Universitat Autònoma de Barcelona

Semantic Representation: From Color to Deep Embeddings

Lu Yu

Universitat Autònoma
de Barcelona

Semantic Representation: From Color to Deep Embeddings

A dissertation submitted by **Lu Yu** to the Universitat Autònoma de Barcelona in fulfilment of the degree of **Doctor of Philosophy** in the Departament de Ciències de la Computació.

Bellaterra, September 12, 2019

Director	**Dr. Joost van de Weijer** Centre de Visió per Computador Universitat Autònoma de Barcelona **Dr. Yongmei Cheng** School of Automation Northwestern Polytechnical University
Thesis Committee	**Dr. Zhunga Liu** School of Automation Northwestern Polytechnical University **Dr. Dingwen Zhang** Mechanical and Electrical Engineering Xidian University **Dr. Yang Yang** Institute of Automation Chinese Academy of Sciences **Dr. Xiaoxu Wang** School of Automation Northwestern Polytechnical University **Dr. Andrew D. Bagdanov** Department of Information Engineering University of Florence

This document was typeset by the author using LaTeX 2$_\varepsilon$.

ISBN: 978-84-945373-4-9

Printed by Ediciones Gráficas Rey, S.L.

Acknowledgements

This Ph.D adventure would not have been possible if it was not for the help, contribution, support and love of many distinct individuals. I would like to express my deepest appreciation to all those who assisted me to complete my Ph.D.

First and foremost, I would like to express my profound gratitude to my supervisor Prof. Joost van de Weijer. You spared no effort to supervise me, encourage me, support me with the greatest patience and responsibility during these four years, from when I had little knowledge about computer vision to the conclusion of this journey. You taught me how to do research, write papers and do presentations from the countless academic meetings and emails. The more important thing I learnt is the way you treat work, students, family and life, which will last for lifetime. Hartelijk bedankt!

I would also like to thank my Chinese supervisor Prof.Yongmei Cheng for her great support and guidance in Northwestern Polytechnical University. Thank you for your help in research. Your constructive suggestions and encouraging comments that considerably helped me to improve my research. I would also like to thank people in Xi'an. Thanks to for your encouragement and help: Haoyue, Xi, Ya, Mingyan, Xuemei, Yanru, Nan, Huaxia, Hucheng, Lanyue, Liping, Wei, Zhe, Jiantao, Jianxin, Xu, Shuai... The good memories in Xi'an would stay with me forever.

Thanks to all great LAMPers for their kind support. I appreciate the guidance and help from Luis, Andy, Abel, Bogdan and Mikhail. My sincere thanks to Marc, Yaxing, Laura, Aymen, Oguz, Lichao, Fei, Chenshen, Kai, Mikel, Olaia, Carola, Aitor, Javad. Many thanks to all the CVC people for welcoming me with open arms: Dena, Yi, Cori, Zhijie, Lei, Arash, Montse, Gigi and all the people in administration... I've enjoyed from collaborating with Prof. Fahad Shahbaz Khan, Prof. C. Alejandro Parraga and Dr. Arnau Ramisa. Thanks to my dearest friends in Barcelona, Fangchang, Tingting, Xiaoqing, Rong, Yi, Jianqiang, Yu, Junpeng, we have spent much good times together. My years in Barcelona have been priceless due to all those amazing people.

Last but not the least, I would like to thank my family for their endless support and care. Thanks to my dear parents, my lovely brother. You never give me any pres-

sure in my study, keep encouraging me, caring about me, giving endless warmness and happiness to me. You are my forever shelter. I couldn't love you all more.

Finally, to my love Xialei. I've been so lucky to meet you. I am so delighted we are sharing this journey together in these seven years. This is the biggest romance for me. Without your inspiration, I would not have been able to finish my Ph.D, both scientifically and emotionally. Thanks!

Abstract

One of the fundamental problems of computer vision is to represent images with compact semantically relevant embeddings. These embeddings could then be used in a wide variety of applications, such as image retrieval, object detection, and video search. The main objective of this thesis is to study image embeddings from two aspects: color embeddings and deep embeddings.

In the first part of the thesis we start from hand-crafted color embeddings. We propose a method to order the additional color names according to their complementary nature with the basic eleven color names. This allows us to compute color name representations with high discriminative power of arbitrary length. Psychophysical experiments confirm that our proposed method outperforms baseline approaches. Secondly, we learn deep color embeddings from weakly labeled data by adding an attention strategy. The attention branch is able to correctly identify the relevant regions for each class. The advantage of our approach is that it can learn color names for specific domains for which no pixel-wise labels exists.

In the second part of the thesis, we focus on deep embeddings. Firstly, we address the problem of compressing large embedding networks into small networks, while maintaining similar performance. We propose to distillate the metrics from a teacher network to a student network. Two new losses are introduced to model the communication of a deep teacher network to a small student network: one based on an absolute teacher, where the student aims to produce the same embeddings as the teacher, and one based on a relative teacher, where the distances between pairs of data points is communicated from the teacher to the student. In addition, various aspects of distillation have been investigated for embeddings, including hint and attention layers, semi-supervised learning and cross quality distillation. Finally, another aspect of deep metric learning, namely lifelong learning, is studied. We observed some drift occurs during training of new tasks for metric learning. A method to estimate the semantic drift based on the drift which is experienced by data of the current task during its training is introduced. Having this estimation, previous tasks can be compensated for this drift, thereby improving their performance. Furthermore, we show that embedding networks suffer significantly less from catastrophic forgetting compared to classification networks when learning new tasks.

Key words: *computer vision, color representation, weakly supervised learning, neuron networks, network distillation, lifelong learning*

Resumen

Uno de los problemas fundamentales de la visión por computador es representar imágenes con descripciones compactas semánticamente relevantes. Estas descripciones podrían utilizarse en una amplia variedad de aplicaciones, como la comparación de imágenes, la detección de objetos y la búsqueda de vídeos. El objetivo principal de esta tesis es estudiar las representaciones de imágenes desde dos aspectos: las descripciones de color y las descripciones profundas con redes neuronales.

En la primera parte de la tesis partimos de descripciones de color modeladas a mano. Existen nombres comunes en varias lenguas para los colores básicos, y proponemos un método para extender los nombres de colores adicionales de acuerdo con su naturaleza complementaria a los básicos. Esto nos permite calcular representaciones de nombres de colores de longitud arbitraria con un alto poder discriminatorio. Los experimentos psicofísicos confirman que el método propuesto supera a los marcos de referencia existentes. En segundo lugar, al agregar estrategias de atención, aprendemos descripciones de colores profundos con redes neuronales a partir de datos con anotaciones para la imagen en vez de para cada uno de los píxeles. La estrategia de atención logra identificar correctamente las regiones relevantes para cada clase que queremos evaluar. La ventaja del enfoque propuesto es que los nombres de colores a usar se pueden aprender específicamente para dominios de los que no existen anotaciones a nivel de píxel.

En la segunda parte de la tesis, nos centramos en las descripciones profundas con redes neuronales. En primer lugar, abordamos el problema de comprimir grandes redes de descriptores en redes más pequeñas, manteniendo un rendimiento similar. Proponemos destilar las métricas de una red maestro a una red estudiante. Se introducen dos nuevas funciones de coste para modelar la comunicación de la red maestro a una red estudiante más pequeña: una basada en un maestro absoluto, donde el estudiante pretende producir los mismos descriptores que el maestro, y otra basada en un maestro relativo, donde las distancias entre pares de puntos de datos son comunicadas del maestro al alumno. Además, se han investigado diversos aspectos de la destilación para las representaciones, incluidas las capas de atención, el aprendizaje semi-supervisado y la destilación de calidad cruzada.

Finalmente, se estudia otro aspecto del aprendizaje por métrica profundo, el aprendizaje continuado. Observamos que se produce una variación del conocimiento aprendido durante el entrenamiento de nuevas tareas. En esta tesis se presenta un método para estimar la variación semántica en función de la variación que experimentan los datos de la tarea actual durante su aprendizaje. Teniendo en cuenta esta estimación, las tareas anteriores pueden ser compensadas, mejorando así su rendimiento. Además, mostramos que las redes de descripciones profundas sufren significativamente menos olvidos catastróficos en comparación con las redes de clasificación cuando aprenden nuevas tareas.

Palabras clave: *visión por computador, representación del color, aprendizaje semi-supervisado, redes neuronales, destilación de redes neuronales, aprendizaje continuado.*

Resum

Un dels problemes fonamentals de la visió per computador és representar imatges amb descripcions compactes semànticament rellevants. Aquestes descripcions podrien utilitzar-se en una àmplia varietat d'aplicacions, com la comparació d'imatges, la detecció d'objectes i la cerca de vídeos. L'objectiu principal d'aquesta tesi és estudiar les representacions d'imatges des de dos aspectes: les descripcions de color i les descripcions profundes amb xarxes neuronals.

A la primera part de la tesi partim de descripcions de color modelades a mà. Existeixen noms comuns en diverses llengües per als colors bàsics, i proposem un mètode per estendre els noms de colors addicionals d'acord amb la seva naturalesa complementària als bàsics. Això ens permet calcular representacions de noms de colors de longitud arbitrària amb un alt poder discriminatori. Els experiments psicofísics confirmen que el mètode proposat supera els marcs de referència existents. En segon lloc, en agregar estratègies d'atenció, aprenem descripcions de colors profundes amb xarxes neuronals a partir de dades amb anotacions per a la imatge, en comptes de per a cada un dels píxels. L'estratègia d'atenció aconsegueix identificar correctament les regions rellevants per a cada classe que volem avaluar. L'avantatge de l'enfocament proposat és que els noms de colors a utilitzar es poden aprendre específicament per a dominis dels que no existeixen anotacions a nivell de píxel.

A la segona part de la tesi, ens centrem en les descripcions profundes amb xarxes neuronals. En primer lloc, abordem el problema de comprimir grans xarxes de descriptors en xarxes més petites, mantenint un rendiment similar. Proposem destil·lar les mètriques d'una xarxa mestre a una xarxa estudiant. S'introdueixen dues noves funcions de cost per a modelar la comunicació de la xarxa mestre a una xarxa estudiant més petita: una basada en un mestre absolut, on l'estudiant pretén produir els mateixos descriptors que el mestre, i una altra basada en un mestre relatiu, on les distàncies entre parells de punts de dades són comunicades del mestre a l'alumne. A més, s'han investigat diversos aspectes de la destil·lació per a les representacions, incloses les capes d'atenció, l'aprenentatge semi-supervisat i la destil·lació de qualitat creuada.

Finalment, s'estudia un altre aspecte de l'aprenentatge per mètrica profund,

l'aprenentatge continuat. Observem que es produeix una variació del coneixement après durant l'entrenament de noves tasques. En aquesta tesi es presenta un mètode per estimar la variació semàntica en funció de la variació que experimenten les dades de la tasca actual durant el seu aprenentatge. Tenint en compte aquesta estimació, les tasques anteriors poden ser compensades, millorant així el seu rendiment. A més, mostrem que les xarxes de descripcions profundes pateixen significativament menys oblits catastròfics en comparació amb les xarxes de classificació quan aprenen noves tasques.

Paraules clau: *visió per computador, representació del color, aprenentatge semisupervisat, xarxes neuronals, destil·lació de xarxes neuronals, aprenentatge continuat.*

Contents

Abstract **iii**

List of figures **xv**

List of tables **xxi**

1 Introduction **1**

1.1 Image embeddings 1

1.1.1 Definition 1

1.1.2 Deep image embeddings 3

1.1.3 Applications 4

1.2 Color embeddings and Deep embeddings 6

1.2.1 Color embeddings 6

1.2.2 Deep embeddings 9

1.3 Objectives and Approach 13

1.3.1 Color embeddings 13

1.3.2 Deep embeddings 14

I Color Embeddings **15**

2 Beyond Eleven Color Names for Image Understanding **17**

2.1 Introduction 17

2.2 Related Work 18

2.3 The augmented color name dataset 20

2.4 Ranking of Additional Color Names 21

2.4.1 Computation of color name mappings 22

2.4.2 Extending the color names set 23

2.4.3 Ranking additional color names 24

2.5 Experimental results 26

2.5.1 Color Naming 27

2.5.2 Image classification 27

2.5.3 Color naming for tracking 31

2.5.4 Color naming for re-identification 34

2.5.5 User preference experiment 35

2.6 Discussion and Conclusions 37

3 Weakly Supervised Domain-Specific Color Naming Based on Attention **39**

3.1 Introduction 39

3.2 Attention Modulation for Color Naming 40

3.2.1 Color Naming Network (CN-CNN) 41

3.2.2 Visual Attention Network (VA-CNN) 42

3.2.3 Modulation Layer 43

3.2.4 Network Training 44
3.3 Color Naming Data Collection 44
3.4 Experiments 45
3.4.1 Implementation Details 45
3.4.2 Color Naming from Weakly Labeled Data 46
3.4.3 Domain-Specific Color Naming 48
3.5 Conclusions 49

II Deep Embeddings 51

4 Learning Metrics from Teachers: Compact Networks for Image Embedding 53

4.1 Introduction 53
4.2 Related work 54
4.3 Preliminaries 56
4.3.1 Metric Learning 56
4.3.2 Network Distillation 57
4.4 Distillation for Metric Learning 58
4.4.1 Knowledge distillation for embedding networks 58
4.4.2 Learning from hints and attention 60
4.5 Experimental Results 62
4.5.1 Retrieval on Fine-grained Datasets 62
4.5.2 Semi-Supervised Learning 65
4.5.3 Very Small Student Networks 67

4.5.4 Cross Quality Distillation 67
4.6 Conclusions 69

5 Semantic Drift Compensation for Lifelong Learning of Embeddings 71
5.1 Introduction 71
5.2 Related Work 73
5.2.1 Learning Embeddings 73
5.2.2 Lifelong Learning 73
5.3 Lifelong Learning for Embedding Networks 74
5.3.1 Embedding Networks 75
5.3.2 Preventing Forgetting 76
5.4 Semantic Drift Compensation 78
5.4.1 Computation of the Semantic Drift 78
5.4.2 Combining preventing forgetting and drift compensation . . . 79
5.5 Experiments 81
5.5.1 MNIST 83
5.5.2 Within-domain continual learning 84
5.5.3 Cross-domain continual learning 86
5.6 Conclusions 87

6 Conclusions and Future Work 89
6.1 Conclusions 89
6.2 Future work 91

Publications 93

Bibliography **114**

List of Figures

1.1 Illustration of input images projecting to the embedding space. The similar inputs stay closer and dissimilar ones stay further in the embedding space. 2

1.2 Illustration of metric learning applied to a face recognition task. The faces of the same person stay closer than the others'. In the left image, red lines indicate the negative pairs (the different persons) and green ones indicate the positive pairs (the same person). This figure is copied from [5]. 4

1.3 Comparison of contrastive loss and triple loss (based on the figure from [140]). 4

1.4 Example of image retrieval results. 5

1.5 The illustration of K-nearest neighbour (KNN). 5

1.6 Example image to show that color descriptor is affected by illuminance (red circle), specularities (blue rectangle) and shadow (black triangle). 6

1.7 Example of images by searching the query of 'color name + object' in Google Image. 7

1.8 (The left)sunflower drawn by Vincent van Gogh, (the right)the pixel-wise color name annotation by eleven basic color names. The color names are represented by their corresponding color. 8

1.9 Examples and corresponding masks from EBAY dataset. 9

1.10 Illustrations of network distillation for classification networks. Figure is from [203]. 10

1.11 Illustration of class-incremental learning, figure is from [135]. 11

2.1 Example images for the color ochre from the augmented color name dataset. 20

2.2 Comparisons of the top 36 retrieved Munsell patches given a color name. We compare results of our method and the naive method to extend the color name set. Results clearly show that the naive approach fails to retrieve all relevant Munsell patches. 21

2.3 (top row) The eleven basic color terms. (second and third row) proposed order in which to add 28 additional color names to the basic color term set. 25

2.4 (a) the original image and the assignment based on (b) the 11 color names mapping; (c) the 15 ranked color name mapping; and the (d) the 25 ranked color name mapping. 26

2.5 (a) example image from Ebay labeled with the color name 'green' and (b) the ground truth mask of the image identifying the pixels which are related with the color name. The results in Table 2.1 show the percentage of pixels on the mask which are labeled in agreement with the ground truth label. 28

2.6 (a) the original image and the assignment based on (b) PLSA; (c) SVM; and the (d) k-nearest neighbors on the 11 color name mapping. . . . 29

2.7 Example images from Flower102 dataset. 30

2.8 Classification accuracy on Oxford Flower102 comparing color names with discriminative color descriptors. 31

2.9 Success plots for (top) various different color name mappings, and (bottom) various compressed color name mappings. 32

2.10 Comparison of 3 different color name representations for trackers in challenging situations such as illumination variation, occlusion, motion blur and in-plane rotation. The example frames are from the Jogging, Soccer and Shaking sequences respectively. The results of 11D, 15D, and 25D are represented by blue, green and red boxes respectively. 33

2.11 Success plots for several attributes, including illumination variation, low resolution, motion blur and occlusion. The increased color name representation outperforms the original color name representation for these attributes. 34

2.12 Examples for top 3 ranking with 11 and 25 color terms. Note that some of the errors which occur when using 11 color names are resolved when using 25 color names. 35

2.13 Setup for our psychophysical experiment. Color patches and color names were presented on a calibrated CRT monitor and the observer pressed buttons on a gamepad to decide whether the color patch was well described by the name or not. The background was mid-grey and a reference white was provided by a D65-colored frame. 36

3.1 Example images of domain-specific color names: (a) 'champagne' colored horse, (b) 'almond' colored hair and (c) 'coral red' lips. 40

3.2 Overview of our proposed framework for weakly supervised color name prediction. Our model is capable of automatically discovering correct regions of interest for image-wise color label predicting and simultaneously providing an end-to-end mapping between color values and color names. 41

3.3 The structure of the color naming network (CN-CNN). 42

3.4 Examples of four categories ('car','dress','pottery' and 'shoes') in eleven basic color are shown. 44

3.5 Examples from domain-specific datasets. One example for each domain-specific color name is shown. 45

3.6 (a) Example image from EBAY labeled with the color name 'green' and (b) the ground truth mask of the image identifying the pixels which are related with the color name. 46

3.7 Examples of Attention map from Eye, Lip, Horse and Tomato datasets. 48

4.1 Graphical illustration of the two knowledge distillation losses we propose for metric learning. L_{KD}^{abs} aims to minimize the distance between the student and teacher embedding of the same image. L_{KD}^{rel} compares the distance in the embedding of the teacher between two images, with the distance of the same two images in the student embedding. It aims to make the two distances as similar as possible. 56

4.2 Illustration of difference between absolute and relative teacher. (left) Example of four data points in the embedding space of teacher. We consider two samples from two classes (indicated by square and star). (middle and right) show the absolute and relative loss for two student embeddings $S1$ and $S2$ (the teacher location of the points is given in dashed lines). The (right) embedding is preferable since it is exactly equal to the teacher (except for a translation). This is only appreciated by the relative teacher, whereas the absolute teacher assigns equal loss to both. 57

4.3 Schematics of teacher-student hint/attention transfer. 60

4.4 R@1 as a function of λ on CUB-200-2011 dataset. 65

4.5 Example images from two fine-grained datasets CUB-200-2011 and Cars-196 used in our experiments. The top row shows examples of high-quality images and the bottom row shows examples of the corresponding low-quality images. 66

5.1 T-SNE visualization of embedding space on CUB dataset. A, B, C indicate prototypes of task 1 after training task 1. The prototypes for six classes after training task 2 are also provided (A', B', C' for task 1 and D, E, F for task 2). 72

5.2 Illustration of semantic drift compensation. (a) Data and prototypes of three classes of task 1 after training task 1. (b) Data of task 2 after training task 1. (c) Drift of data of task 2 while training task 2. (d) This vector field is used to approximate the drift of the prototypes of task 1. 78

5.3 Examples of the drift vectors in the cases of E-FT (top) and E-EWC (bottom). (a) and (d) represent the embedding of 5 classes of task 1 after training task 1; (b) and (e) represent the embedding of another 5 classes of task 2 after training task 1; (c) and (f) show the embeddings of 10 classes of two tasks together. The saved prototypes of the previous task(indicated by round) are estimated to new positions (indicated by triangle) by our proposed SDC in the new model which is observed to be closer to the real mean (indicated by star). The dotted arrows are the SDC vectors. 80

5.4 Examples of the drift vectors in the cases of E-LwF (top) and E-MAS (bottom). 81

5.5 Results on CUB-200-2011 (top), Stanford 40 Actions (middle) and Flowers (bottom) in a four task scenario. The results show the average incremental accuracy. A table version is available in Table 5.1. 83

5.6 Comparison of distances to the original mean and the drift-compensated mean with E-FT, E-LwF, E-EWC and E-MAS on CUB-200-2011 dataset after learning four tasks. The x-axis is the distance between the embedding to the drift-compensated mean and the y-axis is the distance to the original mean. Points are colored for different tasks (blue, yellow, pink for respectively task 1,2,3). If the data point appears in the top left area it means that the drift-compensation has improved the location of the prototype. 85

List of Tables

2.1 Percentage of correctly classified pixels in the Ebay dataset for various classifiers. 27

2.2 Classification accuracy on Oxford Flower102 with different methods. 30

2.3 Re-id performance comparison of different color descriptors on the Mart-1501 dataset. The best results are obtained using 25 color names.[1] Yangyang's 16-d color features for re_identification with our experiment settings [189]. 36

2.4 User communication results on Munsell patches of 39 color names. . 37

2.5 User communication results on Munsell patches of 25 color names. . 37

3.1 Details of our network 42

3.2 Comparison of state-of-the-art methods, testing on the EBAY dataset, training with class-agnostic dataset and new class-specific dataset. We indicate with test type which methods are supervised (S) or unsupervised (U). 46

3.3 Comparison of our model learned using different components on the EBAY dataset. We abbreviate attention, centric information and alternating learning as AM, C, AL. 47

3.4 Color naming results on Eye, Lip, Horses and Tomato dataset respectively comparing to using classification network (pre-trained AlexNet). 47

4.1 Retrieval Performance on the CUB-200-2011 and Cars-196 dataset. 'ML':metric learning loss, 'hint':hint loss; 'AT': attention loss; KD (abs): absolute teacher loss; KD (rel):relative teacher loss. 62

4.2 Comparison on Stanford Online Products dataset. 63

4.3 Semi-supervised results on CUB-200-2011. 66

4.4 Parameter Comparison of Different Networks 67

4.5 Performance on CUB-200-2011 with MobileNet-0.25. 67

4.6 Cross quality results on the CUB-200-2011 and Cars-196 datasets with low resolution and unlocalized object degradations. 68

5.1 Results on CUB-200-2011, Stanford 40 Actions and Flowers in a four task scenario. The results show the average incremental accuracy. . . 82

5.2 Classification accuracy (%) for the two-tasks scenario on MNIST. Results are presented as accuracy on T1/T2 (avg). 83

5.3 Classification accuracy (%) for sequences of 2 tasks on cross-domain setting. 84

5.4 Classification accuracy (%) for sequences of 3 tasks. 86

1 Introduction

For humans, vision is one of the major senses for interacting with our environment. Understanding how the brain recognizes objects is a central challenge for understanding human vision, and for designing artificial vision systems. For a three-year-old child, his understanding of the world is already impressive. He is able to easily distinguish cats from dogs, hamburgers from stones, family members from others. Regardless of the change of the pose, the variance of the illumiance, the difference of the environment around, which produce a different pattern on the retina, he still recognizes the object without much effort. Objects in the real world usually do not suddenly change their identity, thus we acquire this ability to recognize an object, despite these variations.

Color is an fundamental aspect of understanding and describing the world around us. As such it is one of the important features for computer vision systems to understand visual data. Humans use color names to communicate colors in language routinely and seemingly without effort. Examples of color names are 'black', 'turquoise' and 'light blue'. Inspired by the usage of color names for humans, one methodology based on color names to color description has been studied.

To understand the visual world the human visual system uses 100 billion of neurons. In recent years deep learning has revolutionized computer vision with a methodology called deep learning, which is motivated on the human brain; similarly as the brain data is processed in several layers. During this process the incoming data is analyzed from low-level features to high-level features. Learning low-level features (like color) is not designed explicitly but now is implicitly done by training these algorithms in an end-to-end sense.

1.1 Image embeddings

1.1.1 Definition

Machine learning problems occur when a task is defined by a series of cases or examples rather than by predefined rules. Such problems are found in a wide variety of application domains, ranging from engineering applications in robotics and pattern recognition (speech, handwriting, face recognition), to Internet applications

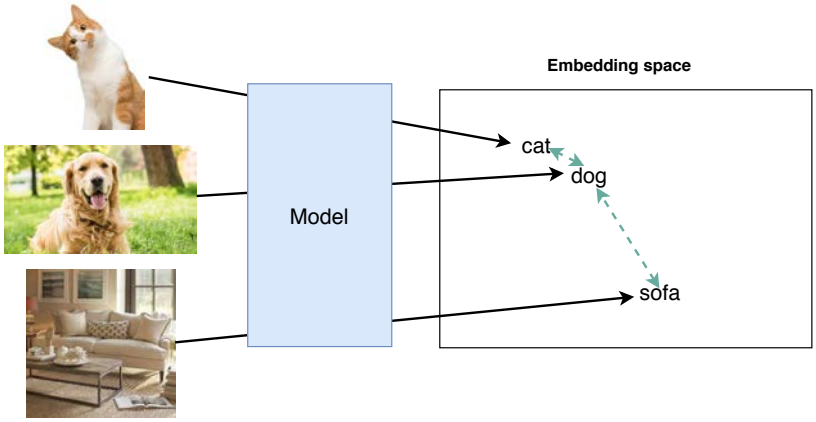


Figure 1.1 – Illustration of input images projecting to the embedding space. The similar inputs stay closer and dissimilar ones stay further in the embedding space.

(text categorization, recommendation system) and medical applications (diagnosis) [46]. *Feature extraction* is a key step in most machine learning mechanisms. It is a process of dimensionality reduction by which an initial set of raw data is reduced to high-level representations defined as *embedding*. Ideally, an embedding is able to capture some of the semantics of the input by placing semantically similar inputs close together despite significant visual differences such as point of view, illumination, or image quality in the embedding space [38, 177, 187]. Fig.1.1 shows that the input data are projected into the embedding space. The embeddings of cat and dog are placed relatively closer than that of sofa and dog. This is consistent with the visual similarity of the objects.

Finding a good data representation is very domain specific and related to available measurements. How the representations are organized largely conditions the success of the machine vision tasks. One can divide embedding methods into two groups:

1. handcrafted embeddings: Prior to deep neural networks for vision, computer vision systems relied on handcrafted features to create a bag-of-words-based vector representation for images and videos. One typical type is used to describe the distributions of the attributes like shape, color, texture, etc. Another type is transferred descriptor such as scale invariant feature transform (SIFT) [107], histogram of oriented gradients (HoG) [30], etc. The computation of handcrafted features, is normally a two-step process. First, the features

of an image x are extracted by the feature extractor $\phi(x)$, then the final hand-crafted embeddings $f(\phi(x))$ are learned due to a classifier such as Support Vector Machine(SVM) [28].

2. end-to-end deep embeddings: The deep learning paradigm enables the creation of complex networks for extracting the deep embeddings, where deep layers act as a set of feature extractors which are quite generic. In other word, deep learning is able to obtain the end-to-end deep embedings $f(x)$ directly from the input images x.

