REPRESENTATION LEARNING OF IMAGE RECOGNITION: DIVERSITY, ASPECT RATIO, INVARIANCE, AND COMPOSITION

by

Qiuyu Chen

A dissertation submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in
Computing and Information Systems

Charlotte

2021

Approved by:

_____
Dr. Jianping Fan

_____
Dr. Aidong Lu

_____
Dr. Jing Yang

_____
Dr. Min Shin

_____
Dr. Weidong Tian

ABSTRACT

QIUYU CHEN. Representation Learning of Image Recognition: Diversity, Aspect Ratio, Invariance, and Composition. (Under the direction of DR. JIANPING FAN)

Deep neural networks (DNN) are proved to be effective and improve the performance dramatically in various kinds of computer vision tasks. The end-to-end learning manner in training DNN consistently shows the powerful modeling ability and consequently mitigates the dedicated efforts for expert feature engineering. On the other hand, it raises the issue that how to improve the black-box network with better representation (feature) learning especially when the learned representations and classifiers are tied together in the manner of supervised learning. In this work, representation learning is studied in four perspectives of different fields, *i.e.* diversity in ensemble learning, aspect ratio in image aesthetics assessment, invariance in identification task, and composition in color attribute recognition.

In light of analyzing the bottleneck of black-box network and designing better representation learning for target tasks, we introduce that: (a) Ensemble learning relies on the diversity of the complementary neural networks, in both feature representations and classifier representations. A diverse representation learning method, namely learning-difficulty-aware embedding, is proposed to adaptively reconcile learning attentions for different categories by training a series of networks with diversified representations sequentially; (b) Widely-adopted data augmentation method in image recognition deteriorates aspect ratios, which is an important factor in image aesthetics assessment. An aspect ratio representation learning method, namely adaptive fractional dilated convolution, is proposed to explicitly preserve the learning representation related to aspect ratios by adjusting the receptive fields adaptively and natively; (c) Identification tasks, *e.g.* person re-identification, aim at learning representations that are robust to interfering variances, *e.g.* lighting variances, view vari-

ances, pose variances. An invariance representation learning method, namely anchor loss, is proposed to train a robust feature extractor, which distills the identity-related representations while disentangling and removing interfering variances by global supervision under local mini-batch training; (d) Color recognition is entangled with compositional representation in both visual perception and language attentions. A compositional learning module with attention to key colors is proposed to learn better color representations. Besides, another compositional learning method, namely classifier as descriptor, is proposed for long-tail color recognition by incorporating the rich knowledge in classifier representations to remove the bias from bias-trained model.

Through extensive experiments and thorough analysis, we demonstrate some novel insights about the impacts of four factors, *i.e.* diversity, receptive field, invariance, and composition. Several methods are proposed to learn better representations for those factors, achieving state-of-the-art results in different tasks.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

1.1 Representation Learning of Image Recognition

With the availability of massive training images and the rapid growth of computational powers, we are now able to develop scalable learning algorithms based on deep convolutional neural networks to support large-scale visual recognition tasks, *e.g.* image classification [6, 7, 8, 9, 10, 3, 11, 12, 13], object detection [14, 15, 16, 17, 18, 19, 20, 21] and semantic segmentation [22, 23, 24, 25]. Compared with traditional learning methods for computer vision where visual feature engineering and data fitting are separate, deep learning provides a unified model in an end-to-end learning manner, *i.e.* fitting the data directly from images to labels with loss functions through iterative optimization. On the one hand, without bells and whistles, deep neural networks result in consistent and significant improvement, achieving satisfactory requirements in real-life applications and benefiting people's life in a new automation era. On the other hand, however, such end-to-end learning encapsulates the models into a black box, which makes the interpretation and imposing conditions less accessible and builds the barriers to improve representation learning. Moreover, vanilla deep neural networks are not satisfactory solutions, and improving representation learning has demonstrated its key power in many milestones during the rapid development of deep learning.

Representation learning is to learn representations of the data that make it easier to extract useful information when building classifiers or other predictors [26]. Many fundamental advances in computer vision using deep learning can be understood from the perspective of representation learning. Residual connection [3] successfully solves the generalization problem in deep networks by exploiting all the parameters

in the network to learn more meaningful features without gradient vanishing [27] and behaving like many combinations of different small networks that learn diverse representations [28]. Dilated convolution [23, 24, 29] targets semantic (high-level) representation learning in dense prediction while preserving the localization ability by dilating the receptive fields in convolutional kernels without the downsampling of feature maps. FPN [30] performs the feature fusion in a top-down manner with lateral connections to extract better representations in the features maps at different scales for object detection. Group feature extraction, *e.g.* ResNext [31], Seperable Conv [11], ShuffleNet Unit [13], efficiently incorporates local connections with less computational cost while preserving the strong modeling ability of networks to learn useful representations. Tan *et al.* [32] systematically study the model scaling and identify that carefully balancing network depth, width, and resolution can lead to better representation learning under the same budget of computation and parameters. Disentanglement representation learning [33, 34, 35] in generative adversarial networks substantially improves the quality of synthetic images and enables the manipulation of semantics and variations for image generation. Dosovitskiy *et al.* [36] utilize self-attention [37] with flexible receptive fields and global connections to replace convolutional layers which can only extract features by gradually expanding receptive fields, achieving representation learning across the entire image even in the lowest layers.

Despite the generality for some methods, *e.g.* residual connection [3] and self-attention with block images [36], good representations could vary from different domains, where some representations are more important to achieving the specific goals. Many methods of representation learning focus on the identified important issues to help target tasks, *e.g.* semantic and localization representation learning for dense prediction [23, 24, 29], multi-scale semantic presentation learning for object detection [30], efficient representation learning [31, 11, 13, 32] for model compression,

disentanglement representation learning [33, 34, 35] for image generation.

In light of the aforementioned success and importance of representation learning for image recognition, we investigate four crucial factors in respecting domains: diverse representation learning for ensemble learning, aspect ratio representation learning for image aesthetics assessment, invariance representation learning for person re-identification, compositional representation learning for color recognition. Specifically, (a) For ensemble learning, where multiple networks are managed together to contribute a stronger model, *e.g.* boosting [38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52] and stacking [53, 54, 55], we target the diversity of complementary networks in decision space and propose a difficulty-aware-embedding method to learn a serious of networks with diversified focus in an easy-to-hard way according to the learning complexity; (b) For image aesthetics assessment [1], where aspect ratios and compositions are altered by widely-adopted data augmentation in general image recognition [8], we develop a new convolution layer, called adaptive fractional dilated convolution, that explicitly learns the representations related to aspect ratios; (c) For person re-identification, where distance-based feature retrieval is used during test [56, 57, 58], we introduce an anchor loss to learn better representations that distill the identity-related features and dissect interfering variance in latent feature space. (e) For color attribute recognition, where we identified that language attention and visual attention are entangled, a color compositional learning module is proposed to improve representation learning taking composition characteristics for visual attention, and debiasing method, called classifier as descriptor, is proposed for long-tail color recognition taking composition characteristics for language attention.

In the following context, we illustrate the motivation of these methods respectively.

1.2 Diversity

Deep learning has a strong modeling ability that could even easily fit random labels in large-scale training data [59]. Thus, the representation learning that models

the training distribution perfectly does not necessarily lead to better generalization. Diversity is an important factor about a good representation learning that alleviates overfitting and generalizes well in testing distribution. From the perspective of optimization and overfitting, diverse representations fully exploit large number of learning parameters in deep neural network and endeavor to find a better local optimal instead of a trivial optimal or complex one. A better local optimal could cover many possible patterns from training data while a trivial optimal may extract monotonous features and a complex optimal may simply memorize the data. As a result, diverse representation learning could be served as an implicit regularization. From another perspective of generalization, diverse representations can be robust to shifts and variations between training distribution and testing distribution because the final decision considers diverse factors and reduces the uncertainty for the test of generalization.

There are mainly two streams for diverse representation learning, single network and ensemble networks. In the single network, where only a single classifier layer is used, diversity in representation learning is achieved by network architecture design or regularization. For network architecture design, the methods could be related to efficient networks, where the goal is to develop better generalization ability under the same budget of computation cost during inference. Diversity plays a key part in the design of efficient architecture by fully exploiting available learning parameters and obtaining better results. For example, group convolution, *e.g.* ResNext [31], Separable Conv [11], ShuffleNet Unit [13], removes global and monotonous connections and attends local and grouped features, achieving diverse representation learning while reducing the computational cost. Besides, the compose of diverse filters within network modules is another approach for diverse network architectures, *e.g.* GoogleNet collections [8, 60, 61, 62] based on the combination of different kernel sizes. For regularization, orthogonality is a well studied approach in diverse representation learning through weight initialization [63, 64], SVD-based hard or-

thogonality constraints [65, 66, 67], Gram-matrix-based soft orthogonality regularizations [64, 68, 69]. Another direction achieving diversity by regularization is through random dropping/whitening either from input, *e.g.* random erasing [70], CutOut [71], or from internal feature maps or connections, *e.g.* DropOut [72], DropConect [73], DropBlock [74], DropPath [75].

In this work, we focus on the diverse representation learning in ensemble networks, *i.e.* the diversity of multiple complementary networks. AdaBoost [40, 39] diversifies complementary networks by adjusting the weights in sample level sequentially. Despite its success before the rise of deep learning, we discover that it is not suitable for strong learners, *i.e.* deep neural networks. One reason is that AdaBoost targets fitting data in training sample level, where a single deep neural network could already achieve almost zero errors in the training dataset. As a result, reweighting on sampling level could lead to either overfitting when only a small portion of samples are effectively weighted in the second iteration, or random ensemble when almost all of the training samples are equally weighted. MixDCNN [76] proposes to reweight the aggregation coefficients at category level by dynamically guiding the diversity among paralleling complementary networks. The diversity of complementary networks that are trained in parallel counts on the randomness in weight initialization and stochastic optimization. Although it shows effectiveness in fine-grained classification tasks, we find that those randomness fails to contribute to the diversity in general image classification, producing similar effects as the average ensemble. Alternatively, DoE [77] diversifies the complementary networks by attending to different task subspaces in category level which contain overlapping to each other. The drawback of the method is that it requires extra effort to carefully fuse the decisions from different subspace into the final decision.

Based on the observation, we propose a novel network embedding method, that learns a series of diverse complementary networks with discriminative attention on

different categories [78]. The discriminative attention (weights) are allocated by feedback from the validation dataset from previous training iterations and calculated in category level, mitigating the overfitting problem in AdaBoost [40, 39]. The guided optimization with weighted objective function provides better promise towards a diverse optimal comparing to MixDCNN [76]. Specifically, we kick off the training of the first deep neural network by an average loss function in a traditional training manner, and then estimate the learning complexities/difficulties respecting the category accuracies of validation results. By assigning larger importance (weights) for hard object classes in a weighted loss function during following iterations, the difficulty-aware embedding algorithm can train multiple complementary deep networks sequentially to achieve higher accuracy rates on recognizing different subsets of object classes in an easy-to-hard way, so that they can compensate and enhance each other. By combining such complementary deep networks adaptively to generate a more discriminative ensemble network, it can achieve higher overall accuracy rates on large-scale visual recognition by effectively maintaining high accuracy rates for the easy object classes (which can be achieved by the first deep network) while improving the accuracy rates for the hard ones at certain degrees (which are achieved by residual complementary networks from the consequent iterations).

1.3 Aspect Ratio

Aspect ratios of objects are handled differently in different tasks. For object detection based region proposal network [15, 16, 17, 18, 19, 20, 21], where the anchor boxes could provide estimated guidance in aspect ratio distribution of objects, aspect ratios are contained in groundtruth annotations and predicted indirectly through the output of offset location. For semantic segmentation, the localization is retained by dense annotations regardless of cropping or distorting in input images, and consequently, the aspect ratios could still be computed by localization prediction. Thus, the aspect ratios in those localization tasks are directly managed by annotations. For

general object classification tasks, aspect ratios are usually neglected, *e.g.* RoI Align Pooling [16] in classifier head of object detection and random-size cropping [8] in data augmentation for image classification. On the one hand, the data augmentation methods proposed by [8], *i.e.* image cropping and resizing, are widely used for preventing overfitting in the image recognition task for the sake of scale and distortion invariance [31, 3, 79, 80, 11, 12, 13]. On the other hand, due to constraints of graphic memory and computational efficiency, mini-batch training, *i.e.* concatenating several images together into mini-batch (a 4-dimensional tensor) by sampling and updating the model parameters by back-propagation iteratively, is adopted when training deep convolutional neural networks.

In this work, we address the task of image aesthetics assessment, where representation learning about aspect ratios matters. When humans evaluate the aesthetics preference of images, aspect ratios of objects and image composition are important factors for visual perception. A robust model for accurate aesthetic assessment, where the goal is to predict an aesthetics score or distribution given an image [1], should learn the representations related to aspect ratios. Unfortunately, the aforementioned general practice of data augmentation [8], alters the compositions and object aspect ratios, which introduces label noise and harms the task of aesthetics assessment (Fig. 3.1). Besides, without cropping and resizing, different images with different aspect ratios can not be concatenated together, *e.g.* only one original-size image can be feed into the network at a time in [81].

To consider the representation learning of aspect ratios related to image aesthetics, randomly cropping the images or extracting features based on sampling patches [82, 83, 84], deteriorates the composition of original images, which is another important factor for aesthetics assessment. In order to preserve composition and adapt mini-batch training in deep learning for image aesthetics assessment, a natural solution is to warp the mini-batch sampling images into the same size. Then, one critical but

unsolved issue is how to seamlessly learn representation for the information of object aspect ratios for robust image aesthetics modeling.

By looking deeper into the neural networks at kernel level, we identify that the representation learning of aspect ratio is related to receptive fields. In deep convolutional neural networks where a collection of convolution and pooling layers are stacked sequentially, the receptive field is magnified as the layer goes deeper. This design is inspired by that high-level semantic features (*e.g.* categories), which require global receptive fields, can be extracted from the gradual aggregation of the low-level visual features (*e.g.* colors), which requires local receptive fields. We propose an adaptive fractional dilated convolution, which adaptively and dynamically adjusts the receptive fields in accordance to the original aspect ratios [85]. The receptive field is dilated along with the image resizing operation dynamically and adaptively, keeping the receptive field (*i.e.* sampling areas) the same as the original one. It thus can extract the object features with the consideration of original aspect ratios. As a result, the aspect ratios are embedded seamlessly and natively at convolution level. Specifically, for each convolutional layer whose kernel size is larger than 1×1,*i.e.* the receptive field is magnified, we dilate convolution kernels adaptively by a calculated dilation rate regarding the original aspect ratios. Since the aspect ratio could be fractional, such dilation rate could be fractional as well. Technically, the interpolation of nearest two integer dilated kernels are used to cope with the misalignment of fractional sampling in discrete image/feature space. We note that the dynamic and adaptive kernel construction process in the proposed convolution uses the same learning parameters as the normal convolution and thus no extra model parameters are introduced. Through this aspect-ratio-aware embedding, we can successfully preserve both composition and aspect ratio in a parameter-free manner.

1.4 Invariance

Variances of image distribution for image recognition can be attributed to object location, image composition, contextual lighting, human pose, background, subjective or noisy annotations, viewpoint, and *et al*. Some variances are beneficial for targeting tasks and representation learning to capture the variance is crucial. For example, variance in generative models based on Generative Adversarial Networks [86] or Variational Auto Encoder [87], provides better interpretability in latent representation and access for controllable manipulation. More recently, the disentanglement representation learning distills the variance across different domains [33, 34, 88] or different manipulations [35] demonstrates huge success for generating high-quality and diverse images. However, many variances are considered noisy and intended to be dissected. The representation learning that is invariant to those factors becomes important. For a well-known example, convolutional layers [89] are designed to be translation invariant. And the fundamental posit for data augmentation is to train a network that is robust to different variance, *e.g*. lighting invariance through color jittering, translation invariance through cropping, distortion invariance through resizing.

In this work, we investigate the representation learning that removes the variance for person re-identification, where the goal is to train a feature extractor that extracts the identity information and to be tested in distance-based retrieval for identification [58, 57, 56]. Firstly, we visit the different mechanisms of invariance between the identification task and general classification task. Comparing to general classification task, identification shares a similar goal that learns representations to be more separable. However, person re-identification requires more separable features since different people could have very similar identification features and interfering noises. A robust person re-identification model relies on the representation learning that exceptionally distills the identity-related features and is extremely invariant to interfering noises. On the contrary, the requirement of invariance for general classification tasks is not

comparable. Even adding some variances during the training proves to be helpful for better representation learning and generalization, *e.g.* label smoothing [61] which is further studied in [90].

To learn a better invariant representation learning for person re-identification, many efforts are made to explicitly decouple the invariance from identity-related features. There are mainly two directions, region-based methods, and disentanglement-based methods. For region-based methods, the general practice is to use extra localization information to divide the invariance representation learning into sub-regions, *e.g.* vertical partition [91], pose [92], segmentation [93], foreground [94], attributes [95]. However, the difficulty of region-based methods lies in how to balance different sub-features and aggregate local representations for final retrieval. For disentanglement-based methods, an auto-encoder is used to decouple the latent representation into variance code and invariance code, *e.g.* DGNet [96]. The disentanglement representation is useful for interpretation and visualization. However, the intention of modeling requirement for invariance code might impair the ultimate goal for identification task, where the invariance code is not used.

Bearing that, we attempt to develop invariance representation learning without the explicit efforts of either localization or disentanglement. Thus, we propose a method for invariance representation learning that attends to remove the variances exceedingly in an end-to-end manner [97]. Specifically, we propose to extract the features from the training samples belonging to the same identity and then aggregate them to build the identity anchor in latent feature space. The latent anchors comply the intrinsic feature distribution since it is extracted from the training model supervised with identity labels. Furthermore, they dissect the variances (*e.g.* background, pose, lighting) by distilling the rich knowledge from the cluster distribution in an explicit manner. Moreover, this training process can successfully break the vision constraint under the local mini-batch training with well-aligned supervision from the viewpoint of global

distribution. Consequently, this method, called the identity-anchor-aware embedding, can inhabit a more stable and guided optimization towards better representation and generalization to learn identity representations.

1.5 Composition

Compositional representation learning is to explicitly learn a decoupled representations in latent space and resemble the representations in the compositional components for final prediction. We note that compositional representation learning is related to some traditional representation learning methods, *e.g.* dictionary learning [98], sparse encoding [99], bag of words [100]. In this work, we focus on the representation learning approaches in deep learning.

One the most directly related field is zero-shot compositional learning (ZSCL) [101, 102, 103, 104, 104], where training and test datasets contain non-overlapping attribute and object pairs while categories of attributes or objects are the overlapped. Through the learning of interaction between decoupled attributes and objects during training, the task targets to generalize the recognition of a novel combination of new attribute and object pairs. Misra *et al.* [101] propose to train a transformation network that learns to predict composed pair for a given object and attribute primitives as input. Those visual primitives are embeddings from pre-trained classifier for attributes and objects respectively, *i.e.* linear SVMs trained on features extracted from VGG networks [7]. In [105], attributes are modeled as multiplication operators that can be applied to objects. where attribute primitives and object primitives are the inputs, and the composed pair classifiers are the outputs. Purushwalkam *et al.* [102] propose to train gating modular networks to produce gating vectors for the multi-layer scoring model taking the attribute and object Glove embeddings [106]. Li *et al.* [103] incorporate symmetry principle in Coupling Network and Decoupling Network, and use relative moving distance in the latent feature pace after the two modules to train the attribute change. Nan *et al.* [104] add reconstruction loss to compositional visual

features besides optimization between the visual attribute-object pair embedding and linguistic attribute-object embedding.

Another task related to compositional representation learning is unsupervised image-to-image translation, where the goal is to translate an image into another domain without paired annotations. Recent methods propose to learn compositional representations in latent space, *e.g.* domain-invariant and domain-specific components in [34], content and style components in [33], multi-level transformation components in [35],

Moreover, memory networks are also related to compositional representation learning, where query representation is constructed by referencing memory components. Memory networks are applied in the NLP research for document Q&A [107, 108], where memorable information is separately embedded into keys (input) and values (output) feature vectors. Keys aim to address relevant memories whose corresponding values are returned. Recently, the memory networks have been applied to some vision problems such as personalized image captioning [109], visual tracking [110], and video instance object [111]. More recently, self-attention [37] achieves huge success in multiple NLP benchmarks [112], where the representation of input sequences are attended to each other to learn better representations. In computer vision, it has been aroused widely interests in incorporating self-attention. An natural extension is for sequential learning tasks to leverage long-range time or spatial contextual information, *e.g.* video processing [113, 114, 111], image classification [115, 116, 36], object detection [117, 118] and vision-language tasks [119, 120, 121].

In this work, we explore the problem of predicting the color attributes of identified objects in images. Compared to most other attributes, however, each color admits a rather specific definition, and its compositionality is well-studied. In physics, color composition is a classic problem, which has been studied for centuries. For example, in the 17^{th} century, Issac Newton proposed a theory of color which includes his famous

prism experiment, in which white light was shown to be composed of a range of color spectrum. In the digital age, most colors can be represented by sets of coordinates in key colors space. Motivated by the composition and interpolation theories of colors, we design a novel attention module, where color attribute prediction is attended to the references and interactions of key colors. Those key colors behave like the prior basis (*e.g.* father colors for the color while hierarchy), spanning a decoupled spaces to make prediction aware of color composition and learn better representation. In addition, an effective debiasing method, called classifier as discriptor, is proposed for long-tail color recognition, outperforming state-of-the-art methods in long-tail recognition [4] and unbiased scene graph generation [5]. Specifically, we use the rich representations of the output from bias-trained classifiers to remove the bias in a manner of distance calculation.

1.6 Contribution and Outline

In summary, our **contributions** are summarized as:

- We investigate four important factors for representation learning of image recognition: diversity, aspect ratio, invariance and composition
- In Chapter 2, a *diverse* representation learning method, which adaptively learns representations in an easy-to-hard way according to the learning difficulties, is proposed for network ensemble.
- In Chapter 3, an *aspect ratio* representation learning method, which intentionally learns representations related to aspect ratios, is proposed for image aesthetics assessment.
- In Chapter 4, an *invariance* representation learning method, which dissects the interfering variances, is proposed for person re-identification.
- In Chapter 5, two *compositional* representation learning methods, one composes

query representations from key color representations and the other leverages rich knowledge from classifiers as representations to remove the bias, is proposed for color recognition.

CHAPTER 2: DIVERSE REPRESENTATION LEARNING FOR NETWORK ENSEMBLE

2.1 Problems And Motivation

2.1.1 Learning Complexity of Different Categories

With the availability of massive training images and the rapid growth of computational powers of GPUs, we are now able to develop scalable learning algorithms to support large-scale visual recognition. By learning high-level features and a N-way softmax in an end-to-end multi-layer manner, deep learning [6, 7, 8, 9, 10, 3]has demonstrated its outstanding performance on large-scale visual recognition because of its strong ability on learning highly invariant and discriminant features. Under the scenario of large-scale visual recognition, some object classes could be harder to be recognized than others because they have higher learning complexities and it could be very hard for the visual recognition systems to achieve high accuracy rates for such hard object classes, on the other hand, some object classes could be easier to be recognized because they have lower learning complexities and it is much easier for the visual recognition systems to achieve high accuracy rates for such easy object classes. As a result, the errors from the hard object classes and the easy ones may have significantly different effects on optimizing the joint objective function for deep network training even they may have close values. Thus it is very attractive to develop new algorithms that can treat the errors from the hard and easy object classes differently and learn their deep networks sequentially in an easy-to-hard way.

2.1.2 Diverse Attentions of Ensemble Learning

Before deep learning becomes so popular, ensemble learning has demonstrated good performance by training and combining multiple complementary weak classifiers to construct more discriminative ensemble classifier [48, 42, 126]. By assigning larger weights to hard samples (which are misclassified by the current weak classifier), boosting can learn a complementary weak classifier at the next training round by paying more attentions on such hard samples and improving their accuracy rates dramatically. By adapting the attentions of the training samples according to their error strengths, the boosting algorithm can train multiple complementary weak classifiers sequentially for achieving higher accuracy rates on distinguishing different subsets of training samples in an easy-to-hard way. Thus it is very attractive to invest whether boosting [48, 42, 126] can be integrated with deep learning [6, 7, 8, 9, 10, 3] to train multiple complementary deep networks for achieving higher overall accuracy rates on large-scale visual recognition.

By using deep neural networks to replace the weak classifiers in the traditional boosting framework, boosting of deep neural networks has been investigated and some interesting researches have been done recently [43, 39, 45, 40, 47, 49, 41]. All these existing deep ensemble algorithms simply use the weighted training errors (proposed by Adaboost [38]) to replace the softmax errors (used in deep learning), where the underlying deep neural networks directly weight the training errors at the sample level but completely ignore which object classes cause such training errors, *e.g.*, when the training errors caused by different object classes have close strengths, they are assumed to have similar contributions on optimizing the joint objective function for deep network training. As mentioned above, large numbers of object classes may have different learning complexities, the training errors from the hard object classes and the easy ones may play significantly different roles in optimizing their joint objective function even they may have close strengths (values). As a result, weighting the

training errors directly at the sample level without considering which object classes cause such errors may not be able to optimize the joint deep network as we expect, *e.g.* such joint deep network can easily be optimized by achieving lower error rates on the easy object classes while spending less efforts on improving the accuracy rates for the hard object classes because it is much harder to achieve the same level of error rate reduction for the hard object classes. Therefore, it is very attractive to invest learning-difficulty-aware embedding algorithms that can weight the training errors at the object class level and train the deep networks for the hard object classes and the easy ones sequentially in an easy-to-hard way, so that such complementary deep networked can be embedded adaptively to generate more discriminative ensemble network.

To generate more discriminative ensemble network [127], one standard approach is to diversify the underlying deep networks being embedded, and three types of solutions can be invested: (a) *weighting the training samples* according to their error rates and most existing deep boosting algorithms [43, 39, 45, 40, 47, 49, 41] belong to this direction; (b) *learning multiple deep networks by using different model parameters or using various sample subsets* and some deep embedding algorithms [76, 128, 50, 129, 54, 55, 44] belong to this direction; (c) *training multiple complementary deep networks*, e.g., such complementary deep networks are trained sequentially for achieving higher accuracy rates on recognizing different subsets of object classes in an easy-to-hard way, so that they can compensate and enhance each other and can be embedded to generate more discriminative ensemble network. According to the best of our knowledge, the third direction (*i.e.* training multiple complementary deep networks sequentially and embedding them adaptively) has not been explored so far.

2.1.3 Diversifying Complementary Networks Adaptively and Sequentially

In this chapter, a learning-difficulty-aware embedding algorithm is developed to train multiple complementary deep networks sequentially for achieving higher accu-

racy rates on recognizing different subsets of object classes in an easy-to-hard way, so that they can compensate and enhance each other and can be embedded to generate more diverse ensemble network. Our proposed embedding algorithm has made the following ***major contributions***: (a) it can weight the training errors at the object class level, *e.g.* automatically assigning larger weights to the hard object classes and smaller weights to the easy ones; (b) it can train multiple complementary deep networks sequentially for achieving higher accuracy rates on recognizing different subsets of object classes in an easy-to-hard way, so that they can compensate and enhance each other; (c) it can embed multiple complementary deep networks adaptively to generate more discriminative ensemble network, which can effectively maintain high accuracy rates for the easy object classes (which can be achieved by the first deep network) while improving the accuracy rates for the hard ones at certain degrees (which are achieved by residual complementary networks from the second iteration); (d) it can achieve higher overall accuracy rates on large-scale visual recognition.

2.2 Related Work

In this section, we briefly review the most relevant researches on three areas: (a) Deep learning; (b) Deep ensemble learning; (c) Automatic machine learning.

Deep Learning. Deep learning has demonstrated its outstanding abilities on large-scale visual recognition [6, 7, 8, 9, 10, 3]. However, large numbers of object classes may have significant differences on their learning complexities, e.g., some object classes may be harder to be recognized than others. As a result, learning a joint deep network for all the object classes may not be an optimal solution for large-scale visual recognition, e.g., even such joint deep network can achieve high accuracy rates on recognizing the easy object classes, it may still obtain very low accuracy rates on recognizing the hard ones. Thus it is very attractive to develop new approaches to learn the deep networks for the easy object classes and the hard ones sequentially in an easy-to-hard way, so that such complementary deep networks can be embedded

adaptively to generate more discriminative ensemble network. By assigning different weights to the training samples adaptively, boosting [38] has provided an easy-to-hard approach to train a set of complementary weak classifiers iteratively.

Deep Ensemble learning. Some deep ensemble algorithms have been developed by seamlessly integrating boosting with deep neural networks to improve the performance in practice [43, 39, 45, 40, 47, 49, 41]. Schewenk and Bengio [39, 45] proposed the first work to integrate Adaboost with deep neural networks for online character recognition application. Zhou et al. [40] extended the Adaboosting neural networks algorithm for credit scoring. All these methods combine the merits of boosting and deep neural networks: they train each base network either using a different sample set by re-sampling with a probability distribution derived from the error weight, or directly using the weighted cost function for the base deep network.

Alternatively, Saberian at al. [51] proposed a margin enforcing loss for multi-class boosting and presented two ways to minimize the resulting risk: (a) the coordinate descent approach, which updates one predictor component at a time; (b) the directional functional derivative approach, which updates all the components jointly. By using the first approach (i.e., coordinate descent), Corte et al. [46] designed the ensemble learning algorithm for binary-class classification by using deep decision trees as the base classifiers and gave the data-dependent learning bound of the convex ensembles, and Kuznetzov et al. [52] further extended it to multi-class version. By applying the second approach (i.e., directional derivative descent), Moghimi et al. [43] developed an algorithm for boosting deep CNNs (convolutional neural networks) based on the least squares between the weights and the directional derivatives, which differs from the original method based on the inner product of the weights and the directional derivative in [51]. All these deep boosting algorithms focus on seeking the optimal ensemble predictor via changing the error weights at the sample level: they either update one component of the predictor per boosting iteration, or update all the com-

ponents simultaneously.

All the existing deep ensemble algorithms [43, 39, 45, 40, 47, 49, 41] directly weight the training errors at the sample level and completely ignore which object classes cause the training errors, e.g., when the training errors caused by different object classes have close strengths (values), they are assumed to have similar contributions on optimizing the joint objective function for deep network training. Under the scenario of large-scale visual recognition, large numbers of object classes may have different learning complexities, even the training errors from the hard object classes may have close values with the errors from the easy ones, they may have significantly different effects on optimizing their joint objective function for deep network learning. Unfortunately, weighting the training errors directly at the sample level without considering which object classes cause such training errors may not be able to achieve the same effects as weighting the training errors at the object class level according to their learning complexities, as a result, such joint deep network can easily be optimized by achieving higher accuracy rates on recognizing the easy object classes but spending less efforts on improving the accuracy rates for the hard object classes. Thus it is very attractive to develop new approaches that can learn the deep networks for the easy object classes and the hard ones sequentially in an easy-to-hard way.

Auto ML. Recently, instead of designing the network structures manually, there are several efforts in developing algorithmic solutions to generate the optimal architectures by reinforcement learning [130, 131, 132], evolution [133] and differentiable architecture search [134]. In addition, motivated by eliminating the overlap singularity inherent and neuron collapses, GUNN [135] is proposed to deepen the neural networks by using the same number of parameters and gradually computing the outputs in a channel-wise manner, which achieves the state-of-the-arts performances on large-scale image recognition. However, it tends to be slow and requires more graphic memory due to the data dependency. Recently, AdaNet [136] is developed to simulta-

neously and adaptively learn both the network structure as well as the node weights, which could be the first work to consider huge diversity of learning complexities for large numbers of object classes. It is able to adapt the network architecture according to the learning complexities for large numbers of object classes. They outperform the manually designed architectures while large number of the computational resources are used. In this work, we improve the algorithms from a different direction by embedding the complementary networks sequentially in an easy-to-hard way, where each complementary network used is more training efficient.

2.3 Proposed Method

For the sake of clarity, we provide the list of main mathematical symbols and the corresponding explanation in Tab. 2.1. As illustrated in Algorithm 1, our learning-difficulty-aware embedding algorithm contains the following key components: (a) Training the t^{th} (current) deep network $f_t(x)$ by focusing on achieving higher accuracy rates for some particular object classes (i.e., hard object classes which have larger error rates and are misclassified by the $(t-1)^{th}$ (previous) deep network $f_{t-1}(x)$), so that $f_t(x)$ can be complementary with $f_{t-1}(x)$; (b) Estimating the weighted error function for the t^{th} deep network $f_t(x)$ according to the normalized distribution of importances $[\varphi_t(C_1), ..., \varphi_t(C_N)]$ for N object classes; (c) Updating the distribution of importances $[\phi_{t+1}(C_1), \cdots, \phi_{t+1}(C_N)]$ for N object classes to push the $(t+1)^{th}$ deep network $f_{t+1}(x)$ at the next training round to achieve higher accuracy rates on recognizing the hard object classes which have higher error rates and are misclassified by the t^{th} (current) deep network $f_t(x)$, so that $f_{t+1}(x)$ can be complementary with $f_t(x)$; (d) Such iterative training process stops when the maximum number of iterations is reached or a certain level of accuracy rates is achieved. Our proposed embedding algorithm uses the deep CNNs (convolutional neural networks) as its weak learners, and all the well-designed deep networks(such as AlexNet [6], VGG [7], GoogleNet [8], ResNet [3] et al.) can be used to train the underlying weak learners (complementary

Table 2.1: The list of main mathematical symbols and the corresponding explanation for learning-difficulty-aware embedding.

Mathematical Symbols	**Explanation**
x_i	the feature of the i^{th} image sample
y_i	the class label of the i^{th} image sample
i	the index of image sample
l	the index of object classes
N	the number of object classes
C_l	the l^{th} class
S	the training set
R	the number of training image samples
R_l	the number of training samples in the l^{th} class
$f_t(x)$	the t^{th} deep network
$\phi_t(C_l)$	the un-normalized importance of the l^{th} class in the t^{th} iteration
$\varphi_t(C_l)$	the normalized importance of the l^{th} class in the t^{th} iteration
$p_t(C_l\|x)$	the probability for the image x to be assigned into the lth object class C_l
$h_t(x;\theta_t)$	heatmap features for image x extracted by the t^{th} network with parameters θ_t
w_{lt}	parameters of fully connected and softmax layers for the t^{th} network and the l^{th} class
$\epsilon_t(C_l)$	the error rate for the lth object class C_l
ε_t	the error rate for the t^{th} deep network $f_t(x)$
λ	hyper-parameter for determining hard object classes
β_t	an increasing function of ε_t
T	the number of complementary deep networks or iterations
$\mathbb{Z}$	a normalization factor
$\mathbb{F}(x)$	ensemble network
ϱ	the number of hard object classes
ρ	the ratio between the number of hard classes ϱ and the total number of classes N
$\mathcal{X}$	the instance space
Ω	the distribution over the instance space
ξ	margin between the confidences from ground-truth and incorrectly-predicted classes
d	VC-dimension
J	Jacobi matrix
z_{lt}	output of fully connected layers for the t^{th} network and the l^{th} class

deep networks).

2.3.1 Diverse Complementary Deep Networks

To learn the t^{th} (current) deep network $f_t(x)$, a deep CNNs is employed to obtain more discriminative representation $h_t(x;\theta_t)$ for the images, followed by a fully connected discriminant layer and a N-way softmax layer. The output of the t^{th} deep network $f_t(x)$ is a distribution of the prediction probabilities for N object classes, denoted

Algorithm 1: Difficulty-Aware Embedding of Complementary Deep Networks

Data: Training set for N object classes: $\left\{(x_i^l, y_i^l) \mid l \in \{1, ..., N\}\right\}$, where l is the class index;Initializing the distribution of importances over N object classes: $\phi_1(C_1) = ... = \phi_1(C_N) = \frac{1}{N}$;Number of complementary deep networks or iterations: T.

Result: The ensemble network: $\mathbb{F}(x) = \frac{1}{\mathbb{Z}} \sum_{t=1}^{T} \log\left(\frac{1}{\beta_t}\right) f_t(x)$

for $t = 1, \ldots, T$ **do**

Normalizing the distribution of importances over N object classes: $\varphi_t(C_l) = \frac{\phi_t(C_l)}{\sum_{j=1}^{N} \phi_t(C_j)}, l = 1, ..., N$;

Training the t^{th} deep network $f_t(x)$ according to the normalized distribution of importances over N object classes $[\varphi_t(C_1), ..., \varphi_t(C_N)]$;

Calculating the error rate $\varepsilon_t(C_l)$ for each object class;

Computing the weighted error rate for the t^{th} deep network $f_t(x)$: $\varepsilon_t = \sum_{l=1}^{N} \varphi_t(C_l)\varepsilon_t(C_l)$;

Setting the parameter $\beta_t = \frac{\lambda\varepsilon_t}{1-\lambda\varepsilon_t}$;

Updating the distribution of importances over N object classes $\phi_{t+1}(C_l)$ as: $\phi_{t+1}(C_l) = \phi_t(C_l)\beta_t^{1-\lambda\varepsilon_t(C_l)}$, $l = 1, ..., N$, so that the hard object classes, which have larger error rates and are misclassified by the t^{th} (current) deep network $f_t(x)$, can receive larger weights (importances) when we train the $(t+1)^{th}$ deep network $f_{t+1}(x)$ at the next round;

as $f_t(x) = [p_t(C_1|x), ..., p_t(C_N|x)]^\top$, where each component $p_t(C_l|x)$ is the probability score for the image x to be assigned into the lth object class C_l, $l = 1, ..., N$:

$$p_t(C_l|x) = \frac{exp\{w_{lt}^\top h_t(x; \theta_t)\}}{\sum_{j=1}^{N} exp\{w_{jt}^\top h_t(x; \theta_t)\}} \tag{2.1}$$

where θ_t and w_{lt} are the model parameters for the t^{th} deep network $f_t(x)$. The t^{th} deep network $f_t(x)$ can assign the image x into the object class $\hat{y}^t$ with maximum probability score:

$$\hat{y}^t = \arg\max_l p_t(C_l|x) \tag{2.2}$$

Suppose that the training set for N object classes is denoted as:

$$S = \{(x_i, y_i) \mid y_i \in \{C_1, ..., C_N\}, i = 1, ..., R\}$$

, where R is the number of training images. To train the t^{th} deep network $f_t(x)$, its model parameters can be obtained by maximizing the objective function in the form of weighted margin as:

$$\mathcal{O}_t(\theta_t, \{w_{lt}\}_{l=1}^N) = \sum_{l=1}^{N} \varphi_t(C_l)\xi_{lt}, \qquad \varphi_t(C_l) = \frac{\phi_t(C_l)}{\sum_{j=1}^{N} \phi_t(C_j)} \tag{2.3}$$

$$\xi_{lt} = \frac{1}{R_l} \sum_{i=1}^{R} \mathbf{1}(y_i = C_l) \log p_t(C_l|x_i) - \frac{1}{R - R_l} \sum_{i=1}^{R} \mathbf{1}(y_i \neq C_l) \log p_t(C_l|x_i) \tag{2.4}$$

where the indication function $\mathbf{1}(y_i = C_l)$ is equal to 1 iff $y_i = C_l$, R_l denotes the number of training images for the lth object class, and $\sum_{l=1}^{N} R_l = R$. $\varphi_t(C_l)$ is the normalized importance score for the lth object class C_l when training the t^{th} deep network $f_t(x)$, ξ_{lt} is used to measure the margin between the average confidences for the correctly-classified images and the misclassified images for the lth object class C_l by the t^{th} deep network $f_t(x)$. If the second item in Eq.(2.4) is small enough and negligible, it becomes:

$$\xi_{lt} \approx \frac{1}{R} \sum_{i=1}^{R} \mathbf{1}(y_i = C_l) \log p_t(C_l|x_i)$$

Then maximizing the objective function in Eq.(2.3) is equivalent to maximizing the weighted likelihood. By using the normalized distribution of importances $[\varphi_t(C_1), ..., \varphi_t(C_N)]$ for N object classes to approximate their learning complexities (e.g., their learning complexities are exponentially proportional to their error rates because hard object classes result in higher error rates), our proposed embedding algorithm can push the t^{th} (current) deep network $f_t(x)$ to focus on achieving higher accuracy rates for recognizing the hard object classes which are misclassified by the $(t-1)^{th}$ (previous) deep network $f_{t-1}(x)$, so that the t^{th} (current) deep network $f_t(x)$ can be complementary with the $(t-1)^{th}$ (previous) deep network $f_{t-1}(x)$.

