TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

DINGDING CAI

FINE-GRAINED CLASSIFICATION OF LOW-RESOLUTION IMAGE

Master of Science thesis

Examiner: Prof. Joni-Kristian Kämäräinen,
Dr. Ke Chen
Examiner and topic approved by the
Faculty Council of the Faculty of
Computing and Electrical Engineering
on 7th December 2017

ABSTRACT

DINGDING CAI: Fine-Grained Classification of Low-Resolution Image
Tampere University of Technology
Master of Science thesis, 42 pages, 0 Appendix pages
15th November 2017
Master's Degree Programme in Information Technology
Major: Data Engineering
Examiner: Prof. Joni-Kristian Kämäräinen, Dr. Ke Chen
Keywords: Low-Resolution, Knowledge Transfer, Deep Learning, Convolutional Neural Network, Single Image Super-Resolution, Fine-Grained Classification

Successful fine-grained image classification methods learn subtle details between visually similar (sub-)categories, but the problem becomes significantly more challenging if the details are missing due to low resolution. Alternatively, encouraged by the recent success of Fully Convolutional Neural Network (FCNN) architectures in single image super-resolution, we propose a novel Resolution-Aware Classification Neural Network (RACNN). More precisely, we combine convolutional image super-resolution and convolutional fine-grained classification together in an end-to-end cascade manner, which first improves the resolution of low-resolution images and then recognises objects in the images. Extensive experiments on the Stanford Cars, Caltech-UCSD Birds 200-2011 and Oxford 102 Category Flowers benchmarks demonstrate that the proposed model consistently performs better than conventional convolutional models on categorising fine-grained object classes in low-resolution images.

PREFACE

I completed my thesis in Laboratory of Signal Processing at Tampere University of Technology with the help of people from this Lab. Besides, apart from this Master's thesis, an academic paper based on this work has been published in October 2017. In the first place, I greatly appreciate my supervisor Professor Joni-Kristian Kämäräinen and my advisor Doctor Ke Chen, without their supervision, I would not probably finish this thesis and publish my first academic paper in this year. Prof. Joni-Kristian Kämäräinen is full of kindness and wisdom and offered me sufficient freedom and support to achieve my goal. Besides, Doctor Ke Chen gave me advice and solved my confusion when I got stuck into my thesis, timely and important. Especially, both of them have the biggest contributions to this thesis and the academic paper. I greatly appreciate what they did for me and admire their capabilities. Meanwhile, I want to give my thanks to my colleagues who are working in the same group (not in order), Antti Hietanen, Antti Ainasoja, Dan Yang, Song Yan, Said Pertuz, Nataliya Strokina, Junsheng Fu, Yanlin Qian, Yue Bai and Wenyan Yang. Under the harmonious and happy atmosphere they maintained, I was enjoying the time very much during my thesis work and thus I could smoothly and successfully accomplish my Mater's thesis.

15th November 2017, Tampere, Finland

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS AND NOTATIONS

Abbreviations

ANN	Artificial Neural Network
CNN	Convolutional Neural Network
DCNN	deep Convolutional Neural Network
DeCAF	Deep Convolutional Activation Features
DRCN	Deeply Recursive Convolutional Network
DRRN	Deep Recursive Residual Network
ESPCN	Efficient Sub-Pixel Convolutional network
FSRCNN	Fast Super-Resolution Convolutional Neural Network
HOG	Histogram of Oriented Gradients
HR	High-Resolution
ILR	Interpolated Low-Resolution
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
kerSLRFR	Kernel Synthesis-based Low-Resolution Face Fecognition
LBP	Local Binary Pattern
LR	Low-Resolution
LR-CNN	Low-Resolution Classification Neural Network
MLP	Multiple-Layer Perception
NN	Nearest Neighbour
PCA	Principal Component Analysis
RACNN	Resolution-Aware Classification Neural Network
ReLU	Rectified Linear Unit
RLSR	Relationship-based Super-Resolution
SIFT	Scale Invariant Feature Transform
SISR	Single Image Super-Resolution
SR	Super-Resolution
SRCNN	Super-Resolution Convolutional Neural Network
SVM	Support Vector Machine
PCSRN	Partially Coupled Super-Resolution Network
POOF	Part-based One-vs-One Features
PReLU	Parametric Rectified Linear Unit
VDSR	Very Deep Super-Resolution
VLRR	Very Low-Resolution Recognition

Notations

A^{i}	the output of the i^{th} hidden layer
B	the number of residual block
B_i	the biases of the i^{th} convolutional layer
ce	cross entropy
$*$	the convolutional operation
f	the whole mapping function of network
F_i	the mapping function of the i^{th} convolution layer
f_i	the size of filters of the i^{th} convolutional layer
$L_{\mathrm{ce}}(\cdot)$	cross entropy loss
L_{i}	the i^{th} hidden layer in multiple-layer perception
$L_{\mathrm{ms}}(\cdot)$	mean square loss
ms	mean square
N	the number of image
n_i	the number of filters of the i^{th} convolutional layer
r	the upscaling factor
σ	non-linear activation function
U	the number of layers U in each residual block
$\mathbf{w}_i$	the weights of i^{th} neuron
W_i	the weights of the i^{th} convolutional layer
$\mathbf{x}$	the input of network
$\underline{\mathbf{x}}$	the local patch
X	the input volume
X_i	the i^{th} input image
$\boldsymbol{X}^{\mathrm{HR}}$	high-resolution image
$\boldsymbol{X}^{\mathrm{LR}}$	low-resolution image
$\boldsymbol{X}^{\mathrm{Res}}$	residual image
$\boldsymbol{X}^{\mathrm{SR}}$	super-resolution image
$\mathbf{y}$	the output of network
Y	the output volume
y_i	the ground truth of the i^{th} input
$\hat{y}_i$	the output label

1. INTRODUCTION

1.1 Overview

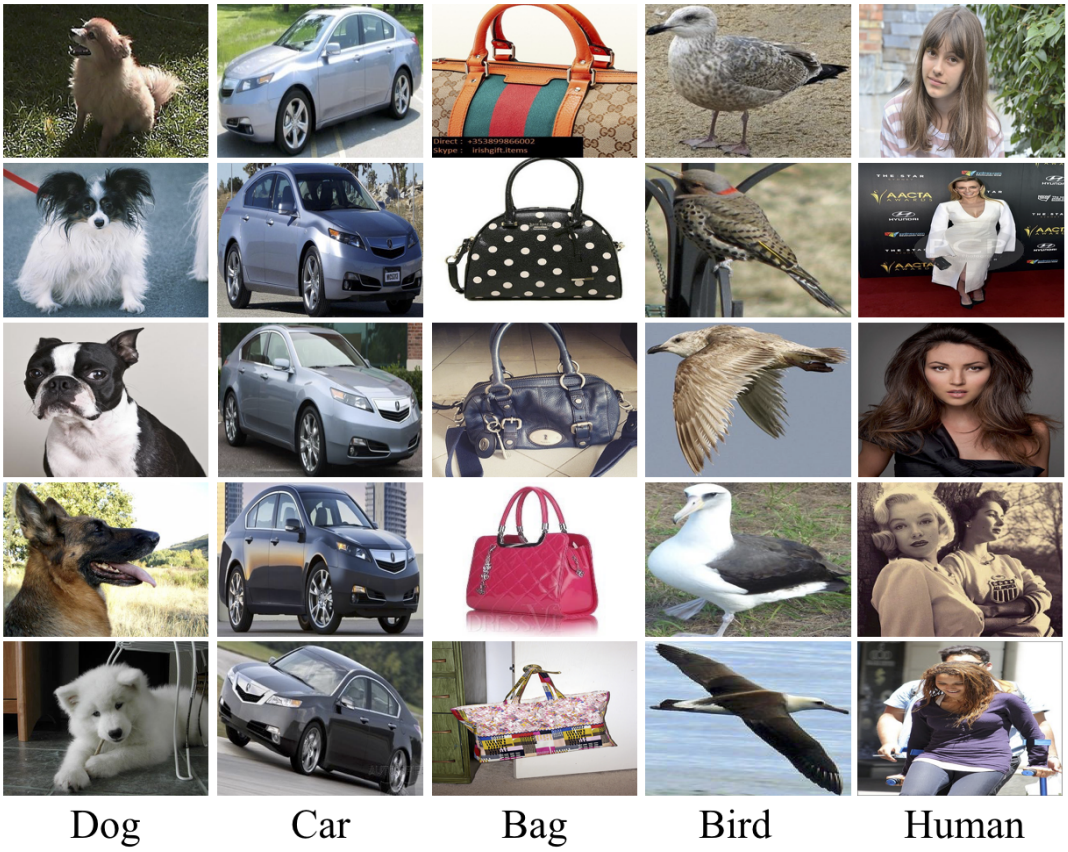


Figure 1.1 *Example images of different categories in ImageNet dataset.*

Image classification is one of the core studies of digital image analysis. The task of image classification is to assign one label to an image according to its semantic content, as shown in Figure 1.1, which has been attracting wide attention in computer vision community. A large number of methods have emerged to cope with the classification task and these methods can be broadly categorised into three groups according to the usage of labelled samples, namely supervised classification, unsupervised classification and semi-supervised classification. The supervised classification

Figure 1.2 *Different breeds of dogs [48] from the same parent category: Dog.*

techniques are the most commonly used nowadays and they require a number of pre-labelled samples as the training data to train the classifiers, Popular classifiers are Support Vector Machines [10, 14], Artificial Neural Networks [17, 29, 57], Decision Tree [36], Rondom Forest [7], K-Nearest Neighbours [34, 18, 37], *etc*. Unsupervised techniques do not require labelled data but are able to classify images by exploring the structure and relationship between the images. In other words, unsupervised classification conceptually is kind of clustering analysis where observations are categorised into the same class if they share some similar contents. Popular techniques for unsupervised ckassification are K-Means Clustering [64], Self-Organised Map [52] and ISODATA Clustering [3]. Semi-supervised classification techniques utilise both labelled data and unlabelled data to build classifiers and take advantages of both supervised and unsupervised techniques especially when there are no sufficient labelled samples avaliable to train the classifiers [11, 33].

Generally speaking, the typical image classification can be defined as cassification at a basic level (*e.g.* dog, automobile, bag, bird, human), as shown in Figure 1.1, furthermore, an increasing number of studies focus on fine-grained visual object classification. Fine-grained object classification [8, 9, 25, 48, 56, 66, 93] classifies objects at a subordinate level under the same parent category, such as the species

of animals [48, 93] or plants [69], the models of man-made objects [56, 66]. Fine-grained classification is more difficult than the ordinary classification task due to the visual and semantic similarity among the subcategories. Subcategories are basically different, but partially share common local structures (*e.g.* nose, fur) as can be observed in Figure 1.2. In this case, the problem of fine-grained classification lies on the subtle differences between similar classes whose fine details play a crucial role in distinguishing their catogories. As a consequence, many methods [2, 9, 25, 40, 65, 69, 104, 109] have been proposed to address this problem and achieved state-of-the-art performance by exploiting on global image statistics [69] or strong local features [104]. Since emergence of Convolutional Neural Network (CNN) architecture [57] and massive public datasets [56, 93], the CNN-based fine-grained image classification methods [1, 8, 15, 54, 61, 107] have dramatically improved the accuracy by a large margin thanks to the capacity of millions learning parameters and today CNN-based methods are the dominant approach in fine-grained image classification.

1.2 Motivation

On the one hand, the high performance achieved by aforementioned CNN-based approaches is mainly based on good quality and relatively high-resolution (HR) images (*e.g.* AlexNet[57] requires 227×227). On the other hand, the performance can collapse when it comes to low-resolution (LR) fine-grained images classification [16, 62], since there are more fine details provided in HR images as compared to LR images, which means that subtle discriminative features for classification are easier to extract from HR images than their LR counterparts. Therefore, the problem becomes more challenging when there are no HR images available or fine-grained examples are small in the images. In this case, the accuracy of fine-grained classification is affected due to lack of fine details. In this setting, the challenge intuitively raises from the problem of how to recover discriminative texture details from LR images. In this work, we attempt to adopt single image super-resolution (SISR) techniques [13, 23, 30, 102, 106] to recover fine details. Inspired by recent state-of-the-art performance achieved by novel CNN-based image super-resolution methods [23, 49], we apply image super-resolution convolutional neural network (SRCNN) to refine the texture details of fine-grained objects in LR images. In particular, we propose a unique end-to-end deep learning framework that combines CNN-based image super-resolution and fine-grained classification – a resolution-aware classification neural network (RACNN) for fine-grained object classification in LR images. To our best knowledge, our work is the first end-to-end learning model for low-resolution fine-grained object classification.

1.3 Summary

Contributions – Our contributions are three-fold:

- Our work is the first attempt to utilise super-resolution specific convolutional layers to improve convolutional fine-grained image classification in an end-to-end manner.
- The high-level concept of our method is generic and super-resolution layers or classification layers can be replaced by any other CNN-based super-resolution networks or classification frameworks, respectively.
- We experimentally verify that the proposed RACNN achieves superior performance on low-resolution fine-grained images which make ordinary CNN collapse.

Our main principle is simple: the higher image resolution, the easier for classification. Our research questions are: Can computational super-resolution recover details required for fine-grained image classification and can such SR layers be added to an end-to-end deep classification architecture? To this end, our RACNN integrates deep residual learning for image super-resolution [49] into typical convolutional classification networks (e.g AlexNet [57], VGGNet [81] or GoogLeNet [85]). On one hand, the proposed RACNN has deeper network architecture (i.e more network parameters) than the straightforward solution of conventional CNN on upsampled images. Our RACNN learns to refine and provide more texture details for low-resolution images to boost fine-grained classification performance. We conduct experiments on three fine-grained benchmarks, the Stanford Cars Dataset [56], the Caltech-UCSD Birds-200-2011 [93] and the Oxford 102 Flower Dataset[69]. Our results answer the aforementioned questions: super-resolution improves fine-grained classification and SR-based fine-grained classification can be designed into a supervised end-to-end learning framework, as depicted in Figure 1.3 illustrating the difference between RACNN and conventional CNN.

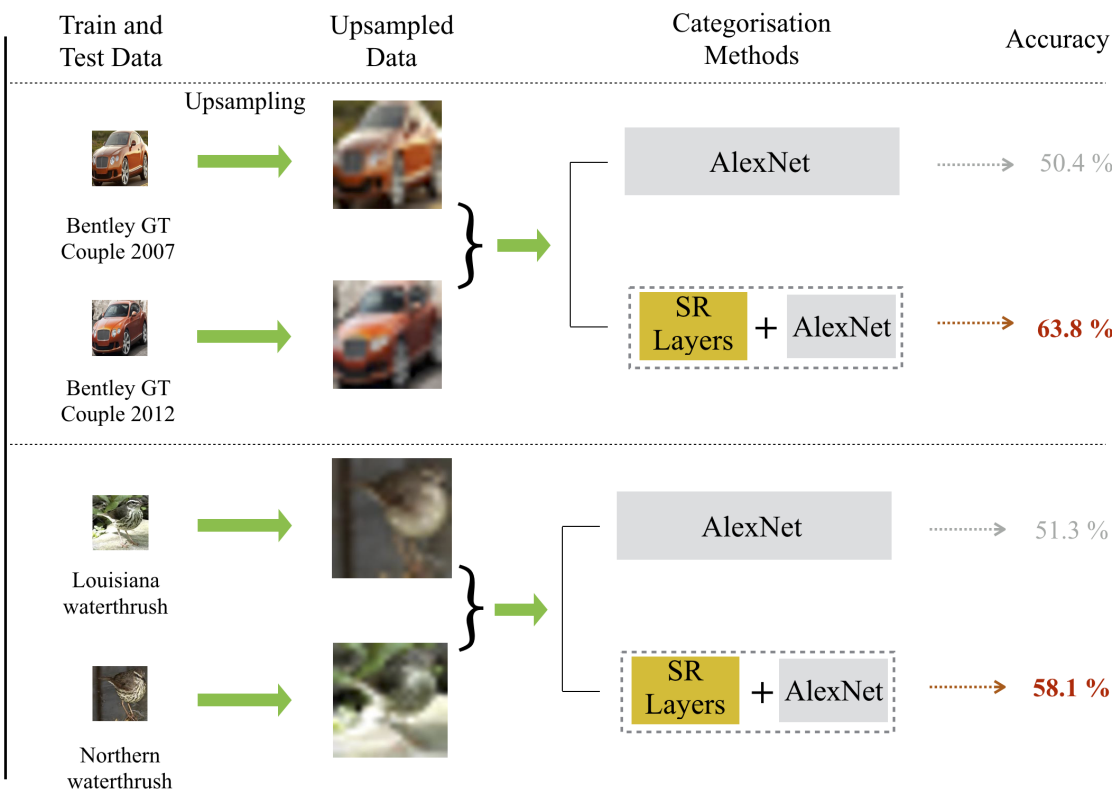


Figure 1.3 *The comparison of conventional AlexNet (the gray box) and our proposed $RACNN_{AlexNet}$ (the dashed box) on Standford Cars Dataset and Caltch-UCSD Birds 200-2011 Dataset. Owing to the introduction of the convolutional super-resolution (SR) layers, the proposed deep convolutional model (the dashed box) achieves superior performance for low resolution images.*

2. LITERATURE REVIEW

We first present the problem of general fine-grained object classification and then further step into the literature focusing on low-resolution image classification. Next, single image super-resolution techniques, especially the CNN-based, are investigated. In the end, deep convolutional neural networks and transfer learning techniques are discussed. Note that this part mainly focus on image super-resulution and convolutional neural network since we concentrate on building a deep end-to-end CNN-based framework for low-resolution image classification by integrating image super-resolution techniques.

