), pp.43–72.

Mobahi, H., Collobert, R. and Weston, J. (2009). Deep learning from temporal coherence in video. *Proceedings of the 26th annual international conference on machine learning*. ACM. 2009, pp.737–744.

Mohan, A., Papageorgiou, C. and Poggio, T. (2001). Example-based object detection in images by components. *IEEE transactions on pattern analysis and machine intelligence*, 23(4), pp.349–361.

Moo Yi, K., Verdie, Y., Fua, P. and Lepetit, V. (2016). Learning to assign orientations to feature points. *Proceedings of the ieee conference on computer vision and pattern recognition*. 2016, pp.107–116.

Moravec, H.P. (1980). *Obstacle avoidance and navigation in the real world by a seeing robot rover*. DTIC Document, 1980, (tech. rep.).

Mrazova, I. and Kukacka, M. (2008). Hybrid convolutional neural networks. *2008 6th ieee international conference on industrial informatics*. IEEE. 2008, pp.469–474.

Murphy, K. (2012). *Machine learning: a probabilistic approach*. MIT Press, 2012.

Mutch, J. and Lowe, D. (2006). Multiclass object recognition with sparse, localized features. *Computer vision and pattern recognition, 2006 ieee computer society conference on*. Vol. 1. IEEE. 2006, pp.11–18.

Nair, V. and Hinton, G. (2010). Rectified linear units improve restricted boltzmann machines. *Proceedings ofs the 27th international conference on machine learning (icml-10)*. 2010, pp.807–814.

Nguyen, A., Yosinski, J. and Clune, J. (2015). Deep neural networks are easily fooled: high confidence predictions for unrecognizable images. *Proceedings of the ieee conference on computer vision and pattern recognition*. 2015, pp.427–436.

Nigam, K., Lafferty, J. and McCallum, A. (1999). Using maximum entropy for text classification. *Ijcai-99 workshop on machine learning for information filtering*. Vol. 1. 1999, pp.61–67.

Nilsback, M.-E. and Zisserman, A. (2006). A visual vocabulary for flower classification. *Cvpr*. Vol. 2. IEEE. 2006, pp.1447–1454.

Nister, D. and Stewenius, H. (2006). Scalable recognition with a vocabulary tree. *Computer vision and pattern recognition, 2006 ieee computer society conference on*. Vol. 2. Ieee. 2006, pp.2161–2168.

Ojala, T., Rautiainen, M., Matinmikko, E. and Aittola, M. (2001). Semantic image retrieval with hsv correlograms. *Proceedings scandinavian conference on image analysis*. Citeseer. 2001, pp.621–627.

Orguner, U. and Gustafsson, F. (2007). Statistical characteristics of harris corner detector. *Statistical signal processing, 2007. ssp'07. ieee/sp 14th workshop on*. IEEE. 2007, pp.571–575.

Oxford English Dictionary (2018). *"invariance, n."* Oxford Univesity Press. 2018.

Ozuysal, M., Calonder, M., Lepetit, V. and Fua, P. (2010). Fast keypoint recognition using random ferns. *IEEE transactions on pattern analysis and machine intelligence*, 32(3), pp.448–461.

Papageorgiou, C. and Poggio, T. (2000). A trainable system for object detection. *International Journal of Computer Vision*, 38(1), pp.15–33.

Paulin, M., Douze, M., Harchaoui, Z., Mairal, J., Perronin, F. and Schmid, C. (2015). Local convolutional features with unsupervised training for image retrieval. *Proceedings of the ieee international conference on computer vision*. 2015, pp.91–99.

Philbin, J., Chum, O., Isard, M., Sivic, J. and Zisserman, A. (2007). Object retrieval with large vocabularies and fast spatial matching. *Cvpr*. 2007.

Pingle, K.K. and Thomas, A.J. (1975). A fast, feature-driven stereo depth program.

Porikli, F. (2005). Integral histogram: a fast way to extract histograms in cartesian spaces. *Computer vision and pattern recognition, 2005. cvpr 2005. ieee computer society conference on*. Vol. 1. IEEE. 2005, pp.829–836.