For the t^{th} deep network $f_t(x)$, the error rate $\epsilon_t(C_l)$ for the lth object class C_l is

defined as:

$$\epsilon_t(C_l) = \frac{1}{2}\sum_{i=1}^{R}\left\{\mathbf{1}(y_i = C_l)\frac{1 - p_t(C_l|x_i)}{R_l} + \mathbf{1}(y_i \neq C_l)\frac{p_t(C_l|x_i)}{R - R_l}\right\} \tag{2.5}$$

The error rate in Eq.(2.5) is calculated in a soft decision way with probability; alternatively, we can also simply compute the error rate in a hard decision way as:

$$\epsilon_t(C_l) = \frac{1}{R}\sum_{i=1}^{R}\mathbf{1}(y_i = C_l \wedge \hat{y}_i^t \neq C_l) \tag{2.6}$$

The error rate for the t^{th} deep network $f_t(x)$ is defined as:

$$\varepsilon_t = \sum_{l=1}^{N}\varphi_t(C_l)\epsilon_t(C_l), \qquad \varphi_t(C_l) = \frac{\phi_t(C_l)}{\sum_{j=1}^{N}\phi_t(C_j)} \tag{2.7}$$

For the t^{th} deep network $f_t(x)$, the error rates are used as the criterion to determine its hard object classes:

$$\epsilon_t(C_l) > \frac{1}{2\lambda}, \qquad l \in \{1, \cdots, N\} \tag{2.8}$$

where the hyper-parameter λ is used to control the threshold of the expected error rate for determining the hard object classes for the t^{th} (current) deep network $f_t(x)$. We set the constraint as $\lambda > \frac{1}{2}$ (i.e., $\frac{1}{2\lambda} < 1$), e.g., larger λ corresponds to higher demand on the accuracy rate or lower threshold of the expected error rate for determining the hard object classes. We then compute the weighted error rate ε_t over all N object classes for the t^{th} (current) deep network $f_t(x)$ and update the distribution of importances ϕ_{t+1} to push the $(t+1)^{th}$ deep network $f_{t+1}(x)$ at the next training round to pay more attentions on the hard object classes which have larger error rates and are misclassified by the t^{th} deep network $f_t(x)$, so that the $(t+1)^{th}$ deep network $f_{t+1}(x)$ at the next training round can be complementary with the t^{th} (current) deep

network $f_t(x)$.

The distribution of importances is initialized equally for all N object classes:

$$\phi_1(C_l) = \frac{1}{N}, \qquad l = 1, \cdots, N$$

and it is updated iteratively along the learning process to emphasize the hard object classes with larger error rates:

$$\phi_{t+1}(C_l) = \phi_t(C_l)\beta_t^{1-\lambda\epsilon_t(C_l)} \tag{2.9}$$

where β_t is an increasing function of ε_t and its range is $0 < \beta_t < 1$, $\lambda\epsilon_t(C_l)$ is the product of λ and $\epsilon_t(C_l)$.

We set β_t as:

$$\beta_t = \frac{\lambda\varepsilon_t}{1 - \lambda\varepsilon_t} \tag{2.10}$$

The normalized distribution of importances for N object classes is defined as:

$$\varphi_{t+1}(C_l) = \frac{\phi_{t+1}(C_l)}{\sum_{j=1}^{N} \phi_{t+1}(C_j)}, \qquad l = 1, ..., N \tag{2.11}$$

By updating the importances for N object classes according to their error rates, our learning-difficulty-aware embedding algorithm can push the $(t+1)^{th}$ deep network $f_{t+1}(x)$ at the next training round to pay more attentions on the hard object classes the t^{th} (current) deep network $f_t(x)$ and achieve higher accuracy rates on recognizing such hard object classes.

To learn the t^{th} (current) deep network $f_t(x)$, our proposed embedding algorithm optimizes both the deep CNNs (for learning discriminative representations) and a weighted N-way softmax jointly by maximizing the objective function $\mathcal{O}_t$ in Eq.(2.3).

To maximize $\mathcal{O}_t$, it is necessary to calculate its gradients with respect to all the parameters, including the weights $\{w_{lt}\}_{l=1}^{N}$ and the set of network parameters θ_t.

For clearance, we denote

$$z_{lt}(x) = w_{lt}^{\top} h_t(x;\theta_t) \triangleq z_{lt}(h_t, w_{lt})$$

Thus the probability score in Eq.(2.1) for the image x to be assigned into the lth object class C_l can be written as:

$$p_t(C_l|x) = \frac{e^{z_{lt}(x)}}{\sum_{j=1}^{N} e^{z_{jt}(x)}} \triangleq p_t^l(z_{1t}, ..., z_{Nt})$$

Then the objective function in Eq.(2.3) can be denoted as:

$$\mathcal{O}_t(\theta_t, \{w_{lt}\}_{l=1}^{N}) \triangleq \mathcal{O}_t(p_t^1, ..., p_t^N)$$

From above descriptions, one can easily observe that the objective function is a composite function. $\mathcal{O}_t$ is a function of multiple variables $p_t^1, ..., p_t^l, ..., p_t^N$, whose each component p_t^l is a function of multiple variables $z_{1t}, ..., z_{Nt}$, and each z_{lt} is a function of multiple variables h_t, w_{lt}, furthermore, each h_t is a function of multiple variables θ_t. By using the chain rule, the gradients for the objective function $\mathcal{O}_t$ over $\{w_{lt}\}_{l=1}^{N}$ and θ_t are computed from the top layer to the bottom one in a backward pass.

$$\frac{\partial \mathcal{O}_t}{\partial w_{lt}} = \sum_{j=1}^{N} \frac{\partial \mathcal{O}_t}{\partial p_t^j} \frac{\partial p_t^j}{\partial z_{lt}} \frac{\partial z_{lt}}{\partial w_{lt}}, \ \frac{\partial \mathcal{O}_t}{\partial \theta_t} = \sum_{j,l=1}^{N} \frac{\partial \mathcal{O}_t}{\partial p_t^j} \frac{\partial p_t^j}{\partial z_{lt}} \frac{\partial z_{lt}}{\partial h_t} \frac{\partial h_t}{\partial \theta_t}$$

$$\frac{\partial \mathcal{O}_t}{\partial p_t^j} = \varphi_t(C_j) \left[\frac{1}{R} \sum_{i=1}^{R} \mathbf{1}(y_i^j = C_j) \frac{1}{p_t(C_j|x_i^j)} - \frac{1}{(N-1)R} \sum_{i=1}^{R} \mathbf{1}(y_i^j \neq C_j) \log p_t(C_j|x_i^j) \right]$$

$$\frac{\partial p_t^j}{\partial z_{lt}} = \begin{cases} p_t^l(1 - p_t^l) & \text{if } j = l \\ -p_t^l p_t^j & \text{if } j \neq l \end{cases}$$

$$\frac{\partial z_{lt}}{\partial w_{lt}} = h_t, \qquad \frac{\partial z_{lt}}{\partial h_t} = w_{lt}, \qquad \frac{\partial h_t}{\partial \theta_t} = J$$

where J is Jacobi matrix. The gradients are back-propagated [15] through the t^{th} (current) deep network $f_t(x)$ to fine-tune the weights $\{w_{lt}\}_{l=1}^{C}$ and the set of network parameters θ_t simultaneously.

2.3.2 Learning-Difficulty-Aware Embedding

After T iterations, we can obtain T complementary deep networks $\{f_1, \cdots, f_t, \cdots, f_T\}$, which are sequentially trained to achieve higher accuracy rates on recognizing different subsets of N object classes in an easy-to-hard way. For recognizing N object classes accurately, all these T complementary deep networks are embedded adaptively to generate more discriminative ensemble network $\mathbb{F}(x)$ [17-19]:

$$\mathbb{F}(x) = \frac{1}{\mathbb{Z}} \sum_{t=1}^{T} \log\left(\frac{1}{\beta_t}\right) f_t(x), \qquad \beta_t = \frac{\lambda \varepsilon_t}{1 - \lambda \varepsilon_t} \tag{2.12}$$

where $\mathbb{Z} = \sum_{t=1}^{T} \log\left(\frac{1}{\beta_t}\right)$ is a normalization factor.

For a given test sample x_{test}, it first goes through all these T complementary deep networks to obtain T deep representations $\{h_1, \cdots, h_T\}$ and then its probability score $p(C_l|x_{test})$ for being assigned into the lth object class C_l is calculated as:

$$p(C_l|x_{test}) = \frac{1}{\mathbb{Z}} \sum_{t=1}^{T} \log\left(\frac{1}{\beta_t}\right) \frac{exp\{w_{lt}^{\top} h_t(x_{test}; \theta_t)\}}{\sum_{j=1}^{N} exp\{w_{jt}^{\top} h_t(x_{test}; \theta_t)\}} \tag{2.13}$$

The given test sample x_{test} can finally be assigned into top-1 object class with the maximum probability score or top-k object classes with top-k scores. By training multiple complementary deep networks sequentially (i.e., they are trained to achieve higher accuracy rates on recognizing different subsets of N object classes in an easy-to-hard way) and embedding them adaptively, our proposed embedding algorithm can generate more discriminative ensemble network $\mathbb{F}(x)$ to achieve higher overall

accuracy rates on large-scale visual recognition, e.g., such ensemble network can effectively achieve high accuracy rates for the easy object classes (which can be obtained by the first deep network $f_1(x)$) while improving the accuracy rates for the hard ones at certain degrees (which are achieved by residual complementary deep networks $\{f_2(x), \cdots, f_t(x), \cdots, f_T(x)\}$ from the second iteration).

2.4 Parameter Selection for Learning-Difficulty-Aware Embedding

In our proposed embedding algorithm, the parameter β_t is used to: (a) update the distribution of importances ϕ for N object classes according to their error rates; (b) estimate the significance of the t^{th} complementary deep network $f_t(x)$ in the ensemble network $\mathbb{F}(x)$. On the other hand, the hyper-parameter λ is used to set the threshold of the expected error rate to determine the hard object classes for the t^{th} complementary deep network $f_t(x)$. Thus it is very important to define the criteria to select the optimal values for these two parameters in our proposed embedding algorithm.

2.4.1 Selecting Optimal β_t

Inspired by [38], we study the optimal parameter selection for deep embedding. In our proposed embedding algorithm, the parameter β_t is selected to be an increasing function of the error rate ε_t with the range $[0, 1]$. β_t is employed in two folds: (1) As defined in Eq.(2.9), β_t is used to update the distribution of importances $[\phi_{t+1}(C_1), \cdots, \phi_{t+1}(C_N)]$ for N object classes according to their error rates, so that the $(t+1)^{th}$ complementary deep network $f_{t+1}(x)$ at the next training round can pay more efforts to achieve higher accuracy rates on recognizing the hard object classes which are misclassified by the t^{th} complementary deep network $f_t(x)$ and have higher error rates; (2) As defined in Eq.(2.12) and Eq.(2.13), the reciprocal of β_t is used to determine the weight or importance of the t^{th} complementary deep network $f_t(x)$ in the ensemble network $\mathbb{F}(x)$.

The error rate is used as the criterion for the t^{th} deep network $f_t(x)$ to determine its hard object classes:

$$\epsilon_t(C_l) > \frac{1}{2\lambda}, \qquad l \in \{1, \cdots, N\}$$

which means that the error rates for the hard object classes are above a threshold $\frac{1}{2\lambda}$. For the lth object class C_l, by assessing it over T complementary deep networks $\{f_1(x), \cdots, f_t(x), \cdots, f_T(x)\}$, we can further define $\epsilon_{min}(C_l)$ as:

$$\epsilon_{min}(C_l) \triangleq \min_{t \in \{1, \cdots, T\}} \{\epsilon_t(C_l)\} \tag{2.14}$$

If $\epsilon_{min}(C_l) > \frac{1}{2\lambda}$, the lth object class C_l is always hard to be recognized by all these T complementary deep networks. The appearances of such always-hard object classes may seriously affect the overall accuracy rates of our learning-difficulty-aware embedding algorithm on large-scale visual recognition.

We use ϱ to denote the number of such always-hard object classes:

$$\varrho = \sum_{l=1}^{N} \mathbf{1}\left(\epsilon_{min}(C_l) > \frac{1}{2\lambda}\right) \tag{2.15}$$

where $\mathbf{1}(\epsilon_{min}(C_l) > \frac{1}{2\lambda}) = 1$ iff $\epsilon_{min}(C_l) > \frac{1}{2\lambda}$ is true. To achieve higher overall accuracy rates on large-scale visual recognition, we should select suitable value for the parameter β_t in Eq.(2.10) to guarantee that $\rho = \frac{\varrho}{N}$ is minimized (e.g., the number of such always-hard object classes ϱ is minimized).

For $0 < \alpha < 1$, we have $x^\alpha \leq 1 - (1 - x)\alpha$. According to Eq.(2.9), the importance for the lth object class C_l is updated according to its error rate $\epsilon_t(C_l)$:

$$\phi_{t+1}(C_l) = \phi_t(C_l)\beta_t^{1-\lambda\epsilon_t(C_l)}$$

Because $\sum_{l=1}^{N}\phi_{t+1}(C_l)=1$, we can get:

$$\begin{aligned}\sum_{l=1}^{N}\phi_{t+1}(C_l)&=\sum_{l=1}^{N}\phi_t(C_l)\beta_t^{1-\lambda\epsilon_t(C_l)}\leq\sum_{l=1}^{N}\phi_t(C_l)(1-(1-\beta_t)(1-\lambda\epsilon_t(C_l)))\\&=\sum_{l=1}^{N}\phi_t(C_l)(1-(1-\beta_t))+\lambda(1-\beta_t)\sum_{l=1}^{N}\phi_t(C_l)\epsilon_t(C_l)\end{aligned}\tag{2.16}$$

According to Eq.(2.7) and Eq.(2.11), we can get:

$$\sum_{l=1}^{N}\phi_t(C_l)\epsilon_t(C_l)=\left(\sum_{l=1}^{N}\phi_t(C_l)\right)\varepsilon_t$$

$$\begin{aligned}\sum_{l=1}^{N}\phi_{t+1}(C_l)&\leq\sum_{l=1}^{N}\phi_t(C_l)(1-(1-\beta_t))+\lambda(1-\beta_t)\left(\sum_{l=1}^{N}\phi_t(C_l)\right)\varepsilon_t\\&=\left(\sum_{l=1}^{N}\phi_t(C_l)\right)[1-(1-\beta_t)(1-\lambda\varepsilon_t)]\end{aligned}\tag{2.17}$$

Because $\sum_{l=1}^{N}\phi_1(C_l)=1$, we can have:

$$\sum_{l=1}^{N}\phi_2(C_l)\leq\left(\sum_{l=1}^{N}\phi_1(C_l)\right)[1-(1-\beta_1)(1-\lambda\varepsilon_1)]=1-(1-\beta_1)(1-\lambda\varepsilon_1)$$

$$\sum_{l=1}^{N}\phi_{T+1}(C_l)\leq\Pi_{t=1}^{T}[1-(1-\beta_t)(1-\lambda\varepsilon_t)]\tag{2.18}$$

By substituting Eq.(2.9) into Eq.(2.18), we can get:

$$\begin{aligned}\Pi_{l=1}^{T}[1-(1-\beta_t)(1-\lambda\varepsilon_t)]&\geq\sum_{l=1}^{N}\phi_{T+1}(C_l)=\sum_{l=1}^{N}\left(\phi_1(C_l)\Pi_{t=1}^{T}\beta_t^{1-\lambda\epsilon_t(C_l)}\right)\\&=\frac{1}{N}\sum_{l=1}^{N}\left(\Pi_{t=1}^{T}\beta_t^{1-\lambda\epsilon_t(C_l)}\right)\geq\frac{1}{N}\sum_{l:\epsilon_{min}(C_l)>\frac{1}{2\lambda}}\left(\Pi_{t=1}^{T}\beta_t^{1-\lambda\epsilon_t(C_l)}\right)\end{aligned}\tag{2.19}$$

Due to $\epsilon_{min}(C_l)>\frac{1}{2\lambda}$, it holds that $\epsilon_t(C_l)>\frac{1}{2\lambda}$ and $1-\lambda\epsilon_t(C_l)<\frac{1}{2}$ for all N object

classes. Recall the constraint $0 < \beta_t < 1$, we can have:

$$\frac{1}{N} \sum_{\epsilon_{min}(C_l) > \frac{1}{2\lambda}} \left(\Pi_{t=1}^{T} \beta_t^{1-\lambda\epsilon_t(C_l)} \right) \geq \frac{1}{N} \sum_{\epsilon_{min}(C_l) > \frac{1}{2\lambda}} \left(\Pi_{t=1}^{T} \beta_t^{\frac{1}{2}} \right) = \frac{\varrho}{N} \Pi_{t=1}^{T} \beta_t^{\frac{1}{2}} \tag{2.20}$$

Combining Eq.(2.19) with Eq.(2.20), we can get:

$$\rho = \frac{\varrho}{N} \leq \frac{\Pi_{t=1}^{T}[1 - (1-\beta_t)(1-\lambda\varepsilon_t)]}{\Pi_{t=1}^{T} \beta_t^{\frac{1}{2}}} = \Pi_{t=1}^{T} \frac{1 - (1-\beta_t)(1-\lambda\varepsilon_t)}{\beta_t^{\frac{1}{2}}} \tag{2.21}$$

To minimize the right-side, we set its partial derivative over β_t to be zero:

$$\frac{\partial}{\partial \beta_t} \left(\Pi_{t=1}^{T} \frac{1 - (1-\beta_t)(1-\lambda\varepsilon_t)}{\beta_t^{\frac{1}{2}}} \right) = 0$$

Because β_t only exists in the t^{th} factor, the above equation is equivalent to:

$$\frac{\partial}{\partial \beta_t} \left(\frac{1 - (1-\beta_t)(1-\lambda\varepsilon_t)}{\beta_t^{\frac{1}{2}}} \right) = 0$$

Solving this equation, β_t can be optimized as:

$$\beta_t = \frac{\lambda\varepsilon_t}{1 - \lambda\varepsilon_t}$$

which is used in Eq.(2.10).

We substitute $\beta_t = \frac{\lambda\varepsilon_t}{1-\lambda\varepsilon_t}$ into Eq.(2.21), we can obtain the upper boundary for the ratio ρ (between the number of always-hard object classes ϱ and the total number of object classes N being recognized) as:

$$\rho = \frac{\varrho}{N} \leq 2^T \Pi_{t=1}^{T} \sqrt{\lambda\varepsilon_t(1 - \lambda\varepsilon_t)} \tag{2.22}$$

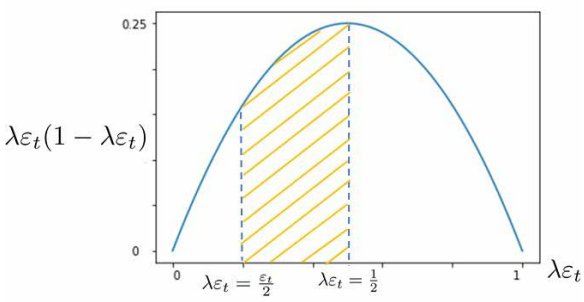


Figure 2.1: The relation between $\lambda\varepsilon_t$ and $\lambda\varepsilon_t(1-\lambda\varepsilon_t)$. The domain with respect to λ is $\frac{1}{2} < \lambda < \frac{1}{2\varepsilon_t}$, so $\frac{\varepsilon_t}{2} < \lambda\varepsilon_t < \frac{1}{2}$ (yellow shaded region).

2.4.2 Selecting Optimal λ

Now we discuss the range for the hyper-parameter λ. The error rate is used as the criterion for the t^{th} deep network $f_t(x)$ to determine its hard object classes:

$$\epsilon_t(C_l) > \frac{1}{2\lambda}, \qquad l \in \{1, \cdots, N\}$$

where λ is used to control the threshold of the expected error rate, e.g., when we have higher demand on the accuracy rate, a smaller threshold of the expected error rate should be used and λ should be larger, and we set the constraint for the hyper-parameter λ as $\lambda > \frac{1}{2}$. On the other hand, the range of $\beta_t = \frac{\lambda\varepsilon_t}{1-\lambda\varepsilon_t}$ is $0 < \beta_t < 1$, and it is required that $\lambda\varepsilon_t < \frac{1}{2}$, i.e., $\lambda < \frac{1}{2\varepsilon_t}$. As a result, λ should be selected between the interval $[\frac{1}{2}, \frac{1}{2\varepsilon_t}]$.

From the relationship between $\lambda\varepsilon_t$ and $\lambda\varepsilon_t(1-\lambda\varepsilon_t)$, as illustrated in Fig. 2.1, we can observe the effect of λ on the upper boundary of the ratio ρ in Eq.(2.22):

- (a) When $\lambda \in [\frac{1}{2}, \frac{1}{2\varepsilon_t}]$, i.e., $\frac{\varepsilon_t}{2} < \lambda\varepsilon_t < \frac{1}{2}$, the condition $0 < \beta_t < 1$ is satisfied, and the upper boundary for the ratio ρ in Eq.(2.22) increases when λ increases, the reason is that when λ increases, the threshold of the expected error rate for determining the hard object classes is smaller, thus the number of always-hard object classes may increase (i.e., the error rates for more object classes could be above such smaller threshold).

- (b) When $\lambda > \frac{1}{2\varepsilon_t}$, i.e., $\lambda\varepsilon_t > \frac{1}{2}$. In this case, the condition $0 < \beta_t = \frac{\lambda\varepsilon_t}{1-\lambda\varepsilon_t} < 1$ is not satisfied, thus updating the distribution of importances ϕ_{t+1} in Eq.(2.9) can not effectively push the complementary deep network $f_{t+1}(x)$ at the next training round to pay more attentions on achieving higher accuracy rates on recognizing the hard object classes. For such situation, large error rates ε_t tend to result in $\lambda\varepsilon_t$ being larger than or approaching $\frac{1}{2}$, and β_t being larger than or approaching 1. Thus the value of λ should be smaller to alleviate large ε_t such that the following constraints can still exist: $\lambda\varepsilon_t < \frac{1}{2}$ and $0 < \beta_t < 1$.

- (c) When $\lambda < \frac{1}{2}$, i.e., $\frac{1}{2\lambda} > 1$, such criterion can not be used for the t^{th} complementary deep network $f_t(x)$ to determine its hard object classes that their error rates satisfy $\epsilon_t(C_l) > \frac{1}{2\lambda}$.

2.5 Generalization Error Bound

Let $\mathcal{X}$ denote the instance space, Ω denote the distribution over the instance space $\mathcal{X}$, and $\mathcal{S}$ denote a training set $\{(x_i, y_i) \mid l \in y_i\{C_1, ..., C_N\}, i \in \{1, ..., R\}\}$ of N object classes. The training samples are chosen i.i.d according to the distribution over the instance space Ω. We are to investigate the gap between the generalization error on Ω and the empirical error on $\mathcal{S}$ for our ensemble network $\mathcal{F}$.

Suppose that the ensemble network $\mathcal{F}$ is the learning-difficulty-aware embedding of T complementary deep networks, and let $\mathcal{G} = \Big\{x \mapsto \sum_{f\in\mathcal{F}} a_f f(x) | a_f \geqslant 0, \sum_{f\in\mathcal{F}} a_f = 1\Big\}$. The combination coefficients $\mathcal{A} = [a_1, ..., a_f, ...]$ can be viewed as a distribution over $\mathcal{F}$. Define $\hat{\mathcal{G}} = \Big\{x \mapsto \frac{1}{\Gamma}\sum_{t=1}^{\Gamma} f_t(x) | f_t \in \mathcal{F}\Big\}$, and each $f_t \in \mathcal{F}$ may appear multiple times in the sum. For any $g \in \mathcal{G}$, there exists a distribution $\mathcal{A} = [a_1, ..., a_f, ...]$, so we can select the complementary deep networks from $\mathcal{F}$ for Γ times independently according to $\mathcal{A}$ and obtain $\hat{g} \in \hat{\mathcal{G}}$, denote $\hat{g} \sim \mathcal{A}$.

Note that g is a N-dimension vector, and each component of g is the confidence of object class, i.e., $g_y(x) = p(y|x)$, $y \in \{C_1, ..., C_N\}$. Based on Eq.(2.13), the label

of the object class for the test sample can be predicted by $\arg\max_y g_y(x) = p(y|x)$. The ensemble network g predicts wrong if $g_y(x) \leq \max_{\bar{y}\neq y} g_{\bar{y}}(x)$. The generalization error rate for the ensemble network can be measured by the probability $\mathbf{P}_\Omega[g_y(x) \leq \max_{\bar{y}\neq y} g_{\bar{y}}(x)]$.

According to probability theory, for any events B_1 and B_2, $\mathbf{P}(B_1) \leq \mathbf{P}(B_2) + \mathbf{P}(\bar{B}_2|B_1)$, we can have:

$$\begin{aligned}\mathbf{P}_\Omega\left[g_y(x) \leq \max_{\bar{y}\neq y} g_{\bar{y}}(x)\right] \leq \mathbf{P}_{\Omega,\hat{g}\sim\mathcal{A}}\left[\hat{g}_y(x) \leq \max_{\bar{y}\neq y} \hat{g}_{\bar{y}}(x) + \frac{\xi}{2}\right] + \\ \mathbf{P}_{\Omega,\hat{g}\sim\mathcal{A}}\left[\hat{g}_y(x) > \max_{\bar{y}\neq y} \hat{g}_{\bar{y}}(x) + \frac{\xi}{2} \mid g_y(x) \leq \max_{\bar{y}\neq y} g_{\bar{y}}(x)\right]\end{aligned} \tag{2.23}$$

where $\xi > 0$ measures the margin between the confidences from the ground-truth (labeled training samples) and the incorrectly-predicted object classes. Using Chernoff bound [51], the second term in the right side of Eq.(2.23) is bounded as:

$$\mathbf{P}_{\Omega,\hat{g}\sim\mathcal{A}}\left[\hat{g}_y(x) > \max_{\bar{y}\neq y} \hat{g}_{\bar{y}}(x) + \frac{\xi}{2} \mid g_y(x) \leq \max_{\bar{y}\neq y} g_{\bar{y}}(x)\right] \leq e^{-\Gamma\xi^2/8} \tag{2.24}$$

Assume that the space $\mathcal{F}$ for the ensemble network is with VC-dimension d, which can be approximately estimated by the number of neurons ν and the number of weights ω in the complementary deep network, i.e., $d = O(\nu\omega)$. Recall that $\mathcal{S}$ is a sample set with R examples from N object classes. Then the effective number of hypotheses for the ensemble network $\mathcal{F}$ over $\mathcal{S}$ is at most $\sum_{i=1}^{d}\binom{RN}{i} = (\frac{eNR}{d})^d$. Thus, the effective number of hypotheses over $\mathcal{S}$ for $\hat{\mathcal{G}} = \left\{x \mapsto \frac{1}{\Gamma}\sum_{t=1}^{\Gamma} f_t(x) | f_t \in \mathcal{F}\right\}$ is at most $(\frac{eNC}{d})^{\Gamma d}$.

Applying Devroye Lemma as in [52], it holds with the probability at least $1 - \delta_\Gamma$ that

$$\mathbf{P}_{\Omega,\hat{g}\sim\mathcal{A}}\left[\hat{g}_y(x) \leq \max_{\bar{y}\neq y} \hat{g}_{\bar{y}}(x) + \frac{\xi}{2}\right] \leq \mathbf{P}_{\mathcal{S},\hat{g}\sim\mathcal{A}}\left[\hat{g}_y(x) \leq \max_{\bar{y}\neq y} \hat{g}_{\bar{y}}(x) + \frac{\xi}{2}\right] + \Delta_\Gamma \tag{2.25}$$

where $\Delta_\Gamma = \sqrt{\frac{1}{2R}[\Gamma d \log \frac{eR^2N}{d} + \log \frac{4e^8(\Gamma+1)}{\delta_\Gamma}]}$.

Likewise, in probability theory for any events B_1 and B_2, $\mathbf{P}(B_1) \leq \mathbf{P}(B_2) + \mathbf{P}(B_1|\bar{B}_2)$, we can have:

$$\begin{aligned}\mathbf{P}_{\mathcal{S},\hat{g}\sim\mathcal{A}}\left[\hat{g}_y(x) \leq \max_{\bar{y}\neq y} \hat{g}_{\bar{y}}(x) + \frac{\xi}{2}\right] \leq \mathbf{P}_{\mathcal{S}}\left[g_y(x) \leq \max_{\bar{y}\neq y} g_{\bar{y}}(x) + \xi\right] + \\ \mathbf{P}_{\mathcal{S},\hat{g}\sim\mathcal{A}}\left[\hat{g}_y(x) \leq \max_{\bar{y}\neq y} \hat{g}_{\bar{y}}(x) + \frac{\xi}{2}|g_y(x) > \max_{\bar{y}\neq y} g_{\bar{y}}(x) + \xi\right]\end{aligned} \tag{2.26}$$

Because

$$\begin{aligned}&\mathbf{P}_{\mathcal{S},\hat{g}\sim\mathcal{A}}\left[\hat{g}_y(x) \leq \max_{\bar{y}\neq y} \hat{g}_{\bar{y}}(x) + \frac{\xi}{2}|g_y(x) > \max_{\bar{y}\neq y} g_{\bar{y}}(x) + \xi\right] \\ \leq \sum_{\bar{y}\neq y} &\mathbf{P}_{\mathcal{S},\hat{g}\sim\mathcal{A}}\left[\hat{g}_y(x) \leq \hat{g}_{\bar{y}}(x) + \frac{\xi}{2}|g_y(x) > g_{\bar{y}}(x) + \xi\right] \leq (N-1)e^{-\Gamma\xi^2/8}\end{aligned} \tag{2.27}$$

By combining Eq.(2.23−2.27) together, it can be derived that

$$\mathbf{P}_{\Omega}\left[g_y(x) \leq \max_{\bar{y}\neq y} g_{\bar{y}}(x)\right] \leq \mathbf{P}_{\mathcal{S}}\left[g_y(x) \leq \max_{\bar{y}\neq y} g_{\bar{y}}(x) + \xi\right] + Ne^{-\Gamma\xi^2/8} + \Delta_\Gamma \tag{2.28}$$

Thus large margin ξ over the training set corresponds to narrow gap between the generalization error on Ω and the empirical error on $\mathcal{S}$, which leads to the better upper bound of the generalization error for our ensemble network.

2.6 Experimental Results and Discussions

In this section, we report our evaluation results for our difficulty-aware embedding algorithm over three popular datasets: MNIST [89], CUB-200-2011 [137], CIFAR-100 [138], and ImageNet1K [139]. We have compared our difficulty-aware embedding algorithm with the state-of-the-art baseline methods and our comparison experiments focus on evaluating the following factors: (a) whether the number T of complementary deep networks being embedded to generate the ensemble network has significant impacts on improving the performance of our difficulty-aware embedding algorithm

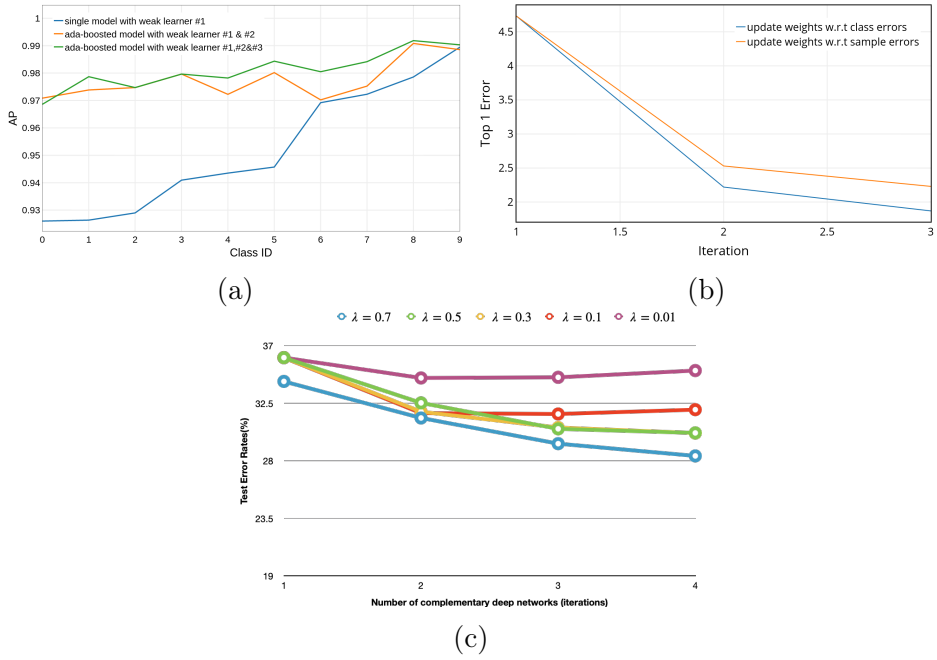


Figure 2.2: (a) The comparison on top 1 error for MNIST dataset when different embedding approaches are used; (b) The comparison on AP (average precision) of the ensemble network for MNIST dataset when different numbers of complementary networks are embedded, where the iteration ♯1 corresponds to using one single deep network (i.e., the first deep network); (c) The effects of the hyper-parameter λ on our difficulty-aware embedding algorithm.

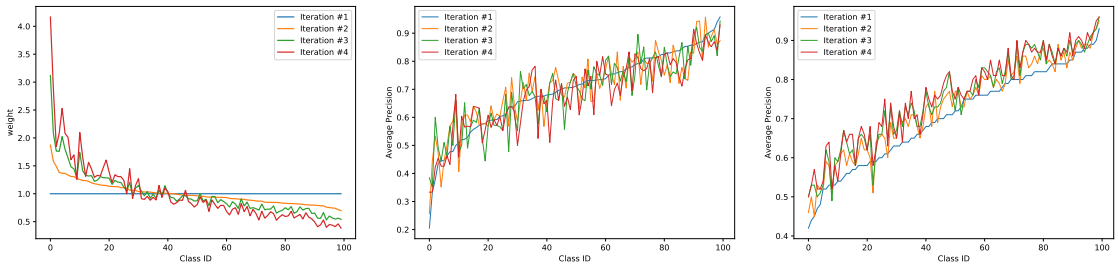


Figure 2.3: The comparison on CIFAR-100 and ResNet56 is used: (a) the distributions of class importances at different iterations; (b) the accuracy rates for the complementary networks at different iterations; (c) the accuracy rates of the ensemble networks when different numbers T of complementary networks are embedded, where the iteration ♯1 corresponds to using one single deep network (i.e., the first deep network).

(i.e., the accuracy rates for our ensemble network); (b) whether selecting different values for the hyper-parameter λ has significant impacts on improving the performance of our difficulty-aware embedding algorithm; and (c) whether our difficulty-aware embedding algorithm can achieve higher overall accuracy rates on large-scale visual recognition, e.g., whether it can improve the accuracy rates for the hard object classes in certain degrees while effectively maintaining high accuracy rates for the easy ones.

2.6.1 Experimental Results on MNIST

MNIST [89] dataset consists of 60,000 training handwritten digit samples and 10,000 test samples. The research in [39] has demonstrated the accuracy improvement on MNIST dataset by updating the sample weights according to their error rates. For fair comparison, we use two approaches to train multiple deep networks iteratively: (1) our difficulty-aware embedding algorithm updates the importances of the object classes according to their error rates; (2) traditional deep boosting approach updates the sample weights like AdaBoost. In our experiments, we train the deep network with the learning rate 0.01 through out the whole 120 epoches.

With our difficulty-aware embedding method, the top 1 error rate on the test dataset decreases from 4.73% to 1.87% when three or more than three complementary networks are embedded to generate the ensemble network (as shown in Fig. 2.2(a)). The top 1 error of our difficulty-aware embedding method (by weighting the errors at the class level) drops more quickly than the traditional deep boosting approach (by weighting the errors at the sample level). Our difficulty-aware embedding method can exploit the idea that different classes (categories) may have different learning complexities and their errors should be treated differently, e.g., even the errors from the hard classes may have similar values (strengths) with the errors from the easy ones, they may have different effects on optimizing the joint objective function for deep network learning. From Fig. 2.2(b), one can easily observe that our difficulty-aware embedding algorithm can significantly improve the accuracy rates for the hard

classes while effectively maintaining the high accuracy rates for the easy ones.

2.6.2 Experimental Results on CUB-200-2011

The CUB-200-2011 [137] dataset contains 11,788 images of 200 bird species, where 5,994 training images and 5,794 testing images are used. It is one of most widely used dataset for fine-grained classification. The dataset provides ground-truth annotations of bounding boxes and parts of birds. In our experiments, we only use the image level class label to train and evaluate our difficulty-aware method. In each iteration, we use the widely used bilinear model B-CNN [140] as the weak learner.