2.1 Fine-Grained Image Classification

Fine-grained classification is a sub-field of image classification, which refers to classifying objects into sub-categories within the same parent category, such as the breeds of birds [8, 93], the species of flowers [69] and the models of cars [56, 76]. A variety of approaches have been proposed for discriminating the fine-grained classes in recent years [28, 48, 58, 74]. The prior research on fine-grained classification roughly involves two procedures: discriminative parts localisation and fine-grained feature extraction. The first step is to identify discriminative regions in images by using geometric constraints, which can be achieved by either using part-based bounding boxes to explicitly train a strongly supervised region detector [12, 53, 107, 109] or implicitly detecting the discriminative parts in unsupervised or weakly supervised fashion [32, 45, 55, 54, 61]. The motivation to localise the discriminative regions in the image is based on the assumption that some fine-grained classes share similar structures or appearance, like noses, heads and legs for dog breeds. To this end, these localised regions are beneficial for discovering discriminative localised features which are crucial to discriminate fine-grained classes. The second step is to extract discriminative and robust features for fine-grained object classification. Some previous approaches [4, 5, 6, 25, 44, 103, 108] employ traditional hand-crafted feature descriptors, such as Histogram of Oriented Gradients (HOG) [20], Local Binary Pattern (LBP) [70], Color Histogram [92] and Scale Invariant Feature Transform (SIFT) [63] to make best utilisation of edge, texture and colour information presented in

images to discriminate fine-grained objects. The Part-based One-vs-One Features (POOF) based on HOG have been successfully employed for fine-grained classification [4, 5, 6], for instance. More recently, owing to the success of deep convolutional neural network (DCNN) architectures [57] on large-scale image classification, deep CNN-based features have shown superiority over hand-crafted features on general image classification as well as fine-grained classification [8, 28, 53, 55, 98, 107]. While the typical pipeline for conventional fine-grained classification roughly comprises of three separate procedures, parts localisation, feature extraction and classification, the emerged DCNNs have turned out to be capable of jointly optimising the whole pipeline which results in significant improvement on object classification tasks. For example, [2, 54, 61] have driven the fine-grained image classification to its state-of-the-art performance in various fields, such as plants [2], birds [93] and cars [56].

2.2 Low-Resolution Image Classification

Research on general image classification has achieved substantial achievements which are often based on the assumption that objects in images are of relatively high resolution [28, 53]. However, this assumption does not always hold in practice. For instance, images are taken from distance or surveillance videos where objects in the images are usually very small [91]. However, only a few works have paid attention to low-resolution image classification [75, 79, 95, 110]. Wang *et al.* [95] study very low-resolution (*e.g.* 8×8) recognition (VLRR) problem starting from the simplest CNN baseline to evolve their network and a final partially coupled super-resolution network (PCSRN) was proposed. The proposed PCSRN jointly learns a VLRR model from both LR and HR training images and then applies the learned model to directly classify LR images. In [110], a novel relationship-based super-resolution (RLSR) method is proposed to reconstruct the HR face image by learning the relationship from the very LR space to the HR space under visual quality and discriminative constraints, then several classic facial classification algorithms (*e.g.* PCA + SVM and PAC + 1NN) are employed to classify the super-resolved face images. Shekhar *et al.* [79] propose a generative approach called kernel synthesis-based LR face recognition (kerSLRFR) which is robust to LR face images classification under different illumination conditions. The proposed kerSLRFR first utilises HR training images to generate multiple LR facial images of the same person with various illumination and then applies synthesised LR images to learn the kernel dictionary algorithm for recognising LR face images.

However, all the aforementioned approaches have a strong assumption that HR images of each class are available during the training phase. In addition, the same

assumption also occurs in Peng's work [75] that studies LR fine-grained classification using deep convolutional neural network which is closely relevant to ours. In [75], they propose a novel fine-to-coarse staged training procedure (Staged-Training) using popular pre-trained AlexNet [57], which effectively transfers fine-to-coarse knowledge from HR training images to the LR testing domain. In the first stage, the Staged-Training AlexNet [75] uses HR fine-grained images to train the network which can learn fine discriminative features in the HR domain (*i.e.* 227×227). In the second stage, these HR images are downscaled to LR domain (*i.e.* 50×50) using bicubic interpolation [47] and the pre-trained network is fine-tuned on the downscaled LR images to learn the discriminative features from fine to coarse. On the contrary, Chevalier *et al.* [16] design a CNN-based fine-grained LR image classifier (LR-CNN) with respect to varying image resolutions, which is both trained and tested exclusively on LR images.

2.3 Single Image Super-Resolution

To address the problem of LR image classification, one of the most commonly used techniques is single image super-resolution (SISR) which refers to reconstructing the HR image from a given LR counterpart. A large number of SISR techniques recently have been proposed with various assumptions and can be rougthly categorised into two groups based on their tasks. Generic SISR algorithms [19, 24, 30, 41, 49, 50, 43, 78, 87, 89, 90, 94, 101, 102] are developed for all sorts of images which are not limited to specific domains, while domain-specific SISR algorithms mainly focus on specific categories of images like faces [88, 99], scenes [84], *etc.*

Yang *et al.* [100] grouped existing SR algorithms into four types according to image priors: namely interpolation-based methods, statistic-based methods, edge-based methods and example-based methods. Interpolation-based methods [43, 47] utilise predefined mathematical formulas to generate HR image from LR image without any training data. Bicubic and biliner intepolations weightedly average neighbouring pixel values of LR image to produce HR pixel intensities, which can effectively reconstruct the low-frequency (smooth) regions but fail in high-frequency (edge) regions. Image statistic-based methods [42, 51] utilise inherent properties of natural images as priors to produce HR images from LR images, like sparsity property and total variation. Edge-based methods [26, 83] attempt to reconstruct HR image using image priors (*e.g.* the depth and width) learnt from edge features and usually yield high-quality edges in reconstructed HR images with reasonable sharpness and artifacts. Patch-based or example-based methods are the predominant techniques for SISR and numerous example-based approaches [19, 23, 27, 30, 41, 78, 101, 106]

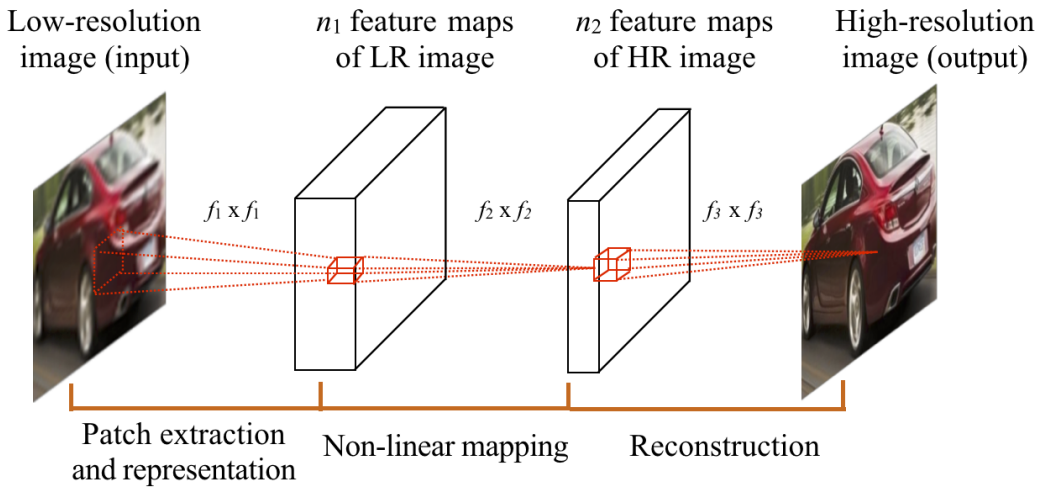


Figure 2.1 *Super-resolution convolutional neural network (SRCNN) proposed by Dong [23]. In SRCNN, the first convolutional layer extracts a set of features of LR image* $\mathbf{x}$*, the second convolutional layer nonlinearly maps these features from LR space to HR space and the last convolutional layer reconstructs these features within HR space to produce the final HR image* $\mathbf{y}$*.*

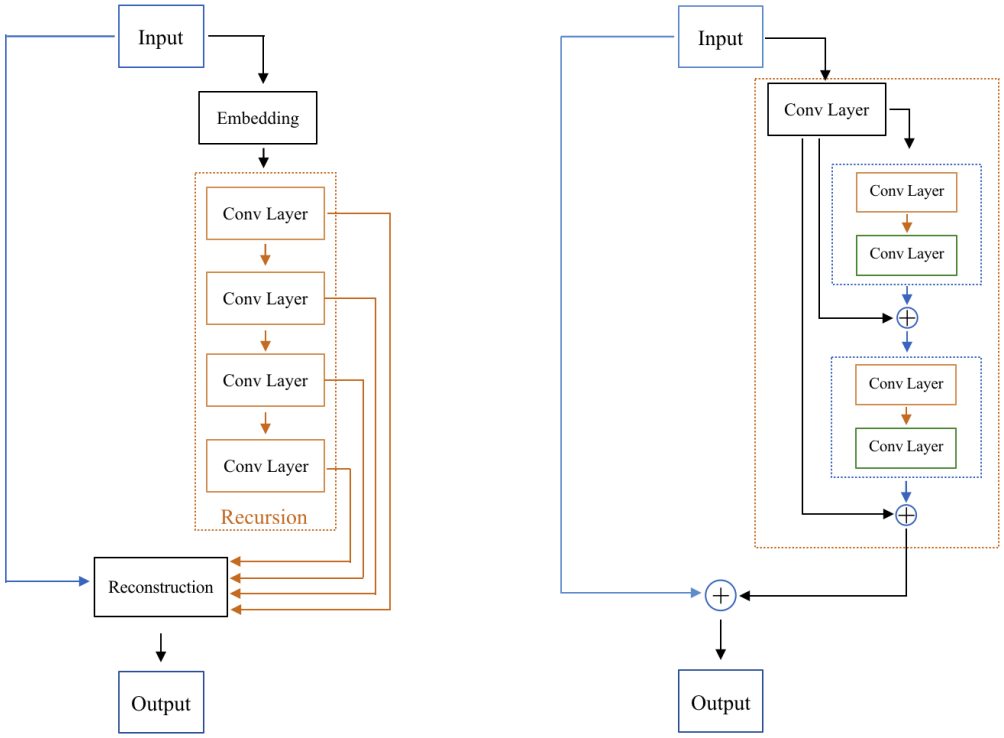
have emerged in the last decade. Training patches are cropped from the training pairs of LR and HR images so that the mapping functions from LR space to HR space can be learnt using these cropped training patches. According to the source of training patches, the mainstream example-based methods can be classified into two main categories: external database driven SR methods and internal database driven SR methods. The internal example-based approaches [27, 30, 41] super-resolve the LR images by exploiting the self-similarity property and generating exemplar patches from the input image itself, while the external example-based approaches [19, 89, 90, 101, 106] use a variety of learning algorithms to learn the mapping between LR and HR patch pairs from external database, such as sparse coding based SR [102], random forest SR [78] and CNN-based SR [23]. In the following, as the key component of this work, CNN-based SISR is investigated in details.

Recently, Convolutional Neural Network has been adopted for single image super-resolution and has achieved state-of-the-art performance. The first attempt using CNN for image SR is Super-Resolution Convolutional Neural Network (SRCNN) proposed in [23] and it contains three fully convolutional layers to learn a nonlinear mapping between LR and HR patches, as illustrated in Figure 2.1. SRCNN requires interpolated LR (ILR) image as the input and implicitly performs three operations in an end-to-end fashion. The first convolutional layer operates n_1 filters with receptive size $f_1 \times f_1$ pixels on the input image to extract the underlying representations in

the ILR space. The second layer operates as a non-linear feature mapping from ILR space to HR space, which is achieved by applying n_2 filters with receptive size $f_2 \times f_2$ on the extracted ILR representations. The last layer reconstructs the feature representations in HR space to generate the HR image using n_3 filter(s) with receptive size $f_3 \times f_3$ to aggregate the representations. SRCNN achieved state-of-the-art performance by jointly optimising all the layers in an end-to-end learning.

Inspired by SRCNN, numerous CNN-based SR approaches have emerged [49, 50, 59, 86, 96] and these follow-ups build deeper and more complex structures by stacking more convolutional layers to yield more accurate inference. Kim *et al.* [49] propose a very deep SR network (VDSR) which is similar to SRCNN, except that VDSR attempts to learn the mapping between ILR image and its residual image (*i.e.* the difference between ILR and HR image) rather than directly from ILR to HR to speed up CNN training for very deep network structure via utilising residual learning and adjustable gradient clipping. The VDSR stacks 20 weight layers with the same receptive size of 3×3 and number of filters 64 for each layer. Unlike SRCNN that only has three fully convolutional layers, the VDSR is capable of performing global residual learning. Meanwhile, in order to control the network parameters, Kim *et al.* [50] propose another deeply recursive convolutional network (DRCN) which adopts a deep recursive layer to avoid adding new weighting layers. Motivated by the observation that introducing more parameters through adding more weight layers leads model to be overfitted [82], the DRCN is capable of addressing this problem via adding the same layers recursively by sharing the same weights without introducing new parameters. To this end, the DRCN consists of 20 layers in total, which can be viewed as three parts, as shown in Figure 2.2(a). The first part is the embedding layer which extracts the feature maps from a input given image. Next, the feature maps are fed into the recursive part which stacks recursive layers with shared weights among these layers for inference. Finally, the reconstructing layer assembles the input image ILR and all the intermediate outputs of recursive layers to produce the final HR image.

Furthermore, a much deeper network is proposed recently in [86] which takes advantage of DRCN [50] and VDSR [49] to build a deep recursive residual network (DRRN) with depth even up to 52 layers, which is capable of capturing global and local details as well as decreasing network parameters by introducing recursive residual blocks. Instead of stacking a single layer, DRRN recursively stacks a residual block comprising of several layers, as illustrated in Figure 2.2(b). Nevertheless, DRRN has two important parameters: the number of layers U in each residual block and the number of residual block B. Interestingly, when $U = 0$ and $B = 18$ DRRN becomes VDSR, which means DRRN is a more generic framework of VDSR or VDSR



(a) DRCN (b) DRRN

Figure 2.2 *Simplified structures of (a) DRCN [50] and (b) DRRN [86]. In DRCN, the red dashed box refers to recursive module, among which each convolutional layer shares the same weights, and the blue line refers to global identity mapping. In DRRN, the blue dashed box refers to residual block , among which there are two convolutional layers without sharing weights, but the red dashed box is the recursive module, among which each residual block shares the same weights with respect to corresponding convolutional layers. The same as in DRCN, the blue line refers to the global identity mapping.*

is a special case of DRRN [86]. To this end, DRRN robustly boosts the performance of SR further by making utilisation of the global residual learning of VDSR and the reduction of parameters of DRCN, as well as the local residual learning of residual blocks.

On the contrary, instead of using the interpolated LR image as input which requires expensive computation, the works of [24, 80] directly super-resolve LR image without any interpolation. Subsequently, they turn out that enabling the networks to directly learn the feature maps in LR space and then upscale the LR image can further boost the performance of accuracy and speed. in [24], they propose a fast super-resolution convolutional neural network (FSRCNN) which adopts a deconvolution operation in the last layer to replace bicubic interpolation [47], as shown in Figure

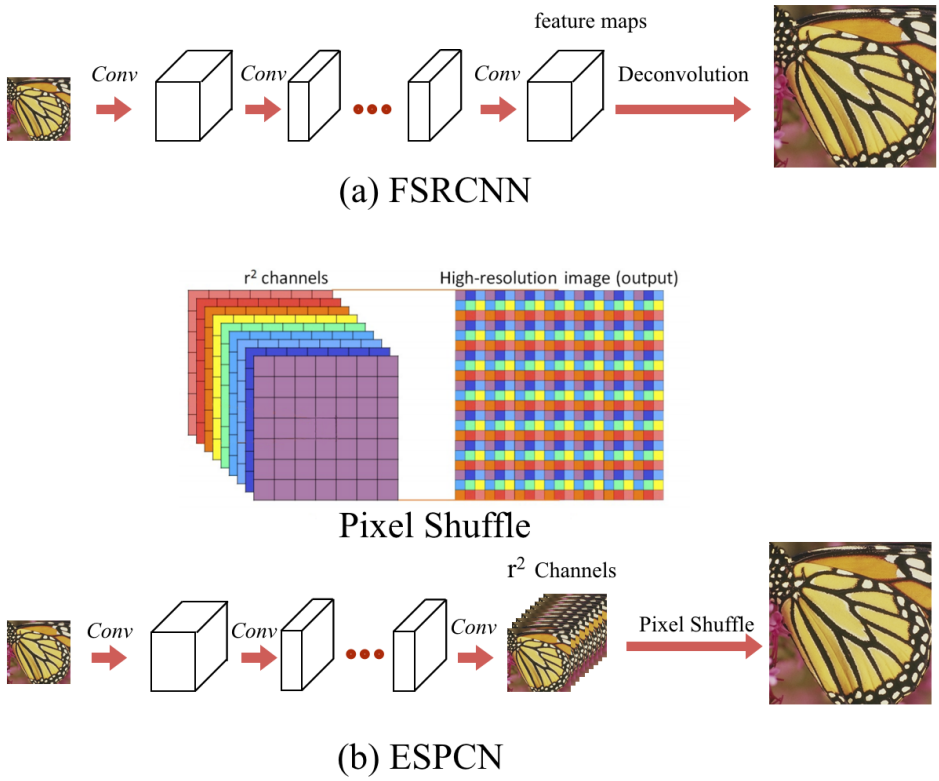


Figure 2.3 *The network structures of FSRCNN [24] and ESPCN [80]. (a) FSRCNN directly learns a deconvolutional layer in the last of network to produce HR image rather than bicubic interpolation. (b) As same as FSRCNN, ESPCN also learns the feature maps in LR space except that ESPCN performs pixel shuffle to reconstruct the HR image instead of deconvolution.*

2.3 (*a*). Alternatively, an effective sub-pixel convolutional neural network (ESPCN) is presented in [80], whose goal is to learn r^2 (where r denotes the upscaling factor) variants of the input LR image only in LR space and then shuffle the pixels to reconstruct the HR counterpart, as depicted in Figure 2.3 (*b*). Literally, the r^2 variants of the LR image learned by the network can be deemed as r^2 pixel-wise downsampled LR images of the HR image, which can be viewed as the inverse process of pixel-shuffling.