This thesis starts by investigating low-level embeddings and progressively shifts towards high-level semantic embeddings. It starts with hand-crafted embeddings for color naming, and proceeds to investigate deep color embeddings.

1.1.2 Deep image embeddings

One important class of deep networks learns feature embeddings which must preserve semantic similarity. These distance-based networks solve the task of Distance Metric Learning. Items deemed similar by users are supposed to be close in the embedding space. For example, face verification [17, 81, 126] is a problem of comparing two face images and determining whether or not they depict the same person, despite the differences in various attributes such as age, illumination, expression, and ethnicity. Fig.1.2 shows the example of faces in the embedding space. Pairs or triplets of similar and dissimilar items are used to teach the network how to organize the output embedding space. Embedding networks are essential for many computer vision fields, including image classification [113, 174], object recognition [45, 93], domain adaptation [44, 143].

An early success of metric learning using deep networks has been in the task of face verification [24]. In this work the authors introduce a Siamese architecture with a contrastive loss. Similar and dissimilar pairs of images are fed into the network, and a low dimension feature space is learned to map samples closer or further depending on their semantic relation (left of Fig.1.3). One limitation of this loss is that a single collapsing point is enforced for all images of the same class, which may be fine in certain cases (e.g. if the target is classification), but less so in others, where we may require a more nuanced embedding space. Triplet networks were proposed to address this limitation. They learn from triplets instead of pairs [58, 169]. An anchor image, a positive image and a negative image were passed through the network at the same time, and the aim of the network is to learn an embedding for which distance between positive pairs is smaller than distance between negative pairs (right of Fig.1.3). This allows more local modifications of the embedding space, and does not require that all the observations of the same class collapse to the same

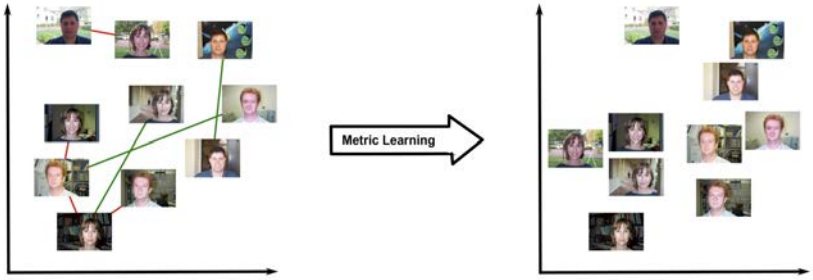


Figure 1.2 – Illustration of metric learning applied to a face recognition task. The faces of the same person stay closer than the others'. In the left image, red lines indicate the negative pairs (the different persons) and green ones indicate the positive pairs (the same person). This figure is copied from [5].

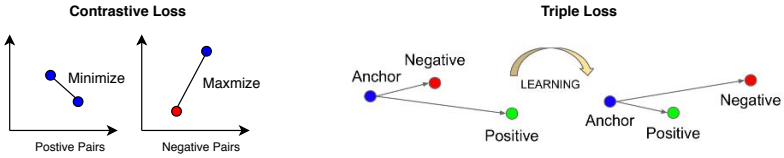


Figure 1.3 – Comparison of contrastive loss and triple loss (based on the figure from [140]).

point.

1.1.3 Applications

Embeddings have been intensively studied and gained significant interest in many fields. From the word embeddings [12, 87, 92],speech embedding [56, 66] to image embedding [1, 182], from low-level embedding [160, 196] to deep embedding [158, 195]. We list two most common applications for embeddings in computer vision fields as follows:

- **Image retrieval**: it aims to find similar images as the given query image from a large database of images. The candidate images are ranked according to the similarity distance (such as Euclidean distance) between the embeddings of test image and the query image. It can work in an unsupervised way with no need of the label of the query. The origins of the technique have been attributed to the early experiments conducted by Kato [69] into the

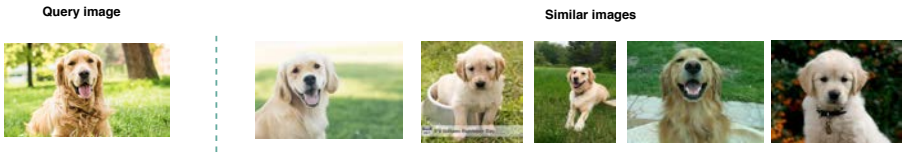


Figure 1.4 – Example of image retrieval results.

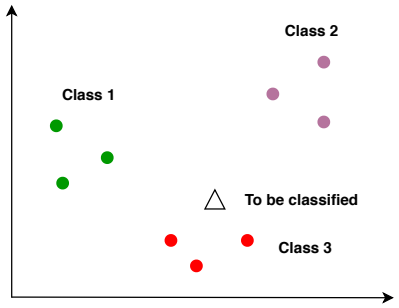


Figure 1.5 – The illustration of K-nearest neighbour (KNN).

automatic retrieval of images by colour and shape feature [34]. In the early stage of image retrieve, the three most common image matching features are colour, shape and texture. As the success of deep neural networks, richer semantic representations are learned for more accurate description of the object. Fig. 1.4 demonstrates an example of image retrieval results with embeddings, where a dog image is given as the query on the left, and the five most similar dog images are shown on right.

- **Image classification**: classification of objects is an important area of research and of practical applications in a variety of fields. Embeddings have played an important role for image classification problems with different types of discriminative classifiers, such as Support Vector Machine (SVM) [26], K-nearest neighbour (KNN) [191], decision tree, knowledge-based classifier and neural network. Fig. 1.5 illustrate how KNN works for image classification as an example: The test sample is classified to the closest class depending on the distance to the nearest k embeddings. This method significantly improves the performance for large scale datasets for which the problem of unbalanced data exists, fine-grained dataset where the task boundary is soft, and zero/few shot learning where the labeled training set is limited.

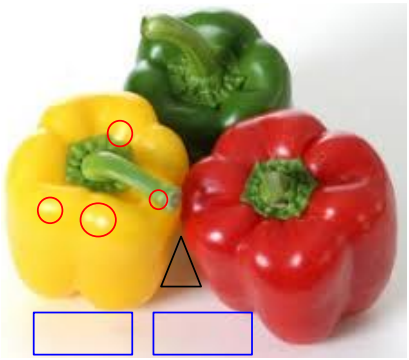


Figure 1.6 – Example image to show that color descriptor is affected by illuminance (red circle), specularities (blue rectangle) and shadow (black triangle).

1.2 Color embeddings and Deep embeddings

In this section we zoom into the aspects of embeddings which are the focus of this thesis: color embeddings and deep embeddings.

1.2.1 Color embeddings

In the Part I of the thesis, discriminative color embeddings are studied. Color is one of the important characteristics of materials in the world around us. It is one of the fundamental attributes in computer vision systems for image understanding. It has been explored in many applications, ranging from object recognition [43] to visual tracking [98] and object detection [73].

However, the challenge of color descriptions is that they are easily to be influenced by many factors in the real world such as unknown illuminance, shadows, specularities, image compression and unknown acquisition system. For example, in Fig.1.6, the visual colors we see are changed from the intrinsic colors due to different factors. Illuminance influence leads to the change of the pepper from yellow to white (marked in red circle), specularities influence causes the surface changing from while to light yellow and light red respectively (in blue rectangle) and shadow influence makes the surface chang from white to black (in black triangle).

In previous works, there exist two main methodologies to address this color description problem. One approach is by means of photometric invariants derived from reflectance models which describes the interaction of light, material and sensors [40, 43, 54, 65]. Given certain assumptions, these descriptors are invariant

(a) Red car (b) Beige sofa (c) orange T-shirt

Figure 1.7 – Example of images by searching the query of 'color name + object' in Google Image.

with respect to scene accidental events such as shadows, illuminant changes etc [36]. Some of these assumptions might be unrealistic in computer vision applications, such as known acquisition device, the requirement of high-quality images without any compression [161]. The main advantage of these methods is that there is no need of training data. The main drawback of these methods is that the certain assumptions limit their applications. It is not applicable for most of the computer vision systems where the data sets are collected from the Internet.

The second methodology to color description is based on color names. Color names are linguistic words which human use to describe colors in the world. such as terms of 'red', 'beige' and 'fuchsia'. Computational color names provide a mapping from color values to corresponding color names. Color name is found to be more robust and discriminative compared to physics-based approaches for a wide variety of computer vision fields, especially for image classification [163], texture classification [74], object recognition [75], visual tracking [32] and person re-identification [190]. The basic color name terms have been studied by Berlin and Kay in [11]. They are defined ad the the color names which are not subsumable under one of the other basic color terms. The number of basic color names varies in different languages. Most computer vision works consider the elevenbasic color terms of the English language: black, blue, brown, grey, green, orange, pink, purple, red, white, and yellow. Fig. 1.7 shows example of an image search from Google Image by the query of 'red car', 'beige sofa' and 'orange T-shirt'.

We have identified two research directions in color embeddings which we explore in this embedding.

Extend the Basic Color Name Set: One of Vincent van Gogh's famous painting 'sunflower' actually is the yellow flowers on a yellow background (see the left one in Fig.1.8). He called this 'light on light'. Due to the high similarity between the

Figure 1.8 – (The left)sunflower drawn by Vincent van Gogh, (the right)the pixel-wise color name annotation by eleven basic color names. The color names are represented by their corresponding color.

majority of foreground (dark yellow) and background (light yellow), to recognize the object in this image using the basic eleven color names for computer vision system is challenging. Color naming result with the eleven color names is shown in Fig.1.8(right). The sunflowers, the wall and the table are all described in 'yellow' due to the coarse and limited color names. Some work on other color representations found that extending the set to more than eleven dimensions might be beneficial [77]. On one side, mapping with more color names is supposed to improve the performance for image understanding. On the other side, extending the color name set is to break the inclusive basic eleven names space to exclusive space. In this exclusive space,color distribution can overlap, which has the possibility to confuse the computer vision system. For example, the subtle difference between 'cyan' and 'turquoise'. *These resulted in the research questions: how do we order the color name set and does image understanding benefit from a larger color name set.*

Weakly Supervised Domain-Specific Color Naming: As we know, the development of computer vision benefits from the amount of data. While labeling the data is always a time-consuming and laborious problem in practice. Supervised methods for color name learning are based either on labeled color patches [8, 117] or on pixel ground-truth masks, providing the color names for all the relevant items in the image [23, 99]. Van de Weijer et al. propose several variants of the PLSA model to learn color names from images retrieved from Google [165] in a semi-supervised manner, where the provided label describes the color of the principal object in the image, but no information on each pixel is provided by the given label. Semi-supervised learning decreases the effort significantly. Although the existing

Figure 1.9 – Examples and corresponding masks from EBAY dataset.

semi-supervised methods [23, 99, 165, 173, 197] don't need the pixel-wise labels during training phase, it still requires pixel masks at the testing phase. Fig.1.9 shows some examples and corresponding masks from EBAY dataset which is a manually labeled small dataset. While in many applications, different sets of color names are needed for the accurate description of objects. For example, the color names of lipsticks, hair, jewelry and so on, eleven basic color names are inadequate to describe them. Labeling data to learn these domain-specific color names is an expensive and laborious task. *Hence, more efficient methods are required to address the drawback of pixel-wise labeling problem for color naming for these specific domains in computer vision applications.*

1.2.2 Deep embeddings

In Part II of the thesis, we investigated the deep metric embeddings from two aspects. Recent works have pointed out serious shortcomings of discriminatively trained networks (attributed to the cross-entropy loss). Embedding networks were found to be more robust to the exposure of adversarial examples, better in the detection of out-of-distributionexamples, superior for transfer learning and obtained improved results for incremental learning [111, 142, 188], which leads us to think that the importance of embedding networks will further increase.

In this thesis, we identify two problems for metric learning: network distillation and lifelong learning for deep metric embeddings, which have been hardly investigated in previous works.

Compact Networks for Metric Learning: As the development of deep learning technology, large networks are known to provide excellent performance [140, 157]. While for many real-world applications, such as portable devices, the networks used to compute embeddings must be highly efficient, and therefore these applications cannot take advantage of the latest state-of-the-art deep networks. Research of network compression and network distillation have been proposed to address this problem. Network compression [48, 108] mainly focus on making the network lighter by reducing the parameters. While network distillation is defined with a

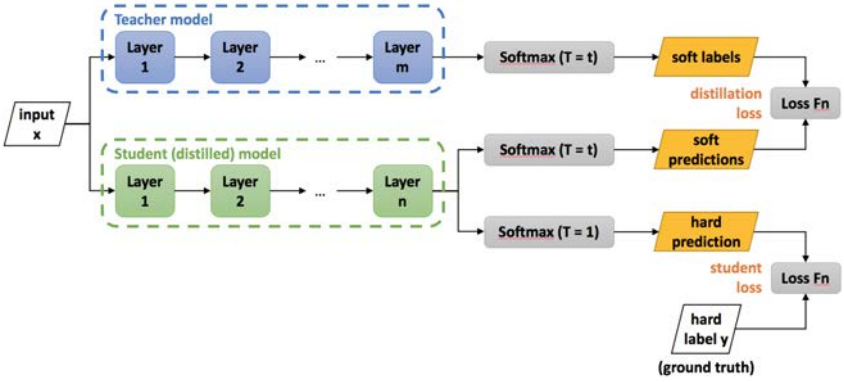


Figure 1.10 – Illustrations of network distillation for classification networks. Figure is from [203].

teacher-student setup. Typically the small student network is guided by the large teacher network by minimizing the cross-entropy loss between the outputs of theses two networks for softmax-based classification task [16, 57, 96, 137], as shown in Fig.1.10.

Bucilua et al. [16] first proposed to 'compress' a large, complex neural network to a smaller, faster model without significant loss in performance. The student network tries to mimic the ensemble networks by using the predicted labels obtained from the teacher networks training from the unlabeled data. It is further improved by Hinton et al. [57] by minimizing the loss of the signal from the logits (just before the softmax) to the probabilities (after the softmax), and introducing temperature scaling to increase the influence of small probabilities. Both of the works were to distill the function approximated by a powerful model/ensemble teacher into a single neural network student. Ba and Caruana study to compress deep and wide networks into shallower but even wider ones in [4]. Later Fitnet [137] introduces the distillation not only on the final probability but also the outputs of intermediate layers which further improving the performance on the small student networks. Different from these one-way transfer between a predefined teacher and a student in model distillation, Y. Zhang et al. [200] propose to use an ensemble of students to learn collaboratively and teach each other throughout the training process. Most of the literature works focus on classification problems. Recently, network distillation is further leveraged to more applications such as image super-resolution [64], Depth Estimation [132, 184], transfer learning [193] and adversary defense [125].

To our knowledge, Only two works have previously addressed knowledge distil-

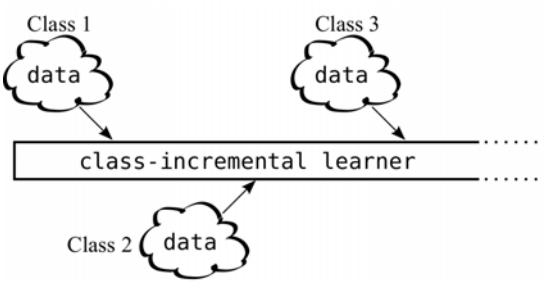


Figure 1.11 – Illustration of class-incremental learning, figure is from [135].

lation for embeddings. Chen et al. [21] study the network compression for metric learning by learning to rank. The drawback of their method is the limited batch size due to the product operation in the formula. PKT [129] models the interactions between the data samples in the feature space as a probability distribution. *Thus a more simple and efficient method is needed to be investigated to improve the existing works on learning metric from the teacher network.*

Lifelong Learning for Metric Learning: Most neural networks are evaluated in a somewhat unrealistic setting, where data of all classes are accessible all the time. While the exposure of data in real-world is sequential, and humans learn the world in a continual manner without forgetting of the old tasks. Artificial intelligence system should be able to incrementally learn new classes like humans, when training data for them becomes available. The illustration of incremental learning is shown in Fig.1.11 from the work [135].

Lifelong learning in deep neural networks is defined to learning a sequence of tasks in a continual fashion, which is more realistic in the real world. At each moment, the model has only access to the data of the current task. However, catastrophic forgetting happens when training in this sequential manner, which is described by McCloskey and Cohen [112]. It refers to a significant drop in performance of previous tasks due. This is caused because the parameters are optimal for the current task, which results in a shift for the features of the previous tasks.

Lifelong learning has explored a variety of strategies to prevent the forgetting of the previous tasks for softmax-based classification networks. It can be roughly divided into three main categories as follows:

1. Using exemplars/synthetic data of old tasks: Keeping exemplars of the old data is a popular way to prevent forgetting. The model saves a small subset of the real training data from previously learned tasks to prevent the network from forgetting previous tasks during the training of new task. These saved

exemplars are combined with the data of the current task, and the network parameters are jointly optimized. A distillation loss such as KL-divergence is typically used to transfer knowledge from one network to another to prevent forgetting information [18, 106, 135]. One of the drawback of saving exemplars is caused by the data imbalance between the old and new classes, especially for a large amount of classes. In [181], they found that the last fully connected layer has a strong bias towards the new classes, and presented to correct this bias with two bias parameters by a linear model. Hou et al. [61] propose to incorporate three components, cosine normalization, less-forget constraint, and inter-class separation, to mitigate the adverse effects of the imbalance.

An alternative is to learn a generative model of previous tasks, and generate synthetic samples (i.e. pseudo-rehearsal) that are combined with samples of new task instead of using the real samples [148, 179].

2. Without using old data: These works prevent forgetting by optimizing network parameters on the current task while preventing the drift of already consolidated weights. Only data of the current task are used to train, without having access to previous data. The first work is proposed by Learning without forgetting (LwF) [96]. It adds an additional loss as a regularization term on probabilities by retaining the knowledge gained earlier to improve performance for both tasks. EWC [80] and R-EWC [101] include a regularization term on the weights that forces parameters of the current network to remain close to the parameters of the network trained for the previous tasks. Zenke et al. [199] propose to compute the consolidation strength of synapses in an online manner, and extends them with a memory to accumulate task-relevant information. Aljundi et al. [2] accumulates the importance for each parameter of the network, based on how sensitive the predicted output function is to a change in this parameter. Dhar et al.[33] propose to use the attention distillation loss to retain information of the previous classes when new classes are added.

   Another way to prevent catastrophic forgetting without saving any old data is by dynamic network expansion mechanism. HAT [144] presents a task-based hard attention mask that maintains the information from previous tasks to reduce catastrophic forgetting. PackNet [110] and Piggyback [109] learn task-specific masks to preserve the information from previous tasks. Expert gate [3] learns networks which are selected based on the outcome of a set of gated auto-encoders.

   In [63, 94, 124], authors propose a dynamic network expansion mechanism

that ensures sufficient model capacity to accommodate for continually incoming tasks.

As we mentioned, the previous works focus on classification networks. To our knowledge, lifelong learning for metric learning has not been investigated.

1.3 Objectives and Approach

In this thesis, we aim to improve the two main types of embeddings discussed above: color embeddings and deep embeddings. For each type of the embeddings, we investigate two aspects of them.

1.3.1 Color embeddings

From the above discussion we arrive at the following two objectives for color embeddings:

Beyond Eleven Color Names for Image Understanding: Studies on other color representations found that extending the set to more than eleven dimensions might be beneficial [77]. Inspired by this observation, we proposed to extend the basic color name set to learn more discriminative color descriptors. There are many possible color names which could be added to the eleven basic color terms. Several studies investigate the color names which are widely used [120]. The problem we need to address is how to augment the color name set and what is the ordering for new color terms to be included.

In Chapter 2, we propose a method which can augment the basic color terms with additional color names according to their complementary nature with the basic color names. Specifically, given a set of color names, we add color name which is less represented by the color names in the set, according to the correlation matrix between the color set combined with basic color names and selected color names and the set of not selected color names. This procedure is iterated to produce a ranking of color names to add to the initial basic color terms. As a result we can compute new color name descriptors of arbitrary length.

Weakly Supervised Domain-Specific Color Naming Based on Attention: The majority of existing color naming methods focuses on the eleven basic color terms of the English language. Because of the limitation of eleven basic color names for specific domains, specific color name set is needed for more accurate description. While labelingfor these domains is time-consuming. To address the drawback of the deep learning approaches for color naming, we propose a weakly-supervised deep learning framework based on attention.

In Chapter 3, a new two-branch network design for color naming based on attention is proposed, which is capable of automatically discovering relevant regions related to weak image labels, and simultaneously learn a mapping between color values and color names. In addition, a large-scale dataset is collected by using a Web image search engine, which contains 11 basic color naming images for 4 categories, and a dataset for domain-specific color naming which includes color names for horses, eye colors, lip colors, and the tomato growing stages.

1.3.2 Deep embeddings

Based on the analysis of deep embeddings, we define two objectives which are listed as follows:

Learning Metrics from Teachers: Compact Networks for Image Embedding: We address the problem of the inability of large network to be deployed on mobile device. Network distillation has been successfully applied to improve image classification for small networks learning from large networks, but has hardly been explored for metric learning. We focus on network distillation techniques for the efficient computation of feature embeddings with small networks in our thesis. In Chapter 4, we propose to use network distillation to efficiently compute image embeddings with small networks. To do so, we propose two new loss functions that guide a small student network form a large teacher network: one based on an absolute teacher, where the student aims to produce the same embeddings as the teacher, and one based on a relative teacher, where the distances between pairs of data points is communicated from the teacher to the student. In addition, we investigate various aspects of distillation for embeddings, including hint and attention layers, semi-supervised learning and cross quality distillation.

Semantic Drift Compensation for Lifelong Learning of Embeddings: The vast majority of methods in lifelong learning have focused on softmax-based classification networks. In contrast to previous work, we study lifelong learning in embedding networks and not on classification networks. In addition, instead of preventing the drift of features, we aim to estimate the drift and compensate for it.

In Chapter 5, we propose an approximation of this semantic drift based on the drift which is experienced by data of the current task during its training. Having an estimate of the drift we show that previous tasks can be compensated for this drift, thereby improving their performance. In addition, we show that embedding networks suffer significantly less from catastrophic forgetting compared to classification networks even when applying simple finetuning to learn new tasks.

Color Embeddings Part I

How do we use color embeddings beyond basic eleven color names?

2 Beyond Eleven Color Names for Image Understanding *

2.1 Introduction

The description of color is important for many computer vision applications. The description of color is difficult because of the many factors that influence the color value, such as shadows, specularities, image compression, image blur, etc. One approach to address this problem is by means of photometric invariants [41, 42, 53] which are derived from reflectance models. These are invariant with respect to scene accidental events such as shadows, illuminant changes etc. However, these color descriptors are based on assumptions which are often unrealistic in computer vision applications, such as known gamma compression, and absence of image compression. In addition, they suffer from a drop in discriminative power [164].

Color names are linguistic labels which humans use to communicate the colors in the world. Examples of color names are 'red', 'olive' and 'beige'. Computational color names provide a mapping from color values to corresponding color names [7, 116, 165]. Because of their high discriminative power and robustness to photometric variations they were found to be an excellent color representation. In comparison to other color descriptors, including descriptors based on photometric invariance theory, the color name descriptors were found to obtain superior results in many application, especially for image classification [76], image retrieval [104], object recognition [73], person reidentification [202], and visual tracking [32].

Berlin and Kay [10] in an influential linguistic study defined the term 'basic color term' as being (among other characteristics) a color name which is not subsumable under one of the other basic color terms. They then identified eleven such terms in English language, namely: black, blue, brown, green, gray, orange, pink, purple, red, white, and yellow. Most work on computational color names follow this convention and compute mappings for the eleven basic color terms [7, 165]. Studies on other color representations found that extending the set to more than eleven dimensions might be beneficial [77]. That resulted in the research questions which is addressed in this chapter, how do we extend the color name set and does image understanding benefit from a larger color name set.

There are many possible color names which could be added to the eleven basic

*This chapter is based on a publication in the Journal of Machine Vision and Applications [196].

color terms [119]. However the problem is how to augment the color name set and what is the ordering for new color terms to be included. Inspired by [77, 119, 165], we propose a method which can augment the basic color terms with additional color names. Given a set of color names, we add color name which is less represented by the color names in the set. This procedure is iterated to produce a ranking of color names to add to the initial basic color terms. As a result we can compute new color name descriptors of arbitrary length (limited only by the size of our color name set). In the experiments we will evaluate color name sets of 15 and 25 and show that they outperform the color representations based on 11 color names. In conclusion the contributions of this chapter are:

- We collect a new dataset of images to train an extended set of color names. The set contains a total of 39 color name categories.
- We propose a method which allows us to rank the additional color names, and therefore construct discriminative color name descriptors of arbitrary size. We also show that a naive extension of the color name descriptor leads to unsatisfying mapping of colors to color names, whereas our approach to extend the color name descriptors obtains much more acceptable mappings.
- We evaluate the new color name descriptor on visual tracking, person re-identification and image classification and show that the performance improves over the standard eleven dimensional color name descriptor. In addition, we design two psychophysical experiments which show that our approach improves agreement to human users when labeling color patches with color names.

In the next section we will explain the database collection, and show our approach to ranking the color names. In Section 2.4 we introduce our approach to extending the color name set beyond eleven color names. In Section 2.5 we evaluate the color name descriptor and we conclude in Section 2.6.

2.2 Related Work

Here we briefly summarize the related work on methods for color description in computer vision.

We distinguish between two main methodologies to the color description problem. The first methodology is based on reflection models which describe the interaction of light, material and sensors [41, 42, 53, 65]. From these reflection models photometric invariant descriptions of the material color can be derived. Given certain assumptions these descriptors can overcome the dependence of the color

description on scene accidental events. Examples are color descriptions which are invariant to illuminant color, shadow-shading and specularities[37, 41, 159]. The main advantage of these methods is that they do not need training data and therefore do not require a laborious and costly labeling phase. The main drawback of these methods is that the assumptions on which they are based (for example white illumination, known acquisition device, etc) limit their application. Typically they require high-quality images without compression artifacts, and are not very effective for the medium quality images which are currently used in the many large scale data sets which have been collected from the Internet.

The second methodology to color description is based on color names. Humans use color names routinely and seemingly without effort. They have been primarily studied in the fields of visual psychology, anthropology and linguistics [49]. Basic color terms have been studied in the influential work of Berlin and Kay [10]. They are defined as those color names in a language which are applied to diverse classes of objects, whose meaning is not subsumable under one of the other basic color terms, and which are used consistently and with consensus by most speakers of the language. Basic color names were found to be shared between languages. The number of basic terms varies from two in some indigenous languages to twelve in for example Russian.

Computational color naming [6, 116, 166] aims to learn a mapping from pixel values to color name labels. A clear example in computer vision where color names are desired is within the context of image retrieval, where a user might want to query for images with "blue sofas". The system recognizes the color name "blue", and orders the retrieved results on "sofa" based on their resemblance to the human usage of "blue'. Later research showed that color names actually also constitute an excellent color descriptor. They were found to be robust to photometric variations, while having in general higher discriminative power than the photometric invariants.

In recent years, these two approaches to color description, namely, the physics-based and the color name methods, have been compared on a wide variety of computer vision applications. In an earlier conference work, we provided an overview of applications where color names and photometric invariants were compared [162]. Constantly, color names were found to outperform the photometric invariance approaches by a significant margin. Color names have been extensively tested in image classification tasks [71, 76], object recognition [73], person re-identification [189] and action recognition [72]. The main reason for the success of color names is the high discriminative power which they possess, while being robust to photometric variations in images. It motivates us to investigate extending the color name set, with the aim to further improve the performance.