From the results in Tab. 2.2 , we can see our difficulty-aware embedding algorithm improves the accuracy from 84.0% to 86.9% after four iterations. It shows that our proposed method can train a series of networks in an easy-to-hard way and improves the performance in fine-grained classification.

2.6.3 Experimental Results on CIFAR-100

We also carry out our experiments on CIFAR-100 dataset [138]. CIFAR-100 dataset has 60,000 images for 100 object classes. There are 500 training images and 100 testing images for each class. In the training stage, we hold out 5,000 images for validation and use 45,000 images for training. When we train the deep networks on CIFAR-100, the initial learning rate is set to 0.1 and we train the deep networks for 300 epoches. The experimental results for comparing our difficulty-aware embedding algorithms with the state-of-the-art approaches are demonstrated in Tab. 2.2 when: (1) different types of deep CNNs (ResNet56 [3], DenseNet [9], AlexNet [6], SE-Net [10]) are used to train the underlying complementary networks in our difficulty-aware embedding algorithm; (2) different numbers of complementary networks are embedded to generate the ensemble networks, where $T = 1$ corresponds to the first iteration and only one single deep network is used (i.e., only the first deep network is used).

To train the first deep network, we treat all 100 object classes in CIFAR-100 with

Table 2.2: The comparisons on the top-1 and top-5 error rates in parentheses.

Datasets	Network	$T = 1$	$T = 3$	$T = 4$
MNIST [89]	MLP [39]	4.73%	1.87%	1.86%
CUB-200-2011 [137]	B-CNN [140]	16.0%	13.9%	13.1 %
CIFAR-100 [138]	ResNet56 [3]	29.53%	24.97%	24.15%
	DenseNet100 [9]	30.78%	28.95%	26.64%
	SE-ResNet-110 [10]	24.31%	22.27%	21.98%
ImageNet1K [139]	ResNet50 [3]	24.18%(7.49%)	22.96%(6.81%)	22.12%(6.79%)
	SE-ResNet-50 [10]	22.64%(6.24%)	20.86%(5.76%)	20.57%(5.58%)
	DenseNet121 [9]	25.88%(8.38%)	23.67%(7.25%)	22.32%(6.17%)
	AlexNet [6]	43.71%(21.24%)	40.83%(19.32%)	39.23%(17.78%)

equal importances as shown in Fig. 2.3(a) (straight line in blue color), and its accuracy rates for all 100 object classes are shown in Fig. 2.3(b) (in blue color), where 100 object classes are sorted in orders of their accuracy rates. One can easily observe that some easy object classes can achieve high accuracy rates by the first deep network but some hard ones may have very low accuracy rates. When the percentage of such hard object classes (with low accuracy rates) is relatively small, using one single deep network (like traditional deep learning approaches [6, 7, 8, 9, 10, 3]) can obtain good average accuracy rates in overall. By updating the importances of the object classes according to their error rates as shown in Fig. 2.3(a) (orange line), our difficulty-aware embedding algorithm can spend more efforts on the hard object classes and improve their accuracy rates dramatically as shown in Fig. 2.3(b) (in orange color).

By increasing the importances of hard object classes and pushing the next complementary network to pay more attentions on them, one can observe that the accuracy rates for such hard object classes may improve dramatically on the validation set (as shown in Fig. 2.3(b)), however, the accuracy improvement on the test set is not so significant as shown in Fig. 2.3(c) and Tab. 2.2 . The reasons for this phenomenon (i.e., disagreement between the validation (training) set and the test set [59]) are: (1) the hard object classes may have huge intra-class visual diversities, thus the test

Table 2.3: The comparisons on the top-1 average error rates.

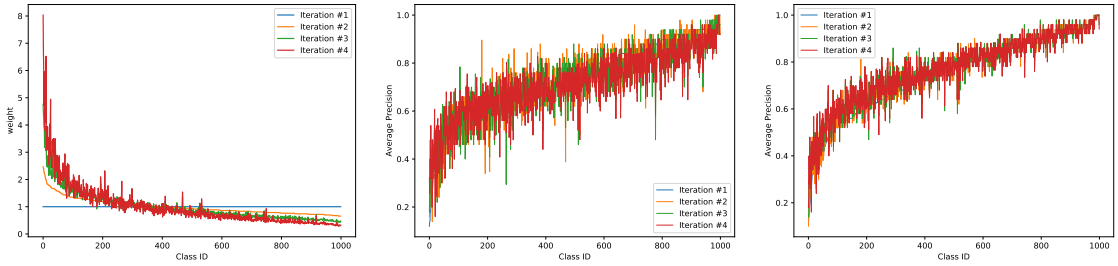
Datasets	Different Approaches	$\tau = 1$	$\tau = 3$	$\tau = 4$
CIFAR-100 [138]	difficulty-aware embedding	29.53%	24.97%	24.15%
	MixDCNN [76]	29.53%	29.35%	29.32%
	BoostedCNN [43, 39]	29.53%	26.82%	26.53%
ImageNet1K [139]	difficulty-aware embedding	24.18%	22.96%	22.12%
	MixDCNN [76]	24.18%	23.15%	23.08%
	BoostedCNN [43, 39]	24.18%	23.98%	23.75%

images may be significantly different from the training and validation images; (2) the hard object classes may have huge inter-class visual similarities with others, thus they are typically hard to be distinguished; (3) from the third iteration, the larger importances may stick on the always-hard object classes as shown in Fig. 2.3(a). As shown in Fig. 2.3(b), one can easily observe that the upper bound for the ratio of such always-hard object classes in CIFAR-100 dataset is close to $\rho \approx 36\%$. The effects of the hyper-parameter λ on the performances of our difficulty-aware embedding algorithm are shown in Fig. 2.2(c).

2.6.4 Experimental Results on ImageNet1K

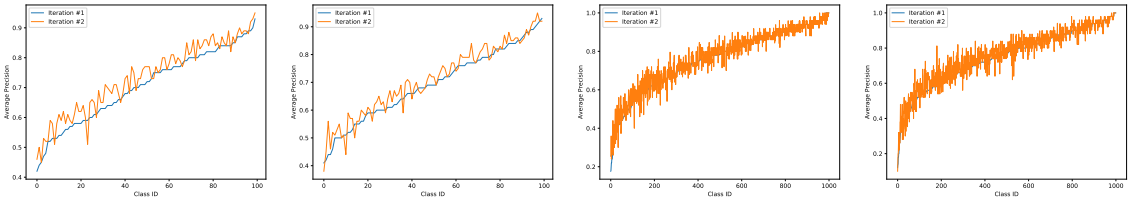
ImageNet1K dataset [139] consists 1,000 object classes, which have 1.2 million images for training, and 50,000 for validation. When we train the deep networks over ImageNet1K dataset, the initial learning rates are set to 0.1. The performances of our ensemble network are shown in Tab. 2.2 when: (1) different types of deep CNNs are used to train the underlying complementary networks in our difficulty-aware embedding algorithm; (2) different numbers of complementary networks are embedded to generate the ensemble network, where the iteration ♯1 corresponds to using one single deep network $\tau = 1$ (i.e., only the first deep network is used).

When we train the first deep network, we treat all 1,000 object classes with equal importances as shown in Fig. 2.4(a) (straight line in blue color), and its accuracy rates for all 1,000 object classes on the validation set are shown in Fig. 2.4(b), where



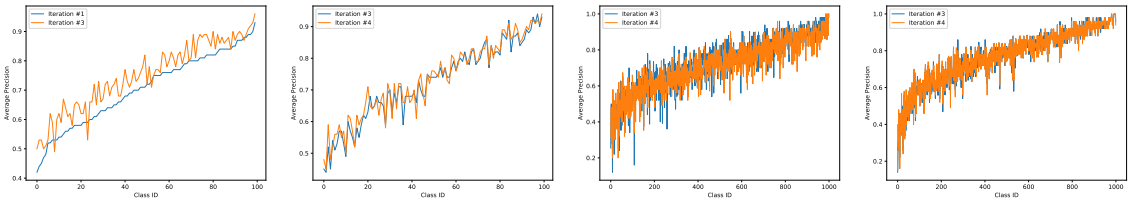
(a) (b) (c)

Figure 2.4: The comparison on ImageNet1K and DenseNet121 is used: (a) the distributions of class importances for different iterations; (b) the accuracy rates for the complementary networks for different iterations; (c) the accuracy rates of the ensemble networks when different numbers T of complementary networks are embedded, where the iteration ♯1 corresponds to using one single deep network $T = 1$ (i.e., only the first deep network is used).



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

Figure 2.5: The comparison on the average accuracy rates between one single joint network (iteration ♯1) and ensemble network by embedding two complementary networks (iteration ♯2), where different types of deep networks are used: (a) ResNet56 on CIFR-100; (b) DenseNet100 on CIFAR-100; (c) ResNet50 on ImageNet1K; (d) DenseNet121 on ImageNet1K.



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

Figure 2.6: The comparison on the average accuracy rates of ensemble networks when different numbers of complementary networks are embedded, where iteration ♯3 when $T = 3$ complementary networks are embedded and ♯4 when $T = 4$ complementary networks are embedded and different types of deep networks are used: (a) ResNet56 on CIFR-100; (b) DenseNet100 on CIFAR-100; (c) ResNet50 on ImageNet1K; (d) DenseNet121 on ImageNet1K.

Table 2.4: The lists of hard object classes for first 3 complementary deep networks in ImageNet1K dataset. The classes are sorted according to the validation errors.

Network No.	**Hard Object Classes**
No. 1 Network	maillot, screen, velvet, laptop, Appenzeller, tiger cat, water jug, sunglass, spotlight, spatula, monitor, hook, tape player, ladle, overskirt, letter opener, pole, cloak, sliding door, dough, chiffonier, sweatshirt, tub, cup, English foxhound, missile, backpack, cassette player, mushroom, bakery
No. 2 Network	hook, projectile, laptop, cuirass, Appenzeller, water jug, sunglass, maillot, tub, letter opener, spatula, spotlight, tiger cat, backpack, cloak, breastplate, ladle, dough, missile, sliding door, English foxhound, cradle, soup bowl, rock crab, stove, Windsor tie, sunglasses, CD player, minivan, drumstick
No. 3 Network	spotlight, projectile, spatula, tape player, Appenzeller, sunglass, dough, maillot, mouse, sunglasses, screen, sweatshirt, chiffonier, laptop, cuirass, rifle, stove, mushroom, ladle, tub, night snake, tiger cat, sliding door, CD player, moving van, sunscreen, letter opener, wok, bakery, carton

all 1,000 object classes are sorted in orders of their accuracy rates. One can easily observe that some easy object classes have achieved high accuracy rates by the first deep network but some hard ones may have very low accuracy rates. By updating the importances of the object classes according to their error rates as shown in Fig. 2.4 (a)

(orange line), the second complementary network can pay more attentions on the hard object classes and their accuracy rates can be improved dramatically on the validation set as shown in Fig. 2.4 (b). On the test set, as shown in Fig. 2.4 (c), our difficulty-aware embedding algorithm can effectively maintain high accuracy rates for the easy ones (that can be achieved by the first deep network) while improving the accuracy rates for the hard object classes at certain degrees (that are achieved by residual complementary networks from the second iteration). As shown in Fig. 2.4 (b), one can easily observe that the number of hard object classes ϱ in ImageNet1K dataset is larger, $\varrho \approx 388$, thus the upper bound for the ratio ρ is close to $\rho \approx 40\%$. As shown in Fig. 2.4 , we can also observe the disagreement on the accuracy improvement between the validation (training) set and the test set [59]. Besides three reasons which we have discussed above, another reason for this phenomenon in ImageNet1K dataset is that: Some hard object classes are fine-grained, thus they are hard to be distinguished because they belong to the same parent concept on the semantic ontology and they are visually similar. As illustrated in Tab. 2.2 , the top-1 accuracy rates could be low but the top-5 accuracy rates could be much better, e.g., such fine-grained object classes are hard to be distinguished from each other (which may result in low top-1 accuracy rates), but such confusion could be vanished gradually when top-5 accuracy rate is calculated.

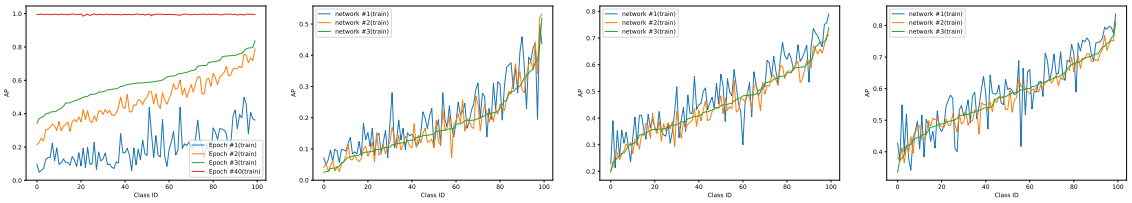
2.6.5 Comparison with Other Ensemble Approaches

For CIFAR-100 and ImageNet1K datasets, we have compared three ensemble approaches: (1) BoostedCNN (by weighting the errors at the sample level) [43, 39]; (2) MixDCNN (by weighting the errors dynamically during joint optimization) [76]; (3) our difficulty-aware embedding algorithm (by weighting the errors at the class level). In this comparison experiment, ResNet50 [3] is used to train the complementary networks being embedded. By embedding the same number of complementary networks to generate the ensemble networks, we have compared their performances. As shown

in Tab. 2.3 , one can easily observe that our difficulty-aware embedding algorithm can achieve lower overall error rates, where the first iteration ♯1 corresponds to using one single deep CNNs (i.e., only the first deep network is used) and $T = 1$ in Tab. 2.3 . The reason is that our difficulty-aware embedding algorithm can adapt the importances of the object classes to their error rates and train the complementary networks iteratively to improve the accuracy rates for the hard object classes at certain degrees (that are achieved by the residual complementary networks from the second iteration) while effectively maintaining the high accuracy rates for the easy ones (that can be achieved at the first deep network). From Tab. 2.3 , one can also observe that one single ResNet50 [3] has already achieved acceptable average accuracy rates. The reason is that ResNet is also an ensemble of relatively shallow networks as pointed out by Veit et al. [79]. Even using one single ResNet can obtain acceptable average accuracy rates in overall, its accuracy rates for the hard object classes are still very low; on the other hand, our difficulty-aware embedding algorithm can improve the accuracy rates for the hard object classes at certain degrees.

2.6.6 Single Deep Network versus Ensemble One

In our difficulty-aware embedding approach, different types of deep CNNs can be used to train the underlying complementary networks. As shown in Fig. 2.5 , one can easily observe that: when different types of deep CNNs are used, our difficulty-aware embedding algorithm (using ensemble network) can outperform the traditional deep learning approaches (using one single deep CNNs) and the same conclusion can also be observed from Tab. 2.2 and Tab. 2.3 , where the first iteration ♯1 corresponds to using one single deep CNNs (in blue color) and ♯2 corresponds to embedding the first deep CNNs and the second complementary one to generate the ensemble network (in orange color). As shown in Fig. 2.6 , one can also observe that: when more complementary networks are embedded, our difficulty-aware embedding algorithm can generate more discriminative ensemble networks, where the third iteration



(a) network#1 (b) epoch#1 (c) epoch#2 (d) epoch#3

Figure 2.7: Illustration for training convergence process of complementary networks dynamically/jointly optimized on CIFAR100: the convergence process of one complementary network(a), the average precision among different networks for the first three epochs(b,c,d). It shows that the difficult categories (small class ID) are similar even with the randomization from the network initialization and optimization process.

♯3 corresponds to embedding 3 complementary networks to generate the ensemble network (in blue color) and ♯4 corresponds to embedding 4 complementary networks to generate the ensemble network (in orange color).

Obviously, when T complementary networks are embedded to generate the ensemble network, the ratio of the network parameters between our difficulty-aware embedding algorithm and traditional deep learning approach (using one single deep CNNs) is T, which could be cost-sensitive. One potential solution for this issue is to treat the ensemble network (obtained by our difficulty-aware embedding algorithm) with higher accuracy rates as a teacher model and train a simple student network via knowledge distillation [141]. By penalizing the mismatches between the predictions from the teacher model (our ensemble network) and the student network to minimize their knowledge diversity, such student network can use one single network with much smaller number of parameters to achieve close accuracy rates effectively as our ensemble network does.

2.6.7 Further Discussion

2.6.7.1 Weighting at Category Level

AdaBoost [38](weighting at the sample level) has achieved higher training accuracy by combining traditional weak learning models (i.e. small networks in [43, 39]), but

it is unsuitable for combining large networks: (a) In deep learning, the training error could approach zero and easily suffers from overfitting [59]; (b) When the training error rates are close to zeros, focusing on the hard samples could make the problem of overfitting even worse because the deep networks will be trained with a small portion of samples. For example, in our experiments on CIFAR100 dataset, the training errors could easily approach zeros, which is in line with [59], thus we use the validation error rates to weight the objective function in the subsequent iterations. At this scenario, weighting at the sample level is not practical to use the validation results and would only leave a small portion of the samples focused and may worsen the overfitting.

Overall, weighting at the sample level may impair the generation ability of strong complementary learning models (ResNet56/ResNet50 in Tab. 2.3). Weighting at the category level provides an effective alternative which acts as a regularization method and guides the optimization process (SGD) to pay more attention on difficult categories by optimizing the weighted objective function subsequently (Fig. 2.3 2.4).

2.6.7.2 Sequentially Guided Optimization

Combining multiple complementary deep networks dynamically/jointly [76, 128] is indeed a reasonable alternative for embedding and it has shown improved results on fine-grained tasks [76] and small neural networks [128], but the improved margin is small with dynamic weighting on non-fine-grained task (CIFAR-100/ImageNet) with deep neural networks (ResNet56/ResNet50) in our experiments (Tab.2.3). The diversity of dynamic weighting is based on randomization from network initialization and optimization process (SGD) while our proposed method is based on learning difficulty of categories. For difficult categories which are hard to learn and converge slowly for all networks, i.e. small ID categories in Fig.2.7a for all complementary networks (Fig.2.7b,2.7c,2.7d), the weights/occupations could be distributed almost equally for the complementary networks if optimized jointly and behaves like average ensemble. However, such hard categories would be focused and learned by guided objective

function in the subsequent networks with the proposed method (Fig. 2.3 2.4). In addition, we have proven the theoretical convergence of our proposed method (Sec. 2.4.1&Eq. 2.22).

2.6.7.3 Generalization Error Bound

As stated before, the training errors of single deep neural network could even approach zeros, which has been investigated in [59]. However, our proposed difficulty-aware embedding could further improve the test accuracy through multiple iterations (Tab. 2.2) and a good selection of λ. It is in line with the theoretical analysis in generalization error bound (Sec. 2.5&Sec. 2.4.2).

In Eq. 2.28, the divergence between training errors $\mathbf{P}_{\Omega}\left[g_y(x) \leq \max_{\bar{y}\neq y} g_{\bar{y}}(x)\right]$ and validation errors $\mathbf{P}_{\mathcal{S}}\left[g_y(x) \leq \max_{\bar{y}\neq y} g_{\bar{y}}(x) + \xi\right]$, is narrowing along the iterations Γ. Thus, though training errors could approach zeros even in the first iteration, the validation errors could still decrease with our difficulty-aware embedding.

For theoretical deduction in Sec. 2.4.2, we introduce a new parameter λ to maintain the convergence of the algorithms and provide the theoretical discussion about the range and meaning of the new parameter λ. The value of λ should be set small with bad single model (relatively weak learner) learning result to alleviate large ε_t to maintain the theoretical convergence. In experiment, we found the value of ε_t should not be set too small because it could make the weights more discriminative and cause the accuracy drop for some easy classes (Fig.2.2(c) 2.3(a) 2.4(a)).

2.7 Conclusions

In this chapter, a diverse representation learning method, namely difficulty-aware embedding, is developed to train multiple diverse complementary deep networks sequentially in an easy-to-hard way. During the iterative training process, the distribution of importances for different object classes is updated such that the current trained network spends more efforts on distinguishing the hard object classes which

are not classified well by the previous trained network. Each individual network focus on a subset of object classes for achieving higher accuracy rates and all these deep networks compensate by learning to recognize different subsets of object classes. These complementary deep networks with diverse capabilities are seamlessly combined to generate more discriminative ensemble classifier. As for the future network, we would like to investigate the adaptive network structures in the difficulty-aware embedding.

CHAPTER 3: ASPECT RATIO REPRESENTATION LEARNING FOR IMAGE AESTHETICS ASSESSMENT

3.1 Problems and Motivation

3.1.1 Aspect Ratio in Image Aesthetic Assessment

Image aesthetics assessment, where the goal is to predict the given image an aesthetic score, has many applications such as album photo recommendation, auxiliary photo editing, and multi-shot photo selection. The task is challenging because it entails computations of both global cues (*e.g.* scene, exposure control, color combination, etc) and localization information (composition, photographic angle, etc).

Early approaches extract aesthetic features according to photographic rules (lighting, contrast) and global image composition (symmetry, rule of thirds), which require extensive manual designs [142, 143, 144, 145, 146, 147]. However, manual design for such aesthetic features is not a trivial task even for experienced photographers. Recent work adopts deep convolutional neural networks for image aesthetics assessment by learning models in an end-to-end fashion. The models mainly use three types of formulations: binary classification labels [148, 82, 81, 149, 150, 151, 84], scores [152, 2, 83], and rankings [153, 154].

In the aforementioned methods, the backbone networks are usually adopted from an image classification network. The data augmentation methods, *i.e.* image cropping and warping, are widely used for preventing overfitting in the image recognition task. However, a shortcoming is that the compositions and object aspect ratios are altered, which may introduce label noise and harm the task of aesthetics assessment (Fig. 3.1). A succinct solution proposed in MNA-CNN [81] is to feed one original-size image

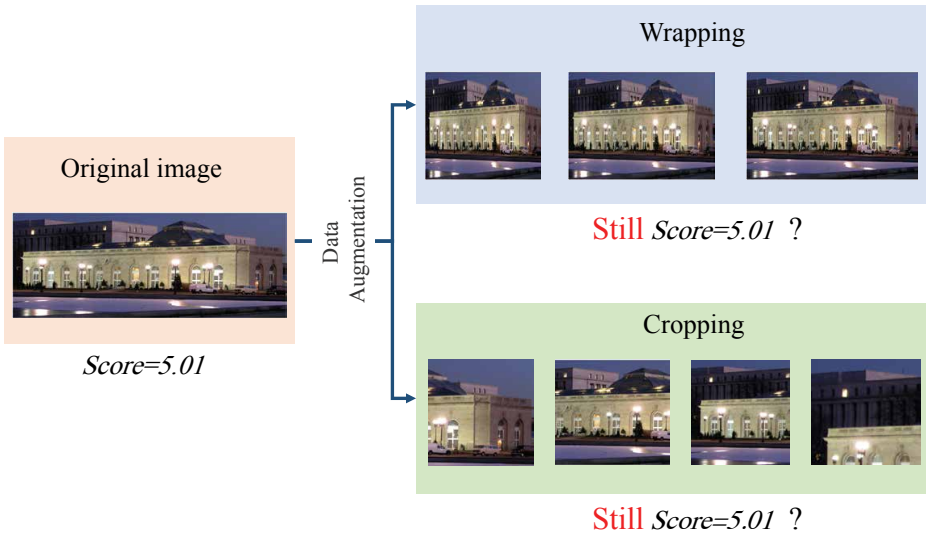


Figure 3.1: Image warping and cropping are widely used for data augmentation, but they alter the object aspect ratios and composition, causing different aesthetics perceptions. Assigning the groundtruth aesthetic score of the original image to the altered image may introduce label noise and deteriorate the discriminative ability.

into the network at a time during training and test (bottom stream in Fig. 3.2). A major constraint of the approach is that images with different aspect ratios cannot be concatenated into batches because the aspect ratio of each image should be preserved. Thus it slows down the training and inference.

3.1.2 Learning Aesthetics Representations About Aspect Ratios

In this chapter, we aim to develop a novel adaptive fractional dilated convolution that is mini-batch compatible. As shown in the top row in Fig. 3.2, our network adaptively dilates the convolution kernels to the composition-preserving warped images according to the image aspect ratios such that the effective receipt field of each dilated convolution kernel is the same as the regular one. Thus we may assign the ground truth aesthetic score of the original image to the wrapped image without introducing label noise. Specifically, as illustrated in Fig. 3.4, the fractional dilated convolution kernel is adaptively interpolated by the nearest two integer dilated kernels with the same kernel parameters. Thus no extra learning parameters are introduced.

The **benefits** of our method can be summarized as follows: (a) By embedding

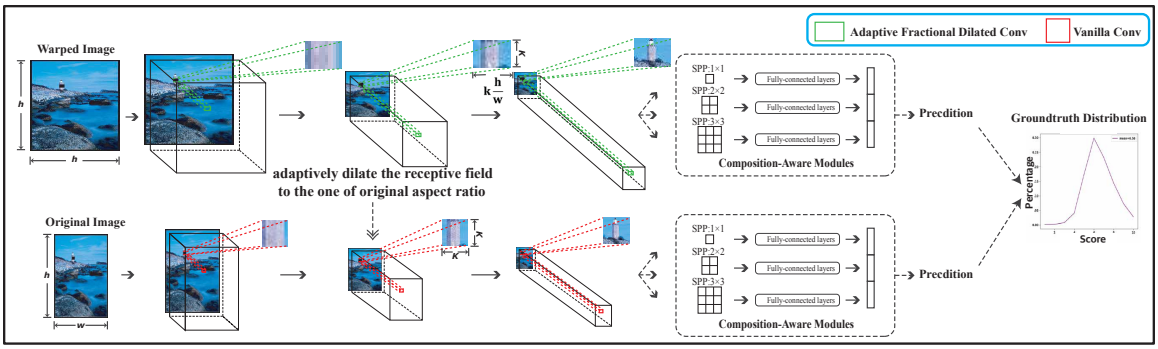


Figure 3.2: Overview of adaptive fractional dilated CNN (above) and the comparison with vanilla CNN (below): Each fractional dilated Conv (above) operated on wrapped input adaptively dilates the same receptive field as the vanilla Conv (below) operated on the original image. It thus helps with the problems: (a) Becomes mini-batch compatible by composition-preserving warping instead of feeding original-size image (b) Preserves aesthetic features related to aspect ratios by adaptive kernel dilation.

the information of aspect ratios to construct the convolution layers adaptively, it can explicitly relate the aesthetic perception to the image aspect ratios while preserving the composition; (b) It is parameter-free and thus can be easily plugged into the popular network architectures; (c) Through the deduction, we show that our proposed method can be mini-batch compatible and easily implemented by common deep learning libraries (*e.g.* PyTorch, Tensorflow); (d) A grouping strategy is introduced to reduce the computational overhead for efficient training/inference; (e) We achieve state-of-the-art performance for image aesthetics assessment on the AVA dataset [1].

3.2 Related Work

In this section, we provide a brief review of some of the most relevant works on: (a) image aesthetics assessment; (b) preserving image aspect ratios and compositions; (c) dilated convolution; (d) dynamic kernels.

Image Aesthetics Assessment The existing methods on image aesthetics assessment can be mainly categorized into three formulations:

(1) *Binary (or mean) aesthetic label*: Murray *et al.* [1] introduced the AVA dataset which was widely used as the training and testing dataset for state-of-the-art bench-

marks. Several methods proposed using binary aesthetic labels (high or low) or mean score labels. Kao *et al*. [148] propose a hierarchical convolutional neural network designed adaptively for three subclasses: scene, object and texture. They also proposed a multi-task CNN, A&C CNN, which joint learns category classification and aesthetic classification. Overall, the A&C CNN outperforms the hierarchical structure due to more training data and less weights in the unified network as opposed to the SVM classifier in the hierarchical structure. Mai *et al*. [81] address the composition in photo aesthetics assessment and propose a method that aggregates multiple sub-networks with different adaptive pooling layer sizes of $12 * 12$, $9 * 9$, $6 * 6$, $4 * 4$ and $2 * 2$ and further incorporates a scene category probability prediction to output final aesthetic binary label. Ma *et al*. [82] select multiple patches from the saliency map of the original image as local feature inputs to feed into VGG16 [7] and an aggregation layer. They further encode the patch localization information in a layer-aware subnet and combine it with a local feature aggregation output to get the final prediction.

(2) *Ranking score*: Some methods propose using ranking among images instead of formulating image aesthetics as an overall binary classification or regression problem. Kong *et al*. [153] trained a joint Euclidean and ranking loss and appended attributes and content category classification layers to perform joint optimization. Schwarz *et al*. [154] introduced triplet loss to reduce the score prediction distance between neighbor image pairs and increase the margin between distant image pairs. They also add ranking loss to lead triplet loss by reducing the norms of encodings belonging to less visual pleasing images which increases the norms of well crafted images.

(3) *Score distribution*: Since the aesthetic assessment is somewhat subjective and binary aesthetic label is sensitive near the boundary, there are some methods proposed to leverage score distribution information. To address the ordered score distribution, Hossein Talebi and Peyman Milanfar [2] introduce Earth Mover's Distance as a loss function to train 10-scale score distribution. Since aesthetics is a subjective property

and outlier opinions are likely, Naila Murray and Albert Gordo [152] introduce the Huber Loss to train 10-scale score distribution. In addition to using the mean score label of multiple raters, Ren *et al*. [150] proposed a sub-network to learn a personal rating offset along the generic aesthetic network and output the personalized score prediction.

Preserving Image Aspect Ratios and Compositions Multi-patch sampling over the original images is used to preserve the aspect ratios and proves to be effective [82, 84, 83]. The widely used data augmentation method in training classification networks, scaling and cropping, modifies the composition of original image. Several methods have been proposed to address the issue but they also have their limitations. He *et al*. [155] proposed the SPP(spatial pyramid pooling) to handle arbitrarily sized input. In the training stage, they feed multiple cropped sized images in order to increase scale-invariance and reduce overfitting. They originally used that training technique in classification and object detection task;however, in order to preserve the complete information of the image, *e.g*. photography viewing angle, some distraction objects, and partial overexposure, the random cropped is not applicable in image aesthetic assessment. Kao *et al*. [148] compare padding with wrapping into fixed sized input and found that wrapping slightly outperforms padding; however, through our experiments we found that it suffers from overfitting due to the absence of data augmentation. In addition to the overfitting, padding and wrapping modifies the original aesthetic information by importing the border and object distortion respectively. Mai *et al*. [81] feed the original sized image into the SPP [155]. We note that it is not suitable for batch training, which requires same size aspect ratio and slows down the training if the mini-batch size is set to 1. Ma *et al*. [82] proposed multiple fixed-size patches selected from the saliency map. The patches sampled from the original image changes the some information related to the area, *e.g*. color histogram and the ratio of salient object and background, which impacts the photo aesthetics. In addition,

the photos that contain partial bad factors, such as distraction objects and partial overexposure, should not be filtered out in the pre-selection phase. The method is convoluted and contains an extra attention model to generate a saliency map, uses complicated path sampling strategy, requires manually designed aggregation structure, and slow training and inference speed(50 patches per image).

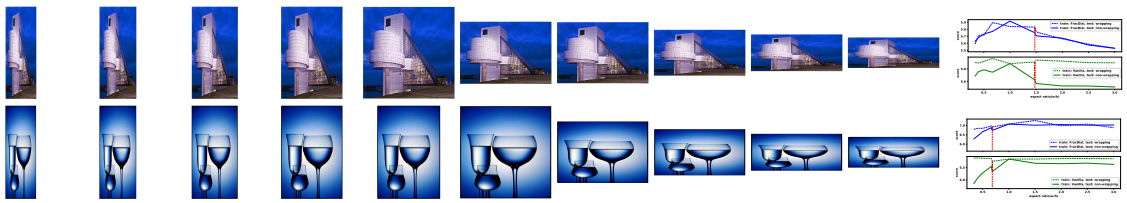
A major concern is that sampling patches from the original image may alter essential aesthetic factors (color histogram, object-background ratio) of the original image and the complete aesthetics features are lost. In contrast, our proposed method adaptively restores the original receptive fields from the composition-preserving warping images in an end-to-end fashion. The approach of MNA-CNN [81] is the most related to ours, as they proposed to preserve image aspect ratios and compositions by feeding the original image into the network, one at a time. A major constraint of the approach is that images with different aspect ratios cannot be concatenated into batches because the aspect ratio of each image should be preserved. Thus it tends to slow down the training and inference processes. On the other hand, our proposed method is mini-batch compatible and can be easily implemented by common deep learning libraries.

Dilated Convolution Our adaptive fractional dilated convolution is motivated by the dilated convolution [23] and atrous convolution [24] in semantic segmentation, but it differs from them in several aspects: (a) The motivation of our adaptive fractional dilated convolution is to dilate the receptive fields to the same one for the normal convolution kernels that are operated on the images with original aspect ratios; (b) The dilation rate can be fractional for our proposed method and be consistent with the fractional aspect ratios of the original images. To achieve this objective, we interpolate two nearest kernels to approximate the misalignment of fractional sampling; (c) The construction of fractional dilated kernel is dynamic and adaptive to the image aspect ratios.

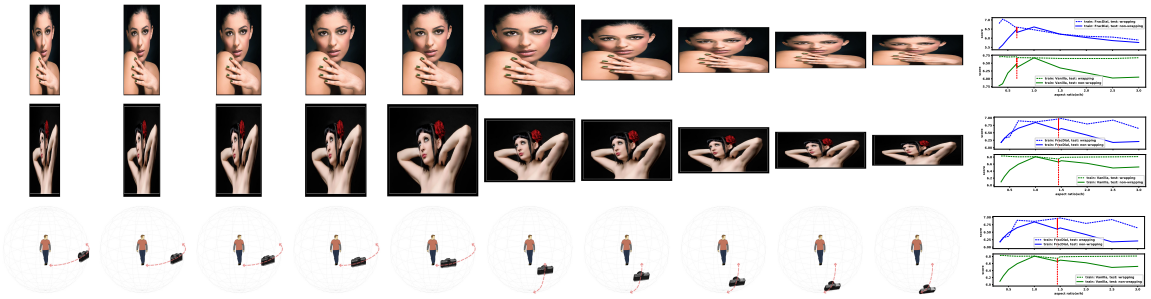
Dynamic Kernel Deformable convolution [156] is proposed to construct the receptive fields dynamically and adaptively by learning better sampling in the convolutional layer. Our proposed method differs from deformable convolution in the following points: *1) Motivation:* The deformable convolution is proposed to learn better sampling in the convolutional layer, on the other hand, our proposed method adapts the receptive fields in the convolution layers into the original aspect ratios. *2) Supervision:* Our proposed method, which is supervised by the strong prior that the aspect ratio is an important factor for image aesthetics assessment, is parameter-free while the deformable convolution requires extra layers to produce the sampling index map. *3) Adaptation:* Our method is adaptive in kernel level for each image while the deformable convolution is adaptive in the feature point level for each sliding window convolution. *4) Implementation:* Our proposed method provides a concise formula for the mini-batch training and could be easily implemented by the common deep learning frameworks. On the other hand, the deformable convolution requires to rewrite the convolution operation in CUDA and tends to be slow. Our proposed method actually interpolates the convolutional layers instead of the feature maps (the sampling operation in the deformable convolution).

3.3 Proposed Method

In this section, we first explain the impacts of aspect ratios for image aesthetics assessment. Then, we introduce the adaptive kernel interpolation to tackle the misalignment due to fractional sampling in the proposed method. In addition, we derive a concise formulation for it in the setting of mini-batch and discuss their computational overhead. Finally, we describe the loss function and an additional composition-aware structure for the composition-preserving warping batch.



(a) Manually designed objects: different wrapping patch could reflect the object designed with different ratio.



(b) Creatures: different wrapping patch could have similar effect of different photography angle.

Figure 3.3: Illustration for aesthetics change related to aspect ratios and the comparison of discrimination ability. Different aspect ratio has different aesthetics perception and thus an accurate model should be discriminative to the change of aspect ratios.

3.3.1 Image Aspect Ratios

Warping imports distortion and modifies the aspect ratios of original images and their objects. The distorted random-size image cropping for data augmentation, which is introduced by Christian *et al*. [8] and widely used by [31, 9, 10], proves to be effective for the task of image classification. Image cropping has minor effects for identifying the salient objects in the images(Fig. 3.1), and it modifies the critical aesthetic information. As shown in Fig. 3.3, the distorted images modify the aspect ratios of objects and bring different aesthetic perception. Specifically, for the manually-designed objects, *e.g*. architectures and cups(Fig. 3.3a), different warping patches could reflect the objects that are designed with different ratios, thus image warping should result in different aesthetic perception.

For the creatures which have well-formed aspect ratios, *e.g*. persons Fig. 3.3b and animals, different warping patches could have similar effects of different photography

angles. We admit that the real object, which is captured from different photography angles, cannot be transformed by using a simple 2D interpolation, *i.e.* the change of image's aspect ratio, but it could have similar perception effects from the sampled images. The photography angle is an important factor for image aesthetics perception, which is highly related to the changes of image aspect ratios. Overall, a model, which can learn such aesthetic information completely, should be discriminative to the changes of aspect ratios.

3.3.2 Adaptive Kernel Interpolation

As stated in Section 3.1, cropping modifies the composition of the original image and causes the loss of some critical aesthetics information. As a result, image cropping introduces somewhat label noises in the training stage. To preserve the composition, we firstly warp the image into a fixed size. For network training, such a simple image warping approach suffers from the problem of overfitting due to the absence of data augmentation. Motivated by SPP [157], we adopt random-size warping during the training stage and feed the mini-batch into the networks with global pooling or SPP modules, which can naturally handle arbitrary-size batch inputs. Overall, the random-size warping provides effective data augmentation for training scale-invariant networks while preserving the image compositions.

To cope with the distortion induced by warping, the receptive field of the convolution kernel should be consistent with the receptive field of the convolution kernel that is operated on the image with original aspect ratio. Our proposed approach tackles the distortion issue by adaptively dilating the kernels to the original aspect ratio, as illustrated in Fig. 3.2. Since the aspect ratio could be fractional, the dilation rate could be a fraction as well. To tackle the misalignment of feature sampling, we use the linear interpolation of two nearest integer dilation rates to construct the fractional dilation kernel.