2.4 Deep Convolutional Neural Network

With respect to the remarkable success achieved by deep convolutional neural networks (DCNN) in computer vision community in the past few years, in this section, we investigate the relevant powerful DCNN-based techniques used in our work.

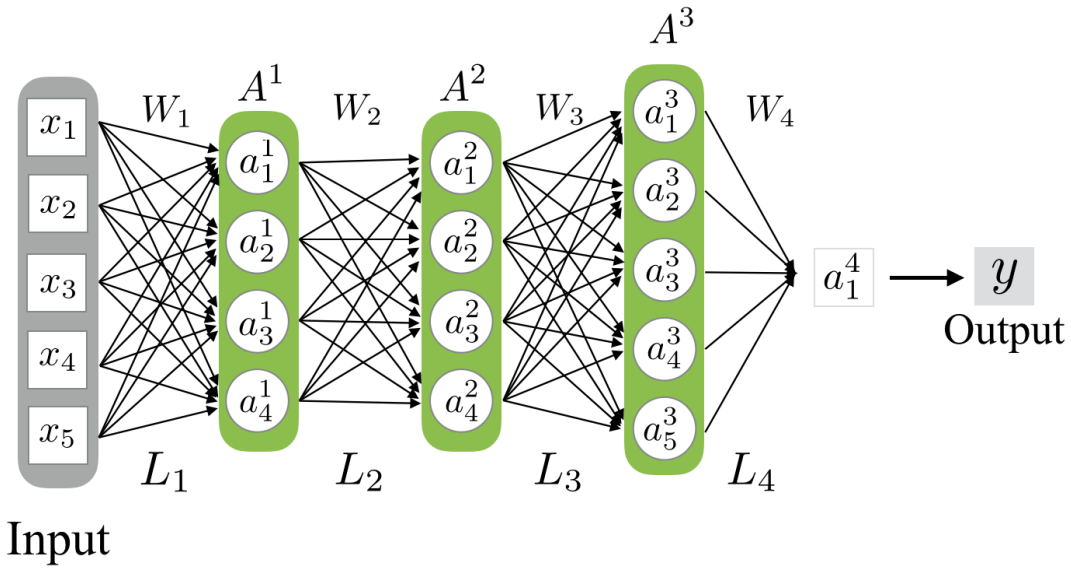


Figure 2.4 *A simple MLP structure with an input layer, four fully-connected layers and an output layer, where* $\mathbf{x}$ *denotes the input data,* $\boldsymbol{L}_i$ *denotes the* i^{th} *hidden layer,* $\boldsymbol{W}_i$ *and* $\boldsymbol{A}^i$ *denotes the weights and output of the* i^{th} *layer, respectively, and* $\mathbf{y}$ *denotes the final output of the network.*

Convolutional Neural Networks (CNN) belong to the family of Artificial Neural Networks (ANN) which we introduce first. ANN originated from the middle of the 20th century and was firstly created as a computional model based on mathematics for emulating biological neural networks of brain by McCulloch *et al.* [67]. ANN is made of interconnected nodes which analogously perform the activities of brain neurons. A simple multiple-layer perception (MLP) ANN comprising of one input layer L_1, two hidden layers L_2 and L_3 and one output layer L_4 is depicted in Figure 2.4, which can perform a series of non-linear mapping functions from input to the final output. The equation is formulated as below:

$$\begin{aligned} \mathbf{A}^l &= \boldsymbol{\phi}(\mathbf{W}_l^T \mathbf{A}^{l-1}),\ \textit{where}\ \mathbf{A}^0 = \mathbf{x},\ l = 1, ..., L-1 \\ \mathbf{y} &= \mathbf{W}_L^T \mathbf{A}^{L-1}, \end{aligned} \tag{2.1}$$

where $\mathbf{A}^l$ and $\mathbf{W}_l^T$ denote the outputs and the transpose of the weights $\mathbf{W}_l$ of the l^{th} layer, respectively, $\boldsymbol{\phi}$ denotes the non-linear activation functions, L denotes the number of layers of the network, $\mathbf{x}$ and $\mathbf{y}$ denote the input and output of the network. Each layer contains multiple artificial neurons and each neuron non-linearly maps all the input values to a single output value, as shown in Figure 2.5, which can be

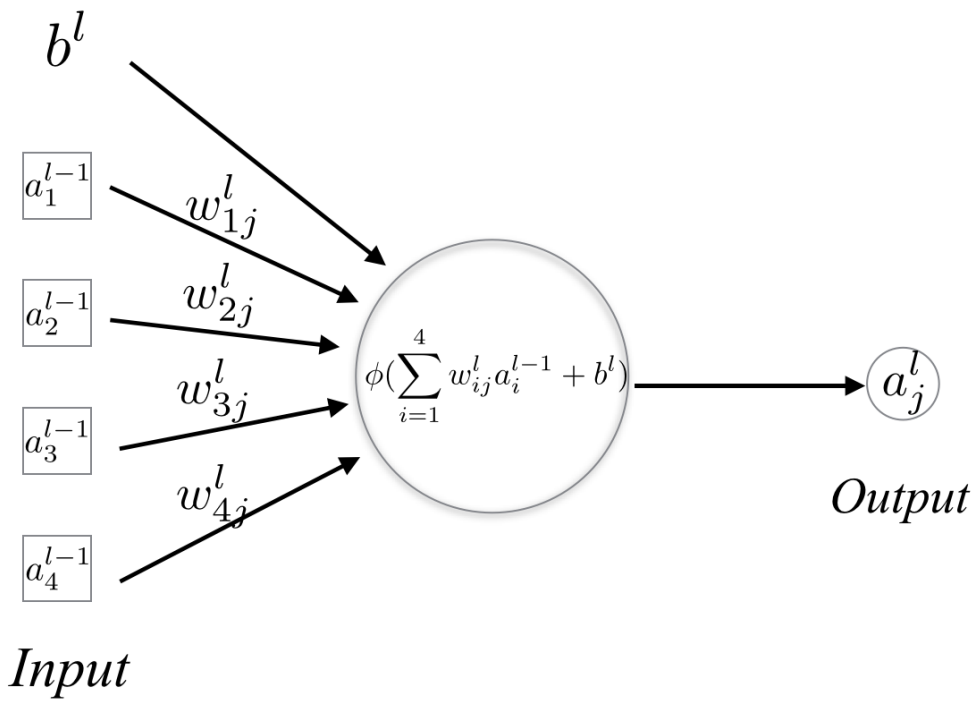


Figure 2.5 *The mathematical model for a single artificial neuron.*

formulated in mathematical equations 2.2:

$$\begin{aligned} a_j^l &= \phi(\sum_{i=1}^{N} w_{ij}^l a_i^{l-1} + b_j^l),\ where\ l = 1, ..., L-1,\ j = 1, ..., K, \\ &= \phi(\sum_{i=0}^{N} w_{ij}^l a_i^{l-1}),\ where\ w_{0j} = 1,\ x_0 = b_j^l,\ l = 1, ..., L-1,\ j = 1, ..., K, \\ &= \phi(\mathbf{w}_j^T \mathbf{a}^{l-1}),\ where\ \mathbf{a}^0 = \mathbf{x},\ l = 1, ..., L-1,\ j = 1, ..., K, \end{aligned} \tag{2.2}$$

where ϕ denotes the non-linear activation function (*e.g.* sigmoid function), $\mathbf{w}_j$ the weights of the j^{th} neuron of the l^{th} layer, $\mathbf{a}^{l-1}$ and a_j^l the inputs and output of the neuron, K and L denote the number of outputs and the amount of layers and $\mathbf{x}$ the input of the network. Thereby, the output for a single fully-connected layer can be presented as below:

$$\mathbf{A}^l = \{a_1^l, a_2^l, a_3^l, ..., a_K^l\}\ \ where\ \ l = 1, ..., L-1, \tag{2.3}$$

where $\mathbf{A}^l$ is the output the l^{th} layer.

One class of the artificial neural networks is Convolutional Neural Network (CNN) which has been successfully applied to visual imagery processing. CNN was initially proposed in [60] to perform handwritten digit recognition and it attempts to spatially

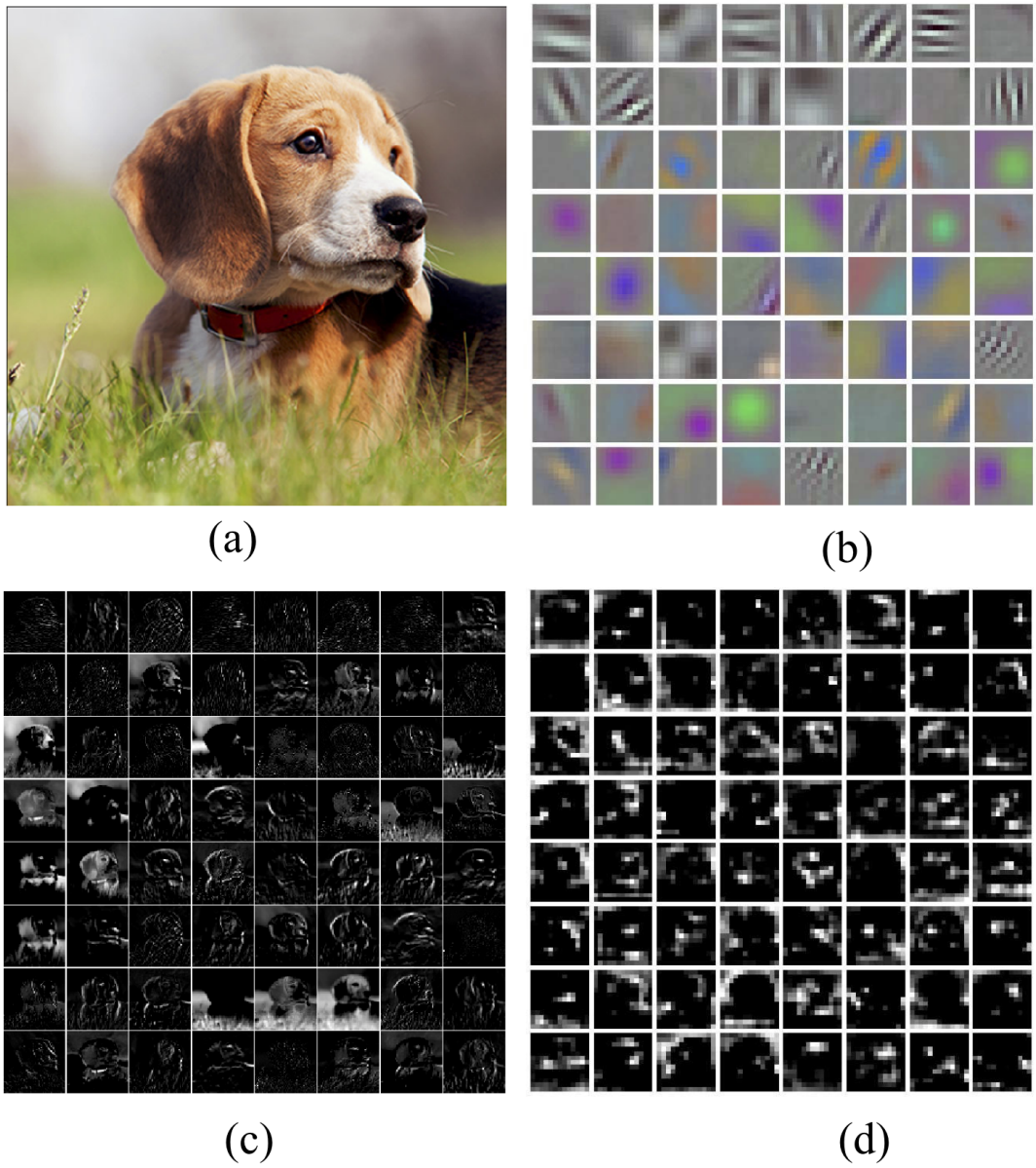


Figure 2.6 *An example of visualisation for AlexNet [57]. (a) The input image. (b) The convolutional filters of the first layer conv1. (c) and (d) are the activation features extracted from the first convolotional layer conv1 and the fourth convolotional layer conv4, respectively. As shown in (c) and (b), most of activation values are close to zeros (black parts) but the silhouette of the dog is visually recognisable in some boxes.*

model high-level abstractions by stacking multiple non-linear convolutional layers in the network. More recently, a big breakthrough for image classification was made by Krizhevsky *et al*. [57] using deep CNN which achieved record-breaking performance in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [77] 2012. CNN-based features performed much better and improved performance by a large margin (*i.e*. error rate of 16.4% vs 26.1%) compared to conventional hand-crafted features. Traditional hand-crafted methods are limited by their ability to capture

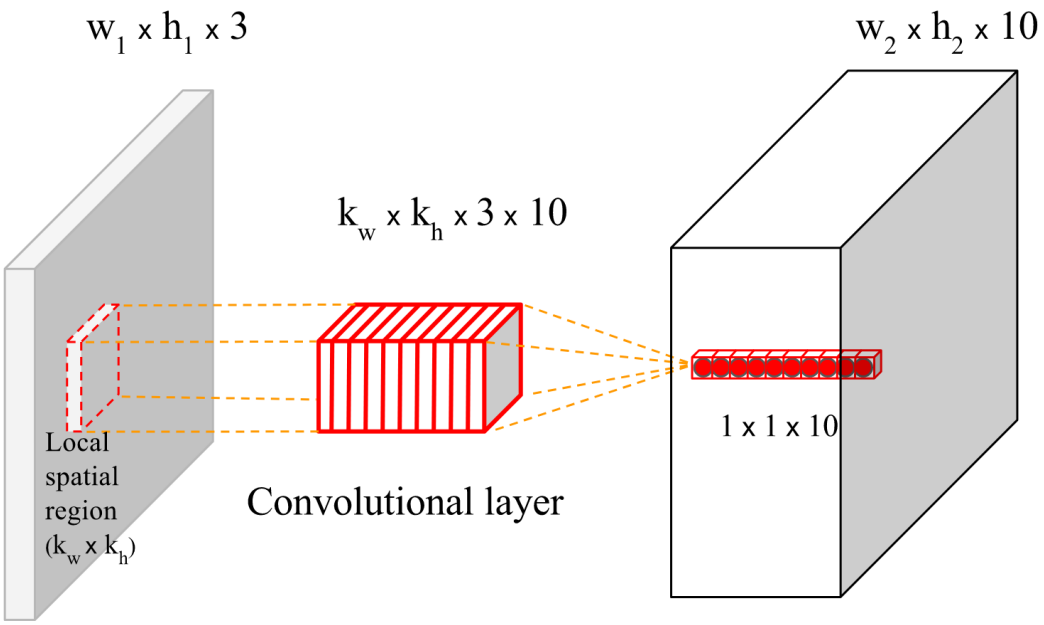


Figure 2.7 *An example of convolutional layer. An input volume (*e.g. *a RGB image with size of $w_1 \times h_1 \times 3$) is convolved by a convolutional layer with* 10 *filters with size of $k_w \times k_n \times 3$ to produce an output volume with size of $w_2 \times h_2 \times 10$. Each of convolutional filter is connected to a local spatial region with full depth (*i.e. *all channels) in the input volume and all the filters (with different weights) look at the same region.*

multiple levels of features. However, by visualising the activation features extracted from intermediate layers of CNN network, it shows that CNN is able to capture salient features of images in different levels [105], as illustrated in Figure 2.6. A typical CNN architecture usually consists of stacked modules, in what follows, we describe several important functional layers commonly used for image classification tasks, namely, convolutional layer, activation layer, pooling layer, fully-connected layer and loss layer.