There have also been several attempts to divide the color space into categories

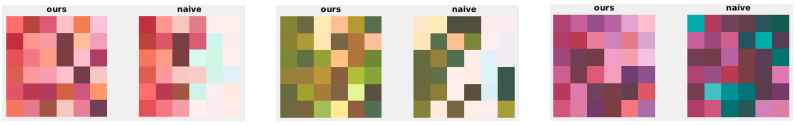
Figure 2.1 – Example images for the color ochre from the augmented color name dataset.

using psychophysics, either by focusing on the regions of consensus [14, 155] or the categorical boundaries [127, 128]. All these models are based on a small subset of agreed focal colors.

2.3 The augmented color name dataset

In this section we explain the collection of the augmented color name dataset. The English language has hundreds of color names apart from the eleven basic color terms. To select a limited set we make use of two recent studies of color names in the English language [118, 119]. These studies investigated which color name words were widely used, had a shared meaning among the speaker population, be salient and therefore identifiable in an array of colors, and can be reliably distinguished in color space. They investigated a total of 28 candidates including beige, burgundy, cyan, fuchsia, lavender, lilac, magenta, maroon, mauve, ochre, olive, peach, plum, rose, salmon, tan, teal, turquoise, violet, burgundy, lilac, lime green, light green, dark green, dark purple, light blue, mustard, olive green, pale yellow and mint green.

The choice of training data to infer the mapping from RGB values to color names should be dictated by its application objective. In this chapter we are interested in color name mappings which can be used for image understanding applications which in general are uncalibrated. We therefore also resort to learning the mapping from uncallibrated data crawled from Google similar as [165]. We collect images from Google by using the search query 'colorname + objects', e.g. 'mauve objects'. An example of six images for 'ochre objects' is provided in Fig. 2.1. The term 'objects' has been added to diversify the query results. A direct query for only the color term



(a) red (b) olive (c) fuchsia

Figure 2.2 – Comparisons of the top 36 retrieved Munsell patches given a color name. We compare results of our method and the naive method to extend the color name set. Results clearly show that the naive approach fails to retrieve all relevant Munsell patches.

leads to color patches which do not represent colors in real-world situations. In total we collect 250 Google image per color name.

All images are considered to be in sRGB and they are gamma corrected accordingly. Even though these images come from a wide range of cameras, the lighting settings are unknown, and image compression is most likely applied. It has been shown that color names learned from such images provide better results in computer vision applications. This is caused by the fact that in computer vision applications also often the lighting is unknown, image or video compression has been applied, etc. For a further discussion on the differences on learning color names from calibrated and uncalibrated images we refer to [9]. To infer the color name from this dataset we transfer the images to histograms in Lab space. Pixels are represented by assigning their Lab space values into a finite vocabulary by assigning each value to a regular $10 \times 20 \times 20$ grid. [†].The dataset and the newly computer color name mappings are available at https://github.com/yulu0724/ColorNamesExtension.

2.4 Ranking of Additional Color Names

In this section we outline our approach to estimating the color name distribution from this data. We start by explaining the method from Van de Weijer et al. [165] for color name estimation and then we propose our approach to use correlation to rank color names.

[†]Because the Lab-space is perceptually uniform we discretize it into equal volume bins. Different quantization levels per channel are chosen because of the different ranges: the intensity axis ranges from 0 to 100, and the chromatic axes range from -100 to 100.

2.4.1 Computation of color name mappings

The objective of computational color naming is to find $p(c|w)$ which is the probability of a color name c, given a color value w, to which we also refer as a color name mapping. For the computation of $p(c|w)$ we will use the algorithm proposed in [165]. However it requires some adaptation to be used for color name sets which include non basic color names.

We apply probabilistic latent semantic analysis (PLSA) [59] to estimate the probability of color values when given a color name. PLSA is a generative model, in this case on how images are generated: the model assumes that images consist of a number of topics (in our case color names) which generate words (in our case RGB values). This model allows us to learn from noisy data such as the data set we collected from Google.

We model the distribution of RGB values w in an image i to be a mixture of color name topics c. In PLSA the conditioned distribution $p(w|i)$ is modeled by

$$p(w|i) = \sum_{c \in C} p(w|c)\, p(c|i), \tag{2.1}$$

where C is the set of color names. Here $p(w|i)$ is the collection of color histograms of the images and is known. Both $p(w|c)$ and $p(c|i)$ are unknown and need to be estimated. This can be done by minimizing the following loss

$$L = \sum_i \sum_w n(i,w) \log p(i,w), \tag{2.2}$$

with the EM algorithm. Here the joint distribution $p(i,w) = p(i)\, p(w|i)$ where $p(i)$ is considered uniform; $n(i,w)$ is the term frequency and can be directly computed from the training set.

Similar as [165] we introduce an additional term which enforces the color name mappings to be unimodal in L space. It enforces the distribution $p(w|c)$ to have a single mode and to decrease monotonically. Enforcing this is appropriate since we consider this to be a property of real color names. It can be obtained by adding a regularization term to the log likelihood:

$$L = \sum_i \sum_w n(i,w) \log p(i,w) - \gamma \sum_c \sum_w \big(p(c|w) - \rho_c(w)\big)^2, \tag{2.3}$$

here ρ_c is computed from the estimated distribution $p(c|w)$ with a grey scale reconstruction (for more details on this procedure we refer to [165]). The second term which is weighted according to γ enforces the estimated distribution to be close to the unimodal distribution by penalizing their difference.

As a second change to standard PLSA, an adjustment was proposed to allow for

the usage of the weak label of the image (the labelling identifying the color name of the image) [165]. This can be done by assuming that $p(c|i)$ is drawn from a Dirichlet distribution of parameter α_{l_i}. Here $\alpha_{l_i}(c) = t \geq 1$ for $c = l_i$, and $\alpha_{l_i}(c) = 1$ otherwise. Here l_i is the label of image i. This leads to the following equation

$$p(c|i) \propto \left(\alpha_{l_i} - 1\right) + \sum_w n(i,w)\, p(c|w,i). \tag{2.4}$$

The computation of the distributions $p(w|c)$ and $p(c|i)$ is done by iteratively applying an EM-like algorithm, where we iterate until convergence between

- minimize Eq. 2.3 as a function of $p(w|c)$ with a conjugate gradient method,
- compute $p(c|i)$ according to Eq. 2.4.

This provides us with the color name mappings $p(w|c)$ which we were aiming for. We use $t = 2$ and $\gamma = 200$ in our experiments.

2.4.2 Extending the color names set

One of the hurdles to extending the basic color term set with other color names is that the resulting color name set can no longer be interpreted as a probability distribution, i.e. for C larger than eleven the $\sum_{c \in C} p(c|w) \geq 1$. For example there are colors which can be described as being clearly 'violet', 'plum', 'purple' at the same time. This does not happen with the eleven basic color terms, because one of their main characteristics is that they are not subsumable under one of the other basic color terms. As a consequence the PLSA algorithm cannot be applied to color name sets which are larger than eleven because it is only valid when $\sum_{c \in C} p(c|w) = 1$. The violation of this equality increases with the number of additional color names.

To stress the fact that we are no longer working with probabilities we write $q(c|w)$ to be the membership of the color name c to the color word w. We allow $\sum_{c \in C} q(c|w) \geq 1$ and enforce $0 \leq q(c|w) \leq 1$. For example a color could have a membership of 1 to 'green', and 0.8 to 'lime'.

As mentioned above the violation is smallest in case we only add a single color name to the color name set. We therefore propose the following procedure for the estimation of $q(w|c)$: (1) for the eleven basic color terms we use the PLSA algorithm, and set $q\left(w|c_{\{1,\ldots,11\}}\right) = p\left(w|c_{\{1,\ldots,11\}}\right)$, (2) for additional color names we compute $q\left(w|c_{\{12,\ldots,39\}}\right)$ by adding a single color name at the time to the basic color terms and apply the PLSA algorithm. E.g. we add color name n to the basic color name set (yielding a total of 12 color names), estimate $p\left(w|c_{\{1,\ldots,11,n\}}\right)$ and set $q\left(w|c_{\{n\}}\right) = p\left(w|c_{\{n\}}\right)$, and repeat this procedure for all color names not in the

basic color name set. As a result of this procedure we have the $q(w|c)$ for the 11 basic color terms and the 28 additional color names. Finally, we obtain $q(c|w)$ by applying the Bayes theorem:

$$q\left(c_j|w\right) = \frac{q\left(w|c_j\right)}{\sum_{i=1}^{11} q\left(w|c_i\right)}, \tag{2.5}$$

where we assume a uniform prior over the color names.

In Fig. 2.2 we illustrate the importance of the iterative construction of the $q(c|w)$ which we propose here. If we would apply a naive extension of the method proposed in [165] the additional color names will compete with the basic color names, and we enforce $\sum_{c \in C} p(c|w) = 1$ to be true. As a result the borders of color names will move around when adding additional color names. This can be considered an undesired effect since colors which would previously considered to be 'red' with a high probability would suddenly be only considered 'burgundy'. In Fig. 2.2 we show the top 36 retrieved Munsell [70] patches given three color names (we consider a total of 329 Munsell patches). The retrieval shows the patches with the highest probability given the color name. Similar results were obtained for the other color names. We never observed the naive results to obtain a better selection of color names. Whereas our iterative scheme to compute $q(c|w)$ provides a relevant set of color patches, the naive approach only manages to return part of the relevant Munsell patches. As a consequence, in a retrieval application where a user looks for 'fuchsia' shoes she would only retrieval part of the relevant shoes from the dataset when based on the naive approach.

2.4.3 Ranking additional color names

In the previous section we have proposed to add 28 new color names to the basic color name set. In this section, we address the question how to rank these new color names. The ranking is of importance for the construction of compact color name representations. For example, if we would like a 15 dimensional color descriptor we would add the first four color names from the ranking to the eleven basic color terms.

When adding color names we would like them to be as different as possible to the ones which have already been selected. A color name which is significantly different from the existing set would increase the discriminative power of the combined color name set, and therefore improve its application to image understanding.

Consider you would like to select the best color name from a color name set C_2 to add to a set of color names C_1. For brevity we will use the notation $B = q(w|c)$

Figure 2.3 – (top row) The eleven basic color terms. (second and third row) proposed order in which to add 28 additional color names to the basic color term set.

(a matrix of 4000×39) where we use $b_i = q(w|c_i)$ and hence $B = [b_1, ...b_{39}]$. We write $\hat{B} = [\hat{b}_1, ...\hat{b}_{39}]$ to indicate the $L2$ normalized column vectors, and $\hat{B}^C$ to be the matrix B which contains the columns of the indexes included in set C. Given a color name set C_1 we will add the color name j^* from C_2 according to

$$j^* = \underset{j \in C_2}{\arg\min} \left(\max \left(\left(\hat{B}^{C_1} \right)^{\mathrm{T}} \hat{b}_j \right) \right). \tag{2.6}$$

This equation considers for each of the potential color names the correlation with all the color names in set C_1. It then selects the color name which has the lowest maximum correlation and could therefore be considered the most different from the existing ones[‡]. We initialize the process with $C_1 = \{1, \ldots .11\}$ containing the basic color terms and C_2 all other color names. Next Eq. 2.6 is applied N times, at each step increasing the color name set C_1 with j^* and removing it from C_2.

In Fig. 2.3 the results are shown when applying Eq. 2.6 until all color names have been selected. For example, as a set of 15 color names we would add 'turquoise','olive green', 'mint green' and 'burgundy' to the basic eleven color terms. Note that our approach selects 'turquoise' to be the 12th color name, which is interesting since there are some linguistic studies which suggest that 'turquoise' could be considered as a twelfth basic color term [204].

To illustrate the learned mappings we apply them to the challenging synthetic image with 11, 15 and 25 color names. The results are shown in Fig. 2.4. Here we only show the color name with the maximum probability for each pixel. Especially on the green-blue border and in the purple-pink-red region new color names are

[‡]We also experimented with selecting the color name with the lowest mean correlation but results were inferior.

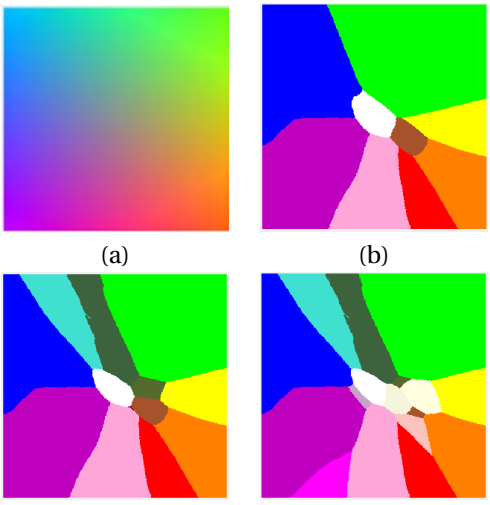


Figure 2.4 – (a) the original image and the assignment based on (b) the 11 color names mapping; (c) the 15 ranked color name mapping; and the (d) the 25 ranked color name mapping.

introduced to allow for more precise color descriptions.

2.5 Experimental results

The eleven basic color names are popular color descriptors and have been shown to obtain excellent results on a large variety of image understanding fields, including image classification [76], action recognition [72], image retrieval [104], person re-identification [202], and visual tracking [32]. In these papers, which compared the color name descriptor against a large variety of color presentations, the color name descriptor came out with superior results. Therefore, in these experiments we will compare our new extended color name descriptor against the standard color name descriptor based on the eleven basic color terms. We will evaluate the descriptor on three relevant computer vision applications namely visual tracking, person re-identification and image classification and we perform an additional user preference experiment.

Table 2.1 – Percentage of correctly classified pixels in the Ebay dataset for various classifiers.

	PLSA	SVM	KNN
Accuracy	72.2%	69.30%	67.66%

2.5.1 Color Naming

In a first experiment we compare the PLSA pipeline we use for color naming against two baselines, namely SVM and k-nearest neighbors. To do so we perform the color name experiment from [166] where the task is to classify pixels from *Ebay dataset* images into the eleven basic color terms. The dataset contains a total of 440 images, consisting of ten images for the eleven color names for four different categories (cars, shoes, dresses, and pottery). All images come with a mask image which identifies the pixels which belong to the named object. Evaluation is only performed for the pixels in the mask. One example of an image and its ground truth mask is given in Fig. 3.6.

All three methods are trained on the L*a*b-histograms of Google images. For the PLSA we use the setup as explained in Section 2.3. For SVM we use linear kernel [§] where we cross validate for optimal c value. For k-nearest neighbor we optimize for k on the validation set and found 25 to be optimal.

The results of this experiment are provided in Table 2.1. We can see that the PLSA algorithm obtains superior results compared to both SVM and k-nearest neighbors. In addition we have applied the three methods to a synthetic image and results are provided in Fig. 2.6. We can see that PLSA manages to obtain smoother edges than k-nearest neighbor and SVM, and that SVM makes many errors for the highly saturated colors (along the borders of the image). These are colors which are less frequent in real images, and therefore have fewer training examples. In conclusion, PLSA based color naming outperforms other popular classifiers for the task of color naming and we will perform the remaining experiments based on the color names which are computed with PLSA.

2.5.2 Image classification

For image classification we perform experiments on the Flower102 dataset [121] which contains 8189 images of 102 different kinds of flower (see Fig. 2.7). It has been

[§] We found that more complex kernels such as for example intersection did not improve results.

(a) (b)

Figure 2.5 – (a) example image from Ebay labeled with the color name 'green' and (b) the ground truth mask of the image identifying the pixels which are related with the color name. The results in Table 2.1 show the percentage of pixels on the mask which are labeled in agreement with the ground truth label.

selected because of the importance of color for flower classification and the real-world challenges such as significant scale and illumination changes. We follow the standard bag-of-words (BOW) [29] approach. In BOW an image is firstly represented by a collection of local image features, and then each local feature is discretized into a visual vocabulary from the represented cues such as color and shape. Then images are represented as a histogram over visual words. For classification we apply an SVM with intersection kernel.

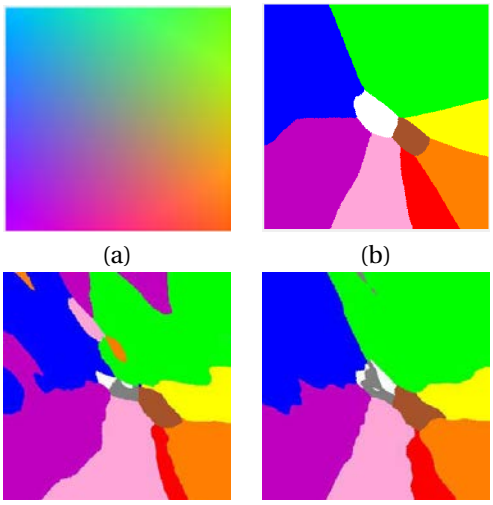
In a first experiment we compare our proposed method for ranking color names (see Section 2.4.3) to two baseline methods:

- RANDOM: a color name set with more than eleven color names is constructed by choosing the eleven basic color terms and randomly adding additional color names until the desired number is reached.

- LABCN: this method is derived from the mean LAB values of the color names. Following the notation of Section 2.4 the mean of each color name is computed according to:

$$\mu_j^{LAB} = \sum_i LAB(w_i)\, p(w_i|c_j), \tag{2.7}$$

which is a weighted mean which is computed by multiply the LAB value of the color value w given by $LAB(w_i)$ with the probability of the color value w belonging to the color name c_j. The ranking is then obtained by replacing



(a) (b) (c) (d)

Figure 2.6 – (a) the original image and the assignment based on (b) PLSA; (c) SVM; and the (d) k-nearest neighbors on the 11 color name mapping.

the selection of Eq. 2.6 by:

$$j^* = \arg\max_{j \in C_2} dist\left(C_1, \mu_j^{LAB}\right), \tag{2.8}$$

where the distance between the set of color names C_1 and the color name j given by $dist\left(C_1, \mu_j^{LAB}\right)$ is defined to be equal to the minimum distance of color name j to any of the member of C_1. Thus, the algorithm computes the LAB color name centers, and starting from the eleven color names, adds iteratively that color name which is furthers away from any of the already selected color names.

We test the three different rankings with 15 and 25 color names on flower classification application. Results are shown in Table. 2.2. As can be seen increasing the set of color names increases the performance, and results improve with 2.1% for our method. A larger number of color terms can enhance the discriminative power but also weaken the photometric invariance. We found that increasing color names beyond 25 color terms did not further improve results.

Next we compare to the two other baselines for ranking the color names. For

Table 2.2 – Classification accuracy on Oxford Flower102 with different methods.

Accuracy	Ours	LABCN	RANDOM
11	37.23%	37.23%	37.23%
15	37.73%	37.58%	37.60%
25	39.34%	38.61%	38.84%

Figure 2.7 – Example images from Flower102 dataset.

eleven color names the methods are equal because they all consider the same eleven basic color names. The results show that our method is slightly better than the RANDOM and LABCN baselines when using 15 color names. The difference gets larger when considering 25 color names. When considering the performance gain with respect to eleven color names our method obtains a gain of 2.1% whereas the baseline methods only improve by around 1.4%. The fact the LABCN does not outperform the RANDOM method could be caused by the fact that even though it is selecting color names which describe colors currently not well described by the color name set, it does not take into account the frequency of these colors occurring.

In a second experiment on the Oxford Flower102 dataset we compare the color name descriptor to the discriminative color descriptors (DD) proposed in [77]. These descriptors are not semantic, which are not linked to human color names, but were found to obtain state-of-the-art results. They proposed two sorts of discrimina-

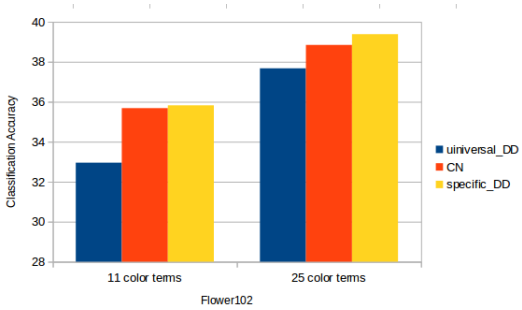


Figure 2.8 – Classification accuracy on Oxford Flower102 comparing color names with discriminative color descriptors.

tive descriptors: database specific color name descriptors which are optimized for a specific classification problem and need to be learned from labeled training data, and universal color name descriptors which are learned from several databases and can then be applied without adapting them to the specific dataset. The color names which we propose in this chapter are universal color descriptors since they do need to be relearned for new datasets.

The results of classification are provided in Fig. 2.8. The results show that the descriptor based on the 25 color name set outperforms the universal discriminative descriptor with the same dimension. The dataset specific color descriptor only slightly outperforms this results. Given that the difference is very small, for many applications it might be preferable to apply the 25 color name descriptor which does not require dataset specific training.

2.5.3 Color naming for tracking

Visual tracking is a challenging problem in computer vision. Recent work has shown that color names provide superior performance when compared to other color representations for visual tracking [32]. Their tracker is based on the CSK tracker [55] which is a correlation filter based tracker which only considers the luminance channel. In [32] they show that extending the tracker with color names provides a significant performance improvement. We will apply the same tracker in our experiments, but we will replace the eleven color name mapping with the mappings we have derived here. Results are provided for color name representations with 15 and 25 color names, where the selection is performed with Eq. 2.6. An additional

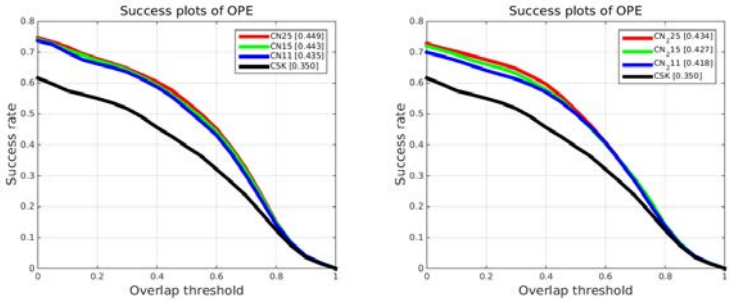


Figure 2.9 – Success plots for (top) various different color name mappings, and (bottom) various compressed color name mappings.

weighting term λ was introduced to balance the luminance and color channels[¶]. Since the introduction of color names in tracking [32], they have been applied in several state-of-the-art trackers [31, 60, 95] showing that color names are among the preferred color representations.

The experiments are performed on an Intel(R) Xeon(R) CPU E5-1620 v3 @ 3.50GHz CPU with 32 GB RAM with a native Matlab implementation. In our approach, we use the same parameter values as suggested by Danelljan et al. [32] for the ACT tracker. We also employ the same dataset, including 35 color sequences used in the evaluation of tracking methods [180] and 6 other color sequences namely: Kitesurf, Shirt, Surfer, Board, Stone and Panda. The sequences used in our experiments pose challenging situations such as motion blur, illumination changes, heavy occlusions, low resolution, fast motion, in-plane and out-of-plane rotations, scale variation, out of view and background clutter. To validate the performance of our approach, we follow the protocol used in [180].

In the first experiment we compare the tracker using three different color name mappings: with the original 11 and with the two new color mappings of 15 and 25. Fig. 2.9(top) shows the success plots. The success plot contains the overlap precision (OP) over a range of thresholds. OP is defined as the percentage of frames where the bounding box overlap exceeds a threshold $th \in [0,1]$. The trackers are ranked using the area under the curve (AUC). As the channels of the color name mappings increase, the performance of the color tracker improves. The 25 dimensional color mapping obtains a 28% relative gain over the original CSK tracker which obtains 35% OPE score.

[¶]We found for optimal results were obtained with a $\lambda = 1$ for 11 color names, a $\lambda = 0.9$ for 15 color, and a $\lambda = 0.8$ for 25 color.

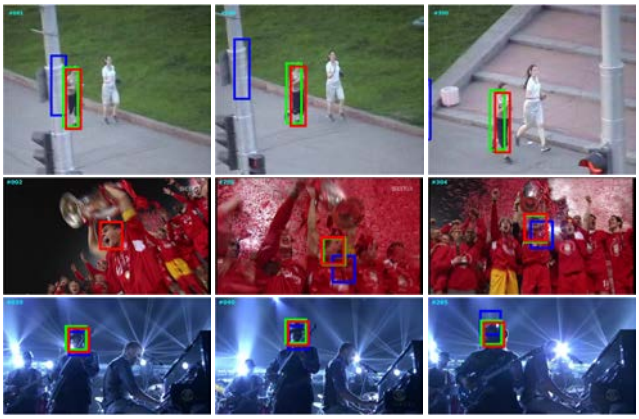


Figure 2.10 – Comparison of 3 different color name representations for trackers in challenging situations such as illumination variation, occlusion, motion blur and in-plane rotation. The example frames are from the Jogging, Soccer and Shaking sequences respectively. The results of 11D, 15D, and 25D are represented by blue, green and red boxes respectively.

Danelljan et al. [32] pointed out that the speed of the tracker decreases with the number of channels and therefore the color name based trackers are significantly slower. However, they proposed to dynamically map the color representation to a lower dimensional representation (they show that 2 dimensions is enough). When we apply the same dynamic dimensionality reduction to our trackers we obtain the results which are presented in Fig. 2.9(bottom). The results slightly deteriorate with respect to the full representation but the speed increases from 89 to 128 fps for 15 dimensions and from 66 to 110 fps for 25 dimensions. In Fig. 2.10, we illustrate the results of the trackers on three sequences. Note for example, in the Jogging sequence, occlusion appears in frame #81, and the traditional low dimensional color name representation used in ACT [32] fails to track the woman, but the tracker using high dimensional color names can re-detect the position of the woman after occlusion.

Finally, we show the performance of the three trackers for several attributes as proposed by Yi et al. [180]. The 25 dimensional color name mapping improves over the standard 11 dimensional color mapping for all the eleven attributes. In Fig. 2.11 the results for four of them are shown. It can be seen that with increasing dimensionality of the color name mapping the performance for illumination variation, low resolution, motion blur and occlusion improves. Especially the performance

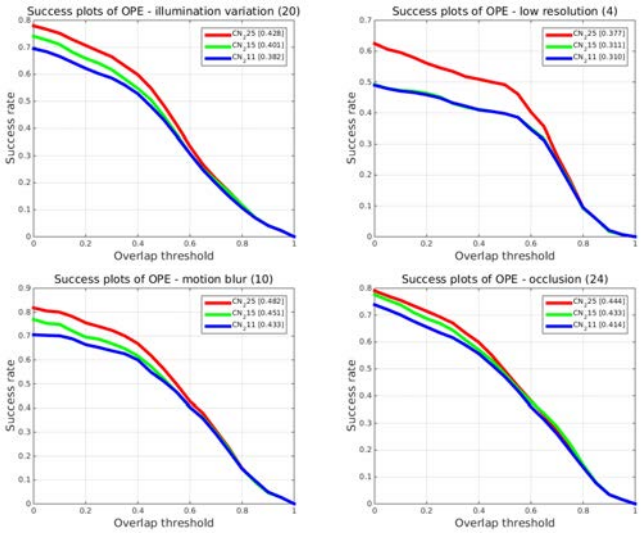


Figure 2.11 – Success plots for several attributes, including illumination variation, low resolution, motion blur and occlusion. The increased color name representation outperforms the original color name representation for these attributes.

gains for low resolution (relative gain of 21%) and illumination variation (relative gain of 12%) are noteworthy.