Suppose that w and h represent the width and height of original images, respec-

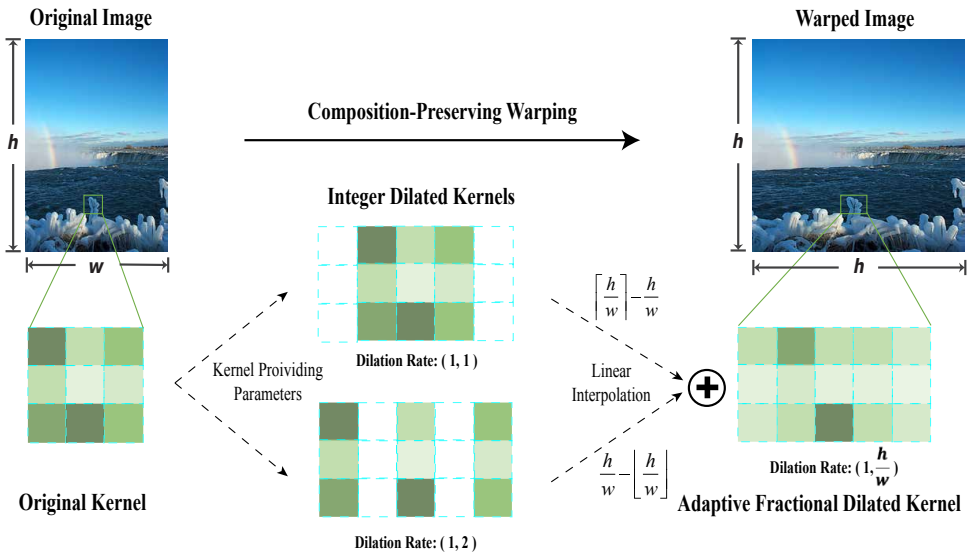


Figure 3.4: Illustration of kernel interpolation: linear interpolation of the nearest two integer dilated kernels shared same kernel parameters are used to tackle the sampling misalignment from fractional dilation rates.

tively. If $h > w$ and $\frac{h}{w}$ is not a integer, as illustrated in Fig. 3.4, AFDC (adaptive fractional dilated convolution) kernel k^n_{AFDC} in n-th layer is constructed as:

$$k^n_{AFDC} = (\lceil r \rceil - r) k^n_{(1,\lfloor r \rfloor)} + (r - \lfloor r \rfloor) k^n_{(1,\lceil r \rceil)} \tag{3.1}$$

where $r = \frac{h}{w}$. For any non-integer r, it is in the interval $\big[\lfloor r \rfloor, \lceil r \rceil\big]$ whose length is equal to 1. $\lfloor r \rfloor$ and $\lceil r \rceil$ are two integers nearest to r. $k^n_{(1,\lfloor r \rfloor)}$ and $k^n_{(1,\lceil r \rceil)}$ are two dilated kernels with the nearest integer dilation rates $\lfloor r \rfloor$ and $\lceil r \rceil$ for nth layer, respectively. More specifically, as shown in Fig. 3.4, $r \in [1, 2]$, $\lfloor r \rfloor = 1$, $\lceil r \rceil = 2$. We note that both $k^n_{(1,1)}$ and $k^n_{(1,2)}$ inherit the same learning parameters from the original kernel.

Likewise, if $w > h$ and $\frac{w}{h}$ is not an integer, then we choose:

$$k^n_{AFDC} = (\lceil r \rceil - r) k^n_{(\lfloor r \rfloor,1)} + (r - \lfloor r \rfloor) k^n_{(\lceil r \rceil,1)} \tag{3.2}$$

If $r = \frac{h}{w}$ is an integer, it is enough for us to employ integer dilated kernel.

Therefore, the fractional dilated kernel is adaptively constructed for each image

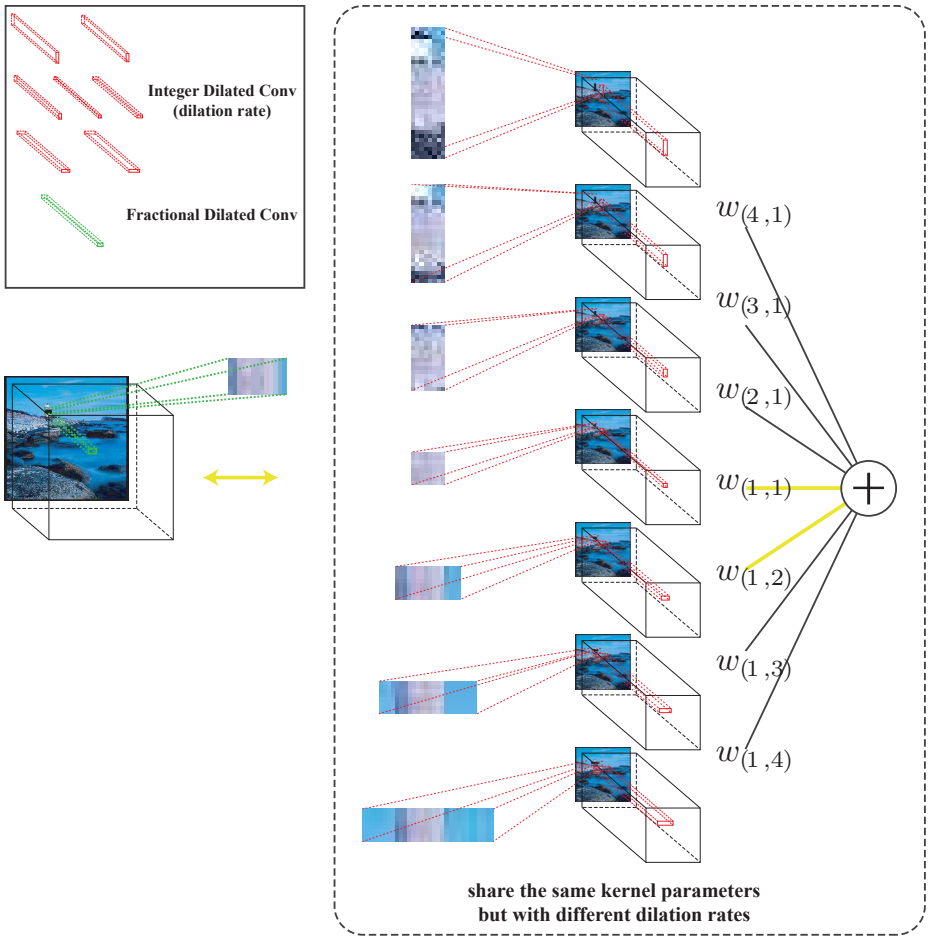


Figure 3.5: Illustration for mini-batch compatibility: the distributive property of convolution operation (*c.f.* Eq. (3.3)) makes the fractional dilated conv easily implemented and compatible for mini-batch computation with a zero-padded weight vector/matrix (*c.f.* Eq. (3.5))

with respect to w and h as shown in Fig. 3.4. In addition, all the integer dilation kernels share the same kernel parameters and thus no extra learning parameters are introduced.

3.3.3 Mini-Batch Computation and Implementation

To implement the dynamic kernel interpolation in Eq. (3.1) and Eq. (3.2) directly, we need to rewrite the kernel-level code due to the diverse kernels in mini-batch. However, through the following deduction, we show that the proposed method can be easily implemented by common deep learning libraries, *e.g.* PyTorch and TensorFlow.

Using the distributive property of convolution operation, the transformation of the feature maps generated by the adaptive fractional dilated Conv kernels in Eq. (3.1) can be formulated as:

$$\begin{aligned} f_{n+1} &= k^n_{AFDC} * f_n \\ &= \left[(\lceil \tfrac{w}{h} \rceil - \tfrac{w}{h}) k^n_{(1, \lfloor \frac{w}{h} \rfloor)} + (\tfrac{w}{h} - \lfloor \tfrac{w}{h} \rfloor) k^n_{(1, \lceil \frac{w}{h} \rceil)} \right] * f_n \\ &= (\lceil \tfrac{w}{h} \rceil - \tfrac{w}{h}) k^n_{(1, \lfloor \frac{w}{h} \rfloor)} * f_n + (\tfrac{w}{h} - \lfloor \tfrac{w}{h} \rfloor) k^n_{(1, \lceil \frac{w}{h} \rceil)} * f_n \end{aligned} \tag{3.3}$$

where f_n denotes the feature maps for the nth layer and $*$ denotes convolution.

In mini-batch training and inference, we can construct multiple kernels with different dilation rates $(rate^i_k, rate^j_k)$ from the same kernel parameters and then use a zero-padded interpolation weight vector $\mathbf{w}$ to compute the operation adaptively for each image as:

$$\begin{aligned} f_{n+1} &= k^n_{AFDC} * f_n \\ &= \sum_k w_{(rate^i_k, rate^j_k)} k^n_{(rate^i_k, rate^j_k)} * f_n \\ &= \mathbf{w} \widetilde{\mathbf{f}}_n \end{aligned} \tag{3.4}$$

which is just the inner product of two vectors:

$$\mathbf{w} = [w_{(rate^i_1, rate^j_1)}, ..., w_{(rate^i_K, rate^j_K)}] \tag{3.5}$$

and

$$\widetilde{\mathbf{f}}_n = [k^n_{(rate^i_1, rate^j_1)} * f_n, ..., k^n_{(rate^i_K, rate^j_K)} * f_n]^\top \tag{3.6}$$

where the number of dilation kernels is K. As shown in Fig. 3.5, the interpolation

weight $w_{(rate_k^i, rate_k^j)}$ for each instance is either $w_{(rate_k^i,1)}$ or $w_{(1,rate_k^j)}$, defined as follows:

$$
\begin{aligned}
w_{(rate^i,1)} &= \begin{cases} r-(rate^i-1), & \text{if } rate^i - r \in [0,1) \\ (rate^i+1)-r, & \text{if } rate^i - r \in (-1,0) \\ 0, & \text{else} \end{cases} \\
w_{(1,rate^j)} &= \begin{cases} r-(rate^j-1), & \text{if } rate^j - r \in [0,1) \\ (rate^j+1)-r, & \text{if } rate^j - r \in (-1,0) \\ 0, & \text{else} \end{cases}
\end{aligned}
\tag{3.7}
$$

In mini-batch in size B and K dilation kernels , the computation of $n+1$th feature maps $\mathbf{F}_{n+1}$ can be formulated as:

$$
\mathbf{F}_{n+1} = \begin{bmatrix} \mathbf{w}^{0\top}\mathbf{F}^0 & \mathbf{w}^{1\top}\mathbf{F}^1 & \dots & \mathbf{w}^{I\top}\mathbf{F}^B \end{bmatrix}^\top \tag{3.8}
$$

where the weight vector for bth instance in mini-batch is:

$$
\mathbf{w}^b = \begin{bmatrix} w^b_{(rate_1^i, rate_1^j)} & w^b_{(rate_2^i, rate_2^j)} & \dots & w^b_{(rate_K^b, rate_K^j)} \end{bmatrix}^\top \tag{3.9}
$$

and the feature vector $\mathbf{F}^b$ is the row vector in:

$$
\begin{bmatrix}
k^n_{(rate_1^i, rate_1^j)} * f_n^1 & k^n_{(rate_2^i, rate_2^j)} * f_n^1 & \dots & k^n_{(rate_K^i, rate_K^j)} * f_n^1 \\
k^n_{(rate_1^i, rate_1^j)} * f_n^2 & k^n_{(rate_2^i, rate_2^j)} * f_n^2 & \dots & k^n_{(rate_K^i, rate_K^j)} * f_n^2 \\
\vdots & \vdots & \vdots & \vdots \\
k^n_{(rate_1^i, rate_1^j)} * f_n^B & k^n_{(rate_2^i, rate_2^j)} * f_n^B & \dots & k^n_{(rate_K^i, rate_K^j)} * f_n^B
\end{bmatrix}
\tag{3.10}
$$

The computation of the above Eq. (3.10) can be done efficiently in the mini-batch as:

$$
\begin{bmatrix} k^n_{(rate_1^i, rate_1^j)} * \mathbf{F}_n & k^n_{(rate_2^i, rate_2^j)} * \mathbf{F}_n & \dots & k^n_{(rate_K^i, rate_K^j)} * \mathbf{F}_n \end{bmatrix} \tag{3.11}
$$

Table 3.1: Computation comparison: training batch size is set to 16, test batch size is set to 32. The speed is the average result for 100 iterations from the test on single GTX 1080Ti. The fractional dilated Conv is embedded for all BottleNets in ResNet50 while * denotes additional embedding dilation for the first 7×7 Conv layer as well.

Network	#Params	#Mult-Adds	Speed (train)	Speed (test)
VGG16	138M	15.3G	8.14 it/s	12.91 it/s
2-dilation	138M	30.7G	2.70 it/s	3.85 it/s
3-dilation	138M	46.1G	1.75 it/s	2.28 it/s
7-dilation	138M	109.1G	0.73 it/s	0.93 it/s
ResNet50	25.6M	3.5G	12.49 it/s	22.80 it/s
2-dilation	25.6M	5.6G	8.32 it/s	14.81 it/s
2-dilation*	25.6M	6.5G	6.20 it/s	9.88 it/s
3-dilation	25.6M	7.5G	6.28 it/s	10.68 it/s
3-dilation*	25.6M	8.9G	4.35 it/s	6.92 it/s
7-dilation	25.6M	10.6G	3.22 it/s	5.28 it/s
7-dilation*	25.6M	18.8G	2.08 it/s	3.12 it/s

We note that the activation function and batch normalization are omitted in the formulas for concise illustration.

The formula in Eq. (3.8) can be interpreted as a dot production followed by a sum reduction between interpolation weight matrix $\mathbf{W}$ and Eq. (3.11), which thus can be efficiently implemented by common deep learning frameworks (Pytorch, Tensorflow, etc.). Each integer dilated Conv, $k^n_{(rate^i_k, rate^j_k)} * \mathbf{F}_n$ in Eq. (3.11), is computed as a normal dilated Conv layer with the shared learning parameters.

Computational overhead The computational overhead is determined by the number of integer dilated kernels and the number of convolutional layers whose kernel sizes are not 1×1. As shown in Table 3.1, the BottleNet in ResNet50 [3] contains two 1×1 kernels and one 3×3 kernel. Since only 3×3 kernel introduces the computational overhead, the computational cost for 2 integer dilations is roughly 1.5 times of the original model, while VVG16 [7] consists of the majority of 3×3 kernels and thus the computation cost is approximately 2 times. Some additional computational overhead is caused by the interpolation operation of different dilation kernels.

Figure 3.6: Grouping strategy to reduce computational overhead: The integer dilated Convs can be shared by properly grouped images according to aspect ratios.

Reducing overhead with a grouping strategy In practice, the aspect ratios, $\frac{w}{h}$, of most of images would fall into $[\frac{1}{2}, 2]$, *e.g.* 97.8% of the training and testing images in the AVA [1] dataset. Training efficiency can be optimized by grouping batches, *e.g.* training with three dilation kernels for the most batches, $DilationRates = \{(2,1),(1,1),(1,2)\}$ for the images whose aspect ratios fall into $[\frac{1}{2}, 2]$. For the datasets with more diverse aspect ratios, a more fine-grained grouping strategy could be applied. As illustrated in Fig. 3.6, images with aspect ratio range $[4, 3]$ (above) and $[\frac{1}{2}, 1]$ (below) share the valid integer dilated Convs in the grouped batches.

Parallel optimization The calculation of multiple integer dilated kernels in each convolutional layer is equivalent to broadening the output channel size by the number of dilation kernels. In another words, the computation of dilated Conv group, $\{k^n_{(rate^i_k, rate^j_k)} * \mathbf{F}_n\}$, can be optimized through parallel computing. WideResNet [80] claims that increasing the width of Conv layers is more accommodating to the nature of GPU computation and helps effectively balance computations more optimally. However, from Table 3.1, the actual training and testing speeds are approximately linearly correlated with # Muti-Adds, which could be attributed to the current im-

plementation of the framework (TensorFlow) and can be improved by further parallel optimization.

We note that many base networks are stacked mainly with the permutation of 1×1 and 3×3 kernels and they can be applicable to embed AFDC in terms of the training and inference speed, *i.e.* [3, 9, 10, 80, 31] in ResNet stream and [11, 12, 13] in MobileNet stream. Besides, the adaptation is easy because our method is parameter-free. Overall, the random-size warping preserves the composition of the original image and also provides data augmentation to train the network with scale invariance. AFDC can adaptively construct fractional dilated kernels according to the spatial distortion information in a computation-efficient manner.

3.3.4 Composition-Aware Structure and Loss

The commonly-used network structures for the task of image classification usually incorporate global pooling before the fully connected layers [31, 3, 9, 8, 10]. The global pooling eliminates spatial variance which is helpful for the task of image recognition by training the networks with spatial invariant ability, but it causes the loss of localization information for image aesthetics assessment. Motivated by spatial pyramid pooling [157], MNA-CNN-Scene [81], several efforts are made to learn the information of spatial image compositions. First, we use multiple adaptive pooling modules [157] to output $g_i * g_i$ grids and feed them into the fully-connected layers (*c.f.* Fig. 3.2). The localization factors for image aesthetics assessment are highly correlated with the image symmetry and the overall image structure. Then, we aggregate the outputs after the fully-connected layers by concatenation. To limit the number of model parameters and prevent from overfitting, the module of each adaptive pooling layer outputs $\frac{num_{features}}{num_{grids}}$ channels.

Following the work in [2], we train our network to predict 10-scale score distribution with a softmax function on the top of the network. To get both the mean score prediction and the binary classification prediction, we calculate the weighted sum of

score distribution $\sum_{i=1}^{10} i \cdot p_i$. We use the ordered distribution distance, Earth Mover Distance [2], as our loss function:

$$EMD(p, \hat{p}) = (\frac{1}{N} \sum_{k=1}^{N} |CDF_p(k) - CDF_{\hat{p}}(k)|^r)^{1/r} \tag{3.12}$$

where $CDF_p(k)$ is the cumulative distribution function as $\sum_{i=1}^{k} p_i$. As stated in Section 3.1 and the results in [2], predicting the score distribution can provide more information about image aesthetics compared to the mean scores or binary classification labels.

3.4 Experimental Results

Following [2, 152, 82, 81, 148], we have evaluated our proposed method over AVA dataset [1]. The AVA contains around 250,000 images and each image contains the 10-scale score distribution rated by roughly 200 people. For a fair comparison, we use the same random split strategy in [2, 154, 152, 82, 81, 1] to generate 235,528 images for training and 20,000 images for test.

3.4.1 Implementation Details

We use ResNet-50 [3] as the backbone network due to its efficiency on computation and graphic memory as discussed in Section 3.3.3. We replace all the 3×3 Conv layers in each BottleNet with our proposed adaptive fraction dilation Conv layers. It is easy to plug AFDC into the common CNN architectures since it does not introduce any extra model parameters. We use the same EMD loss in Eq. (3.12) with $r = 2$ for better back propagation. To accelerate training, we use the grouping strategy discussed in Section 3.3.3. For the first 12 epochs, we train the model with three dilation kernels, 1×2, 1×1, 2×1 on the grouped images since the aspect ratios for 97.8% training and validation images fall between $[\frac{1}{2}, 2]$. Then we train the model with seven dilation kernels, 1×4, 1×3, 1×2, 1×1, 2×1, 3×1, 4×1, for the remaining

6 epochs and select the best model from the results in the validation dataset. We note that the training and test speed could be further accelerated by a more fine-grained grouping strategy. We transfer the network parameters (pre-trained on ImageNet) before the fully connected layer and set the initial learning rate to 0.01 for the first 6 epochs. Then we dampen the learning rate to 0.001 for the rest of the training epochs. We find that setting initial learning rate to 0.001 with a decay rate 0.95 after every 10 epochs can produce comparable results but converges more slowly. The weight and bias momentums are set to 0.9. We transfer the network parameters (pre-trained on ImageNet) before fully connected layer. During the test, we average the prediction results from four warping size $\{224 \times 224, 256 \times 256, 288 \times 288, 32 \times 320\}$.

3.4.2 Ablation Study

In this section, we introduce the steps to build the final model and analyze the effects of each module step by step: (1) Replacing random cropping with composition-preserving random warping; (2) Replacing vanilla Conv with AFDC in the aspect-ratio-preserving pre-trained model on ImageNet; (3) Adding SPP modules to learn image composition.

Random Warping. For the data augmentation, input images in NIMA [2] are rescaled to 256×256, and then a crop of size 224×224 is randomly extracted. They also report that training with random crops without rescaling produces the results that are not compelling due to the inevitable changes in image compositions. In order to preserve the complete composition, we replace the random-cropping with random-size warping by randomly warping each batch into square size in $[224, 320]$ during each iteration. The network suffers from overfitting without using random warping. We note that non-square-size warping may further help with generalization and potentially train AFDC more robustly.

From Table 3.2, we generate slightly better results (Vanilla Conv (ResNet50)) compared with NIMA [2]. We use the same loss (EMD loss) and network (ResNet50, our

Table 3.2: Test result comparison on AVA [1]: The evaluation metrics are following [2]. Reported accuracy values (cls. acc.) are based on binary image classification. MSE (mean squared error), LCC (linear correlation coefficient) and SRCC (Spearman's rank correlation coefficient) are computed between predicted and ground truth mean scores. EMD measures the closeness of the predicted and ground truth rating distributions with $r = 1$ in Eq. (3.12). AFDC (random-size cropping) transfers the model trained with widely used data augmentation method in ImageNet, while AFDC (aspect-ratio-preserving pretrain) transfers the model trained with aspect-ratio-preserving data augmentation.

network	cls. acc.	MSE	EMD	SRCC	LCC
NIMA(VGG16)[2]	0.8060	-	0.052	0.592	0.610
NIMA(Inception-v2)[2]	0.8151	-	0.050	0.612	0.636
NIMA(ResNet50, our implementation)	0.8164	0.3169	0.0492	0.6166	0.6388
Vanilla Conv (ResNet50)	0.8172	0.3101	0.0481	0.6002	0.6234
AFDC (random-size cropping pretrain)	0.8145	0.3212	0.0520	0.6134	0.6354
AFDC (aspect-ratio-preserving pretrain)	0.8295	0.2743	0.0445	0.6410	0.6653
AFDC + SPP	**0.8324**	**0.2706**	**0.0447**	**0.6489**	**0.6711**

implementation) as NIMA [2]. Comparable results have shown that random warping is an effective data augmentation alternative and it preserves the image composition.

Aspect-Ratio-Preserving Pretrain. We replace the vanilla convolution layers with AFDC in ResNet50. In our experiments, we find that, fine-tuning the fractional dilated convolution network results in similar validation accuracy compared to the original network (*c.f.* AFDC (random-size cropping pretrain) in Table 3.2). Compatible validation results might be attributed to the pre-trained model which has a distortion-invariant ability. The widely used data augmentation [8] for network training on ImageNet contains random cropping on a window whose size is distributed evenly between 8% to 100% of the original image area with the aspect ratio constrained to $[\frac{3}{4}, \frac{4}{3}]$. The model is trained with distortion invariance, which has the opposite interest of our method that tries to preserve the original aspect ratio.

For better transfer learning, we pre-train the ResNet50 [3] on ImageNet [139] without distortion augmentation. Specifically, we sample the 8% to 100% crop size to the image area with a square window, which is slightly modified comparing to the data augmentation method in [8]. As in Table 3.2, transferring the model from the

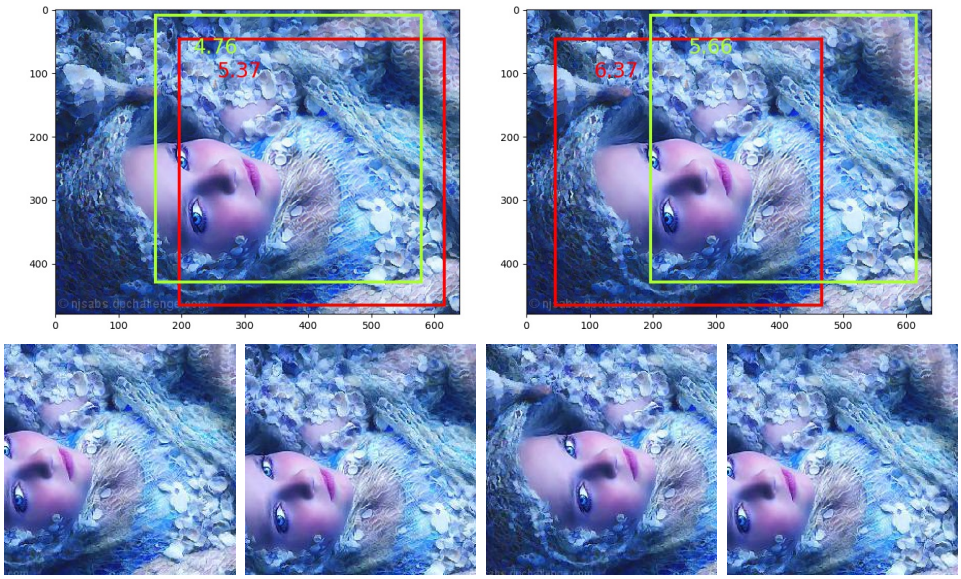


Figure 3.7: The cropping results for the model trained with global pooling (left) and SPP (right). The two cropping samples are obtained by using a sliding window with the lowest score (green) and the highest score (red). The image is firstly resized to 256. A sliding window search with size 224 and stride 10 is applied.

aspect-ratio-preserving pre-train, we improve the overall test results (AFDC (aspect-ratio-preserving pre-train)) by a margin from the vanilla Conv counterpart.

Composition-Aware Structure. For better representation learning of composition, we use three different scales for SPP, $\{1 \times 1, 2 \times 2, 3 \times 3\}$. The network with a global pooling layer is equivalent to using only one scale, 1×1. From Table 3.2, the network with SPP modules (AFDC+SPP) generates better results comparing to the network with the global pooling layer (AFDC). The experimental results have shown that incorporating the localization information could benefit the learning of image compositions. In Fig. 3.7, the automatic cropping example demonstrates that the ability of localization/composition discrimination is important to find a good cropping result when the global cue in each cropping box has a similar distribution (color, lighting *et al.*). The model leaned with SPP modules can infer cropping respecting the image compositions, *e.g.* the relative position of eye and face in the example. We also tried $num_{grids} = 5$ and found that the results were not compelling due to the

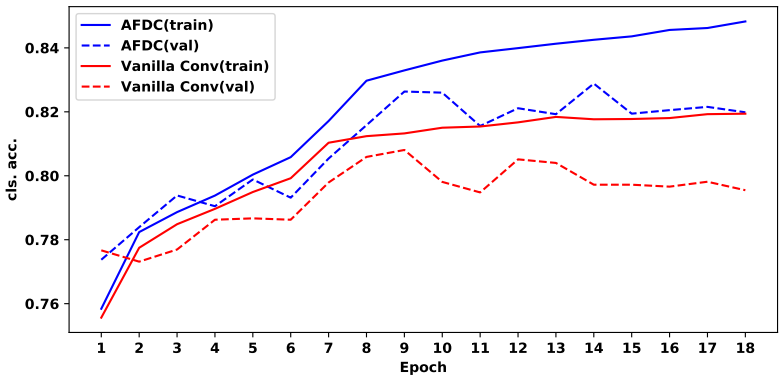


Figure 3.8: The comparison of learning curves: the backbone networks here are all ResNet-50 [3].

overfitting from extra model parameters. Three different scales are quite consistent with the common aesthetic rules (global information, symmetrical composition in horizontal and vertical direction, the rules of the thirds).

3.4.3 Effectiveness of AFDC

Learning Representation and Generalization From the experiments in Fig. 3.8, we argue that preserving aspect ratio information is essential for learning photo aesthetics since our method not only improves the validation results but also improves the training results. Without extra learning parameters, AFDC improves both learning representation and generalization ability. As discussed in Section 3.1, preserving the image aesthetics information completely omits the label noises caused by random warping and thus facilitates the learning process. The additional aesthetic features related to the aspect ratios allow the model to be more robust and discriminative. To further probe the effects of embedding aspect ratio, we compare different ways to incorporate the dilated convolution and the results are reported in Table 3.3. When trained with vanilla Conv (top rows in Table 3.3), AFDC is superior to other dilated Conv methods during the test. It implies the potential optimal between nearest two integer dilated kernels. After training with AFDC (bottom rows in Table 3.3), it further validates the effectiveness of AFDC, which is guided by the helpful supervision

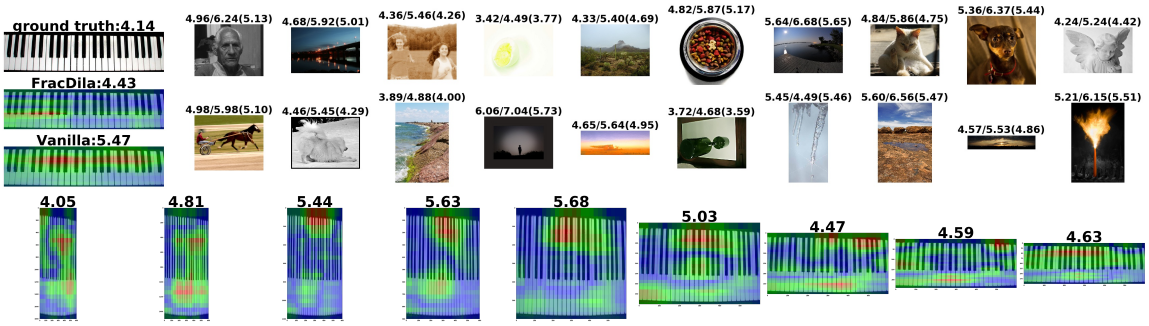


Figure 3.9: The top 20 images with the biggest difference between the AFDC and Vanilla CNN. The scores are represented as FracDilated/Vanilla(GroundTruth). The activation maps from AFDC change along the aspect ratios. The activation is generated by a sliding blocking window and average the score change in the output space.

of aspect ratios. We note that such experiments are accessible because our method is parameter-free.

Overall, our proposed AFDC can learn more discriminative and accurate representations related to aesthetics perception, resulting in better generalization by leveraging extra supervision from the information of image aspect ratios.

Discriminative to Aspect Ratios As shown in the right plots from Fig. 3.3, along the change of objects' aspect ratios by different wrapping size, the proposed fractional dilated Conv can be discriminative(blue dashed line) while the vanilla model infers similar scores(green dashed line). Moreover, the proposed method produces a multi-modal score distribution, which reflects it learns complex interpretation about the relation between aspect ratio and aesthetic perception. It is consistent with that designing a better aspect ratio of the architectures/cups or finding a good photography angle is a complex process. From the activation maps in Fig. 3.9, we can see the model actually activates different spatial parts along the change of the texture in images while the vanilla model can not differentiate it due to the same-sized wrapping. In addition, the proposed method captures the noticeable trend about aesthetic perception. For example, the left-side images from row 3 cause more unpleasing aesthetic perception than the right-side images, while the images from row 4 do the opposite.

For the images from row 2, the change of aesthetic perception caused by distortion is not significant and the proposed model generates less differential scores.

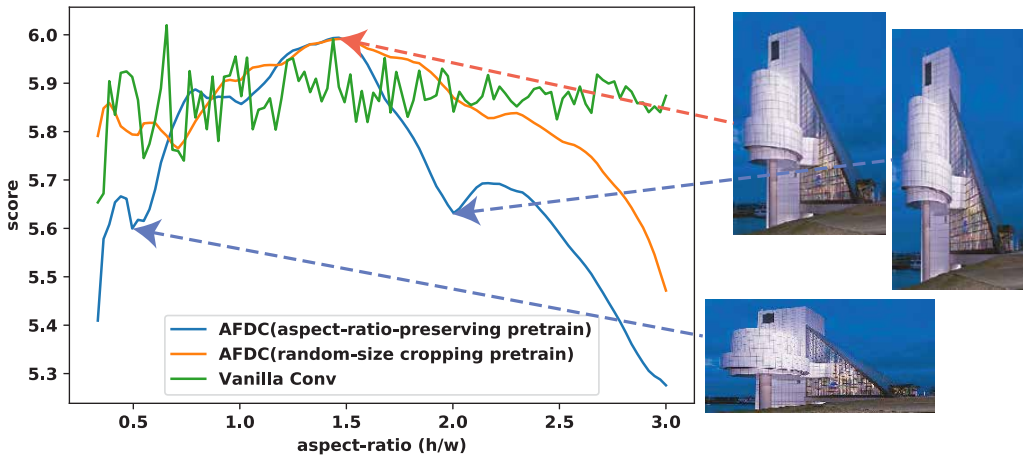


Figure 3.10: Comparison of discrimination to the change of aspect ratios.

To further investigate the response to aspect ratios, we resize the same image into different aspect ratios and test the results on different trained models. As shown in Fig. 3.10, AFDC (blue line) is discriminative to the change of aspect ratios. The small fluctuation of vanilla Conv (green line) is attributed to sampling change from resizing process. The model with random-size cropping pretrain on Imagenet (orange line) is less discriminative to capture the aesthetics perception related to aspect ratio due to its distortion-invariant pretrain. Moreover, the proposed method produces a multi-modal score distribution, which reflects that it learns complex relation between the aspect ratio and the aesthetics perception. It is in line with the notion that designing better aspect ratios or finding aesthetically pleasing photography angles is not trivial.

Due to the constraint of training dataset, we admit that the learned perception related to the aspect ratios is not satisfactory yet even the model learns from different aspect ratios. As a matter of factor, the learning ability is available for our proposed method when training on a more specific targeted dataset. It could be utilized in

Table 3.3: The test result comparison of different convolutions: The results are obtained with trained parameters by vanilla Conv (above) and AFDC (below). Test processes are conducted by different calculation methods for interpolation weights, **w** in Eq. (3.5). Vanilla Conv, constant dilation, nearest integer dilation and second nearest integer dilation can be interpreted as feeding one-hot interpolation weight vector into the networks.

Train	Test	cls.acc.	MSE	EMD
vanilla	vanilla	0.8172	0.3101	0.0481
	constant dilation rate = [2,1]	0.8072	0.5163	0.0610
	second nearest integer dilation	0.8091	0.5368	0.0620
	mean of nearest two integer dilations	0.8117	0.4558	0.0576
	nearest integer dilation	0.8114	0.4322	0.0562
	adaptive fractional dilation	0.8132	0.4133	0.0553
AFDC	vanilla	0.8085	0.3210	0.0581
	constant dilation rate = [2,1]	0.8132	0.3182	0.0576
	second nearest integer dilation	0.8156	0.3003	0.0476
	mean of nearest two integer dilations	0.8274	0.2771	0.0457
	nearest integer dilation	0.8277	0.2757	0.0457
	adaptive fractional dilation	**0.8295**	**0.2743**	**0.0445**

automatic/auxiliary photo enhancement with not only color space transformation but also with spatial transformation, *e.g.* profile editing, multi-shot selection and automatic resizing.

3.4.4 Comparison With the State-of-the-Art Results

We have compared our adaptive fractional dilated CNN with the state-of-the-art methods in Table 3.4. The results of these methods are directly obtained from the corresponding papers. As shown in Table 3.4, our proposed AFDC outperforms other methods in terms of cls.acc and MSE, which are the most widely targeted metrics. Compared with NIMA(Inception-v2) [2] which uses the same EMD loss, our experimental results have shown that preserving the image aesthetic information completely results in better performance on image aesthetics assessment. We follow the same motivation from MNA-CMM-Scene [81], while our proposed method is applicable to mini-batch training which contains images with different aspect ratios. The experimental results have shown adaptive embedding at kernel level is an effective way to

Table 3.4: Comparison with the SOTA methods: The four patches are warping size $\{224, 256, 288, 320\}$.The single patch is warping size 320 selected from the best results.

Method	cls. acc.	MSE	SRCC
MNA-CNN-Scene [81]	76.5%	-	-
Kong *et al.* [153]	77.3%	-	0.558
AMP [152]	80.3%	0.279	0.709
Zeng *et al.* (resnet101) [151]	80.8%	0.275	0.719
NIMA (Inception-v2) [2]	81.5%	-	0.612
MP-Net [82] (50 cropping patches)	81.7%	-	-
Hosu *et al.* [83] (20 cropping patches)	81.7%	-	**0.756**
A-Lamp [82] (50 cropping patches)	82.5%	-	-
MP_{ada} [84]($\geq$ 32 cropping patches)	83.0%	-	-
ours (single warping patch)	82.98%	0.273	0.648
ours (4 warping patches)	**83.24%**	**0.271**	0.649

learn more accurate aesthetics perception. Compared with multi-patch based methods [82, 83, 84], our unified model, which learns the image aesthetic features directly from the complete images in an end-to-end manner, can better preserve the original aesthetic information and alleviate the efforts to aggregate sampling prediction, *e.g.* complicated path sampling strategy and manually designed aggregation structure in [82]. Moreover, our method is much more efficient without feeding multiple cropping patches sampled from original images and could be more applicable for the application. Furthermore, it is much succinct due to its parameter-free manner and can be easily adapted to popular CNN architectures.

3.5 Conclusion

In this chapter, an adaptive dilated convolution network is developed to explicitly learn representations of aspect ratios for image aesthetics assessment. Our proposed method does not introduce extra model parameters and can be plugged into popular CNN architectures. Besides, a grouping strategy has been introduced to reduce computational overhead. Our experimental results have demonstrated the effectiveness of our proposed approach. Even our adaptive dilated convolution network was pro-

posed to support image aesthetics assessment, it can also be applied in other scenarios when image cropping or warping may introduce label noises. Moreover, adaptive kernel construction in a parameter-free manner provides an intuitive approach to design dynamic embedding at kernel level, which aims at better learning representation and generalization.

CHAPTER 4: INVARIANCE REPRESENTATION LEARNING FOR PERSON RE-IDENTIFICATION

4.1 Problems and Motivation

4.1.1 Variances of Person Re-Identification

Person re-identification (ReID) is an essential component of intelligent computer vision systems. It has drawn increasing interest in many applications, such as surveillance, activity analysis and long-term tracking. Given an image of a person-of-interest captured by one camera, the goal is to re-identify this person from images captured by multiple cameras without overlapping viewpoints. As an instance-level recognition problem, the ReID task is inherently challenging. First, intra-class variations are typically huge due to significant changes of visual appearances caused by camera viewing conditions, human pose variations, occlusions, *et al*. Second, the inter-class variations can be quite small because people may wear similar clothes.