Press, W., Flannery, B., Teukolsky, S. and Vetterling (1992). *Numerical recipies in c: the art of scientific computing*. Cambridge: Cambridge University Press, 1992.

Rabiner, L.R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2), pp.257–286.

Redmon, J., Divvala, S., Girshick, R. and Farhadi, A. (2016). You only look once: unified, real-time object detection. *Proceedings of the ieee conference on computer vision and pattern recognition*. 2016, pp.779–788.

Ren, S., He, K., Girshick, R. and Sun, J. (2015). Faster r-cnn: towards real-time object detection with region proposal networks. *Advances in neural information processing systems*. 2015, pp.91–99.

Roberts, L.G. (1963). *Machine perception of three-dimensional solids*. PhD thesis. Massachusetts Institute of Technology, 1963.

Romera-Paredes, B. and Torr, P.H.S. (2016). Recurrent instance segmentation. *European conference on computer vision*. Springer. 2016, pp.312–329.

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6), p.386.

Rosenfeld, A. (1969). Picture processing by computer. *ACM Computing Surveys (CSUR)*, 1(3), pp.147–176.

Rosin, P.L. (1999). Measuring corner properties. *Computer Vision and Image Understanding*, 73(2), pp.291–307.

Rosten, E. and Drummond, T. (2005). Fusing points and lines for high performance tracking. *Computer vision, 2005. iccv 2005. tenth ieee international conference on*. Vol. 2. IEEE. 2005, pp.1508–1515.

Rosten, E. and Drummond, T. (2006). Machine learning for high-speed corner detection. *Computer Vision–ECCV 2006*, pp.430–443.

Rosten, E., Porter, R. and Drummond, T. (2010). Faster and better: a machine learning approach to corner detection. *IEEE transactions on pattern analysis and machine intelligence*, 32(1), pp.105–119.

Rosten, E., Reitmayr, G. and Drummond, T. (2005). Real-time video annotations for augmented reality. *Advances in Visual Computing*, pp.294–302.

Rublee, E., Rabaud, V., Konolige, K. and Bradski, G. (2011). Orb: an efficient alternative to sift or surf. *Computer vision (iccv), 2011 ieee international conference on*. IEEE. 2011, pp.2564–2571.

Rubner, Y., Tomasi, C. and Guibas, L.J. (2000). The earth mover's distance as a metric for image retrieval. *International journal of computer vision*, 40(2), pp.99–121.

Rui, Y., Huang, T.S. and Chang, S.-F. (1999). Image retrieval: current techniques, promising directions, and open issues. *Journal of visual communication and image representation*, 10(1), pp.39–62.

Rumelhart, D. and Hinton, G. (1986). Learning representations by back-propagating errors. *NATURE*, 323, p.9.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M. et al. (2015). Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3), pp.211–252.

Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M. and Monfardini, G. (2009). The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1), pp.61–80.

Schapire, R.E. (1990). The strength of weak learnability. *Machine learning*, 5(2), pp.197–227.

Schapire, R.E. and Singer, Y. (1999). Improved boosting algorithms using confidence-rated predictions. *Machine learning*, 37(3), pp.297–336.

Scherer, D., Müller, A. and Behnke, S. (2010). Evaluation of pooling operations in convolutional architectures for object recognition. *International conference on artificial neural networks*. Springer. 2010, pp.92–101.

Schmid, C. (1996). *Appariement d'images par invariants locaux de niveaux de gris. application à l'indexation d'une base d'objets*. PhD thesis. Institut National Polytechnique de Grenoble-INPG, 1996.

Schmid, C. and Mohr, R. (1997). Local grayvalue invariants for image retrieval. *IEEE transactions on pattern analysis and machine intelligence*, 19(5), pp.530–535.

Schmidhuber, J. (2015). Deep learning in neural networks: an overview. *Neural Networks*, 61, pp.85–117.

Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R. and LeCun, Y. (2014). Overfeat: integrated recognition, localization and detection using convolutional networks. *International conference on learning representations (iclr2014), cbls, april 2014*. 2014.

