INDIAN STATISTICAL INSTITUTE

DOCTORAL THESIS

On Automatic Identification of Retail Products in Images of Racks in the Supermarkets

Submitted in partial fulfillment of the requirements
for the degree of

Doctor of Philosophy
in
Computer Science

by

Bikash Santra

Under the supervision of

Prof. Dipti Prasad Mukherjee

Electronics and Communication Sciences Unit
Computer and Communication Sciences Division

December 2021

To my beloved parents . . .

Acknowledgement

I would like to express my deep and heartfelt gratitude to my respected guide and mentor, PROF. DIPTI PRASAD MUKHERJEE, for his evergreen blessings, important advice, support, good wishes and constant encouragement all alone the journey of this work. I thank him from bottom of my heart for accepting me as his doctoral student. His criticism to bring out the best in me, has shaped the foundation of research in me. His belief in independent thinking, has always encouraged me to think in my own way. I thank him for always helping me by sharing his knowledge and experience. He is phenomenal in channelizing our half-baked ideas to the full-blown solutions with his excellent supervision and depth understanding of the problems. Without his undisputed support, invaluable insights and advises, this thesis would not have been true in the light of daylight.

I would like to thank all the faculty members of Electronics and Communication Sciences Unit (ECSU) of Indian Statistical Institute (ISI) for offering me their helpful suggestions and various supports. I express my gratitude to them for creating a constructive and healthy research atmosphere in the unit. I thank all the staffs of ECSU for their active support in multiple occasions. I would like to acknowledge TCS Ltd. for providing us the interesting problem, based on which this thesis work is carried out. I thank Avishek Kumar Shaw, Rajashree Ramakrishnan, Shilpa Yadukumar Rao and Pranoy Hari of TCS Ltd. for their support in conducting our joint research work with TCS Ltd.

Now, I would love to thank my fellow researcher friend Dr. Angshuman Paul. He is the person, who has always inspired me, helped me to get rid of multiple issues, motivated me to not to loose patience, shared his insights and thoughts on various problems. I am deeply indebted to Dr. Partha Pratim Mohanta for his continuous support in various forms during my days in ISI. I like to thank the finest researchers Dr. Dipankar Chakrabarti, Dr. Swapna Agarwal, Udita Ghosh and Aditya Panda for working with me. I thank Samriddha Sanyal for sharing his knowledge and thoughts with me, which immensely helped me in many ways. I like to thank the enthusiastic people around me, Suvra Jyoti Choudhury, Suchismita Das, Saikat Sarkar, Suman Ghosh, Sankarsan Seal, Rupam Paul, Nasib Ullah, Dr. Yusuf Akhtar, Rivu Chakraborty, Aratrik Chatterjee, Tamal Batabyal, Archan Ray, Nishant Kumar and other researchers of ECSU, whose gracious presence helped me to always keep motivated. I must thank all my friends for filling my life with pleasure and amusement. I also thank my respected teachers for their encouragement.

Finally, to my immediate/extended family, particularly my parents Susanta Santra and Mita Santra, my wife Antara Sharma, sister Tripti Santra, brother Prakash Santra, grand father Haripada Maity, and uncle Biswarup Maity, thank you for your love, encouragement, patience and unwavering belief in me. Without you, I would not be the person I am today. My father is my true source of inspiration to be a good human being. Above all I would love to thank my wife for her immense care, affection and consistent support, for all the early mornings and late nights, and for keeping me sane over the past few years. But most importantly, thank you for being my friend and biggest critic of my life. I owe you everything.

Bikash Santra
Indian Statistical Institute, Kolkata

Abstract

An image of a rack in a supermarket displays a number of retail products. The identification and localization of these individual products from the images of racks is a challenge for any machine vision system. In this thesis, we aim to address this problem and suggest a set of computer vision based solutions for automatic identification of these retail products. We design a novel classifier that differentiates the similarly looking yet non-identical (fine-grained) products for improving the performance of our machine vision system. The proposed fine-grained classifier simultaneously captures both object-level and part-level (image of an object consists of multiple parts or image patches) cues of the products for accurately distinguishing the fine-grained products. A graph-based non-maximal suppression strategy is proposed that selects a winner region proposal among a group of proposals representing a product. This solves an important bottleneck of conventional greedy non-maximal suppression algorithm for disambiguation of overlapping region proposals generated in an intermediate step of our proposed system. We initiate the solution of the problem of automatic product identification by developing an end-to-end annotation-free machine vision system for recognition and localization of products on the rack. The proposed system introduces a novel exemplar-driven region proposal strategy that overcomes the shortcomings of traditional exemplar-independent region proposal schemes like selective window search. Finally, we find the empty spaces (or gaps between products) in each shelf of any rack by creating a graph of superpixels for the rack. We extract the visual features of superpixels from our graph convolutional and Siamese networks. Subsequently, we send the graph along with the features of superpixels to a structural support vector machine for discovering the empty spaces of the shelves. The efficacy of the proposed approaches are established through various experiments on our In-house dataset and three publicly available benchmark datasets: Grozi-120 [Merler et al., IEEE CVPR 2007, 1-8], Grocery Products [George et al., Springer ECCV 2014, 440-455], and WebMarket [Zhang et al., Springer ACCV 2007, 800-810].

List of Tables

List of Figures

List of Symbols

P	Product image
A	Affine transformation matrix or homography
$\mathtt{n}_f$	Normal to the plane representing the face of the rack
$\mathtt{n}_c$	Normal to the image (camera) plane
π	Plane parallel to the ground plane
m	Number of individual products or product classes
$\mathbb{D}$	Database of product templates
$\mathbb{D}_t$	t^{th} product template
I	Rack image or shelf image
H	Sub-image/patch/region proposal of a rack I
ϖ	Number of matched key-points in $\mathbb{D}_t$
$\varkappa$	Number of matched key-points in I
(x_t, y_t)	Co-ordinate of key-point on $\mathbb{D}_t$
$\mathfrak{f}_t$	Feature vector at (x_t, y_t) on $\mathbb{D}_t$
(x_I, y_I)	Co-ordinate of key-point on I
$\mathfrak{f}_I$	Feature vector at (x_I, y_I) on I
k	Number of scales between products and rack
d_1, d_2	*Hamming distance* between keypoints
$matchThreshold$	Threshold for distance to be a match of keypoints
$ratioThreshold$	Threshold for ratio between the feature distances
M	Maximum distance between two feature vectors eligible to be a match
ζ	Number of clusters of key-points in I
$minimumPoints$	Number of points to form a cluster obtained using DBSCAN algorithm
$maximumRadius$	Maximum distance between any two key-points in a cluster obtained using DBSCAN algorithm
ρ_t	Number of clusters of matched key-points in I for $\mathbb{D}_t$
φ	Number of elements in a cluster
X_I^{n}	Column matrix representation of key-point $(x_I^{\text{n}}, y_I^{\text{n}})$
X_t^{n}	Column matrix representation of key-point $(x_t^{\text{n}}, y_t^{\text{n}})$
$\mathfrak{S}(A)$	Least squared sum of transformation A
$w, w', \tilde{w}$	Width of a product/sub-image/patch in pixels
$h, h', \tilde{h}$	Height of a product/sub-image/patch in pixels
h_p	Height of a product in cm
w_p	Width of a product in cm
τ	Number of valid sub-images during scale estimation in ERP
$\mathbb{H}, \mathbb{H}'$	Set of sub-images/patches/region proposal
l	Class label of a sub-image/product

$\mathbb{L}$	Set of labels of the products
s	Classification score of a sub-image/product
$\mathbb{S}, \mathbb{S}'$	Set of classification scores
x, y	Horizontal and vertical axes in 2D respectively
sc_x	cm-to-pixel x-scale
sc_y	cm-to-pixel y-scale
r_1, r_2, r_3, r_4	Corner points of a product
r'_1, r'_2, r'_3, r'_4	Corner points of a region proposal
$\mathfrak{c}$	Number of clusters in a rack for one scale using a product
χ, χ', χ''	Number of region proposals
C	Represent any color channel R, G, or B of an image
C'	Normalized C
μ	Mean
σ	Standard deviation
J	*all-ones* matrix
ϑ	Number of rack or shelf images
$scoreThresh$	Threshold for classification scores of region proposals
$\mathbf{y}$	True *one-hot* label vector for any product P
$\mathscr{y}$	Element of the label vector $\mathbf{y}$
n	Number of training (product) images in a dataset
$P^{(j)}$	j^{th} training (product) image
$\mathbf{y}^{(j)}$	True label vector for $P^{(j)}$
$P'^{(j)}$	Reconstructed image of $P^{(j)}$ using RC-Net
$\mathbf{y}'^{(j)}$	Predicted label vector for $P^{(j)}$ using RC-Net
$\theta, \theta', \theta''$	Trainable parameters of the encoder, decoder, and classifier in RC-Net respectively
$\theta^*, \theta'^*, \theta''^*$	Trained $\theta, \theta', \theta''$ respectively
$\mathscr{L}_{rl}(\cdot, \cdot; \theta, \theta')$	Reconstruction loss of RC-Net
$\mathscr{L}_{cl}(\cdot, \cdot; \theta'')$	Classification loss of RC-Net
$\tilde{\theta}$	Trainable parameters of conv-LSTM
$\tilde{\theta}^*$	Trained $\tilde{\theta}$
$\mathscr{L}_{lcl}(., .; \tilde{\theta})$	Loss of conv-LSTM
$\tilde{\mathbf{y}}$	Predicted label vector using conv-LSTM
$\hbar$	Hidden state of conv-LSTM unit
$\mathscr{c}$	Output of conv-LSTM unit
c	Number of channels of an image
$\mathbb{R}$	Set of real numbers
$\mathbf{X}$	Matrix representation of any product image P
f	Number of convolution filters or convolution feature maps
$\mathbf{X}_{in}, \mathbf{X}_{out}$	Input and output to and from SCAE respectively
$\mathfrak{z}$	Any one from the set $\{in, out\}$
h_d, w_d, c_d	Heights, widths and channels respectively for $\mathbf{X}_d, d \in \{in, out\}$
k_h, k_w	Height and width of a convolution filter respectively
$\boldsymbol{\kappa}$	Set of convolution filters
$*$	Convolution operation
s	Stride in convolution operation
s_w, s_h	Stride for width and height in convolution operation respecctively
ψ	Number of rows/columns padded in each side of the input

$\mathbf{M}$	A specific matrix representation of input $\mathbf{X}_{in}$
$\mathbf{K}$	A specific matrix representation of convolution filters in $\boldsymbol{\kappa}$
$\mathbf{W}_r$	Weights of output convolution layer of an SCAE with single hidden layer
$\mathbf{W}_p$	Weights of the fully-connected layer of an SCAE with single hidden layer
$\mathbf{X}'$	Reconstructed output for the input $\mathbf{X}$
$\mathbf{y}'$	Predicted label vector for the input $\mathbf{X}$
$\mathfrak{L}_p(\cdot,\cdot)$	Prediction/classification loss of SCAE
$\mathfrak{L}_r(\cdot,\cdot)$	Reconstruction loss of SCAE
$\mathtt{D}_1, \mathtt{D}_2$	Subsets of a (training) dataset $\mathtt{D}$ of product images
ϱ	A positive real number
$\mathbf{y}_p$	True label vector for the classification task
$\mathbf{y}_r$	True label vector for the reconstruction task
$\hat{\mathbf{y}}_p, \hat{\mathbf{y}}_r$	Predictions for $\mathbf{y}_p$ and $\mathbf{y}_r$ respectively
$B_{\mathbf{M}}, B_{\mathbf{y}}, B_{\mathbf{K}}, B_{\mathbf{W}_p}, B_{\mathbf{W}_r}$	Positive constants
$\mathtt{E}$	Representative set of training images
$\mathbf{E}^{(i)}$	Elements of $\mathtt{E}$
α	A real number
ϵ	A positive constant
$N(\cdot)$	Average reconstruction error for SCAE
a	A small positive number
σ_r, σ_p, g	Positive real constants
$\phi(\cdot)$	An activation function
$\boldsymbol{\Phi}_{\mathbf{K}}$	$\phi(\mathbf{K})$
η	Number of keypoints
(x_q, y_q)	Spatial location of the top left corner of q^{th} patch for a product image
$\mathscr{P}_q$	q^{th} patch of a product image (or part proposal)
$\mathscr{x}$	A volume of convolution maps
$\varkappa_1, \varkappa_2$	Number of rows and columns of conv maps
$\mathbf{v}$	A vector
A	Result of bilinear pooling
$\mathbf{A}$	Flattened 1D vector for A
a	Element of the vector $\mathbf{A}$
β	Number of discriminatory parts of a product
ν	Number of part proposals in a group
$S_{q_1q_2}$	Cosine similarity between q_1^{th} and q_2^{th} patches
S	Similarity matrix
$\mathcal{P}$	Sequence of patches
$\mathbf{y}_F$	Final classification score obtained from FGC
γ	Weightage of part-level classification score in FGC
n_T	Number of test product images
n'_T	Number of correctly recognized test product images
ps_H	Potential score of a region proposal H
$\mathtt{G}$	Directed acyclic graph of region proposals
$\mathbb{V}$	Set of vertices/nodes in $\mathtt{G}$
$\mathbb{E}$	Set of edges in $\mathtt{G}$
e	Cost/weight of an edge in $\mathtt{G}$
$\mathtt{S}$	Dummy source vertex in $\mathtt{G}$
$\mathtt{T}$	Dummy sink vertex in $\mathtt{G}$

$\mathbb{P}$	Set of predecessors of any vertex in $\mathtt{G}$
$\emptyset$	Null set
o	*Intersection-over-union* between two region proposals
$\mathbb{O}$	Set of *Intersection-over-union* values
z, z'	Index of region proposals
$IoUthresh$	Threshold for *intersection-over-union* between two region proposals
x	Superpixel
$\widetilde{\mathtt{G}}$	Graph of superpixels
$\widetilde{\mathbb{V}}$	Set of vertices of $\widetilde{\mathtt{G}}$
$\widetilde{\mathbb{E}}$	Set of edges of $\widetilde{\mathtt{G}}$
N	Number of superpixels
e	Edge of $\widetilde{\mathtt{G}}$
$\mathbf{p}(\cdot,\cdot)$	Pair-wise feature vector
$\mathbf{u}(\cdot)$	Unary feature vector
$\mathcal{A}$	Adjacency matrix of $\widetilde{\mathtt{G}}$
$\widetilde{\mathcal{A}}$	Adjacency matrix of $\widetilde{\mathtt{G}}$ considering self loops
$\mathcal{X}$	Structured data for any shelf image I
$\mathbf{f}_L$	Feature vector of a node of $\widetilde{\mathtt{G}}$ obtained from initial linear feature extractor
$\mathbf{f}_G$	Feature vector of a node of $\widetilde{\mathtt{G}}$ obtained from graph convolutional network
$\mathfrak{d}_1, \mathfrak{d}_2, \mathfrak{d}_3$	Dimension of feature vector
I_N	Identity matrix of size N
$\widetilde{\mathbf{D}}$	Intermediate variable of graph convolutional network
ℓ	Layer of a neural network
$\mathbf{H}^{(\ell)}$	Input to the ℓ^{th} layer of graph convolutional network
$\mathbf{W}^{(\ell)}$	Weight matrix of graph convolutional network for layer ℓ
ϑ_{trn}	Number of training shelf images
$I^{(\mathrm{k})}$	k^{th} training image
$\mathcal{X}^{(\mathrm{k})}$	Structured data for $I^{(\mathrm{k})}$
$Y^{(\mathrm{k})}$	True label vector for the superpixels in $I^{(\mathrm{k})}$
Ω	Set $\{0,1\}$
Y	Feasible label vector for $\widetilde{\mathtt{G}}$
$\hat{Y}$	Predicted label vector for $\widetilde{\mathtt{G}}$
$\mathcal{Y}$	Set of all possible feasible label vectors for $\widetilde{\mathtt{G}}$
y	Label of the vertex of $\widetilde{\mathtt{G}}$
$\mathrm{E}(\cdot,\cdot)$	Potential function
$\mathbf{w}$	A weight vector
$\boldsymbol{\omega}(\cdot,\cdot)$	A joint feature vector for an input $\mathcal{X}$ and its any label vector Y
$\mathrm{E}_\mathbf{u}(\cdot,\cdot)$	Potential function for unary feature vector
$\mathrm{E}_\mathbf{p}(\cdot,\cdot,\cdot,\cdot)$	Potential function for pair-wise feature vector
$\mathbb{I}(\cdot), \mathbb{I}'(\cdot,\cdot)$	Indicator functions
$\boldsymbol{\omega}_\mathbf{p}(\cdot,\cdot,\cdot,\cdot)$	Joint feature vector for pair-wise features
$\boldsymbol{\omega}_\mathbf{u}(\cdot,\cdot)$	Joint feature vector for unary features
λ	Positive constant
ε	Slack variable
$\Delta(\cdot,\cdot)$	Hamming loss
$F(\cdot)$	Prediction function
I_{gt}	True binary mask for shelf image I
B	True binary mask labeling superpixels for any shelf image

$\hat{B}$	Predicted binary mask for any shelf image
ϑ_{tst}	Number of test shelf images

Contents

CHAPTER 1

Introduction

For a long time, computer vision practitioners have been attempting to build machine vision system to automatically detect merchandise stacked in the racks of supermarket. By detection (or identification), we refer to recognition and precise localization of products visible in the images of racks in a supermarket. It is assumed that an ideal marketing image of the individual product is available to the vision system. The objective of such a vision system is (1) to generate an inventory of products available in the store at any point of time from the images of racks stacked with products (referred as out-of-stock detection problem), (2) to validate the plan of product display (often referred as *planogram*) with the actual display of merchandise (referred as planogram compliance problem), and finally (3) to provide a value-added experience to users (referred as shopping assistance problem).