2.5.4 Color naming for re-identification

In several recent studies [103, 189, 202], color names have been extensively used to encode color information for person re-identification. To validate our approach, we perform the experiments on the challenging Market-1501 dataset [202] for the person re-identification task. The dataset comprises of 32668 annotated bounding boxes of 1501 identities. We follow the bag-of-words pipeline as described in [202]. A visual vocabulary is constructed using the standard K-means algorithm on the training bounding boxes. For fair comparison, we fixed the size of visual vocabulary to 350 words for all color descriptors. For each local color feature, a Multiple Assignment (MA) strategy is employed to locate its nearest neighbor under Euclidean distance. The MA parameter is fixed to 10 visual word indices. The performance is measured by using a cumulative matching characteristic (CMC) curve which plots the probability of correct identification compared to the candidates returned by

Figure 2.12 – Examples for top 3 ranking with 11 and 25 color terms. Note that some of the errors which occur when using 11 color names are resolved when using 25 color names.

the method. A rank-1 score is then computed which denotes the expectation of the correct match.

Table 2.3 shows the performance comparison using different color descriptors on the Market-1501 dataset. The color descriptor with 11 dimensions achieves the mAP score of 12.02% and rank-1 score of 31.80%. The performance improves by increasing the number of color names bins. The best results are obtained using 25 color names with a mAP score of 13.45% and rank-1 score of 34.65%. We have also run the same experiment with the 16 dimensional color name representation proposed in [189] for the task of person re-identification. We found the results to be inferior to ours. In Fig. 2.12 an example of queries with 11 and 25 color names are provided. The 11 color name representation fails to distinguish between several color tones which are better described in the 25 dimensional representation.

2.5.5 User preference experiment

In Fig. 2.2 we illustrated the importance of our proposed method for the computation of extended color name sets when compared to the *naive* approach, which directly applies PLSA to the extended color name set. We have designed two psychophysical experiments to quantify the difference between our method and the naive approach and the difference between our method and the two baselines

Table 2.3 – Re-id performance comparison of different color descriptors on the Mart-1501 dataset. The best results are obtained using 25 color names.[1] Yangyang's 16-d color features for re_identification with our experiment settings [189].

	11	15	16[1]	25
mAP	12.02	12.85	10.93	13.45
r = 1	31.80	33.46	30.34	34.65

Figure 2.13 – Setup for our psychophysical experiment. Color patches and color names were presented on a calibrated CRT monitor and the observer pressed buttons on a gamepad to decide whether the color patch was well described by the name or not. The background was mid-grey and a reference white was provided by a D65-colored frame.

LABCN and RANDOM. In the experiment we focus on the color names where the two methods do not agree.

We performed the forced choice psychophysical experiment where observers had to decide whether a given color patch was described by a given color name (shown in writing at the top) or not. The stimuli were presented on a calibrated CRT monitor (Sony GDM F500-R) run by a Visage Mk1 stimulus generator. The screen boundary consisted of a 5cm wide frame that acted as reference white (D56, 64 Cd/m^2). The experimental setup was as follows: after 2 minutes of dark adaptation, observers were presented with an image patch centered on a mid-gray screen, and a color name written on top. Their task was to press the left or right buttons of a gamepad to decide whether the name described the color of the patch correctly (yes-no choice). Once observers made their choices (there were no time constraints), the screen was refreshed, the next patch and color name appeared and

Table 2.4 – User communication results on Munsell patches of 39 color names.

	Ours	Naive
Accuracy	55.5%	45.5%

Table 2.5 – User communication results on Munsell patches of 25 color names.

	Ours/LABCN	Ours/RANDOM
Accuracy	57.7%/42.3%	53.0%/47.0%

the trial was repeated. Setup for our psychophysical experiment is shown in Fig. 2.13. The patches were randomly selected from a list of 162 samples where there was disagreement between the two methods. The color names were obtained from each of the two methods tested, ours and naive method. There were 10 subjects (university students, eight male and two female) and they all had normal color vision tested by the Ishihara color-blindness test. We also made sure all of them were familiarized to the same color terms before the experiment by showing them a series of cards with images of objects (obtained from Google images) categorized under the same color name. We did two runs of the experiment per subject. The first run was considered "training" and was discarded. The results of the second run are presented in table 2.4, which shows the percentage of times subjects preferred each method's categorization. Our method's solution was preferred 10% more than that of naive method.

In the second psychophysical experiment we compare the ranking which is computed with our method against the two baselines LABCN and RANDOM. Note that the previous psychophysical experiment was based on 39 color names, but since in that case the order is not significant (all methods will finally pick the same 39 color names) we have performed this experiment with 25 color names. Other than that the setup is the same as in the previous experiment. Results as summarized in table 2.5 show that agreement with our method is around 15.4% higher than the LABCN baseline and 6.0% higher than RANDOM baseline.

2.6 Discussion and Conclusions

Color description is an important part of image understanding. It is a difficult problem because colors vary due to accidental events such as shadow, shading,

specularities, viewing angle, image compression, etc. The most popular approach to address this problem is by means of photometric invariants derived from reflectance models. However, it was found that descriptors based on color names often obtained better results for computer vision applications. Color names are therefore applied in many applications such as image classification, object detection, action recognition, texture recognition, object tracking, and person re-identification.

Traditionally color name mappings are restricted to the eleven basic color name terms. In this chapter, we proposed a method to compute the color name mappings for large color name sets. For this purpose we collected a new data set of 28 additional color names. We have shown that a naive extension of the color name descriptor leads to unsatisfying mapping of colors to color names. To solve this problem we propose an iterative scheme to extend the color name descriptor. In addition, we propose a method to rank the additional color names. Using the ranking we can compute color name representations of arbitrary length. In our experiments we evaluate the impact of increasing the number of color names to visual tracking, person re-identification, and image classification. In all cases adding color names was found to improve the results significantly. In addition, two psychophysical experiments show that our approach has a larger agreement to human users of labeling patches with color names.

The recent advances of Deep Learning have influenced computer vision research greatly. Driven by the availability of large datasets and improved hardware (GPU computation) these algorithms can jointly learn feature representations and classifiers [84, 88]. They have been shown to be successful on many computer vision application [145] and outperform hand crafted features. They have also been shown to effectively learn attributes of objects, including color, and texture attributes [105, 141, 172]. However, due to the absence of large color name datasets color naming with deep networks has been limited to the eleven basic color names [173]. The dataset proposed in this chapter could be used to train networks for larger color name sets. In that case the discussion on the difficulties of extending the color name set beyond the basic color terms (see Section 2.4.3) should be taken into account when designing the loss function of the network. A simple softmax loss would enforce the probabilities over the color names to sum to one, and would most probably demonstrate some of the shortcomings we have shown in this chapter that 'naive' approaches have.

3 Weakly Supervised Domain-Specific Color Naming Based on Attention*

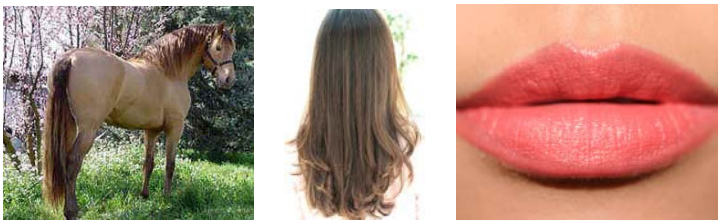
3.1 Introduction

Color is a basic characteristic of visual objects in the world. As one of the important features of visual data, colors are crucial in understanding the world, and they can be used to distinguish one object from another in our daily life. Humans use color names to refer to a specific color and to communicate color information with other humans. Examples of color names are 'blue', 'crimson' and 'amber'. Computational color naming aims to identify color names in images; this is usually done by learning a mapping between color values and color names. Computational color naming is important for applications in human computer interaction, including online shopping, fashion analysis, image retrieval and person re-identification [23, 100] and Chapter 2.

For the purpose of this chapter, we divide computational color naming models in methods which are trained in a supervised or semi-supervised manner. Supervised methods are based either on labeled color patches [9, 117] or on pixel ground-truth masks, providing the color names for all the relevant items in the image [23, 100]. The work of Van de Weijer et al. [165] proposed a method to learn color names from images retrieved from Google in a semi-supervised manner. We refer with *semi-supervised* to the fact that the provided label describes the color of the principal object in the image, but no information on the exact pixels which are described by the label is given. An advantage of semi-supervised methods is that they reduce the label effort significantly. However, the existing unsupervised methods [23, 100, 165, 173, 197] still require pixel masks at the testing phase. The methods are therefore semi-supervised at training, but supervised at test time.

The vast majority of the existing color naming approaches use the eleven basic color terms [117, 165, 197] which were defined in the seminal study of Berlin and Kay [50]. Even though these color names are widely used, many applications apply different domain-specific color names. In Fig. 3.1 several examples of color names within various applications are provided: a 'champagne' colored horse, 'almond' colored hair and 'coral red' lips. Because different application domains use different

*This chapter is based on a publication in the international Conference of Pattern Recognition (ICPR 2018) [194].



(a) (b) (c)

Figure 3.1 – Example images of domain-specific color names: (a) 'champagne' colored horse, (b) 'almond' colored hair and (c) 'coral red' lips.

sets of color names, the laborious labeling process would need to be performed repeatedly. Therefore, in this chapter we aim for a method which can learn from weakly labeled data, and which does not require any supervision at testing time. Learning from weakly labeled data has been studied before for image classification [115, 183], image segmentation [67, 131], saliency detection [170], object detection [13, 20], and object recognition [35, 82, 122, 175].

To address the drawback of the deep learning approaches for color naming, we propose a weakly-supervised deep learning framework based on attention. The main contribution of this chapter is a new two-branch network design for color naming based on attention, which is capable of automatically discovering relevant regions related to weak image labels, and simultaneously learn a mapping between color values and color names. In addition, we collect a large-scale dataset using a Web image search engine, which contains 11 basic color naming images for 4 categories, and a dataset for domain-specific color naming which includes color names for horses, eyes colors, lips colors, and the tomato growing stages. Experiments show that our attention network correctly identifies the relevant image regions, and at the same time learns a mapping from image values to color names.

3.2 Attention Modulation for Color Naming

The aim of this chapter is to predict the color name that best describes the principal object in the image. The method is to be trained from weakly-labeled data, which means a color name label is provided for the image, but that no segmentation mask or bounding box is provided to identify the principal object. We assume that images contain a single principal object which can be described by a single color name.

To train from the weakly-supervised data the method has to perform two tasks:

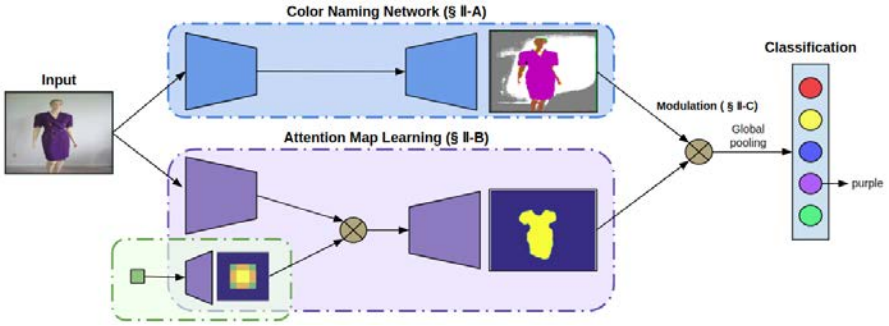


Figure 3.2 – Overview of our proposed framework for weakly supervised color name prediction. Our model is capable of automatically discovering correct regions of interest for image-wise color label predicting and simultaneously providing an end-to-end mapping between color values and color names.

identify the principal object in the scene and predict the color name which best describes its color. In the design of the network, which is provided in Fig.4.1, we have two parallel branches, one for each task. The first branch is a shallow convolutional neural network which aims to predict a pixelwise color name map. The second branch computes an attention map which identifies the regions which contain the relevant color name information. The two branches are combined with a modulation part which combines the automatically learned attention map with each channel of the predicted color naming map.

3.2.1 Color Naming Network (CN-CNN)

The color naming network takes a color image $I \in \mathbb{R}^{H\times W\times 3}$ as an input and produces an estimate of the color name distribution $Y \in \mathbb{R}^{H\times W\times C}$ where C is the number of color names. The structure of the CN-CNN is illustrated in Fig. 3.3 (see also top row of Table 3.1). Specifically, first it passes through several convolutional and pooling layers after which we apply deconvolution to arrive back at the original image size. Then the features after one convolutional layer from the original input are concatenated to the part after the deconvolution layer with a skip layer [146]. One soft-max layer is then added to normalize the distribution of all dimensions.

Before training the CN-CNN in an end-to-end fashion, we initialize the network by using the weak-labels of the images. We train the CN-CNN by minimizing a

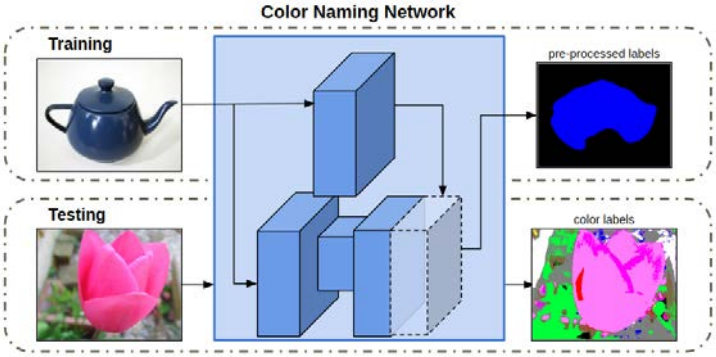


Figure 3.3 – The structure of the color naming network (CN-CNN).

Table 3.1 – Details of our network

CN-CNN	Type	Conv+BN+Relu	Maxpool	Deconv+BN+Relu	Conv+BN+Relu	Concat	Conv	Softmax			
	Filters	72		72	72		11				
	Stride or Upsample	1*1	3*3 / 2	3*3 / 2	1*1		1*1				
	Output	227*227	113*113	227*227	227*227		227*227				
VA-CNN	Type	FCN-8S (1-31)	FC+Relu	FC+Relu	Deconv	Modulation	FCN-8S (36-43)	Conv+Relu	Modulation	Avepool	Softmax
	Filters		512	512	1			1			
	Stride or Upsample		7*7	1*1	8*8 / 4			3*3		Global	
	Output		8*8	8*8	8*8			227*227		1*1	

weighted cross entropy loss:

$$L = \sum_i \sum_y \sum_x m_i(x,y) \log Y_i(x,y,l_i), \tag{3.1}$$

where the summations are over the spatial coordinates x and y and image indexes i, and l_i is the ground truth label of image i and m_i is a mask. Since not all the pixels in the image are correctly described by the weak-label of the image, we use a mask which is computed with a standard saliency algorithm [51] that has a value of one for the salient part of the image and zero otherwise. This mask provides a very rough estimate of the important parts of the image, but we found this to be sufficient to provide an initialization of the network. Note that this loss is not used when training end-to-end with the whole two-branch network.

3.2.2 Visual Attention Network (VA-CNN)

Direct training on images with only the weak-labels is expected to lead to unsatisfactory performance. To further improve the visual attention network (VA-CNN), which should identify the relevant parts of the principal object in the image. To

obtain this, we propose to use an attention network branch (in purple in Fig. 4.1). This branch has a color image as input and aims to compute an attention map $A \in \mathbb{R}^{H \times W \times 1}$ as output. The architecture of the attention network is based on the popular fully convolutional semantic segmentation network FCN-8s[146] followed by one ReLu layer (see for details Table 3.1). The final output of the network provides the importance of each pixel for the task of color naming.

One drawback of FCN is that it cannot learn the spatial prior. However, the principal salient object in the image is most likely in the center of the image [68]. We therefore add a spatial prior layer into the visual attention network. This layer exists of a single pixel input with value equal to one, followed by a deconvolution layer which outputs the spatial prior. This spatial prior is then used to modulate the downsampled features of the FCN network, in the same way as the modulation layer which is explained in the following section. By backpropagating, the weights of this deconvolutional layer learns the spatial prior of the dataset.

Attention model have been applied in various network architectures. They are used to attend to the relevant region in the image related to text in captioning [186] or visual question answer networks [185]. Also they have been studied in various computer vision tasks, including image recognition [175], and saliency detection [85].

3.2.3 Modulation Layer

In this weakly supervised learning system, neither the ground-truth color names of each pixel nor the ground-truth of the confidence map is provided for directly training the CN-CNN or VA-CNN. We therefore propose an indirect method to jointly train both branches with only weak-labels. For this purpose the pixel wise color name predictions Y and the visual attention map A with a modulation layer to output the final color name prediction for the image $Z \in \mathbb{R}^C$. The modulation layer does a channel wise multiplication of the feature maps of Y with the attention map A according to

$$\hat{Y}_k\left(x, y, l_i\right) = A\left(x, y\right) Y_k\left(x, y, l_i\right), \tag{3.2}$$

where Y_k denotes the k-th channel with $k = \{1, \cdots, C\}$; A is the attention map; Score aggregation is then performed on $\hat{Y}$ using average pooling to predict image-level score $\hat{y}$ for the k-th category.

The back propagation for the modulation layer is as follows:

$$\frac{\partial(\hat{Y})}{\partial(Y_k)} = A, \tag{3.3}$$

Figure 3.4 – Examples of four categories ('car','dress','pottery' and 'shoes') in eleven basic color are shown.

$$\frac{\partial(\hat{Y})}{\partial(A)} = \sum_{k=1}^{C} Y_k. \tag{3.4}$$

3.2.4 Network Training

Both CN-CNN and VA-CNN can be trained by minimizing the cross entropy loss $L(l, \hat{y})$, where l is the ground truth label. We found that it was difficult to train the network jointly, and therefore propose an alternating training scheme. Specifically, after the CN-CNN is trained, we fix this part and fine-tuning the VA-CNN to learn attention map. After several epochs we stop training the VA-CNN branch and freeze it, and change to train the CN-CNN part again, and we repeat this process till the loss converges.

3.3 Color Naming Data Collection

For the purpose of this chapter we collect two datasets: one of domain-specific color names, and one class-specific basic color term dataset. Both datasets are weakly-labeled. **Domain-specific color name dataset:** we collect several domain-specific datasets from Google search engine by using the query of 'color name + object': 5 colors of eyes, 7 colors of lips, 9 colors of horses and tomatoes in 6 growing stages. Then, we manually removed the noisy images. Each class has 40 images for training, 10 for validation and 20 for testing. In total, 50 images for each class of each group for domain-specific color naming learning. The dataset

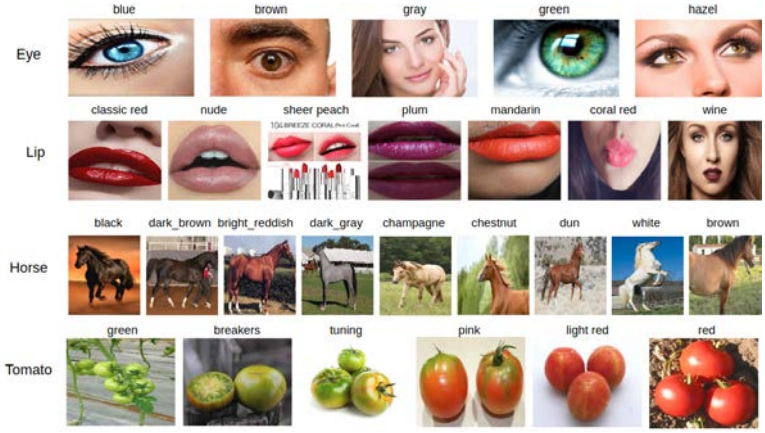


Figure 3.5 – Examples from domain-specific datasets. One example for each domain-specific color name is shown.

is available at https://github.com/yulu0724/AttentionColorName. Examples are shown in Fig. 3.5.

Class-specific basic color term dataset: Since existing methods report on the eleven based color terms we also collect a class-specific dataset for these color names. We collected 2200 images from Google Image on-line by using the query of 'color name'+'object'. We choose the 11 basic color names as the indicated in [11], the difference is that four specific categories 'car','dress','pottery' and 'shoes' are selected as our 'object' class (the same categories as the EBAY color name dataset in Section 3.4.2) to decrease the probability of false positives, and adapt to our method. Hence for red, the query is 'red+car', 'red+dress', 'red+pottery' and 'red+shoes'. Then, we manually removed the noisy images. We retrieve 50 images for each color and object, so 200 images in total for each color name. Four special categories examples for the 11 color names are given in Fig.4.5.

3.4 Experiments

3.4.1 Implementation Details

We implemented our method with Matconvnet framework. The CN-CNN part is first pre-trained using the saliency method [51] to get a rough mask of the principal

Table 3.2 – Comparison of state-of-the-art methods, testing on the EBAY dataset, training with class-agnostic dataset and new class-specific dataset. We indicate with test type which methods are supervised (S) or unsupervised (U).

dataset			pixel_wise					image_wise				
	Method	test type	car	dress	shoes	pottery	**overall**	car	dress	shoes	pottery	**overall**
class-agnostic dataset	PLSA	S	56.00	80.00	77.00	70.00	**70.60**	74.00	85.00	94.00	82.00	**83.40**
	SS_net	S	-	-	-	-	**74.00**	73.18	91.82	91.18	83.36	**84.89**
	Ours	S	51.38	80.27	77.64	71.03	**71.83**	71.32	86.36	88.18	80.91	**81.82**
	Ours	U	-	-	-	-	-	63.64	79.07	81.82	74.55	**74.77**
class-specific dataset	PLSA	S	54.52	82.75	75.37	71.98	**71.15**	69.09	93.64	89.09	87.27	**84.77**
	Classification	U	-	-	-	-	-	66.36	78.18	70.91	72.73	**72.05**
	Ours	S	57.88	85.35	78.32	75.54	**74.27**	73.64	94.55	94.55	86.36	**87.27**
	Ours	U	-	-	-	-	-	72.72	94.54	84.55	87.27	**86.59**
	Human	-	-	-	-	-	**-**	-	-	-	-	**88.98**

(a) (b)

Figure 3.6 – (a) Example image from EBAY labeled with the color name 'green' and (b) the ground truth mask of the image identifying the pixels which are related with the color name.

object as explained in Section 3.2.1. Next we perform alternating training of the two branches using the weakly labeled data. The newly added layers in our network are initialized with Xavier method. All the training images are resized to 227×227 in our experiments for both of the CN-CNN and VA-CNN. Both of the models are optimized using Stochastic Gradient Descent (SGD) method with a batch size of 32 and 6 respectively, and a momentum of 0.9. The learning rate is set to 0.01 initially and divided by 10 after 20 epochs.

3.4.2 Color Naming from Weakly Labeled Data

Most existing methods on color naming are trained with the eleven basic color terms. We start with an ablation study to evaluate our method, and next compare it to other methods.

We compare results on the EBAY dataset which contains a total of 440 images,

Table 3.3 – Comparison of our model learned using different components on the EBAY dataset. We abbreviate attention, centric information and alternating learning as AM, C, AL.

	Accuracy
Ours	55.45
Ours+AM	84.09
Ours+AM+C	84.77
Ours+AM+C+AL	86.59

Table 3.4 – Color naming results on Eye, Lip, Horses and Tomato dataset respectively comparing to using classification network (pre-trained AlexNet).

Dataset	Ours										Classification
Eye	blue	brown	gray	green	hazel					**overall**	**overall**
Accuracy	65.00	85.00	65.00	70.00	10.00					**59.00**	**49.00**
Lip	classic_red	sheer_peach	coral_red	mandarin	nude	plum	wine			**overall**	**overall**
Accuracy	65.00	40.00	55.00	70.00	60.00	35.00	65.00			**55.72**	**45.00**
Horse	black	dark_brown	bright_reddish	dark_gray	champagne	chestnut	dun	white	brown	**overall**	**overall**
Accuracy	80.00	45.00	15.00	85.00	80.00	70.00	30.00	90.00	45.00	**60.00**	**58.89**
Tomato	green	breakers	tuning	pink	light red	red				**overall**	**overall**
Accuracy	55.00	25.00	60.00	35.00	65.00	80.00				**53.33**	**50.83**

consisting of ten images for the eleven color names for four different categories (cars, shoes, dresses, and pottery). All images come with a mask image which identifies the pixels which belong to the named object. This Evaluation is only performed for the pixels in the mask, see example in Fig. 3.6.

Ablation Study: We perform an ablative study to analyze the contribution of the critical components of our proposed model. The results are on our class-specific dataset and summarized in Table 3.3. They show a drop of about 2% without applying alternating learning, 2.5% drop without further adding centric information, and a significant drop when removing all of these, which demonstrates the relevance of the components we propose.

Comparison with the State-of-the-art: In Table 3.2 we compare our results testing on the EBAY dataset with previous related work: PLSA [165], SS_net [173] with different training data. All methods train from weakly labeled data, however it is important to stress that only our method can be applied unsupervised at test time, while the other methods need a mask of the object (this is indicated with U and S in Table 3.2). We provide results for pixel-wise accuracy which is defined as the percentage of correctly classified pixels, and image-wise accuracy which is defined as the percentage of images which is correctly labeled. For the pixel-wise accuracy we only use the CN-CNN network.

When comparing the methods based on the class-agnostic data set, we see that

Figure 3.7 – Examples of Attention map from Eye, Lip, Horse and Tomato datasets.

our method struggles to learn a good attention model. This is to be expected since there are many possible objects in both the train and test dataset. However, when we use the class-specific dataset with similar objects as in the EBAY dataset (cars, shoes, dresses, and pottery) results improve significantly. Our pixel-wise accuracy improves with 3% over PLSA. On the image-wise evaluation we obtain even 86.59% which is higher than any of the other methods which require a mask at test time. Our results of 87.27% are obtained when we use the ground-truth segmentation mask as our attention map; note that all other methods (indicated with S) also use this mask at test time. As a comparison we also provide results with an image classification network; we use a pre-trained AlexNet and finetune on the training dataset. The results are more than 10% lower than our method.

Finally, we compare our testing results with human evaluation on EBAY. Humans are asked to choose the main color label for the object in each image; eight candidates without color blindness are asked to give one color label for each of 110 randomly chosen images from the EBAY dataset. We compute testing accuracy comparing to the ground truth of EBAY dataset and report the average accuracy (88.98%) as the human evaluation baseline. This shows that our results have narrowed the gap with humans from 4% to around 2%.

3.4.3 Domain-Specific Color Naming

The main objective of this chapter is to provide a method which can be applied to new sets of domain-specific color names with only weakly labeled data. Here we evaluate our method on the four groups from our domain-specific dataset. We

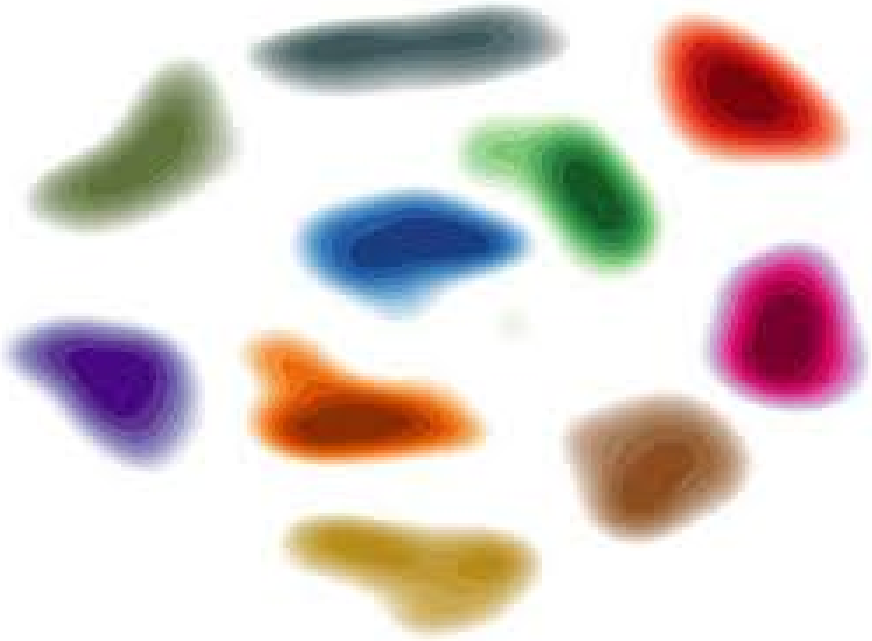
compare to the previously discussed classification network; the other methods cannot be applied in this setting.