Shapiro, L.G. and Haralick, R.M. (1981). Structural descriptions and inexact matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (5), pp.504–519.

Shechtman, E. and Irani, M. (2007). Matching local self-similarities across images and videos. *Computer vision and pattern recognition, 2007. cvpr'07. ieee conference on*. IEEE. 2007, pp.1–8.

Shi, J. and Tomasi, C. (1994). Good features to track. *Computer vision and pattern recognition, 1994. proceedings cvpr'94., 1994 ieee computer society conference*. IEEE. 1994, pp.593–600.

Shi, X. and Manduchi, R. (2004). Invariant operators, small samples, and the bias-variance dilemma. *Computer vision and pattern recognition, 2004. cvpr 2004. proceedings of the 2004 ieee computer society conference on*. Vol. 2. IEEE. 2004, pp.II–II.

Shirley, P. and Marschner, S. (2009). Fundamentals of computer graphics.

Simard, P., LeCun, Y. and Denker, J.S. (1993). Efficient pattern recognition using a new transformation distance. *Advances in neural information processing systems*, pp.50–50.

Simo-Serra, E., Trulls, E., Ferraz, L., Kokkinos, I., Fua, P. and Moreno-Noguer, F. (2015). Discriminative learning of deep convolutional feature point descriptors. *Proceedings of the ieee international conference on computer vision*. 2015, pp.118–126.

Simon, M. and Rodner, E. (2015). Neural activation constellations: unsupervised part model discovery with convolutional networks. *Proceedings of the ieee international conference on computer vision*. 2015, pp.1143–1151.

Simonyan, K., Vedaldi, A. and Zisserman, A. (2012). Descriptor learning using convex optimisation. *Computer vision–eccv 2012*. Springer, 2012, pp.243–256.

Simonyan, K., Vedaldi, A. and Zisserman, A. (2014). Learning local feature descriptors using convex optimisation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(8), pp.1573–1585.

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. *International Conference on Learning Representations*.

Sivic, J. and Zisserman, A. (2003). Video google: a text retrieval approach to object matching in videos. *Null*. IEEE. 2003, p.1470.

Sivic, J. and Zisserman, A. (2006). Video google: efficient visual search of videos. *Toward category-level object recognition*. Springer, 2006, pp.127–144.

Smith, J.R. and Chang, S.-F. (1995). Single color extraction and image query. *Image processing, 1995. proceedings., international conference on*. Vol. 3. IEEE. 1995, pp.528–531.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1), pp.1929–1958.

Strecha, C., Fransens, R. and Van Gool, L. (2006). Combined depth and outlier estimation in multi-view stereo. *Computer vision and pattern recognition, 2006 ieee computer society conference on*. Vol. 2. IEEE. 2006, pp.2394–2401.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. and Rabinovich, A. (2015). Going deeper with convolutions. CVPR. 2015.

Szegedy, C., Toshev, A. and Erhan, D. (2013). Deep neural networks for object detection. *Advances in neural information processing systems*. 2013, pp.2553–2561.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. and Wojna, Z. (2016). Rethinking the inception architecture for computer vision. *Proceedings of the ieee conference on computer vision and pattern recognition*. 2016, pp.2818–2826.

Szeliski, R. (2010). *Computer vision: algorithms and applications*. 2nd ed. Springer Science & Business Media, 2010.

Terrillon, J.-C., David, M. and Akamatsu, S. (1998). Automatic detection of human faces in natural scene images by use of a skin color model and of invariant moments. *Automatic face and gesture recognition, 1998. proceedings. third ieee international conference on*. IEEE. 1998, pp.112–117.

Thompson, D.W. (1942). On growth and form. *On growth and form.*

Tola, E., Lepetit, V. and Fua, P. (2008). A fast local descriptor for dense matching. *Computer vision and pattern recognition, 2008. cvpr 2008. ieee conference on*. IEEE. 2008, pp.1–8.

Tola, E., Lepetit, V. and Fua, P. (2010). Daisy: an efficient dense descriptor applied to wide-baseline stereo. *IEEE transactions on pattern analysis and machine intelligence*, 32(5), pp.815–830.