The block diagram of the machine vision system under discussion is shown in Figure 1.1. In this chapter, we interchangeably use rack image as shelf image and product image as product template. A set of typical product images from the publicly available GroZi-120 dataset [Merler et al., 2007] is shown in

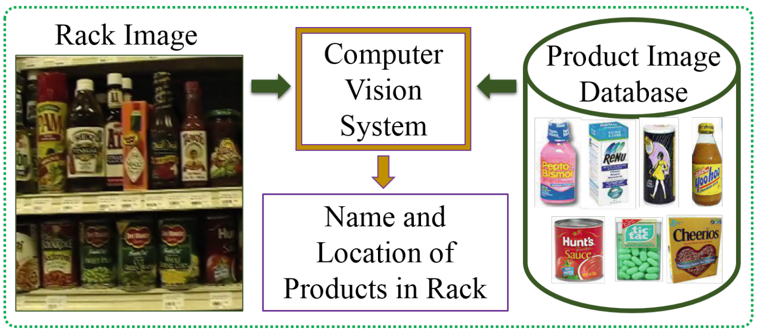
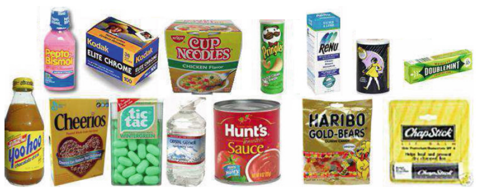
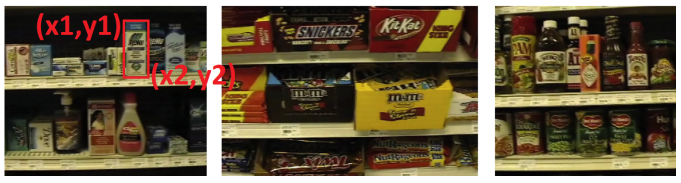


Figure 1.1: A typical computer vision system for detection of products in supermarkets



(a)



(b)

Figure 1.2: GroZi-120 dataset [Merler et al., 2007]: (a) sample product images typically used for marketing, (b) sample rack images where products are to be recognized and localized. $(x1, y1)$ and $(x2, y2)$ are the spatial co-ordinates of upper-left and bottom-right corners of a detected bounding box respectively.

Figure 1.2(a). Example rack images, where the product images of Figure 1.2(a) are to be detected, are shown in Figure 1.2(b).

Few attempts have been made to solve the above-mentioned problem using RFID, sensors, or barcodes [López-de Ipiña et al., 2011, Kulyukin and Kutiyanawala, 2010, Nicholson et al., 2009]. There are ubiquitous sensor based system (like AmazonGo [Bishop, 2016]) to monitor recognition and selection of products by a consumer. Most sensor based systems require fabrication at the manufacturer's end resulting in cost escalation of the product. Moreover, to assess the out-of-stock problem, a retailer needs to wait till the product leaves the store. Consequently, planogram compliance problem cannot be addressed with such sensors. Individual product based sensor has the problem of assessing the status of multiple products at one go. Devices for ubiquitous system have scalability issue and require significant investment. In contrast, computer vision based methods use hand phone camera or rack mounted camera to collect data. Overall, computer vision based approaches provide an inexpensive feasible alternative compared to sensor based approaches for automatic identification of retail products in the supermarkets. Given this, we explain why automatic identification of products is important.

1.1 Why Automatic Identification of Retail Products?

Commercial Benefits An estimate by Metzger *et al.* shows that out-of-stocks in supermarkets generally remain within a range of 5 to 10% [Metzger, 2008]. In [Gruen et al., 2002], Gruen *et al.* conduct a research on the impacts of out-of-stocks in retail stores worldwide. They find the following statistics due to out-of-stock situation: 31% shoppers move to another stores, 22% shoppers purchase another brand of the products, and 11% customers do not buy at all. The strategy for arrangement (planogram) of products in one or consecutive racks increases sales. Planogram establishes a close relation between shoppers, retailers, distributors, and manufacturers. It is observed that 100% optimized planogram compliance can increase sales up to 7 to 8% [Shapiro, 2009]. Hence, out-of-stock detection and checking of planogram compliance contribute to profit in retail businesses [Medina et al., 2015].

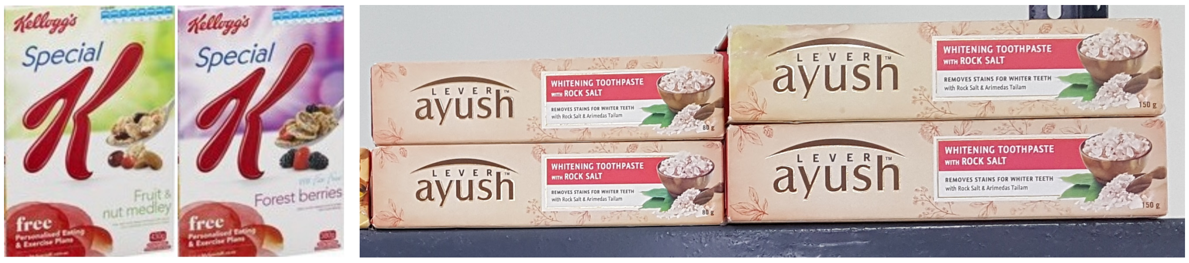
Table 1.1: Challenges in automatic recognition of retail products

Category	**Sub-category**
Retail Store Environment	• Complexity of scene • Data distribution • Variability of products • Fine-grained classification
Imaging in Retail Store	• Blurring • Uneven lighting conditions • Unusual viewing angle • Specularity

Enhanced Consumer Experience Real time information of availability of a particular product at a given location of the store reduces shopping time of a buyer. For visually challenged customers, information of availability of a particular product is a valuable consumer experience. There are close to 30 million people in the world who are suffering from blindness [WHO, 2014] and product availability information is always a value-added service for them. Next we discuss the challenges in designing a machine vision system for automatic identification of retail products.

1.2 Challenges in Identifying Retail Products

Table 1.1 summarizes the possible challenges of the product detection system. The racks are typically cluttered and often not organized in a regular fashion. Ideal marketing images of different products available to the vision system are often taken using different cameras resulting in different distributions of image intensities. Also, due to different imaging parameters, length of the product package (in some unit of length, say, cm) is mapped to different pixel resolutions for product and rack images. Examples of differences between product templates and rack images are evident in Figure 1.2(a) and 1.2(b). Product packages come in different shapes and sizes. There are minor promotional variations in product packaging and a product detection system must differentiate such minor variations. This identification of minor



(a) (b)

Figure 1.3: Fine-grained variations (a) color and text, and (b) size

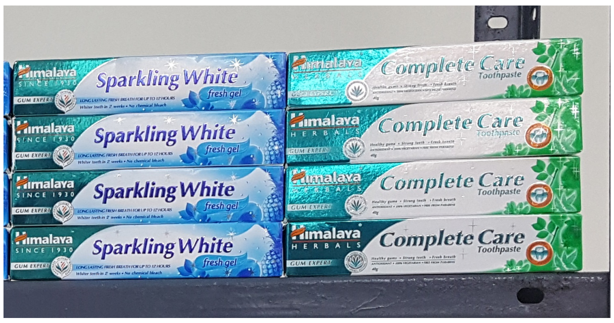


Figure 1.4: Rack image with vertically stacked products

signature variation in shape or color for a wide variety of products demand fine-grained classification. Figure 1.3 demonstrates a few examples of visually similar products having minute changes in color, text or size.

The rack images are captured using handheld devices. This often results in image blur due to camera shake and jitter (see the rack image in the middle in Figure 1.2(b)). Identification of products becomes difficult due to uneven illumination at the stores (see Figure 1.4). The challenge often extends to a scenario when the images of racks sometimes get distorted due to the glossy product packages and stacking (see Figure 1.4).

These reasons altogether pose significant challenge on top of typical object detection system studied in computer vision. The retail product detection problem bundles up various modalities of object detection problems like multiple object detection [Vo et al., 2010, Villamizar et al., 2016, Oh et al., 2016], detection of the multiple instances of the same object [Haladová and Šikudová, 2014, Aragon-Camarasa and Siebert, 2010], multiple object localization [He and Chen, 2008, Foresti and Regazzoni, 1994], multi-view object detection [Torralba et al., 2007], and fine-grained classification [Ge et al., 2016, Yao et al., 2017, Sun et al., 2017, Huang et al., 2017]. Next we analyze the features used in the attempts to recognize retail products.

1.3 Features for Detection of Retail Products

The feature descriptors for the problem under consideration are broadly classified as key-point based, gradient based, pattern based, color based and deep learning based features. The related works under these classifications are tabulated in Table 1.2. Next, we present a brief discussion on each of the groups.

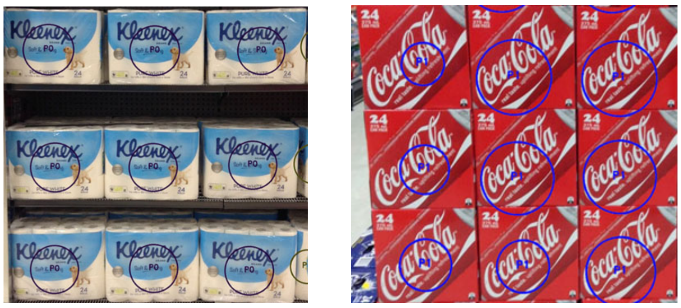
1.3.1 Key-point based Features

Key-point based features are the most used for recognition of retail products. Retail merchandises are packaged in colorful and catchy outfits. As a result, the image of product package generates a number

Table 1.2: Feature descriptors and corresponding approaches where these features are used.

Categories	Feature Descriptors	Approaches
Key-point based Features	SIFT [Lowe, 1999, 2004]	[Merler et al., 2007, Auclair et al., 2007, Zhang et al., 2007, 2009], [George and Floerkemeier, 2014, Varol et al., 2014, Bao et al., 2014], [Baz et al., 2016, Zhang et al., 2016b,a, Tonioni and Di Stefano, 2017]
	Dense SIFT [Bosch et al., 2007]	[Cleveland et al., 2016]
	SURF [Bay et al., 2006, 2008]	[Bigham et al., 2010, Winlock et al., 2010, Kejriwal et al., 2015], [Saran et al., 2015, Alhalabi and Attas, 2016, Brenner et al., 2016], [Yörük et al., 2016, Zientara et al., 2017b,a, Franco et al., 2017]
	AB SURF [Thakoor et al., 2013]	[Thakoor et al., 2013]
	Neo SURF [Ray et al., 2018]	[Ray et al., 2018]
	BRIGHT [Iwamoto et al., 2013]	[Higa et al., 2013]
Gradient based Features	Morphological Gradient [Dougherty, 1992]	[Frontoni et al., 2014]
	HOG [Dalal and Triggs, 2005]	[Marder et al., 2015, Varol and Kuzu, 2015, Pietrini et al., 2019]
	Sobel Operator [Sobel, 2014]	[Saran et al., 2015]
	Canny Edge Detector [Canny, 1987]	[Varol and Kuzu, 2015]
Pattern based Features	Haar-like Features [Papageorgiou et al., 1998]	[Merler et al., 2007]
	Recurring Patterns [Liu and Liu, 2013]	[Liu and Tian, 2015, Liu et al., 2016a, Goldman and Goldberger, 2017]
Color based Features	Color Histogram [Novak and Shafer, 1992]	[Merler et al., 2007, Bigham et al., 2010, Winlock et al., 2010], [Varol et al., 2014, Saran et al., 2015, Pietrini et al., 2019]
	Saliency [Itti et al., 1998]	[Thakoor et al., 2013, Marder et al., 2015, Zientara et al., 2017a]
	Color Constancy Model [Jameson and Hurvich, 1989]	[Gevers and Smeulders, 1999a,b, 2000, Gevers, 2001], [Diplaros et al., 2003, Gevers and Stokman, 2004, Diplaros et al., 2006]
Deep Learning based Features	CaffeNet [Jia et al., 2014]	[Dingli and Mercieca, 2016, Jund et al., 2016]
	AlexNet [Krizhevsky et al., 2012]	[Franco et al., 2017, Goldman and Goldberger, 2017]
	Inception-V3 [Szegedy et al., 2016]	[Chong et al.]
	VGG-f [Chatfield et al., 2014]	[Karlinsky et al., 2017]
	CNN [LeCun et al., 2012, Bengio et al., 2013]	[Zientara et al., 2017a, Sun et al., 2020, Yılmazer and Birant, 2021]

of key-points suitable for image matching. The key-points in most cases are detected using SIFT [Lowe, 1999, 2004] and SURF [Bay et al., 2006, 2008]. The methods in this category that deserve special attention are [Thakoor et al., 2013, Ray et al., 2018]. These approaches propose new variants of SURF namely AB-SURF [Thakoor et al., 2013] and NSURF [Ray et al., 2018] in detecting products. Overall Table 1.2 shows the importance of key-point based features. Local image characteristics in and around key-points are captured using a histogram [Lowe, 1999, 2004, Bay et al., 2006, 2008]. Stability of these histograms as features is one of the reasons for popularity of key-point based features for detecting retail products.



(a) (b)

Figure 1.5: Example images indicating recurring patterns by circles: the images are taken from [Liu and Tian, 2015].

1.3.2 Gradient based Features

Gradient based features (e.g., histogram of oriented gradients (HOG) or Sobel operator) are used for template based matching of product images extracted from images of racks. The geometric shapes like corners or edges embedded in product and rack images are also utilized for template matching. Similarly, as in case of key-point based features, gradient based local image characteristics are also captured using a histogram for detecting retail products [Marder et al., 2015, Varol and Kuzu, 2015].

1.3.3 Pattern based Features

In identifying retail products, the most common pattern based features are Haar or Haar-like features [Papageorgiou et al., 1998] and recurring patterns [Liu and Liu, 2013]. In this category, the recurring patterns play a vital role in detecting products as in [Liu and Tian, 2015, Liu et al., 2016a, Goldman and Goldberger, 2017]. In many real-life situation, similar yet non-identical objects often appear in a group like cars on the street, faces in a crowd and in context of this paper, products on a supermarket rack. The authors of [Liu and Liu, 2013] state that *much of our understanding of the world is based on the perception and recognition of shared or repeated structures*. In order to capture such repeated structures or recurrence nature, each product in a supermarket rack, act as a unit in a recurring pattern. Figure 1.5 demonstrates two example images of rack where the circles indicate recurring patterns. Recently, the authors of [Goldman and Goldberger, 2017] utilize the concept of recurring patterns in their proposed solution.

1.3.4 Color based Features

In detecting products, the color histogram [Novak and Shafer, 1992] and classical saliency features [Treisman and Gelade, 1980, Itti et al., 1998, Bruce and Tsotsos, 2007] of products can be considered as color based features. However, saliency and color histogram are sensitive to illumination changes common to a retail store. In order to tackle such illumination effects in color images, the authors of [Gevers and Smeulders, 1999a,b, 2000, Gevers, 2001, Diplaros et al., 2003, Gevers and Stokman, 2004, Diplaros et al., 2006] present various color based features using color constancy models for recognition of objects. List of color based features for product identification are given in Table 1.2.

1.3.5 Deep Learning based Features

In detecting retail products, all previously discussed four categories of features are hand crafted. In contrast, deep learning based features are derived from CNN pipeline [LeCun et al., 2012, Bengio et al., 2013]. For retail product detection, either the outputs of an intermediate layer [Franco et al., 2017, Dingli and Mercieca, 2016, Goldman and Goldberger, 2017] of a network are used as features or the network as a whole is utilized for both feature extraction and classification [Zientara et al., 2017a, Karlinsky et al., 2017, Dingli and Mercieca, 2016, Jund et al., 2016, Chong et al.]. In Table 1.2, we have compiled deep learning related references. Next we present the taxonomy of the state-of-the-art methods of recognition system of retail products.

1.4 A Taxonomy for Detecting Retail Products

The first serious attempt [Gevers and Smeulders, 1999a] of recognition of retail products in isolation (i. e., identification of individual product image cropped from the rack image) was in 1999. Naturally, localization issue is not addressed in this work. It took almost another eight years to take a more involved approach for recognition and localization of multiple retail products. In 2007, Merler *et al.* [Merler et al., 2007] introduce the retail product detection problem along with a dataset containing rack and product images. Since then many research papers have been published directly related to retail product detection system. In Table 1.3, we propose a taxonomy for automatic detection of retail products.

From the pattern of development over the last decade, we find two major sequential steps as noted in Table 1.3. In the first layer of taxonomy, a probable region (containing a product) on the rack is identified based on an objectness (or productness) measure. We group the methods in the first layer in five different approaches: block, geometric transformation, saliency, detector, and user-in-the-loop based methods. Moreover, block based methods are classified into sliding window and grid based methods.