Table 3.4 gives the results of color naming for the Eye, Lip and Horse color and Tomato growing stage. Our method outperforms the classification network on all groups. The attention network manages to identify the relevant objects as can be seen from the attention maps of some testing images in Fig. 3.7, where highlighted yellow regions indicate high-interest parts, and blue means low-interest parts. The smaller gains on the Horse and Tomato groups can be explained by the fact that the main object occupies most of the image and in that case the classification network also manages to extract the relevant color name.

3.5 Conclusions

In this chapter we have proposed a new network for the learning of domain-specific color names from weakly labeled data. This two-branch network learns, in an end-to-end fashion, a color name probability map for each pixel and an attention map. When joined, these maps result in a color name prediction for the image. Our method is the first color name method which does not require hand-labeled masks at testing time. Results show that the attention maps correctly identify the relevant image regions and that the network successfully learns domain-specific color names. In addition, we show that the pixel-wise and image-wise predictions of the network obtain state-of-the-art results on the EBAY dataset.

Deep Embeddings Part II



How do we transfer knowledge for embedding networks?

4 Learning Metrics from Teachers: Compact Networks for Image Embedding*

4.1 Introduction

Deep neural networks obtain impressive performance for many computer vision applications, some of which have subsequently been turned into products for the general population. However, the applicability of these techniques is often limited by their high computational cost. To reduce network traffic and server costs, as well as for scalability, it is desirable to place as much of the computation as possible on the end-user side of the application. However, this is often a mobile device with limited computing power and battery life, and thus cannot compute large networks in real-time. This creates a strong demand for methods that transfer the knowledge from large networks to smaller ones, but without a significant drop in performance.

One important class of deep networks learns feature embeddings. To be successful, feature embeddings must preserve semantic similarity, i.e. items deemed similar by users must be close in the embedding space, despite significant visual differences such as point of view, illumination, or image quality. To bridge this gap between the semantic and visual domains, pairs or triplets of related and unrelated items are used to teach the network how to organize the output embedding space [24, 58, 169]. Embeddings were found efficient on the tasks of out-of-distribution detection [111] and transfer learning [142]. Furthermore, embedding networks are essential for computer vision, as evidenced by the large variety of tasks in which they are used, including feature-based object retrieval [45], face recognition [140], feature matching [25], domain adaptation [143], weakly supervised learning [171], ranking [169], or zero-shot learning [154].

Large networks are known to provide excellent feature embeddings [134, 140], but are often impractical for real-life applications, as mentioned before. To obtain efficient neural networks, research has focused on two main research directions: network compression and network distillation. Network compression reduces the number of parameters in the network [47, 90], while network distillation uses a teacher-student setup in which a, typically large, teacher network is used to guide a small student network [16, 57]. This is done by using a loss function that

*This chapter is based on a publication in the International Conference of Computer Vision and Pattern Recognition (CVPR 2019) [195].

minimizes cross-entropy between the outputs of the student and teacher network for classification. The main idea underlying network distillation is that uncertainty in the estimates of the teacher network, e.g. about whether an image contains a cat and dog, provides relevant information for the student. There are several differences between network compression and knowledge distillation. First of all, the underlying assumption of network compression is that the knowledge of the network is in the weights, whereas knowledge distillation assumes that the knowledge of the network is in the activations which arise from particular data. A second important difference is that compression algorithms typically end up with a similar network architecture than the initial large network but with less parameters (i.e. same number of layers, and layer types). In contrast, network distillation puts no restrictions on the student network design. Therefore, we focus on network distillation techniques for the efficient computation of feature embeddings with small networks.

In this chapter, we use network distillation to obtain efficient networks to learn feature embeddings. We propose two different ways of teaching metrics to students: one based on an *absolute teacher*, where the student aims to produce the same embeddings as the teacher, and one based on a *relative teacher*, where the teacher communicates only the distances between pairs of data points to the student. Using the CUB-200-2011 (birds), Cars-196 and Stanford Online Products datasets, we show that network distillation can significantly improve retrieval performance compared to directly training the student network on the data. We also found that the relative teacher consistently outperforms the absolute teacher. We evaluate various aspects of knowledge distillation, namely the usage of hint and attention layers, and the possibility to train from unlabelled data. We also show that a teacher with access to high-quality images can be used to improve embeddings learned with a student network with access to low-quality images.

4.2 Related work

There is a large number of works on metric learning, see for example the survey [86]. Here we focus on metric learning using deep networks.

Metric Learning Initially metric learning with deep networks was based on Siamese architecture with contrastive loss [24]. Later Triplet networks were proposed which allow more local modifications of the embedding space, and do not require that all the observations of the same class collapse to the same point [58, 169]. The progress of Siamese and Triplet networks has been hampered because of the pair (or triplet) sampling problem which arises from the huge potential space from which pairs can be sampled. For instance, in a dataset with N samples, N^2 pairs could be possibly

sampled, and it is therefore unfeasible to consider all of them. Therefore, hard negative mining was proposed to focus only on the pairs which induced the highest loss [149], with the expectation that the network would learn the most from them. Unfortunately, this led to severe overfitting in many cases, and semi-hard negative mining was introduced as a solution [140]. However, both hard and semi-hard negative mining have a high computational cost, which led several authors to limit the hard negative mining process to the current mini-batch [102, 152, 154, 171].

Network Distillation Bucila et al. [16] compress a large network into a small one. Their method aims to approximate a large teacher network with a single fast and compact student network. This was further improved by Hinton et al. [57] by moving the teacher signal from the logits (just before the softmax) to the probabilities (after the softmax), and introducing temperature scaling to increase the influence of small probabilities. With these improvements, they achieved some surprising results on MNIST, and also showed that the acoustic model of a heavily used commercial system could be significantly improved by distilling the joint knowledge of an ensemble of models into a single one. FitNet [138] introduced hint layers with additional losses on intermediate layers of the network to communicate knowledge of the teacher to the student. They show that this helps to train deep and thin networks, which cannot be trained from scratch without teacher supervision. In addition, these students can outperform the teacher network while using less memory. Zhang et al. [201] show that a group of students, without a teacher, which are jointly trained with similar losses as between teacher and student, can outperform standard ensemble learning. Network distillation can also be used to compress multiple teachers into a single student network [39]. Most literature on network distillation focuses on image classification, but recently several works have investigated applying the theory to object detection [19] and pedestrian detection [147].

Only two works have previously addressed knowledge distillation for embeddings. Chen et al. [22] bring the 'learning to rank' technique into deep metric learning for knowledge transfer. It is formalized as a rank matching problem between teacher and student networks. Their list-wise loss can easily overflow due to the product operation when computing the probability of the permutation, which severely limits the batch size that can be used. PKT [129] applies a different approach where they model the interactions between the data samples in the feature space as a probability distribution. In our experiments we show that our proposed relative teacher outperforms the DarkRank and PKT significantly.

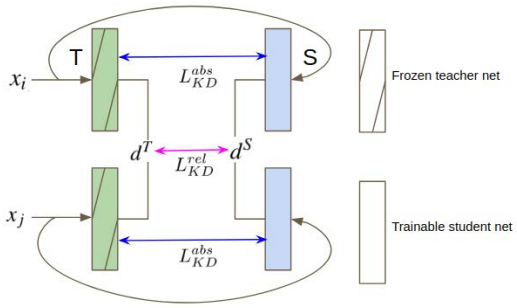


Figure 4.1 – Graphical illustration of the two knowledge distillation losses we propose for metric learning. L_{KD}^{abs} aims to minimize the distance between the student and teacher embedding of the same image. L_{KD}^{rel} compares the distance in the embedding of the teacher between two images, with the distance of the same two images in the student embedding. It aims to make the two distances as similar as possible.

4.3 Preliminaries

In this chapter, we will apply network distillation to metric learning networks. This section will briefly introduce both.

4.3.1 Metric Learning

A fundamental step in most computer vision applications is transforming the initial representation of the images (i.e. pixels) into another one with more desirable properties. This process is often denoted as *feature extraction*, and projects the images to a high-level representation that captures the semantic characteristics relevant to the task. How images are organized in this high-level representation is crucial to the success of many applications. For example, image retrieval, k-NN or Nearest Class Mean classifiers are directly based on the distances between these high-level image representations. Metric learning addresses this problem and intends to map the input feature representations to an embedding space where the L2 distance correlates with the desired notion of similarity. In this work we will focus on deep, or end-to-end, metric learning, where the whole feature extraction network is trained jointly to generate the best possible representation.

Siamese networks map data to an output space where distance represents the

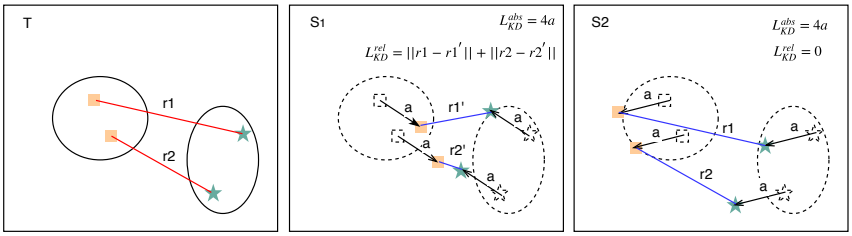


Figure 4.2 – Illustration of difference between absolute and relative teacher. (left) Example of four data points in the embedding space of teacher. We consider two samples from two classes (indicated by square and star). (middle and right) show the absolute and relative loss for two student embeddings $S1$ and $S2$ (the teacher location of the points is given in dashed lines). The (right) embedding is preferable since it is exactly equal to the teacher (except for a translation). This is only appreciated by the relative teacher, whereas the absolute teacher assigns equal loss to both.

semantic dissimilarity between the images [15, 24]. Triplet networks were proposed by Hoffer et al. [58] based on the work of Wang et al. [169]. In contrast to Siamese networks, they use triplets formed by an anchor (x_a), a positive instance (x_p) and a negative instance (x_n), as input. The anchor and the positive instances correspond to the same category, while the negative instance is from a different one. The objective is to guarantee that the negative instance is further away from the anchor than the positive (plus a margin m). The Triplet loss is given by:

$$L_T = \max(0, d_+ - d_- + m), \tag{4.1}$$

where d_+ and d_- are the distances between the anchor and the positive and negative instances respectively. The Triplet network imposes only *local* constraints on the output embedding, which can simplify convergence when compared to the Siamese network, reportedly harder to train. In the experiments we will show results of network distillation for embeddings learned with triplet losses.

4.3.2 Network Distillation

Network distillation [57, 138] aims to transfer the knowledge of a large teacher network T to a small student network S. The objective for network distillation for

classification networks is defined as:

$$L_{KD} = H(y_{true}, P^S_{\tau=1}) + \lambda H(P^T_\tau, P^S_\tau), \tag{4.2}$$

where λ is used to balance the importance of two cross-entropy losses H: the first one corresponds to the traditional loss between the predictions of the student network and the ground-truth labels y_{true}, and the second one between the *annealed* probability outputs of the student and teacher networks. This loss encourages the student to make similar predictions as the teacher network. The information of the teacher P^T_τ could be more valuable than the ground truth y_{true} for the student network, because it also contains information of which classes could possibly be confused with the true label for a particular image. More precisely, P^T_τ and P^S_τ are:

$$P^T_\tau = \text{softmax}(\frac{a_T}{\tau}),\ P^S_\tau = \text{softmax}(\frac{a_S}{\tau}), \tag{4.3}$$

where a_S and a_T are the (pre-softmax) activations of the student and teacher networks respectively, and temperature τ is a relaxation which is introduced to soften the signal arising from the output of the networks. It was found that for complex classification tasks $\tau = 1$ obtained good results [19]. $P^S_{\tau=1}$ is equal to the output of the standard student network without any temperature scaling.

4.4 Distillation for Metric Learning

Wide and deep networks with large amounts of parameters are known to obtain excellent results [150], however they are very time consuming and memory demanding. Network distillation is proven to be one of the solutions to handle this problem in the classification field [57]. In this section we extend the theory of knowledge distillation to networks that aims to project images into an embedding space. In addition we will discuss the incorporation of hint and attention transfer between student and teacher.

4.4.1 Knowledge distillation for embedding networks

Traditional network distillation has focused on networks which perform classification, and are trained with a cross-entropy loss [57, 138]. During training the output class distribution produced by the student is enforced to be close to that of the teacher. This is shown to obtain much better results than directly training the student on the available data; the main reason for this performance difference is that confusions between classes of the teacher reveal relevant information to the

student, thereby providing a richer training signal than ground truth labels would provide [57].

Here we extend the technique of knowledge distillation to networks that are used to project input data into an embedding (from now on called *embedding networks*). These embeddings are then typically used to perform distance computation. For example, to provide a ranked list of similar data (ordered according to the distance). For knowledge distillation it is important to consider what is the knowledge that is contained in the embedding network. One could consider the actual embedding (meaning the coordinates of the embedding) to be the knowledge of the network. Another point of view would be to consider the distances which are computed based on the embedding network to be the actual knowledge, since this is actually the main purpose of the embedding network. We will consider both these points of view and design two different teachers: one teacher, called *absolute teacher* which teaches the exact coordinates to the student and one teacher, called *relative teacher* which only teaches the distance between data pairs to the student.

In the first approach, the absolute teacher, we directly minimize the distance between the student (F^S) and teacher (F^T) embeddings. This is done by minimizing:

$$L_{KD}^{abs} = \left\| F^S(x_i) - F^T(x_i) \right\|, \tag{4.4}$$

where $\|.\|$ refers to the Frobenius norm.

As a second approach, we consider the relative teacher, which enforces the student network to learn any embedding as long as it results in similar distances between the data points. This is done by minimizing the following loss:

$$L_{KD}^{rel} = \left| d^S - d^T \right|, \tag{4.5}$$

where d^S and d^T are, respectively, the distances between the student and teacher embeddings of images x_i and x_j:

$$\begin{aligned} d^S &= \left\| F^S(x_i) - F^S\left(x_j\right) \right\|, \\ d^T &= \left\| F^T(x_i) - F^T\left(x_j\right) \right\|. \end{aligned} \tag{4.6}$$

The minimization loss in Eq. 4.5 is equal to the loss used in the classical problem of multidimensional scaling (MDS) [27]. There the dissimilarities between points is known and the goal is to find coordinates for the points in some (low-dimensional) space where the dissimilarities between the points is equal to their dissimilarity.

A graphical illustration which shows the relevant distances that are used by the absolute and relative teacher is provided in Fig. 4.1. The teacher networks are frozen

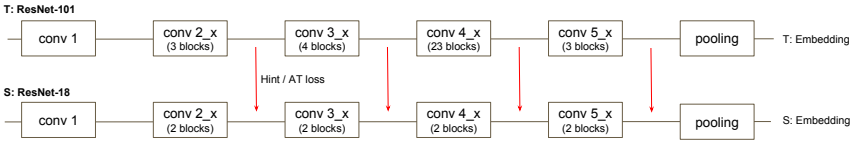


Figure 4.3 – Schematics of teacher-student hint/attention transfer.

during training of the student network. The absolute teacher minimizes the distance between the student and teacher embedding for each training sample. In case of the relative teacher, one should consider pairs of data points, since during training the student network is optimized to obtain similar distances between instances of data points.

As reported by several other authors [57, 201], we train the student network by simultaneously considering the standard metric learning loss L_{ML} (see Eq. 5.3) and the loss L_{KD}^{T} imposed by the teacher, according to:

$$L = L_{ML} + \lambda L_{KD}^{T}, \tag{4.7}$$

where $T \in \{abs, rel\}$ and λ is a trade-off parameter between the different losses, which is learned by cross validation.

In Fig. 4.2 an illustration of the two distillation losses is given for two different student embeddings, indicated by $S1$ and $S2$. The embedding $S2$ is preferable because it is equal to the teacher embedding except for a translation which does not influence the ranking of data points. The S1 embedding actually changes the relation between samples, and would not obtain similar results as the teacher network. However, if we consider the absolute loss for these two scenarios we see it assigns equal loss to both embeddings. The relative loss does correctly assign a lower (zero) loss to the S2 embedding. By focusing on the relevant parameter (namely the distance), we expected that relative teachers are able to better guide the student to a similar embedding than the student network.

4.4.2 Learning from hints and attention

In this section we consider two techniques that have shown to improve results for distillation of classification networks. The techniques we consider are: the introduction of hint layers [138] and the usage of attention [198]. Both were proposed to improve the learning of student networks. We are interested to know if these techniques also generalize to knowledge distillation for embedding networks.

Romero et al. [138] propose to improve knowledge distillation by introducing an additional loss on intermediate representations learned by the teacher (called hints). The loss which incorporates the hint layers is given by:

$$L_{hint} = \left\| F^S_{hint}(x_i) - F^T_{hint}(x_i) \right\|, \tag{4.8}$$

where $F^T_{hint} \in R^{w \times h \times d}$ where w, h and d are dimensions of the activation map of the hint layer.

In this work [138], they first train the network until the hint loss, and then train the whole network only based on the distillation loss. In contrast, we propose to learn with both losses simultaneously, as was also done in [19, 147]. Combining the knowledge distillation loss of either the absolute or relative teacher we would obtain as a final objective function:

$$L = L_{ML} + \lambda L^T_{KD} + \mu L_{hint}, \tag{4.9}$$

where $T \in \{abs, rel\}$ and μ is used to balance the relative weight of the hint loss.

Zagoruyko and Komodakis [198] improve the performance of student networks by forcing them to mimic intermediate attention maps of a powerful teacher network. Attention maps convey what spatial locations in the image are considered relevant to the teacher network for its interpretation. Communicating this information can therefore guide the student network in learning the task at hand.

They propose to compute activation-based spatial attention according to:

$$A^T_{sum}(x_i) = \sum_{l=1}^{C^k} \left| F^T_{kl}(x_i) \right|^2, \tag{4.10}$$

where $F^T_{kl}(x_i) \in R^{w \times h}$ refers to the l-th map of the activation of the k-th layer for image i. Here C^k denotes the number of feature maps in the k-th layer of the teacher net. We use $|.|$ to refer to the pixel-wise absolute value, as a results $A^T_{sum}(x_i) \in R^{w \times h}$. A similar equation is used to compute $A^S_{sum}(x_i)$ from the student activation maps F^S. The attention loss is then defined as:

$$L_{AT} = \left\| \frac{A^T_{sum}(x_i)}{\left\| A^T_{sum}(x_i) \right\|_2} - \frac{A^S_{sum}(x_i)}{\left\| A^S_{sum}(x_i) \right\|_2} \right\|. \tag{4.11}$$

This enforces the student to assign its attention to the same locations which were deemed important by the teacher.

The full objective function for the attention based metric learning network

Table 4.1 – Retrieval Performance on the CUB-200-2011 and Cars-196 dataset. 'ML':metric learning loss, 'hint':hint loss; 'AT': attention loss; KD (abs): absolute teacher loss; KD (rel):relative teacher loss.

	CUB-200-2011					Cars-196				
Recall@K	1	2	4	8	16	1	2	4	8	16
Student (ResNet-18)	51.7	63.7	74.2	83.7	90.9	46.7	59.5	71.6	82.3	90.6
PKT [129]	53.1	64.2	75.4	84.6	91.6	46.9	59.9	72.1	82.8	90.8
DarkRank [22]	56.2	67.8	77.2	85.0	91.5	74.3	83.6	90.0	94.2	96.9
ML+KD (abs)	54.9	66.5	76.5	85.0	91.3	70.6	80.7	88.0	93.2	96.0
ML+KD (rel)	**58.0**	**69.0**	**79.4**	**87.8**	**93.6**	**76.6**	**85.4**	**91.2**	**95.0**	**97.3**
ML+KD (abs)+hint	55.0	66.5	76.6	84.9	91.1	71.3	81.2	88.1	92.7	95.9
ML+KD (rel)+hint	57.4	68.8	79.1	87.4	93.1	76.4	85.5	91.3	95.1	97.2
ML+KD (abs)+AT	55.0	66.3	76.9	85.3	91.8	71.1	81.3	88.3	93.1	96.0
ML+KD (rel)+AT	**58.1**	**69.2**	**79.6**	**85.3**	**91.3**	**76.4**	**85.7**	**91.7**	**95.0**	**97.2**
Teacher (ResNet-101)	58.9	70.4	80.7	88.2	93.5	74.8	83.6	89.9	93.8	96.5

becomes:

$$L = L_{ML} + \lambda L_{KD}^{T} + \kappa L_{AT}, \tag{4.12}$$

where κ defines the relative weight of the attention loss.

In Fig. 4.3 we show how the hint and attention layers are incorporated between a ResNet-101 teacher and a ResNet-18 student network. Both hint and attention losses are applied on multiple layers[†]. Results for this scheme will be presented in the experimental section.

4.5 Experimental Results

We show results on several benchmark datasets. Our method is implemented with the PyTorch framework [130]. We will release a GitHub page with code upon acceptation.

4.5.1 Retrieval on Fine-grained Datasets

Datasets: We evaluate our framework for the task of image retrieval on three fine-grained datasets:

- CUB-200-2011: this dataset was introduced in [167]. It has 200 classes with

[†]For the student we take the output of each block, and compare it to the last but one layer for each block of the teacher. The dimensionality of these layers is the same.

Table 4.2 – Comparison on Stanford Online Products dataset.

	Stanford Online Products			
Recall@K	1	10	100	1000
Student (ResNet-18)	61.7	78.6	90.2	96.8
ML+KD (abs)	68.0	82.7	92.1	97.4
ML+KD (rel)	67.7	83.0	92.0	97.2
Teacher (ResNet-101)	69.5	84.4	93.1	97.9

11,788 images in total.

- Cars-196: this dataset contains 16,185 images of 196 cars classes and was introduced in [83].
- Stanford Online Products: this dataset introduced in [154] contains 120,053 images of 22,634 products collected from eBay.com.

Example images of CUB-200-2011 and Cars-196 are shown in Fig. 4.5. We follow the evaluation protocol proposed in [154]. By excluding some classes from the training of the embedding, we can evaluate at testing time how good the embedding generalizes to unseen classes. Therefore, the first half of classes are used for training and the remaining half for testing. For instance, on CUB-200-2011 dataset, 100 classes (5,864 images) are for training and the remaining 100 classes (5,924 images) are for testing. We divide the training set into 80% as training and 20% as validation.

Experimental Details: For these experiments, we use a ResNet-101 as the teacher network and a ResNet-18 as the student network (see also Fig. 4.3). The comparison of the number of parameters of these two networks is shown in Table 4.4. After the average pooling layer, a linear 512-dimensional embedding layer is added and the triplet loss is used for training both teacher and student networks. The Adam [79] optimizer is used with a learning rate of $1e^{-5}$, and a mini-batch of 32 images. We apply hard negative mining [149] on triplet loss. For preprocessing, we follow the previous work [123], we resize all images to 256×256, and crop 224×224 patches randomly. Horizontal flip is used for data augmentation. We fine-tune both student and teacher networks from pre-trained ImageNet models with the same preprocessing. During test time, we only use the 224×224 pixel center crop to predict the final feature representation used for retrieval. The optimal parameters are selected according to the performance on the validation set for all of the experiments. We use the whole training set to retrain with optimal parameters for a fixed number of epochs.

For evaluation, we use the Recall@K metric [154]: each image in the test set

is projected using the trained network, and if one of the K closest images in the embedding space has the same label, it is considered as a positive result. The final score is the fraction of positive results obtained on all the test images. Furthermore, the reported results in all tables are the average over three repeated experiments.

Baselines: We start by considering the results of the student and teacher network in Table 4.1 and Table 4.2. Not surprisingly, the teacher network is able to leverage the additional capacity to learn better embeddings. On the CUB-200-2011 dataset, we obtain a R@1 accuracy of 58.9% for the teacher and 51.7% for the student. This is consistent for the other evaluated recall levels, although the gap narrows with higher K. This shrinking performance gap is mirrored in the Car-196 dataset, with the teacher net attaining a 28.1% better R@1, and a 5.9% better R@16. On the Stanford Online Products dataset, the gap between the teacher and student network is 7.8%.

We also compare our method with the DarkRank method [22] and PKT [129][‡]. The experiments show that the relative teacher network significantly outperforms DarkRank and PKT , obtaining 1.8% and 4.9% more on CUB-200-2011 and 2.3% and 29.7% on Cars-196.

Absolute and Relative Loss: Next we incorporate the additional knowledge distillation losses to the student metric learning objective (indicated by *ML+KD*). Table 4.1 shows that results improve for every dataset and recall level, regardless of the loss used. The performance improvement at Recall@1 is 3.2% and 6.3% respectively for the absolute and relative teacher on CUB-200-2011. On Cars-196 we see a similar behavior, again the relative teacher is outperforming the absolute teacher. The student trained with the relative teacher has an stunning performance gain of almost 30.0%. It is interesting to note that the relative teacher even outperforms the teacher by 1.8% while having less parameters. On Stanford Online Product dataset, both absolute and relative teacher obtain similar results, and outperform the direct training of the student network with 6.0%. In conclusion, the proposed distillation methods consistently manage to improve the performance of student network, especially those trained with the relative teacher.

Hint and attention losses: Here we investigate if hint [138] and attention [198] layers are beneficial for knowledge distillation of embedding networks (see also Section 4.4.2). We combine them with our proposed absolute and relative losses according to Eq. 4.9 and Eq. 4.12. The results are summarized in Table 4.1. We found that adding a hint layer was not stable. This is probably because the hint layer is similar as the absolute teacher forcing the network to learn the exact same embedding as the teacher, and therefore only helps when combined with the absolute teacher. Adding attention layers in general provided a small gain but the gain was

[‡]For these results we used the code made available by the authors.

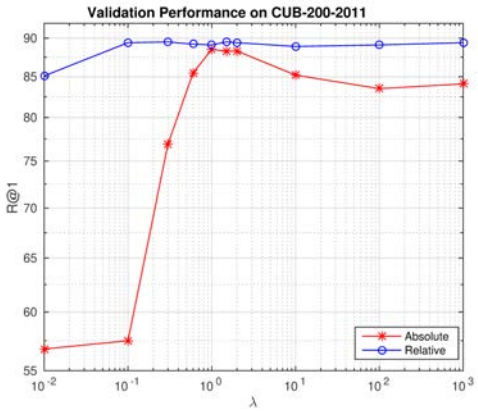


Figure 4.4 – R@1 as a function of λ on CUB-200-2011 dataset.

not as large as reported for classification networks [198].

Sensitivity to λ As shown in Figure 4.4, we compare R@1 performance as a function of different λ values on the validation set of CUB-200-2011 for both absolute and relative teachers. It is noteworthy that the relative teacher has a stable performance on a large range of the trade-off parameter λ, while the absolute teacher only works in a very narrow range. It suggests that in practice the selection of the λ parameter is not that essential for the relative teacher.

4.5.2 Semi-Supervised Learning

One of the interesting properties of network distillation is that it allows for the usage of unlabeled data. This was observed by [201] and we apply this idea here to distillation for embedding networks. The knowledge distillation losses of Eqs. 4.4 and 4.5 do not require any labels. Knowing the estimation of the teacher network for unlabeled data can help the student network to better approximate the teacher network. In addition, the existing problems of pair sampling can be avoided in semi-supervised learning for the student network because for the unlabeled images we do not apply the triplet loss as it requires labels. In the experiments we evaluate the benefit of adding unlabeled data to the student network for embedding learning on the CUB-200-2011 dataset.