Convolutional Layer: The convolutional layer is the core component of a CNN and is capable of extracting the salient features by recognising the local correlations in the images. A ConvLayer often comprises of a certain number of filters and each filter contains a set of weights which can be learnt by training the network. An illustrative example of convolutional layer is shown in Figure 2.7. The filters in the convolutional layer are densely connected to the local spatial regions in the input volume and carry out most of the computational tasks. Specifically, each filter in a convolutional layer acts as an artificial neuron which locally performs convolutional operations to obtain the feature map, which is achieved by sliding the weights matrix on the input volume region-by-region vertically and horizontally to carry out mathematical element-wise multiplication. Therefore, each convolutional

filter is applied on the whole input volume and all the subregions share the same weights of the filter, which results in controlling the amount of parameters in the network. The simple example is shown in Figure 2.8 and the equation can be formed as below:

$$\begin{aligned} y &= \sum_{i=1,j=1}^{K_h,K_w} w_{i,j}x_{i,j} + b, \ \textit{where}\ w_{i,j} \in \mathbf{w},\ x_{i,j} \in \underline{\mathbf{x}} \\ &= \mathbf{w} * \mathbf{x} + b, \ \textit{where}\ \ y \in \mathbf{Y},\ \ \mathbf{x} \subset \mathbf{X} \end{aligned} \tag{2.4}$$

$$\mathbf{Y} = \mathbf{w} * \mathbf{X} + b \tag{2.5}$$

where K_h and K_w denote the size of the filter, $\mathbf{X}$ and $\mathbf{Y}$ are the input volume and the output volume, $\mathbf{w}$ and $\underline{\mathbf{x}}$ the filter and local patch in $\mathbf{X}$, $*$ denotes the convolutional operation and b the bias.

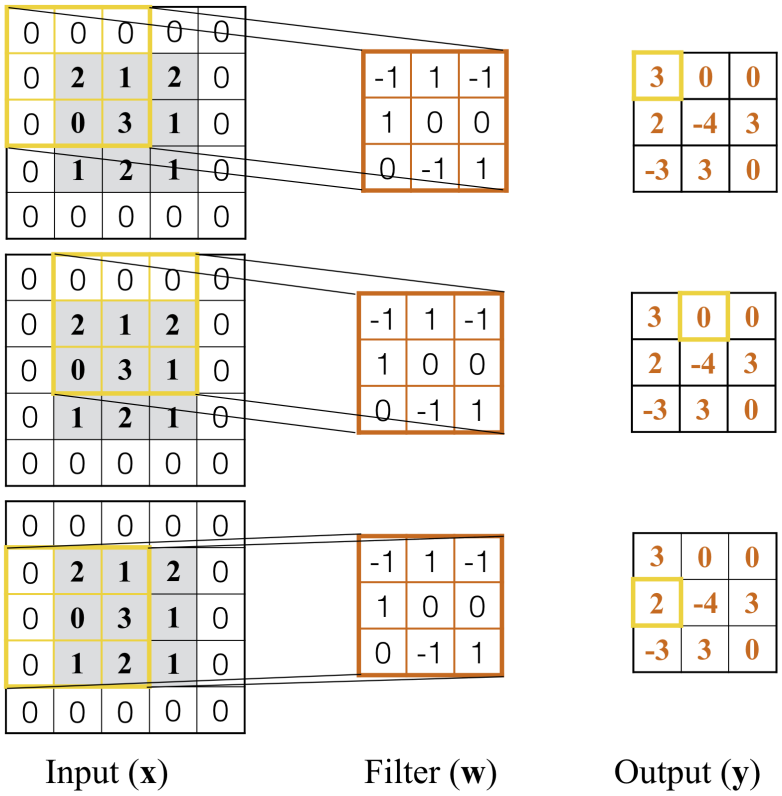


Figure 2.8 *An example of matrix multiplication. Note that the input is a* 3×3 *matrix with zero-paddings which is to obtain an output with the same spatial size.*

Activation Layer: In deep neural network, the non-linearity is basically implemented by activation layer which applies non-linear function on the feature maps. Here, we describe several activation functions commonly used in neural network. *Sigmoid* function constrains real-valued numbers to range between [0, 1] so that large negative numbers become 0 and large positive number become 1, which means

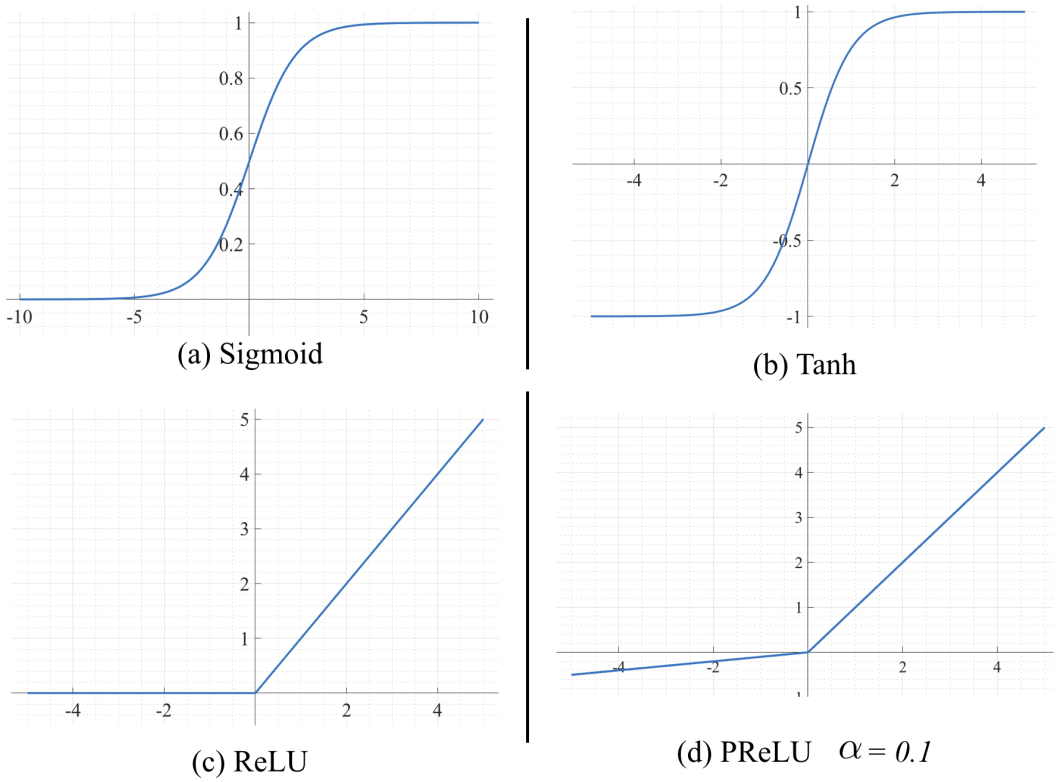


Figure 2.9 *Commonly used activation functions in neural network.*

the activation value is always non-negative. While the hyperbolic tangent function $Tanh$ produces real-valued numbers to range of [-1, 1] and is simply a scaled $Sigmoid$ function, as shown in Equation 2.6 and 2.7. More recently, the Rectified Linear Unit ($ReLU$) [68] has become the most popular activation function used in neural network. $ReLU$ simply constrains the negative numbers to zeros and keeps positive numbers unchanged and its equation is shown in Equation 2.8. In addition, the Parametric Rectified Linear Unit ($PReLU$) is introduced in [38] to generalise the ordinary $ReLU$ activation function, which allows a parameter α to be learnt along with other network parameters for negative numbers, as formulated in Equation 2.9. In addition, these activation functions are illustrated in Figure 2.9.

$$sigmoid : \sigma_s(x) = \frac{1}{1 + e^{-x}} \tag{2.6}$$

$$\begin{aligned} tanh : \sigma_t(x) &= \frac{1 - e^{-2x}}{1 + e^{-2x}} \\ &= \frac{2}{1 + e^{-2x}} - 1 \\ &= 2\sigma_s(2x) - 1 \end{aligned} \tag{2.7}$$

$$relu : \sigma_r(x) = max(0, x) \tag{2.8}$$

$$prelu : \sigma_p(x) = max(0, x) - \alpha\ max(0, -x) \tag{2.9}$$

Pooling Layer: Pooling operation aims to aggregate spatial features and reduce the size of representation so as to reduce the number of parameters and the cost of computation in the network. Max-pooling and average-pooling are the most widely-used pooling functions which explore the representation over small local regions to generate statistical features. In other words, the same representational features are likely to be applicable in different subregions of the image.

Fully-connected Layer: In a fully-connected layer, neurons have full connections to the outputs of previous layer, that is, each single neuron of a fully-connected layer is connected to all the activations from the previous layer. A typical fully-connected layer was demonstrated by the MLP neural network in Figure 2.4. Unlike the weight-sharing scheme in convolutional layer, fully-connected layer requires full connections to the input volume so that fully-connected layer usually has much more parameters compared to convolutional layer.

Loss Layer: One of the essential components in neural network is the loss layer which drives the neural network to learn the objectives from massive training data. The loss (error) between the output of network and the true label is calculated by loss function and then utilised to supervise the training process of the network via back-propagation [60]. There are various loss functions used in neural network such as contrastive loss [35], cross-entropy loss and euclidean distance loss (also known as mean squared error), more specifically, contrastive loss enables the network to learn the parameters which are capable of gathering the neighbour data but separating non-neighbour data, and cross-entropy loss is adopted to measure the performance of a classification network which produces a probability distribution of predicted class over all classes, while euclidean distance loss simply measures the difference between the predicted output and ground truth.

2.5 Transfer Learning

Deep neural networks often contain millions of parameters due to various deep structures, which usually requires a huge amount of training data to train the network. However, for some domain-specific tasks (*e.g.* fine-grained classification), sometimes, there are no enough training data available to enable loss function converge at a good minimum but to overfit the network. To mitigate this difficulty, the technique of transfer learning [31] is employed to make the best utilisation of existing datasets like ImageNet (containing millions of images with 1000 categories) to assist

the training of domain-specific tasks. In simple terms, transfer learning (also known as domain adaptation) aims to adapt the knowledge from source domain with a large dataset to target domain with a small dataset. [72] shows that transfer learning can boost the performance of the target task, even though the feature spaces or topics are different between source domain and target domain. Donahue *et al.* [22] present that deep convolutional activation features (DeCAF) pre-trained on ImageNet can be adapted to generic object classification tasks and achieved fairly good results on different domains, such as fine-grained classification on Caltech-UCSD-Birds 200 [93] and scene recognition on SUN-397 [97]. Furthermore, in [71], Oquab *et al.* successfully demonstrate that transfer learning can be applied on object detection and localisation by fine-tuning the convolutional layers pre-trained on ImageNet for classification.

3. METHODOLOGY

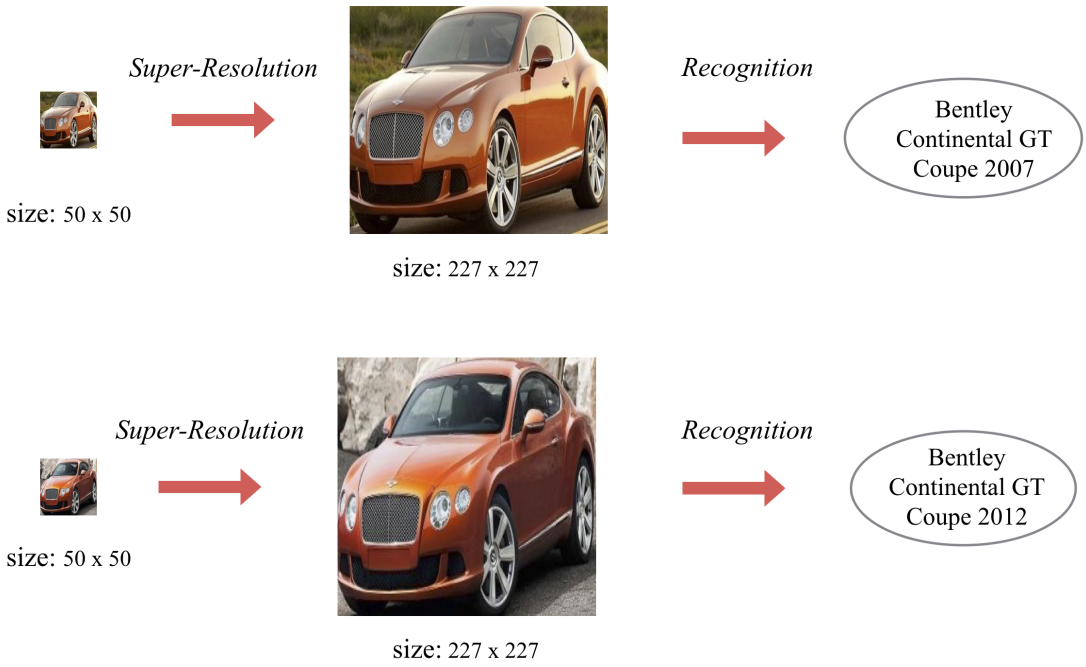


Figure 3.1 *The pipeline for fine-grained low-resolution image classification.*

To tackle the problem fine-grained classification in low-resolution (LR) images, an intuitive and simple idea is to increase the resolution of LR images first and then recognise the objects in the images, as shown in Figure 3.1. It literally comprises two procedures, namely, image super-resolution and classification. To this end, we propose a novel resolution-aware classification neural network which involves image super-resolution convolutional neural network and image classification convolutional neural network.

3.1 Fully Convolutional Super-Resolution Network

In this section, we present a fully convolutional super-resolution network. The goal of the super-resolution network is to recover texture details of low-resolution images to feed them into the following image classification network.

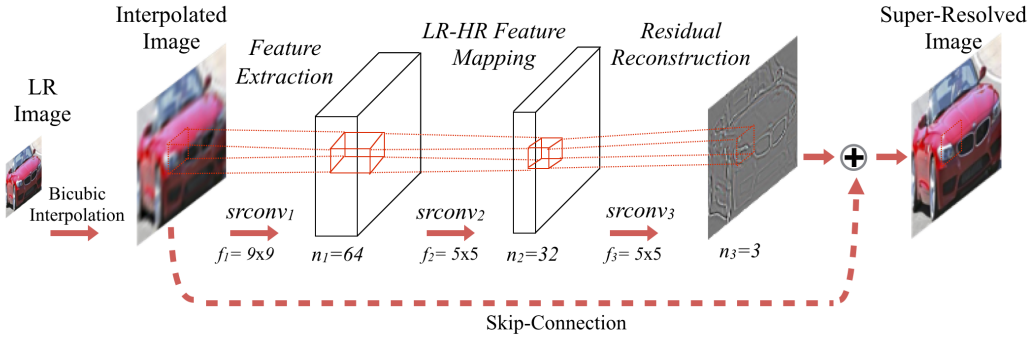


Figure 3.2 *The structure of residual super-resolution convolutional neural network.*

Given K training pairs of low-resolution and high-resolution images $\{\boldsymbol{X}^{\mathrm{LR}}, \boldsymbol{X}^{\mathrm{HR}}\}^i, i = 1, 2, \cdots, K$, a direct CNN-based mapping function $g(\boldsymbol{X}^{\mathrm{LR}})$ from $\boldsymbol{X}^{\mathrm{LR}}$ (input observation) to $\boldsymbol{X}^{\mathrm{HR}}$ (output target) [23, 24] is learned by minimising the mean square loss

$$L_{\mathrm{ms}}(\boldsymbol{X}^{\mathrm{LR}}, \boldsymbol{X}^{\mathrm{HR}}) = \frac{1}{2}\sum_{i=1}^{K} \|\boldsymbol{X}^{\mathrm{HR}} - g(\boldsymbol{X}^{\mathrm{LR}})\|^2. \tag{3.1}$$

Inspired by Super-Resolution Convolutional Neural Network (SRCNN) [23] and the more recent state-of-the-art residual super-resolution convolutional network VDSR [49], we design our convolutional super-resolution layers as shown in Figure 3.2. Similar to [49], instead of directly minimising the loss function in Equation 3.1, our convolutional super-resolution network learns a mapping function from the interpolated LR images $\boldsymbol{X}^{\mathrm{LR}}$ to residual images $\boldsymbol{X}^{\mathrm{Res}} = \boldsymbol{X}^{\mathrm{HR}} - \boldsymbol{X}^{\mathrm{LR}}$. Thus, the loss function of the proposed convolutional super-resolution network is as the following:

$$\min \quad \frac{1}{2}\sum_{i=1}^{K} \|\boldsymbol{X}^{\mathrm{Res}} - g(\boldsymbol{X}^{\mathrm{LR}})\|^2, \quad \textit{where } \boldsymbol{X}^{\mathrm{Res}} = \boldsymbol{X}^{\mathrm{HR}} - \boldsymbol{X}^{\mathrm{LR}}. \tag{3.2}$$

The better performance of residual learning yields from the fact that, since the input ($\boldsymbol{X}^{\mathrm{LR}}$) and output images ($\boldsymbol{X}^{\mathrm{HR}}$) are largely similar, it is more meaningful to learn their residue (or difference) where similarities are removed. It is obvious that detailed imagery information in the form of residual images is easier for CNNs to learn than direct LR-HR CNN models [23, 24].

We utilise three typical stacked convolutional-ReLU layers with zero-padding filters in the super-resolution network. Following [23], the empirical basic setting of the

layers is $f_1 = 9$, $n_1 = 64$, $f_2 = 5$, $n_2 = 32$, $f_3 = 5$ and $n_3 = 3$, which are also illustrated in the Fig. 3.2, where f_m and n_m denote the size and number of the filters of the m^{th} layer, respectively.