In the second layer of taxonomy, each method is partitioned into two groups namely unsupervised and supervised approaches of object detection. While using the terms supervised and unsupervised approaches, we have relied on the classical definitions used in the machine learning literature [Szeliski, 2010]. The unsupervised methods mainly include template based matching. The supervised methods refer to building a model using a train-set. The trained model is used to test a new set of data unseen to the model.

The Table 1.3 also presents different areas of applications and corresponding categories of the problem. The areas of applications are (AI) Shopping assistive system (AII) Out-of-stock detection (AIII) Planogram compliance. The categories of the detection problem addressed in the listed papers in Table 1.3 are:

(DI) Detection of single product: This relates to accurate identification and localization of only one product at a time in a rack image.

(DII) Detection of multiple products: This relates to accurate identification and localization of all the products in a rack image in one go.

(DIII) Recognition of products: This relates to recognition or classification of isolated products where

Table 1.3: Taxonomy of computer vision based approaches for product detection in retail stores (*: these methods crop product images from rack image either manually or using planogram information): for details, refer to text in Section 1.4.

			Unsupervised Methods	Supervised Methods	Area of Application	Category of Problem
Automatic Product Detection in Retail Stores	Block based Methods	Sliding Window based Methods	[Merler et al., 2007]		AI	DII
			[Marder et al., 2015]		AII	DII
			[Saran et al., 2015]		AIII	DII
			[Ray et al., 2018]		AIII	DII
				[Pietrini et al., 2019]	AII, AIII	DII
		Grid based Methods	[Zhang et al., 2007]		AI	DIV
			[Zhang et al., 2009]		-	DIV
			[Bigham et al., 2010]		AI	DI
			[Higa et al., 2013]		AII	DII
				[George and Floerkemeier, 2014]	AI	DII
				[George et al., 2015]	AI	DII
	Geometric Transformation based Methods		[Merler et al., 2007]		AI	DII
			[Auclair et al., 2007]		-	DII
			[Bao et al., 2014]		AIII	DII
			[Kejriwal et al., 2015]		AII	DII
			[Alhalabi and Attas, 2016]		AI	DI
			[Brenner et al., 2016]		AI	DI
			[Yörük et al., 2016]		-	DII
			[Zhang et al., 2016b]		AIII	DII
			[Zhang et al., 2016a]		-	DII
			[Zientara et al., 2017b]		AI	DI
			[Tonioni and Di Stefano, 2017]		AIII	DII
				[Cleveland et al., 2016]	AIII	DII
	Saliency based Methods		[Winlock et al., 2010]		AI	DII
			[Thakoor et al., 2013]		AIII	DI
			[Frontoni et al., 2014]		AIII	DI
			[Franco et al., 2017]		AI	DII
				[Zientara et al., 2017a]	AI, AIII	DII
				[Franco et al., 2017]	AI	DII
				[Goldman and Goldberger, 2017]	-	DII
				[Sun et al., 2020]	-	DII
				[Yılmazer and Birant, 2021]	AII	DII
	Detector based Methods			[Merler et al., 2007]	AI	DII
				[Varol et al., 2014]	AIII	DII
				[Varol and Kuzu, 2015]	AIII	DII
				[Karlinsky et al., 2017]	-	DII
	User-in-the-loop Methods		[Liu and Tian, 2015]*		AIII	DIII
			[Liu et al., 2016a]*		AIII	DIII
			[Gevers and Smeulders, 1999a]		-	DIII
			[Gevers and Smeulders, 1999b]		-	DIII
			[Gevers and Smeulders, 2000]		-	DIII
			[Gevers, 2001]		-	DIII
			[Diplaros et al., 2003]		-	DIII
			[Gevers and Stokman, 2004]		-	DIII
			[Diplaros et al., 2006]		-	DIII
				[Advani et al., 2015]*	-	DIII
				[Baz et al., 2016]*	-	DIII
				[Dingli and Mercieca, 2016]*	AI	DIII
				[Jund et al., 2016]*	-	DIII
				[Chong et al.]	AIII	DIII
				[Santra et al., 2020a]*	-	DIII

localization is not important.

(DIV) Retrieval of rack images: Given a pool of rack images, the goal is to retrieve the rack images containing the query product.

Note that out of all state-of-the-art methods for detecting retail products (provided in Table 1.3), only five methods [Zientara et al., 2017b, Tonioni and Di Stefano, 2017, Liu and Tian, 2015, Liu et al., 2016a, Baz et al., 2016] assume the presence of planogram in order to locate the products in a rack. In these methods, planogram informs the algorithm about the particular product expected in a given location of the rack. Naturally, test for absence or presence of the expected product at a given location reduces the challenge of discovering a product in absence of planogram information. Next we briefly discuss and assess each group of the taxonomy. We start with the first approach, block based methods.

1.4.1 Block based Methods

In block based methods, several overlapping or non-overlapping blocks are selected from the rack image as potential regions containing products. Consequently, local features (like SIFT [Lowe, 1999, 2004], and SURF [Bay et al., 2006, 2008]) are computed from each such block and also from each of the product templates. For each block of rack image, the features are matched with those of product images. The product image with highest matching score is selected as the product for the block. The final detection result is generated after applications of various post processing techniques [Franco et al., 2017]. As mentioned earlier, the block based methods are classified into two categories: (a) Sliding window, and (b) Grid based methods. A graphical illustration of sliding window based methods is presented in Figure 1.6 while the overview of grid based methods is graphically demonstrated in Figure 1.7.

PROS AND CONS: The primary advantages of block based methods are that the schemes are simple and easy to implement. The critical disadvantage is: how to choose the number and size of overlapping or non-overlapping blocks? In most cases, authors have chosen these parameters either experimentally or from prior knowledge. Thus, accurate localization of products cannot be guaranteed in many cases. Moreover, the overlapping block based methods are computationally expensive.

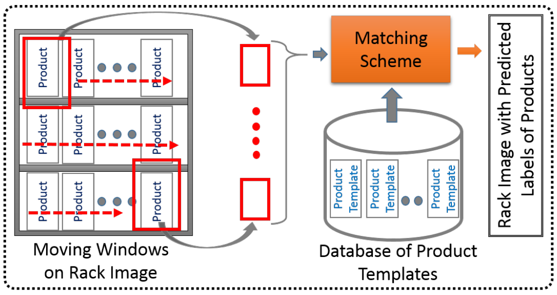

Product
Moving Windows on Rack Image
Matching Scheme
Product Template
Database of Product Templates
Rack Image with Predicted Labels of Products

Figure 1.6: Block diagram of a sliding window based method

The block based methods consider enormous number of sliding windows of different scales and sizes to locate the products in a rack. In other words, these methods exhaustively search for the products in the rack. As a result, these methods are robust against rotation and scaling of products in the rack.

On a different point, the slow execution of these methods is a major drawback in designing a real time system like shopping assistive application. To avoid exhaustive search for products in a rack, the geometric transformation based matching or graph theoretic approach looks like a promising direction of research. Next section presents the geometric transformation based methods.

1.4.2 Geometric Transformation based Methods

In retail store setting, images of racks captured by a handheld device undergo geometric transformation due to oblique view of camera with respect to the rack. As a result approaches in this group attempt to calculate features which are invariant to affine or projective object to image transformation. Most of the approaches in this group evaluate key-point based local features (using SIFT, SURF etc.) for the rack and product images. The key-point correspondences between rack image and product images are obtained using various techniques like clustering of key-points or Hough voting. Finally, using these key-point correspondences, products are recognized and localized in the rack image. In Figure 1.8, we demonstrate a typical geometric transformation based method.

PROS AND CONS: Geometric transformation based methods typically assume that the key-points are identified correctly and key-point correspondences are established accurately. Naturally, the performance of the schemes discussed above are dependent on assumptions related to key-points.

If the products displayed on a rack are planar, homography estimation is not strictly necessary. Also, SIFT and SURF features are not sensitive to affine transformations between product and rack images. Unfortunately in retail store setting it is difficult to ensure the correct estimation of key-points. key-points in a rack image are often missed due to poor illumination. On the other hand, more than desired number of key-points are detected in a noisy rack image with cluttered background. This yields many incorrect geometric transformations between products and rack images.

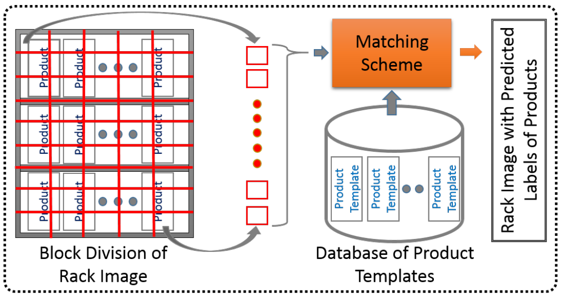

Product
Block Division of Rack Image
Matching Scheme
Product Template
Database of Product Templates
Rack Image with Predicted Labels of Products

Figure 1.7: Block diagram of a grid based method

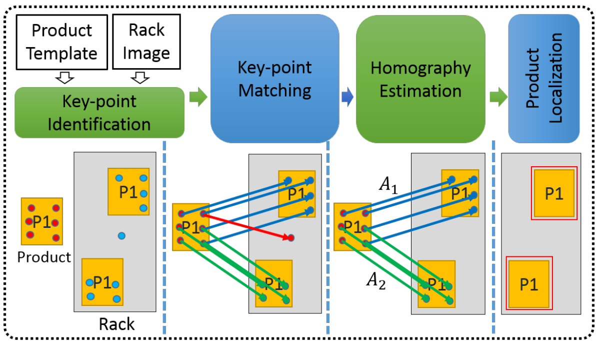


Figure 1.8: Block diagram of a geometric transformation based method: colored dots denote the key-points, P1 represents a product and A_1, A_2 are the homographies between the product P1 & rack.

However, for correct estimation of geometric transformations, scaling and rotation issues between product and rack images are automatically addressed. The entire approach is fast and suitable for real time implementation. Overall, geometric transformation based methods are promising and can be integrated with other approaches for a reliable result. Next we present saliency based methods.

1.4.3 Saliency based Methods

Saliency based methods localize products in a rack image by utilizing saliency maps [Treisman and Gelade, 1980, Zientara et al., 2017a], gradient image [Frontoni et al., 2014], or by finding out potential regions [Franco et al., 2017] of rack image. Once the salient region of a rack image is determined, the local features of those interest regions are calculated and matched with that of product images. The block diagram of a typical saliency based method is presented in Figure 1.9.

PROS AND CONS: Saliency based methods are two-layered. In the first layer, the salient regions are identified in a rack image. The second layer matches the salient regions with products. In most cases, the first layer of these methods do not miss to identify regions containing products. But at the same time, the first layer tends to over-estimate the salient regions. As a result the saliency based localization methods usually fail when rack image contains partially-occluded products.

The second layer minimizes false detection due to cluttered background of the rack image. Like block and geometric transformation based methods, the salient region based methods also take care of rotations and scaling of products in rack. The implementation of second layer is relatively fast as the recognition is executed only for the salient regions. Shopping assistive system implemented with any of these methods can always be operated in real time.

Newer salinecy based deep learning tools like R-CNN [Girshick et al., 2014], Fast R-CNN [Girshick, 2015], Faster R-CNN [Ren et al., 2015], Mask R-CNN [He et al., 2017], and SSD [Liu et al., 2016c] are yet to be explored for the problem under consideration. These methods require training data comprises

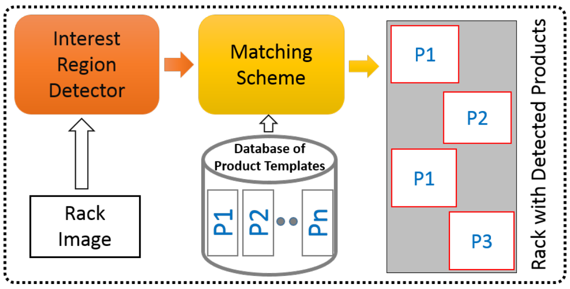


Figure 1.9: Block diagram of a saliency based method: P1, P2, · · · , Pn are the products.

of annotated rack images where each product is labeled with bounding boxes. However, in a retail store environment, capturing images of racks and annotating the same for building a significant training dataset, is a painstaking activity. In contrast, the template of the new packaging of a product is made publicly available before its actual arrival at the store. Therefore, template driven approaches are preferred for detecting products in supermarket.

Overall, saliency based methods require attention for detecting partially-occluded products, which is a normal situation in a retail store. Next section presents detector based methods.

1.4.4 Detector based Methods

For various real world objects like face [Viola and Jones, 2001] or pedestrian [Papageorgiou and Poggio, 1999], there exists reliable dedicated detectors. Detector based methods (e.g. [Merler et al., 2007]) separately train a machine learning tool (like AdaBoost [Freund et al., 1999]) with certain domain-specific visual features (e.g. Haar-like features [Papageorgiou et al., 1998]) of product images to find out bounding boxes of products in the rack image. Once the bounding boxes are detected, the local visual features are extracted from the regions of rack image for matching with the product images. Figure 1.10 demonstrates a graphical illustration of detector based methods.

PROS AND CONS: Like saliency based methods, these are also two-layered approaches. First layer detects the bounding boxes using a object detector while the second layer takes care of the classification

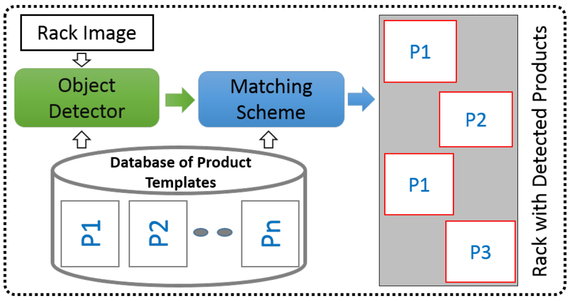


Figure 1.10: Block diagram of a detector based method: P1, P2, · · · , Pn are the products.

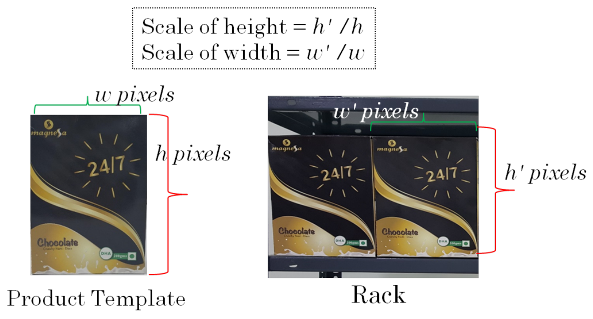
of products. The object detector trained with partially-occluded objects can determine the bounding boxes for partially-occluded products in a rack. The object detector of the first layer could also identify the product class for a bounding box. Utilizing this class information, the second layer may even perform finer classification of products (e.g. challenges shown in Figure 1.3). However, this is not yet explored for methods in this category. The object detector trained with rotated and scaled objects could be a promising application. These approaches are suitable for real time applications.

In retail store setting, the intensity distributions of training and test images are not necessarily identical (as discussed in Section 1.2). As a result, the detector based approaches, especially those using learning schemes like AdaBoost, may fail to identify the bounding boxes. However, the statistical learning based object detector as in [Karlinsky et al., 2017] does not suffer seriously from such problems. Next we present user-in-the-loop based methods.

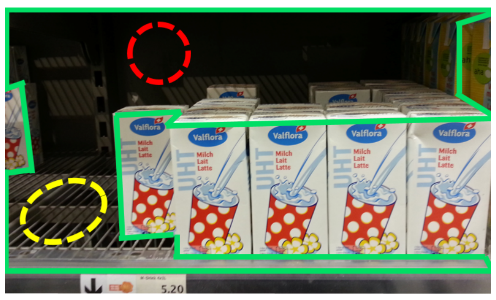
1.4.5 User-in-the-loop Methods

User-in-the-loop methods do not automatically localize products on the rack. In the rack image, products are cropped out either manually or by utilizing product's information provided in a planogram. Subsequently, local features of cropped products are matched with that of product images. Thus, user-in-the-loop methods do not address primary challenges of localization of products on the rack.

Pros and Cons: These methods do not localize products in a rack image. Only classification or recognition performances (not detection) are judged using these user-in-the-loop approaches. As a result, these methods always show better detection performances than other related methods. In a realistic store level scenario with difficult challenges like identification of multiple shelves in a rack or identification of rack start and rack end points in a rack image, user-in-the-loop looks like a promising approach. This approach has the potential to detect a novel product (not already available with product dataset) or to identify a gap (missing product) in the rack space. Both these applications of novel product identification



(a)



(b)

Figure 1.11: (a) Graphical illustration of scale between the template of a product and the product present in a rack, and (b) empty spaces marked with the green polygon in a shelf image where red and yellow dotted circles show different textures.

and gap identification have major commercial impacts for retailers.

Next we discuss the objective of the thesis which addresses a few unsolved issues that the above mentioned state-of-the-art methods (see Table 1.3) have somehow missed or failed to address in context of identification of retail products.

1.5 Objective of the Thesis

The objective of this thesis is to design an end-to-end efficient solution that should achieve better performance in identifying retail products in the supermarkets. The thesis goes into details of important building blocks of this end-to-end solution and offer both theoretical and practical solutions.