To address these challenges of intra-class diversity and inter-class inseparability, a lot of efforts have been devoted to deep learning for its strong capability on discriminative feature extraction. Most of existing methods put the training process of person ReID under classification framework, where intermediate features are extracted to compute the similarity between query and gallery images during test. Some tailor-made neural networks are proposed to incorporate localization/attention or disentanglement for feature alignment. The former one aligns features in 2D spatial dimension and the latter one targets latent semantic alignment, but the essence of those approaches is instance-level alignment (*c.f*. Fig. 4.1 left). In addition, a large variety of loss functions have been proposed for metric learning in person ReID. For exam-

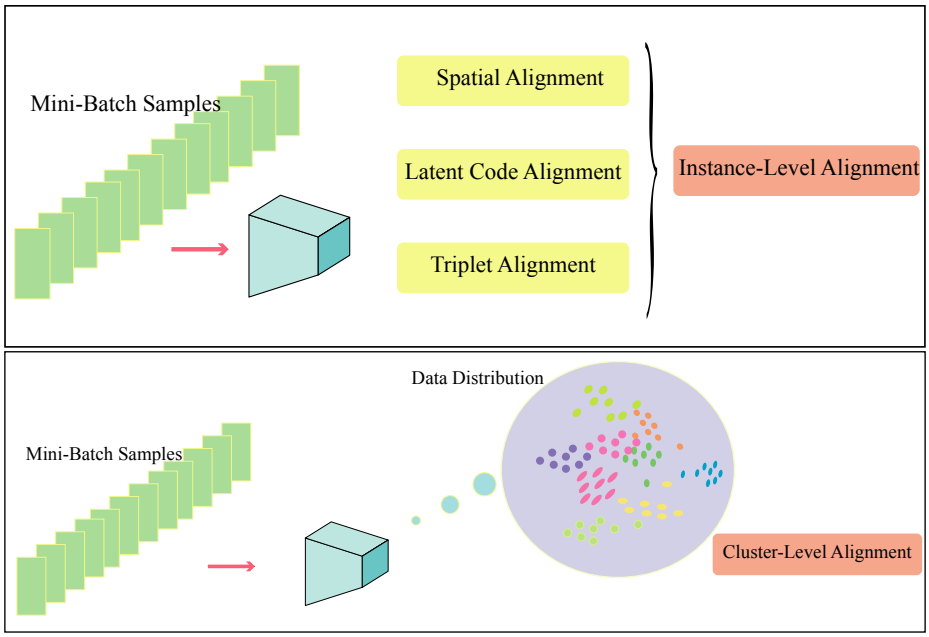


Figure 4.1: From the view of alignment modality: instance-level alignment (top) and cluster-level alignment (bottom).

ple, the two most prevalent loss functions are classification loss, *e.g.* cross entropy loss, and metric-learning based loss, *e.g.* hard-negative mining triplet loss [158]. For those advanced networks driven by such popular loss definitions, although successful, we argue that they can still be categorized as instance-level alignment (*c.f.* Fig. 4.1 left). The intrinsic reason of such constraint is attributed to the adoption of general classification framework, where the interaction it builds can only dwell within the sampled mini-batch but cannot see more neighbors in the distribution of the whole dataset. As a result, it inhibits the growth of intra-class compactness and inter-class separability.

4.1.2 Invariance Representations From a Global Viewpoint

Aiming to break through the aforementioned limitations and step beyond the instance-level feature alignment, we propose a succinct and efficient method to enable cluster-level interaction in feature space, targeting the alignment from an overview of latent feature distribution (*c.f.* Fig. 4.1 right). ReID is in essential a metric learning

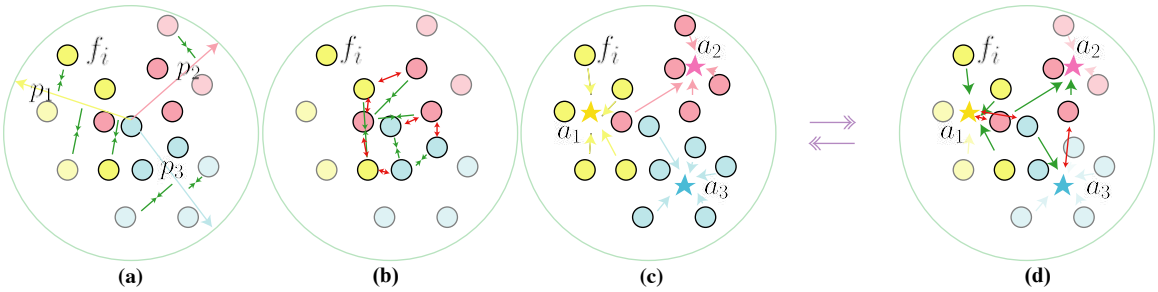


Figure 4.2: From the view of optimization: Anchor loss provides more stability and consistency for optimization. (a) Classification loss pulls feature sample f_i towards the corresponding classifier vector p_j in fully connected layer; (b) Triplet loss probes the interaction within sampled mini-batch (denoted as solid color); (c-d) Anchor loss enables the sampled mini-batch to see the anchors aggregated from all the siblings through iterative aggregation(c) and alignment(d).

problem. When projected to the learned feature space, feature points are expected to gather into compact clusters respecting their labels and such cluster-level interaction may inhabit better formulation of the clusters. We define the center of each feature cluster as the anchor. In a computational efficient manner, the anchors generated from aggregation serve as a supervision from the distribution of whole dataset and enable the model to see other training images in the dataset indirectly. In practice, after the saturation of traditional training, we manipulate two iterative steps to further intensify the cluster compactness: (a) Aggregate cluster features across dataset, stepping beyond the limitation of a classification framework; (b) Align features under the guidance from aggregated anchors. We claim such cluster-level feature alignment is much more promising for identity-related representation.

Besides the view of feature alignment modality (*c.f.* Fig. 4.1), the proposed method, called anchor loss, provides consistent optimization for metric learning which benefits training as well as generalization process. The classification loss tries to align the features in orders according to the classification labels. Specifically, the inner product $\mathbf{f}_i^\top \mathbf{p}_j$ between classifier p_j in the full-connected layer and the feature vector f_i is increased as p_j and f_i pulling towards each other (*c.f.* Fig. 4.2(a)). It has promising convergence but unnecessary penalize the intra-class variance if classifier

diversifies channel-wise focus on decision. Also, it handles training mini-batch samples by simply averaging individual losses, thus can only build sample connection from the identity implicitly. Triplet loss [158] tries to align features in more explicitly way. It optimizes the intra-class and inter-class distance by mini-batch interaction with proper sampling. From Fig. 4.2(b), we can see that the optimization direction of triplet loss is highly dependent on the mini-batch sampling and inevitably introduces uncertainty and inconsistency. On the other hand, anchor loss enables the sampled mini-batch to see the anchors $\mathbf{a}_j$ aggregated from all the siblings, bearing more consistency (*c.f.* Fig. 4.2(c-d)). Anchors generated from aggregation provide strong guidance and propagate the global information from the distribution of dataset to local mini-batch training. Based on extensive experiments, we demonstrate that anchor loss can consistently boost the generalization.

Overall, in this proposed paper, we learn a metric to overcome intra-class diversity and inter-class confusion for person ReID based on anchor-based min-batch training. Although each mini-batch of samples is a small subset of the dataset, we can successfully capture the global information during training through anchors which play key roles in the proposed cluster-level feature alignment. A small number of representative anchors propagate rich knowledge from the distribution of dataset into local training batch in a computational efficient manner.

4.2 Related Work

Person Re-Identification A large group of person re-identification network focuses on feature alignment. In general, there are two kinds of feature alignment: (a) spatial feature alignment by attention and localization (b) latent feature alignment by disentanglement.

In spatial feature alignment, it can be categorized as self-supervised and extra-supervised methods. We consider hand-crafted splitting as one representation of self-supervision. Sun *et al*. [91] propose PCB to split the intermediate features hor-

izontally in order to align the feature in local spatial parts and is widely used by [159, 160, 161, 162, 163]. They have a strong assumption that the spatial distributions of human bodies and human poses are exactly matching. Besides, self-derived attention serves as a tool to bias the allocation of available resources towards the most informative parts of an input. Quite some works [164, 165, 166, 167] proposed similar and effective part-aligning CNN networks for locating context regions and then extract these regional features for ReID. Extra-supervision leverages human part detector [168], human pose [169] or human body parsing [170] to provide more accurate localization. For example, [92, 171, 172, 167] incorporate external pose attention maps to align the feature in deformable spatial space of human body. SPReID [170] utilizes a parsing model to generate five different predefined human part masks to compute more reliable part representations, which achieves promising results on various person ReID benchmarks. Dense semantic alignment [93] went one step further, it addressed the body misalignment by leveraging the estimation of the dense semantics of a person image, and constructed a set of densely semantically aligned part images for re-identification. Other methods for spatial alignment include the attention from attribute [173], forground mask [94], *et al.*

For latent feature alignment, DG-net [96] proposed a disentanglement solution by GAN and Autoencoder to decouple input into appearance code and structure code, and extract pose-invariant features. Whereas, there has been a growing interest in using generative models to augment training data and enhance the invariance to input changes [174, 175, 176].

Metric Learning: Center Loss and Triplet Loss In end-to-end learning process, several methods propose to explore iteration within mini-batch for feature alignment . Zheng *et al.* [177] propose a verification loss to align the pairwise features. Hermans *et al.* [158] target the triplet samples and point out the triplet loss on hard examples mining is superior to batch-all triplet loss. In face recognition, parametric

center loss [178] is proposed to align the intra-class distance and it is used by [179] for person reID, which looks similar to our proposed anchors. Whereas, our motivation is essentially different. Center loss treats the parametric center as an auxiliary decision factor similar to the classifier p_j in Fig. 4.2(a), which is jointly optimized under classification framework and only builds the connection from identity label implicitly. On the contrary, our proposed anchor loss complies the embedded feature distribution and distills the knowledge from sibling samples (*c.f.* Fig. 4.2(c)) to enable the interaction in cluster level explicitly. More recently, Wen *et al.* [180] revisit center loss and propose to use the classifier layer as the center for each class, which further validates our motivation difference. Moreover, it suffers from large instability due to random initialization for parametric center. On the other hand, our method provides constant improvement because the aggregation distills knowledge from dataset distribution.

In summary, those methods never push towards the constraint of classification framework in mini-batch training and scrutinize the modality in cluster-level feature alignment. Probably due to the concern about training efficiency, cross-dataset aggregation is not fully investigated in deep CNN methods. However, we conduct a comprehensive study on different variants of the cluster-level interaction, which has demonstrated our method could be trained effectively and efficiently with small training efforts.

4.3 Proposed Method

Person ReID aims to establish the identity correspondences between each query image and gallery images across different cameras. We use Convolutional neural network (CNN) to extract image features due to its strong representation power. To learn discriminative representation that is robust against intra-class variation and interclass confusion, we take advantage of three different loss functions to train the model: (1) cross-entropy classification loss L_{cls} (Fig. 4.2(a)); (2) triplet loss L_{trip} (Fig. 4.2(b)); (3) anchor loss L_{anchor} (Fig. 4.2(d)). L_{cls} and L_{trip} [158] are widely

used for person ReID, and they are both instance-level optimization as illustrated in Fig. 4.2(a-b). To propagate rich knowledge from outside samples into each mini-batch, the designed anchor loss L_{anchor} targets cluster-level supervision which has two options:

(a) Anchor Loss for Intra-Class Compactness: L_{anchor} pulls the feature vector towards the anchor where its label belongs:

$$L_{anchor} = \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \sum_{j=1}^{C} \delta(y_i = j) D(\mathbf{f}_i, \mathbf{a}_j) \tag{4.1}$$

where $|\mathcal{B}|$ denotes the number of samples in the mini-batch $\mathcal{B}$, and C is the number of classes. $\mathbf{f}_i$ and y_i are the feature vector and the label of the sample i in $\mathcal{B}$, respectively. $\mathbf{a}_j$ is the anchor for the $j-$th class. $D(\mathbf{f}_i, \mathbf{a}_j)$ is the distance between the sample $\mathbf{f}_i$ and the anchor $\mathbf{a}_j$. $\delta(y_i = j) = 1$ if the condition is satisfied, *i.e.* the label of sample i in the mini-batch equals j; otherwise, $\delta(y_i = j) = 0$.

(b) Triplet Anchor Loss for Intra-Class Compactness & Inter-Class Separability: Motivated by hard sample mining [158], we add extra inter-class penalty to target the hard/confused anchor mining:

$$L_{TripletAnchor} = \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \sum_{j=1}^{C} \delta(y_i = j)[D(\mathbf{f}_i, \mathbf{a}_j) - \min_{k \neq j} D(\mathbf{f}_i, \mathbf{a}_k) + margin] \tag{4.2}$$

Triplet anchor loss not only pulls the samples to the anchor in the same class close, but also pushes the negative samples further away than the distance between anchor and positive samples. It could be more discriminative by simultaneously taking into account intra-class compactness and inter-class separability. On the other hand, it may import inconsistency during the optimization in a similar way as triplet loss (*c.f.* Fig. 4.2(b)).

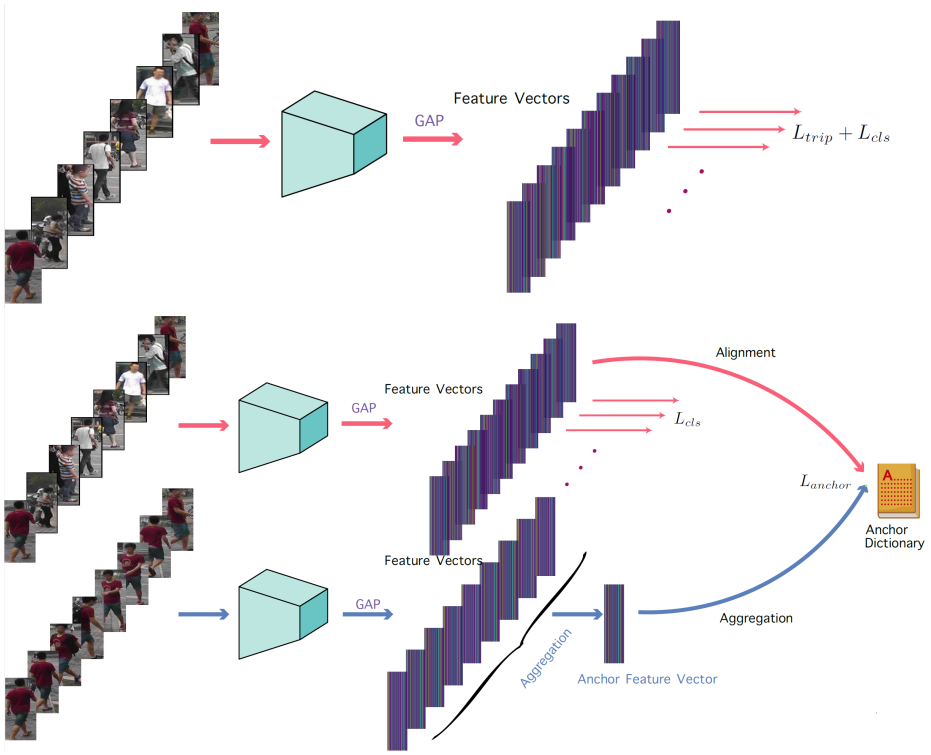


Figure 4.3: Two training stages of the proposed metric learning framework. Stage I (top): instance-level alignment with $L_{cls} + L_{trip}$. Stage II (bottom): feature aggregation respecting class label to generate anchors, and cluster-level alignment with $L_{cls} + L_{anchor}$.

4.3.1 Two-Staged Training

As shown in Fig. 4.2, anchors generated from aggregation contribute more consistent optimization and less noisy guidance during training process. However, it is based on the assumption that distribution of embedded features is approximately cluster-formed. During the early training phase when the feature distribution is still random and stochastic, such aggregated anchors may contain misleading information and impair the training process. The cluster-level supervision could be more effective after the saturation of traditional training stage, and therefore our training consists of two stages:

Stage I: Train the model in the traditional manner with loss function $L = L_{cls} + L_{trip}$, cultivating the initial formulation of clusters (*c.f.* Fig. 4.3(left));

Stage II: Train the model under the cluster level supervision with loss function $L = L_{cls} + L_{anchor}$, capturing distribution of whole dataset in embedded feature space (*c.f.* Fig. 4.3(right)).

4.3.2 Generation and Update of Anchors

During the learning process, we take the anchors as the **global** supervision from data distribution to align feature towards a better representative embedding in **local** mini-batch training, *i.e.* pushing towards the target anchor (Eq. (4.1)) and pulling samples away from the confusion anchor (Eq. (4.2)). Aggregation and alignment are iteratively performed to further reduce the intra-class variance and inter-class entanglement.

4.3.2.1 Generation

When the latent features are extracted over training dataset respecting their labels, we consider two approaches to estimate optimal anchors during aggregation:

(a) Average aggregation: When the embedded features are approximately cluster-formed, aggregate the embedded features f_i for each class in training dataset $\mathcal{T}$:

$$\mathbf{a}_j = \frac{\sum_{i \in \mathcal{T}} \delta(y_i = j)\mathbf{f}_i}{\sum_{i \in \mathcal{T}} \delta(y_i = j)} \tag{4.3}$$

(b) Voting by confidence: Taking the prediction probability $P(j|i)$ for class j as the contribution, aggregate the embedded features by a weighted mean:

$$\mathbf{a}_j = \frac{\sum_{i \in \mathcal{T}} \delta(y_i = j)P(j|i)\mathbf{f}_i}{\sum_{i \in \mathcal{T}} \delta(y_i = j)P(j|i)} \tag{4.4}$$

Eq. (4.3) treats each sample's contribution to anchor equally and could be an effective estimation to eliminate the variance of latent feature distribution caused by pose, camera view condition, background, *et al*. It may work well supposing the training samples are equally distributed in terms of variance. Eq. (4.4) takes the

classifier confidence as the contribution and help to revealing the early portrait of anchors. Intuitively, when the feature cluster distribution is still stochastic in the early training stage, easy samples, which may contains less noise and variance thus converges faster, could be close to optimal anchors and guide the hard samples moving towards the estimated optimal centers.

4.3.2.2 Update Frequency

Ideally, we should calculate anchors by either Eq. (4.3) or Eq. (4.4) after each forward and backward process when the training parameters are updated. However, such a process is unrealistic in terms of training efficiency and thus we consider three options for the update frequency:

(a) Constant (Algorithm 2): When the model is trained until initial convergence, the cluster-level feature aggregation is calculated and serves as the fixed anchors during the following fine-tuning process;

(b) Each Epoch (Algorithm 3): When the model is trained until some epoch, E_{start}, the anchors $\mathbf{c}_j$ are updated after each following training epoch by either Eq. (4.3) or Eq. (4.4);

(c) Each Iteration (Algorithm 4): In each iteration trained with anchor loss, the anchor for class j is updated as:

$$\mathbf{a}_j^{t+1} = [1 - \eta \cdot \sum_{i \in \mathcal{B}} \delta(y_i = j)] \cdot \mathbf{a}_j^t + \eta \cdot \sum_{i \in \mathcal{B}} \delta(y_i = j)\mathbf{f}_i \tag{4.5}$$

where $\mathbf{a}_j^t$ and $\mathbf{a}_j^{t+1}$ are the anchors for class j at the $t-$th and $t+1-$th iterations respectively. To approximate the anchors calculated per iteration, we design the weight as $\eta = \frac{1}{\sum_{i \in \mathcal{T}} \delta(y_i = j)}$ where $\mathcal{T}$ is the total training dataset.

Option (a) take the anchors calculated from the initial convergence as the optimal one and only one aggregation process is performed. It is based on the observation that the clusters are almost formed after the convergence of **Stage I** training and thus

Algorithm 2: Fix Anchors

for $i \leftarrow 1$ *to* ***StartEpoch*** E_{start} **do**
  Update model parameter W by $L = L_{trip} + L_{cls}$;
Update $\mathbf{c}_j$ for each class j;
for $i \leftarrow E_{start}$ *to* ***EndEpoch*** E_{end} **do**
  for *each iteration* **do**
    Update model parameter W by $L = L_{anchor} + L_{cls}$;

Algorithm 3: Update Anchors Each Epoch

for $i \leftarrow 1$ *to* ***StartEpoch*** E_{start} **do**
  Update model parameter W by $L = L_{trip} + L_{cls}$;
for $i \leftarrow E_{start}$ *to* ***EndEpoch*** E_{end} **do**
  for *each iteration* **do**
    Update model parameter W by $L = L_{anchor} + L_{cls}$;
  Update $\mathbf{c}_j$ for each class j;

Algorithm 4: Update Anchors Each Iteration

for $i \leftarrow 1$ *to* ***StartEpoch*** E_{start} **do**
  Update model parameter W by $L = L_{trip} + L_{cls}$;
for $i \leftarrow E_{start}$ *to* ***EndEpoch*** E_{end} **do**
  for *each iteration* **do**
    Update model parameter W by $L = L_{anchor} + L_{cls}$;
    Update $\mathbf{c}_j$ for each class j;

may provide stable optimization. Option (b) adaptively updates the anchors after each training epoch in a manner similar to EM optimization: Estimate the anchor location according to the current feature cluster distribution in aggregation step and maximize the cluster compactness in alignment step. Option (c) is a trade-off option between training efficiency and adaptive estimation for anchors, which can be viewed as an approximate approach to option (b). From our experiments demonstrated later, we show this method could have comparable performance with option (b) and thus provides an alternative when the size of training samples are large or in the context of online learning.

4.4 Experiments and Analysis

4.4.0.1 Experiment Setup

We adopt the bag of tricks proposed by [179], *i.e.* warm-up learning rate scheduler, random erasing augmentation [70], label smoothing, no stride down-sampling in last bottleneck of ResNet50 and bnneck (one additional batch normalization layer after classifier). We use L_2 distance for the anchor loss and its variants, which benefits stable training. We experiment our methods on three datasets, Market1501 [56], DukeMTMC-ReID [58] and CUHK03 [57]. Market-1501 have 12,936 training images with 751 different identities. Gallery and query sets have 19,732 and 3,368 images respectively with another 750 identities. DukeMTMC-ReID includes 16,522 training images of 702 identities, 2,228 query and 17,661 gallery images of another 702 identities. CUHK03-NP is a new training-testing split protocol for CUHK03, it contains two subsets which provide labeled and detected (from a person detector) person images. The detected CUHK03 set includes 7,365 training images, 1,400 query images and 5,332 gallery images. The labeled set contains 7,368 training, 1,400 query and 5,328 gallery images respectively. The new protocol splits the training and testing sets into 767 and 700 identities. We note that our method is only used during the training stage and the evaluation methods stay the same with previous approaches.

Firstly, we present the experimental comparison without triplet loss to analyze three factors: starting time, aggregation methods and anchor loss functions. Secondly, we delve into the effects of aggregation anchors from a reconstruction experiments. Thirdly, the comparison, when triplet loss is incorporated in the **Stage I** training until convergence, will be further analysed. Lastly, we demonstrate the advantages of our method comparing to the parametric center loss [178].

4.4.1 Ablation Study for Three Factors

From the experimental results in Table 4.1, we conclude the impacts of three factors:

Table 4.1: Ablation study for three factors on Market1501 dataset: starting epoch E_{start}, aggregation methods (Eq. (4.3)&Eq. (4.4)) and anchor loss choices(Eq. (4.1)&Eq. (4.2)). f and y denote the extracted feature and its label. Before E_{start} $L = L_{cls}$ is used in the first stage training. Afterwards, either $L = L_{cls} + L_{Anchor}$ or $L = L_{cls} + L_{TripletAnchor}$ is applied. We updates the anchors each epoch as in Algorithm 3.

E_{start}	Rank@1		
	$L_{Anchor}(f, y, a_{avg})$	$L_{Anchor}(f, y, a_{weighted})$	$L_{TripletAnchor}(f, y, a_{avg})$
-	93.79%	93.79%	93.79%
0	93.85%	**94.06%**	83.86%
10	**93.32%**	93.29%	93.29%
40	93.97%	93.91%	**94.09%**
70	94.09%	94.09%	**94.15%**
120	**94.18%**	94.03%	94.09%
E_{start}	mAP		
	$L_{Anchor}(f, y, a_{avg})$	$L_{Anchor}(f, y, a_{weighted})$	$L_{TripletAnchor}(f, y, a_{avg})$
-	84.69%	84.69%	84.69%
0	83.93%	**84.95%**	67.48%
10	**83.77%**	83.59%	83.65%
40	**85.49%**	85.45%	85.43%
70	85.75%	**85.89%**	85.81%
120	**85.98%**	85.96%	85.90%

(a) When to start Attributed to better cluster distribution, stating aggregation and alignment after the convergence of initial training results in better generalization. From Fig. 4.4, one can see the anchors change rapidly during the early training phase. The transition becomes steady as the training process towards saturation. It is in line with our analysis that anchor loss may impose unexpected prior and abet densely distributed clusters when applied early during training, impairing the generalization consequently.

(b) How to calculate anchors When the feature alignment is still stochastic, *i.e.* early training phase, calculating anchors with probability contribution (Eq. (4.4)) produces better results. Easy samples, which converge earlier, may contain less noises and variance, revealing the approximation of optimal anchors, *e.g.* the anchors generated from reconstruction in Fig. 4.5. After the initial training approaches convergence, the

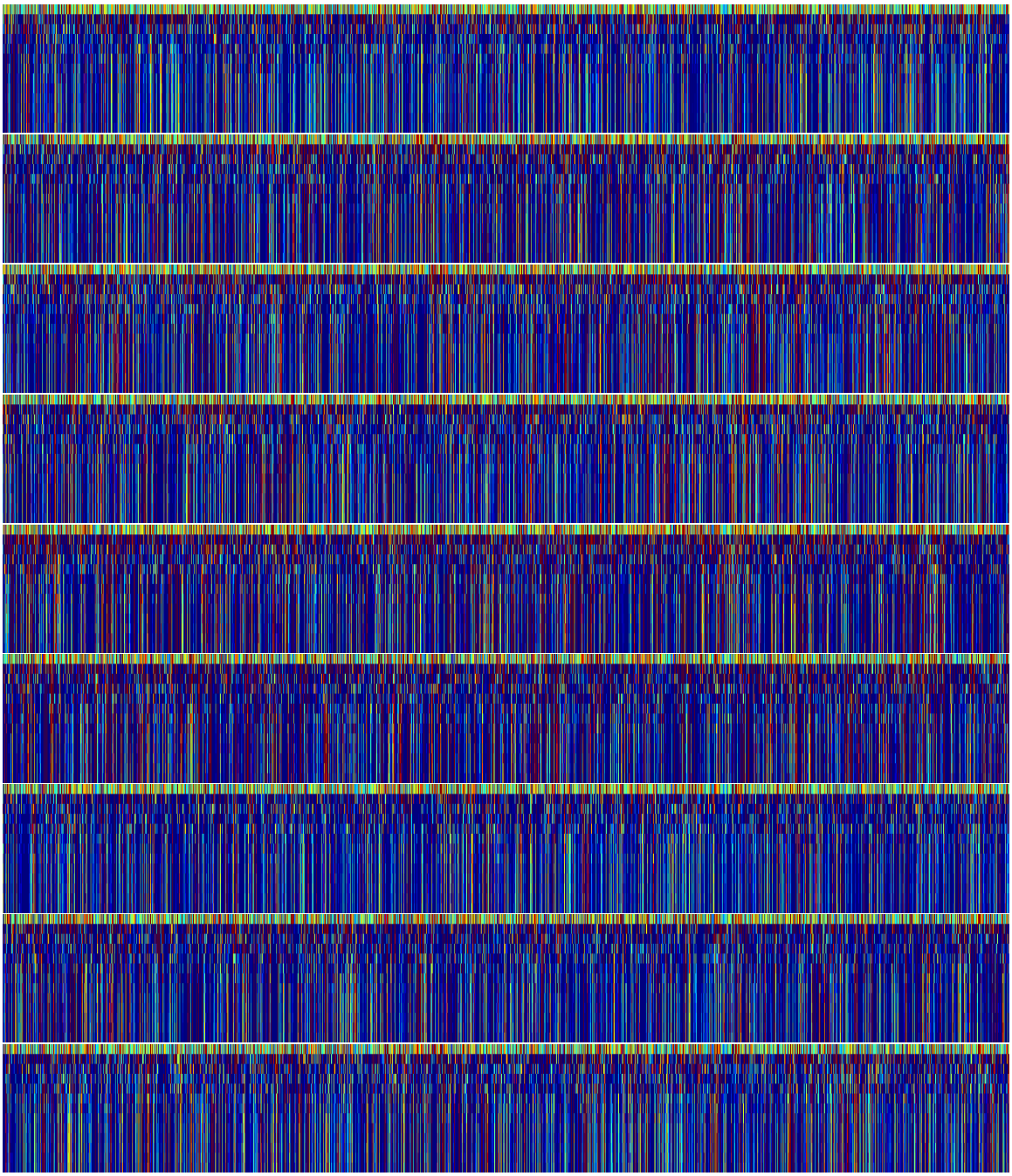


Figure 4.4: The change of sampled anchors (id = 0,1,...,7,8) along training epochs (*i.e.* 0, 10, 20, ... 120): Each image corresponds to an ID (0,1,...,7,8). Each row in image represents the anchor feature vector (2048 dimension) in the sampled epoch. Total 13 epochs from 0 to 120 with step of 10 is sampled. Zoom in to see the details. The sampled anchors are calculated from the checkpoints of training without anchor loss.

benefits of voting by confidence become less significant.

(c) How to look at anchors Triplet anchor loss and intra-class anchor loss achieve comparable results when starting the aggregation and alignment in the intermediate stage of training convergence. After the initial training saturation, intra-class anchor loss performs slightly better, implying the consistency weights more in training **Stage II**. In experiments, we also find triplet anchor loss needs a proper tuning for the margin hyper-parameter. Training is unstable when set to a high margin while easily saturated when set to a low one. The best results are found when $margin = 0$, where however training converges slower comparing to intra-class anchor loss. It validates our initial assumption that, when the anchors are well aggregated after **Stage I** training convergence, optimization with consistency provides more benefits for generalization.

4.4.2 What Would the Anchors Look Like

In order to further validate our assumption about the benefits of aggregation, we train a decoder network to reconstruct the images from the feature maps before GAP (global average pooling), where encoder is the well-trained model without anchor loss. A decoder structure slightly modified from DG-Net [96] is used. Then we generate anchors in image space by feeding the average aggregated feature maps before GAP

Figure 4.5: Reconstruction results of anchors in the training dataset (ID=0,1,2,3...,19).

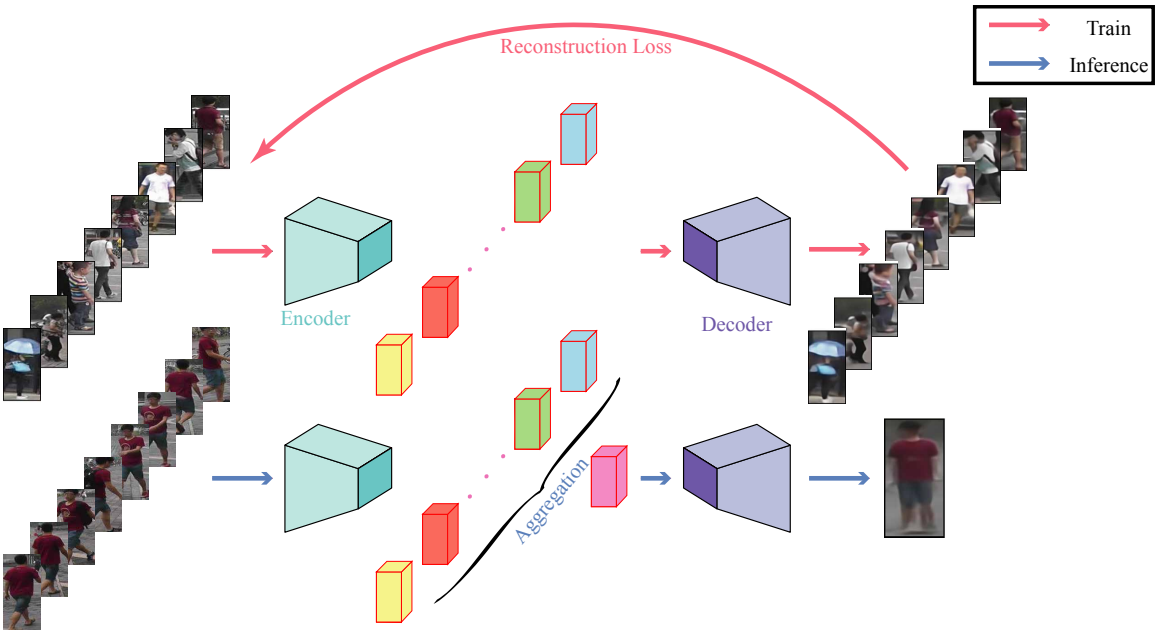


Figure 4.6: Reconstruction pipeline of anchors in image space: Encoder is transfered and frozen during the training of decoder. During the inference, the decoder reconstructs the anchor into image space.

into trained decoder (*c.f.* Fig. 4.6). We note that GAP applied anchor feature map produce the same as anchor feature vector and 2D maps are used to preserve the spatial information for better reconstruction. This anchor feature vector can be taken as the initial anchor during the start of training **Stage II**. As the results in Fig. 4.5, the anchors could be a feature vector which dissects the view-variance, pose-variance as well as background-variance. Comparing to the sample images from same class (*c.f.* Fig. 4.6), anchors generated from aggregation acts like a implicit regularization to remove noise and variance. It complies that the average aggregation distills the constitutional id-related features over the sampling distribution.

4.4.3 Further Discussion

Triplet Loss From the results in Table 4.3, our methods consistently improve the results over the original model no matter which variant is chosen. As in Fig. 4.7, our methods boost the generalization after the training **Stage I** converges, where the triplet loss reaches saddle point and there is still room to further intensify the cluster

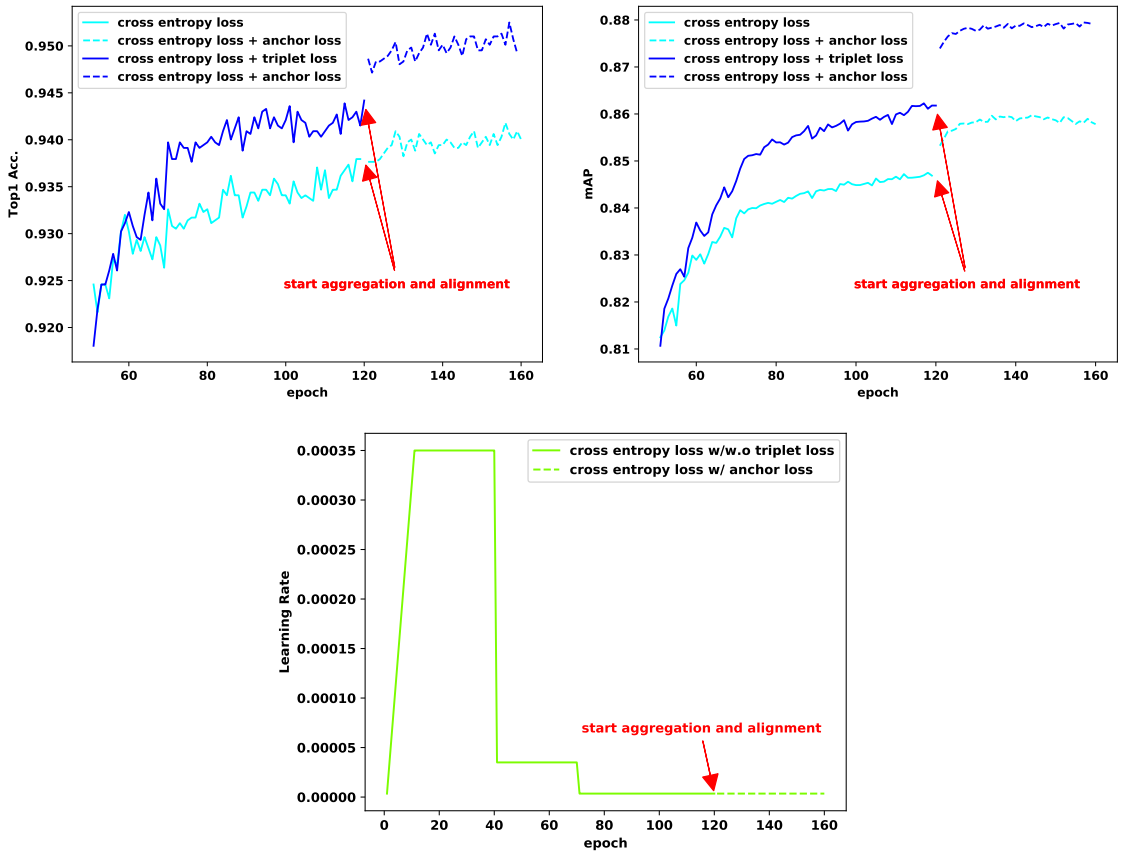


Figure 4.7: Test result of anchor loss from checkpoints: Without bells and whistles, anchor losses boost the performance significantly after the initial training is saturated.

compactness (Fig. 4.8 left). After the **Stage II** training, the intra-class distance is further reduced (Fig. 4.8 right) and boost the generalization in terms of both rank@1 accuracy and mAP (Table 4.3). As illustrated in Fig. 4.2 and the experiments in Section 4.4.1, our methods perform more effectively after the training **Stage I** is converged, where a stable feature distribution is provided for aggregation. On the other hand, triplet loss may provide beneficial effects during the initial stochastic training process. Triplet loss can inhabit more compacted feature embedding for each class in euclidean space than cross-entropy loss, which has been discussed in BNNeck [179]. Consequently, the improvement that the proposed anchor loss brings, is more significant for the model trained with L_{trip} than the one trained without L_{trip} (*c.f.* Table 4.3&Fig. 4.7). In summary, anchor loss stimulates stable and effective

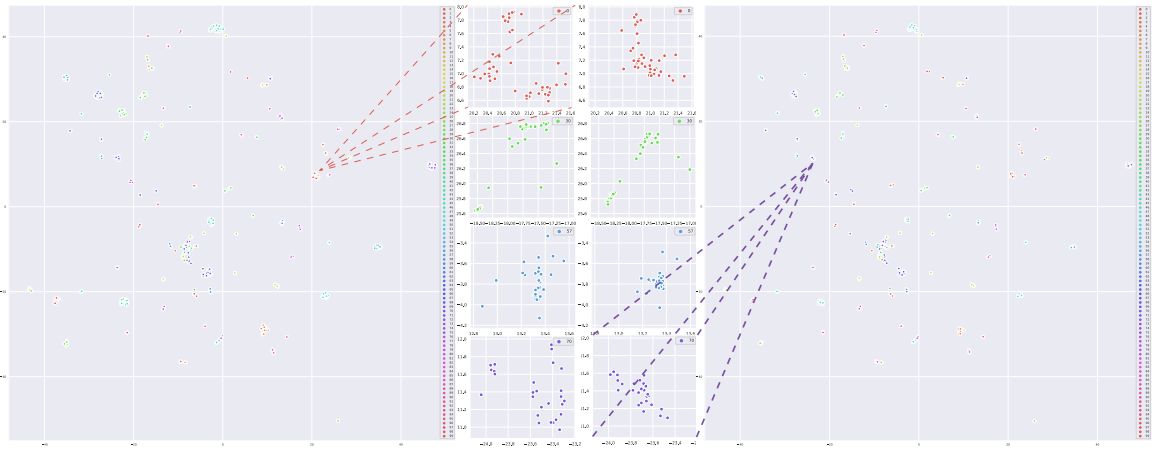


Figure 4.8: T-SNE visualization of the samples (ID=0,1,...99) on training dataset: Training result of Stage I with L_{trip} (left) v.s. Stage II with L_{anchor} (right). Zoom in to see details.

optimization to find better local optimal when triplet loss suffers from stochastic saddle point (*c.f.* Fig. 4.8).