Tombari, F., Franchi, A. and Di Stefano, L. (2013). Bold features to detect texture-less objects. *Proceedings of the ieee international conference on computer vision*. 2013, pp.1265–1272.

Triggs, B. (2004). Detecting keypoints with stable position, orientation, and scale under illumination changes. *European Conference on Computer Vision-ECCV 2004*, pp.100–113.

Trzcinski, T., Christoudias, M., Fua, P. and Lepetit, V. (2013). Boosting binary keypoint descriptors. *Proceedings of the ieee conference on computer vision and pattern recognition*. 2013, pp.2874–2881.

Trzcinski, T., Christoudias, M. and Lepetit, V. (2015). Learning image descriptors with boosting. *IEEE transactions on pattern analysis and machine intelligence*, 37(3), pp.597–610.

Trzcinski, T., Christoudias, M., Lepetit, V. and Fua, P. (2012). Learning image descriptors with the boosting-trick. *Advances in neural information processing systems*. 2012, pp.269–277.

Trzcinski, T. and Lepetit, V. (2012). Efficient discriminative projections for compact binary descriptors. *European conference on computer vision*. Springer. 2012, pp.228–242.

Tuytelaars, T. and Mikolajczyk, K. (2008). Local invariant feature detectors: a survey. *Foundations and trends® in computer graphics and vision*, 3(3), pp.177–280.

Tuytelaars, T. and Schmid, C. (2007). Vector quantizing feature space with a regular lattice. *Computer vision, 2007. iccv 2007. ieee 11th international conference on*. IEEE. 2007, pp.1–8.

Uijlings, J., Sande, K. van de, Gevers, T. and Smeulders, A. (2013). Selective search for object recognition. *International journal of computer vision*, 104(2), pp.154–171.

Uzyıldırım, F.E. and Özuysal, M. (2016). Instance detection by keypoint matching beyond the nearest neighbor. *Signal, Image and Video Processing*, 10(8), pp.1527–1534.

Vaillant, R., Monrocq, C. and LeCun, Y. (1994). Original approach for the localisation of objects in images. *IEE Proceedings-Vision, Image and Signal Processing*, 141(4), pp.245–250.

Valiant, L.G. (1984). A theory of the learnable. *Communications of the ACM*, 27(11), pp.1134–1142.

Van De Sande, K., Gevers, T. and Snoek, C. (2010). Evaluating color descriptors for object and scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 32(9), pp.1582–1596.

Varma, M. and Babu, B.R. (2009). More generality in efficient multiple kernel learning. *Proceedings of the 26th annual international conference on machine learning*. ACM. 2009, pp.1065–1072.

Varma, M. and Ray, D. (2007). Learning the discriminative power-invariance trade-off. *Computer vision, 2007. iccv 2007. ieee 11th international conference on*. IEEE. 2007, pp.1–8.

Viola, P. and Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. *Cvpr*. Vol. 1. IEEE. 2001, pp.I–511.

Von der Malsburg, C. (1973). Self-organization of orientation sensitive cells in the striate cortex. *Kybernetik*, 14(2), pp.85–100.

Von Gioi, R.G., Jakubowicz, J., Morel, J.-M. and Randall, G. (2010). Lsd: a fast line segment detector with a false detection control. *IEEE transactions on pattern analysis and machine intelligence*, 32(4), pp.722–732.

Wang, X., Yang, M., Zhu, S. and Lin, Y. (2013). Regionlets for generic object detection. *Proceedings of the ieee international conference on computer vision*. 2013, pp.17–24.

Weber, M., Welling, M. and Perona, P. (2000a). Towards automatic discovery of object categories. *Computer vision and pattern recognition, 2000. proceedings. ieee conference on*. Vol. 2. IEEE. 2000, pp.101–108.

Weber, M., Welling, M. and Perona, P. (2000b). Unsupervised learning of models for recognition. *Computer Vision-ECCV 2000*, pp.18–32.