The performance limit of object recognition systems using computer vision is being pushed constantly for over last 50 years now [Andreopoulos and Tsotsos, 2013]. In contrast, the community is not that active in this particular problem space that we have been discussing in this thesis. In this context, there exists many challenges [Santra and Mukherjee, 2019], some of which we address in this thesis.

First of all, we intend to generate some exemplar-driven region proposals estimating the scale between product templates and rack in order to detect products on the rack. Note that the region proposals are generated from the examples of product templates. Hence, this process is referred to as exemplar-driven. We then aim to classify the fine-grained products with uneven illuminations. Next we plan to accurately detect the vertically stacked products on the rack introducing a graph theoretic approach. Finally, we try to identify the empty spaces in each shelf of any rack. In a nutshell, the building blocks of this thesis are as follows:

- Estimate the scale between product templates and rack (see Figure 1.11(a))
- Generate exemplar-driven region proposals
- Classify fine-grained products (refer Figure 1.3)
- Classify the products with uneven illuminations (see Figure 1.4)
- Identify vertically stacked products (look at Figure 1.4)
- Identify empty spaces on the shelves (see Figure 1.11(b))

In order to fulfil the above objectives, the proposed solutions should have the following key characteristics [Santra and Mukherjee, 2019]:

- **Real-time** From the consumer perspective, the system must operate in real-time such that availability of products can be checked immediately. From the retailer's stand point, the system should operate close to real-time given that the store must respond to consumer need for replenishing the stock. From the perspective of system integrator, there is always a trade off between processing the image at the hand held device (where the rack image is captured) or at the background or cloud.

- **Accuracy** The system should consistently operate at high level of accuracy for a wide range of products in order to establish its acceptability within the consumers and retailers. There should be minimum or no user interaction.
- **Robustness** The key challenges come from mismatch in scale between a product template and the rack image, uneven illumination, variation in camera angle, and unstable image capturing due to hand held devices. From the machine learning stand point (especially considering deep learning architecture), the major bottleneck is the availability of single instance of the product image. Naturally, synthetic generation of training images using data augmentation technique requires special attention for machine learning based technique to improve its performance.

Organization of the thesis and chapter-wise contributions are described next.

1.6 Organization of the Thesis and Chapter-wise Contributions

The overall plan of the entire thesis is shown in Figure 1.12, where the contributory chapters are highlighted with blue color. Following this introductory Chapter 1, we present an end-to-end annotation-free machine vision system by introducing our novel exemplar-driven region proposal scheme in Chapter 2. In Chapter 3, we propose a fine-grained classifier for improving the product detection performance of our system (presented in Chapter 2). In Chapter 4, we propose an efficient graph-based non-maximal suppression of region proposals for accurately detecting stacked products using our system (proposed in Chapter 2). In Chapter 5, we provide a unique solution for identification of the empty spaces on the shelves of a rack in supermarket. Finally, Chapter 6 presents the concluding remarks along with

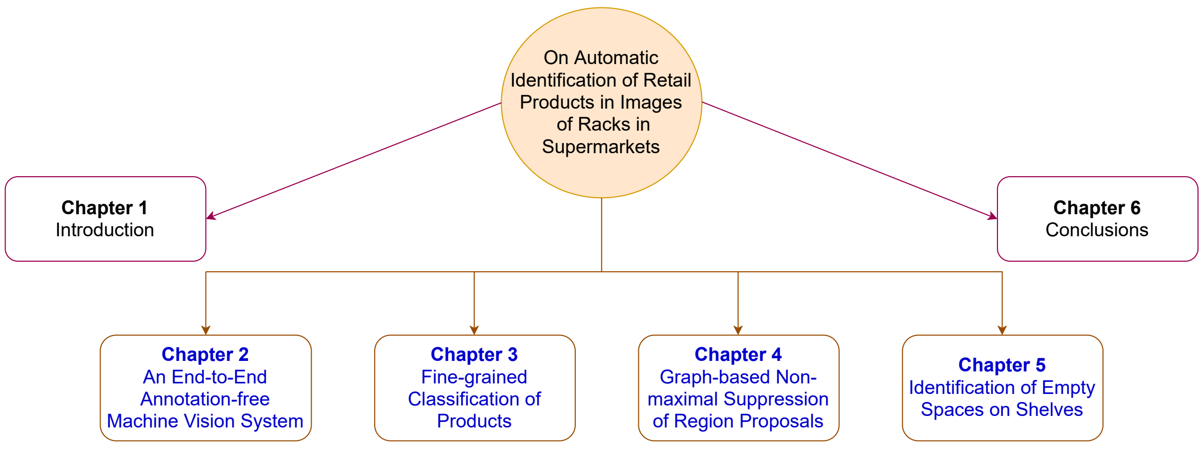

On Automatic Identification of Retail Products in Images of Racks in Supermarkets
Chapter 1 Introduction
Chapter 6 Conclusions
Chapter 2 An End-to-End Annotation-free Machine Vision System
Chapter 3 Fine-grained Classification of Products
Chapter 4 Graph-based Non-maximal Suppression of Region Proposals
Chapter 5 Identification of Empty Spaces on Shelves

Figure 1.12: Chapter-wise organization of the thesis

the possible future scopes of research for the problem of retail product detection. Next we discuss the chapter-wise summary and contributions.

1.6.1 Contributions of Chapter 2

In this chapter, we build an end-to-end machine vision system [Santra et al., 2021b] for detecting products in the supermarkets given a single instance (or template image) per product. Recent deep learning based solutions [Girshick et al., 2014, Girshick, 2015, Redmon et al., 2016] can efficiently detect the products utilizing annotated images of racks (marking each product by rectangular bounding boxes) during training of these methods. But (manually) annotating the images of racks is an infeasible step with the fast changing line of products. Thus, as an alternative, we propose an efficient annotation-free system for identification of products by introducing an exemplar-driven region proposal approach (i.e., the process of generating region proposals in a shelf image with the help of an example of template of each product). We find that the exemplar-independent region proposal schemes (where the region proposals are extracted from the entire rack image without the help of product templates) like selective window search in R-CNN [Uijlings et al., 2013, Girshick et al., 2014] often generates multiple region proposals on the background region of the rack image. And this leads to false detections in the rack images. On the contrary, the proposed exemplar-driven region proposal approach generates region proposals around the products and hence, our method overcomes the false detection issue of exemplar-independent region proposal schemes. The proposed two-stage exemplar-driven region proposal works with the example or template of the product. The first stage estimates the scale (see Figure 1.11(a)) between the template images of products and the rack image. The second stage generates proposals of potential regions using the estimated scale. Subsequently, the potential regions are classified using convolutional neural network. The generation and classification of region proposal do not need annotation of rack image in which products are recognized. In the end, the products are identified removing ambiguous overlapped region proposals using greedy non-maximal suppression.

1.6.2 Contributions of Chapter 3

The proposed product identification system of Chapter 2 often shows poor performance due to mis-classification of similar yet non-identical (i.e fine-grained) products. A few such fine-grained products are illustrated in Figure 1.3. Therefore, in this chapter, we look into the classification of very similar images (fine-grained classification) of variants of retail products. We propose a novel solution that simultaneously captures object-level and part-level cues of the product images. The object-level cues of the product images are captured with our novel reconstruction-classification network (RC-Net). The marketing images of these products captured in a studio-like environment, are the templates for this recognition problem. The variation in the quality and illumination between a rack image and corresponding template images are difficult to bridge. The generalization ability of a classifier tackles this variation in quality between rack and template images. The proposed RC-Net, being a supervised convolutional auto-encoder (SCAE), has the generalization ability and hence, resolves the illumination difference between product templates and rack images to some extent. An SCAE essentially optimizes both reconstruction and classification losses to recognize the objects. We show that a linear SCAE becomes uniformly stable [Le et al., 2018] and provides a necessary generalization bound that we derive in this chapter. Further, we derive the generalization bound for a non-linear SCAE (consisting of single hidden layer) with RELU activation function. On the other hand, the fine-grained differences between the products are addressed by capturing the part-level cues of the products. For annotation-free modelling of part-level cues, the discriminatory parts of the product images are identified around the key-points in an unsupervised manner. The ordered sequences of these discriminatory parts, encoded using convolutional LSTM [Xingjian et al., 2015], describe the products uniquely. Finally, the part-level and object-level models jointly determine the products explicitly explaining coarse to finer descriptions of the products. This bi-level architecture is embedded in our product detection system (developed in Chapter 2) for recognizing variants of retail products on the rack.

1.6.3 Contributions of Chapter 4

The proposed exemplar-driven region proposal scheme (in Chapter 2) [Santra et al., 2021b] generates (mostly overlapped) region proposals around the products on the rack. Subsequently, the region proposals are classified using our fine-grained classifier (proposed in Chapter 3). Finally, we had implemented a greedy non-maximal suppression (greedy-NMS) [Felzenszwalb et al., 2010] to disambiguate the overlapping proposals in Chapters 2 and 3. Greedy-NMS discards the proposals (with lower classification scores) that are overlapped with the proposal with higher classification score. This greedy approach sometimes eliminates the (geometrically) better fitted region proposals with (marginally) lower classification scores. In this chapter, we introduce a novel graph-based non-maximal suppression (G-NMS) [Santra et al., 2020b] that removes this critical bottleneck of greedy-NMS by looking not only at the classification scores but also at the product classes of the overlapping region proposals. G-NMS first determines the *potential confidence scores (pc-scores)* of the region proposals by defining the groups of overlapping regions. The *pc-scores* of the proposals elegantly combine classification scores, class labels and extent of overlaps of the proposals to accurately disambiguate the overlapping proposals. Subsequently, a directed acyclic graph (DAG) is strategically constructed with the proposals utilizing their *pc-scores* and overlapping groups. Unlike the DAG in [Ray et al., 2018], we build the DAG in a way that all the vertically stacked products (see Figure 1.4) are correctly identified. Finally, the maximum weighted path of the DAG detects the products that are present in the rack.

1.6.4 Contributions of Chapter 5

The discussions so far have concentrated on designing an efficient product detection system. However, automatic detection of empty spaces in the images of the shelves is an uncharted territory. In this chapter, our objective is to automatically identify the empty spaces anywhere on the shelves. Different empty spaces may have different textures, colors or other features. For example, in Figure 1.11(b), the regions highlighted by the red and yellow dotted circles on the shelf present different textures. The absence of unique inherent characteristics of empty spaces amplify the challenges to solve gap detection problem. Note that, gap, empty region and empty space are interchangeably used in this thesis. In order to tackle every type of empty spaces, we pose this challenge as a segmentation problem and propose a unique solution which automatically identifies the gaps/empty spaces anywhere in the entire image of a shelf. Recent advanced deep learning based image segmentation approaches [Ronneberger et al., 2015, Chen et al., 2017, Chaurasia and Culurciello, 2017, Lin et al., 2017, Zhao et al., 2017, Chen et al., 2018, Li et al., 2018, Fan et al., 2020] show excellent performance, but with enormous amount of annotated images. For the empty space identification problem, benchmark labeled data (i.e., the annotated shelf image) is still not available. However, the publicly available benchmark datasets of retail products include a limited number of images of shelves. We release few annotated images of these selves for identification of empty spaces at `https://github.com/gapDetection/gapDetectionDatasets`. This chapter introduces an efficient graph-based method of superpixels of the shelf images modelled with a minimal set of annotated shelf images. First, an image of a shelf is over-segmented into a number of superpixels to construct a graph of superpixels. Subsequently, a graph convolutional network [Kipf and Welling, 2016] and a Siamese network [Koch et al., 2015] architecture are uniquely built for extracting the features representing nodes and edges of the graph respectively. Finally, a structural support vector machine [Xue et al., 2008] based inference model formulated based on the graph of superpixels, is solved as a one-slack support vector machine [Joachims et al., 2009] for segmenting the empty and non-empty regions from the shelf image.

1.6.5 Contributions of Chapter 6

In this chapter, we conclude our thesis by summarizing our contributions and future directions of research in context of the problem under discussion. Further at the end of this chapter, we highlight some challenges that originate due to the developments explored in this thesis.

CHAPTER 2

An End-to-End Annotation-free Machine Vision System

2.1 Introduction

Humans effortlessly identify merchandise displayed on the racks in a supermarket. But the integration of such skill in a smart machine vision system poses many challenges [Santra and Mukherjee, 2019]. In the previous chapter, we have listed a number of works like [Merler et al., 2007, Zhang et al., 2007, George and Floerkemeier, 2014, Marder et al., 2015, Franco et al., 2017, Ray et al., 2018] which focused on creating machine vision system for automatic identification of retail products. In this chapter, we introduce a machine vision system that improves the product detection performance compared to previous attempts.

The advent of deep learning based schemes like R-CNN [Girshick et al., 2014], Fast R-CNN [Girshick, 2015], YOLO [Redmon et al., 2016], SSD [Liu et al., 2016c] and Mask R-CNN [He et al., 2017] shows significant improvement in performance for object detection. These methods cannot be used straightway as they need annotated rack images for training. These annotations are difficult to obtain with frequent changes in promotional packages and product display plan. The exemplar-independent R-CNN [Girshick et al., 2014] and its variants do not use the product template for generating region proposals. Rather they depend on the annotation of the rack image to learn shape and scale of different products. In order to rectify the generated product proposals, exemplar-independent R-CNN optionally implements *bounding box regression* which again demands annotated rack images. On the contrary, the proposed annotation-free system (which does not require annotated rack images) introduces an exemplar-

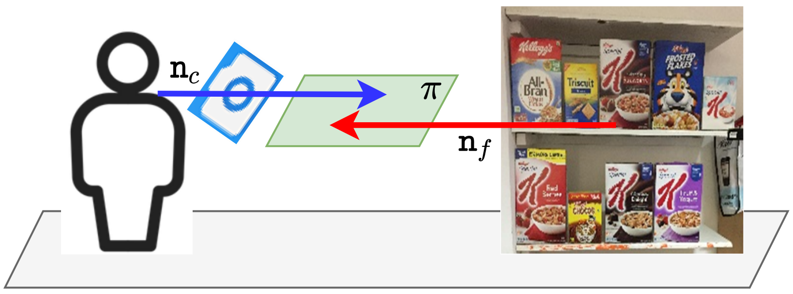


Figure 2.1: Illustration of fronto-parallel imaging of a rack.

driven region proposal scheme to extract the region proposals around the products displayed on the rack using individual marketing images of the products.

The proposed solution is obtained under the following assumptions: (i) All rack images are captured where the camera is nearly fronto-parallel with respect to face of the rack. Assume that $\mathtt{n}_f$ is the normal to the plane representing the face of the rack. Also assume that the normal to the image (camera) plane is $\mathtt{n}_c$. If $\mathtt{n}_f$ and $\mathtt{n}_c$ are either collinear or mutually parallel contained in a plane, say π, and the plane π is parallel to the ground plane (on which the cameraman and the racks are standing), then the image capturing position is fronto-parallel. In connection to this, we assume that perspective effects and projective transforms are not considered during capturing of the images of racks. This is shown in Figure 2.1. (ii) The physical dimension of each product template is available in some suitable unit of length. In case of absence of physical dimension of the products, we use the context information of retail store. The context information assume similar products are arranged together in rack for shopper's convenience. This context information essentially helps us to assume that the physical dimension is equivalent to the dimensions of a product in pixels. The use of physical dimensions is based on the previous assumption of fronto-parallel viewing. If the images of racks are fronto-parallel, then the object-to-image scale is accurately evaluated. Subsequently, accurate estimation of scale leads to generation of tight-fit region proposals for the products on shelves leading to better recognition.

As shown in Figure 2.2, our proposed system first introduces a two-stage exemplar-driven region proposal algorithm using the information of individual product images. The region proposal is exemplar-driven as the proposals are generated based on individual example or image templates of products. Each proposal is then classified by a convolutional neural network (CNN) [He et al., 2016]. The product classification does not need any annotation of the rack image from where the individual products are identified.

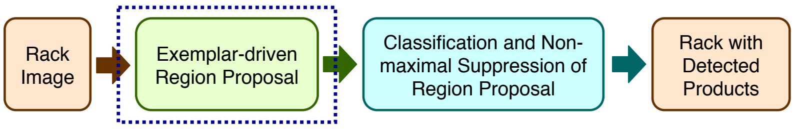

Rack Image
Exemplar-driven Region Proposal
Classification and Non-maximal Suppression of Region Proposal
Rack with Detected Products

Figure 2.2: Flow chart of the proposed scheme, blue colored dotted-rectangle highlights our contribution.

Subsequently, greedy non-maximal suppression technique [Felzenszwalb et al., 2010] is implemented to remove the overlapping and ambiguous region proposals. The contribution of this work compared to state-of-the-art methods is two fold:

(a) We propose an automated approach to estimate the scale between product(s) and rack image. The previous attempts either move windows of different scales over the rack image [Ray et al., 2018, Marder et al., 2015, Merler et al., 2007] or divide the rack image into a number of grids of different resolutions [George and Floerkemeier, 2014, Zhang et al., 2007] to find out potential regions in the rack.

(b) We introduce an exemplar-driven region proposal scheme for detecting objects in a scene crowded with products in contrast to annotation based region proposal scheme (without taking the help of product images) using *bounding box regression* [Girshick et al., 2014] in R-CNN or its variants.

The rest of the chapter is structured as follows. Section 2.2 describes the proposed system. The experimental analysis is performed in Section 2.3. Finally, Section 2.4 summarizes the chapter with concluding remarks.