Table 4.3: Ablation study of applying anchor loss after the initial training stage, *i.e.* epoch 120.

Frequency	Market1501		DukeMTMC	
	rank@1	mAP	rank@1	mAP
constant	95.34%	87.91%	88.3%	78.9%
epoch	**95.37%**	**87.99%**	88.3%	**79.1%**
iteration	95.25%	88.11%	**88.5%**	79.1%

Stage I	Stage II	aggregation method	Rank@1	mAP
L_{cls}	-	-	93.79%	84.69%
	$L_{cls} + L_{Anchor}$	avg	94.18%	85.98%
$L_{cls} + L_{triplet}$	-	-	94.42%	86.18%
	$L_{cls} + L_{Anchor}$	avg	**95.37%**	87.99%
	$L_{cls} + L_{Anchor}$	weighted	95.04%	87.95%
	$L_{cls} + L_{TripletAnchor}$	avg	95.25%	87.84%
	$L_{cls} + L_{TripletAnchor}$	weighted	95.13%	87.87%
	$L_{cls} + L_{Anchor} + L_{triplet}$	avg	95.16%	87.87%
	$L_{cls} + L_{TripletAnchor} + L_{triplet}$	avg	95.04%	87.94%
	$L_{cls} + L_{TtripletAnchor} + L_{triplet}$	weighted	95.28%	**88.07%**

Applicability In terms of the comparison about update frequency for anchors, three methods (Algorithm 2,Algorithm 3,Algorithm 4) derive comparable results as shown in Table 4.2. Considering the cluster formulation, the anchors generated at the end of traditional training are already well-complied with data distribution regarding their

Table 4.4: The performance of different models is evaluated on cross-domain datasets. Market1501 → DukeMTMC means that we train the model on Market1501 and evaluate it on DukeMTMC-reID. () denotes the models trained and tested with input size 384×192.

Methods	Market1501 → DukeMTMC		DukeMTMC → Market1501	
	Rank@1	mAP	Rank@1	mAP
Resnet50	27.9%(24.3%)	15.5%(13.0%)	47.7%(47.2%)	21.7%(21.1%)
Resnet50(ours)	35.3%(33.3%)	20.9%(18.6%)	49.6%(47.3%)	23.0%(21.8%)
Resnet50-ibn-a	40.7%(37.9%)	25.9%(23.2%)	56.0%(50.6%)	27.8%(24.5%)
Resnet50-ibn-a(ours)	46.1%(46.0%)	30.3%(29.1%)	55.3%(52.0%)	28.2%(25.2%)

identity labels (*c.f.* Fig. 4.8 left). After Stage II fine-tuning, the aggregated anchors stay close to the initial one since all the samples are pulled towards their anchors in the optimization (*c.f.* Fig. 4.8 right). From another perspective, those three methods in Table 4.2 are alternatives concerning computational cost and training availability while provide comparable result. For example, when training with large training dataset or in the context of online learning, Algorithm 2 and Algorithm 4 would be preferred with little sacrifice of performance. Hence, our proposed method could be tremendously flexible and widely applicable in terms of training efficiency.

Robustness A natural question about the **Stage II** fine-tuning is that whether the further cluster-level alignment tends to be domain dependent and overfit the domain distribution. From the cross-domain testing experiments in Table 4.4, our methods invariably outperform the baseline model. It verifies the proposed methods are an effective and robust approach to embed images into identity-related space for metric learning.

4.4.4 Non-Parametric Anchor vs Parametric Center

Although anchors in our work looks like the centers proposed in [178], they are intrinsically not the same: the former is non-parametric while the latter is parametric. In fact there is no essential difference between the center loss $\min_{\mathbf{c}_j} ||\mathbf{f}_i - \mathbf{c}_j||^2$ and the classification loss $\max_{\mathbf{p}_j}(\mathbf{f}_i^\top \mathbf{p}_j)$, both of which are distance metrics, *i.e.* L_2 distance and inner product. The role played by centers $\mathbf{c}_j$ [178] corresponds to the role of

the hyperplanes $\mathbf{p}_j$ in traditional classification. As a result, parametric center loss still conforms to instance-level alignment similar to classification loss under the **local** mini-batch training framework, and cannot build the **global** connection in cluster level. On the other hand, in the proposed anchor loss (Eq. (4.1)&Eq. (4.2)), anchors $\mathbf{a}_j$ are not optimization variables but calculated from cluster distribution instead. The anchors are iteratively updated from the aggregation of dataset features, which enables them to have the global view of feature distribution during the local mini-batch training.

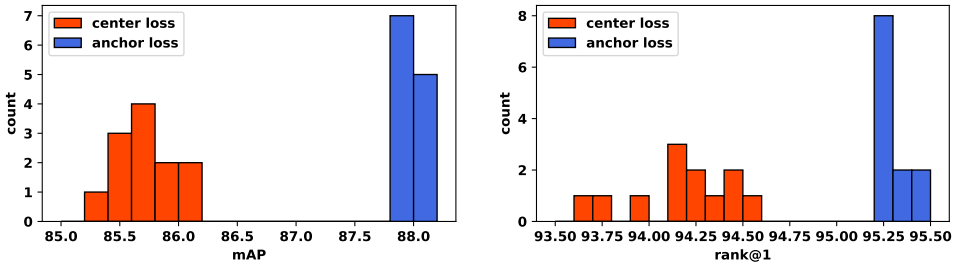


Figure 4.9: Test result comparison between our method and parametric center loss: We train 12 models independently for each method on Market1501 dataset.

We train 12 models independently using the the proposed anchor loss in comparison with another 12 independently trained models using the center loss [178] on Market1501 dataset. Fig. 4.9 illustrates the model performance histogram in terms of mAP (left) and rank1 (right). As can be observed, our proposed anchor loss consistently outperform those center loss [178]. It validates that anchor loss distills the knowledge in latent feature space from the images belonging to the same identity and aggregate them into anchors to guide the training towards well-aligned embedding. Such embedding complies intrinsic feature distribution and thus helps the both training and generalization. Furthermore, the result variance of parametric center loss is much higher than anchor loss, implying the dependency to random initialization for parametric center which may impose some stochastic prior to the cluster formulation. On the contrary, our methods consistently outperform center loss with small variance.

Table 4.5: Comparison with SOTA on CUHK03: CUHK03 evaluation with the setting of 767/700 training/test split on both the labeled and detected images. * denotes our implementation.

Methods	Labeled		Detected	
	Rank@1	mAP	Rank@1	mAP
MLFN(CVPR2018) [181]	49.2%	54.7%	47.8%	52.8%
PCB(ECCV2018) [91]	-	-	57.5%	63.7%
Mancs(ECCV2018) [182]	69.0%	53.9%	65.5%	60.5%
MGN [183]	67.4%	68.0%	66.0%	68.0%
Mltb [184](CVPR2019)	66.5%	70.1%	64.2%	66.6%
CASN+PCB [185] (CVPR2019)	73.7%	68.0%	71.5%	64.4%
MHN-4 (PCB) [186](ICCV2019)	75.1%	70.6%	71.6%	66.1%
Resnet50 [179]*	63.36%	61.60%	60.07%	51.79%
Resnet50(ours)	**76.36%**	**74.50%**	**72.36%**	**70.32%**

Table 4.6: Comparison of SOTA on Market1501 dataset and DukeMTMC-reID dataset. () denotes the results with a larger input size 384×192.

Methods	Venue	Market-1501		DukeMTMC-reID	
		Rank@1	mAP	Rank@1	mAP
Mancs	ECCV2018	93.1%	82.3%	84.9%	71.8%
PCB+RPP [91]	ECCV2018	93.8%	81.6%	83.3%	69.2%
VPM [162]	CVPR2019	93.8%	80.8%	83.6%	72.6%
AANet152 [173]	CVPR2019	93.9%	83.4%	87.7%	74.3%
IANet [187]	CVPR2019	94.4%	83.1%	87.1%	73.4%
MltB [184]	CVPR2019	94.7%	84.5%	85.8%	72.9 %
DG-Net [96]	CVPR2019	94.8%	86.0%	86.6%	74.8 %
MVP Loss [188]	ICCV2019	91.4%	80.5%	83.4%	70.0%
Auto-ReID [161]	ICCV2019	94.5%	85.1%	-	-
OSNet [189]	ICCV2019	94.8%	84.9%	88.6%	73.5%
MHN-6 [186]	ICCV2019	95.1%	85.0%	89.1%	77.2%
P^2-Net [160]	ICCV2019	95.2%	85.6%	86.5%	73.1%
BDB+Cut [190]	ICCV2019	95.3%	86.7%	89.0%	76.0%
ABD-Net [69]	ICCV2019	95.6%	88.3%	89.0%	78.6%
Resnet50 [179]	CVPRW2019	94.1%(93.6%)	85.7%(85.8%)	86.2%(86.9%)	75.9%(76.8%)
Resnet50 [179]	CVPRW2019	94.5%	85.9%	86.4%	76.4%
Resnet50(ours)	-	95.4%(94.9%)	88.0%	88.3%(89.1%)	79.1%(79.6%)
Resnet50-ibn-a [179]	CVPRW2019	95.2%(95.5%)	87.2%(88.2%)	89.0%(89.7%)	79.4%(80.0%)
Resnet50-ibn-a [179]	CVPRW2019	95.0%	87.2%	89.4%	78.8%
Resnet50-ibn-a(ours)	-	**95.7%(95.8%)**	**88.9%(89.7%)**	**90.2%(91.0%)**	**80.6%(81.8%)**

4.5 Comparison with the State-of-the-Art Methods

We compare our method with the recent state-of-the-art methods in Table 4.5 and Table 4.6. Comparing to spatial alignment by either attention or localization [91, 173, 187, 184, 160, 190], our method is much succinct without extra modules or classifier heads to handle subspace alignment. We have made several attempts

to incorporate spatial alignment in our baseline model, *e.g.* PCB [91], only to find slightly worse results. Based on our observation from reconstruction experiment (*c.f.* Fig. 4.6), we notice that decoder can be trained well to reconstruct images in both training and test dataset, implying spatially diversified features are preserved in feature maps before GAP (global average pooling). However, the feature vectors aggregated after GAP is aligned, meaning GAP could effectively eliminate the spatial variance while preserving the channel-wise diversity when a unified strong model is well trained, *e.g.* the strong baseline model[179]. Hence, we infer that the benefits of spatial alignment is marginal in our context. Comparing to channel attention and several variants [189, 186, 69] which aim to generate diverse and uncorrelated feature embedding with the efforts on convolution filters in self-supervised manner, our methods achieve the cluster encoding with focus on feature space by an explicit supervision of aggregated anchors from other images. DG-Net [96] endeavors to disentangle appearance code and structure code by GAN and auto-encoder, our method accomplishes similar effect that dissects the variance and preserves the identity-related features in the direction towards aggregated anchors (*c.f.* Fig. 4.5). IBN-Net [191] is proposed to reduce appearance variance and keep discriminative feature extraction by unifying both instance batch normalization and batch normalization. Luo *et al.* [179] apply it as the backbone network for person re-identification and we report our implementation result in Table 4.6. We note that resnet50-ibn-a network has the same parameter size and computational cost with original resnet50. Without bells and whistles, our method consistently boosts the performance in terms of both Rank@1 and mAP comparing to corresponding baseline (resnet50 and resnet50-ibn-a [179]), achieving the state-of-the-art results on Market-1501, DukeMTMC-reID(Table 4.6) and CUHK03(Table 4.5) datasets.. Specially, due to further reduce of intra-class variance towards a compact cluster in latent feature space, our method improves mAP significantly and benefits the robustness for the application of person re-identification.

In summary, comparing to the recent state-of-the-art methods, our methods provide a representation learning method with global invariance supervision, validating that exploration of interaction of clusters observed from dataset feature distribution improves both training and generalization.

4.6 Conclusion

In this chapter, we investigate the person re-identification from the view of alignment and find an interesting and effective approach to delve in invariance representation learning with a global supervision of cross-instance feature distribution. By performing aggregation and alignment iteratively, our proposed anchor loss is enabled to interact with more images indirectly from aggregated anchors, which pass the distilled knowledge from the feature distribution and provide a consistent optimization to further boost the performance significantly after traditional training convergence. It shows the representation learning with global invariance guided by the aggregation of dataset distribution, which steps beyond the general classification framework, is essential and beneficial for identity-related representations.

CHAPTER 5: COMPOSITIONAL REPRESENTATION LEARNING FOR COLOR ATTRIBUTE RECOGNITION[1]

5.1 Problems and Motivation

The study of color has a rich history. Many scientific breakthroughs were the result of color research, from Newton's theory in optics to Planck's black-body radiation law that started the quantum revolution in the early 20^{th} century. In the digital era, color is a fundamental attribute in almost all visual understanding systems, being especially useful for search, since color is one of the most used properties when describing an image. Digital image libraries like Google Photo host an enormous and ever-increasing number of images (10 billion images in 2010 to four trillion images in 2020) making the need for accurate and scalable training of color prediction models increasingly important for color-based indexing and search tasks. Although there have been many works dedicated to understanding different attribute types of objects in images [192, 193, 193, 193, 194, 195], color attribute prediction, has not received the same level of attention and in-depth analysis. In this work, we show the benefits of a joint language and vision pipeline for creating datasets to train new color prediction models that are scalable and use color-specific properties to improve color-recognition performance. We also identify and analyze subtle but important problems around color recognition, from the difference between human perception and language description to the skewed nature of color distributions for image objects.

We present analysis and results to draw more attention to the important, but often overlooked problem of color recognition and prediction, and set some interesting directions for future research.

[1]This work was done during an internship at Adobe Research

Figure 5.1: Illustration of color attribute prediction for detected objects in images.

Object color prediction is a well-constrained yet not-trivial problem. There are major difficulties in large-scale color prediction. First, color depends not only on human perception, but also on human language. Color perception by humans is surprisingly complex, ranging from color-shift due to luminance changes (*e.g.*, Purkinje shift [196]) to the unique human ability to maintain color constancy despite changes in the illuminant [196]. Human language concepts for colors are even more diverse i.e. people give different names to the perceived colors. Thus, training a model to predict colors that is in agreement with a majority of users is nontrivial. This clearly necessities the creation of a large-scale color dataset, which encompasses the distribution of colors as expressed by human language. Hence, following several recent works [197, 198, 199, 200], we use extracted color-object pairs from captions. We then ground each color-object pairs to a best candidate localized bounding box from an object detection model. Despite some noise due to either caption or object detection, this method corresponds well to human color annotations that we collected for test dataset and results in good predictions that align with the distribution of human colors. Through the relation study from three annotators and comprehensive experiments for color recognition, this work will enable future work to benchmark

the color recognition model aligned with visual perception and human textual color concept. The second major difficulty for color prediction is dataset bias. Most objects annotated will have common colors which makes the dataset biased toward common colors. In this work, we also propose a succinct and effective solution to remove the bias from the bias-trained model.

Color recognition has some unique properties compared to other attributes. First, unlike many attribute types such as actions and compositions, in general, color classes admit more consistent visual forms. Thus, it seems intuitive that we can inject our prior understanding of color into the recognition model by giving it a typical example (prototype) of each color. Furthermore, it has been shown that colors can be achieved through mixing and transforming several key colors. Thus, if we force the model to learn the relation between the object in question and the prior colors, the model can recognize the relative similarity between the color classes, and thus, may have better performance. We test this idea using an attention-based architecture, which significantly outperforms the baseline.

In this work, our contributions are as follows:

- We introduce a large-scale dataset for color attribute prediction consisting of $213,180$ auto-generated color-object pairs with grounding and fully human-annotated gold-standard test sets which benefits the analysis for understanding the color recognition and color description.

- We proposed a compositional representation learning module using some unique properties of colors that allow it to significantly outperform the baseline. The comparisons and analysis also open up potential future research directions.

- We propose a new method to solve data bias in color prediction, which leverages representations in the classifier, outperforming current state-of-the-arts methods proposed for long-tail recognition.

5.2 Related Work

Attribute Prediction Color attribute prediction is a subset of general attribute prediction which has been widely studied. [201, 202, 203] train typical multi-label classification models to predict the attributes. [204, 205] incorporate attributes with object labels in a multi-task training manner. Attribute recognition in various domains has been investigated for face attributes [192], people attributes [193], pedestrian attributes [194], action [195], person Re-ID [95], and *et al*. Therefore, attribute recognition is a fundamental problem to promote visual concept understanding. However, the recognition of color, which is one of the most common and important attributes, has yet be studied. More recently, Zero-shot compositional learning (ZSCL) is an important sub-area in attribute prediction research [101, 102, 103, 104, 104], the test set consists of compositionally novel object-attribute pairs that are not found in the training set and the algorithm must learn to decouple and compose objects and attribute to properly generalize to ZSCL setting. Although relationship and interaction learning between objects and attributes are important in ZSCL, we found that extra object information is not helpful for color recognition because color attributes are less object-dependent comparing to other attributes.

Memory Networks and Self-Attention Memory networks are applied in the NLP research for document Q&A [107, 16, 108] . where memorable information is separately embedded into keys (input) and values (output) feature vectors. Keys aim to address relevant memories whose corresponding values are returned. Recently, the memory networks have been applied to some vision problems such as personalized image captioning [109], visual tracking [110], and video instance object [111]. Another similar approach to our key color attention module is self-attention [37], which is a highly successful technique in multiple NLP benchmarks [112]. Self-attention is also widely applicable in several computer vision problems, *e.g*. video processing [113, 114, 111], image classification [115, 116], object detection [117, 118] and

vision-language tasks [119, 120, 121]. Different from existing methods, our attention module works by anchoring the attention prediction with reference to known color priors.

Long-Tail Recognition and Model Bias There are several advances for long-tail recognition during training stage, *e.g.*, re-sampling [206, 207, 208, 209] and category balancing losses [20, 210, 211]. More recently, post-training calibration methods have also shown success in long-tail recognition benchmarks [4] and unbiased scene graph detection benchmarks [5]. This direction has several advantages: (a) Agnostic to training paradigm; (b) Not dependent on any hypothesis about data distribution or optimization – based solely on analysis of the learned model; (c) Balancing performance of head and tail categories do not require model retraining Kang *et al.* [4] posit that the norms of classifier weights are correlated to the bias in model prediction, and propose *tau* normalization to classifier weights. Despite its success in several long-tail recognition tasks, it ignores the bias from feature extractors and our experiments show that the debiasing by τ normalization is sup-par on our long-tail color recognition tasks. Instead, our debiasing method considers the bias from both feature extractor and classifiers by utilizing the prediction from the bias-trained model. Tang *et al.* [5] report that the prediction of mean features from training data shows the prior bias from the trained model, and therefore, deducting the mean feature could effectively remove the bias. However, it is based on the assumption that the bias of the model is in line with training data distribution. Instead, our method calculated the mean features for each category that are more robust to the change of training data distribution.

5.3 Stock Color Dataset

5.3.1 Dataset Building and Annotations

As described above, color prediction is not a trivial task given the variance in human perception and language concepts. Thus, training a model that agrees with the

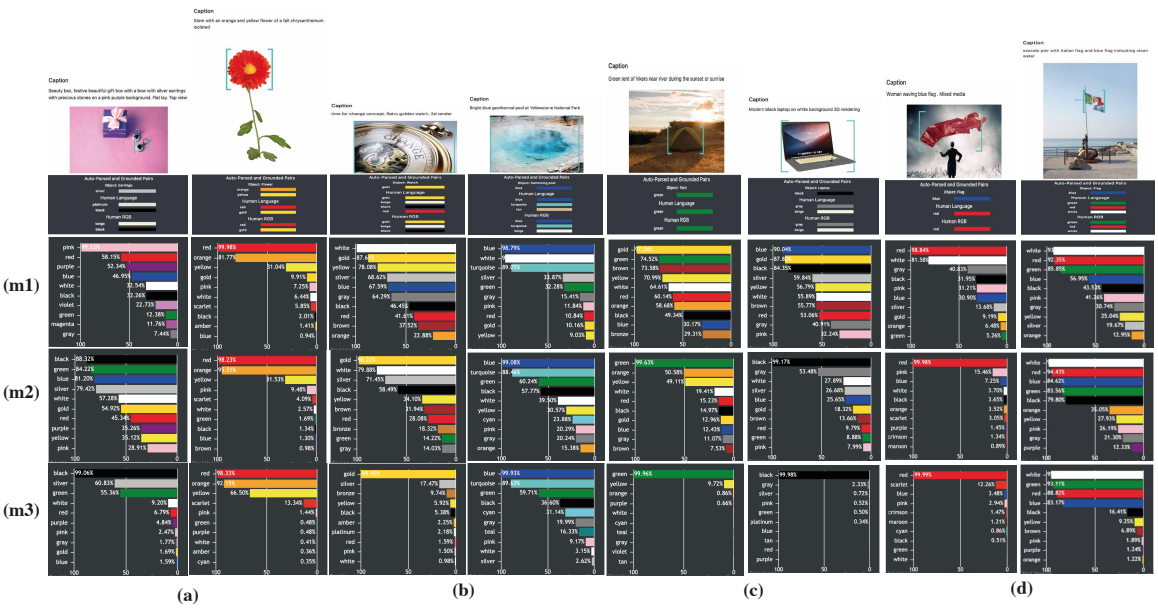


Figure 5.2: Comparison of different models and collected labels in test dataset of Stock Color: (m1) 1D DCNN (Color Histogram), (m2) Detection backbone network, (m3) Attention module; (a&b) Ambiguity of color attribute collection and illustration of language attention for color description, which differs in color lexicon and number of colors; (c) Color histogram failure cases: loss of contextual information (left) and semantic features (right); (d) Failure cases of auto-extracted labels due to either caption (left) or object detection (right). Three labels collected are illustrated under the image region: auto-extracted color and object (top), extra labels collected by human language description for referring objects (middle) and extra labels collected by human perception to referenced RGB color plates (bottom).

majority of users requires the use of large-scale datasets. Large-scale human annotation is cost-prohibitive, thus we need to design a scalable method of obtaining color object pairs. Recently, large scale image captions corpus such as Google's Conceptual Captions [212] have been used successfully to obtain a large amount of data to train joint vision language models. We follow the same path and use a language parser to get the object-attribute relations. The object is then grounded into the image using an object detection model, and the object-attribute pairs are filtered to contain only object-color pairs. The whole process depends on the quality of the caption, thus we rely on the Stock Captions dataset, which is our own large-scale per-image captioning dataset with precise and compact descriptions.

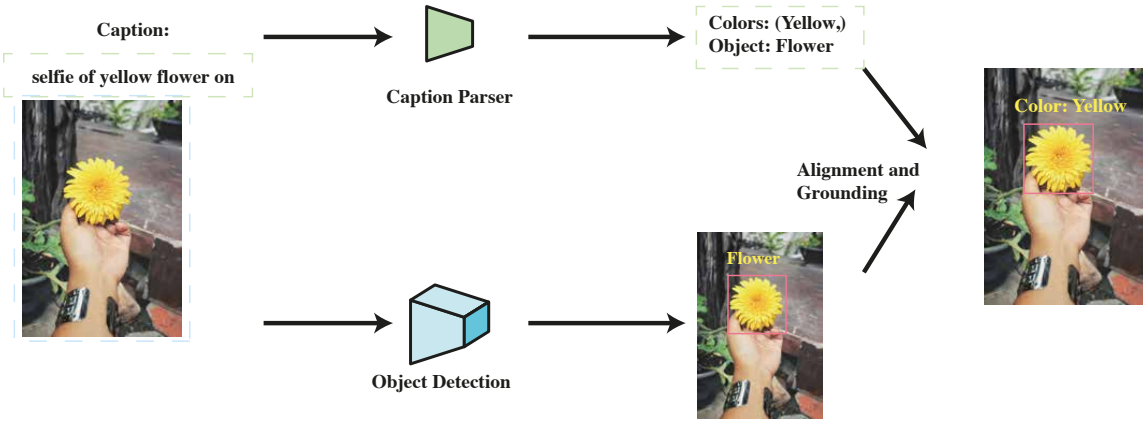


Figure 5.3: Illustration of grounding color attributes to detected bounding box

More specifically, we build a color attribute dataset by the following steps: (1) We collected $150M$ image captions from the Stock Captions dataset. We then further processed to extract $4.52M$ color-object pairs by our language parser; (2) An object detection model, namely EfficientNet-B3-FPN [32] pre-trained on Open Images dataset V4 [213], is used to localize objects from images; (3) Non-ambiguous grounding rules are applied to associate parsed color-object pairs with detected object bounding boxes, resulting in $219,911$ of bounding boxes with multi-color attribute and object labels from $219,033$ images. In the following, we illustrate the details for each step and demonstrate the final dataset.

Caption Parsing We use a basic NLP pipeline [214] to parse each caption into a computation graph structure [215] where candidate caption object nodes are extracted and then grounded to the corresponding image object bbox or mask. Each object node consists of zero or more attributes defined in an attribute type hierarchy. Type categories includes: color, material, shape, and texture. The NLP pipeline performs the following operations: Step 1: perform part-of-speech tagging, phrase chunking, and shallow semantic parsing to identify noun and verb phrases and object dependencies. Step 2: generate a computation graph from the semantic parse structure that identifies the objects and their attributes of interest. Step 3: Extract the descriptors for candidate objects from the computation graph and provide each

to the Color-Object grounding process described next.

Color-Object Grounding As shown in Fig. 5.3, the parsed objects from the caption are compared with detected objects from the image, resulting in the grounded color-object labels with detected bounding boxes. To make the grounded labels clean, we remove images where multiple multiple instances of the same object are detected. As shown in Fig. 5.2, some errors of the grounded labels are attributed to two factors: (a) The color description in the caption is wrong; (b) Object detection detects a non-referring object.

Extra Human Annotations As discussed in the earlier section, our dataset is constructed automatically by grounding detected objects with object names used in the caption and some errors are identified. Additionally, as discussed before, color description from different people differs in number of colors and color lexicon used. Caption-derived colors often include only one (main) color for multi-colored objects To solve this, we label all object instances in the test set using human-annotators without reference to auto-extracted labels. We require annotation of all *distinct* colors. Furthermore, color names applied to objects do not always one-to-one correspond with their canonical appearance. To study this, we ask annotators to not only label the color name for a given object but also select the closest color-hue as it appears in the image with reference to color plates. This annotation gives us color perception without language features. After the extra annotations, we obtain the following observations: (1) Three annotations differ in color lexicons (Fig. 5.2 (a)) and number of colors (Fig. 5.2 (b)); (2) Human annotations corrects some errors in auto-extracted labels (Fig. 5.2 (d)).

5.3.2 Evaluation

Dataset Splits In addition to the auto-extracted dataset, we also investigate color recognition on a subset of Visual Genome (VG) [216] dataset, where 30 most frequent color attributes and 150 object categories are selected. For VG dataset, we follow the

Table 5.1: Dataset Splits: * denotes bounding boxes with no more than two attributes.

	VG Color				Stock Color			
	train	val	test	total	train	val	test	total
Images	51,116	3,996	22,371	77,483	181,576	9,600	21,284	212,460
BBoxes	176,557	13,953	78,303	268,813	182,280	9,640	21,376	213,296
BBoxes*	170,793	13,692	76,143	260,628	182,178	9,635	21,367	213,180

same splits as Scene Graph Detection [217, 218, 219, 5], where 70% training and 30% test out of total 108k images are used. 5K validation images are held from training splits. For Stock Color, we select 27 colors which have at least 30 samples in the whole dataset and randomly split them into training (90%) and test (10%) subsets. Table 5.1 shows the final dataset we created for color recognition study.

Evaluation Metrics We formulate the color attribute prediction as a multi-label prediction problem, *i.e.* each object could have more than 1 color. Therefore, we evaluate model performance in a way of query and retrieval according to the ranking of predictions, where Precision@K and Recall@K are used. Suppose $\tilde{y}_i^{topK}$ are top K predictions for instance i in test dataset $\mathcal{T}$, and y_i are the ground truth colors, then Precision@K and Recall@K are:

$$\begin{aligned} Precision@K &= \frac{1}{|\mathcal{T}|}\sum_{i\in\mathcal{T}}\frac{|\tilde{y}_i^{topK}\cap y_i|}{|\tilde{y}_i^{topK}|} \\ &= \frac{1}{|\mathcal{T}|}\sum_{i\in\mathcal{T}}\frac{|\tilde{y}_i^{topK}\cap y_i|}{|K|} \end{aligned} \tag{5.1}$$

and

$$Recall@K = \frac{1}{|\mathcal{T}|}\sum_{i\in\mathcal{T}}\frac{|\tilde{y}_i^{topK}\cap y_i|}{|y_i|} \tag{5.2}$$

, where $|*|$ denotes the number of elements in the set.

Since 96.7% of instances from VG dataset and 99.9% of instances from Stock Color dataset contain no more than 2 color attributes we adopt Precison@1 and Recall@2 for the performance evaluation. We consider the mean recall of all the color categories as

Table 5.2: Annotation study on Stock Color Dataset: Oracle test takes the ground truth labels as the query to test the accuracy. Because extra human annotation is required to label colors with sequence respecting the priority, the sequence is adjusted in favor of tail colors to get better mRecall@1 and mRecall@2 during the oracle test.

	Caption (Auto-Extracted)		Human Language		Human RGB	
	Precision@1	Recall@2	Precision@1	Recall@2	Precision@1	Recall@2
Oracle Test	100%	99.99%	100%	96.97%	100%	97.29%
Caption (Auto-Extracted)	100%	99.99%	73.78%	61.37%	63.83%	53.20%
Human Language	64.31%	72.57%	100%	96.97%	80.60%	75.81%
Human RGB	56.75%	63.17%	82.58%	76.04%	100%	97.29%
	Caption (Auto-Extracted)		Human Language		Human RGB	
	mRecall@1	mRecall@2	mRecall@1	mRecall@2	mRecall@1	mRecall@2
Oracle Test	99.35%	99.99%	82.81%	97.23%	83.70%	97.63%
Caption (Auto-Extracted)	99.35%	99.99%	26.60%	26.95%	24.14%	24.46%
Human Language	32.79%	40.52%	58.38%	86.53%	34.12%	49.70%
Human RGB	29.37%	35.07%	36.82%	53.91%	60.88%	89.92%

the evaluation metrics for long-tail color recognition, imitating category-balanced test settings. Specifically, the category-balanced variation can be derived from Eq. (5.2) for the total color category set $\mathcal{C}$ as:

$$mRecall@K = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \sum_{i \in \mathcal{T}} \frac{\delta(c \in y_i)}{\sum_{i \in \mathcal{T}} \delta(c \in y_i)} \frac{|\tilde{y}_i^{topK} \cap y_i|}{|y_i|} \tag{5.3}$$

where $\delta(c \in y_i) = 1$ if the color category c in y_i and otherwise $\delta(c \in y_i) = 0$. And mRecall@1 and mRecall@2 are used for evaluation.

5.3.3 Annotation Study

We evaluate the model on our dataset with three labels: auto-extracted labels from the caption, extra human language annotation, and extra human RGB color with reference to color plate. In order to test the agreement among three labels, we evaluate the accuracy using different combinations of labels as prediction and ground truth. As shown in Table 5.2, we can see that there is disagreement among three labels, which are in line with our analysis that the number of colors selected and the color lexicon used can vary from different annotations (*c.f.* Fig. 5.2 (a&b&c)). Furthermore, we can further conclude that the ambiguity for color attribute description lies in additional

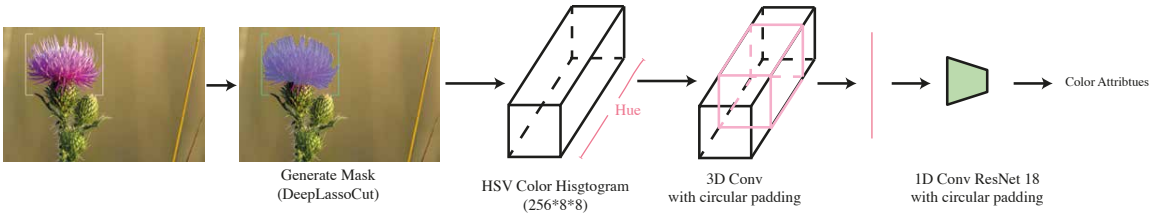


Figure 5.4: Baseline: 1D Conv network for predicting color attributes from 3D color histogram.

two folds: (a) The top1 picked color from extra human language annotation dissents largely to auto-extracted labels from caption (64.31% precision@1). Since we annotate the extra human language labels with sequence respecting the priority, the dominant colors picked from extra human language annotation, *i.e.* top1 color, do not match well the caption-driven description from our auto-extracted labels. Besides, another potential reason is due to different task purposes when describing colors. It could be attributed to the distribution divergence between caption-driven color description and attended-color description. Caption-driven color descriptions could be served for different and diverse purposes. (b) On the other hand, due to more completeness of color description from extra human language annotation, Precision@1 is better when using caption labels to test against human language labels. It suggests that this annotation could be more reliable and useful in terms of testing the generalization of colors since more candidates are chosen.

5.4 Benchmarking

Vision Features and Object Labels For color prediction given the localization (bounding box) and referring object category, the vision features can be extracted from image I_i and localization bounding box loc_i for instance i. Supposing referring object label is obj which is given for additional contextual information, the final prediction $\tilde{y}_i$ in a query ranking sequence is denoted as:

$$\tilde{y}_i = \{\tilde{y}_i^c\}_{c \in \mathcal{C}} = f(I_i, loc, obj)$$

. If only use the *obj* for the prediction and ignore the vision features, a reasonable prediction matches the color distribution for the object category *obj* within the training dataset. This test result, named language only test from Table 5.3, provides a naive baseline for our color attribute prediction. Because Stock Color dataset contains more diverse objects (321 object categories) and the samples in each category could be sparse, this language-only model performs better in VG Color dataset where there are 150 object categories.

Vision Baseline 1: Color Histogram Because color prediction is related to low-level features and color histogram could effectively preserve such low-level features, it is interesting to begin with a baseline experiment over 3D HSV color histogram (*c.f.* Fig. 5.4). We adopt $256 * 8 * 8$ HSV bins by preserving values on Hue channel which contains more color-related information and grouping Saturation and Value channels into 8 bins respectively. To remove the background of the detected bounding box, we first use instance segmentation to predict the object masks given the bounding box and then perform the color histogram calculation over the mask region. A 2D Conv with kernel size $3 * 8 * 8$ is applied to aggregate histogram features into 1D vector and then a Resnet-18 model, where all 2D Conv kernels are replaced by 1D Conv kernels with symmetrical padding, is used to extract the features. We note that a fully connected layer instead of the global pooling layer is connected before the classifier layer since spatial location along Hue channel is important for color recognition.

As shown in Table 5.3, this baseline model results in 63.92% and 68.52% precision@1 on VG Color dataset and Stock Color (Caption) dataset respectively. Since the segmentation model used is never trained on two datasets, there are some errors for extracting the object masks. We experimented on calculating the color histogram without segmentation masks, observing worse precision@1 (around 5%) than incorporating masks. Taken some errors from the masks, it shows that the 1D CNN model could effectively learn the color category information from the color histogram.

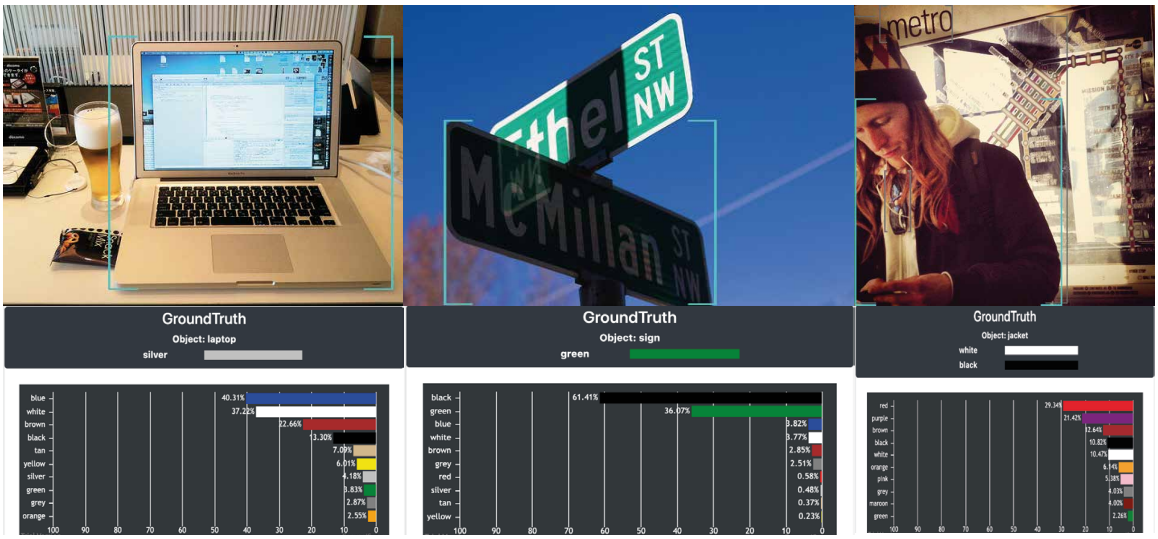


Figure 5.5: Common failure cases of color histogram baseline: It demonstrates the constraints of predicting colors from color histogram due to either loss of semantic information (left) or contextual information, *e.g.* environmental lighting (middle) and photography filter (right)

Table 5.3: Benchmarking Results: VG Color and Stock Stock

	VG Color		Stock Color (Caption)	
	Precision@1	Recall@2	Precision@1	Recall@2
Oracle Test	100%	97.24%	100%	99.99%
Language Only	46.84%	59.43%	39.97%	57.01%
Resnet18 1D Conv (color histogram)	63.92%	74.51%	68.52%	83.01%
Faster RCNN ResNext101 FPN	75.35%	83.40%	81.85%	91.59%
+ Weighted BCE Loss	76.60%	84.72%	82.45%	91.69%
+ Detection pretrain w/ attribute loss	78.83%	85.79%	83.52%	92.53%
Task modules w/o object embedding gating [102]	78.36%	85.59%	83.26%	92.47%
Task modules w/ object embedding gating [102]	78.39%	85.57%	83.00%	92.37%
Key-Color Attention	79.11%	86.31%	85.19%	93.17%

However, it has two major downfalls: (a) The loss of semantic information: Color attributes are affected by human visual attention and language attention, both of which are related to semantic information. As shown in Fig. 5.5 left, we naturally describe colors of chassis instead of the screen when referring to the laptop. (b) The loss of contextual information: The color histogram from object region could easily vary as the environmental lighting (Fig. 5.5 middle) and photography filter (Fig. 5.5 right) change. However, humans tend to describe the original colors of objects despite the change of those factors.