This model conceptually can be considered to implement four functional procedures. For simplcity, each operation is viewed as a convolotional layer followed by a Rectified Linear Unit (ReLU, *max(0, x)*) [68] layer which is a non-linear activation response function.

1. *LR Feature Extraction:* the first nonlinearly convolutonal layer $srconv_1$ applies 64 filters with size of $9 \times 9 \times 3$ on the input interpolated LR image $\boldsymbol{X}^{\mathrm{LR}}$, which aims at extracting patches from the input image $\boldsymbol{X}^{\mathrm{LR}}$ and representing the patches in the form of 64 feature maps. These feature maps can be viewed as the representations of the residual image in the LR space. This operation can be expressed as two steps:

$$F_1(\boldsymbol{X}^{LR}) = W_1 * \boldsymbol{X}^{LR} + B_1, \tag{3.3}$$

where $*$ denotes the convolution operation, W_1 (with dimension of $9 \times 9 \times 3 \times 64$) and B_1 (with dimension of 1×64) denote the weights and the biases of the filters of the first convolutional layer.

$$F_1(\boldsymbol{X}^{LR}) = max(0, F_1(\boldsymbol{X}^{LR}), \tag{3.4}$$

Equation 3.4 refers to applying the non-linear ReLU operation on the filter responses of the first layer.

2. *LR-HR Feature Mapping:* the second layer $srconv_2$ nonlinearly maps the residual representations from LR space to HR space by using 32 convolutional filters with dimension of $5 \times 5 \times 64$. This layer can possibly be implemented by using more convolutional non-linear layers for obtaining a better performance. As the same as the above,

$$F_2(\boldsymbol{X}^{LR}) = W_2 * F_1(\boldsymbol{X}^{LR}) + B_2, \tag{3.5}$$

where W_2 (with dimension of $5 \times 5 \times 64 \times 32$) and B_2 (with dimension of 1×32) denote the weights and the biases of the filters of the second convolutional layer.

$$F_2(\boldsymbol{X}^{LR}) = max(0, F_2(\boldsymbol{X}^{LR}), \tag{3.6}$$

applies the non-linear ReLU operation on the filter response of the second layer.

3. *Residual Reconstruction:* The third layer $srconv_3$ operates as a reconstructing process, which utilises 3 filters with size of $5 \times 5 \times 32$ to linearly construct the corresponding residual image in the HR space based on the representations obtained from the second layer. The corresponding function is:

$$\boldsymbol{X}^{Res} = W_3 * F_2(\boldsymbol{X}^{LR}) + B_3, \tag{3.7}$$

where W_3 (with dimension of $5 \times 5 \times 32 \times 3$) and B_3 (with dimension of 1×3) denote the weights and the biases of the filters of the last convolutional layer. The reconstructed residual image $\boldsymbol{X}^{\mathrm{Res}}$ visually shows that the missing parts mainly focuses on the high-frequent details, like edges shown in Fig. 3.2.

4. *Skip-Connection:* This step can also be viewed as a special convolutional layer with 3 filters whose values are all ones with size of $1 \times 1 \times 3$. Thus, it works just as a conveyor to transmit the shared data (*i.e.* $\boldsymbol{X}^{\mathrm{LR}}$) in both LR and HR image to mitigate the effect of heavy computation. The whole SR network finally sums the residual image $\boldsymbol{X}^{\mathrm{Res}}$ (learned from the input image $\boldsymbol{X}^{\mathrm{LR}}$) with the input image $\boldsymbol{X}^{\mathrm{LR}}$ to obtain the corresponding super-resolved image $\boldsymbol{X}^{\mathrm{SR}}$:

$$\boldsymbol{X}^{SR} = \boldsymbol{X}^{Res} + \boldsymbol{X}^{LR} \tag{3.8}$$

These operations are to be implemented by a fully convolutional neural network in an end-to-end manner. All the weights (*i.e.* W_1, W_2 and W_3) and biases (*i.e.* B_1, B_2 and B_3) of the convolutional filters are initialised using Guassian function and are to be optimised by training the network on massive image data. Note that for the fully convolutional SR network, it does not require any specific dimensions for the input image, thus this SR network can be flexibly applied on images with any resolutions.

3.2 Image Classification Network

In this part, we describe the image classification CNN which is to classify the object presented in the image and assign one label to it. A number of CNN frameworks [57, 81, 39, 85] have been proposed for image classification, and in this work we consider three popular convolutional neural networks AlexNet [57], VGG-Net [81] and GoogLeNet [85]. All of them typically consist of a number of Convolutional-ReLU-Pool stacks followed by several fully-connected layers.

We first discuss the typical image classification with deep convolutional neural network. Given a set of N training images and corresponding class labels $\{\mathbf{X}_i, y_i\}, i =$

$1, 2, \cdots, N$, the goal of a conventional CNN model is to learn a mapping function $y = f(X)$. The typical cross-entropy loss $L_{\text{ce}}(\cdot)$ on softmax classifer is adopted to measure the performance between class estimates $\hat{y} = f(X)$ and ground truth class labels y :

$$L_{\text{ce}}(\hat{y}, y) = -\sum_{j=1}^{K} y_j \log(\hat{y}_j), \tag{3.9}$$

where j refers to the index of element in vectors, and K denotes the output dimension of softmax layer (the number of classes). The softmax layer applies softmax function (3.10) on final outputs of the network to calculate the categorical distribution:

$$\hat{y}_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}, \text{for} \quad j = 1, ..., K \tag{3.10}$$

where $\boldsymbol{z} = z_1,\ z_2,\ ...,\ z_K$ denote the outputs of the last fully-connected layer of the network and $\hat{y}_j$ denotes the probability distribution of the j^{th} output over all K outcomes. In this sense, classification CNN solves the following minimisation problem with gradient descent back propagation:

$$\min \quad \sum_{i=1}^{N} L_{\text{ce}}(f(X_i), y_i). \tag{3.11}$$

where y_i is the ground truth of input X_i. When the network is well-trained after training phrase, the estimated class label for a given image is determined to be the most possible label over K class labels based on the probability distribution:

$$\hat{y} = arg\, max(\hat{y}_j), \text{for} \quad j = 1, ..., K \tag{3.12}$$

3.2.1 AlexNet

AlexNet was proposed in [57] and is the baseline deep convolutional neural network for large-scale image classification with ImageNet dataset [21]. It consists of 5 convolutional-ReLU layers (*conv1-relu1*, *conv2-relu2*, *conv3-relu3*, *conv4-relu4* and *conv5-relu5*), 3 max-pooling layers (*pool1*, *pool2*, and *pool3*), 2 normalisation layers (*norm1* and *norm2*), 2 dropout layers (*drop6 and drop7*), 3 fully-connected-ReLU layers (*fc6-relu6*, *fc7-relu7* and *fc8*) and a softmax layer (*prob*). For simplicity, we just visualise eight learnable layers (*i.e.* convolutional layers and fully-connected layers), as shown in Figure 3.3.

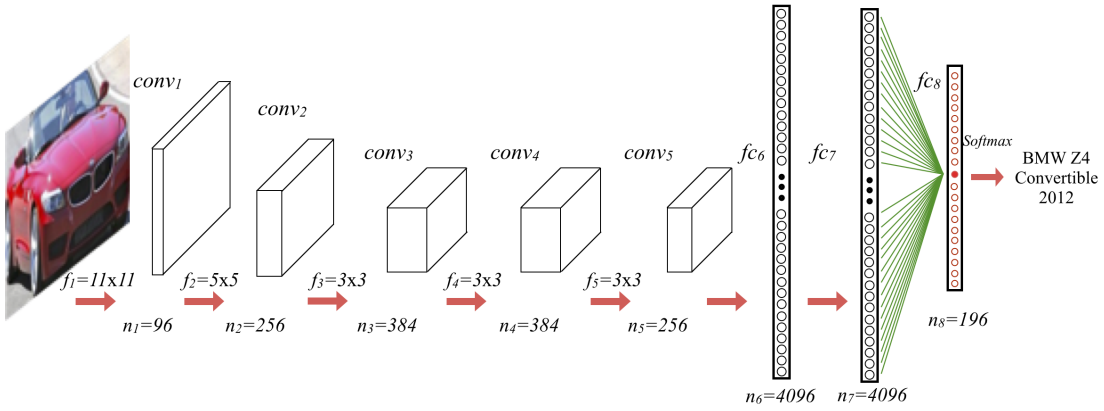


Figure 3.3 *The visual structure of AlexNet classification convolutional neural network. Note that we only visualise the convolutional layers and fully-connection layers.*

The first convolutional layer contains 96 filters (with dimension $11 \times 11 \times 3$) with a stride of 4 pixels (note that stride is the distance of every movement of the filter), which is followed by a local response normalisation (LRN) layer and a max-pooling layer. The outputs of pooling layer are to be filtered by the second convolutional layer with 256 filters with the size $5 \times 5 \times 96$. Next, the third and fourth layers both have 384 filters (with size $3 \times 3 \times 256$ and $3 \times 3 \times 192$, respectively), and 256 filters with size $3 \times 3 \times 192$ for the last convolutional layer, after which another LRN layer and max-pooling layer are applied again before the fully-connected layers. The first two fully-connected layers have 4096 neurons each, but for the last fully-connected layer, the amount of neurons is to be the total number of labels of datasets(*e.g.* 196 for Stanford Cars dataset [56] and 200 for Caltech-UCSD Birds-200-2011 dataset [93]). Finally, the output of last fully-connected layer is to be fed into a softmax layer which generates a normalised probability distribution for each label over all class labels.

3.2.2 VGGNet

VGGNet [81] is made deeper (from 8 layers of AlexNet [57] to 16-19 layers) and more advanced over AlexNet by using very small (3×3) convolution filters to investigate the effect of convolutional network depth. In our work, we choose the VGGNet-16 with 16 layers for our experiments (denoted as VGGNet in the rest of the thesis). It comprises 13 convolutional-ReLU layers with the same receptive filtering size 3×3 and 3 fully-connected-ReLU layers, which simply can be grouped into 6 small blocks (*i.e.* 5 convolutional blocks and 1 fully-connected block). See Figure 3.4 for better

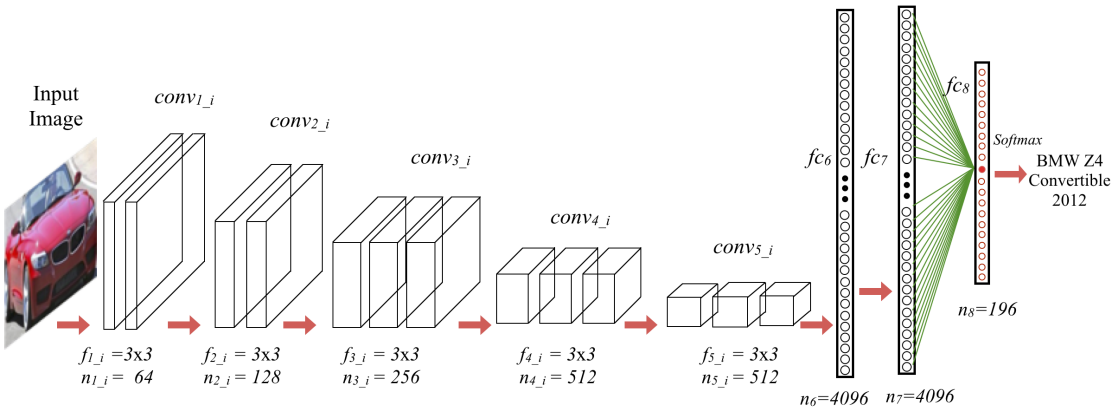


Figure 3.4 *The structure of VGGNet-16 classification convolutional neural network. For the sake of simplicity, only the convolutional layers and fully-connection layers are illustrated in the figure.*

understanding.

The layers in each convolutional block have the same number of filters which produce the same size of feature maps and each convolutional block is followed by a max-pooling layer to reduce the dimension of feature maps. The first block contains two convolutional layers with 64 filter each and the second one also have two layers but with 128 filters for each. There are 256 convolutional filters for each layer in the third block while the layers in the fourth block have 512 filter similar to the fifth block. The last block contains three fully-connected layers followed by a softmax layer, the same as in AlexNet [57], 4096 neurons for the first two layers and 196 or 200 neurons for the last fully-connected layer in our experiments.

3.2.3 GoogLeNet

GoogLeNet [85] was proposed for ImageNet Large-Scale Visual Recognition Competition 2014 (ILSVRC14) and it secured the first place in both classification and detection tasks. GoogLeNet comprises 22 parametrical layers but has much less number of parameters than AlexNet [57] and VGGNet [81] owing to the smaller amount of weights of the fully-connected layer. GoogLeNet generally generates three outputs at various depths for each input, but for simplicity, only the last output (*i.e.* the deepest output) is considered in our experiments. These parametrical layers can be grouped into three parts, as depicted in Figure 3.5, namely, convolutional layers, inception modules and fully-connected layer, among which the inception module

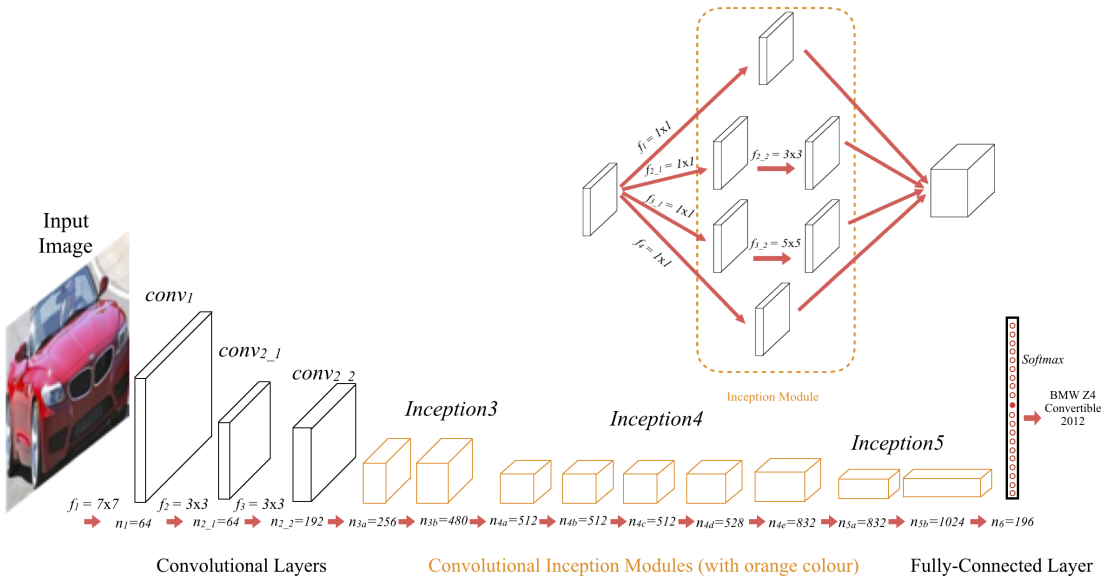


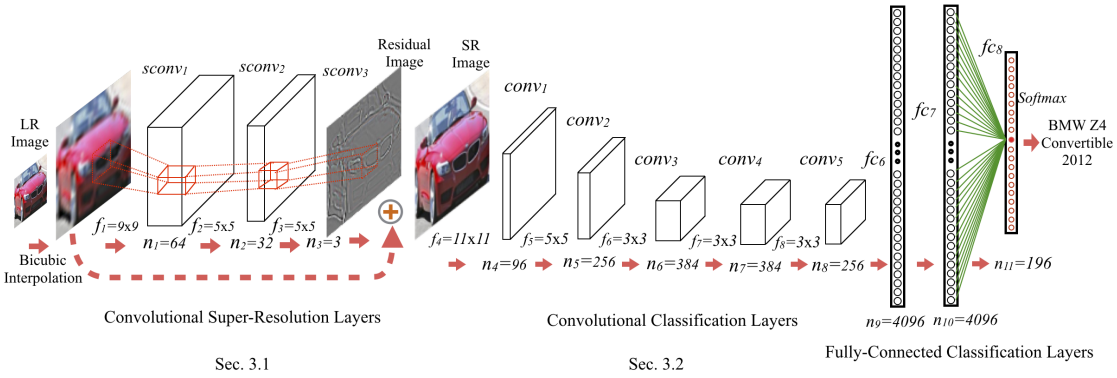
Figure 3.5 *The structure of GoogLeNet convolutional neural network. Note that the orange blocks represent the distinct inception modules which are assembled from six convolutional layers and one max-pooling layer, and only the convolutional layers and fully-connection layer are illustrated.*

is the main hallmark of GoogLeNet as well as responsible for the state-of-the-art performance.