2.2 Method M1: Annotation-free Product Identification

Overall methodology of the proposed scheme is demonstrated in Figure 2.2. Our solution takes care of both recognition and localization of products (including multiple products stacked horizontally or vertically) in the rack. The following subsections present the modules of our scheme.

2.2.1 Exemplar-driven Region Proposal

The probable locations of products in a rack are determined through the proposed exemplar-driven regional proposal (ERP) scheme. As shown in Figure 2.3, the proposed ERP takes a rack image as input and returns a number of region proposals for the rack (green rectangular boxes in Figure 2.3). This region proposal scheme is a two-stage process: estimation of scale between the product image and the rack image, and the extraction of region proposal (see Figure 2.3).

Let m be the number of individual products in a database of product templates $\mathbb{D}$. In other words,

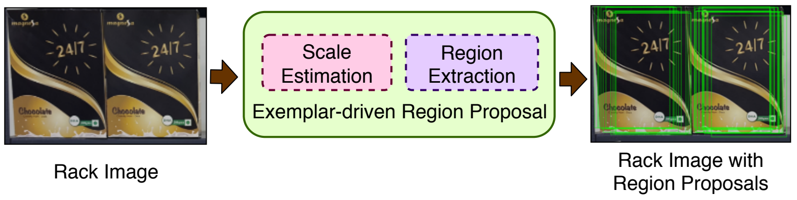

Rack Image
Scale Estimation
Region Extraction
Exemplar-driven Region Proposal
Rack Image with Region Proposals

Figure 2.3: Flow chart of the proposed two-stage exemplar-driven regional proposal. Green rectangular boxes on the rack are the regional proposals generated by our ERP.

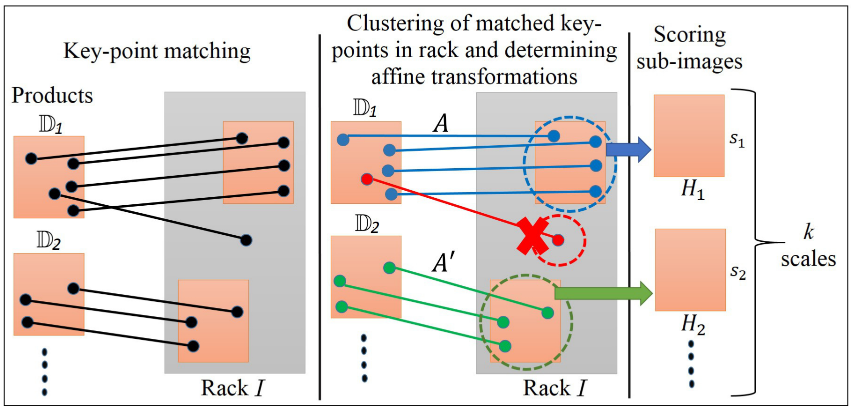


Figure 2.4: Scale estimation procedure: black dots and lines represent key-points and matched correspondences. Blue, red, and green circles indicate the clusters of matched key-points in rack. Correspondences of the clustered key-points in the product are also highlighted using the respective colors of clusters. s_1, s_2 are the scores of sub-images H_1, H_2 extracted from rack respectively.

m is the number of product classes. We refer each product template as $\mathbb{D}_t$, $t = 1, 2, \cdots, m$. A few examples of $\mathbb{D}_t$ are displayed in Figure 1.2(a). Let the physical dimensions of the products be available in some unit of length (say cm). Assume the given rack image is I which displays multiple products. Figure 1.2(b) illustrates three examples of such I. However, the physical dimension of the rack I is unknown. Given this setting, our aim is to localize the products $\mathbb{D}_t$ in rack I.

We first extract the Binary Robust Invariant Scalable Key-points (BRISK) [Leutenegger et al., 2011, Mukherjee et al., 2015] descriptors of all the template images of products and rack image. A BRISK descriptor finds key-points in the image and defines a 64-dimensional feature vector per key-point. In the feature vector, each feature value is a 8-bit number. Hence the feature vector is a 512-bit number. Assume we obtain ϖ and $\varkappa$ BRISK key-points in the tth product $\mathbb{D}_t$ and the rack I respectively. Let (x_t^i, y_t^i) and $\mathfrak{f}_t^i$, $i = 1, 2, \cdots, \varpi$ be the BRISK key-points and corresponding feature vectors of the tth product. Also let (x_I^j, y_I^j) and $\mathfrak{f}_I^j$, $j = 1, 2, \cdots, \varkappa$ be the BRISK key-points and corresponding feature vectors of the rack image I. Using these BRISK key-points, we generate a set of region proposals for the rack image I through two successive stages: scale estimation and region extraction that are explained in the following sections.

Stage-1: Scale Estimation

The scale between the products and rack plays an important role in extracting the potential regions. The computer vision practitioners in their previous attempts take care of the scale between the products and rack considering variable sizes of search windows [Merler et al., 2007, Franco et al., 2017, Ray et al., 2018] or grids [George and Floerkemeier, 2014, Marder et al., 2015, Zhang et al., 2007] on the rack. On the contrary, we estimate k possible scales between the m number of products $\mathbb{D}_t$ and rack image I using the physical dimensions of products.

The scale estimation process starts by calculating the geometric transformations between the images of products and rack. Each transformation is assigned a classification score. Using transformations with top-k scores, we estimate the k possible scales between products and shelf. The overall process of estimating scale is demonstrated in Figure 2.4. In the following paragraphs, we detail the entire process of estimation of scales.

Step 1: Matching of Key-points The BRISK key-points of the tth product with those of the rack I are first matched using the approach presented in [Lowe, 2004]. Note that the matching of the key-points refers to the matching of the feature vectors at those key-points. The procedure of matching the BRISK key-points is clearly explained through the following example.

Assume we have a feature vector $\mathfrak{f}_t$ (in the product $\mathbb{D}_t$) which we want to match with the two feature vectors $\mathfrak{f}_I^1$ and $\mathfrak{f}_I^2$ in the rack image I. Further assume, the *Hamming distance* between $\mathfrak{f}_t$ and $\mathfrak{f}_I^1$ is d_1 while the *Hamming distance* between $\mathfrak{f}_t$ and $\mathfrak{f}_I^2$ is d_2 and let $d_1 > d_2$. Given this, we aim to find out the correct match of $\mathfrak{f}_t$ in the rack I.

Since the BRISK feature vectors are 512-bit binary numbers, the Hamming distance between $\mathfrak{f}_t$ and any one of $\mathfrak{f}_I^1$ and $\mathfrak{f}_I^2$ could be maximum 512-bit binary number. However, we assume that a potential match between two feature vectors is valid only when the distance between the two feature vectors is much less compared to their maximum distance (which is 512 bits). The distance between two feature vectors eligible to be a match is taken as $M = matchThreshold \times 512$, where $matchThreshold \in [0,1]$.

Therefore, if both d_1 and d_2 are lower than M, then both $\mathfrak{f}_I^1$ and $\mathfrak{f}_I^2$ are eligible to be the potential matches of $\mathfrak{f}_t$. Next we move to another test (which is called *ratio test*) for finding out the correct match of $\mathfrak{f}_t$ in the rack I.

Since $d_1 > d_2$, $\mathfrak{f}_I^2$ is the better match of $\mathfrak{f}_t$ compared to the match of $\mathfrak{f}_I^1$ with $\mathfrak{f}_t$. If d_1 and d_2 values are close, this implies that $\mathfrak{f}_I^1$ and $\mathfrak{f}_I^2$ both are the potential matches of $\mathfrak{f}_t$. This results in an ambiguity. To address this, we allow only one match of $\mathfrak{f}_t$ in the rack I for accurately identifying the unique correspondences between the key-points of $\mathbb{D}_t$ and I.

We remove this ambiguity by rejecting both $\mathfrak{f}_I^1$ and $\mathfrak{f}_I^2$ as a match of $\mathfrak{f}_t$ if the ratio $\frac{d_2}{d_1}$ is greater than a threshold, $ratioThreshold \in [0,1]$. Else the better match $\mathfrak{f}_I^2$ is defined as the correct match for $\mathfrak{f}_t$ in I.

Step 2: Clustering of Matched Key-points in Rack The matching of BRISK key-points is performed to find out the probable locations of the tth product $\mathbb{D}_t$ in the rack I. But the matching of key-points is not sufficient to correctly determine the locations of the $\mathbb{D}_t$ in I when I contains multiple instances of the $\mathbb{D}_t$. In other words, the matched key-points of $\mathbb{D}_t$ in I can represent multiple instances of $\mathbb{D}_t$. Thus we cluster ζ number of key-points in I to locate the multiple instances of $\mathbb{D}_t$. More formally, the matched key-points (x_I^j, y_I^j), $j = 1, 2, \cdots, \zeta$ in I are clustered using the DBSCAN [Ester et al., 1996, Shen et al., 2016] clustering technique.

The DBSCAN clustering method is implemented with two parameters $minimumPoints$ and $maximumRadius$. We require at least $minimumPoints$ number of points to form a cluster. And for each point (x_I^j, y_I^j) in a cluster, there exists at least one point $(x_I^{j'}, y_I^{j'})$ in the cluster such that $\|(x_I^{j'}, y_I^{j'}) - (x_I^j, y_I^j)\| \leq maximumRadius$. This stage of our region proposal scheme aims to determine an affine transformation A of the tth product $\mathbb{D}_t$ with its potential match in the rack I. We set $minimumPoints = 3$ as we require exactly three point correspondences to determine an affine transformation. Note that, the number of matches of the product $\mathbb{D}_t$ in I is the number of clusters determined by the DBSCAN algorithm. In the second block of Figure 2.4, the clusters defined by blue and green circles are selected for the next step of our algorithm as these clusters have exactly three or more than three point correspondences. The clusters denoted by red circles (containing less than three points) are discarded.

Step 3: Determining Affine Transformations of Products Assume we obtain ρ_t clusters of matched key-points (x_I^j, y_I^j) in the rack I for the tth product $\mathbb{D}_t$. In Figure 2.4, $\rho_t = 2$. Let $(x_I^{\mathfrak{n}}, y_I^{\mathfrak{n}})$, and $(x_t^{\mathfrak{n}}, y_t^{\mathfrak{n}})$, $\mathfrak{n} = 1, 2, \cdots, \varphi$ be the key-points in the cth cluster of I and corresponding matched key-points in the tth product $\mathbb{D}_t$. Let $X_I = [x_I^{\mathfrak{n}}\ y_I^{\mathfrak{n}}\ 1]^T$ and $X_t = [x_t^{\mathfrak{n}}\ y_t^{\mathfrak{n}}\ 1]^T$ be the representations of the key-points. Given this, we aim to determine the affine transformation matrix A with six unknowns between $\mathbb{D}_t$ and cth cluster of I such that

$$X_I \equiv AX_t. \tag{2.1}$$

While $minimumPoints$ is set to 3 during clustering of key-points, we can have more than three correspondences of key-points between the tth product and cth cluster in I i.e., $\varphi \geq 3$. In that case (2.1) is solved as an over-determined problem, minimizing the least squared sum $\mathfrak{S}(A)$ using Levenberg-Marquardt algorithm [Levenberg, 1944, Marquardt, 1963], where

$$\mathfrak{S}(A) \equiv \text{argmin}_A \sum_{\mathfrak{n}=1}^{\varphi} \|X_I^{\mathfrak{n}} - AX_t^{\mathfrak{n}}\|_2^2. \tag{2.2}$$

Step 4: Extracting Sub-images from Rack Once the affine matrix A is obtained, we transform the four corner points $(0,0), (w,0), (0,h)$, and (w,h) of the tth product template $\mathbb{D}_t$ on I, where h and w are the width and height (in pixels) of $\mathbb{D}_t$. Using these four transformed points, we fit the largest possible rectangular bounding box and crop the region covered by the bounding box from I. Let the cropped sub-image be H. If the sub-image H matches (with high classification score) with the product $\mathbb{D}_t$, the transformation is acceptable. Thus the sub-image H determines the correctness of the transformation A. Examples of this sub-image extraction are illustrated in the third block of Figure 2.4 (see H_1 and H_2).

Step 5: Scoring the Sub-images As the sub-image H is obtained from $\mathbb{D}_t$, the class label of H should be $\mathbb{D}_t$. In order to match labels of H with $\mathbb{D}_t$, we obtain the class label and classification score of H using our classification module. The pre-trained CNN model [He et al., 2016] fine-tuned for product classification is detailed in Section 2.3. H is fed into the fine-tuned pre-trained CNN classifier to obtain the class label and classification score of H. If the label of H matches with $\mathbb{D}_t$, then H is considered as a valid sub-image and sent for further processing in estimating scale between the products and rack.

In this way, we perform the extraction and scoring of sub-images for all the clusters corresponding to all the products $\mathbb{D}_t$. Finally, we get a set of τ valid sub-images $\mathbb{H} = \{H_g\}$ with the set $\mathbb{L} = \{l_g\}$ of corresponding class labels, and the set $\mathbb{S} = \{s_g\}$ of corresponding classification scores, where $g = 1, 2, 3, \cdots, \tau$. The Figure 2.4 shows two valid sub-images H_1 and H_2 (along with the classification scores s_1 and s_2) extracted from the rack for the products $\mathbb{D}_1$ and $\mathbb{D}_2$ respectively.

Step 6: Estimating k Possible Scales A number of sub-images $H_1, H_2, H_3, \ldots$ with respective classification scores are obtained after Step 5. Top k number of sub-images H_g sorted based on descending classification scores are selected to define k different scales between sub-images H_g and corresponding products $\mathbb{D}_t$.

Let $\mathbb{S}' \subseteq \mathbb{S}$ containing top-k scores. Hence we obtain the sets $\mathbb{H}' \subseteq \mathbb{H}$ and $\mathbb{L}' \subseteq \mathbb{L}$ corresponding to $\mathbb{S}'$. For each $H \in \mathbb{H}'$, we obtain the label l of H from $\mathbb{L}'$ and determine the cm-to-pixel ratio of width (or x-scale) and height (or y-scale) using the physical dimensions of lth labeled product $\mathbb{D}_l$ and pixel dimensions of the sub-image H of I. Consequently, we get k number of cm-to-pixel x-scales, sc_x^u and y-scales, sc_y^u, $u = 1, 2, \cdots, k$.

Stage-2: Region Extraction

Figure 2.5 presents a few examples of transformed corner points of the products in a rack using the affine matrices as described in the Stage-1 of our scheme (refer Section 2.2.1). We can see the incorrect

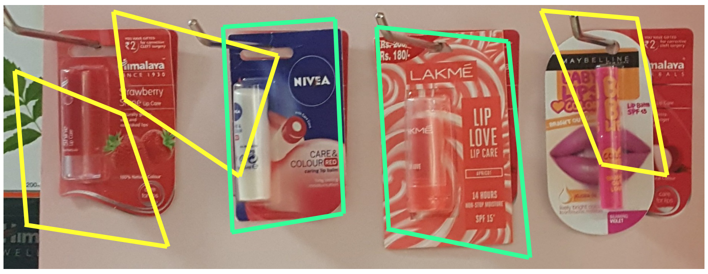


Figure 2.5: Examples of transformed corner points (highlighted with the quadrilaterals) of products in rack using estimated affine transformations. The tight fitted rectangle, which covers a quadrilateral, is cropped and classified. Green and yellow rectangles provide correct and incorrect classifications respectively.

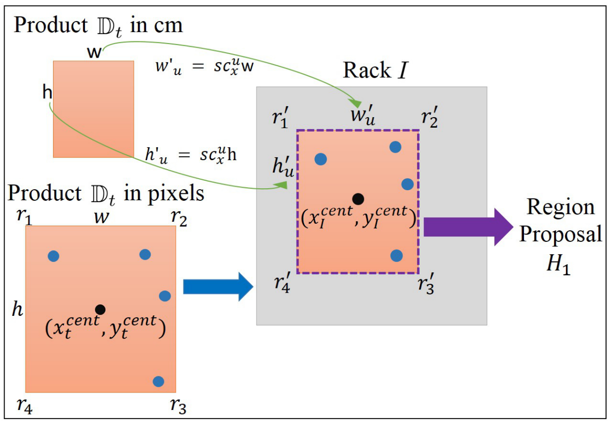


Figure 2.6: The process of generating region proposals. Blue dots represent the key-points. Black dots show the centroid of the key-points.

transformations of corner points (which produce yellow quadrilaterals in the figure) of the products in rack. These incorrect transformations extract many sub-images which either do not cover any product or display a tilted/skewed image of the true product. Furthermore, the affine transformations are not calculated in Stage-1 when there exists only one or two matches of the key-points between the products and the rack image. This necessitates introduction of the Stage-2 of our region proposal algorithm. The Stage-2 of the proposed region proposal scheme exhaustively searches for the potential regions around the product displayed on the rack.

Similar to Stage-1, key-points of the products and rack are matched followed by clustering of key-points in the rack as shown in 1st and 2nd block of Figure 2.4. But the $minimumPoints$ parameter of DBSCAN clustering algorithm is set to 1. In the second stage, in extracting potential regions from the rack, we do not want to lose a single proposal even if that is due to the match of single key-point between the products and the rack.