We randomly select half of the training images per class as labeled data, and consider the rest as unlabeled data. Thus, here we have two teacher-student learn-

Table 4.3 – Semi-supervised results on CUB-200-2011.

	Recall@K	1	2	4
50% labeled	Student (ResNet-18)	51.0	63.0	74.0
	ML+KD (abs)	51.7	63.2	73.9
	ML+KD (rel)	**56.0**	**66.7**	**77.6**
	Teacher (ResNet-101)	58.1	70.0	80.1
50% labeled +50% unlabeled	(ML+KD (abs))/KD (abs)	53.9	65.2	75.8
	(ML+KD (rel))/KD (rel)	**57.2**	**68.0**	**78.2**
50% unlabeled	only KD (abs)	49.8	60.8	71.0
	only KD (rel)	55.5	67.0	77.4

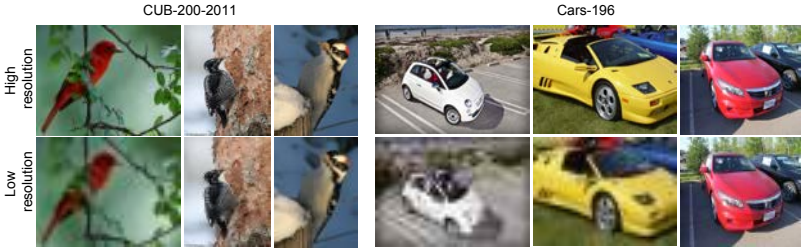


Figure 4.5 – Example images from two fine-grained datasets CUB-200-2011 and Cars-196 used in our experiments. The top row shows examples of high-quality images and the bottom row shows examples of the corresponding low-quality images.

ing mechanisms, one is used on the labeled training set with both the ground truth annotations and information transferred from the teacher, and the other one is applied to the unlabeled training set with only information from the teacher by means of a distillation loss. The results of this experiment can be seen in Table 4.3. The first row (50% labeled) shows our approach using only the remaining labeled data, with similar performance as in the previous experiments: the performance obtained by the relative teacher is closer to teacher network and better than the absolute teacher. In the second row we add the remaining 50% of unlabeled data. This leads to improved Recall@K with both losses, but especially for the relative teacher.

Finally, in the third row, we consider the case where we have access to a trained

Table 4.4 – Parameter Comparison of Different Networks .

Network	ResNet-101	ResNet-18	MobileNet-0.25
Parameters	~48.1 M	~11.3 M	~0.3 M

Table 4.5 – Performance on CUB-200-2011 with MobileNet-0.25.

Recall@K	1	2	4	8	16
Student (MobileNet-0.25)	27.5	35.8	46.0	58.5	70.6
ML+KD (rel)	44.6	56.0	66.4	77.3	86.1
Teacher (ResNet-101)	58.9	70.4	80.7	88.2	93.5

teacher network, but no labelled data at all, to train the student network. Here we would like to highlight the results of the relative teacher, since it manages to increase performance by 4.5% compared to the student network trained with 50% of labeled training data.

4.5.3 Very Small Student Networks

MobileNets [62] are efficient but light-weight networks that can be easily matched to the design requirements for mobile and embedded vision applications. To show the potential of our method on very small networks, we propose to use the MobileNet-0.25 (0.25 is the width multiplier) network as our student network and use ResNet-101 as the teacher network. The number of parameters per network is given in Table 4.4. We can see that the number of parameters of MobileNet-0.25 is almost 40 times smaller than that of the previous student network (ResNet-18) and 160 times smaller than that of the teacher network (ResNet-101).

Table 4.5 shows retrieval performance results on CUB-200-2011 with MobileNet-0.25 as the student network. We can observe that the Recall@K for $K = 1,2$ of the teacher network is almost 2 times higher than the student network. After our relative teacher is applied, the performance gain is 17.1% at Recall@1 and 15.5% at Recall@16 higher compared to the original student model.

4.5.4 Cross Quality Distillation

As an additional experiment we do the distillation of embeddings to transfer knowledge between different domains. This was originally proposed in a classification setting by Su et al. [156] who, in order to improve the recognition on low-quality data, use distillation with a teacher trained with high-quality data. The student then

Table 4.6 – Cross quality results on the CUB-200-2011 and Cars-196 datasets with low resolution and unlocalized object degradations.

		Low Resolution			Unlocalized		
	Recall@K	1	2	4	1	2	4
CUB-200-2011	Student (ResNet-18)	44.4	54.7	65.3	43.6	54.5	66.9
	ML+KD (abs)	45.7	56.8	68.3	43.0	54.5	66.1
	ML+KD (rel)	**46.2**	**57.4**	**68.6**	**45.9**	**57.9**	**69.3**
	Teacher (ResNet-18)	53.7	65.2	74.7	54.8	67.2	78.7
Cars-196	Student (ResNet-18)	37.5	50.0	62.6	54.0	67.3	78.2
	ML+KD (abs)	58.6	70.7	80.7	57.7	70.4	80.6
	ML+KD (rel)	**58.9**	**71.0**	**81.1**	**61.9**	**74.4**	**84.2**
	Teacher (ResNet-18)	71.0	81.2	88.7	67.8	79.1	87.9

is trained with the low-quality data and the guidance from the teacher which has access to the high-quality data. Here we will apply cross quality distillation with the proposed losses for metric learning. Since in this experiment the objective is not to reduce the number of parameters but to bridge a domain gap, we use the same architecture (ResNet-18) for the teacher and the student. To train the embeddings we use triplet loss and, as in the previous experiments, we train the students with two teachers: relative and absolute.

We consider two cross quality distillation experiments on CUB-200-2011 and Cars-196. The first experiment considers *low and high-resolution* images. To get the low-resolution images, we downsample them to 50 x 50 and then upsample them again to 224 x 224 (see examples in Fig. 4.5).

The second experiment considers *unlocalized signal degradation*, where the input images are cropped according to the given bounding boxes for the teachers, but not cropped for the students. The results can be seen in Table 4.6.

It can be seen that incorporating the additional knowledge distillation losses improves the results for most settings, with the relative teachers consistently surpassing the absolute ones, as in the previous experiments.

The improvement by the distillation is more noticeable on the Cars-196 dataset which is also observed in [156]. Since it is a more challenging dataset which has cars with different colors belonging to the same category, the information provided by the teacher becomes more crucial.

4.6 Conclusions

We have investigated network distillation with the aim of computing efficient image embedding networks. We have proposed two losses with the aim to communicate the teacher network knowledge to the student network. We evaluate our approach on several datasets, and report significant improvements: we obtain a 6.3% gain on CUB-200-2011, a 29.9% gain on Cars-196 and a 6.3% gain on Stanford Online Products for Recall@1 when compared to a student network of the exact same capacity which was trained without a teacher network. Furthermore, we apply our distillation loss to MobileNet-0.25. It greatly improves the Recall@1 by 17.1%. We also verify the benefit of adding unlabeled data for embedding learning. In addition, we demonstrate that an embedding learned on high-quality images can be used to improve the student network which has only access to low quality images.

5 Semantic Drift Compensation for Lifelong Learning of Embeddings

5.1 Introduction

Future learning machines should be able to adapt to an ever-changing world. They should be capable of continuously learning new tasks without forgetting previously learned tasks. Other than the generally applied setup, where training data for all tasks is simultaneously available to the machine learning algorithm, in lifelong learning tasks are learned in a consecutive manner. At each moment the algorithm has only access to the data of a single task. For deep neural networks, one could consider finetuning the network on the data of the latest task. However in the absence of training data of previous tasks, the network will suffer from a phenomenon called *catastrophic forgetting* [112]. This refers to a drastic drop in performance on previous tasks. Lifelong learning studies strategies to mitigate the impact of catastrophic forgetting for continual learning [80, 97, 135].

Lifelong learning has explored a variety of strategies to prevent deep neural networks from forgetting previously learned tasks. Li et al. [97] propose a method called learning-without-forgetting (LwF). They use the same data to supervise learning of the new tasks and to provide unsupervised output guidance on the old tasks to prevent forgetting. Elastic weight consolidation (EWC) [80] estimates the Fisher matrix to weight a regularization term favouring changes to neurons which were found to be less important in previous tasks, and which prevents the relevant neurons from adapting to the new task. Further research on lifelong learning includes research on different regularization terms [2, 101], various ways of selecting sub-networks by learning masks [109, 110, 144], and the usage of exemplars [106, 135]. All these works study lifelong learning in classification networks.

In this chapter, we investigate lifelong learning for embedding networks. These networks map data to embedding spaces in which distances correspond to semantic dissimilarities between data points [24]. They are typically used for image retrieval [168], face recognition [140], feature matching [25], etc. However, they can also be used for classification when combined with, for example, a nearest class mean classifier [114]. The advantages of embedding networks, when compared to classification networks, is an ongoing discussion. Recent works have pointed out serious shortcomings of classification networks, mostly attributed to

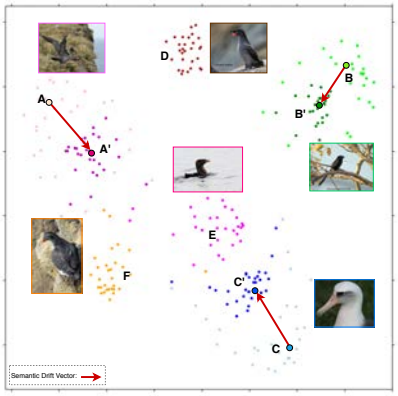


Figure 5.1 – T-SNE visualization of embedding space on CUB dataset. A, B, C indicate prototypes of task 1 after training task 1. The prototypes for six classes after training task 2 are also provided (A', B', C' for task 1 and D, E, F for task 2).

the cross-entropy loss (which is based on a softmax operation). Embedding networks were found to be more robust to the exposure of adversarial examples, and better in the detection of out-of-distribution examples [111, 136]. Furthermore, deep embeddings were reported to be superior to classification networks for transfer learning [142] and preliminary results suggest that they might be less prone to catastrophic forgetting than classification networks [188].

As discussed above, the focus of lifelong learning has been on preventing networks to forget by reducing the drift of features* which are important for previous tasks when learning new tasks. In this chapter, we propose a new approach, where, rather than preventing the drift, we aim at estimating the drift that occurs during the training of new tasks (see Fig. 5.1). Having an estimate of the drift we show that previous tasks can be compensated for this drift, thereby improving their performance. We investigate the estimation of drift for lifelong learning in embedding networks. We will evaluate the networks for classification by using the nearest class mean (NCM) classifier [114]. We will refer to the class mean with the term *prototype*. We will show how the drift of prototypes learned in previous tasks can be approximated while only having access to data of the current task. Furthermore, the proposed method can easily be combined with existing methods that prevent forgetting, such as EWC [80] and LwF [97], to further improve results. Finally, one of the important observations of this chapter is that embedding networks are significantly less prone to catastrophic forgetting than classification networks.

*We use features to refer to the activations along the feature directions.

5.2 Related Work

5.2.1 Learning Embeddings

Siamese networks [24] were first proposed to learn embeddings for the task of face verification. They use contrastive loss, which is used to ensure that pairs from the same category are close and pairs from different categories are far. One limitation of this loss is that a single collapsing point is enforced for all images of the same class, which may be fine in certain cases (e.g. if the target is classification), but less so in others, where we may require a more nuanced embedding space. Triplet networks [58, 169] were proposed to address this limitation. The inputs are an anchor image, a positive image and a negative image instead of pairs. The aim of a triplet network is to learn embeddings for which the distance between the similar pairs is smaller than the distance between the dissimilar pairs.

A primary issue when training deep metric learning models is the lack of informative pairs or triplets. This motivated the development of efficient hard negative mining strategies [140, 149, 171]. For example, Simo-Serra et al. [149] proposed to sample only the hardest positive and negative pairs in a very large mini-batch to do the backward pass. Wang et al. [171] proposed to sample the hardest negative pairs as part of triplets. To further reduce the redundancy of sampling, several works propose fast backpropagation methods by considering all possible pairs or triplets in the mini-batch [102, 152, 153]. Song et al. [153] proposed a lifted structured loss to consider all the positive and negative pairs within a batch. Finally, Wu et al. [178] proposed distance weighted sampling, which selects more informative and stable examples than traditional approaches.

5.2.2 Lifelong Learning

Lifelong learning in deep neural networks considers learning a sequence of tasks in a continual fashion, which is more similar to how biological systems learn in the real world. However, networks trained in this manner suffer from a phenomenon called catastrophic forgetting, which is described by McCloskey and Cohen [112] in the late 1980s. The continual learning literature can be roughly divided into three main categories as follows:

Rehearsal-based methods. The idea is to save a small subset of the training data from previously learned tasks in order to prevent the network from forgetting previous tasks during the training of new task. These saved exemplars are regularly combined (i.e. rehearsed) with the data of the new task, and the network parameters are jointly optimized. One way is to save real exemplars from previous tasks and to use distillation loss to prevent information forgetting [18, 106, 135]. An alternative

is to learn a generative model of previous tasks, and generate synthetic samples (i.e. pseudo-rehearsal) that are combined with samples of new task [148, 179].

Regularization-based method These works focus on optimizing network parameters on the current task while preventing the drift of already consolidated weights. Only samples for the new task are used to train, without having access to previous data. Learning without forgetting (LwF) [97] aims at adapting a learned model to new tasks while retaining the knowledge gained earlier as a regularization term on probabilities to improve performance for both tasks. EWC [80] and a variant R-EWC [101] include a regularization term on the weights that forces parameters of the current network to remain close to the parameters of the network trained for the previous tasks. Zenke et al. [199] propose to compute the consolidation strength of synapses (represented by the network weights) in an online manner, and extends them with a memory to accumulate task-relevant information. Aljundi et al. [2] compute the weight importance in a unsupervised manner.

Architecture-based method Another way to prevent catastrophic forgetting is by growing a sub-network for each task, as done in [91, 109, 110, 139, 144]. In HAT [144], the authors propose a task-based hard attention mechanism that maintains the information from previous tasks without affecting the learning of a new task. PackNet [110] learns a sparse masks per task as a result of pruning. Piggyback [109] learns task-specific masks reusing the weights from models pre-trained on ImageNet. Expert gate [3] learns a dedicated network for each task. These networks are selected based on the outcome of a set of gated auto-encoders.

All of the methods discussed above focus on preventing forgetting during the learning of new tasks. Our method does not focus on preventing the forgetting, but instead proposes to estimate the drift of features that happens due to the learning of new tasks. Having an approximation of the drift, we can then compensate the prototypes of previous tasks. In experiments we show that our method can be combined with methods which prevent forgetting. An additional difference with existing literature is that we present the first work on lifelong learning for embedding networks, whereas previous work focuses on classification networks. Only memory aware synapses [2] also provides some experimental results on a cross-modal embedding task.

5.3 Lifelong Learning for Embedding Networks

In this section, we consider a lifelong learning setup where a deep network has to learn several tasks. We study this setup for the task of learning an embedding space. This space should be semantically meaningful in that it maps data from the same

class to nearby coordinates and data of different classes to coordinates which are at least a margin away. During the training of task t we only have access to data D^t which contains pairs $(\mathbf{x}_i, y_i)$ where $\mathbf{x}_i$ is an image of class $y_i \in C^t$. For each task we consider that there is data of a limited set of classes $C^t = \{c_1^t, c_2^t ..., c_{m^t}^t\}$, where m^t is the number of classes considered in task t. We consider the generally studied case where there is no overlap between the classes of different tasks: $C^t \cap C^s = \varnothing$ for $t \neq s$.

After training all n tasks we evaluate the learned embedding on all classes $C = \bigcap_i C^i$. For evaluation we consider a *task-agnostic* setting where the algorithm has no access to the task label at test time, a setting which is also considered in [3, 18, 101]. It should be noted that several algorithms assume a *task-guided* setting where knowledge of the task label is required for the application of the method [109, 110, 144], and thus these methods cannot be evaluated in the task-agnostic setting considered here.

5.3.1 Embedding Networks

We start by explaining the training of an embedding network for a single task. Embedding networks map data into a low-dimensional output where distance represents the semantic dissimilarity between the images [15, 24]. They simultaneously perform feature extraction as well as metric learning. In the learned embedding space it is possible to apply any simple metric, such as L2-distance, to determine the similarity between the original images. Chopra et al. [24] proposed to use Siamese networks with the contrastive loss as objective function. This objective needs related and unrelated pairs of images, and ensures that the distance between related pairs will be low, and the distance between unrelated pairs larger than a threshold. The contrastive loss for a single pair is formulated as:

$$\mathscr{L}_C = \frac{1}{2}(1-\tau_{ij})d_{ij}^2 + \frac{1}{2}\tau_{ij}\left\{\max(0, m-d_{ij})\right\}^2, \tag{5.1}$$

where d is defined as the Euclidean distance between the outputs of the Siamese networks

$$d_{ij} = \text{dist}(\mathbf{z}_i, \mathbf{z}_j) = \left\|\mathbf{z}_i - \mathbf{z}_j\right\|, \tag{5.2}$$

where $\mathbf{z}_i = F(\mathbf{x}_i)$ is the output embedding for image $\mathbf{x}_i$, τ_{ij} is 0 if the inputs are from the same class ($y_i = y_j$) and 1 otherwise ($y_i \neq y_j$), and $m > 0$ is a margin value above which negative pairs are not pushed further apart.

For some embeddings it is found that the contrastive loss is hard to train, and other losses have been proposed. The triplet loss is proposed by Hoffer et al. [58]

based on the work of Wang et al. [169].

The objective function forces the negative instance to be further away from the anchor than the positive ones (plus a margin m). The Triplet loss is given by:

$$\mathscr{L}_T = \max(0, d_+ - d_- + m), \tag{5.3}$$

where d_+ and d_- are the Euclidean distances between the embeddings of the anchor $\mathbf{z}_a$ and the positive instance $\mathbf{z}_p$ and the negative instance $\mathbf{z}_n$ respectively.

Having trained an embedding network we can use the embedding space for classification. Here we will use nearest class mean (NCM) classifier which is defined as:

$$c_j^* = \underset{c \in C}{\operatorname{argmin}} \operatorname{dist}\big(\mathbf{z}_j, \mu_c\big), \tag{5.4}$$

$$\boldsymbol{\mu}_c = \frac{1}{n_c} \sum_i \big[y_i = c\big] \mathbf{z}_i, \tag{5.5}$$

where n_c is the number of training images for class c and $[P] = 1$ if P is true, and 0 otherwise. We will refer to $\boldsymbol{\mu}_c$ as the *prototype* of class c. The termonology of prototypes was also used in several works [151, 188] to refer to class representative points in an embedding space.

5.3.2 Preventing Forgetting

Deep neural networks are known to suffer from the catastrophic forgetting problem, tending to forget the knowledge from the previous tasks when sequentially learning new tasks. This problem has been extensively studied for classification networks [18, 80, 97, 101, 135, 148, 179]. To our knowledge, there is no prior work to prevent forgetting the knowledge from the previous tasks on embedding networks. In the following, we adapt several existing techniques to embeddings. We will indicate the variant for embeddings the following notation convention: we use LwF to indicate the original method on a classification method, and E-LwF to indicate the adapted method to an embedding network.

Finetuning (E-FT) After convergence on a task is reached, we continue training on a new task with standard stochastic gradient descent, and repeat this until all tasks are learned. We will report this as a baseline. It is reported to lead to catastrophic levels of forgetting for classification networks, where performance on previous tasks is often close to random [80, 97, 135].

Alignment Loss (E-LwF) [97] This method was proposed on classification net-

works. It aims to match the softmax output of the network of previous models on current data.

Instead, on embedding networks, we constrain the parameters drift by minimizing the distance between the output embeddings of image $\mathbf{x}_i$ during training the current task ($\mathbf{z}_i^t$) with respect to its embedding in the previous task ($\mathbf{z}_i^{t-1}$). This leads to the following loss:

$$\mathscr{L}_{LwF} = \left\| \mathbf{z}_i^t - \mathbf{z}_i^{t-1} \right\|, \tag{5.6}$$

where $\|.\|$ refers to the Frobenius norm.

E-EWC [80] This method was proposed on classification networks to keep the network parameters close to the optimal parameters for the previous task while training the current task. This can also be leveraged on embedding networks. The function that we minimize in EWC is:

$$\mathscr{L}_{EWC} = \sum_p \frac{1}{2} \mathscr{F}_p^{t-1} (\theta_p^t - \theta_p^{t-1})^2, \tag{5.7}$$

where $\mathscr{F}^{t-1}$ is the Fisher information matrix computed after the previous task $t-1$ was learned, and the summation goes over all parameters θ_p of the network.

E-MAS [2] This method was proposed to accumulate an importance measure for each parameter of the network based on how sensitive the predicted output function is to a change in this parameter, which can be directly applied to embeddings. The function that we minimize in MAS is:

$$\mathscr{L}_{MAS} = \sum_p \frac{1}{2} \Omega_p (\theta_p^t - \theta_p^{t-1})^2, \tag{5.8}$$

where Ω_p is estimated by the sensitivity of the squared l_2 norm of the function output to their changes.

These losses can be added to the metric learning loss to prevent forgetting while training embeddings in a continual manner according to:

$$\mathscr{L} = \mathscr{L}_{ML} + \gamma \mathscr{L}_C, \tag{5.9}$$

where $C \in \{LwF, EWC, MAS\}$, γ is trade-off between the metric learning loss and the other losses.

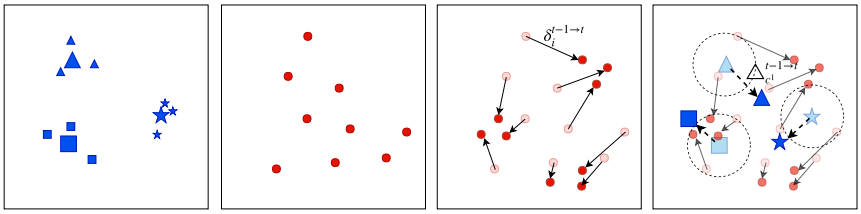


Figure 5.2 – Illustration of semantic drift compensation. (a) Data and prototypes of three classes of task 1 after training task 1. (b) Data of task 2 after training task 1. (c) Drift of data of task 2 while training task 2. (d) This vector field is used to approximate the drift of the prototypes of task 1.

5.4 Semantic Drift Compensation

Embeddings suffer from drift when learned in a sequential manner. When the data or exemplars from the previous tasks are not available, using the original prototype in the nearest mean classifier usually results in a performance drop. We aim at reducing the error that drift causes and propose a drift compensation method to update previously computed prototypes. The main idea is to estimate the unknown drift according to the known drift of the current date during the training of the current task.

5.4.1 Computation of the Semantic Drift

In Section 5.3.1, we discussed how prototypes of the classes can be computed for a single task. Here we extend this theory to the lifelong learning setting. We refer to the prototype mean as $\boldsymbol{\mu}^t_{c^s}$ which is the mean for class c^s after learning task t computed with Eq. 5.5. Recall that class c^s is learned during task s (we removed the sub-index i from c^s_i for conciseness). When $t > s$ we have no access anymore to data of task s and we cannot compute the true prototype mean (by applying Eq. 5.5 again). We call the difference between the true class mean and the estimate of the class mean the *semantic drift*:

$$\boldsymbol{\Delta}^{s\rightarrow t}_{c^s} = \boldsymbol{\mu}^t_{c^s} - \boldsymbol{\mu}^s_{c^s}. \tag{5.10}$$

Since we cannot compute $\boldsymbol{\mu}^t_{c^s}$ directly we have to find alternative ways to approximate the semantic drift $\triangle^{s\rightarrow t}_{c^s}$. We start by proposing a method to compute $\boldsymbol{\Delta}^{t-1\rightarrow t}_{c^s}$ from which we can then derive $\boldsymbol{\Delta}^{s\rightarrow t}_{c^s}$.

When training task t we do not have access to the data of task s and therefore we cannot observe how the embeddings $\mathbf{z}_i$, for which $y_i \in C^s$, drift during training

of task t. However, we can measure the drift of the current data during the training of task t.

$$\boldsymbol{\delta}_i^{t-1\rightarrow t} = \mathbf{z}_i^t - \mathbf{z}_i^{t-1}\ ,\ y_i \in C^t, \tag{5.11}$$

here we use the notation $\mathbf{z}_i^t$ to refer to the embedding of point i after training task t. At the start of training task t we have access to $\mathbf{z}_i^{t-1}$ which is the embedding of data point i after training task $t-1$.

We propose to approximate the semantic drift $\boldsymbol{\Delta}_{c^s}^{t-1\rightarrow t}$ from the sparse vector field $\boldsymbol{\delta}_i^{t-1\rightarrow t}$. We do this by interpolating this vector field at the prototype location $\boldsymbol{\mu}_{c^s}^{t-1}$ using:

$$\hat{\boldsymbol{\Delta}}_{c^s}^{t-1\rightarrow t} = \frac{\sum_i \left[y_i \in C^t\right] w_i \boldsymbol{\delta}_i^{t-1\rightarrow t}}{\sum_i \left[y_i \in C^t\right] w_i} \tag{5.12}$$

with

$$w_i = e^{-\frac{\left\|\mathbf{z}_i^{t-1} - \boldsymbol{\mu}_{c^s}^{t-1}\right\|^2}{2\sigma^2}}, \tag{5.13}$$

where σ is the standard deviation of the Gaussian kernel.

In summary, for all the data points in task t we can monitor the semantic drift during the training of task t. Finally, this results in a set of drift vectors $\boldsymbol{\delta}_i^{t-1\rightarrow t}$. These vectors are used to compute the semantic drift of all the previously learned prototypes $\hat{\boldsymbol{\mu}}_{c^s}^{t-1}$. This is done by assigning a weight to the drift vectors according to their distance to the prototypes, and compute the prototype drift as weighted mean of the nearby drift vectors (with Eq. 5.12). The process is illustrated in Fig. 5.2.

We can now apply the semantic drift compensation (SDC) according to

$$\hat{\boldsymbol{\mu}}_{c^s}^t = \boldsymbol{\mu}_{c^s}^s + \hat{\boldsymbol{\Delta}}_{c^s}^{s\rightarrow s+1} + ... + \hat{\boldsymbol{\Delta}}_{c^s}^{t-1\rightarrow t}, \tag{5.14}$$

where total compensation is the sum of the compensations which where measured during all the previous steps. Normally a recursive scheme would be applied where you update all previously learned prototypes at each new task:

$$\hat{\boldsymbol{\mu}}_{c^s}^t = \hat{\boldsymbol{\mu}}_{c^s}^{t-1} + \hat{\boldsymbol{\Delta}}_{c^s}^{t-1\rightarrow t}. \tag{5.15}$$

5.4.2 Combining preventing forgetting and drift compensation

Many approaches to lifelong learning have focused on preventing the network from using parameters which were found to be relevant for previous tasks [2, 80, 97].