Weinberger, K.Q. and Saul, L.K. (2009). Distance metric learning for large margin nearest neighbor classification. *JMLR*, 10, pp.207–244.

Weiss, S. and Kulikowski, C. (1991). Computer systems that learn.

Weng, J., Ahuja, N. and Huang, T.S. (1992). Matching two perspective views. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(8), pp.806–825.

Westlake, N., Cai, H. and Hall, P. (2016). Detecting people in artwork with cnns. In: G. Hua and H. Jégou, eds. *Computer vision – ECCV 2016 workshops*. Springer, 2016, pp.825–841.

Willats, J. (1997). *Art and representation: new principles in the analysis of pictures*. Princeton University Press, 1997.

Winder, S. and Brown, M. (2007). Learning local image descriptors. *Computer vision and pattern recognition, 2007. cvpr'07. ieee conference on*. IEEE. 2007, pp.1–8.

Winder, S., Hua, G. and Brown, M. (2009). Picking the best daisy. *Cvpr*. IEEE. 2009, pp.178–185.

Witkin, A. (1984). Scale-space filtering: a new approach to multi-scale description. *Acoustics, speech, and signal processing, ieee international conference on icassp'84*. Vol. 9. IEEE. 1984, pp.150–153.

Wolpert, D.H. (1996). The lack of a priori distinctions between learning algorithms. *Neural computation*, 8(7), pp.1341–1390.

Wu, Q., Cai, H. and Hall, P. (2014). Learning graphs to model visual objects across different depictive styles. *Computer vision–eccv 2014*. Springer, 2014, pp.313–328.

Xiao, B., Song, Y.-Z., Balika, A. and Hall, P.M. (2008). Structure is a visual class invariant. *Joint iapr international workshops on statistical techniques in pattern recognition (spr) and structural and syntactic pattern recognition (sspr)*. Springer. 2008, pp.329–338.

Xiao, B., Yi-Zhe, S. and Hall, P. (2011). Learning invariant structure for object identification by using graph methods. *Computer Vision and Image Understanding*, 115(7), pp.1023–1031.

Xiao, L. (2010). Dual averaging methods for regularized stochastic learning and online optimization. *JMLR*, 11, pp.2543–2596.

Yang, B., Yan, J., Lei, Z. and Li, S.Z. (2016). Craft objects from images. *Computer vision and pattern recognition (cvpr), 2016 ieee conference on*. IEEE. 2016, pp.6043–6051.

Yi, K.M., Trulls, E., Lepetit, V. and Fua, P. (2016). Lift: learned invariant feature transform. *European conference on computer vision*. Springer. 2016, pp.467–483.

Yosinski, J., Clune, J., Bengio, Y. and Lipson, H. (2014). How transferable are features in deep neural networks? *Advances in neural information processing systems*. 2014, pp.3320–3328.

Young, T. (1802). The bakerian lecture: on the theory of light and colours. *Philosophical transactions of the Royal Society of London*, 92, pp.12–48.

Zagoruyko, S. and Komodakis, N. (2015). Learning to compare image patches via convolutional neural networks. *Proceedings of the ieee conference on computer vision and pattern recognition*. 2015, pp.4353–4361.

Zeiler, M. and Fergus, R. (2014). Visualizing and understanding convolutional networks. *European conference on computer vision*. Springer. 2014, pp.818–833.

Zeisl, B., Georgel, P., Schweiger, F., Steinbach, E., Navab, N. and Munich, G. (2009). Estimation of location uncertainty for scale invariant feature points. *Proceedings of the british machine vision conference*. 2009.

Zhang, J., Marszałek, M., Lazebnik, S. and Schmid, C. (2007). Local features and kernels for classification of texture and object categories: a comprehensive study. *IJCV*, 73(2), pp.213–238.

Zhang, Z., Deriche, R., Faugeras, O. and Luong, Q.-T. (1995). A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. *Artificial intelligence*, 78(1-2), pp.87–119.

Zheng, F. and Webb, G.I. (2005). A comparative study of semi-naive bayes methods in classification learning. *Proceedings of the fourth australasian data mining conference (ausdm05)*. 2005, pp.141–156.