To be specific, the first convolutional layer extracts 64 feature maps with size 114×114 from the input image ($227 \times 227 \times 3$) by operating 64 filters with a large receptive field (*i.e.* 7×7) like in AlexNet [57]. The following two convolutional layers operate more filters (receptive field 3×3) thus more feature maps (192) are obtained with size 57×57 and are fed into the following inception modules. Nine inception modules are stacked on top of each other and all of them share the similar architecture which consists of four convolutional layers with convolution size 1×1 for dimension reduction, two convolutional layers (with convolution sizes 3×3 and 5×5) for feature extraction and one max-pooling layer with size 3×3. These inception modules can be divided into three groups and each group shares the same height and width dimensions regarding the feature maps. Concretely, the first group has two inception modules which generate $n_{3a} = 256$ and $n_{3b} = 480$ feature maps with size 28×28, next, there are five inception modules in the second group and the number of feature maps (with size 14×14) increases from 512 ($n_{4a} - n_{4c}$) to 528 (n_{4d}) and then to 832 (n_{4e}) in order. Two more inception modules are to produce more feature maps (1024) with even smaller size 7×7. Different from AlexNet [57] and VGGNet [81], GoogLeNet employs only one fully-connected layer at the end, which dramatically

mitigates the suffering of training tens of millions of parameters.

3.3 Resolution-Aware Classification Neural Network



Resolution-Aware Classification Neural Network

Figure 3.6 *Pipeline of the proposed Resolution-Aware Classification Neural Network (RACNN) for fine-grained classification with low-resolution images. Convolutional classification layers from AlexNet are adopted for illustrative purpose, which can be readily replaced by those from other CNNs such as VGGnet or GoogLeNet.*

The proposed architecture literally consists of two sub-nets: super-resolution CNN and classification CNN. We combine the super-resolution CNN with the classification CNN together to form conceptually the super-resolution and classification layers in our network which is called Resolution-Aware Classification Neural Network (RACNN). The key difference between the proposed RACNN and conventional classification CNN lies in the introduction of the fully convolutional super-resolution layers, as depicted in Figure 3.6.

Intuitively, RACNN takes an interpolated low-resolution image as input and super-resolves the image via the convolutional super-resolution layers and then feeds it into the convolutional classification layers and fully-connected layers and finally outputs one estimated label for it. Literally, the super-resolution layers are responsible for extracting discriminative fine details which are helpful for accurately recognising the object in the LR image.

Table 3.1 *The configuration of $RACNN_{AlexNet}$ architecture. Note that each convolutional layer is followed by a non-linear ReLU layer which is omitted in the table and the output size of fc_8 layer depends on the the amount of classes of dataset (*e.g. *196 for Stanford Cars).*

name of layer	type of layer	size of output	filter size/ number	stride/ padding	number of parameters
$srconv_1$	convolution	$227 \times 227 \times 64$	9×9 / 64	1 / 4	15,616
$srconv_2$	convolution	$227 \times 227 \times 32$	5×5 / 32	1 / 2	51,232
$srconv_3$	convolution	$227 \times 227 \times 3$	5×5 /3	1 / 2	2,403
$conv_1$	convolution	$55 \times 55 \times 96$	11×11 / 96	4 / 0	34,944
$norm_1$	LRN	$55 \times 55 \times 96$	-	-	-
$pool_1$	max-pooling	$27 \times 27 \times 96$	3×3 / -	3 / 0	-
$conv_2$	convolution	$27 \times 27 \times 256$	5×5 / 256	1 / 2	614,656
$norm_2$	LRN	$27 \times 27 \times 256$	-	-	-
$pool_2$	max-pooling	$13 \times 13 \times 256$	3×3 / -	3 / 0	-
$conv_3$	convolution	$13 \times 13 \times 384$	3×3 / 384	1 / 1	885,120
$conv_4$	convolution	$13 \times 13 \times 384$	3×3 / 384	1 / 1	1327,488
$conv_5$	convolution	$13 \times 13 \times 256$	3×3 / 256	1 / 1	884,992
$pool_3$	max-pooling	$6 \times 6 \times 256$	3×3 / -	3 / 0	-
fc_6	linear	4096	- / 4096	-	37,748,737
$drop_6$	dropout(0.5)	4096	-	-	-
fc_7	linear	4096	- / 4096	-	16,777,217
$drop_7$	dropout(0.5)	4096	-	-	-
fc_8	linear	196	- / 196	-	802,817
output	softmax	196	-	-	-

In our experiments, we adopt AlexNet [57], VGGNet [81] and GoogLeNet [85] as the classification layers of RACNN which are named as $\text{RACNN}_{\text{AlexNet}}$, $\text{RACNN}_{\text{VGGNet}}$ and $\text{RACNN}_{\text{GoogLeNet}}$, respectively. Compared with $\text{RACNN}_{\text{AlexNet}}$, $\text{RACNN}_{\text{VGGNet}}$ has more than double amount of parameters (*i.e.* 135 million vs. 59 million) due to the deeper depth, on the contrary, $\text{RACNN}_{\text{GoogLeNet}}$ has deeper structure than both of them but with much less parameters (*i.e.* 6 million) thanks to the district architecture of inception module. The configuration details of $\text{RACNN}_{\text{AlexNet}}$, $\text{RACNN}_{\text{VGGNet}}$ and $\text{RACNN}_{\text{GoogLeNet}}$ can be found in Table 3.1, Table 3.2 and Table 3.3. Note that, the classification layers can be replaced with any other classification networks.

Table 3.2 *The configuration of $RACNN_{VGGNet}$ architecture.*

name of layer	type of layer	size of output	filter size/ number	stride/ padding	number of parameters
$srconv_1$	convolution	$224 \times 224 \times 64$	9×9 / 64	1 / 4	15,616
$srconv_2$	convolution	$224 \times 224 \times 32$	5×5 / 32	1 / 2	51,232
$srconv_3$	convolution	$224 \times 224 \times 3$	5×5 /3	1 / 2	2,403
$conv_{1\_1}$	convolution	$224 \times 224 \times 64$	3×3 / 64	1 / 1	1,792
$conv_{1\_2}$	convolution	$224 \times 224 \times 64$	3×3 / 64	1 / 1	36,928
$pool_1$	max-pooling	$112 \times 112 \times 64$	2×2 / -	2 / 0	-
$conv_{2\_1}$	convolution	$112 \times 112 \times 128$	3×3 / 128	1 / 1	73,856
$conv_{2\_2}$	convolution	$112 \times 112 \times 128$	3×3 / 128	1 / 1	147,584
$pool_2$	max-pooling	$56 \times 56 \times 128$	2×2 / -	2 / 0	-
$conv_{3\_1}$	convolution	$56 \times 56 \times 256$	3×3 / 256	1 / 1	295,168
$conv_{3\_2}$	convolution	$56 \times 56 \times 256$	3×3 / 256	1 / 1	590,080
$conv_{3\_3}$	convolution	$56 \times 56 \times 256$	3×3 / 256	1 / 1	590,080
$pool_3$	max-pooling	$28 \times 28 \times 256$	2×2 / -	2 / 0	-
$conv_{4\_1}$	convolution	$28 \times 28 \times 512$	3×3 / 512	1 / 1	1,180,160
$conv_{4\_2}$	convolution	$28 \times 28 \times 512$	3×3 / 512	1 / 1	2,359,808
$conv_{4\_3}$	convolution	$28 \times 28 \times 512$	3×3 / 512	1 / 1	2,359,808
$pool_4$	max-pooling	$14 \times 14 \times 512$	2×2 / -	2 / 0	-
$conv_{5\_1}$	convolution	$14 \times 14 \times 512$	3×3 / 512	1 / 1	2,359,808
$conv_{5\_2}$	convolution	$14 \times 14 \times 512$	3×3 / 512	1 / 1	2,359,808
$conv_{5\_3}$	convolution	$14 \times 14 \times 512$	3×3 / 512	1 / 1	2,359,808
$pool_5$	max-pooling	$7 \times 7 \times 512$	2×2 / -	2 / 0	-
fc_6	linear	4096	- / 4096	-	102,760,449
$drop_6$	dropout(0.5)	4096	-	-	-
fc_7	linear	4096	- / 4096	-	16,777,217
$drop_7$	dropout(0.5)	4096	-	-	-
fc_8	linear	196	- / 196	-	802,817
output	softmax	196	-	-	-

3.4 Network Setting and Training

In this section, we present the training strategies adopted in the experiments. As mentioned, the crucial difference between RACNN and conventional CNN is the super-resolution part, thus the training procedure mainly comprises two stages, *i.e.*

Table 3.3 *The configuration of $RACNN_{GoogLeNet}$ architecture.*

name of layer	type of layer	size of output	filter size/ number	stride/ padding	number of parameters
$srconv1$	convolution	$227 \times 227 \times 64$	9×9 / 64	1 / 4	15,616
$srconv2$	convolution	$227 \times 227 \times 32$	5×5 / 32	1 / 2	51,232
$srconv3$	convolution	$227 \times 227 \times 3$	5×5 /3	1 / 2	2,403
$conv1/_{7\times7\_s2}$	convolution	$114 \times 114 \times 64$	7×7 / 64	2 / 3	9,472
$pool1/_{3\times3\_s2}$	max-pooling	$57 \times 57 \times 64$	3×3 / -	2 / 0	-
$pool1/norm_1$	LRN	$57 \times 57 \times 64$	- / -	-	-
$conv2/_{3\times3\_reduce}$	convolution	$57 \times 57 \times 64$	1×1 / 64	1 / 0	4,160
$conv2/_{3\times3}$	convolution	$57 \times 57 \times 192$	3×3 / 192	1 / 1	110,784
$conv1/norm2$	LRN	$57 \times 57 \times 192$	- / -	-	-
$pool2/_{3\times3\_s2}$	max-pooling	$28 \times 28 \times 192$	3×3 / -	2 / 0	-
$inception\_3a$	inception	$28 \times 28 \times 256$	- / 256	-	163,696
$inception\_3b$	inception	$28 \times 28 \times 480$	- / 480	-	388,736
$pool3/_{3\times3\_s2}$	max-pooling	$14 \times 14 \times 480$	3×3 / -	2 / 0	-
$inception\_4a$	inception	$14 \times 14 \times 512$	- / 512	-	376,176
$inception\_4b$	inception	$14 \times 14 \times 512$	- / 512	-	449,160
$inception\_4c$	inception	$14 \times 14 \times 512$	- / 512	-	510,104
$inception\_4d$	inception	$14 \times 14 \times 528$	- / 528	-	605,376
$inception\_4e$	inception	$14 \times 14 \times 832$	- / 832	-	868,352
$pool4/_{3\times3\_s2}$	max-pooling	$7 \times 7 \times 832$	3×3 / -	2 / 0	-
$inception\_5a$	inception	$7 \times 7 \times 832$	- / 832	-	1,043,456
$inception\_5b$	inception	$7 \times 7 \times 1024$	- / 1024	-	1,444,080
$pool5/_{7\times7\_s1}$	ave-pooling	$1 \times 1 \times 1024$	7×7 / -	1 / 0	-
$pool5/_{drop_7\times7\_s1}$	dropout(0.4)	1024	-	-	-
fc_6	linear	196	- / 196	-	200,705
$output$	softmax	196	-	-	-

training for super-resolution and training for classification.

3.4.1 Training for Image Super-Resolution

The introduced part of RACNN is to super-resolve the input LR images for the sake of better classification. We first train the three convolutional SR layers by enforcing the minimal of the mean square loss in Equation 3.2 on ILSVRC 2015 ImageNet

object detection dataset [77] which consists of 11,142 high-resolution images. For the goal of direct utilisation of output of convolutional SR layers (*i.e. SR Image* in Figure 3.6), we train SR layers in RGB color space with all the channels, instead of only on luminance channel Y in YCbCr color space [49, 50]. Specifically, we generate LR images from HR images (*e.g.* 227×227 pixels) via firstly down-sampling HR images to LR 50×50 pixels and then up-scaling to the original image size by bicubic interpolation [47]. We then sample image patches using sliding window and thus obtain sufficient pairs of LR and HR image patches. Following [23], the super-resolution layers are firstly initialised with Guassian function and then trained with image patches by setting learning rates being 1 and weight decays being 0.1 for the first two SR layers (*srconv1* and *srconv2*) and both learning rate and weight decay being 0.1 for the third SR layer (*srconv3*).

3.4.2 Training for Fine-Grained Classification

In this stage, the RACNN is trained in an end-to-end learning manner. Evidently, the proposed RACNN is deeper than its corresponding classification CNN (*e.g.* $\text{RACNN}_{\text{AlexNet}}$ vs. AlexNet) in light of the convolutional SR layers, which can store more knowledge due to more network parameters. For fair comparison, the same SR layers are prefixed to the classification CNN and initialised with Gaussian function without pre-training. We employ the corresponding conventional classification network (*e.g.* AlexNet) as the baseline with respect to the proposed fine-grained image classification network (*e.g.* $\text{RACNN}_{\text{AlexNet}}$). Note that p-RACNN and g-RACNN separately denote the networks whose SR layers are initialised with pre-trained weights (see Section 3.4.1) and Guassian weights in the experiments.

Similar to [75], we utilise the weights pre-trained on ImageNet [77] to initialise the classification layers of network, which transfers all classification layers except the last fully-connected layer. The dimension of the final fully-connected layer is replaced with the size of object classes in the dataset (*i.e.* 196 for Stanford Cars Dataset, 200 for Caltech-UCSD Birds-200-2011 Dataset and 102 for Oxford 102 Category Flowers Dataset) and the weights are initialised with a Gaussian distribution. The whole network is fine-tuned with a small learning rate. Concretely, given the pre-trained weights of convolutional SR layers, p-RACNN is end-to-end trained by minimising the cross entropy loss function (see Equation 3.11) for classification with learning rates 0.1 and weight decays 0 for all layers except the last fully-connected layer with both learning rate and weight decay 1. Likewise, the baseline network and the comparative counterpart g-RACNN are also trained in the same manner as p-RACNN, except that the learning rates and weight decays of SR layers in g-RACNN are set

to 1 and 0.1 for the first two layers ($srconv_1$ and $srconv_2$) and both learning rate and weight decay are 0.1 for the third layer ($srconv_3$) just as in the training for image super-resolution.

4. EXPERIMENT AND EVALUATION

4.1 Datasets and Settings

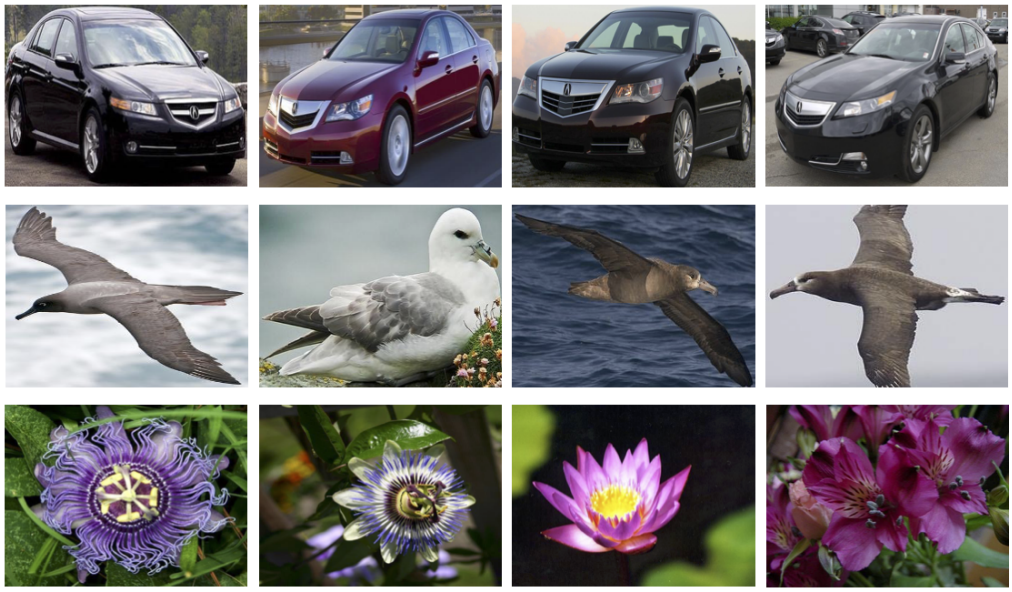


Figure 4.1 *Samples from Stanford Cars (the top row), Caltech-UCSD Birds 200-2011 (the middle row) and Oxford 102 Category Flowers (the bottom row).*

There are many datasets released for this fine-grained visual classification task, such as Stanford Dogs [48], Oxford-IIIT Pet [73], FGVC-Aircraft [66], *etc.* In this work, we evaluate our RACNN on three commonly used fine-grained classification datasets: the Stanford Cars dataset [56], the Caltech-UCSD Birds-200-2011 dataset [93] and the Oxford 102 Category Flowers dataset [69]. Selected samples from these datasets are shown in Figure 4.1.

Stanford Cars Dataset [56] was released by Krause *et al.* for fine-grained classification and contains 16,185 images from 196 classes of cars and each class is typically at the level of Brand, Model, Year. By following the standard evaluation protocol [56], we split the data into 8,144 images for training and 8,041 for testing.