Let us assume that we obtain ρ_t clusters of the matched key-points in the rack image I. In Figure 2.6, we show only one cluster (say, cth cluster) for the tth product $\mathbb{D}_t$. As shown in Figure 2.6, let $(x_I^{\mathfrak{n}}, y_I^{\mathfrak{n}})$ and $(x_t^{\mathfrak{n}}, y_t^{\mathfrak{n}})$, $\mathfrak{n} = 1, 2, 3, 4$ be the matched key-points in the cth cluster of the rack I and the tth product $\mathbb{D}_t$ respectively. For the cth cluster, we now extract potential regions from I using the geometric alignment of the matched key-points and k estimated cm-to-pixel scales. In the procedure for extracting such potential regions, we first calculate the centroids (x_I^{cent}, y_I^{cent}) and (x_t^{cent}, y_t^{cent}) of the matched key-points $(x_I^{\mathfrak{n}}, y_I^{\mathfrak{n}})$ in the cth cluster of I and $(x_t^{\mathfrak{n}}, y_t^{\mathfrak{n}})$ in the tth product (see the black dots in Figure 2.6).

In Section 2.2.1, we have already derived k number of cm-to-pixel x-scales sc_x^u and y-scales sc_y^u between the products and rack, $u = 1, 2, \cdots, k$. However, rest of the process is described for only uth scale considering $k = 1$. Let h and w be the height and width in pixels of $\mathbb{D}_t$ while h_p and w_p be the height and width in cm of $\mathbb{D}_t$. For uth cm-to-pixel scale sc_x^u and sc_y^u, the transformed width and height

Algorithm 1 Exemplar-driven Region Proposal

```
    Input: The rack image $I$
    Output: The region proposals $H$
 1: for each product $\mathbb{D}_t$ in the database $\mathbb{D}$ do
 2:     Find the BRISK key-points of $\mathbb{D}_t$
 3: end for
 4: Find the BRISK key-points of the rack image $I$
                                        ▷ STAGE-1: SCALE ESTIMATION
 5: for each $\mathbb{D}_t$ in $\mathbb{D}$ do
 6:     Match the key-points of $\mathbb{D}_t$ with that of $I$
 7:     Find the clusters of matched key-points in $I$
 8: end for
 9: for each cluster do
10:     Calculate affine matrix $A$ between the cluster of matched key-points of $\mathbb{D}_t$ and $I$ using (2.1)
11:     Extract a sub-image $H$ of $I$ applying $A$ on $\mathbb{D}_t$
12:     Find the class label and classification score of $H$ from CNN
13: end for
14: Find top-$k$ classification scores and corresponding labels from all $H$
15: Find $k$ cm-to-pixel scales between the products and rack using physical dimensions of the products
    in the database $\mathbb{D}$
                                        ▷ STAGE-2: REGION EXTRACTION
16: for each $\mathbb{D}_t$ in $\mathbb{D}$ do
17:     Match the key-points of $\mathbb{D}_t$ with that of $I$
18:     Find the clusters of matched key-points in $I$
19: end for
20: for each cluster do
21:     for each cm-to-pixel scale do
22:         Extract potential region $H$ from $I$ using (2.3)
23:     end for
24: end for
```

of $\mathbb{D}_t$ in I are $w'_u = sc^u_x \, \mathrm{w}_p$ pixels and $h'_u = sc^u_y \, \mathrm{h}_p$ pixels respectively (see Figure 2.6). In case physical dimensions, that is, h_p and w_p are not available, following assumption (ii) described in third paragraph of Section 2.1, we set $\mathrm{h}_p = h$ and $\mathrm{w}_p = w$.

As shown in Figure 2.6, let r_1, r_2, r_3 and r_4 be the four corner points of the tth product $\mathbb{D}_t$. So, $r_1 = (0,0), r_2 = (w,0), r_3 = (w,h)$, and $r_4 = (0,h)$. Therefore the centroid (x^{cent}_t, y^{cent}_t) must lie within the rectangle formed by these four corner points. Let the four corner points be transformed to r'_1, r'_2, r'_3 and r'_4 in the rack I for the cth cluster (see Figure 2.6). Then for the uth scale, the co-ordinates of the transformed points in I are determined as follows:

$$
\begin{aligned}
r'_1 &= (x^{cent}_I - \frac{w'_u}{w}\, x^{cent}_t, y^{cent}_I - \frac{h'_u}{h}\, y^{cent}_t), \\
r'_2 &= (x^{cent}_I - \frac{w'_u}{w}\, x^{cent}_t + w'_u, y^{cent}_I - \frac{h'_u}{h}\, y^{cent}_t),
\end{aligned} \tag{2.3}
$$

$$r'_3 = (x_I^{cent} - \frac{w'_u}{w} x_t^{cent} + w'_u, y_I^{cent} - \frac{h'_u}{h} y_t^{cent} + h'_u),$$
$$r'_4 = (x_I^{cent} - \frac{w'_u}{w} x_t^{cent}, y_I^{cent} - \frac{h'_u}{h} y_t^{cent} + h'_u).$$

Let H_1 be the rectangular region of I covered by these four points r'_1, r'_2, r'_3 and r'_4 (refer Figure 2.6). This H_1 is essentially a region proposal. The process is repeated for all k scales, and ρ_t clusters for the tth product. The number of clusters varies from one product to another. Thus, for the m number of products, the total number of clusters can be determined as $\mathfrak{c} = \sum_{t=1}^{m} \rho_t$. By iterating the process for m products, our exemplar-driven region proposal scheme generates $\chi = mk\mathfrak{c}$ number of region proposals H_z, $z = 1, 2, \cdots, \chi$. The Algorithm 1 summarizes the proposed region proposal scheme. Next we discuss the classification and non-maximal suppression of the region proposals.

2.2.2 Classification and Non-maximal Suppression

Figure 2.7 demonstrates the initial set of overlapping region proposals obtained from the previous step. These initial proposals with their classification scores and labels (obtained from the pre-trained CNN [He et al., 2016] classifier which is adapted for our product classification task as described in Section 2.3) are input to greedy non-maximal suppression (greedy-NMS) [Felzenszwalb et al., 2010] scheme. Consequently, the greedy-NMS provides the final output with detected products as shown in Figure 2.7. Note that we interchangeably use the words proposal, region and detection. The goal of greedy-NMS is to retain at most one detection per group of overlapping detection. The *intersection-over-union* (IoU) [Girshick et al., 2014] between two regions defines the overlap amount (between those two regions). The overlapping group of any proposal H_z is a set of proposals which overlap with H_z by an amount more than a preset IoU value $IoUthresh$. Example group of overlapping detection can be seen in Figure 2.7. The greedy-NMS retains at most one detection (having highest classification score) per overlapping

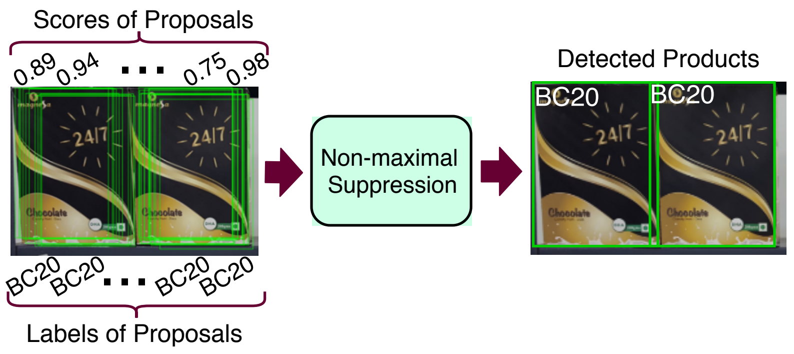


Figure 2.7: Flow chart of the non-maximal suppression scheme. BC20 and $0.89, 0.94, \cdots, 0.98$ are the class labels and classification scores of the proposals respectively obtained from the pre-trained CNN [He et al., 2016] adapted to our product classification task.

group of regions. In other words, greedy-NMS eliminates all regions having lower classification scores and overlapping with a region with the higher score. Next section presents various experiments and their results.

2.3 Experiments

The proposed solution is implemented in python and tested in a computing system with the following specifications: 64 GB RAM, Intel Core i7-7700K CPU @ 4.2GHz$\times$8 and 12GB TITAN XP GPU. The following section presents the experimental settings and implementation details of the methods.

2.3.1 Experimental Settings

Exemplar-driven Region Proposal In the stage-1 of our algorithm (see Section 2.2.1), we choose $ratioThreshold = 0.73$ and $matchThreshold = 0.45$ for matching the key-points between the products and rack through the following experiment. Initially, we assume that the two key-points can be considered as a match if the difference between their feature vectors is lower than the 50% of maximum possible distance i.e., lower than $matchThreshold \times 512$, where $matchThreshold = 0.5$. On the other hand, $ratioThreshold$ is initially set to 0.8 following [20]. With this initialization of the parameters and extensive experimentation with 180 product templates, we obtain the best result for $matchThreshold = 0.45$ and $ratioThreshold = 0.73$.

As mentioned in Section 2.2.1, $minimumPoints$ of the DBSCAN algorithm is always 3 and $maximumRadius$ of the same is empirically set to 60 pixels for clustering the matched key-points in the rack. We consider one estimated scale ($k = 1$) for generating region proposals. In the stage-2 of our algorithm (see Section 2.2.1), the values of $ratioThreshold$ and $matchThreshold$ parameters are same as these are in the stage-1. For clustering the key-points in rack, $minimumPoints = 1$ (as mentioned in Section 2.2.1) to obtain the exhaustive set of proposals and $maximumRadius = 60$ pixels to execute DBSCAN.

Classification of Regions Here we successively discuss the standardization of data, data augmentation, and domain adaptation of CNN for classification of regions.

Data Standardization By design [Paszke et al., 2019], the pytorch implementation of pre-trained ResNet-101 CNN model requires input images of size $224 \times 224 \times 3$ pixels. We transform the product image into a fixed-size of 224×224 color images without altering the aspect ratio (see Figure 2.8).

A product image of $w \times h \times 3$ pixels is first resized to $224 \times 224\frac{h}{w} \times 3$ pixels if $w > h$, else resized to $224\frac{w}{h} \times 224 \times 3$. The resized product image is then superimposed in a white frame of $224 \times 224 \times 3$ pixels such a way so that the smaller side of the image is aligned in the middle of the frame

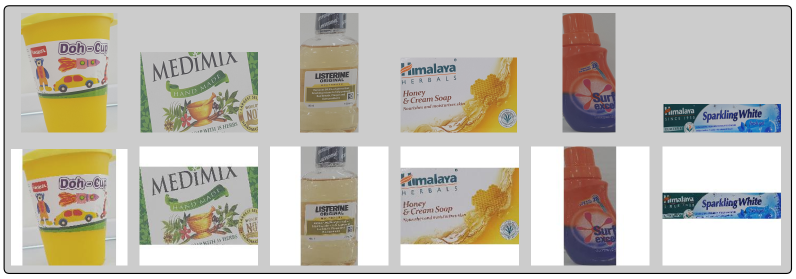


Figure 2.8: Original (top-row) vs. transformed (bottom-row) product images

while other side perfectly fits the frame. Figure 2.8 demonstrates some examples of product images and corresponding transformed product images. Consequently, the transformed product image is standardized by normalizing each colour channel R, G and B of the image. Assume C denotes any color channel R, G, or B. Let C' be the normalized version of C such that $C' = \frac{C-\mu J}{\sigma}$, where μ and σ are the mean and standard deviation of all pixel intensities of C over all training samples, and J is the 224×224 *all-ones* matrix.

Data Augmentation The detection of products is a problem where only one template image is available per product class. But to train a learning-based scheme like CNN, we require a huge amount of training samples per class. Thus using various photometric and geometric transformations, we generate $\sim 10^4$ training samples from the single template of a product. The training samples are augmented using three python libraries: keras [1], augmentor [2], and imgaug[3]. Considering supermarket like scenario, the synthesis process applies the following photometric transformations: random contrast adjustment, random brightness adjustment, noise (salt & pepper and Gaussian) addition, and blurring (Gaussian, mean and median). Consequently the geometric transformations such as rotation, translation, shearing, and distortion are applied on the photometrically transformed synthesized images. These augmented training samples are used for training the CNN.

Domain Adaptation of CNN The classification of region proposal into any one of the product classes uses pytorch [Paszke et al., 2017] implementation of the ResNet-101 CNN model [He et al., 2016]. The final layer (i.e., last fc layer) of the ResNet-101 network has 1000 nodes for 1000-way classification of ImageNet [Deng et al., 2009] dataset. For our problem, let a product dataset has m product classes. In order to adapt the network to our task, the last fc layer of the ResNet-101 network is replaced by a newly introduced fc layer having m nodes. The weights of the new connections are initialized with random values drawn from $[-1, 1]$. Now the entire network is trained with our augmented training samples.

The training of the CNN is performed by calculating cross-entropy loss and optimizing the loss using mini-batch stochastic gradient descent (SGD) [Robbins and Monro, 1951, Ghassabeh and Moghaddam,

[1]https://github.com/keras-team/keras accessed as on 06/2021
[2]https://github.com/mdbloice/Augmentor accessed on 06/2021
[3]https://github.com/aleju/imgaug accessed as on 06/2021

Table 2.1: Non-trainable parameters and their values in our implementation of the proposed algorithm

Building Blocks of the Proposed Scheme	Parameters	Values
Scale Estimation (Stage-1 of Section 2.2.1)	$matchThreshold$	0.45
	$ratioThreshold$	0.73
	$minimumPoints$	3
	$maximumRadius$	60 pixels
	number of scales, k	1
Region Extraction (Stage-2 of Section 2.2.1)	$matchThreshold$	0.45
	$ratioThreshold$	0.73
	$minimumPoints$	1
	$maximumRadius$	60 pixels
Classifier (Section 2.2.2)	Initial Learning Rate	0.01
	Momentum	0.9
greedy-NMS (Section 2.2.2)	$scoreThresh$	0.5
	$IoUthresh$	0.07

2013]. We uniformly sample 2^5 augmented images to form a mini-batch in each iteration of SGD. The SGD is started at a learning rate 0.01 and with a momentum of 0.9. After each 10 epoch, we update the learning rate by a factor of 0.1. However, the parameters of the network have been trained for 200 epoch. The adapted ResNet-101 propagates (in forward direction) the transformed product images through its layers and the output of last fc layer is then passed through the *softmax* [Yu et al., 2018, Hu et al., 2019] function to obtain the class probabilities for any region proposal. This classification strategy in association with our exemplar-driven region proposal (ERP) scheme is referred to as **ERP+CNN**.

Non-maximal Suppression The greedy non-maximal suppression (greedy-NMS) are performed on the proposals with the classification score above the threshold $scoreThresh = 0.5$. In greedy-NMS, we empirically choose $IoUthresh = 0.07$ for detecting products. The values of the non-trainable parameters of the proposed algorithm are listed in Table 2.1. Next we present the results and comparisons.

2.3.2 Results and Analysis

In order to perform experimental analysis and a comparative study, we reproduce the competing methods in [Girshick et al., 2014, Ray et al., 2018, Marder et al., 2015, Merler et al., 2007, Franco et al., 2017, George and Floerkemeier, 2014, Zhang et al., 2007]. The methods in [Girshick et al., 2014, Ray et al., 2018, George and Floerkemeier, 2014] are referred to as R-CNN, U-PC and MLIC respectively. The authors of [Marder et al., 2015, Merler et al., 2007, Zhang et al., 2007, Franco et al., 2017] present more than one method. The histogram of oriented gradients (HOG) and bag of words (BoW) based schemes are implemented from [Marder et al., 2015] while color histogram matching (CHM) based scheme is reproduced from [Merler et al., 2007]. In case of [Zhang et al., 2007], we implement its best scheme S1.

Table 2.2: Results of various methods on In-house dataset

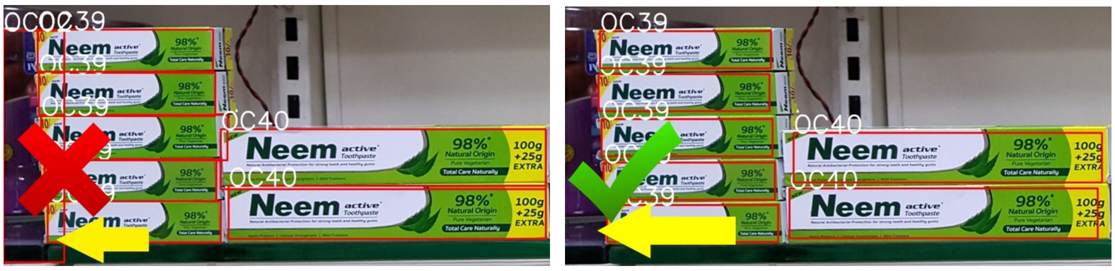
Methods	F_1 Score (%) on Categories of In-house Dataset					
	BC	DEO	LC	OC	PW	MIX
S1 [Zhang et al., 2007]	41.01	45.21	47.87	48.08	54.91	49.01
CHM [Merler et al., 2007]	48.04	33.56	60.12	30.77	36.40	44.74
MLIC [George and Floerkemeier, 2014]	64.45	50.08	54.91	40.23	59.97	48.76
R-CNN [Girshick et al., 2014]	82.04	83.76	87.99	79.72	88.05	73.16
HOG [Marder et al., 2015]	62.00	28.52	49.37	28.73	44.06	50.62
BoW [Marder et al., 2015]	65.05	45.10	70.72	53.31	71.23	59.91
GBoW [Franco et al., 2017]	72.07	49.98	68.29	46.79	77.22	53.41
GDNN [Franco et al., 2017]	82.12	55.49	82.55	51.32	87.64	61.98
SET [Karlinsky et al., 2017]	83.61	**83.95**	88.36	81.77	88.16	74.22
U-PC [Ray et al., 2018]	84.77	52.59	86.29	55.65	81.15	65.49
ERP+CNN	**90.86**	83.76	**92.49**	**89.80**	**92.12**	**82.98**

From [Franco et al., 2017], we implement both bag of words (BoW) and deep neural network (DNN) based methods which are referred to as GBoW and GDNN respectively. The method of [Karlinsky et al., 2017] is denoted by SET for which the publicly available code is downloaded from github (`https://github.com/leokarlin/msmo_star_model`). All the competing deep networks are trained under similar setup like ours as explained in the second last paragraph of the previous section. Note that R-CNN refers to the vanilla R-CNN method as explained in [Girshick et al., 2014].