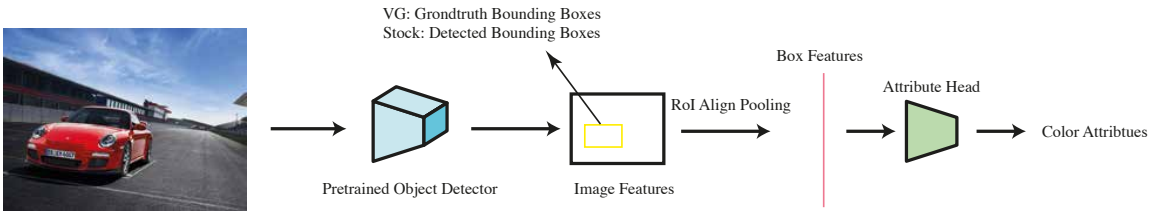


Figure 5.6: Pipeline of detection backbone for color attribute prediction

Vision Baseline 2: Object Detection Backbone As analyzed above, color attributes prediction should preserve semantic features and global contextual features. Therefor, we use Faster R-CNN PFN [15] with ResNext101 backbone [31], which are widely used for Scene Graph Detection [217, 218, 219, 5] and Visual Question Answering [119, 120, 121] to extract object features given the groudtruth bounding boxes. In order to extract features that are object-aware and background-excluded, the detection model is pretrained on VG (Visual Genome dataset [216]) using the same training and test splits of attribute prediction task. For multi-label classification, we use binary cross entropy as our loss function. Due to the human description attention from either VG annotation or Stock captions, some negative labels are not necessarily negative but just not annotated. Hence, we adopt a weighted binary cross entropy loss to emphasize the positive labels. In addition, other attributes (*e.g.* texture, materials) could be correlated to colors, we pretrain the object detection model with attribute loss, where the 400 attributes more than colors in VG dataset are used. Following the widely used pretrain manner from VQA tasks [220, 119, 120], the attribute prediction branch concatenates the box features and Glove embedding of object labels. As shown Table 5.3, pre-training with other attributes, which could be potentially correlated to color attributes, benefits the presentation pretraining of feature extractor for color prediction. Moreover, it could be an effective label augmentation method for robust pretrained feature extractors especially when the color labels used for training are noisy.

Additional Object Labels Besides the vision features provided, *i.e.* image and

Table 5.4: Benchmarking Results: tested on cross-domain annotations

	Stock Color (Human Language)		Stock Color (Human RGB)	
	Precision@1	Recall@2	Precision@1	Recall@2
Oracle Test	100%	96.97%	100%	97.29%
Language Only	41.25%	45.71%	36.61%	40.55%
Resnet18 1D Conv (color histogram)	67.08%	68.63%	59.26%	60.20%
Faster RCNN ResNext101 FPN	75.86%	74.03%	65.61%	63.84%
+ Weighted BCE Loss	76.03%	74.05%	65.64%	63.95%
+ Detection pretrain w/ attribute loss	76.40%	74.58%	65.93%	64.34%
Task modules w/o object embedding gating [102]	76.17%	74.60%	65.81%	64.50%
Task modules w/ object embedding gating [102]	76.03%	74.42%	65.69%	64.33%
Key-Color Attention	75.91%	73.53%	65.44%	63.89%

the object localization, it is attempting to see the impacts of additional object labels. Following the success of constructing dynamic gating modules according to the object embedding in zero-shot attribute object compositional learning [102], an attention module with a gating network is added before the final classifier layers. Specifically, we use 3-layer gating modules, each of which contains 24 gating layers. As shown in Table 5.3, the performance is on par with the baseline. Moreover, for better comparison for additional object labels, a counterpart with the same task modular but without the object gating network is tested, *i.e.* adding additional fully connected layers, which is comparable to the task modules with object gating network. Different from [102], the detection model is pretrained on VG dataset and should contain the object-related features. Besides, color attributes are much less correlated with object categories comparing to other attributes, *e.g.* materials, textures and actions. For instance, standing, sitting, and walking attributes could only be applied to animals and humans. On the other hand, color is more universally applicable though it definitely could have an object-related distribution as shown in the language-only test.

Final Result Analysis From Table 5.3, the model trained from caption-parsed labels, could generalize well for human language and human RGB annotations, even outperforming the test using groundthruth caption labels (*c.f.* Table 5.2). The human language annotations for colors are more attended, accurate, and completed,

while the caption-parsed labels are more short and noisy. Taken that our labels are auto-extracted from the captions and the cost of label collection could be neglected, it shows the promising results towards the learning with a reconciliation between human color annotations and color description of captions. Although the features of color histogram model could be more aligned with human RGB annotations, color histogram model is trained with parsed-caption labels, which digresses from human RGB annotations. Thus, it is still inferior to the detection model.

5.5 Color Compositional Learning Through Attention

Most colors can be represented by sets of coordinates in key colors space. So, it should be possible and beneficial to force the model to produce its prediction based on the relations between the extracted features and key reference colors. We call it as *color composition hypothesis*.

To test the hypothesis, we design a module that can learn the relation between key/prior knowledge examples and the query object for color recognition. One intuitive way to achieve this is through the attention mechanism similar to memory networks which is widely used in Q&A [107, 108]. Different from memory networks in Q&A where the keys are changed respecting input reference documents, our method attends to the color prior keys which are intentionally plugged in. Those prior keys serves as primitive visual color words in latent feature space, which behaves like primary colors in RGB space. The specific implementation and analysis are detailed in the following subsections.

5.5.1 Color Composition Module

In our experiments, we choose all the colors in the target dataset as it might span a decoupled space in agreement with the classifiers. To match the colors to the features from the images, however, we need to transform the color to the feature space. To do this, we consider two approaches to extract the feature vectors for the key colors:

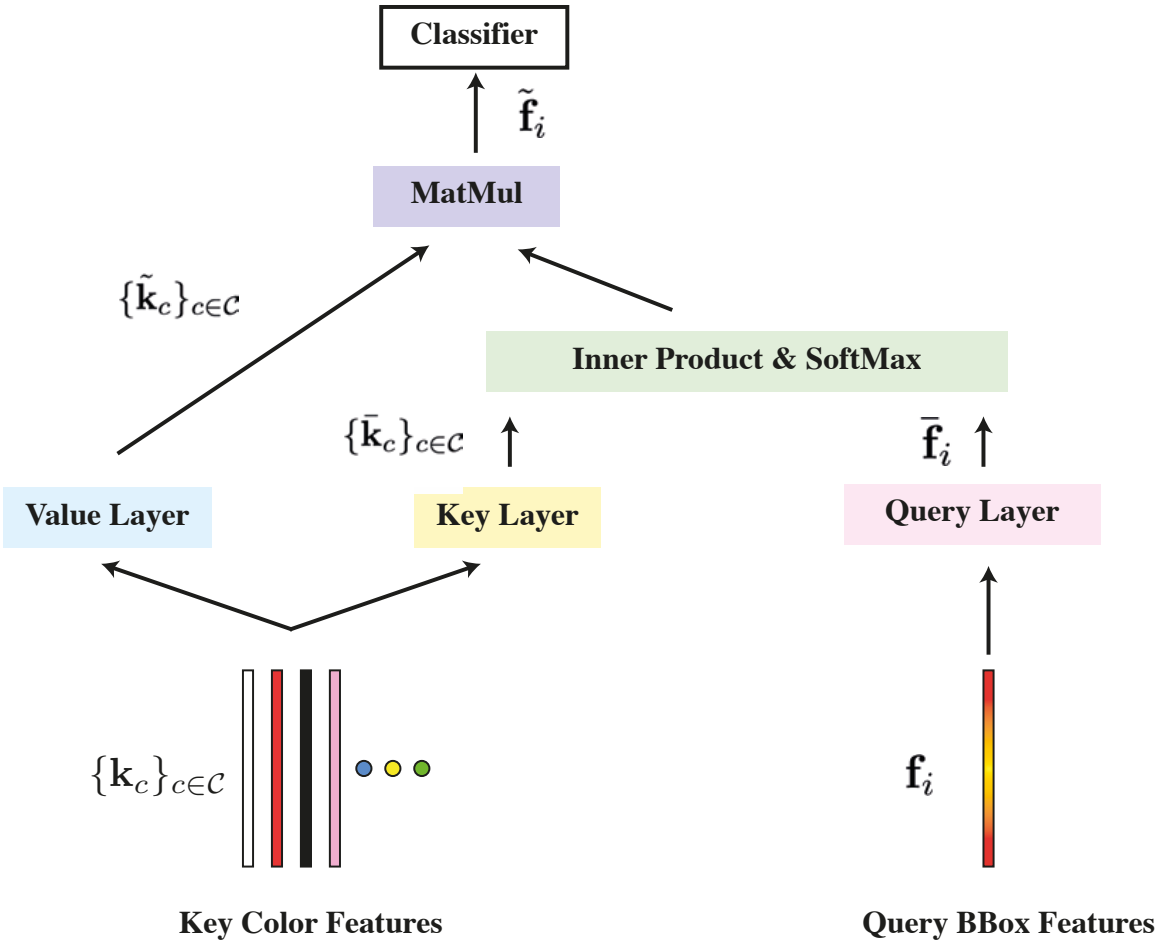


Figure 5.7: Attention module with references to key color features: The computation between query object feature $\mathbf{f}_i$ and key color features $\{\mathbf{k}_c\}_{c\in\mathcal{C}}$ is illustrated in Eq. (5.5). Each key color feature $\mathbf{f}_i$ denotes one key color corresponding to the color c in the total colors of prediction $\mathcal{C}$. Value, key and query layer are linear transformations with learning parameters.

(a) Pure RGB color images: According to the definition of color names and their corresponding RGB values, the images filled with monotonous RGB values and a center bounding box are fed into the feature extractor to obtain the primary key color features; (b) Mean features of sampled bounding boxes: We extract the features from the sampled bounding boxes with the same color attribute and calculate the mean of those features for the representation of respecting key color. In the implementation, to save graphic memories and avoid the turbulence from sampling, we adopt the

Table 5.5: Ablation study for primary key colors on Stock Color dataset: * Denotes degeneration, *i.e.* model always predicts most common colors, white and red, in top 2 predictions. The key colors are all 26 colors from Stock Color, and the representation keys are extracted by random initialization, features from monotonous RGB images, and mean features from the corresponding color labels.

	Learnable	Precision@1	Recall@2
Pure RGB images	✗	85.19%	93.17%
	✓	85.00%	93.08%
Category mean features	✗	85.03%	92.95%
	✓	84.10%	92.28%
Random keys	✗	23.88%*	38.69%*
	✓	23.88%*	38.69%*

calculation as the update with momentum λ:

$$\mathbf{k}_c = \begin{cases} (1-\lambda)\mathbf{k}_c + \lambda \mathbf{f}_i, & \text{if } c \in y_i \\ \mathbf{k}_c, & \text{otherwise} \end{cases} \tag{5.4}$$

where k_j denotes the color feature key for attribute color c and is initialized from all training features with respecting color labels extracted from the pre-trained detection model before the training. With the key color representations defined, we design the model using the attentions as:

$$\tilde{\mathbf{f}}_i = \sum_{c \in \mathcal{C}} \frac{exp(\mathbf{f}_i^T \bar{\mathbf{k}}_c)}{\sum_{c \in \mathcal{C}} exp(\mathbf{f}_i^T \bar{\mathbf{k}}_c)} \tilde{\mathbf{k}}_c \tag{5.5}$$

where $\bar{\mathbf{k}}_c$ and $\tilde{\mathbf{k}}_c$ are transformed keys and values from reference prior keys $\mathbf{k}_c$ by linear key layer and value layer respectively (*c.f.* Fig. 5.7,). Following the success of transformer [37], we use multi-head attention to calculate contributions of key color features in 16 grouped features.

From Table 5.6, we can see that the attention module improves the results from the baseline, *i.e.* the final detection baseline model. As shown in Table 5.5, pure RGB color keys outperform mean feature variation, which is in line with our motivation

that the simplicity benefits the compositional learning from color keys. Succinct and primitive color keys (*i.e.* pure RGB key colors) decoupled the variations and noises from other factors (*e.g.* lighting and filtering) and provide better representation learning through the aggregation in the attention module. Although simplicity matters in the compositional learning from color keys, the meaningful embedding, and proper priors are crucial. The model can degenerate if using the random initialized color keys (*c.f.* Table 5.5).

Learnable Key Colors. To save the computations for feeding reference images in monotonous RGB values during every training iteration which consumes large graphic memory for the detection network, we fix the key color vectors by extracting them from a pre-trained network. Since most feature learners are not designed to handle pure color images and it might result in sub-optimal, we explore variants of the model using learnable keys. In specific, we initialize key color vectors with the calculated values as illustrated above and update the key color vectors by backpropagation along with the training. From Table 5.5, one can see the learnable version of key colors are comparable to the fixed version, implying that the transformation of attention is powerful enough to learn a good embedding for prior key colors. Besides, since the key layer transformation is linear and sharable across all keys, the gradients through the key layer might guide the keys moving close to each other and harm the diversity of the key colors.

5.5.2 The Mechanism of Attention

Compositional Representations by Diverse Attention From Figure 5.8 left, it is clear that the attention hike does not correspond to target classes. Instead, the activation on the attention of keys diversifies in two folds, group-wise diversity in different attention heads and sample-wise diversity in different query images. With deeper analysis, we add additional auxiliary key loss to reinforce the color concept learning in the attention module. Specifically, we get the each key color prediction by

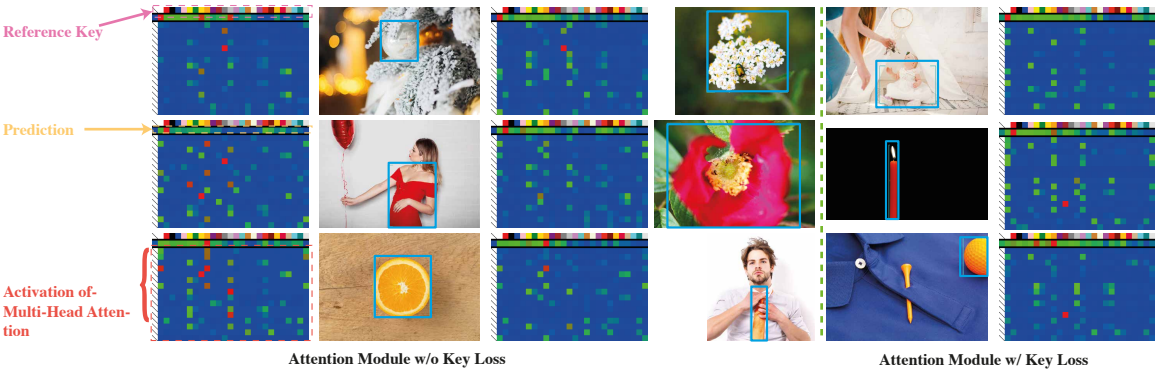


Figure 5.8: Visualization of sampled heatmaps from attention and prediction: The first line illustrates the RGB value of the reference key colors. The second line denotes the prediction including the background in first element. The rest of lines shows the heatmap from multi-head attention modulue to each corresponding key color.

Table 5.6: Ablation study for the number of key colors: The key color features are extracted from the corresponding pure RGB color images. Head and tail categories are defined according to the sample distribution from the whole dataset, *i.e.* head categories are the most common categories.

Colors	Number	Precision@1	Recall@2
All categories	26	85.19%	93.17%
head categories	8	84.84%	93.11%
tail categories	8	85.06%	92.75%
head categories	3	85.03%	93.08%
head categories	1	22.38%*	38.69%*

using the key feature k_c as query feature in Fig. 5.7 and Eq. (5.5). Then the key loss with its corresponding key colors is added. As result in Fig. 5.7 right, the attention resembles a mixture of colors still. Hence, we conclude that with our design, the model will automatically learn compositional representation, even with the guidance of the key-loss. As a result, it effectively probes all the provided prior key colors and transforms query box features into a better representation that benefits color recognition by diverse attention.

Cardinality of Prior Color Space From Table 5.6, we can see that different numbers of key colors chosen (3, 8, and 26 colors) result in comparable results. Moreover, the choices of the key color set (head categories and tail categories) do not affect the

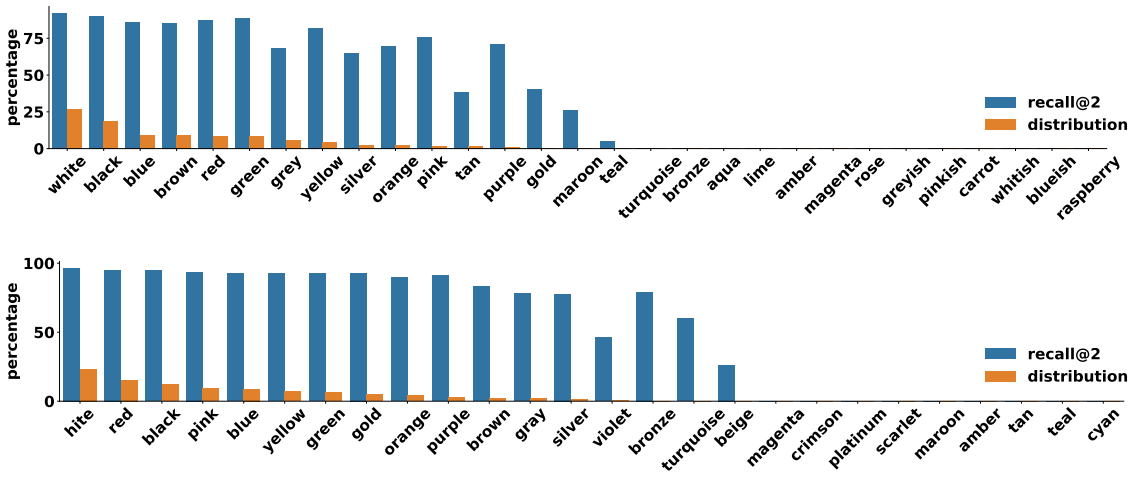


Figure 5.9: Dataset category distribution and the results from benchmark model for VG Color dataset (top) and Stock Color (bottom).

performance very much. We infer that the cardinality in prior color spaces, *i.e.* the number of keys composing the attention, is small in order to learn effective representation for color recognition. In addition, we note that multi-head attention with diverse activation would augment the cardinality in prior color space.

Future Work In summary, we exhibit an attention module based on the *the color composition hypothesis* and find it could effectively improve the performance for color recognition by a diverse activation that effectively interacts with prior keys. Through some experiments and analysis, we conclude that the simplicity benefits the attention module in two folds: (a) The features of a pure RGB image are effective which serves as primitive visual words to generate diverse representation; (b) The cardinality required in the multi-head attention module for effective learning is small. In this work, we target color recognition and the further extensions of this attention in other tasks will be left for future work.

5.6 Proposed Method for Long-Tailed Color Recognition

Due to imbalance distribution from the dataset, we found many tail-colors are never predicted in the top2 predictions, *i.e.* recall@2 for those colors are zeros (*c.f.*

Fig. 5.9), which is not acceptable for most applications.

We consider mean recall (*c.f.* Eq. (5.3)) of all the color categories as the evaluation metrics for long-tail color recognition, imitating category-balanced test setting. Due to the label distribution discussed in Section 5.4, we take mRecall@1 and mRecall@2 for evaluation.

5.6.1 Bias in Classifiers

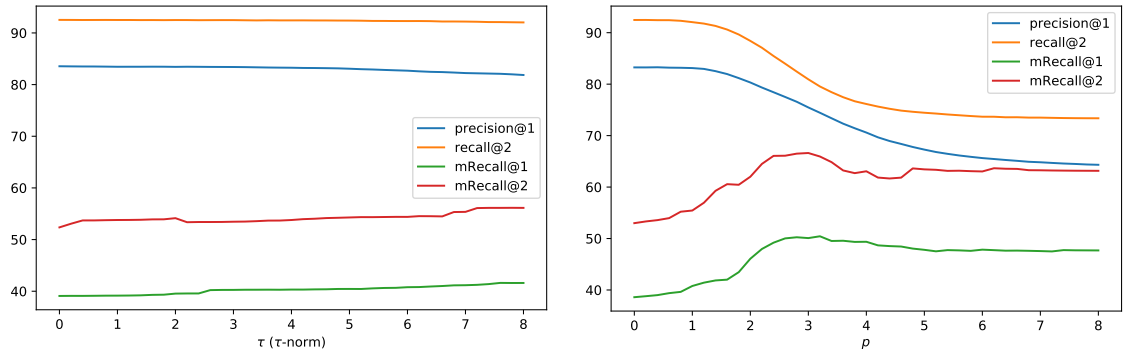


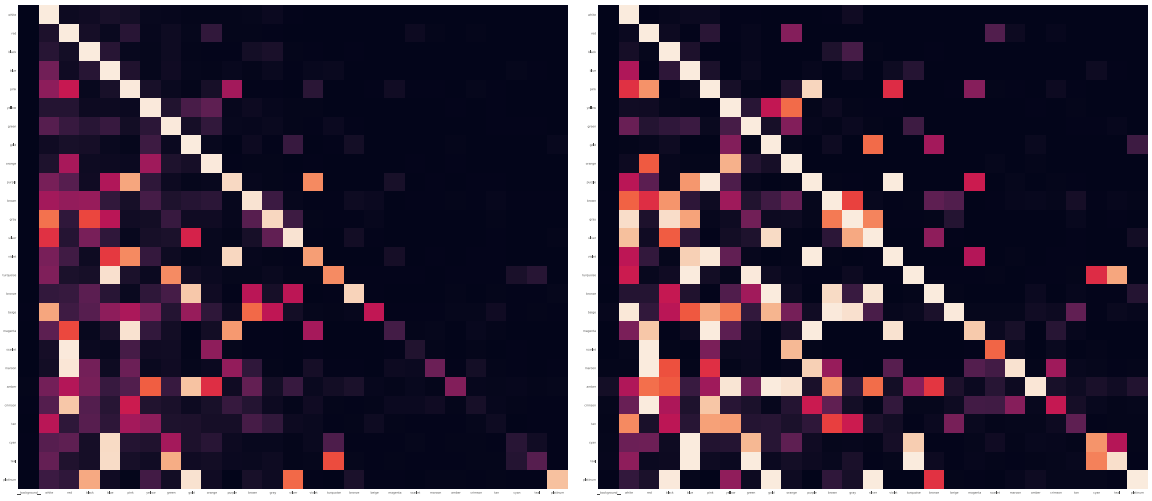
Figure 5.10: Comparison of removing bias on Stock Color: Plots shows the model evaluation results in validation dataset as the change of debiasing factor *tau* for τ-norm [4] (left) and p for ours (right).

Kang *et al.* [4] found the bias lies in the classifier and normalization for the classifier weights $\hat{\mathbf{W}}$ could effectively be used to remove the bias for long-tail recognition. Specifically, it scales each row $\mathbf{w}_j$ in the classifier weights by:

$$\hat{\mathbf{w}}_j = \frac{\mathbf{w}_j}{||\mathbf{w}_j||^\tau} \tag{5.6}$$

where τ is a controling debias factor. And then unbias prediction for sample i becomes $\hat{\tilde{y}}_i = \hat{\mathbf{W}}\mathbf{f}_i$.

Despite some effects for τ-norm, the mRecall values saturate even when the τ value reaches high value (*c.f.* Fig. 5.10), which is different from the peak phenomenon from three long-tail recognition tasks in [4], where the optimal *tau* values were found in [1, 2]. It shows that removing the bias only from the classifier solely can not effectively



Stock Color (before TDE) Stock Color (after TDE)

Figure 5.11: The prediction of mean features in the training Stock Color dataset and the illustration of co-activation after removing the bias in features by TDE [5]: Each row denotes the model prediction (in column) by feed the mean features of all training data in each belonging to the same category (left). After applying TDE [5] to each prediction, it activates the corresponding category while co-activating the correlated head colors (right).

remove the bias for our color. Kang *et al*. [4] found the bias lies in the classifier and a normalization operation for the classifier weights $\hat{\mathbf{W}}$ could effectively be used to remove the bias for long-tail recognition. Specifically, it scales each row $\mathbf{w}_j$ in the classifier weights by:

$$\hat{\mathbf{w}}_j = \frac{\mathbf{w}_j}{||\mathbf{w}_j||^\tau} \tag{5.7}$$

where τ is a controling debias factor. And then unbias prediction for sample i becomes $\hat{\hat{y}}_i = \hat{\mathbf{W}}\mathbf{f}_i$.

Despite some effects for τ-norm, the mRecall values saturate even when the τ value reaches high value (*c.f*. Fig. 5.10), which is different from the peak phenomenon from three long-tail recognition tasks in [4], where the optimal *tau* values were found in $[1, 2]$. It shows that removing the bias only from the classifier solely can not effectively remove the bias for our color.

5.6.2 Bias in Trained-Model, Entangled Features, and Co-Activation

Based on TDE (total direct effect), Tang *et al.* [5] propose to remove the bias by the prediction of mean features from the whole training dataset. They achieve state-of-the-art in unbiased scene graph detection task by identifying the bias in object features. In the context of color recognition, the bias might lie in box features before the classifier. Thus, we applied TDE [5] to the prediction of mean features for each category k, which provides representative outputs despite the hard samples in the training dataset. Supposing $\mathbf{f}_i$ is the feature vector of the instance i and the prediction of mean feature in color category c is:

$$\tilde{y}^c = \mathbf{W}\frac{\sum_{i\in\mathcal{T}}\delta(k\in y_i)\mathbf{f}_i}{\sum_{i\in\mathcal{T}}\delta(k\in y_i)} + \mathbf{B} \tag{5.8}$$

where $\mathbf{W}$ and $\mathbf{B}$ denotes the weights and bias in classifier layer. And after removing the bias, the prediction of mean feature is:

$$\hat{y}^c = \tilde{y}^c - (\mathbf{W}\frac{\sum_{i\in\mathcal{T}}\mathbf{f}_i}{|\mathcal{T}|} + \mathbf{B}) \tag{5.9}$$

We note that this would be equivalent to calculate the mean predictions for all the samples belonging to the same category since both of the classifier and averaging operation are linear.

From Fig. 5.11 left, the mean features of tail colors are not activated well to the corresponding ground truth. It implies that those tail-color samples in the training dataset are underfitted under normal instance-balanced sampling and the bias in the training dataset dominates the model prediction. After removing the bias by TDE, it successfully removes the bias in some degree and activates those tail colors (*c.f.* Fig. 5.11 right). As shown in Fig. 5.11, each row denotes the prediction of the corresponding color feature after removing the bias from TDE. And the features still tend

to co-activate some correlated head colors besides themselves. It implies the entangled features and co-activation between head colors and tail colors. Similar to the tail distribution of the relation in scene graph detection, the tailed color distribution is also correlated to some head colors due to the language attention. For example, the head relation "on" is highly correlated to tailed relations, "sitting on", "walking on", "standing on". And head color "red" is linked to the tail color "scarlet". Moreover, some samples in head colors could be described to the corresponding "tail" color if different language perference or different people's comments are used. Even after removing the bias, such co-activation and feature entanglement can not be effectively reduced by only removing the bias in features.

5.6.3 Compositional Representations in Classifier

Besides the co-activation illustrated above, such co-activation is asymmetrical and bias towards head colors, *i.e.* the majority of activation lies in the lower-left areas (*c.f.* Fig. 5.11). If taking the prediction of the output layer as a sparse encoded feature, it demonstrates that the model tends to describe the colors using head colors, which has been effectively trained from the sufficient data. Moreover, the mean prediction is sparse and separable, which could be effectively used as centers for distance-based classification. Based on the observation above, we take the prediction as the feature and calculate the distance to each center c for the prediction of color c as:

$$\hat{w}_i^c = -||\tilde{y}_i - \tilde{y}^c||^p \tag{5.10}$$

where p is a debiasing factor which is L2 distance when $p = 2$ and the $||*||$ denotes the norm. Then use unbiased prediction above to off the original prediction to get the final prediction as:

$$\hat{y}_i = \{\tilde{y}^c + \hat{w}^c\}_{c \in \mathcal{C}} \tag{5.11}$$

. Comparing to TDE [5], our method calculates the feature bias for each category, which could be more robust to the change of data distribution. As analyzed before, because the classifier layer is linear, the prediction of mean features is the same as mean prediction of the sampled features. For TDE, it supposes the bias totally depends on the feature distributions of the whole training dataset $\{\mathbf{f}_i\}_{i\in\mathcal{T}}$, which, in another view, equals to prediction of the whole training dataset $\{\tilde{y}_i\}_{i\in\mathcal{T}}$. However, the change of mean prediction of the whole training samples can not necessarily lead to the change of bias in model training. For example, taking a trained model which predicts perfectly on the head colors and overfits the training samples in head colors, adding more trivial samples for those head colors in the fine-tune training would only slightly affect the model training when the gradients are too small. In that scenario, the bias calculation in TDE will be changed and not in line with the effects of the trained model. On the contrary, our method incorporates the mean features for each category, which is more robust to the change of the data distribution.

As shown in Fig. 5.10 right, the ability to remove the bias for our method could be stronger than τ-norm, which only removes the bias from classifiers. Through the change of controlling factor p in Eq. (5.11), we can find the optimal value for the balance-category test condition. on the other hand, the mRecall saturates even using a large *tau*. Our method considers the bias in the classifier by leveraging the sparse encoding ability in the classifier layer. Besides, the mean features calculated for each category reflect bias in the feature extractor.

5.6.4 Further Discussion: Classifier as Descriptor

As discussed before, we observe that the color description for referenced object differs in the human preference of color used, which might lead to the bias prediction of the bias-trained model.

In the view of optimization for classifiers, the rich information in this preference distribution is backpropagated through classifiers. Specifically, many samples labeled

Table 5.7: Results of removing the bias: VG Color and Stock Color

	VG Color				Stock Color			
	Precision@1	Recall@2	mRecall@1	mRecall@2	Precision@1	Recall@2	mRecall@1	mRecall@2
Benchmark(Attention)	78.79%	85.88%	24.98%	34.80%	85.19%	93.17%	41.40%	53.08%
NCM [4]	61.93%	68.85%	28.70%	49.45%	70.48%	79.19%	50.23%	63.92%
τ-normalized [4]	76.06%	84.52%	27.24%	36.27%	83.70%	92.78%	41.20%	54.01%
TDE [5]	72.87%	78.25%	29.09%	40.93%	81.40%	88.62%	49.09%	62.94%
ours	72.59%	75.60%	**29.86**%	**52.61**%	76.95%	83.80%	**55.19**%	**73.29**%

as head colors could reasonably be labeled as tail colors from different preferences will be trained only through head classifier but not tail classifier. Hence, for those tail-color classifiers which are not effectively trained due to under-sampling, those head-color classifiers would also contain some information in respecting correlated tail colors which are under-preferred from labels. Such preference encodes the correlation between different colors and will be reflected by the prediction of bias-trained model. As shown in Fig. 5.11, tail-color mean features (bottom rows) are heavily activates bias towards the correlated head colors by the corresponding classifier (left columns). For example, the mean feature of platinum (in the last row) is activated by the correlated head classifiers, black, gold, and silver. And our method, classifier as descriptor, effectively exploiting the correlations from the classifiers.

In another view of visual descriptor, the tail colors would describe themselves through the selection to other related colors as keywords. Moreover, the representation encoded by classifiers are sparse and separable (*c.f.* Fig. 5.11) due to different preference description for difference color. Through the descriptor from the classifiers which learns the information of color preference distribution, we can obtain meaningful representations where the tail-color samples could take effectively trained head colors as visual words to represent themselves.

5.6.5 Experimental Results

As shown in Table 5.7 and, our method outperforms other methods in mRecall@1 and mRecall@2, showing the strong ability to remove the bias from bias-trained model. Kang *et al.* [4] also studied Nearest Class Mean classifier (NCM) and Classi-

Table 5.8: Results of removing the bias: tested on cross-domain annotations

	Stock Color (Human Language)				Stock Color (Human RGB)			
	Precision@1	Recall@2	mRecall@1	mRecall@2	Precision@1	Recall@2	mRecall@1	mRecall@2
Benchmark(Attention)	75.91%	73.53%	26.89%	38.41%	65.44%	63.89%	24.89%	31.94%
NCM [4]	66.62%	64.93%	31.28%	34.40%	59.59%	60.46%	27.71%	30.21%
τ-normalized [4]	74.78%	72.92%	27.09%	37.18%	64.19%	63.16%	25.19%	33.37%
TDE [5]	73.91%	71.29%	29.33%	46.49%	63.69%	64.08%	26.46%	41.74%
ours	71.21%	68.98%	**30.66**%	**47.05**%	61.69%	62.09%	**26.81**%	**42.62**%

fier Re-training (cRT). NCM is to first compute the mean feature representation for each class on the training set and then perform the nearest neighbor search either using cosine similarity or the Euclidean distance computed on L2 normalized mean features. Despite its simplicity, they found it is a strong baseline and it is very similar to our method. Our method differs from it in two folds: (a) It neglects the trained classifier, which could contain useful information. On the contrary, our method takes the classifier as a meaningful sparse encoder, which takes the classifier as a descriptor. We tried only to use the head-color classifier as the sparse encoder, which could still result in good performance though the accuracy will be slightly dropped. Besides, as shown in Fig. 5.11, the poorly-trained tail-color classifier could be still useful to activate the tail colors. (b) We introduce a controlling factor p, which could balance the recall and mRecall in an effective and flexible manner. cRT re-trains the classifier with class-balanced sampling. We found it provides marginal improvement comparing to other methods.

In summary, we investigate the different methods to remove the bias for long-tail color recognition, and proposed a succinct and effective method to remove the bias in the bias-trained model. Through the analysis and experiments, it shows the bias-trained classifier could serve as a meaningful descriptor and be used as a sparse encoder to describe the visual anchors (mean features) in a separable space. Though the classifier as a descriptor by feeding visual anchors to classifiers, it distills the rich knowledge in both feature extractors and classifiers, spanning a space that removes the bias. Beyond the color recognition, We will leave the generalization study for this

method for more tasks in future work.

5.7 Conclusion

In this chapter, we investigated color attribute prediction by learning a language description from caption and grounding these to image objects identified by our computer vision system. We introduce a benchmark dataset with automatically generated labels and golden labels corresponding to human language color concepts. Furthermore, we explore techniques to improve color recognition performance through compositional representation learning that is inspired by color decomposition. For the long tail color recognition, we propose another method by incorporating the compositional representations in classifier, outperforming the state-of-the-arts debiasing methods on the benchmark dataset. We hope this dataset, along with our analysis of color decomposition and debiasing can help set up new directions for future research into this problem.

CHAPTER 6: SUMMARY AND FURTURE DIRECTION

6.1 Summary

In this dissertation, we investigated four factors in representation learning of image recognition, *i.e.* diversity, aspect ratio, invariance, and composition. We summarized the works in recent advances of deep learning that is related to the four factors and demonstrated some insights by diving deep into different tasks respectively.

In Chapter 1, we introduced the four factors in representation learning of image recognition. Despite some generality of some methods, good representations could vary from different domains, where some representations are more important to achieve the specific goals. Many methods of representation learning focus on the identified important issues to help target tasks. We walked through different domains and recent research advances that are related to the four factors. For diversity, diverse representation learning plays an important role in the development of single network architectures (*e.g.* group convolution, diverse filters, regularization *et al.*) and ensemble algorithms (*e.g.* boosting, re-weighting, task division *et al.*). And we aimed at ensemble learning in this work and introduced motivation of the diverse representation learning with discriminative attention on different categories. For aspect ratio, we visited the impacts of aspect ratios for different tasks. Some tasks do not require implicit efforts about the representation learning of aspect ratios because of direct supervision of the annotation, *e.g.* object detection, and semantic segmentation. Many general tasks even treat the variations of aspect ratios as an effective way for data augmentation, *e.g.* classification. However, image aesthetics assessment should take the representation learning about aspect ratios into consideration. And we briefly went through the drawbacks of some previous methods to address the

issues. For invariance, some tasks implicitly learn the representations of variances (*e.g.* generative model) and some tasks only require the representations where the variances are dissected (*e.g.* data augmentation, identification). Then we unfolded the invariance representation learning in the task of person re-identification and illustrated some previous methods that require explicit efforts of either localization or disentanglement. For composition, it is well studied in many different tasks, *e.g.* zero-shot compositional learning, unsupervised image-to-image translation, document Q&A. We introduced the compositional representation learning for the task of color attribute recognition, where the motivation of two methods are illustrated, *i.e.* one for general color recognition and the other for debiasing.

In Chapter 2, a diverse representation learning method, namely learning-difficulty-aware embedding, was proposed for ensemble learning. We provided the theoretical analysis for the proposed method. Through extensive experiments, it consistently outperforms the previous methods. In addition, the analysis and experiments revealed the drawbacks of previous methods and insights for diversity,. Guided optimization (comparing to MixDCNN) and regularization in category level (comparing to AdaBoost) are the important mechanisms for the success of diverse representation learning for our ensemble learning method.

In Chapter 3, an aspect ratio representation learning method, namely adaptive fractional dilated convolution, was developed for explicitly learning aesthetics representations related to aspect ratios for image aesthetics assessment. We embed the aspect ratios seamlessly into kernel level by adaptive construction of kernels in a parameter-free approach and solve the issues of aspect ratio representation learning natively and efficiently in an end-to-end manner. Through the experiments, we showed that our representation learning method outperforms the previous methods that are convoluted and slow.