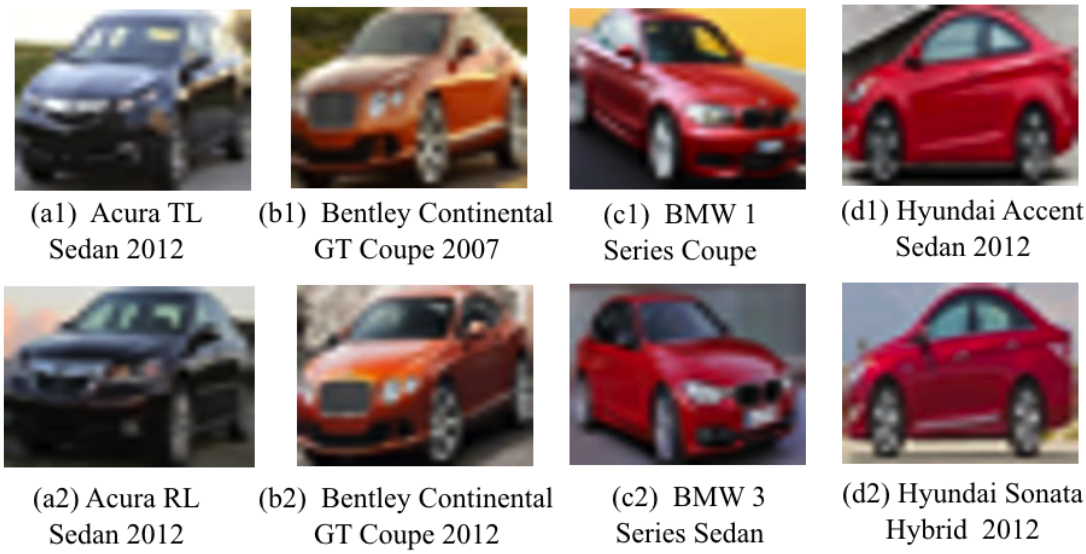


Figure 4.2 *Samples of interpolated low-resolution* (50×50) *images after removing background from the Stanford Cars.*

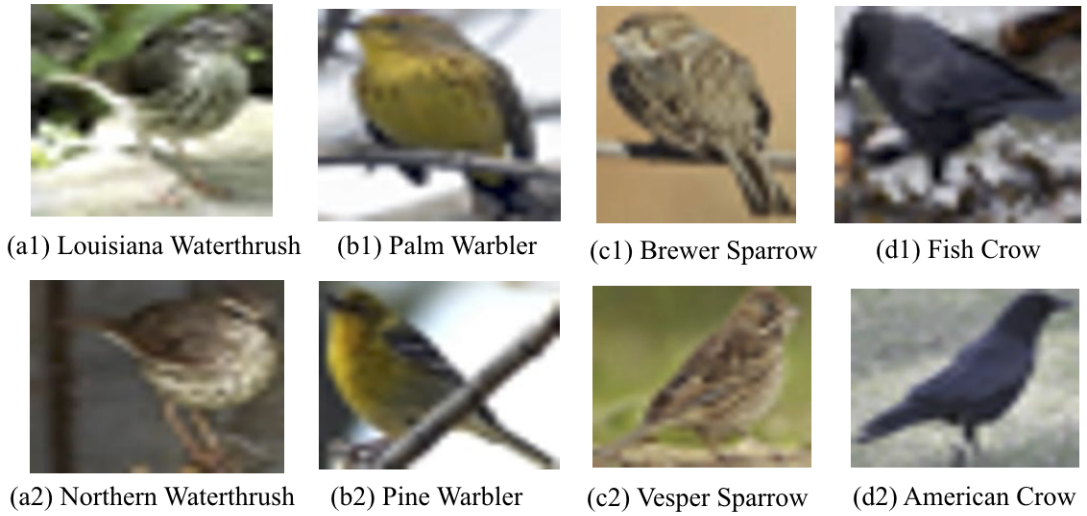


Figure 4.3 *Samples of interpolated low-resolution* (50×50) *images after removing background from the Caltech-UCSD Birds 200-2011.*

Caltech-UCSD Birds-200-2011 Dataset [93] is another challenging fine-grained image dataset aimed at subordinate category classification by providing a comprehensive set of benchmarks and annotation types for the domain of birds. The dataset contains 11,788 images of 200 bird species, among which there are 5,994 images for training and 5,794 for testing [93].

Oxford 102 Category Flowers Dataset [69] totally consists of 8,189 images of flowers which commonly appear in the United Kingdom. These images belong to 102 subcategories and each subcategory contains between 40 and 258 images. In the standard evaluation protocol [69], the whole dataset is divided into 1,020 images for training, 1,020 for validation and 6,149 for testing, note that the training and validation data are merged together to train the networks in this project.

Most of images from these datasets are object-centered and require to be first cropped with provided bounding boxes (if available) to remove the background. Furthermore, we artficially generate the LR images for low-resolution fine-grained classification in the scope of this work. Specifically, the cropped images are down-sampled to small LR images (*e.g.* with the size of 50×50 pixels) and then up-scaled to a large unified size (*i.e.* 227×227 pixels) by bicubic interpolation [47] in order to fit the conventional classification CNNs (*e.g.* AlexNet and GoogLeNet), which follows the similiar settings in [75]. The generated LR images samples from the benchmarks are illustrated in Figure 4.2 and Figure 4.3, which precisely verify our motivation to mitigate the suffering from low visual discrimination due to low-resolution. The proposed RACNN is implemented on Caffe [46]. We adopt *the average per-class accuracy* [62, 75] for these three datasets (the higher value denotes the better performance).

4.2 Comparative Evaluation

In Figure 4.4, we first compare our results with AlexNet [57] and Staged-Training AlexNet [75] for fine-grained classification in low-resolution (50×50) images. It is evident that our $\text{RACNN}_{\text{AlexNet}}$ consistently achieves the best performance on both benchmarks. Precisely, AlexNet achieves 50.4% and 51.3% accuracy (collected from [75]) for the Stanford Cars and Caltech-UCSD Birds datasets, respectively. Knowledge transfer between varying resolution images (*i.e.* Staged-Training AlexNet [75]) can improve classification accuracy, that is 59.5% for the Stanford Cars and 55.3% for the Caltech-UCSD Birds. However, the staged-training AlexNet [75] relies on the strong assumption, *i.e.* high-resolution images available for training, which limits to its usage to other tasks when only low-resolution images are available. Note that our method is more genetic and transforms knowledge of super-resolution

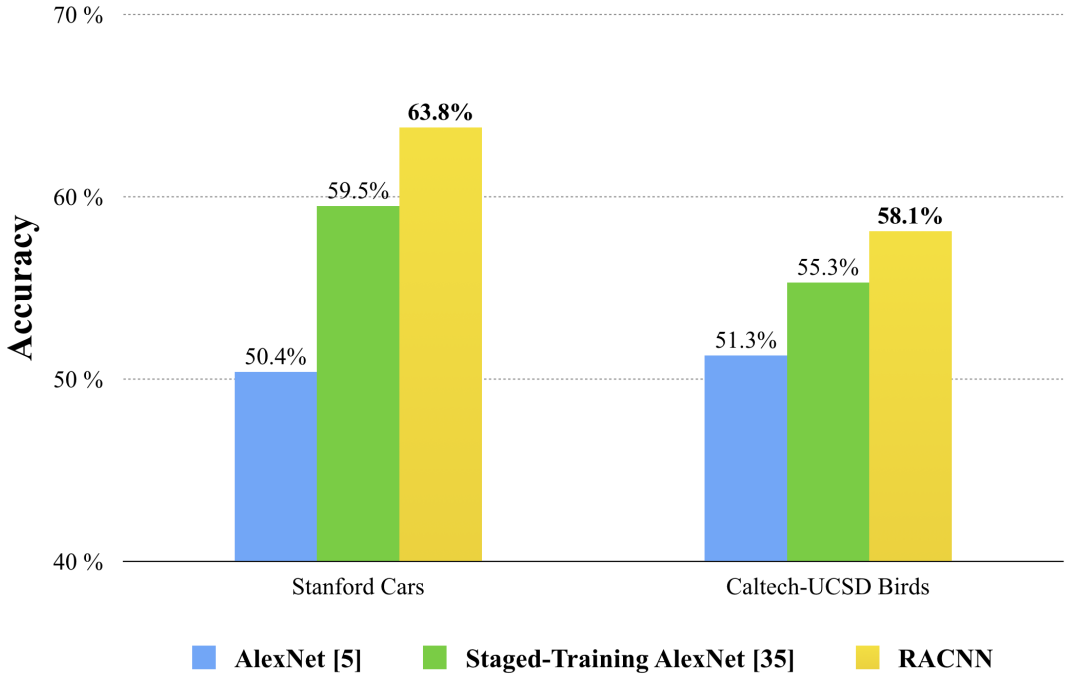


Figure 4.4 *Comparative evaluation on the state-of-the-art methods [[57, 75]] and our RACNN on Cars and Birds Datasets (average per-class accuracies).*

across datasets, which indicates that our method can be readily applied to other low-resolution image classification tasks. The proposed $\text{RACNN}_{\text{AlexNet}}$ significantly beats its direct competitor AlexNet, *i.e.* 63.8% vs. 50.4% on the Stanford Cars dataset and 58.1% vs. 51.3% on the Caltech-UCSD Birds dataset. With the same settings and training samples, the performance gap can only be explained by the novel network structure of RACNN.

4.3 Evaluation of Super-Resolution Layers

In this experiment, we employ all layers in the AlexNet, VGGNet and GoogLeNet [85] as classification layers in RACNN. Note that, different from the previous experiments, we freeze all classification layers by setting learning rates and weights decays to 0 besides the last fully-connected layers of both baseline CNNs, and our RACNN is then fine-tuned with low-resolution data, Such a setting treats classification layers in RACNN as an identical classifier for evaluating the effect of adding convolutional SR layers. RACNN with initial Gaussian and pre-trained weights are called as g-RACNN and p-RACNN respectively. Comparative results are shown in Table 4.1 and Figure 4.5. Both g-RACNN and p-RACNN consistently outperform the baseline CNNs in all experiments, and we observe similar situation in Section4.2. With the same experimental setting except different initial weights for convolutional

Table 4.1 *Evaluation on effect of convolutional SR layers. We fix all convolutional classification layers and fully-connected layers except the last fully-connected layer. g-RACNN and p-RACNN denote the proposed RACNN from with standard Gaussian and pre-trained weights of convolutional SR layers. Note that best results are shown in* ***bold****.*

Methods	Stanford Cars [56]	Caltch-UCSD Birds [93]	Oxford Flowers [69]
AlexNet [57]	43.75%	44.99%	70.03%
g-RACNN$_{\text{AlexNet}}$	45.77%	47.17%	71.91%
p-RACNN$_{\text{AlexNet}}$	**47.90%**	**51.23%**	**74.24%**
VGGNet [81]	41.49%	43.46%	67.82%
g-RACNN$_{\text{VGGNet}}$	42.86%	44.72%	68.03%
p-RACNN$_{\text{VGGNet}}$	**44.65%**	**49.33%**	**69.17%**
GoogLeNet [85]	46.85%	48.52%	69.28%
g-RACNN$_{\text{GoogLeNet}}$	50.37%	55.16%	69.77%
p-RACNN$_{\text{GoogLeNet}}$	**50.76%**	**57.30%**	**73.51%**

super-resolution layers, the results of g-RACNN and p-RACNN are reported. Test set accuracies in Table 4.1 and Figure 4.5 show that p-RACNN is superior to g-RACNN.

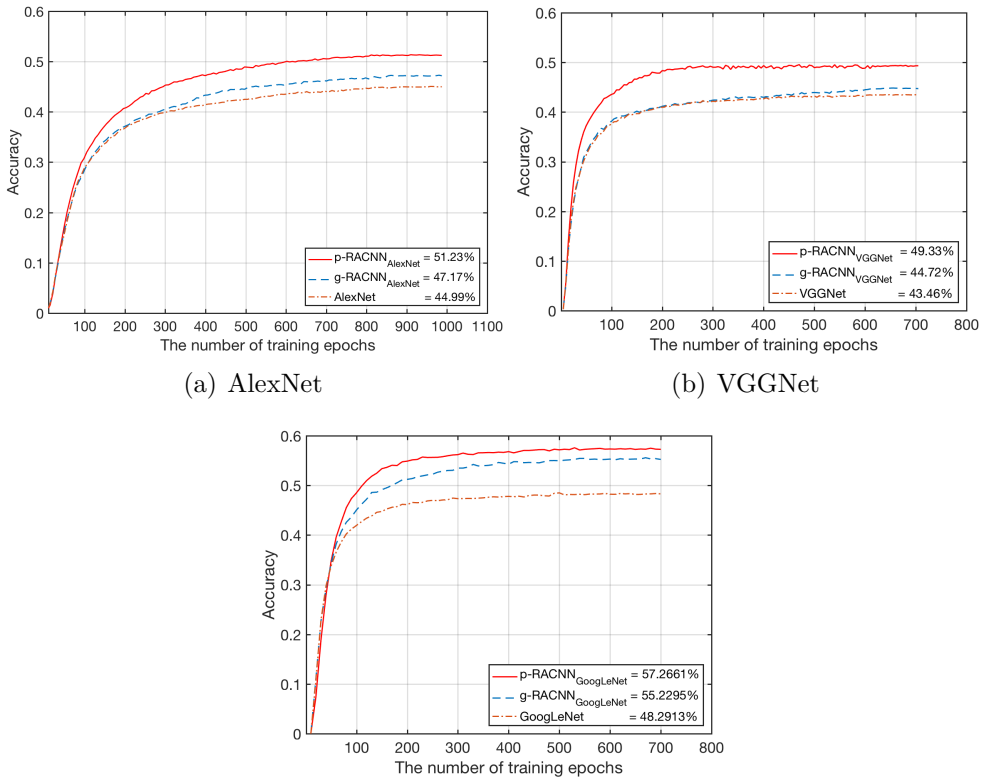
Specifically, p-RACNN$_{\text{AlexNet}}$ achieves 2.1%, 4.1% and 2.3% improvements compared to g-RACNN$_{\text{AlexNet}}$ on Stanford Cars, Caltech-UCSD Birds and Oxford Flowers datasets respectively. Meanwhile, p-RACNN$_{\text{VGGNet}}$ boosts the accuracies from 42.86% to 44.65% on Stanford Cars, from 44.72% to 49.33% on Caltech-UCSD Birds and from 68.03% to 69.17% on Oxford Flowers. Moreover, p-RACNN$_{\text{GoogLeNet}}$ shares the same tendencies as p-RACNN$_{\text{AlexNet}}$ and p-RACNN$_{\text{VGGNet}}$, which separately improves the results to 50.76%, 57.30% and 73.51% from 50.37%, 55.16% and 69.77% on three datasets.

On the one hand, all the RACNN-based approaches achieve better results than their ordinary counterparts (without SR layers), which is caused by the deeper structures owned by RACNN-based methods. On the other hand, p-RACNN and g-RACNN share the same network structure but differ only in network weights initialisation of convolutional SR layers. In this sense, better performance of p-RACNN is credited to the knowledge about refining low-resolution images (*i.e.* pre-trained weights), which verifies our motivation to boost low-resolution image classification via image super-resolution. It is noteworthy that since the feature extraction layers are frozen, the networks are not fine-tuned to low-resolution specific features, but all performance

Table 4.2 *Training time for the proposed RACNN and its competing CNNs (seconds / epoch)*

Methods	Stanford Cars [56]	Caltch-UCSD Birds [93]	Oxford Flowers [69]
AlexNet	11	8	3
$\text{RACNN}_{\text{AlexNet}}$	111	80	25
VGGNet	133	82	34
$\text{RACNN}_{\text{VGGNet}}$	356	215	90
GoogLeNet	20	25	8
$\text{RACNN}_{\text{GoogLeNet}}$	136	120	59

boosts are owing to recovered high-resolution details by the super-resolution layers.



(a) AlexNet (b) VGGNet

(c) GoogLeNet

Figure 4.5 *The accuracy on testing dataset during training process of AlexNet, VGGNet and GoogLeNet on the Caltech-UCSD Birds Dataset.*

In addition, with respect to the details of computation, we use one Lenovo Y900 Desktop running Linux Ubuntu 14.04 operating system with one Intel i7-6700K

CPU and one Nvidia GTX-980 GPU with 4 GB memory in our experiments. The proposed RACNN can consistently achieve superior performance with the price of deeper network structure, which thus causes higher computational complexity than its competitors (*i.e.* AlexNet, VGGNet, GoogLeNet) as shown in Table 4.2.

4.4 Evaluation on Varying Resolution

Table 4.3 *Comparison with varying resolution level (Res. Level) on the Caltech-UCSD Birds 200-2011 Dataset. Note that best results are shown in* ***bold****.*

Resolution Level	AlexNet [57]	g-RACNN$_{\text{AlexNet}}$	p-RACNN$_{\text{AlexNet}}$
25×25	31.58%	43.68%	**45.06%**
50×50	44.99%	47.17%	**51.23%**
100×100	51.01%	51.24%	**52.88%**

We further evaluate our proposed RACNN method with respect to varying resolutions on the Caltech-UCSD Birds 200-2011 Dataset [93]. All low-resolution images are first up-scaled to the same image size, *i.e.* 227×227, before training models. The better performance of RACNN$_{\text{AlexNet}}$ over conventional AlexNet is achieved for cross-resolution fine-grained image classification, which is shown in Table 4.3.

We observe that our method's margin is better for lower resolution images (*e.g.* 25×25) than high resolution images (*e.g.* 100×100). In details, p-RACNN$_{\text{AlexNet}}$ increases the accuracy by above 13% for 25×25 pixel images but less than 2% improvement on 100×100 resolution images. The reason is that the SR layers of RACNN play a significant role in introducing texture details especially when missing more visual cues of object classification in lower quality images, which further demonstrates our observation and motivation.