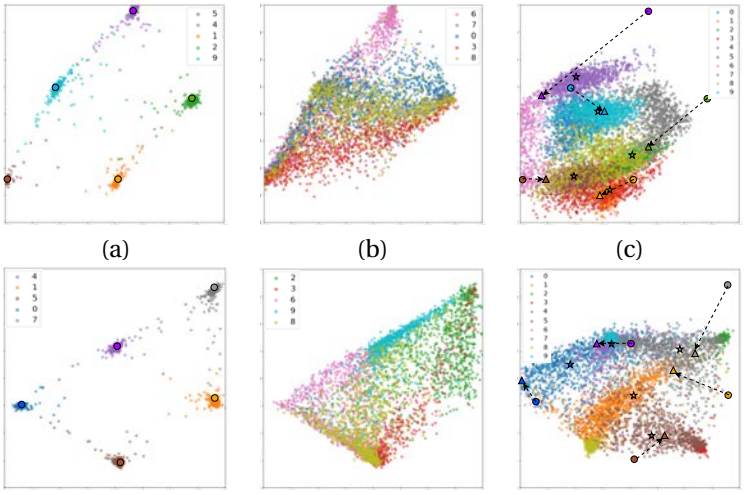


Figure 5.3 – Examples of the drift vectors in the cases of E-FT (top) and E-EWC (bottom). (a) and (d) represent the embedding of 5 classes of task 1 after training task 1; (b) and (e) represent the embedding of another 5 classes of task 2 after training task 1; (c) and (f) show the embeddings of 10 classes of two tasks together. The saved prototypes of the previous task(indicated by round) are estimated to new positions (indicated by triangle) by our proposed SDC in the new model which is observed to be closer to the real mean (indicated by star). The dotted arrows are the SDC vectors.

Our method is based on an entirely different approach where we accept the fact that if we share parameters between the tasks, and we want all tasks to be able to improve (i.e. backpropagate) to all these parameters, this will result in a drift for the previously learned tasks. Approximating this drift allows us then to compensate for it. Since our approach applies a different methodology to prevent forgetting, it is interesting to see if it is complementary to these other methods. We therefore propose to combine existing methods (E-LwF, E-EWC and E-MAS) with semantic drift compensation and will evaluate this in the experimental results.

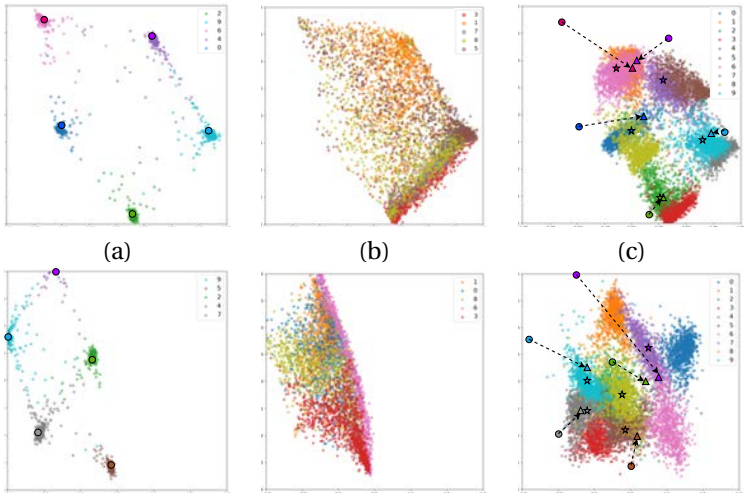


Figure 5.4 – Examples of the drift vectors in the cases of E-LwF (top) and E-MAS (bottom).

5.5 Experiments

In this section we follow the protocol for evaluating incremental learning methods [2, 101, 135]. For the multi-class datasets, the classes are arranged in a fixed random order. Each method is trained in a class-incremental way on the available data and evaluated on the test part of the dataset. For all the results in the chapter we report the *average incremental accuracy* (evaluated on the test data of the dataset, considering only those classes that have already been trained) repeating the experiment three times.

Datasets. We have used the following datasets:

MNIST: It is a dataset of handwritten digits (from 0 to 9) composed of $28 * 28$ pixel greyscale images, with 60K training images and 10K test images. We divide the ten classes into two disjoint tasks randomly.

Fine-grained datasets: We run experiments in three additional datasets: CUB-200-2011 [176], Stanford 40 Action [192] and Flowers [121]. All of them are divided into four tasks randomly. CUB-200-2011 has 200 classes of birds with 11,788 images in total. Stanford 40 Action contains 40 types of actions with 9,532 images in total. Flowers consists of 102 flower categories of which we randomly choose 100 with

Table 5.1 – Results on CUB-200-2011, Stanford 40 Actions and Flowers in a four task scenario. The results show the average incremental accuracy.

	CUB-200-2011				Stanford 40 Actions				Flowers			
	T1	T2	T3	T4	T1	T2	T3	T4	T1	T2	T3	T4
FT	84.3	37.2	25.3	18.4	84.6	34.6	23.5	15.4	96.7	48.8	32.0	24.5
E-FT	82.8	63.8	48.8	38.7	84.4	78.7	75.7	66.3	94.5	90.1	82.5	78.2
E-FT+SDC	**82.8**	**74.2**	**63.9**	**47.3**	**84.4**	**81.0**	**79.9**	**73.3**	**94.5**	**91.9**	**88.7**	**86.5**
LwF	84.4	61.5	44.4	31.2	83.6	68.8	62.1	50.3	96.9	87.4	74.9	64.6
E-LwF	82.3	73.1	67.5	61.2	84.1	80.3	78.7	72.2	94.6	90.0	85.9	81.9
E-LwF+SDC	**82.3**	**73.5**	**68.7**	**64.0**	**84.1**	**80.7**	**79.0**	**72.4**	**94.6**	**90.2**	**86.8**	**83.4**
EWC	85.1	43.3	36.3	28.5	85.2	45.2	33.3	25.1	97.0	60.3	41.3	29.5
E-EWC	82.5	71.1	66.2	59.7	84.8	79.4	76.8	69.1	94.5	90.9	85.9	84.6
E-EWC+SDC	**82.5**	**74.3**	**69.4**	**65.4**	**84.8**	**81.4**	**79.4**	**74.3**	**94.5**	**91.1**	**88.8**	**87.4**
MAS	83.3	62.5	55.4	49.6	85.2	55.4	47.5	38.2	96.5	78.6	68.7	65.4
E-MAS	82.8	70.4	64.7	59.6	84.4	78.1	74.9	68.4	94.9	88.7	83.7	82.6
E-MAS+SDC	**82.8**	**72.2**	**67.1**	**61.6**	**84.4**	**79.8**	**77.5**	**71.1**	**94.9**	**89.5**	**85.8**	**83.8**

8189 images in total.

Implementation Details. All models are implemented with Pytorch and trained on Titan-X GPUs. Adam [78] is used for the optimization. When training an embedding network, all layers in the network are shared across all tasks, which allows us to perform inference without explicitly knowing the task in the task-agnostic setting.

MNIST: LeNet-5 network [89] with contrastive loss [15, 24] is because of its efficient deployment for character recognition. There is no pre-processing for the inputs. The final embeddings are represented in 64 dimensions without normalization. Each training step consists of 30 epochs with mini-batch of size 256. The learning rate starts from $1e-4$ and is divided by 20 epochs. We set the trade-off between the E-LwF loss and contrastive loss $2e-3$, $1e4$ for both E-EWC and E-MAS. We choose $\sigma = 0.01$ to compute the weights of the SDC vectors empirically for MNIST.

Fine-grained datasets: ResNet-101 [52] with Triplet loss [58] is adopted as the backbone network pretrained from ImageNet. The training images (resized to 256×256) are randomly cropped and flipped, and no other data augmentation is used. The input images are directly fed into the network and all possible pairs are considered in the mini-batch. We train all the models for 50 epochs with mini-batch size of 32, and learning rate $1e-5$. The final embeddings are normalized in 512 dimensions. The trade-off between the E-LwF, E-EWC, E-MAS and Triplet loss is 1, $1e7$ and $1e6$ respectively. We choose $\sigma = 0.15$ to compute the weights of the SDC vectors empirically for all of these three fine-grained datasets.

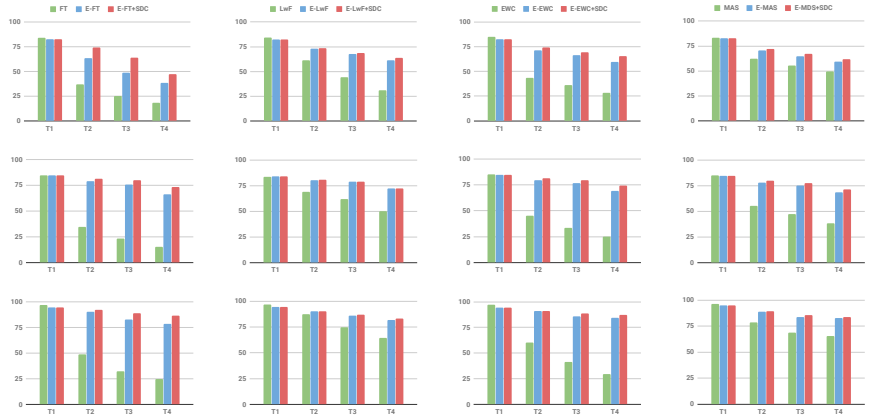


Figure 5.5 – Results on CUB-200-2011 (top), Stanford 40 Actions (middle) and Flowers (bottom) in a four task scenario. The results show the average incremental accuracy. A table version is available in Table 5.1.

Table 5.2 – Classification accuracy (%) for the two-tasks scenario on MNIST. Results are presented as accuracy on T1/T2 (avg).

Methods	Original Mean	Compensated Mean (+SDC)
E-FT	13.4/99.3 (56.4)	26.2/98.4 (62.3)
E-LwF	60.5/99.3 (79.9)	80.6/87.2 (83.9)
E-EWC	74.4/51.8 (63.1)	92.8/42.2 (67.5)
E-MAS	36.2/95.3 (65.8)	84.7/83.6 (84.2)

5.5.1 MNIST

Experiments and Analysis Table 5.2 shows the average accuracy for two tasks after training on the second task with E-FT, E-LwF, E-EWC and E-MAS on MNIST (we use the notation *E-name method* to stress that the technique is applied on an embedding network), and the accuracy after applying our semantic drift compensation to update the original prototype of the first task. We can see that all methods outperform finetuning in terms of average accuracy. E-LwF performs best among all the methods without SDC and is a 23.5% superior to E-FT. Meanwhile, SDC improves all of the methods significantly, expecially for E-MAS. It obtains the best overall average accuracy for MNIST with a gain of 18.4% over SDC.

Visualization To better understand the effectiveness of SDC, we conduct experiments on MNIST with a 2-dimensional embedding. The other settings are the

Table 5.3 – Classification accuracy (%) for sequences of 2 tasks on cross-domain setting.

Method	Birds → Flower		Flower → Birds		Flower → Scene		Scene → Flower		Scene → Birds		Birds → Scene		Average
E-FT	8.6	88.1	26.4	64.9	27.7	63.5	14.2	84.6	39.6	64.6	25.3	63.5	47.6
E-FT+SDC	9.7	88.1	31.6	65.7	44.7	63.7	15.1	85.4	49.6	65.4	37.3	62.7	**51.6**
E-LwF	10.6	88.6	61.1	61.1	61.4	65.7	10.6	86.6	22.0	61.0	30.0	68.4	52.3
E-LwF+SDC	13.7	88.6	68.6	61.1	69.9	66.2	18.4	86.5	28.0	61.1	32.2	68.7	**55.3**
E-EWC	40.1	83.0	59.5	60.1	40.3	61.3	52.8	81.4	56.4	56.8	41.7	60.5	57.8
E-EWC+SDC	45.2	83.9	74.0	59.7	63.6	61.3	48.8	82.7	57.8	56.5	46.9	60.9	**61.8**
E-MAS	41.5	76.9	61.1	53.5	50.9	57.9	51.9	76.7	57.2	48.6	43.0	56.0	56.3
E-MAS+SDC	49.9	78.8	76.4	53.3	75.1	58.4	52.4	79.1	59.1	49.2	50.6	55.9	**61.5**

same as we described in the implementation details of MNIST. In Fig.5.3 we show examples of the drift vectors which are estimated by SDC in the case of E-FT and E-EWC. We can see that the approximated drift vectors improve the locations of the prototypes to be closer to the correct positions. As a result, the accuracy of the overall method remains higher while training new tasks.

In Fig. 5.4 we show examples of the drift vectors which are estimated by SDC in the case of E-LwF and E-MAS.

5.5.2 Within-domain continual learning

Classification Accuracy with Embedding Networks To further evaluate the effectiveness of our methods, we conduct experiments on three fine-grained datasets: CUB-200-2001, Stanford 40 Action, and Flowers on the four tasks scenario. Results are shown in Fig.5.5 (we focus on the blue and red bars here). All the results are reported by the average incremental accuracy. Here we analyze the average results after training the last task (T4) in more detail.

It can be seen that E-LwF, E-EWC and E-MAS outperform E-FT on all of three datasets. An absolute gain is obtained of 5.9% on actions and of 22.5% on birds. The performance of these three methods to prevent forgetting is comparable. Next, we can see that SDC improves the results of all methods especially for E-FT with 7.0% on actions and 8.6% on birds. Meanwhile, the gain for E-EWC and E-MAS are larger than for the E-LwF methods on birds and actions. Essentially, E-EWC and E-MAS indirectly limit the drift of the embedding by constraining the important weights, whereas E-LwF is directly constraining the embedding, which in the end results in less drift. On the Flowers dataset, SDC slightly improves the result for all of the methods except for FT where a larger improvement is observed. We attribute this to the relative simplicity of the Flowers dataset, the average accuracy of four tasks without SDC already is beyond 80% with embedding networks, and there is less room to compensate the drift to further improve the performance. It is interesting to notice that applying E-FT together with SDC can achieve as good result as the

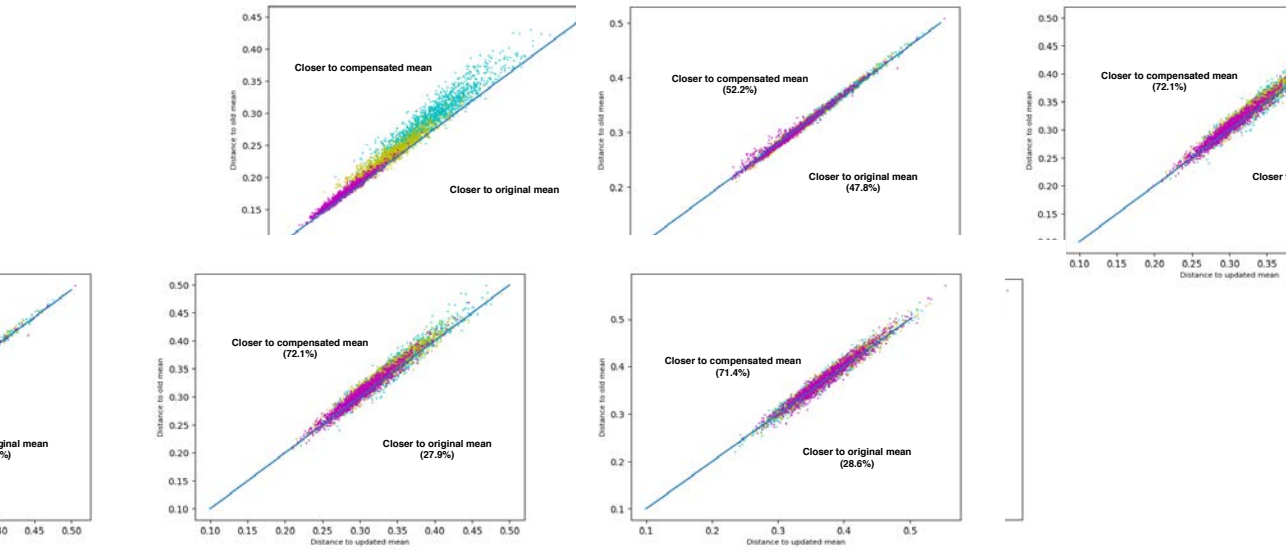


Figure 5.6 – Comparison of distances to the original mean and the drift-compensated mean with E-FT, E-LwF, E-EWC and E-MAS on CUB-200-2011 dataset after learning four tasks. The x-axis is the distance between the embedding to the drift-compensated mean and the y-axis is the distance to the original mean. Points are colored for different tasks (blue, yellow, pink for respectively task 1,2,3). If the data point appears in the top left area it means that the drift-compensation has improved the location of the prototype.

best performance among the three methods to prevent forgetting on Actions and Flowers dataset. This suggests that good performance can be obtained without saving the previous models as long as SDC is applied.

In Table 5.1, we show the results on these three fine-grained datasets: CUB-200-2001, Stanford 40 Action, and Flowers on the four tasks scenario. All the results are reported by the average incremental accuracy. Here we show the average results after training the last task (T4) in more detail. The results are exactly the same as the ones used for Fig 5.5.

Fig.5.6 shows the comparison of distances between the data to the original prototype and the drift compensated prototype (for E-FT, E-LwF, E-EWC and E-MAS respectively)). Points above the line indicate points for which the drift compensation has lowered the distance to their prototype. We can see that data from earlier tasks tend to have a larger distance to their prototype, and that for the vast majority of point SDC improves the distance to the prototype.

Table 5.4 – Classification accuracy (%) for sequences of 3 tasks.

	Birds	Flower	Scene	Average
E-FT	28.9	35.6	35.1	33.2
E-FT+SDC	31.7	34.9	44.9	**37.2**
E-LwF	39.8	78.6	44.2	54.2
E-LwF+SDC	41.1	80.5	44.9	**55.5**
E-EWC	42.5	56.5	51.1	50.0
E-EWC+SDC	45.5	64.4	52.3	**54.1**
E-MAS	42.6	56.4	50.2	49.7
E-MAS+SDC	46.5	64.8	50.1	**53.8**

Embedding Learning versus Softmax Classifier Since we evaluate our embedding networks on the task of classification, we can also compare them to results of classification networks. To fairly compare with embedding network, we only change the last layer into a classifier, keeping all other layers the same. In order to test in the task-agnostic setting with soft-max classifier, we first train a single-head classifier for each task, during test, we compute the probability of each head and take the maximum as the true prediction.

The results of softmax classifier with ResNet-101 are presented on Fig.5.5 (indicated by the green bar). We observe that the average results drop dramatically as the number of tasks increases, especially in the setting Finetuning. After four tasks, embedding networks significantly outperform classification networks, with several performance gains being more than double. Furthermore, applying simple finetuning on embedding networks outperforms the lifelong learning methods for Actions and Flowers. This confirms that embedding networks are much more effective for lifelong learning than classification networks.

5.5.3 Cross-domain continual learning

Two-task experiments We consider sequences of two tasks on three datasets for cross datasets evaluation as used previously in [2, 97]: MIT Scenes [133] for indoor scene classification (5,360 samples), Birds [176] and Flowers [121]. All the possible combinations are included in Table 5.3. Again, combining SDC boosts the performance for all four methods E-FT, E-LwF, E-EWC and E-MAS, respectively. In average, the gain varies from 3% on E-LwF to 5.2% on E-MAS. Surprisingly, E-FT+SDC is only slightly worse than E-LwF without SDC.

Longer sequences We consider the more challenging setting with all the possible combinations of the three datasets mentioned before. In total, we have six different

combinations and we show the average of all results for each dataset after training the last task. As shown in 5.4, the conclusion is consistent with the two-tasks scenario on cross-domain, while the gap between E-FT and other methods gets larger, which is possibly due to the complexity of this setting.

5.6 Conclusions

We have studied lifelong learning in embedding networks. A new method is proposed to approximate the semantic drift of prototypes during the training of new tasks. The method is complementary to several existing methods for lifelong learning originally designed for classification networks. Experiments on both within and cross-domain settings show that our method obtains large gains when combined with finetuning. It also consistently improves the performance when combined with existing approaches.

One observation is that embedding networks are more robust to catastrophic forgetting than classification networks. The dramatic effect of catastrophic forgetting when applying finetuning, as observed on classification networks, is much less pronounced for embedding networks. On Action and Flower , in the within-domain experiment, finetuning obtains better results on embedding network than several popular methods (EWC, LwF, MAS) applied to classification network.

6 Conclusions and Future Work

In this thesis, we aim at improving the color embeddings and deep embeddings from different aspects. In this chapter we summarize the approaches proposed in this thesis for these two image embeddings. The chapter ends with future research directions.

6.1 Conclusions

In this thesis, we have investigated the image embeddings from two aspects: color embeddings and deep embeddings. In the first part, we studied color embeddings based on color naming method in Chapter 2 and 3. To learn more discriminative color features, a method was proposed to extending the basic eleven color name set in Chapter 2. The results on image classification, person re-identification and object tracking have shown the benefit of extending color name set beyond the basic color name set. In Chapter 3, we performed an attention-based method for domain-specific color naming in a weakly supervised manner. The corresponding part of the given label is learned end-to-end for more accurate description.

In the second part of the thesis, we focused on the problems of deep metric learning in Chapter 4 and 5. In Chapter 4, two new losses were proposed to guide the student network to mimic the information transferred from the teacher network. In Chapter 5, lifelong learning for embedding networks is investigated.

The methods proposed and the results obtained in this thesis are summarized in the paragraphs below:

Chapter 2: Beyond Eleven Color Names for Image Understanding. We collect a dataset of 28 additional color names. To ensure that the resulting color representation has high discriminative power we propose a method to order the additional color names according to their complementary nature with the basic color names. This allows us to compute color name representations with high discriminative power of arbitrary length. In the experiments we show that these new color name descriptors outperform the existing color name descriptor on the task of visual tracking, person re-identification and image classification.

Chapter 3: Weakly Supervised Domain-Specific Color Naming Based on Attention. We aim to learn color names from weakly labeled data. For this purpose,

we add an attention branch to the color naming network. The attention branch is used to modulate the pixel-wise color naming predictions of the network. In experiments, we illustrate that the attention branch correctly identifies the relevant regions. Furthermore, we show that our method obtains state-of-the-art results for pixel-wise and image-wise classification on the EBAY dataset and is able to learn color names for various domains.

Chapter 4: Learning Metrics from Teachers: Compact Networks for Image Embedding. We propose two new loss functions that model the communication of a deep teacher network to a small student network. We evaluate our system in several datasets, including CUB-200-2011, Cars-196, Stanford Online Products and show that embeddings computed using small student networks perform significantly better than those computed using standard networks of similar size. Results on a very compact network (MobileNet-0.25), which can be used on mobile devices, show that the proposed method can greatly improve Recall@1 results from 27.5% to 44.6%. Furthermore, we investigate various aspects of distillation for embeddings, including hint and attention layers, semi-supervised learning and cross quality distillation.

Chapter 5: Semantic Drift Compensation for Lifelong Learning of Embeddings. In lifelong learning, deep neural networks are trained on a sequence of tasks. At each moment they only have access to data of the current task. In this setting, networks suffer from catastrophic forgetting which refers to the drastic drop in performance on previous tasks. This is caused because network weights adapt to be optimal for the current task, resulting in a shift of the features, and thus predictions for previous tasks become less accurate. The vast majority of methods in lifelong learning have focused on preventing this drift from happening, often by imposing regularization on neurons which are important for previous tasks.

In contrast to previous work, we study lifelong learning in embedding networks and not on classification networks. In addition, instead of preventing the drift of features, we aim to estimate the drift and compensate for it. We call the drift in the embedding space the semantic drift and propose an approximation of it based on the drift which is experienced by data of the current task during its training. We show that semantic drift compensation (SDC) can compensate partially for the semantic drift which occurs during the training of new tasks. On several experiments, considering both within-domain and cross-domain task learning, we show that SDC consistently improves results. In addition, we show that embedding networks suffer significantly less from catastrophic forgetting even when applying simple finetuning to learn new tasks. When combined with lifelong learning techniques, and/or the method proposed in this thesis, they consistently outperform methods trained on classification networks.

6.2 Future work

For the future work we are especially interested in pursuing the research line on deep embedding networks of Part II of the thesis. Network distillation and lifelong learning for metric learning have been studied in Chapter 4 and 5. They had been hardly investigated in prior works. We show that embedding networks suffer significantly less from catastrophic forgetting than classification networks to learn new tasks. It would be interesting to dive deeper in the advantage of metric learning compared to softmax-based classification networks. Given their advantages, it would be interesting to see if embedding networks could replace the dominance which classification networks currently have.

In addition, in Chapter 5, we proposed a method to compensate the drift occurs for metric learning networks when learning of new tasks. The drift is compensated session-based. Applying our drift compensation algorithm for each mini-batch would be the next step. Meanwhile, the task has the boundary to fed into the system, which is not realistic in the open world. In the future, we would investigate the online drift compensation without the limitation of the task boundaries.

Embeddings were found efficient on the tasks of out-of-distribution detection and transfer learning. Furthermore, embedding networks are essential for computer vision, as evidenced by the large variety of tasks in which they are used, including zero/few shot learning. Future research will also focus on investigating image embeddings on some challenging zero/few shot tasks.

Summary of published works

1. **Lu Yu**, Lichao Zhang, Joost van de Weijer, Fahad Shahbaz Khan, Yongmei Cheng, and C Alejandro Parraga. Beyond eleven color names for image understanding. JOunal of Machine Vision and Applications, 29(2):361–373, 2018.

2. **Lu Yu**, Yongmei Cheng, and Joost van de Weijer. Weakly supervised domain-specific color naming based on attention. In 2018 24th International Conference on Pattern Recognition (ICPR), pages 3019–3024. IEEE, 2018.

3. **Lu Yu**, Vacit Oguz Yazici, Xialei Liu, Joost van de Weijer, Yongmei Cheng, and Arnau Ramisa. Learning metrics from teachers: Compact networks for image embedding. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 2907–2916, 2019.

Bibliography

[1] Zeynep Akata, Scott Reed, Daniel Walter, Honglak Lee, and Bernt Schiele. Evaluation of output embeddings for fine-grained image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2927–2936, 2015.

[2] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *European Conference on Computer Vision (ECCV)*, pages 139–154, 2018.

[3] Rahaf Aljundi, Punarjay Chakravarty, and Tinne Tuytelaars. Expert gate: Lifelong learning with a network of experts. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3366–3375, 2017.

[4] Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? In *Advances in neural information processing systems*, pages 2654–2662, 2014.

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

[6] R. Benavente, M. Vanrell, and R. Baldrich. Parametric fuzzy sets for automatic color naming. *Journal of the Optical Society of America A*, 25(10):2582–2593, 2008.

[7] R. Benavente, M. Vanrell, and R. Bladrich. A data set for fuzzy colour naming. *COLOR research and application*, 31(1):48–56, 2006.

[8] Robert Benavente, Joost Van de Weijer, Maria Vanrell, Cordelia Schmid, Ramon Baldrich, Jakob Verbeek, and Diane Larlus. Color names. *Color in Computer Vision*, pages 11–43, 2012.

[9] Robert Benavente, Joost Van de Weijer, Maria Vanrell, Cordelia Schmid, Ramon Baldrich, Jakob Verbeek, and Diane Larlus. Color names. In *Color in Computer Vision*. Wiley, 2012.

[10] B. Berlin and P. Kay. *Basic color terms: their universality and evolution*. Berkeley: University of California, 1969.

[11] Brent Berlin and Paul Kay. *Basic color terms: Their universality and evolution*. Univ of California Press, 1991.

[12] Jiang Bian, Bin Gao, and Tie-Yan Liu. Knowledge-powered deep learning for word embedding. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 132–148. Springer, 2014.

[13] Hakan Bilen, Marco Pedersoli, and Tinne Tuytelaars. Weakly supervised object detection with convex clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1081–1089, 2015.

[14] Robert M Boynton and Conrad X Olson. Locating basic colors in the osa space. *Color Research & Application*, 12(2):94–105, 1987.

[15] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a "siamese" time delay neural network. In *Annual Conference on Neural Information Processing Systems (NIPS)*, pages 737–744, 1994.

[16] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th international conference on Knowledge discovery and data mining*, pages 535–541. ACM, 2006.