In [Santra and Mukherjee, 2019], we notice that different evaluation indicators are used in different state-of-the-art methods for validating the solutions. Keeping context of retail products in mind, in this thesis, the (product detection) accuracy of various methods are evaluated by calculating F_1 score that we discuss in Appendix A.

The experiments are carried out on one In-house and three publicly available benchmark datasets: GroZi [Merler et al., 2007], WebMarket [Zhang et al., 2007], and Grocery Products [George and Floerkemeier, 2014]. The In-house dataset consists of six categories of products: *breakfast cereals* (BC), *deodorant* (DEO), *lip care* (LC), *oral care* (OC), *personal wash* (PW) and *mixed* (MIX). The detailed descriptions of these datasets are provided in Appendix B. In all these datasets, the products are captured in a controlled environment while the racks are imaged in the wild. So the product images differ from rack images in scale, pose and illumination. Note that the benchmark datasets do not provide physical dimensions of the products. So we use context information of retail store that similar products and the products of similar shapes are put together for shopper's convenience. Because of the context, we can assume that the physical dimensions of the products displayed in a rack are almost similar. Next we present the experimental results on these datasets.

Table 2.2 presents the average F_1 scores of various methods for all the racks of six categories of



(a) R-CNN [Girshick et al., 2014] (b) **ERP+CNN**

Figure 2.9: Example output of (a) exemplar-independent region proposal (in R-CNN) and (b) proposed exemplar-driven region proposal (in **ERP+CNN**) schemes. The red cross mark highlights the incorrect/false (see yellow arrow in (a)) detection by R-CNN while that false detection is removed by our **ERP+CNN** (see the green tick mark in (b) pointed with yellow arrow).

the dataset. Table 2.2 shows that our proposed ERP based scheme **ERP+CNN** yields ∼4% higher F_1 score than the best among other competing methods for five categories (BC, LC, OC, PW, MIX). An example of the qualitative results of exemplar-independent region proposal based scheme (in R-CNN) and our exemplar-driven region proposal based scheme (in **ERP+CNN**) are compared in Figure 2.9. In case of exemplar-independent R-CNN, as shown using yellow arrow in Figure 2.9(a), the false detection is marked by red cross. Whereas the green tick mark (pointed with yellow arrow) in Figure 2.9(a) highlights that the proposed exemplar-driven region proposal scheme in **ERP+CNN** does not identify that false detection from background regions as objects.

Repeatability Test We also perform the repeatability test of the proposed exemplar-driven region proposal scheme in **ERP+CNN** and exemplar-independent region proposal scheme in R-CNN on the six categories of the In-house dataset. For each category of the dataset, we collect images of a rack with variations in illumination, scale & pose and group them together. For each of such groups, we determine *standard error of the mean* $(SEM) = \frac{\sigma_{F1}}{\sqrt{\vartheta}}$ of the F_1 scores of rack images in the group, where σ_{F1} defines the standard deviation of F_1 scores of ϑ number of rack images. Consequently, we find out average *SEM* over all such groups for each category of the dataset. We have observed that lower the average *SEM*

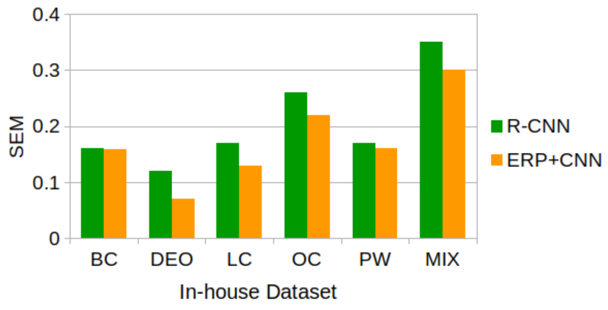


Figure 2.10: Repeatability test: the average *SEM* of exemplar-independent (R-CNN) and exemplar-driven region proposal (**ERP+CNN**) schemes on the In-house dataset

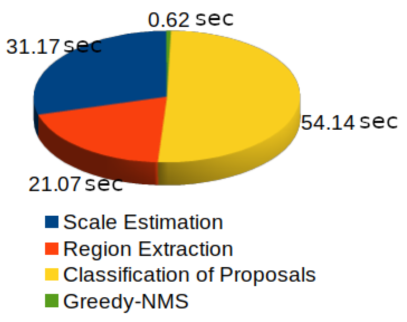
Table 2.3: Performances of various methods on benchmark datasets

Methods	F_1 Score (%) on Benchmark Datasets		
	Grocery Products	WebMarket	GroZi
S1 [Zhang et al., 2007]	58.39	49.19	31.71
CHM [Merler et al., 2007]	51.20	52.81	24.70
MLIC [George and Floerkemeier, 2014]	59.07	53.33	33.10
R-CNN [Girshick et al., 2014]	78.99	72.01	40.91
HOG [Marder et al., 2015]	58.11	43.03	28.33
BoW [Marder et al., 2015]	59.91	55.15	26.83
GBoW [Franco et al., 2017]	74.34	65.59	39.66
GDNN [Franco et al., 2017]	73.09	71.13	43.99
SET [Karlinsky et al., 2017]	79.05	72.13	43.78
U-PC [Ray et al., 2018]	76.20	67.79	40.10
ERP+CNN	**81.05**	**78.76**	**47.49**

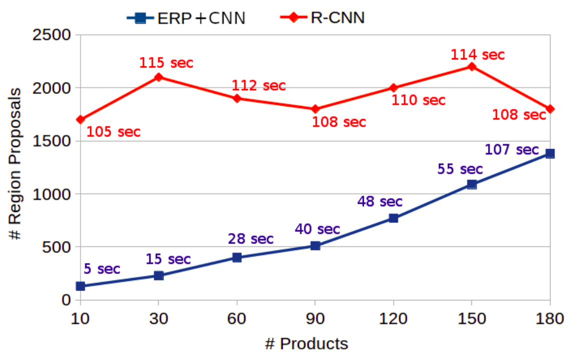
is, higher repeatable the method is. In Figure 2.10, the average *SEM* of the exemplar-independent and our exemplar-driven schemes are shown on six categories of In-house dataset. The figure infers that our exemplar-driven region proposal scheme is more repeatable than exemplar-independent region proposal scheme in detecting retail products.

We select 74 rack images (displaying 184 products), 36 rack images (displaying 155 products), and 28 rack images (displaying 20 products) respectively from Grocery Products, WebMarket, and GroZi datasets. These rack images comply with our assumptions on physical dimensions mentioned in the 3rd paragraph of Section 2.1. Second column of the Table 2.3 presents results of our proposed scheme and other competing methods on the Grocery Products dataset. The proposed **ERP+CNN** scheme outperforms R-CNN by $\sim$2% and yields better results than other competing methods. Third column of the Table 2.3 lists the results of various methods on the WebMarket dataset. The proposed **ERP+CNN** is found to be the winner by at least a margin of $\sim$6%. The performances of various methods on the GroZi dataset are inferior as compared in the fourth column of the Table 2.3. Like previous two benchmark datasets, the proposed scheme shows its superiority over other methods for this dataset. Since the template images are collected from the web, the products in rack mostly differ from the templates. This is the primary challenge of this dataset and due to this, the accuracy does not cross 50%.

Time Analysis We also present an analysis on execution time of the proposed scheme **ERP+CNN**. Figure 2.11(a) shows time taken by each module of our scheme for detecting products when the dataset contains 180 products. In that case, the entire process of detecting products takes about 107 seconds out of which scale estimation, region extraction, classification of region proposals and non-maximal suppression of regions consume $31.17, 21.07, 54.14$, and 0.62 secs, respectively. Thus the proposed ERP, which includes scale estimation and region extraction procedures, takes about $31.17 + 21.07 = 52.24$



(a)



(b)

Figure 2.11: (a) Execution time of different modules of the proposed **ERP+CNN** for processing one rack with 180 products in the dataset and (b) number of products vs. number of region proposals (with corresponding execution time for processing one rack) generated by R-CNN and the proposed **ERP+CNN**

sec. It can clearly be seen that the classification module of our scheme runs for a longer period of time than other modules. Our analysis finds that the data standardization process in the classification module is responsible for it.

However, the total execution time of our proposed scheme mainly depends on the number of region proposals. The number of proposals again depends on the number of products in a dataset. This can be clearly observed in Figure 2.11(b). For example, running time of the proposed scheme is 40 sec for processing 500 proposals in a rack populated with 90 products in the dataset. The same is 48 sec for handling 800 proposals generated with 120 products in the dataset. We have tested our scheme for at most 180 products and the corresponding execution times are plotted in Figure 2.11(b). We find that our exemplar-driven region proposal based scheme generates much lesser number of proposals and takes lesser time than exemplar-independent region proposal based scheme R-CNN. This can be clearly seen in Figure 2.11(b). Next section concludes the chapter.

2.4 Summary

In this chapter, we build a machine vision system for identifying the products (without using any annotations of the rack images) given single instance (or template image) per product. The proposed machine vision system implements a two-stage exemplar-driven region proposal scheme, where the first stage estimates the scale between products and rack, and the second stage generates region proposals (around the products in the rack) with the estimated scale.

In this chapter, it is evident that the proposed exemplar-driven region proposal scheme shows its superiority over exemplar-independent region proposal method, where the images of racks (i.e., test images) are crowded with products. Since our exemplar-driven scheme elegantly generates proposals around the products on the rack, the number of false region proposals are significantly reduced.

Since the physical dimensions of the products are available for In-house dataset, the scale between templates of the products and the image of rack is correctly estimated. For benchmark datasets, the context information of retail store is used for estimating the scale. The context information sometimes lead to inferior result compared to using physical dimension of product template.

Our analysis finds that the proposed system occasionally results in inaccurate identification of products due to incorrect classification of (very) similar but non-identical products. We refer these similar but non-identical products as fine-grained products. In the next chapter, we look into classification of fine-grained products for improving the product detection performance of our system that we have introduced in this chapter.

CHAPTER 3

Fine-grained Classification of Products

3.1 Introduction

As mentioned in Chapter 1, classifying (very) similar (but non-identical) variants of products is one of the most challenging tasks. We refer this task as fine-grained classification which is the focus of this chapter. The fine-grained variations are usually due to slight variations in text, size, or color of the package. A set of examples are shown in Figure 3.1. In Figure 3.1(a), only the sizes of the packets and the text printed on the packets vary for 805g to 500g of the same product, *Nutri Grain*. In Figure 3.1(b) and (c), the text and color vary due to change in the flavour of the same product, *Nivea Men* and *Nivea deodorant* respectively. Note that one product (say, 50g of potato chip) from two different manufacturers, should have two distinctly different packages. Hence, as per our definition, identification of distinctly different packages is an example of coarse-grained classification. The proposal in this chapter classifies such

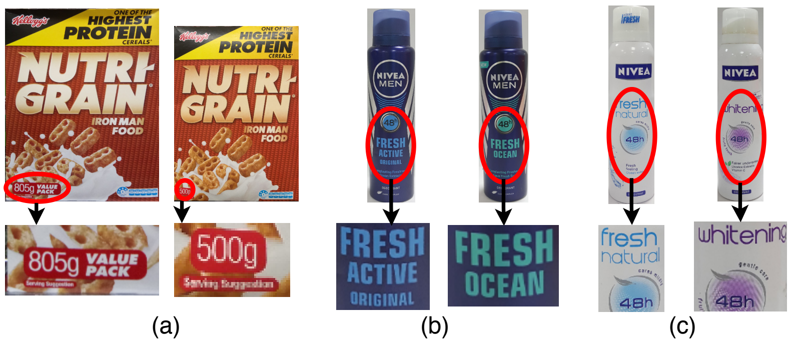


Figure 3.1: Top row presents examples of fine-grained products having minute differences (see red ellipses) in (a) size and text and, (b)-(c) color and text which are highlighted in bottom row.

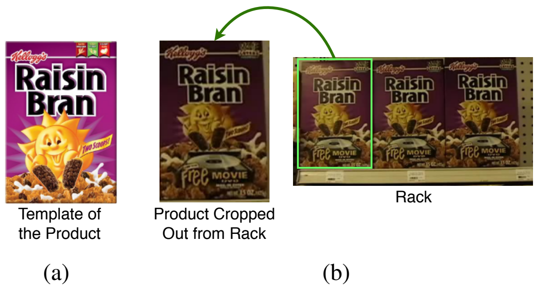


Figure 3.2: Differences in (a) training image (template of a product) and (b) test image (cropped product from rack using green bounding box) from GroZi-120 [Merler et al., 2007] dataset

coarse-grained products along with the fine-grained products. The product images shown in Figure 3.1 are used as templates and used for training. In contrast the quality of rack images, from where the product images are cropped and used as test images for our problem, is affected due to store level illuminations. An example of such differences between training and test images are shown in Figure 3.2. Both marginal variations in image content and illumination, make fine-grained classification of product images much more challenging compared to classical fine-grained object classification [Huang et al., 2016, Ge et al., 2019, Xiao et al., 2015, Peng et al., 2017, Flores et al., 2019, Xie et al., 2017].

This chapter presents a novel solution for the fine-grained classification of retail products utilizing object-level and part-level cues. The solution emulates human-like approach. First, features due to overall content of the product are captured followed by features for finer discriminating characteristics. For example, in Figure 3.1(a), first product can be differentiated from the second by looking at the weights 805g and 500g mentioned on the package. Next we present an overview of our solution.

3.1.1 An Overview

The proposal is a two-step process comprising object-level classification and part-level classification, the overviews of which are presented next.

Object-level Classification

In the first step of our proposed approach, we introduce a reconstruction-classification network (RC-Net) which is essentially a deep *supervised convolutional autoencoder* (SCAE) similar to the *supervised autoencoder* (SAE) [Le et al., 2018]. RC-Net is introduced to address the illumination differences between the train and test images (see Figure 3.2) for accurate classification of products. In other words, the RC-Net is built in a way such that it should have the better generalization ability to resolve such differences in appearance between the train and test images. Note that learning a model from a finite set of training images such that the model performs equally well on new images, possibly with wider variations in appearance, is usually referred to as generalization.

Background of RC-Net The generalization of neural net (NN) is often characterized through some form of regularization [Bousquet and Elisseeff, 2002, Zhang, 2002, Mohri et al., 2015]. Both theoretical [Liu et al., 2016b] and empirical [Maurer et al., 2016] studies show that the addition of auxiliary tasks into an NN is an efficient regularization technique to ensure generalization of the network.

In [Le et al., 2018], Le *et al.* present an auxiliary task model for NN by adding a secondary task, reconstruction of inputs (considered as a regularizer) along with the primary classification task. Their auxiliary model is called *supervised autoencoder* (SAE). In [Le et al., 2018], it is theoretically shown that linear SAE is uniformly stable. That means the difference between the performances of the models trained with two subsets of a dataset (where the second subset is obtained by randomly replacing one sample from any class of the first subset) is bounded in an interval. Subsequently, it is shown that linear SAE provides a bound for the generalization. As shown in [Le et al., 2018], the primary advantage of linear SAE is that it provides tighter bounds of generalization than that for learning with auxiliary tasks as in [Liu et al., 2016b, Maurer et al., 2016]. Tighter generalization bounds indicate that the generalization performance of a model varies less with different test samples.

SAE is a fully connected NN (or multi-layer perceptron) and SAE performs remarkably well for one-dimensional data. On the other hand, the convolutional version of fully connected NN (i.e., CNN) shows outstanding performance in classifying two-dimensional (or three-dimensional) data, particularly the images [Krizhevsky et al., 2012, Simonyan and Zisserman, 2015, He et al., 2016, Luan et al., 2018, Lyu et al., 2019]. Motivated by this fact, we present the convolutional version of SAE, which we call *supervised convolutional autoencoder* (SCAE), for improving the classification performance of retail products. The SCAE can handle wider variation of appearance including variation in illumination of test images as shown in our experiments in Section 3.4. Moreover, we show that linear SCAE without non-linear activation functions, provides a bound on generalization error like linear SAE.