5. CONCLUSION

In this work, we proposed and verified a simple yet effective resolution-aware classification neural network for fine-grained object classification with low-resolution images. Our proposed framework integrates residual image super-resolution with general classification networks for solving low-resolution fine-grained object classification problem in an end-to-end fashion. This framework has been verified on three popular benchmark datasets and the results of extensive experiments indicate that the introduction of convolutional super-resolution layers to conventional CNNs can indeed recover fine details for low-resolution images to boost the performance of low-resolution fine-grained classification. Moreover, we also conduct experiment on cross-resolution image classification problem and the result supports that our approach still works on varying resolution classification tasks. The concept of this thesis is general and the existing convolutional super-resolution and classification networks can be readily combined to cope with low-resolution as well as cross-resolution image classification. In future work, this concept can be applied on other computer vision tasks, such as human action recognition in low-resolution videos and human face recognition in surveillance records.

BIBLIOGRAPHY

[1] Z. Akata, S. Reed, D. Walter, H. Lee, and B. Schiele, "Evaluation of output embeddings for fine-grained image classification," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[2] A. Angelova and S. Zhu, "Efficient object detection and segmentation for fine-grained recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 811–818.

[3] G. H. Ball and D. J. Hall, "A clustering technique for summarizing multivariate data," *Systems Research and Behavioral Science*, vol. 12, no. 2, pp. 153–155, 1967.

[4] T. Berg and P. Belhumeur, "Poof: Part-based one-vs.-one features for fine-grained categorization, face verification, and attribute estimation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[5] T. Berg and P. N. Belhumeur, "How do you tell a blackbird from a crow?" in *Proceedings of International Conference on Computer Vision*, 2013, pp. 9–16.

[6] T. Berg, J. Liu, S. Woo Lee, M. L. Alexander, D. W. Jacobs, and P. N. Belhumeur, "Birdsnap: Large-scale fine-grained visual categorization of birds," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 2011–2018.

[7] A. Bosch, A. Zisserman, and X. Munoz, "Image classification using random forests and ferns," in *Proceedings of International Conference on Computer Vision*, 2007, pp. 1–8.

[8] S. Branson, G. Van Horn, S. Belongie, and P. Perona, "Bird species categorization using pose normalized deep convolutional nets," in *Proceedings of British Machine Vision Conference*, 2014.

[9] S. Branson, C. Wah, F. Schroff, B. Babenko, P. Welinder, P. Perona, and S. Belongie, "Visual recognition with humans in the loop," in *Proceedings of European Conference on Computer Vision*, 2010, pp. 438–451.

[10] M. Brown, S. R. Gunn, and H. G. Lewis, "Support vector machines for optimal classification and spectral unmixing," *Ecological Modelling*, vol. 120, no. 2, pp. 167–179, 1999.

[11] G. Camps-Valls, T. V. B. Marsheva, and D. Zhou, "Semi-supervised graph-based hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 45, no. 10, pp. 3044–3054, 2007.

[12] Y. Chai, V. Lempitsky, and A. Zisserman, "Symbiotic segmentation and part localization for fine-grained categorization," in *Proceedings of International Conference on Computer Vision*, 2013, pp. 321–328.

[13] H. Chang, D.-Y. Yeung, and Y. Xiong, "Super-resolution through neighbor embedding," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2004.

[14] O. Chapelle, P. Haffner, and V. N. Vapnik, "Support vector machines for histogram-based image classification," *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 1055–1064, 1999.

[15] K. Chen and Z. Zhang, "Learning to classify fine-grained categories with privileged visual-semantic misalignment," *IEEE Transactions on Big Data*, 2016.

[16] M. Chevalier, N. Thome, M. Cord, J. Fournier, G. Henaff, and E. Dusch, "LR-CNN for fine-grained classification with varying resolution," in *IEEE International Conference of Image Processing*, 2015, pp. 3101–3105.

[17] S.-B. Cho, "Neural-network classifiers for recognizing totally unconstrained handwritten numerals," *IEEE Transactions on Neural Networks*, vol. 8, no. 1, pp. 43–53, 1997.

[18] M. J. Collins, C. Dymond, and E. A. Johnson, "Mapping subalpine forest types using networks of nearest neighbour classifiers," *International Journal of Remote Sensing*, vol. 25, no. 9, pp. 1701–1721, 2004.

[19] D. Dai, R. Timofte, and L. Van Gool, "Jointly optimized regressors for image super-resolution," in *Computer Graphics Forum*, vol. 34, 2015, pp. 95–104.

[20] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, 2005, pp. 886–893.

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

[22] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition," in *Proceedings of International Conference on Machine Learning*, 2014, pp. 647–655.

[23] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 2, pp. 295–307, 2016.

[24] C. Dong, C. C. Loy, and X. Tang, "Accelerating the super-resolution convolutional neural network," in *Proceedings of European Conference on Computer Vision*, 2016, pp. 391–407.

[25] R. Farrell, O. Oza, N. Zhang, V. I. Morariu, T. Darrell, and L. S. Davis, "Birdlets: Subordinate categorization using volumetric primitives and pose-normalized appearance," in *Proceedings of International Conference on Computer Vision*, 2011, pp. 161–168.

[26] R. Fattal, "Image upsampling via imposed edge statistics," in *ACM Transactions on Graphics*, vol. 26, no. 3, 2007, p. 95.

[27] G. Freedman and R. Fattal, "Image and video upscaling from local self-examples," *ACM Transactions on Graphics*, vol. 30, no. 2, p. 12, 2011.

[28] J. Fu, H. Zheng, and T. Mei, "Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[29] G. Giacinto and F. Roli, "Design of effective neural network ensembles for image classification purposes," *Image and Vision Computing*, vol. 19, no. 9, pp. 699–707, 2001.

[30] D. Glasner, S. Bagon, and M. Irani, "Super-resolution from a single image," in *Proceedings of International Conference on Computer Vision*, 2009, pp. 349–356.

[31] X. Glorot, A. Bordes, and Y. Bengio, "Domain adaptation for large-scale sentiment classification: A deep learning approach," in *Proceedings of International Conference on Machine Learning*, 2011, pp. 513–520.

[32] C. Goring, E. Rodner, A. Freytag, and J. Denzler, "Nonparametric part transfer for fine-grained recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 2489–2496.

[33] M. Guillaumin, J. Verbeek, and C. Schmid, "Multimodal semi-supervised learning for image classification," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 902–909.

[34] R. Haapanen, A. R. Ek, M. E. Bauer, and A. O. Finley, "Delineation of forest/nonforest land use classes using nearest neighbor methods," *Remote Sensing of Environment*, vol. 89, no. 3, pp. 265–271, 2004.

[35] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, 2006, pp. 1735–1742.

[36] M. Hansen, R. Dubayah, and R. DeFries, "Classification trees: an alternative to traditional land cover classifiers," *International Journal of Remote Sensing*, vol. 17, no. 5, pp. 1075–1081, 1996.

[37] P. J. Hardin, "Parametric and nearest-neighbor methods for hybrid classification: a comparison of pixel assignment accuracy," *Photogrammetric Engineering and Remote Sensing*, vol. 60, no. 12, pp. 1439–1448, 1994.

[38] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of International Conference on Computer Vision*, 2015, pp. 1026–1034.

[39] ——, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[40] A. B. Hillel and D. Weinshall, "Subordinate class recognition using relational object models," in *Advances in Neural Information Processing Systems*, 2007, pp. 73–80.

[41] J.-B. Huang, A. Singh, and N. Ahuja, "Single image super-resolution from transformed self-exemplars," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5197–5206.

[42] J. Huang and D. Mumford, "Statistics of natural images and models," in *IEEE Conference on Computer Vision and Pattern Recognition*, 1999, pp. 541–547.

[43] M. Irani and S. Peleg, "Improving resolution by image registration," *Computer Vision, Graphics, and Image Processing: Graphical Models and Image Processing*, pp. 231–239, 1991.

[44] A. Iscen, G. Tolias, P.-H. Gosselin, and H. Jégou, "A comparison of dense region detectors for image search and fine-grained classification," *IEEE Transactions on Image Processing*, vol. 24, no. 8, pp. 2369–2381, 2015.

[45] M. Jaderberg, K. Simonyan, A. Zisserman *et al.*, "Spatial transformer networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 2017–2025.

[46] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *ACM International Conference on Multimedia*, 2014, pp. 675–678.

[47] R. Keys, "Cubic convolution interpolation for digital image processing," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 29, no. 6, pp. 1153–1160, 1981.

[48] A. Khosla, N. Jayadevaprakash, B. Yao, and F.-F. Li, "Novel dataset for fine-grained image categorization: Stanford dogs," in *IEEE Conference on Computer Vision and Pattern Recognition Workshop on Fine-Grained Visual Categorization*, vol. 2, 2011, p. 1.

[49] J. Kim, J. Kwon Lee, and K. Mu Lee, "Accurate image super-resolution using very deep convolutional networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1646–1654.

[50] ——, "Deeply-recursive convolutional network for image super-resolution," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1637–1645.

[51] K. I. Kim and Y. Kwon, "Single-image super-resolution using sparse regression and natural image prior," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 6, pp. 1127–1133, 2010.

[52] T. Kohonen, "The self-organizing map," *Neurocomputing*, vol. 21, no. 1, pp. 1–6, 1998.

[53] J. Krause, T. Gebru, J. Deng, L.-J. Li, and L. Fei-Fei, "Learning features and parts for fine-grained recognition," in *Proceedings of International Conference on Pattern Recognition*, 2014, pp. 26–33.

[54] J. Krause, H. Jin, J. Yang, and L. Fei-Fei, "Fine-grained recognition without part annotations," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5546–5555.

[55] J. Krause, B. Sapp, A. Howard, H. Zhou, A. Toshev, T. Duerig, J. Philbin, and L. Fei-Fei, "The unreasonable effectiveness of noisy data for fine-grained

recognition," in *Proceedings of European Conference on Computer Vision*. Springer, 2016, pp. 301–320.

[56] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3d object representations for fine-grained categorization," in *International Conference on Computer Vision Workshops*, 2013, pp. 554–561.

[57] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012.

[58] N. Kumar, P. N. Belhumeur, A. Biswas, D. W. Jacobs, W. J. Kress, I. C. Lopez, and J. V. Soares, "Leafsnap: A computer vision system for automatic plant species identification," in *Proceedings of European Conference on Computer Vision*, 2012, pp. 502–516.

[59] W.-S. Lai, J.-B. Huang, N. Ahuja, and M.-H. Yang, "Deep laplacian pyramid networks for fast and accurate super-resolution," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[60] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel, "Handwritten digit recognition with a back-propagation network," in *Advances in Neural Information Processing Systems*, 1990, pp. 396–404.

[61] T.-Y. Lin, A. RoyChowdhury, and S. Maji, "Bilinear cnn models for fine-grained visual recognition," in *Proceedings of International Conference on Computer Vision*, 2015, pp. 1449–1457.

[62] Y. Liu, Y. Qian, K. Chen, J.-K. Kämäräinen, H. Huttunen, L. Fan, and J. Saarinen, "Incremental convolutional neural network training," in *International Conference of Pattern Recognition Workshop on Deep Learning for Pattern Recognition*, 2016.

[63] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[64] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, no. 14, 1967, pp. 281–297.

[65] S. Maji, L. Bourdev, and J. Malik, "Action recognition from a distributed representation of pose and appearance," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 3177–3184.

[66] S. Maji, E. Rahtu, J. Kannala, M. Blaschko, and A. Vedaldi, "Fine-grained visual classification of aircraft," *arXiv preprint arXiv:1306.5151*, 2013.

[67] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[68] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of International Conference on Machine Learning*, 2010, pp. 807–814.

[69] M.-E. Nilsback and A. Zisserman, "Automated flower classification over a large number of classes," in *Indian Conference on Computer Vision, Graphics & Image Processing*, 2008, pp. 722–729.

[70] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 971–987, 2002.

[71] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1717–1724.

[72] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[73] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. V. Jawahar, "Cats and dogs," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[74] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. Jawahar, "Cats and dogs," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3498–3505.

[75] X. Peng, J. Hoffman, X. Y. Stella, and K. Saenko, "Fine-to-coarse knowledge transfer for low-res image classification," in *IEEE International Conference of Image Processing*, 2016, pp. 3683–3687.

[76] S. Reed, Z. Akata, H. Lee, and B. Schiele, "Learning deep representations of fine-grained visual descriptions," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 49–58.

[77] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[78] S. Schulter, C. Leistner, and H. Bischof, "Fast and accurate image upscaling with super-resolution forests," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3791–3799.

[79] S. Shekhar, V. M. Patel, and R. Chellappa, "Synthesis-based robust low resolution face recognition," *arXiv preprint arXiv:1707.02733*, 2017.

[80] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1874–1883.

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

[82] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting." *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[83] J. Sun, Z. Xu, and H.-Y. Shum, "Image super-resolution using gradient profile prior," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.

[84] L. Sun and J. Hays, "Super-resolution from internet-scale scene matching," in *IEEE International Conference on Computational Photography*, 2012, pp. 1–12.

[85] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[86] Y. Tai, J. Yang, and X. Liu, "Image super-resolution via deep recursive residual network," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[87] Y.-W. Tai, S. Liu, M. S. Brown, and S. Lin, "Super resolution using edge prior and single image detail synthesis," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 2400–2407.

[88] M. F. Tappen and C. Liu, "A bayesian approach to alignment-based image hallucination," in *Proceedings of European Conference on Computer Vision*. Springer, 2012, pp. 236–249.

[89] R. Timofte, V. De Smet, and L. Van Gool, "Anchored neighborhood regression for fast example-based super-resolution," in *Proceedings of International Conference on Computer Vision*, 2013, pp. 1920–1927.

[90] ——, "A+: Adjusted anchored neighborhood regression for fast super-resolution," in *Proceedings of Asian Conference on Computer Vision*, vol. 30, no. 13, 2014, pp. 111–126.

[91] A. Torralba, R. Fergus, and W. T. Freeman, "80 million tiny images: A large data set for nonparametric object and scene recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 11, pp. 1958–1970, 2008.

[92] J. Van De Weijer, C. Schmid, J. Verbeek, and D. Larlus, "Learning color names for real-world applications," *IEEE Transactions on Image Processing*, vol. 18, no. 7, pp. 1512–1523, 2009.

[93] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The Caltech-UCSD Birds-200-2011 Dataset," California Institute of Technology, Tech. Rep. CNS-TR-2011-001, 2011.

[94] Q. Wang, X. Tang, and H. Shum, "Patch based blind image super resolution," in *Proceedings of International Conference on Computer Vision*, vol. 1, 2005, pp. 709–716.

[95] Z. Wang, S. Chang, Y. Yang, D. Liu, and T. S. Huang, "Studying very low resolution recognition using deep networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4792–4800.

[96] Z. Wang, D. Liu, J. Yang, W. Han, and T. Huang, "Deep networks for image super-resolution with sparse prior," in *Proceedings of International Conference on Computer Vision*, 2015, pp. 370–378.

[97] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, "Sun database: Large-scale scene recognition from abbey to zoo," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 3485–3492.

[98] Z. Xu, S. Huang, Y. Zhang, and D. Tao, "Augmenting strong supervision using web data for fine-grained categorization," in *Proceedings of International Conference on Computer Vision*, 2015, pp. 2524–2532.

[99] C.-Y. Yang, S. Liu, and M.-H. Yang, "Structured face hallucination," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1099–1106.

[100] C.-Y. Yang, C. Ma, and M.-H. Yang, "Single-image super-resolution: a benchmark," in *Proceedings of European Conference on Computer Vision*, 2014, pp. 372–386.

[101] J. Yang, Z. Lin, and S. Cohen, "Fast image super-resolution based on in-place example regression," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1059–1066.

[102] J. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE Transactions on Image Processing*, vol. 19, no. 11, pp. 2861–2873, 2010.

[103] B. Yao, G. Bradski, and L. Fei-Fei, "A codebook-free and annotation-free approach for fine-grained image categorization," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[104] B. Yao, A. Khosla, and L. Fei-Fei, "Combining randomization and discrimination for fine-grained image categorization," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[105] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proceedings of European Conference on Computer Vision*. Springer, 2014, pp. 818–833.

[106] R. Zeyde, M. Elad, and M. Protter, "On single image scale-up using sparse-representations," in *International Conference on Curves and Surfaces*, 2010, pp. 711–730.

[107] N. Zhang, J. Donahue, R. Girshick, and T. Darrell, "Part-based R-CNNs for fine-grained category detection," in *Proceedings of European Conference on Computer Vision*, 2014.

[108] N. Zhang, R. Farrell, and T. Darrell, "Pose pooling kernels for sub-category recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3665–3672.

[109] N. Zhang, R. Farrell, F. Iandola, and T. Darrell, "Deformable part descriptors for fine-grained recognition and attribute prediction," in *Proceedings of International Conference on Computer Vision*, 2013, pp. 729–736.

[110] W. W. Zou and P. C. Yuen, "Very low resolution face recognition problem," *IEEE Transactions on Image Processing*, vol. 21, no. 1, pp. 327–340, 2012.

APPENDICES