[17] Qiong Cao, Yiming Ying, and Peng Li. Similarity metric learning for face recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2408–2415, 2013.

[18] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *European Conference on Computer Vision (ECCV)*, pages 532–547, 2018.

[19] Guobin Chen, Wongun Choi, Xiang Yu, Tony Han, and Manmohan Chandraker. Learning efficient object detection models with knowledge distillation. In *Annual Conference on Neural Information Processing Systems (NIPS)*, pages 742–751, 2017.

[20] Xinlei Chen and Abhinav Gupta. Webly supervised learning of convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1431–1439, 2015.

[21] Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. Darkrank: Accelerating deep metric learning via cross sample similarities transfer. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[22] Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. Darkrank: Accelerating deep metric learning via cross sample similarities transfer. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2018.

[23] Zhiyi Cheng, Xiaoxiao Li, and Chen Change Loy. Pedestrian color naming via convolutional neural network. In *Asian Conference on Computer Vision*, pages 35–51. Springer, 2016.

[24] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 539–546. IEEE, 2005.

[25] Christopher B Choy, JunYoung Gwak, Silvio Savarese, and Manmohan Chandraker. Universal correspondence network. In *Annual Conference on Neural Information Processing Systems (NIPS)*, pages 2414–2422, 2016.

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

[27] Trevor F Cox and Michael AA Cox. *Multidimensional scaling*. Chapman and hall/CRC, 2000.

[28] Nello Cristianini, John Shawe-Taylor, et al. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[29] Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, volume 1, pages 1–2. Prague, 2004.

[30] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, (CVPR)*, volume 1, pages 886–893, 2005.

[31] Martin Danelljan, Gustav Häger, Fahad Shahbaz Khan, and Michael Felsberg. Learning spatially regularized correlation filters for visual tracking. In *ICCV*, 2015.

[32] Martin Danelljan, Fahad Shahbaz Khan, Michael Felsberg, and Joost van de Weijer. Adaptive color attributes for real-time visual tracking. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 1090–1097. IEEE, 2014.

[33] Prithviraj Dhar, Rajat Vikram Singh, Kuan-Chuan Peng, Ziyan Wu, and Rama Chellappa. Learning without memorizing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5138–5146, 2019.

[34] John P Eakins and Margaret E Graham. Content based image retrieval: A report to the jisc technology applications programme, 1999.

[35] Rob Fergus, Li Fei-Fei, Pietro Perona, and Andrew Zisserman. Learning object categories from internet image searches. *Proceedings of the IEEE*, 98(8):1453–1466, 2010.

[36] Graham D Finlayson, Bernt Schiele, and James L Crowley. Comprehensive colour image normalization. In *European conference on computer vision*, pages 475–490. Springer, 1998.

[37] Graham D. Finlayson, Bernt Schiele, and James L. Crowley. Comprehensive colour image normalization. In *ECCV '98: ProcP of the 5th European Conference on Computer Vision-Volume I*, pages 475–490. Springer-Verlag, 1998.

[38] Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Tomas Mikolov, et al. Devise: A deep visual-semantic embedding model. In *Advances in neural information processing systems*, pages 2121–2129, 2013.

[39] Jiyang Gao, Zhen Li, Ram Nevatia, et al. Knowledge concentration: Learning 100k object classifiers in a single cnn. In *arXiv preprint arXiv:1711.07607*, 2017.

[40] J-M Geusebroek, Rein Van den Boomgaard, Arnold W. M. Smeulders, and Hugo Geerts. Color invariance. *IEEE Transactions on Pattern analysis and machine intelligence*, 23(12):1338–1350, 2001.

[41] J.M. Geusebroek, R. van den Boomgaard, A.W.M. Smeulders, and H. Geerts. Color invariance. *PAMI*, 23(12):1338–1350, 2001.

[42] Th. Gevers and A. Smeulders. Color based object recognition. *Pattern Recognition*, 32:453–464, 1999.

[43] Theo Gevers and Arnold WM Smeulders. Color-based object recognition. *Pattern recognition*, 32(3):453–464, 1999.

[44] Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *2011 international conference on computer vision*, pages 999–1006. IEEE, 2011.

[45] Albert Gordo, Jon Almazán, Jerome Revaud, and Diane Larlus. Deep image retrieval: Learning global representations for image search. In *European Conference on Computer Vision (ECCV)*, pages 241–257. Springer, 2016.

[46] Isabelle Guyon, Steve Gunn, Masoud Nikravesh, and Lofti A Zadeh. *Feature extraction: foundations and applications*, volume 207. Springer, 2008.

[47] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *International Conference on Learning representations (ICLR)*, 2016.

[48] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*, pages 1135–1143, 2015.

[49] C.L. Hardin and L. Maffi, editors. *Color Categories in Thought and Language*. Cambridge University Press, 1997.

[50] Clyde L Hardin and Luisa Maffi. *Color categories in thought and language*. Cambridge University Press, 1997.

[51] Jonathan Harel, Christof Koch, and Pietro Perona. Graph-based visual saliency. In *Advances in neural information processing systems*, pages 545–552, 2007.

[52] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[53] G. Healey. Segmenting images using normalized color. *IEEE Trans. Syst., Man, Cybern.*, 22:64–73, 1992.

[54] Glenn Healey. Segmenting images using normalized color. *IEEE Transactions on Systems, Man, and Cybernetics*, 22(1):64–73, 1992.

[55] João F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. Exploiting the circulant structure of tracking-by-detection with kernels. In *European conference on computer vision*, pages 702–715. Springer, 2012.

[56] John R Hershey, Zhuo Chen, Jonathan Le Roux, and Shinji Watanabe. Deep clustering: Discriminative embeddings for segmentation and separation. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 31–35. IEEE, 2016.

[57] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *Annual Conference on Neural Information Processing Systems (NIPS)*, 2014.

[58] Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In *International Workshop on Similarity-Based Pattern Recognition*, pages 84–92. Springer, 2015.

[59] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM, 1999.

[60] Zhibin Hong, Zhe Chen, Chaohui Wang, Xue Mei, Danil Prokhorov, and Dacheng Tao. Multi-store tracker (muster): A cognitive psychology inspired approach to object tracking. In *CVPR*, 2015.

[61] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 831–839, 2019.

[62] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[63] Wenpeng Hu, Zhou Lin, Bing Liu, Chongyang Tao, Zhengwei Tao, Jinwen Ma, Dongyan Zhao, and Rui Yan. Overcoming catastrophic forgetting for continual learning via model adaptation. 2018.

[64] Zheng Hui, Xiumei Wang, and Xinbo Gao. Fast and accurate single image super-resolution via information distillation network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 723–731, 2018.

[65] Cong Phuoc Huynh and Antonio Robles-Kelly. A solution of the dichromatic model for multispectral photometric invariance. *International Journal of Computer Vision*, 90(1):1–27, 2010.

[66] Viren Jain and Lawrence K Saul. Exploratory analysis and visualization of speech and music by locally linear embedding. In *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 3, pages iii–984. IEEE, 2004.

[67] Seong Joon Oh, Rodrigo Benenson, Anna Khoreva, Zeynep Akata, Mario Fritz, and Bernt Schiele. Exploiting saliency for object segmentation from image level labels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4410–4419, 2017.

[68] Tilke Judd, Krista Ehinger, Frédo Durand, and Antonio Torralba. Learning to predict where humans look. In *Computer Vision, 2009 IEEE 12th international conference on*, pages 2106–2113. IEEE, 2009.

[69] Toshikazu Kato. Database architecture for content-based image retrieval. In *image storage and retrieval systems*, volume 1662, pages 112–123. International Society for Optics and Photonics, 1992.

[70] Kenneth Low Kelly and Deane Brewster Judd. *Color: universal language and dictionary of names*, volume 440. US Department of Commerce, National Bureau of Standards, 1976.

[71] Fahad S Khan, Joost Weijer, Andrew D Bagdanov, and Maria Vanrell. Portmanteau vocabularies for multi-cue image representation. In *Advances in neural information processing systems*, pages 1323–1331, 2011.

[72] Fahad Shahbaz Khan, Rao Muhammad Anwer, Joost van de Weijer, Andrew D Bagdanov, Antonio M Lopez, and Michael Felsberg. Coloring action recognition in still images. *International journal of computer vision*, 105(3):205–221, 2013.

[73] Fahad Shahbaz Khan, Rao Muhammad Anwer, Joost van de Weijer, Andrew D Bagdanov, Maria Vanrell, and Antonio M Lopez. Color attributes for object detection. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3306–3313. IEEE, 2012.

[74] Fahad Shahbaz Khan, Rao Muhammad Anwer, Joost van de Weijer, Michael Felsberg, and Jorma Laaksonen. Compact color–texture description for texture classification. *Pattern Recognition Letters*, 51:16–22, 2015.

[75] Fahad Shahbaz Khan, Joost Van De Weijer, and Maria Vanrell. Top-down color attention for object recognition. In *2009 IEEE 12th International Conference on Computer Vision*, pages 979–986. IEEE, 2009.

[76] Fahad Shahbaz Khan, Joost Van de Weijer, and Maria Vanrell. Modulating shape features by color attention for object recognition. *International Journal of Computer Vision*, 98(1):49–64, 2012.

[77] Rahat Khan, Joost Van de Weijer, Fahad Shahbaz Khan, Damien Muselet, Christophe Ducottet, and Cecile Barat. Discriminative color descriptors. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 2866–2873. IEEE, 2013.

[78] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning representations (ICLR)*, 2014.

[79] Diederik P Kingma and Lei Ba. J. adam: a method for stochastic optimization. In *International Conference on Learning representations (ICLR)*, 2015.

[80] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. page 201611835, 2017.

[81] Martin Koestinger, Martin Hirzer, Paul Wohlhart, Peter M Roth, and Horst Bischof. Large scale metric learning from equivalence constraints. In *2012 IEEE conference on computer vision and pattern recognition*, pages 2288–2295. IEEE, 2012.

[82] Jonathan Krause, Benjamin Sapp, Andrew Howard, Howard Zhou, Alexander Toshev, Tom Duerig, James Philbin, and Li Fei-Fei. The unreasonable effectiveness of noisy data for fine-grained recognition. In *European Conference on Computer Vision*, pages 301–320. Springer, 2016.

[83] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *IEEE International Conference on Computer Vision Workshops*, pages 554–561, 2013.

[84] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[85] Jason Kuen, Zhenhua Wang, and Gang Wang. Recurrent attentional networks for saliency detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3668–3677, 2016.

[86] Brian Kulis et al. Metric learning: A survey. *Foundations and Trends® in Machine Learning*, 5(4):287–364, 2013.

[87] Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. From word embeddings to document distances. In *International conference on machine learning*, pages 957–966, 2015.

[88] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[89] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[90] Yann LeCun, John S Denker, and Sara A Solla. Optimal brain damage. In *Annual Conference on Neural Information Processing Systems (NIPS)*, pages 598–605, 1990.

[91] Jeongtae Lee, Jaehong Yun, Sungju Hwang, and Eunho Yang. Lifelong learning with dynamically expandable networks. 2018.

[92] Omer Levy and Yoav Goldberg. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 302–308, 2014.

[93] Wei Li, Rui Zhao, and Xiaogang Wang. Human reidentification with transferred metric learning. In *Asian conference on computer vision*, pages 31–44. Springer, 2012.

[94] Xilai Li, Yingbo Zhou, Tianfu Wu, Richard Socher, and Caiming Xiong. Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting. In *International Conference on Machine Learning*, pages 3925–3934, 2019.

[95] Yang Li and Jianke Zhu. A scale adaptive kernel correlation filter tracker with feature integration. In *ECCV Workshop*, 2014.

[96] Yuncheng Li, Jianchao Yang, Yale Song, Liangliang Cao, Jiebo Luo, and Li-Jia Li. Learning from noisy labels with distillation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1910–1918, 2017.

[97] Zhizhong Li and Derek Hoiem. Learning without forgetting. *Transactions of Pattern Recognition and Machine Analyses (PAMI)*, 40(12):2935–2947, 2018.

[98] Pengpeng Liang, Erik Blasch, and Haibin Ling. Encoding color information for visual tracking: Algorithms and benchmark. *IEEE Transactions on Image Processing*, 24(12):5630–5644, 2015.

[99] Si Liu, Jiashi Feng, Csaba Domokos, Hui Xu, Junshi Huang, Zhenzhen Hu, and Shuicheng Yan. Fashion parsing with weak color-category labels. *IEEE Transactions on Multimedia*, 16(1):253–265, 2013.

[100] Si Liu, Jiashi Feng, Csaba Domokos, Hui Xu, Junshi Huang, Zhenzhen Hu, and Shuicheng Yan. Fashion parsing with weak color-category labels. *IEEE Transactions on Multimedia*, 16(1):253–265, 2014.

[101] Xialei Liu, Marc Masana, Luis Herranz, Joost Van de Weijer, Antonio M Lopez, and Andrew D Bagdanov. Rotate your networks: Better weight consolidation and less catastrophic forgetting. In *International Conference on Pattern Recognition (ICPR)*, 2018.

[102] Xialei Liu, Joost van de Weijer, and Andrew D Bagdanov. Rankiqa: Learning from rankings for no-reference image quality assessment. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[103] Xiaokai Liu, Hongyu Wang, Yi Wu, Jimei Yang, and Ming Hsuan Yang. An ensemble color model for human re-identification. In *Applications of Computer Vision (WACV), 2015 IEEE Winter Conference on*, pages 868–875. IEEE, 2015.

[104] Ying Liu, Dengsheng Zhang, Guojun Lu, and Wei-Ying Ma. Region-based image retrieval with high-level semantic color names. In *Multimedia Modelling Conference, 2005. MMM 2005. Proceedings of the 11th International*, pages 180–187. IEEE, 2005.

[105] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3730–3738, 2015.

[106] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Annual Conference on Neural Information Processing Systems (NIPS)*, pages 6467–6476, 2017.

[107] David G Lowe et al. Object recognition from local scale-invariant features. In *iccv*, volume 99, pages 1150–1157, 1999.

[108] Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. Thinet: A filter level pruning method for deep neural network compression. In *Proceedings of the IEEE international conference on computer vision*, pages 5058–5066, 2017.

[109] Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *European Conference on Computer Vision (ECCV)*, pages 67–82, 2018.

[110] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7765–7773, 2018.

[111] Marc Masana, Idoia Ruiz, Joan Serrat, Joost van de Weijer, and Antonio M Lopez. Metric learning for novelty and anomaly detection. In *BMVA British Machine Vision Conference (BMVC)*, 2018.

[112] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

[113] Thomas Mensink, Jakob Verbeek, Florent Perronnin, and Gabriela Csurka. Metric learning for large scale image classification: Generalizing to new classes at near-zero cost. In *European Conference on Computer Vision*, pages 488–501. Springer, 2012.

[114] Thomas Mensink, Jakob Verbeek, Florent Perronnin, and Gabriela Csurka. Distance-based image classification: Generalizing to new classes at near-zero cost. *Transactions of Pattern Recognition and Machine Analyses (PAMI)*, 35(11):2624–2637, 2013.

[115] Volodymyr Mnih and Geoffrey E Hinton. Learning to label aerial images from noisy data. In *Proceedings of the 29th International conference on machine learning (ICML-12)*, pages 567–574, 2012.

[116] A. Mojsilovic. A computational model for color naming and describing color composition of images. *IEEE Transactions on Image Processing*, 14(5):690–699, 2005.

[117] Aleksandra Mojsilovic. A computational model for color naming and describing color composition of images. *Image Processing, IEEE Transactions on*, 14(5):690–699, 2005.

[118] D. Mylonas, D. Griffin, L.D. Purver, P. Katemake, and Davidoff J. The role of primary colours in colour naming. *Under review*, 2016.

[119] Dimitris Mylonas and Lindsay MacDonald. Augmenting basic colour terms in english. *Color Research & Application*, 2015.

[120] Dimitris Mylonas and Lindsay MacDonald. Augmenting basic colour terms in english. *Color Research & Application*, 41(1):32–42, 2016.

[121] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Computer Vision, Graphics & Image Processing, 2008. ICVGIP'08. Sixth Indian Conference on*, pages 722–729. IEEE, 2008.

[122] Li Niu, Wen Li, and Dong Xu. Visual recognition by learning from web data: A weakly supervised domain generalization approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2774–2783, 2015.

[123] Michael Opitz, Georg Waltner, Horst Possegger, and Horst Bischof. Bier-boosting independent embeddings robustly. In *International Conference on Computer Vision (ICCV)*, 2017.

[124] Oleksiy Ostapenko, Mihai Puscas, Tassilo Klein, Patrick Jahnichen, and Moin Nabi. Learning to remember: A synaptic plasticity driven framework for continual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11321–11329, 2019.

[125] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 582–597. IEEE, 2016.

[126] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, et al. Deep face recognition. In *bmvc*, volume 1, page 6, 2015.

[127] C Párraga, R Benavente, R Baldrich, and M Vanrell. Psychophysical measurements to model intercolor regions of color-naming space. *Journal of Imaging Science and Technology*, 53(3):31106–1, 2009.

[128] C Alejandro Parraga and Arash Akbarinia. Nice: A computational solution to close the gap from colour perception to colour categorization. *PloS one*, 11(3):e0149538, 2016.

[129] Nikolaos Passalis and Anastasios Tefas. Learning deep representations with probabilistic knowledge transfer. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 268–284, 2018.

[130] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[131] Deepak Pathak, Philipp Krahenbuhl, and Trevor Darrell. Constrained convolutional neural networks for weakly supervised segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1796–1804, 2015.

[132] Andrea Pilzer, Stephane Lathuiliere, Nicu Sebe, and Elisa Ricci. Refine and distill: Exploiting cycle-inconsistency and knowledge distillation for unsupervised monocular depth estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9768–9777, 2019.

[133] Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 413–420. IEEE, 2009.

[134] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 512–519, 2014.

[135] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5533–5542. IEEE, 2017.

[136] Oren Rippel, Manohar Paluri, Piotr Dollar, and Lubomir Bourdev. Metric learning with adaptive density discrimination. In *International Conference on Learning representations (ICLR)*, 2016.

[137] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*, 2014.

[138] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. In *International Conference on Learning representations (ICLR)*, 2015.

[139] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

[140] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815–823, 2015.

[141] Gabriel Schwartz and Ko Nishino. Discovering perceptual attributes in a deep local material recognition network. *arXiv preprint arXiv:1604.01345*, 2016.

[142] Tyler Scott, Karl Ridgeway, and Michael C Mozer. Adapted deep embeddings: A synthesis of methods for k-shot inductive transfer learning. In *Annual Conference on Neural Information Processing Systems (NIPS)*, pages 76–85, 2018.

[143] Ozan Sener, Hyun Oh Song, Ashutosh Saxena, and Silvio Savarese. Learning transferrable representations for unsupervised domain adaptation. In *Annual Conference on Neural Information Processing Systems (NIPS)*, pages 2110–2118, 2016.

[144] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning (ICML)*, pages 4555–4564, 2018.

[145] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 806–813, 2014.

[146] Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(4):640–651, 2017.

[147] Jonathan Shen, Noranart Vesdapunt, Vishnu N Boddeti, and Kris M Kitani. In teacher we trust: Learning compressed models for pedestrian detection. In *arXiv preprint arXiv:1612.00478*, 2016.

[148] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *Annual Conference on Neural Information Processing Systems (NIPS)*, pages 2990–2999, 2017.

[149] Edgar Simo-Serra, Eduard Trulls, Luis Ferraz, Iasonas Kokkinos, Pascal Fua, and Francesc Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *IEEE International Conference on Computer Vision (ICCV)*, pages 118–126, 2015.

[150] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning representations (ICLR)*, 2015.

[151] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Annual Conference on Neural Information Processing Systems (NIPS)*, pages 4077–4087, 2017.

[152] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Annual Conference on Neural Information Processing Systems (NIPS)*, pages 1857–1865, 2016.

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

[154] Oh Hyun Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4004–4012, 2016.

[155] Julia Sturges and TW Whitfield. Locating basic colours in the munsell space. *Color Research & Application*, 20(6):364–376, 1995.

[156] Jong-Chyi Su and Subhransu Maji. Adapting models to signal degradation using distillation. In *BMVA British Machine Vision Conference (BMVC)*, 2017.

[157] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[158] Evgeniya Ustinova and Victor Lempitsky. Learning deep embeddings with histogram loss. In *Advances in Neural Information Processing Systems*, pages 4170–4178, 2016.

[159] Koen E. A. van de Sande, Theo Gevers, and Cees G. M. Snoek. Evaluating color descriptors for object and scene recognition. *PAMI*, 32(9):1582–1596, 2010.

[160] Joost Van de Weijer and Fahad Shahbaz Khan. Fusing color and shape for bag-of-words based object recognition. In *International Workshop on Computational Color Imaging*, pages 25–34. Springer, 2013.

[161] Joost van de Weijer and Fahad Shahbaz Khan. An overview of color name applications in computer vision. In *Computational Color Imaging*, pages 16–22. Springer, 2015.

[162] Joost Van De Weijer and Fahad Shahbaz Khan. An overview of color name applications in computer vision. In *International Workshop on Computational Color Imaging*, pages 16–22. Springer, 2015.

[163] Joost Van de Weijer and Cordelia Schmid. Applying color names to image description. In *2007 IEEE International Conference on Image Processing*, volume 3, pages III–493. IEEE, 2007.

[164] Joost van de Weijer and Cordelia Schmid. Applying color names to image description. In *IEEE International Conference on Image Processing (ICIP)*, San Antonio, USA, 2007.

[165] Joost Van De Weijer, Cordelia Schmid, Jakob Verbeek, and Diane Larlus. Learning color names for real-world applications. *Image Processing, IEEE Transactions on*, 18(7):1512–1523, 2009.

[166] Joost van de Weijer, Cordelia Schmid, Jakob Verbeek, and Diane Larlus. Learning color names for real-world applications. *IEEE Transactions on Image Processing*, 18(7):1512–1524, july 2009.

[167] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. *Computation & Neural Systems Technical Report, CNS-TR-2011-001*, 2011.

[168] Jian Wang, Feng Zhou, Shilei Wen, Xiao Liu, and Yuanqing Lin. Deep metric learning with angular loss. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2612–2620. IEEE, 2017.

[169] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu. Learning fine-grained image similarity with deep ranking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1386–1393, 2014.

[170] Lijun Wang, Huchuan Lu, Yifan Wang, Mengyang Feng, Dong Wang, Baocai Yin, and Xiang Ruan. Learning to detect salient objects with image-level supervision. In *Conf. Comput. Vis. Pattern Recognit.(CVPR)*, pages 136–145, 2017.

[171] Xiaolong Wang and Abhinav Gupta. Unsupervised learning of visual representations using videos. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2794–2802, 2015.

[172] Yan Wang, Sheng Li, and Alex C Kot. On branded handbag recognition. *IEEE Transactions on Multimedia*, 18(9):1869–1881, 2016.

[173] Yuhang Wang, Jing Liu, Jinqiao Wang, Yong Li, and Hanqing Lu. Color names learning using convolutional neural networks. In *Image Processing (ICIP), 2015 IEEE International Conference on*, pages 217–221. IEEE, 2015.

[174] Zhengxiang Wang, Yiqun Hu, and Liang-Tien Chia. Image-to-class distance metric learning for image classification. In *European Conference on Computer Vision*, pages 706–719. Springer, 2010.

[175] Zhouxia Wang, Tianshui Chen, Guanbin Li, Ruijia Xu, and Liang Lin. Multi-label image recognition by recurrently discovering attentional regions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 464–472, 2017.

[176] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010.

[177] Jason Weston, Frédéric Ratle, Hossein Mobahi, and Ronan Collobert. Deep learning via semi-supervised embedding. In *Neural Networks: Tricks of the Trade*, pages 639–655. Springer, 2012.

[178] Chao-Yuan Wu, R Manmatha, Alexander J Smola, and Philipp Krahenbuhl. Sampling matters in deep embedding learning. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2840–2848, 2017.

[179] Chenshen Wu, Luis Herranz, Xialei Liu, Yaxing Wang, Joost van de Weijer, and Bogdan Raducanu. Memory replay gans: learning to generate images from new categories without forgetting. In *Annual Conference on Neural Information Processing Systems (NIPS)*, 2018.

[180] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Online object tracking: A benchmark [c]. computer vision and pattern recogniton (cvpr). In *2013 IEEE Conference on, IEEE*, 2013.

[181] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the*

IEEE Conference on Computer Vision and Pattern Recognition, pages 374–382, 2019.

[182] Yongqin Xian, Zeynep Akata, Gaurav Sharma, Quynh Nguyen, Matthias Hein, and Bernt Schiele. Latent embeddings for zero-shot classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 69–77, 2016.

[183] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2691–2699, 2015.

[184] Dan Xu, Wanli Ouyang, Xiaogang Wang, and Nicu Sebe. Pad-net: Multi-tasks guided prediction-and-distillation network for simultaneous depth estimation and scene parsing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 675–684, 2018.

[185] Huijuan Xu and Kate Saenko. Ask, attend and answer: Exploring question-guided spatial attention for visual question answering. In *European Conference on Computer Vision*, pages 451–466. Springer, 2016.

[186] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, pages 2048–2057, 2015.

[187] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*, 2014.

[188] Hong-Ming Yang, Xu-Yao Zhang, Fei Yin, and Cheng-Lin Liu. Robust classification with convolutional prototype learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3474–3482, 2018.

[189] Yang Yang, Jimei Yang, Junjie Yan, Shengcai Liao, Dong Yi, and Stan Li. Salient color names for person re-identification. In *Computer Vision (ECCV), 2014 European Conference on*, pages 536–551. Springer, 2014.

[190] Yang Yang, Jimei Yang, Junjie Yan, Shengcai Liao, Dong Yi, and Stan Z Li. Salient color names for person re-identification. In *Computer Vision–ECCV 2014*, pages 536–551. Springer, 2014.

[191] Yiming Yang. An evaluation of statistical approaches to text categorization. *Information retrieval*, 1(1-2):69–90, 1999.

[192] Bangpeng Yao, Xiaoye Jiang, Aditya Khosla, Andy Lai Lin, Leonidas Guibas, and Li Fei-Fei. Human action recognition by learning bases of action attributes and parts. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1331–1338. IEEE, 2011.

[193] Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4133–4141, 2017.

[194] Lu Yu, Yongmei Cheng, and Joost van de Weijer. Weakly supervised domain-specific color naming based on attention. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 3019–3024. IEEE, 2018.

[195] Lu Yu, Vacit Oguz Yazici, Xialei Liu, Joost van de Weijer, Yongmei Cheng, and Arnau Ramisa. Learning metrics from teachers: Compact networks for image embedding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2907–2916, 2019.

[196] Lu Yu, Lichao Zhang, Joost van de Weijer, Fahad Shahbaz Khan, Yongmei Cheng, and C Alejandro Parraga. Beyond eleven color names for image understanding. *Machine Vision and Applications*, 29(2):361–373, 2018.

[197] Zejian Yuan, Badong Chen, Jianru Xue, Nanning Zheng, et al. Illumination robust color naming via label propagation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 621–629, 2015.

[198] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *International Conference on Learning representations (ICLR)*, 2016.

[199] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning (ICML)*, pages 3987–3995. JMLR. org, 2017.

[200] Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4320–4328, 2018.

[201] Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

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

[203] Neta Zmora, Guy Jacob, Lev Zlotnik, Bar Elharar, and Gal Novik. Neural network distiller, June 2018.

[204] Heinrich Zollinger. Why just turquoise? remarks on the evolution of color terms. *Psychological Research*, 46(4):403–409, 1984.