In Chapter 4, an invariance representation learning method, namely anchor loss,

was introduced to further reduce the intra-class variances for person re-identification. We demonstrated that the global supervision from the feature distribution of cross-instance aggregation provided an effective way for invariance representation learning that distills the identity-related features and removes interfering variances. The invariance was learned without extra supervision or explicit priors through only extra epochs of fine-tuning with anchor loss. This succinct method boosted the performance significantly and achieved state of the art in multiple benchmark datasets.

In Chapter 5, two compositional representation learning methods, *i.e.* attention module with reference to key colors and a debiasing method, were presented for color attribute recognition. We introduced the dataset building and annotations for this new task, *i.e.* color attribute recognition. This overlooked problem was entangled with both language attention and visual attention. In visual attention, global contextual information and high-level semantic features were important for accurate color recognition that is aligned with language description from the comparison experiment with 1D Conv color histogram. The proposed color compositional modular, which was motivated by color theory, learned better presentations for color recognition. In language attention, the bias in annotations due to language preferences could be revealed by the classifier representations in the bias-trained model. And our debiasing method by incorporating the representations from classifiers improves the performance for long-tail color recognition significantly.

6.2 Future Direction

Representation learning is always involved with the development of deep learning methods for image recognition. The research of representation learning is an open-ended journey along with the advances with deep learning in different image recognition tasks. In this dissertation, we summarized the related methods from the perspective of representation learning for captured four factors in Chapter 1 and delve into some specific points for the algorithm development. To our best knowledge, the

systematic reviews and mechanism research for the four factors are still remained to be visited yet. Due to the rapid development of computer vision and the quick adoption of applications, many methods were proposed and widely used before the well understanding of their mechanisms. In other words, the understanding of algorithms is often behind the proposal and application. For example, Label Smoothing [61] was proposed in 2016 and has been widely used throughout different state-of-the-art methods. However, until recently, the mechanisms and understanding of Label Smoothing began to been researched from the perspectives of representation distribution [90], label noise [221], knowledge distillation [222], and *et al*. As the benchmarking of different tasks for computer vision becomes saturated, the research of explanation has aroused interests. Besides, due to large-scale learning parameters and stochastic optimization, the general approach to understand the mechanisms is based on experimental validation. Specifically, designing some interesting experiments to verify some hypotheses is more practical. For example, the research for validating the hypothesis, that deep networks have a strong ability to memorize large-scale training samples even without pattern recognition, is based on an experimental design of random label fitting [59]. And the understanding of mechanisms of residual connections from the perspective of ensemble learning is based on the experimental validation of layer re-ordering and deletion [28].

Representation learning provides a good perspective to look into the underlying mechanisms. In the following, we briefly illustrate some potential research problems related to the four factors of representation learning.

Diversity Though the research works about diverse representation learning are scattered around different domains, *e.g*. regularization, model compression, ensemble learning, and *et al*., the systematic reviews and analysis about the diversity have not been fully studied, neither for single network or ensemble learning. For single networks, how to measure the diversity of learned representations and what kind of

diverse representations are beneficial to improve the generalization ability of single networks, are remained to be delved deep into. For ensemble networks, some interesting but unresolved questions are: (a) How to compare and measure the diversity of complementary networks? In this work, we measure the diversity directly according to test accuracy in each category. Another possible approach that is more general and intuitive is to design the measurement of alignment and disagreement of different complementary networks in both decisions and learned features, *e.g.* appending some transformation layers to learn a shared latent embedding. (b) What factors could impact the diversity of complementary networks and what are the mechanisms of each factor? For example, the factors that could potentially impact the diversity are network architecture, weight initialization, dataset distribution, optimization methods, and *et al*. It is attempting to conduct comprehensive experiments to get some observations and design the following experiments to verify the explainable hypothesis. (c) How to balance the diversity of individual networks and the aggregation for ensemble networks? In this work, we derived the reweighting formulas for the balance between diversity and aggregation where the training of each complementary network is managed and involved. However, is there a more clever and automatic way to ensemble multiple existing networks? One obvious approach is to train an extra layer to calibrate the output decisions of multiple complementary networks. Another direction is to use meta-learning to train a meta-network for decision aggregation or feature fusion.

Aspect Ratio Representation learning of aspect ratios is proved to be important for image aesthetics assessment. To our best knowledge, it has not been fully understood in other fields. For example, aspect ratio representation learning could be beneficial for the detection algorithm of face manipulation. It is unknown that how the learned presentations of aspect ratios could contain the artifacts information of the face manipulation.

Invariance Invariance plays an important part in representation learning. How to test the invariance has not been thoroughly delved into yet. An interesting direction is to test the sensibility in the activation of classifiers to different variation input. On the other hand, the disentanglement and image reconstruction could provide another perspective to recover the variance representations that have been dissected and to compare the representations between recovered variance representations and original invariance representations. In addition, the disentangle learning is actively researched in image-to-image translation where both invariance and variance representations are important, and whether it is beneficial for the tasks that care the only invariance could be further researched into. In another word, does the extra modeling of variances could help with the modeling of invariance presentations?

Composition The aggregation of compositional representation learnings varies from different methods, *e.g.* feature fusion, attention, addition, concatenation *et al*. In this work, we adopt the attention for color compositional learning modular and distance calculation for debiasing method. It is important to systematically compare the different aggregation methods to obtain some understanding of mechanisms.

Overall, as the algorithms of deep learning have achieved applicable results in many benchmark tasks, representation learning could be a promising perspective to unfold some underlying mechanisms of the algorithms. In return, it could contribute to the further development of more efficient algorithms and provide more structured guidance in applying the algorithms and problem identification in real-world scenarios.

REFERENCES

[1] N. Murray, L. Marchesotti, and F. Perronnin, "Ava: A large-scale database for aesthetic visual analysis," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 2408–2415, IEEE, 2012.

[2] H. Talebi and P. Milanfar, "NIMA: neural image assessment," *IEEE Trans. Image Processing*, vol. 27, no. 8, pp. 3998–4011, 2018.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[4] B. Kang, S. Xie, M. Rohrbach, Z. Yan, A. Gordo, J. Feng, and Y. Kalantidis, "Decoupling representation and classifier for long-tailed recognition," in *International Conference on Learning Representations*, 2020.

[5] K. Tang, Y. Niu, J. Huang, J. Shi, and H. Zhang, "Unbiased scene graph generation from biased training," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3716–3725, 2020.

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[7] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations, ICLR*, 2015.

[8] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

[9] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks.," in *CVPR*, vol. 1, p. 3, 2017.

[10] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7132–7141, 2018.

[11] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *CoRR*, vol. abs/1704.04861, 2017.

[12] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pp. 4510–4520, 2018.

[13] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pp. 6848–6856, 2018.

[14] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448, 2015.

[15] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, pp. 91–99, 2015.

[16] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017.

[17] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*, pp. 21–37, Springer, 2016.

[18] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

[19] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7263–7271, 2017.

[20] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

[21] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: Scalable and efficient object detection," *arXiv preprint arXiv:1911.09070*, 2019.

[22] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440, 2015.

[23] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015.

[24] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2018.

[25] D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee, "Yolact: real-time instance segmentation," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 9157–9166, 2019.

[26] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[27] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256, JMLR Workshop and Conference Proceedings, 2010.

[28] A. Veit, M. Wilber, and S. Belongie, "Residual networks behave like ensembles of relatively shallow networks," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pp. 550–558, 2016.

[29] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[30] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117–2125, 2017.

[31] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pp. 5987–5995, IEEE, 2017.

[32] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International Conference on Machine Learning*, pp. 6105–6114, PMLR, 2019.

[33] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, "Multimodal unsupervised image-to-image translation," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 172–189, 2018.

[34] H.-Y. Lee, H.-Y. Tseng, J.-B. Huang, M. Singh, and M.-H. Yang, "Diverse image-to-image translation via disentangled representations," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 35–51, 2018.

[35] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4401–4410, 2019.

[36] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2021.

[37] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, pp. 5998–6008, 2017.

[38] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[39] H. Schwenk and Y. Bengio, "Adaboosting neural networks: Application to on-line character recognition," in *International Conference on Artificial Neural Networks*, pp. 967–972, Springer, 1997.

[40] L. Zhou and K. K. Lai, "Adaboosting neural networks for credit scoring," in *The Sixth International Symposium on Neural Networks (ISNN 2009)*, pp. 875–884, Springer, 2009.

[41] I. O. Tolstikhin, S. Gelly, O. Bousquet, C.-J. Simon-Gabriel, and B. Schölkopf, "Adagan: Boosting generative models," in *Advances in Neural Information Processing Systems*, pp. 5424–5433, 2017.

[42] J. Friedman, T. Hastie, R. Tibshirani, *et al.*, "Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors)," *The annals of statistics*, vol. 28, no. 2, pp. 337–407, 2000.

[43] M. Moghimi, S. J. Belongie, M. J. Saberian, J. Yang, N. Vasconcelos, and L.-J. Li, "Boosted convolutional neural networks.," in *BMVC*, pp. 24–1, 2016.

[44] N. Cohen, R. Tamari, and A. Shashua, "Boosting dilated convolutional networks with mixed tensor decompositions," in *6th International Conference on Learning Representations, ICLR*, 2018.

[45] H. Schwenk and Y. Bengio, "Boosting neural networks," *Neural computation*, vol. 12, no. 8, pp. 1869–1887, 2000.

[46] C. Cortes, M. Mohri, and U. Syed, "Deep boosting," in *International Conference on Machine Learning*, pp. 1179–1187, 2014.

[47] Z. Peng, Y. Li, Z. Cai, and L. Lin, "Deep boosting: joint feature selection and analysis dictionary learning in hierarchy," *Neurocomputing*, vol. 178, pp. 36–45, 2016.

[48] R. E. Schapire and Y. Singer, "Improved boosting algorithms using confidence-rated predictions," *Machine learning*, vol. 37, no. 3, pp. 297–336, 1999.

[49] H. Drucker, R. Schapire, and P. Simard, "Improving performance in neural networks using a boosting algorithm," in *Advances in neural information processing systems*, pp. 42–49, 1993.

[50] T.-K. Kim, I. Budvytis, and R. Cipolla, "Making a shallow network deep: Conversion of a boosting classifier into a decision tree by boolean optimisation," *International journal of computer vision*, vol. 100, no. 2, pp. 203–215, 2012.

[51] M. J. Saberian and N. Vasconcelos, "Multiclass boosting: Theory and algorithms," in *Advances in Neural Information Processing Systems*, pp. 2124–2132, 2011.

[52] V. Kuznetsov, M. Mohri, and U. Syed, "Multi-class deep boosting," in *Advances in Neural Information Processing Systems*, pp. 2501–2509, 2014.

[53] G. Martinez-Munoz, N. Larios, E. Mortensen, W. Zhang, A. Yamamuro, R. Paasch, N. Payet, D. Lytle, L. Shapiro, S. Todorovic, *et al.*, "Dictionary-free categorization of very similar objects via stacked evidence trees," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 549–556, IEEE, 2009.

[54] L. Deng, D. Yu, and J. Platt, "Scalable stacking and learning for building deep architectures," in *2012 IEEE International conference on Acoustics, speech and signal processing (ICASSP)*, pp. 2133–2136, IEEE, 2012.

[55] J. Li, H. Chang, and J. Yang, "Sparse deep stacking network for image classification," in *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[56] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, and Q. Tian, "Scalable person re-identification: A benchmark," in *Computer Vision, IEEE International Conference on*, 2015.

[57] W. Li, R. Zhao, T. Xiao, and X. Wang, "Deepreid: Deep filter pairing neural network for person re-identification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 152–159, 2014.

[58] E. Ristani, F. Solera, R. Zou, R. Cucchiara, and C. Tomasi, "Performance measures and a data set for multi-target, multi-camera tracking," in *European Conference on Computer Vision*, pp. 17–35, Springer, 2016.

[59] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," in *5th International Conference on Learning Representations, ICLR*, 2017.

[60] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*, pp. 448–456, PMLR, 2015.

[61] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.

[62] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, 2017.

[63] L. Huang, X. Liu, B. Lang, A. Yu, Y. Wang, and B. Li, "Orthogonal weight normalization: Solution to optimization over multiple dependent stiefel manifolds in deep neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, 2018.

[64] D. Xie, J. Xiong, and S. Pu, "All you need is beyond a good init: Exploring better solution for training extremely deep convolutional neural networks with orthonormality and modulation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6176–6185, 2017.

[65] K. Jia, D. Tao, S. Gao, and X. Xu, "Improving training of deep neural networks via singular value bounding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4344–4352, 2017.

[66] M. Harandi and B. Fernando, "Generalized backpropagation,\'{E} tude de cas: Orthogonality," *arXiv preprint arXiv:1611.05927*, 2016.

[67] M. Ozay and T. Okatani, "Optimization on submanifolds of convolution kernels in cnns," *arXiv preprint arXiv:1610.07008*, 2016.

[68] N. Bansal, X. Chen, and Z. Wang, "Can we gain more from orthogonality regularizations in training deep networks?," *Advances in Neural Information Processing Systems*, vol. 31, pp. 4261–4271, 2018.

[69] T. Chen, S. Ding, J. Xie, Y. Yuan, W. Chen, Y. Yang, Z. Ren, and Z. Wang, "Abd-net: Attentive but diverse person re-identification," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 8351–8361, 2019.

[70] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random erasing data augmentation," *arXiv preprint arXiv:1708.04896*, 2017.

[71] T. DeVries and G. W. Taylor, "Improved regularization of convolutional neural networks with cutout," *arXiv preprint arXiv:1708.04552*, 2017.

[72] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[73] L. Wan, M. Zeiler, S. Zhang, Y. Le Cun, and R. Fergus, "Regularization of neural networks using dropconnect," in *International conference on machine learning*, pp. 1058–1066, PMLR, 2013.

[74] G. Ghiasi, T.-Y. Lin, and Q. V. Le, "Dropblock: a regularization method for convolutional networks," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 10750–10760, 2018.

[75] G. Larsson, M. Maire, and G. Shakhnarovich, "Fractalnet: Ultra-deep neural networks without residuals," *arXiv preprint arXiv:1605.07648*, 2016.

[76] Z. Ge, A. Bewley, C. McCool, P. Corke, B. Upcroft, and C. Sanderson, "Fine-grained classification via mixture of deep convolutional neural networks," in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1–6, IEEE, 2016.

[77] T. Zhao, Q. Chen, Z. Kuang, J. Yu, W. Zhang, and J. Fan, "Deep mixture of diverse experts for large-scale visual recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 5, pp. 1072–1087, 2018.

[78] Q. Chen, W. Zhang, J. Yu, and J. Fan, "Embedding complementary deep networks for image classification," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[79] A. Veit, M. J. Wilber, and S. Belongie, "Residual networks behave like ensembles of relatively shallow networks," in *Advances in neural information processing systems*, pp. 550–558, 2016.

[80] S. Zagoruyko and N. Komodakis, "Wide residual networks," *arXiv preprint arXiv:1605.07146*, 2016.

[81] L. Mai, H. Jin, and F. Liu, "Composition-preserving deep photo aesthetics assessment," in *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 497–506, 2016.

[82] S. Ma, J. Liu, and C. W. Chen, "A-lamp: Adaptive layout-aware multi-patch deep convolutional neural network for photo aesthetic assessment," in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 722–731, 2017.

[83] V. Hosu, B. Goldlucke, and D. Saupe, "Effective aesthetics prediction with multi-level spatially pooled features," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9375–9383, 2019.

[84] K. Sheng, W. Dong, C. Ma, X. Mei, F. Huang, and B.-G. Hu, "Attention-based multi-patch aggregation for image aesthetic assessment," in *Proceedings of the 26th ACM international conference on Multimedia*, pp. 879–886, 2018.

[85] Q. Chen, W. Zhang, N. Zhou, P. Lei, Y. Xu, Y. Zheng, and J. Fan, "Adaptive fractional dilated convolution network for image aesthetics assessment," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14114–14123, 2020.

[86] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio, "Generative adversarial nets," in *NIPS*, 2014.

[87] C. Doersch, "Tutorial on variational autoencoders," *arXiv preprint arXiv:1606.05908*, 2016.

[88] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, "Stargan: Unified generative adversarial networks for multi-domain image-to-image translation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8789–8797, 2018.

[89] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[90] R. Müller, S. Kornblith, and G. Hinton, "When does label smoothing help?," *arXiv preprint arXiv:1906.02629*, 2019.

[91] Y. Sun, L. Zheng, Y. Yang, Q. Tian, and S. Wang, "Beyond part models: Person retrieval with refined part pooling (and a strong convolutional baseline)," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 480–496, 2018.

[92] J. Miao, Y. Wu, P. Liu, Y. Ding, and Y. Yang, "Pose-guided feature alignment for occluded person re-identification," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 542–551, 2019.

[93] Z. Zhang, C. Lan, W. Zeng, and Z. Chen, "Densely semantically aligned person re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 667–676, 2019.

[94] M. Tian, S. Yi, H. Li, S. Li, X. Zhang, J. Shi, J. Yan, and X. Wang, "Eliminating background-bias for robust person re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5794–5803, 2018.

[95] Y. Lin, L. Zheng, Z. Zheng, Y. Wu, Z. Hu, C. Yan, and Y. Yang, "Improving person re-identification by attribute and identity learning," *Pattern Recognition*, vol. 95, pp. 151–161, 2019.

[96] Z. Zheng, X. Yang, Z. Yu, L. Zheng, Y. Yang, and J. Kautz, "Joint discriminative and generative learning for person re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2138–2147, 2019.

[97] Q. Chen, W. Zhang, and J. Fan, "Cluster-level feature alignment for person re-identification," *arXiv preprint arXiv:2008.06810*, 2020.

[98] I. Tošić and P. Frossard, "Dictionary learning," *IEEE Signal Processing Magazine*, vol. 28, no. 2, pp. 27–38, 2011.

[99] M. Ranzato, Y.-L. Boureau, Y. LeCun, *et al.*, "Sparse feature learning for deep belief networks," *Advances in neural information processing systems*, vol. 20, pp. 1185–1192, 2007.

[100] Y. Zhang, R. Jin, and Z.-H. Zhou, "Understanding bag-of-words model: a statistical framework," *International Journal of Machine Learning and Cybernetics*, vol. 1, no. 1-4, pp. 43–52, 2010.

[101] I. Misra, A. Gupta, and M. Hebert, "From red wine to red tomato: Composition with context," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1792–1801, 2017.

[102] S. Purushwalkam, M. Nickel, A. Gupta, and M. Ranzato, "Task-driven modular networks for zero-shot compositional learning," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3593–3602, 2019.

[103] Y.-L. Li, Y. Xu, X. Mao, and C. Lu, "Symmetry and group in attribute-object compositions," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11316–11325, 2020.

[104] Z. Nan, Y. Liu, N. Zheng, and S.-C. Zhu, "Recognizing unseen attribute-object pair with generative model," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 8811–8818, 2019.

[105] T. Nagarajan and K. Grauman, "Attributes as operators: factorizing unseen attribute-object compositions," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 169–185, 2018.

[106] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

[107] A. H. Miller, A. Fisch, J. Dodge, A. Karimi, A. Bordes, and J. Weston, "Key-value memory networks for directly reading documents," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016* (J. Su, X. Carreras, and K. Duh, eds.), pp. 1400–1409, The Association for Computational Linguistics, 2016.

[108] S. Sukhbaatar, J. Weston, R. Fergus, *et al.*, "End-to-end memory networks," in *Advances in neural information processing systems*, pp. 2440–2448, 2015.

[109] C. Chunseong Park, B. Kim, and G. Kim, "Attend to you: Personalized image captioning with context sequence memory networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 895–903, 2017.

[110] T. Yang and A. B. Chan, "Learning dynamic memory networks for object tracking," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 152–167, 2018.

[111] S. W. Oh, J.-Y. Lee, N. Xu, and S. J. Kim, "Video object segmentation using space-time memory networks," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 9226–9235, 2019.

[112] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[113] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7794–7803, 2018.

[114] C. Sun, A. Myers, C. Vondrick, K. Murphy, and C. Schmid, "Videobert: A joint model for video and language representation learning," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 7464–7473, 2019.

[115] I. Bello, B. Zoph, A. Vaswani, J. Shlens, and Q. V. Le, "Attention augmented convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3286–3295, 2019.

[116] H. Zhao, J. Jia, and V. Koltun, "Exploring self-attention for image recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10076–10085, 2020.

[117] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European Conference on Computer Vision (ECCV)*, 2020.

[118] H. Hu, J. Gu, Z. Zhang, J. Dai, and Y. Wei, "Relation networks for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3588–3597, 2018.

[119] J. Lu, D. Batra, D. Parikh, and S. Lee, "Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks," in *Advances in Neural Information Processing Systems*, pp. 13–23, 2019.

[120] Y.-C. Chen, L. Li, L. Yu, A. El Kholy, F. Ahmed, Z. Gan, Y. Cheng, and J. Liu, "Uniter: Universal image-text representation learning," in *European Conference on Computer Vision*, pp. 104–120, Springer, 2020.

[121] J. Lu, V. Goswami, M. Rohrbach, D. Parikh, and S. Lee, "12-in-1: Multi-task vision and language representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10437–10446, 2020.

[122] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *ICML deep learning workshop*, vol. 2, Lille, 2015.

[123] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Advances in neural information processing systems*, pp. 4077–4087, 2017.

[124] S. Guerriero, B. Caputo, and T. Mensink, "Deep nearest class mean classifiers," in *International Conference on Learning Representations, Worskhop Track*, 2018.

[125] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "icarl: Incremental classifier and representation learning," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2001–2010, 2017.

[126] P. Viola and M. J. Jones, "Robust real-time face detection," *International journal of computer vision*, vol. 57, no. 2, pp. 137–154, 2004.

[127] X. Zhang, Z. Li, C. Change Loy, and D. Lin, "Polynet: A pursuit of structural diversity in very deep networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 718–726, 2017.

[128] J. Schmidhuber, "Multi-column deep neural networks for image classification," in *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3642–3649, IEEE Computer Society, 2012.

[129] C.-Y. Lee, P. W. Gallagher, and Z. Tu, "Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree," in *Artificial intelligence and statistics*, pp. 464–472, 2016.

[130] H. Pham, M. Y. Guan, B. Zoph, Q. V. Le, and J. Dean, "Efficient neural architecture search via parameter sharing," in *ICML*, 2018.

[131] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8697–8710, 2018.

[132] C. Liu, B. Zoph, M. Neumann, J. Shlens, W. Hua, L.-J. Li, L. Fei-Fei, A. Yuille, J. Huang, and K. Murphy, "Progressive neural architecture search," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 19–34, 2018.

[133] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le, "Regularized evolution for image classifier architecture search," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 4780–4789, 2019.

[134] H. Liu, K. Simonyan, and Y. Yang, "DARTS: Differentiable architecture search," in *International Conference on Learning Representations*, 2019.

[135] S. Qiao, Z. Zhang, W. Shen, B. Wang, and A. L. Yuille, "Gradually updated neural networks for large-scale image recognition," in *International Conference on Machine Learning (ICML)*, 2018.

[136] C. Cortes, X. Gonzalvo, V. Kuznetsov, M. Mohri, and S. Yang, "Adanet: Adaptive structural learning of artificial neural networks," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 874–883, JMLR. org, 2017.

[137] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The caltech-ucsd birds-200-2011 dataset," 2011.

[138] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," tech. rep., Citeseer, 2009.

[139] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 248–255, Ieee, 2009.

[140] T.-Y. Lin, A. RoyChowdhury, and S. Maji, "Bilinear cnn models for fine-grained visual recognition," in *Proceedings of the IEEE international conference on computer vision*, pp. 1449–1457, 2015.

[141] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[142] R. Datta, D. Joshi, J. Li, and J. Z. Wang, "Studying aesthetics in photographic images using a computational approach," in *Computer Vision - ECCV 2006, 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006, Proceedings, Part III*, pp. 288–301, 2006.

[143] S. Dhar, V. Ordonez, and T. L. Berg, "High level describable attributes for predicting aesthetics and interestingness," in *The 24th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2011, Colorado Springs, CO, USA, 20-25 June 2011*, pp. 1657–1664, 2011.

[144] Y. Ke, X. Tang, and F. Jing, "The design of high-level features for photo quality assessment," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2006), 17-22 June 2006, New York, NY, USA*, pp. 419–426, 2006.

[145] M. Nishiyama, T. Okabe, I. Sato, and Y. Sato, "Aesthetic quality classification of photographs based on color harmony," in *The 24th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2011, Colorado Springs, CO, USA, 20-25 June 2011*, pp. 33–40, 2011.

[146] X. Sun, H. Yao, R. Ji, and S. Liu, "Photo assessment based on computational visual attention model," in *Proceedings of the 17th International Conference on Multimedia 2009, Vancouver, British Columbia, Canada, October 19-24, 2009*, pp. 541–544, 2009.

[147] H. Tong, M. Li, H. Zhang, J. He, and C. Zhang, "Classification of digital photos taken by photographers or home users," in *Advances in Multimedia Information Processing - PCM 2004, 5th Pacific Rim Conference on Multimedia, Tokyo, Japan, November 30 - December 3, 2004, Proceedings, Part I*, pp. 198–205, 2004.

[148] Y. Kao, K. Huang, and S. Maybank, "Hierarchical aesthetic quality assessment using deep convolutional neural networks," *Signal Processing: Image Communication*, vol. 47, pp. 500–510, 2016.

[149] W. Wang, M. Zhao, L. Wang, J. Huang, C. Cai, and X. Xu, "A multi-scene deep learning model for image aesthetic evaluation," *Sig. Proc.: Image Comm.*, vol. 47, pp. 511–518, 2016.

[150] J. Ren, X. Shen, Z. L. Lin, R. Mech, and D. J. Foran, "Personalized image aesthetics," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pp. 638–647, 2017.

[151] H. Zeng, Z. Cao, L. Zhang, and A. C. Bovik, "A unified probabilistic formulation of image aesthetic assessment," *IEEE Transactions on Image Processing*, vol. 29, pp. 1548–1561, 2019.

[152] N. Murray and A. Gordo, "A deep architecture for unified aesthetic prediction," *CoRR*, vol. abs/1708.04890, 2017.

[153] S. Kong, X. Shen, Z. L. Lin, R. Mech, and C. C. Fowlkes, "Photo aesthetics ranking network with attributes and content adaptation," in *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part I*, pp. 662–679, 2016.

[154] K. Schwarz, P. Wieschollek, and H. P. A. Lensch, "Will people like your image? learning the aesthetic space," in *2018 IEEE Winter Conference on Applications of Computer Vision, WACV 2018, Lake Tahoe, NV, USA, March 12-15, 2018*, pp. 2048–2057, 2018.

[155] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, 2015.

[156] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, pp. 764–773, 2017.

[157] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," in *European conference on computer vision*, pp. 346–361, Springer, 2014.

[158] A. Hermans, L. Beyer, and B. Leibe, "In defense of the triplet loss for person re-identification," *arXiv preprint arXiv:1703.07737*, 2017.

[159] Y. Fu, Y. Wei, Y. Zhou, H. Shi, G. Huang, X. Wang, Z. Yao, and T. Huang, "Horizontal pyramid matching for person re-identification," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 8295–8302, 2019.

[160] J. Guo, Y. Yuan, L. Huang, C. Zhang, J.-G. Yao, and K. Han, "Beyond human parts: Dual part-aligned representations for person re-identification," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3642–3651, 2019.

[161] R. Quan, X. Dong, Y. Wu, L. Zhu, and Y. Yang, "Auto-reid: Searching for a part-aware convnet for person re-identification," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3750–3759, 2019.

[162] Y. Sun, Q. Xu, Y. Li, C. Zhang, Y. Li, S. Wang, and J. Sun, "Perceive where to focus: Learning visibility-aware part-level features for partial person re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 393–402, 2019.

[163] F. Zheng, C. Deng, X. Sun, X. Jiang, X. Guo, Z. Yu, F. Huang, and R. Ji, "Pyramidal person re-identification via multi-loss dynamic training," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8514–8522, 2019.

[164] W. Li, X. Zhu, and S. Gong, "Harmonious attention network for person re-identification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2285–2294, 2018.

[165] L. Shuang, B. Slawomir, C. Peter, and W. Xiaogang, "Diversity regularized spatiotemporal attention for videobased person re-identification," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 369–378, 2018.

[166] L. Dangwei, C. Xiaotang, Z. Zhang, and H. Kaiqi, "Learning deep context-aware features over body and latent parts for person re-identification," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 384–393, 2017.

[167] J. Xu, R. Zhao, F. Zhu, H. Wang, and W. Ouyang, "Attention-aware compositional network for person re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2119–2128, 2018.

[168] Z. Haiyu, T. Maoqing, S. Shuyang, S. Jing, Y. Junjie, Y. Shuai, W. Xiaogang, and T. Xiaoou, "Spindle net: Person re-identification with human body region guided feature decomposition and fusion," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017.

[169] J. Liu, B. Ni, Y. Yan, P. Zhou, S. Cheng, and J. Hu, "Pose transferrable person re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4099–4108, 2018.

[170] M. M. Kalayeh, E. Basaran, M. Gökmen, M. E. Kamasak, and M. Shah, "Human semantic parsing for person re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1062–1071, 2018.

[171] M. Saquib Sarfraz, A. Schumann, A. Eberle, and R. Stiefelhagen, "A pose-sensitive embedding for person re-identification with expanded cross neighborhood re-ranking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 420–429, 2018.

[172] A. Subramaniam, A. Nambiar, and A. Mittal, "Co-segmentation inspired attention networks for video-based person re-identification," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 562–572, 2019.

[173] C.-P. Tay, S. Roy, and K.-H. Yap, "Aanet: Attribute attention network for person re-identifications," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7134–7143, 2019.

[174] Z. Zhedong, Z. Liang, and Y. Yi, "Unlabeled samples generated by gan improve the person re-identification baseline in video," in *ICCV*, 2017.

[175] L. Xiang, W. Ancong, and Z. Wei-Shi, "Adversarial open-world person re-identification," in *ECCV*, 2018.

[176] H. Yan, X. Jinsong, W. Qiang, Z. Zhedong, Z. Zhaoxiang, and Z. Jian, "Multi-pseudo regularized label for generated samples in person re-identification," *TIP*, 2018.

[177] Z. Zheng, L. Zheng, and Y. Yang, "A discriminatively learned cnn embedding for person reidentification," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 14, no. 1, pp. 1–20, 2017.

[178] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *European conference on computer vision*, pp. 499–515, Springer, 2016.

[179] H. Luo, W. Jiang, Y. Gu, F. Liu, X. Liao, S. Lai, and J. Gu, "A strong baseline and batch normalization neck for deep person re-identification," *IEEE Transactions on Multimedia*, 2019.

[180] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A comprehensive study on center loss for deep face recognition," *International Journal of Computer Vision*, vol. 127, no. 6-7, pp. 668–683, 2019.

[181] X. Chang, T. M. Hospedales, and T. Xiang, "Multi-level factorisation net for person re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2109–2118, 2018.

[182] C. Wang, Q. Zhang, C. Huang, W. Liu, and X. Wang, "Mancs: A multi-task attentional network with curriculum sampling for person re-identification," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 365–381, 2018.

[183] G. Wang, Y. Yuan, X. Chen, J. Li, and X. Zhou, "Learning discriminative features with multiple granularities for person re-identification," in *Proceedings of the 26th ACM international conference on Multimedia*, pp. 274–282, 2018.

[184] W. Yang, H. Huang, Z. Zhang, X. Chen, K. Huang, and S. Zhang, "Towards rich feature discovery with class activation maps augmentation for person re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1389–1398, 2019.

[185] M. Zheng, S. Karanam, Z. Wu, and R. J. Radke, "Re-identification with consistent attentive siamese networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5735–5744, 2019.

[186] B. Chen, W. Deng, and J. Hu, "Mixed high-order attention network for person re-identification," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 371–381, 2019.

[187] R. Hou, B. Ma, H. Chang, X. Gu, S. Shan, and X. Chen, "Interaction-and-aggregation network for person re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9317–9326, 2019.

[188] H. Sun, Z. Chen, S. Yan, and L. Xu, "Mvp matching: A maximum-value perfect matching for mining hard samples, with application to person re-identification," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 6737–6747, 2019.

[189] K. Zhou, Y. Yang, A. Cavallaro, and T. Xiang, "Omni-scale feature learning for person re-identification," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3702–3712, 2019.

[190] Z. Dai, M. Chen, X. Gu, S. Zhu, and P. Tan, "Batch dropblock network for person re-identification and beyond," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3691–3701, 2019.

[191] X. Pan, P. Luo, J. Shi, and X. Tang, "Two at once: Enhancing learning and generalization capacities via ibn-net," in *The European Conference on Computer Vision (ECCV)*, September 2018.

[192] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[193] L. Bourdev, S. Maji, and J. Malik, "Describing people: A poselet-based approach to attribute classification," in *2011 International Conference on Computer Vision*, pp. 1543–1550, IEEE, 2011.

[194] Y. Deng, P. Luo, C. C. Loy, and X. Tang, "Pedestrian attribute recognition at far distance," in *Proceedings of the 22nd ACM international conference on Multimedia*, pp. 789–792, 2014.

[195] K. Soomro, A. R. Zamir, and M. Shah, "A dataset of 101 human action classes from videos in the wild," *Center for Research in Computer Vision*, vol. 2, no. 11, 2012.

[196] D. H. Foster, "Color constancy," *Vision research*, vol. 51, no. 7, pp. 674–700, 2011.

[197] Z. Li, Q. Tran, L. Mai, Z. Lin, and A. L. Yuille, "Context-aware group captioning via self-attention and contrastive features," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3440–3450, 2020.

[198] C. Lu, R. Krishna, M. Bernstein, and L. Fei-Fei, "Visual relationship detection with language priors," in *European conference on computer vision*, pp. 852–869, Springer, 2016.

[199] S. Schuster, R. Krishna, A. Chang, L. Fei-Fei, and C. D. Manning, "Generating semantically precise scene graphs from textual descriptions for improved image retrieval," in *Proceedings of the fourth workshop on vision and language*, pp. 70–80, 2015.

[200] S. Garg, J. R. A. Moniz, A. Aviral, and P. Bollimpalli, "Learning to relate from captions and bounding boxes," *arXiv preprint arXiv:1912.00311*, 2019.

[201] C. H. Lampert, H. Nickisch, and S. Harmeling, "Learning to detect unseen object classes by between-class attribute transfer," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 951–958, IEEE, 2009.

[202] G. Patterson and J. Hays, "Coco attributes: Attributes for people, animals, and objects," in *European Conference on Computer Vision*, pp. 85–100, Springer, 2016.

[203] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, "Sun database: Large-scale scene recognition from abbey to zoo," in *2010 IEEE computer society conference on computer vision and pattern recognition*, pp. 3485–3492, IEEE, 2010.

[204] D. Mahajan, S. Sellamanickam, and V. Nair, "A joint learning framework for attribute models and object descriptions," in *2011 International Conference on Computer Vision*, pp. 1227–1234, IEEE, 2011.

[205] S. J. Hwang, F. Sha, and K. Grauman, "Sharing features between objects and their attributes," in *CVPR 2011*, pp. 1761–1768, IEEE, 2011.

[206] L. Shen, Z. Lin, and Q. Huang, "Relay backpropagation for effective learning of deep convolutional neural networks," in *European conference on computer vision*, pp. 467–482, Springer, 2016.

[207] D. Mahajan, R. Girshick, V. Ramanathan, K. He, M. Paluri, Y. Li, A. Bharambe, and L. van der Maaten, "Exploring the limits of weakly supervised pretraining," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 181–196, 2018.

[208] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel, "Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness.," in *International Conference on Learning Representations*, 2019.

[209] K. Cao, C. Wei, A. Gaidon, N. Arechiga, and T. Ma, "Learning imbalanced datasets with label-distribution-aware margin loss," in *Advances in Neural Information Processing Systems*, pp. 1567–1578, 2019.

[210] J. Shu, Q. Xie, L. Yi, Q. Zhao, S. Zhou, Z. Xu, and D. Meng, "Meta-weight-net: Learning an explicit mapping for sample weighting," in *NeurIPS*, 2019.

[211] Z. Liu, Z. Miao, X. Zhan, J. Wang, B. Gong, and S. X. Yu, "Large-scale long-tailed recognition in an open world," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2537–2546, 2019.

[212] P. Sharma, N. Ding, S. Goodman, and R. Soricut, "Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (Melbourne, Australia), pp. 2556–2565, Association for Computational Linguistics, July 2018.

[213] A. Kuznetsova, H. Rom, N. Alldrin, J. R. R. Uijlings, I. Krasin, J. Pont-Tuset, S. Kamali, S. Popov, M. Malloci, A. Kolesnikov, T. Duerig, and V. Ferrari, "The open images dataset V4," *Int. J. Comput. Vis.*, vol. 128, no. 7, pp. 1956–1981, 2020.

[214] M. Elhoseiny, S. Cohen, W. Chang, B. Price, and A. Elgammal, "Automatic annotation of structured facts in images," in *Proceedings of the 5th Workshop on Vision and Language*, (Berlin, Germany), pp. 1–9, Association for Computational Linguistics, Aug. 2016.

[215] M. Elhoseiny, S. Cohen, W. Chang, B. Price, and A. Elgammal, "Sherlock: Scalable fact learning in images," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[216] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, *et al.*, "Visual genome: Connecting language and vision using crowdsourced dense image annotations," *International journal of computer vision*, vol. 123, no. 1, pp. 32–73, 2017.

[217] D. Xu, Y. Zhu, C. B. Choy, and L. Fei-Fei, "Scene graph generation by iterative message passing," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5410–5419, 2017.

[218] R. Zellers, M. Yatskar, S. Thomson, and Y. Choi, "Neural motifs: Scene graph parsing with global context," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5831–5840, 2018.

[219] K. Tang, H. Zhang, B. Wu, W. Luo, and W. Liu, "Learning to compose dynamic tree structures for visual contexts," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6619–6628, 2019.

[220] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang, "Bottom-up and top-down attention for image captioning and visual question answering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6077–6086, 2018.

[221] M. Lukasik, S. Bhojanapalli, A. Menon, and S. Kumar, "Does label smoothing mitigate label noise?," in *International Conference on Machine Learning*, pp. 6448–6458, PMLR, 2020.

[222] L. Yuan, F. E. Tay, G. Li, T. Wang, and J. Feng, "Revisiting knowledge distillation via label smoothing regularization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3903–3911, 2020.