Unlike linear SCAE, a generic formulation on the bound of generalization error for any non-linear SCAE is not straightforward, as the formulation entirely depends on the type of activation functions used in the different fully connected (fc) or convolution (conv) layers of the network. Since we often utilize *rectified linear unit* ($ReLU$) activation function in CNN (in fact, our RC-Net also uses $ReLU$), we theoretically show that a bound on generalization error can also be derived for a non-linear SCAE with $ReLU$ activation function. This somehow gives us an idea on the enhanced generalization ability of our RC-Net, that essentially helps in product recognition under varying store-level illuminations.

Part-level Classification

The second step of our scheme essentially boosts the classification performance of the first step. The discriminatory parts of the products are searched (in an unsupervised manner) and organized as an ordered sequence to uniquely describe the products. These sequences of parts are modeled using convolutional

LSTM (conv-LSTM) [Xingjian et al., 2015]. Note that part-level annotations are not used keeping in mind implementation challenges of the system for fast changing product lines [Santra and Mukherjee, 2019]. Finally, classification scores from both the steps (object and part-level classification), jointly decide labels of the products in the test images.

We have considered both fine-grained and coarse-grained products in this work. Our object-level classifier i.e., RC-Net, efficiently classifies the coarse-grained products, and on top of that, our part-level classifier differentiates the fine-grained products. We intentionally lay stress on the classification of fine-grained products in this chapter, assuming coarse-grained classification is an obvious task to do.

The rest of the chapter is organized as follows. Related works and contributions are presented in Section 3.2. Section 3.3 explains the proposed fine-grained classification scheme. Experiments and results are provided in Section 3.4 followed by the summary in Section 3.5.

3.2 Related Work and Contributions

Generic Fine-grained Classification The state-of-the-art methods for fine-grained classification of objects can be classified into three groups (a) part based models [Huang et al., 2016, Ge et al., 2019], (b) two-level attention models [Xiao et al., 2015, Peng et al., 2017], and (c) CNN with second-order pooling [Lin et al., 2015, Lin and Maji, 2017]. The part based models first determine the key parts of the objects. These key parts are then used to design a CNN (like part-stacked CNN [Huang et al., 2016]) or LSTM (like complementary parts model [Ge et al., 2019]) for classification. Two-level attention models jointly learn object and part information to solve the problem under discussion [Xiao et al., 2015, Peng et al., 2017]. The convolution layers at the bottom of a deep CNN capture the finer descriptions of an object [Huang et al., 2016]. Moreover, in the CNN, second order pooling (like bilinear pooling [Lin et al., 2015, Lin and Maji, 2017]) models local pairwise feature interactions in a translation invariant manner. Due to this, bilinear pooling is integrated in our part-level modeling of the products. Compared to deep CNNs, our proposed RC-Net performs better due to improvement in generalizability. Moreover, our novel part-level module improves the overall fine-grained classification performance.

SegNet [Badrinarayanan et al., 2017] is a convolutional encoder-decoder architecture designed for segmentation of images. Convolutional autoencoder (CAE) differs from SegNet as CAE reconstructs the input while SegNet produces segmentation mask in its vanilla implementation [Badrinarayanan et al., 2017]. As mentioned earlier in Section 3.1, the proposed RC-Net is a supervised CAE (SCAE). The primary difference between SegNet and RC-Net is that the RC-Net not only reconstructs the input but also determines the classification score for the input image. Therefore, unlike SegNet, RC-Net simultaneously performs the reconstruction and classification of input product images. Due to this, our proposed network is referred to as reconstruction-classification network or RC-Net.

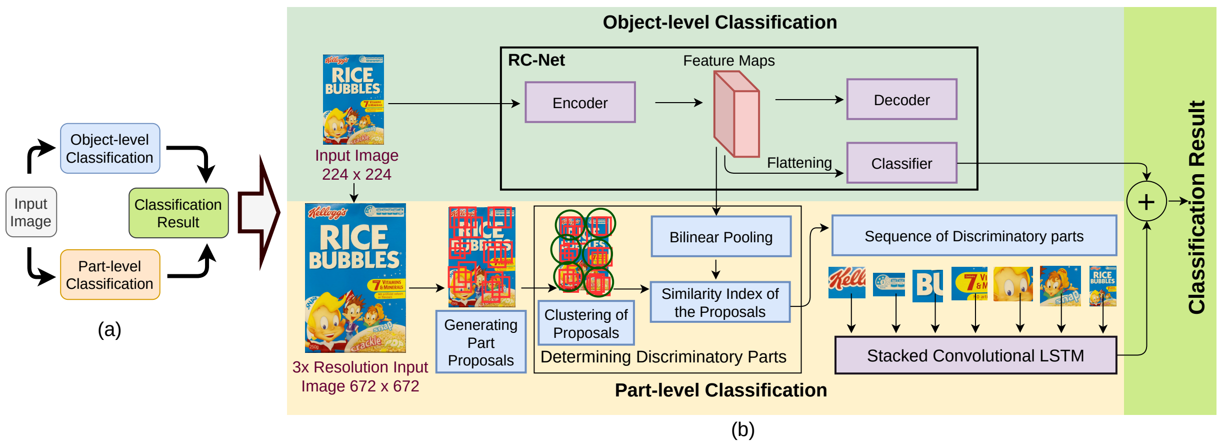

Input Image
Object-level Classification
Part-level Classification
Classification Result
(a)
Object-level Classification
RC-Net
Feature Maps
Encoder
Decoder
Flattening
Classifier
Input Image 224 x 224
Bilinear Pooling
Similarity Index of the Proposals
Sequence of Discriminatory parts
3x Resolution Input Image 672 x 672
Generating Part Proposals
Clustering of Proposals
Determining Discriminatory Parts
Stacked Convolutional LSTM
Part-level Classification
Classification Result
(b)

Figure 3.3: (a) Flowchart and (b) corresponding block diagram (highlighting intermediate steps) of the proposed scheme. For part-level classification, example input image is zoomed three times of its original size (for details, see Section 3.3.2).

Deep Reconstruction-classification Network Ghifary *et al.* [Ghifary et al., 2016] present a deep reconstruction-classification network (DRCN) for unsupervised domain adaptation. The architecture of the proposed RC-Net (24 convolutional layers and 2 fully connected layers) is much more deeper than DRCN (6 convolutional layers and 3 fully connected layers) for capturing the fine-grained representation of the products. The layers in encoder-decoder module of DRCN include both convolutional and fully connected layers. On the contrary, all the layers in encoder-decoder module of our proposed network are convolutional like SegNet [Badrinarayanan et al., 2017]. This improves the reconstruction power of the proposed network by preserving the local relationship between the neighboring pixels in the image. This is evident in the results shown in Section 3.4.1.

Contributions The contributions of this work are six-folds.

(a) A deep SCAE i.e., the RC-Net for fine-grained classification of products is introduced. The encoder-decoder architecture of RC-Net (or SCAE) significantly improves the classification performance under varying illumination due to enhanced generalization ability of SCAE.

(b) A theoretical bound of generalization is derived for linear SCAE and non-linear SCAE with *ReLU* activation function.

(c) For mathematical formulation of generalization error for SCAE, a uniform matrix multiplication setup is introduced for both fc and conv layers.

(d) In order to boost the performance of fine-grained classification, we introduce a conv-LSTM [Xingjian et al., 2015] based part-level classification model to encode discriminatory parts (identified by a unique unsupervised technique) of the products without using annotation of sub-parts of the products.

(e) Overall, we propose a unique fine-grained classification scheme where proposed RC-Net, bilinear pooling [Lin et al., 2015, Lin and Maji, 2017] and Conv-LSTM are utilized in tandem for weighted object-level and part-level classification.

(f) In order to improve the performance of our machine vision system (proposed in Chapter 2) for detecting retail products, the proposed fine-grained classifier is embedded in the system without using annotation of racks on which the products are being displayed. This will be discussed in Section 3.4.2.

Next we present the proposed classification strategy.

3.3 Method M2: Classification of Fine-grained Products

As mentioned in Section 3.1, the proposed scheme first extracts the coarse representations (or object-level features) of the products followed by detection of fine-grained representations (or local key features) of the products. The schematic of the proposed scheme is shown in Figure 3.3. Next we explain the above-mentioned modules of the proposed scheme.

3.3.1 Object-level Classification

The object-level classification of the products are performed using RC-Net that we explain next.

RC-Net

It consists of three modules: encoder, decoder and classifier as shown in Figure 3.4. The encoder-decoder architecture of RC-Net is built following SegNet [Badrinarayanan et al., 2017] for reconstruction of the product images. The classifier of RC-Net is a fully-connected network, which accepts the output of the encoder for classification.

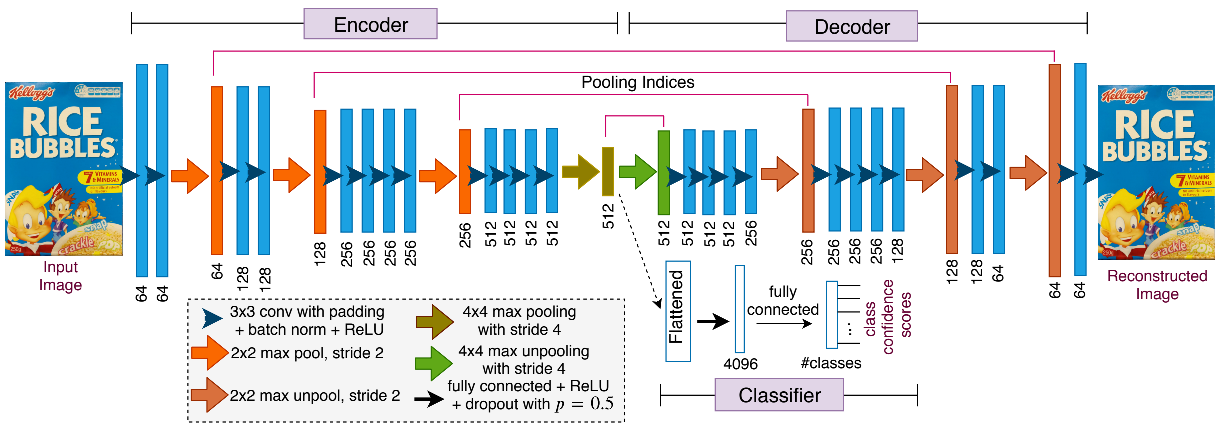


Figure 3.4: Architecture of the proposed RC-Net. Colored bars denote the feature maps. The number of feature maps is given just below the bar.

Table 3.1: Various layers of RC-Net are shown in sequence. conv3-64 represents 3×3 padded convolution operation performed 64 times, max-pool2 denotes 2×2 max pooling (with stride 2), max-unpool2 indicates 2×2 max unpooling (with stride 2), and fc-4096 represents fully connected layer with 4096 nodes. Similar interpretation is applicable for rest of the notations. After each conv layer, there exists batch normalization and ReLU layers. The classifier includes batch normalization, ReLU and deterministic dropout [Santra et al., 2020a] layers after fc-4096. m is the number of classes.

Encoder$\rightarrow$	conv3-64 conv3-64	max- pool2	conv3-128 conv3-128	max- pool2	conv3-256 conv3-256 conv3-256 conv3-256	max- pool2	conv3-512 conv3-512 conv3-512 conv3-512	max- pool4
Decoder$\rightarrow$	max- unpool4	conv3-512 conv3-512 conv3-512 conv3-512	max- unpool2	conv3-256 conv3-256 conv3-256 conv3-256	max- unpool2	conv3-128 conv3-128	max- unpool2	conv3-64 conv3-64
Classifier$\rightarrow$	fc-4096	fc-m						

Let P be an image of a product labelled with the *one-hot* vector $\mathbf{y} = (y_1, y_2, \cdots, y_m)$ such that, each $y_i \in \{0,1\}$ and $\sum_{i=1}^{m} y_i = 1$, where m is the number of distinct products (or classes). Assume, $P^{(1)}, P^{(2)}, \cdots, P^{(n)}$ be the n number of training data with the true label vectors $\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \cdots, \mathbf{y}^{(n)}$ for which RC-Net yields the reconstructed images $P'^{(1)}, P'^{(2)}, \cdots, P'^{(n)}$ and predicted label vectors $\mathbf{y}'^{(1)}, \mathbf{y}'^{(2)}, \cdots, \mathbf{y}'^{(n)}$ respectively.

Let θ and θ' denote all the trainable parameters (i.e., weights and biases) of the encoder and decoder respectively, and θ'' represents the trainable parameters of the classifier of proposed RC-Net. Given this, our objective is to optimize the parameters θ, θ' and θ'' for minimizing the *average reconstruction loss* and the *average classification loss* simultaneously,

$$\theta^*, \theta'^*, \theta''^* = \underset{\theta,\theta',\theta''}{\arg\min} \frac{1}{2n} \sum_{j=1}^{n} \left[\mathscr{L}_{rl}\left(P^{(j)}, P'^{(j)}; \theta, \theta'\right) + \mathscr{L}_{cl}\left(\mathbf{y}^{(j)}, \mathbf{y}'^{(j)}; \theta''\right) \right], \tag{3.1}$$

where $\mathscr{L}_{rl}(\cdot,\cdot;\theta,\theta')$ and $\mathscr{L}_{cl}(\cdot,\cdot;\theta'')$ are the *reconstruction loss* and *classification loss* respectively. In the proposed RC-Net architecture, *reconstruction loss* $\mathscr{L}_{rl}(\cdot,\cdot;\theta,\theta')$ is the conventional *binary cross-entropy loss* [Goodfellow et al., 2016, Pawara et al., 2020] and the *classification loss* $\mathscr{L}_{cl}(\cdot,\cdot;\theta'')$ is the *cross-entropy loss* [Goodfellow et al., 2016, Pawara et al., 2020].

The complete architecture of the proposed RC-Net is shown in Figure 3.4. The configurations of various layers of the proposed RC-Net are further tabulated in Table 3.1. However, initialization of the weights is important for any deep neural network with multiple numbers of layers, operations and paths. The weights of our network are strategically initialized with the weights of a pre-trained network (which is trained using the visual information of more than 14 million training images from 1000 classes of ImageNet dataset [Deng et al., 2009]) as follows.

The sequence of the convolution layers of encoder (see Figure 3.4) is identical with the sequence of first twelve convolution layers of the VGG-19 network [Simonyan and Zisserman, 2015]. So, the weights of the convolution layers of the encoder are initialized with the weights of first twelve convolution layers of the pre-trained pytorch [Paszke et al., 2017] implementation of VGG-19. In case of decoder, the initial weights of eight convolution layers, which exactly adhere with that of VGG-19, are initialized with the weights of pre-trained VGG-19. The adherence between the convolution layers of RC-Net and VGG-19 are $1 \leftarrow 12$, $2 \leftarrow 11$, $3 \leftarrow 10$, $5 \leftarrow 8$, $6 \leftarrow 7$, $7 \leftarrow 6$, $9 \leftarrow 4$, and $11 \leftarrow 2$, where $\ell \leftarrow \ell'$ represents that ℓth convolution layer of RC-Net is identical with ℓ'th convolution layer of VGG-19. The initial weights of the remaining layers of decoder are set following [Ronneberger et al., 2015].

The ability of simultaneous reconstruction and classification of products makes the RC-Net different from benchmark CNNs like VGG [Simonyan and Zisserman, 2015]. RC-Net is a combination of convolutional autoencoder and CNN classifier. The convolutional autoencoder tries to optimize the *reconstruction loss* while the classifier wants to optimize the *classification loss*.

In RC-Net, the addition of *classification loss* (or supervised loss) to the *reconstruction loss* (or unsupervised loss) makes the convolutional autoencoder capable of learning the underlying pattern of an object with class information of the object. Conversely, the addition of *reconstruction loss* to the *classification loss* forces the classifier to learn class discriminatory information along with the underlying pattern of an object. This way, the reconstruction and classification joint loss enforces RC-Net to balance both extraction of underlying structure and inference of correct prediction of an object. In other words, *reconstruction loss* regularizes the *classification loss* for the classification task. As discussed in Section 3.1, RC-Net is an SCAE. The theoretical justification for improvement of classification performance of SCAE over a CNN is presented next.

Generalization Ability of SCAE

SCAE is a CNN based classifier including both conv and fc layers. Therefore it is essential to represent both conv and fc in an uniform matrix multiplication setup for formulating the SCAE. Next we present SCAE in an unified mathematical framework.

Assume $\mathbf{X} \in \mathbb{R}^{h \times w \times c}$ and $\mathbf{y} \in \mathbb{R}^{m}$ are the product image and the one-hot vector representing the product class respectively. The height, width and color channels of the product image are $h, w,$ and c respectively and m is the number of product classes in a dataset. Given this, our aim is to learn a classifier for the inputs $\mathbf{X} \in \mathbb{R}^{h \times w \times c}$ to predict the targets $\mathbf{y} \in \mathbb{R}^{m}$. We train the classifier with a finite batch of independent and identically distributed n training images of products. Before explaining our CNN based classifier, we first formulate the matrix multiplication setup for performing conv operations.

Convolutional (conv) Operation Let $\mathbf{X}_{in} \in \mathbb{R}^{h_{in} \times w_{in} \times c_{in}}$ be a product image which is the input to a conv layer and we obtain a 3D output $\mathbf{X}_{out} \in \mathbb{R}^{h_{out} \times w_{out} \times c_{out}}$, where $h_{\mathfrak{z}}, w_{\mathfrak{z}},$ and $c_{\mathfrak{z}}$, $\mathfrak{z} \in \{in, out\}$ are the

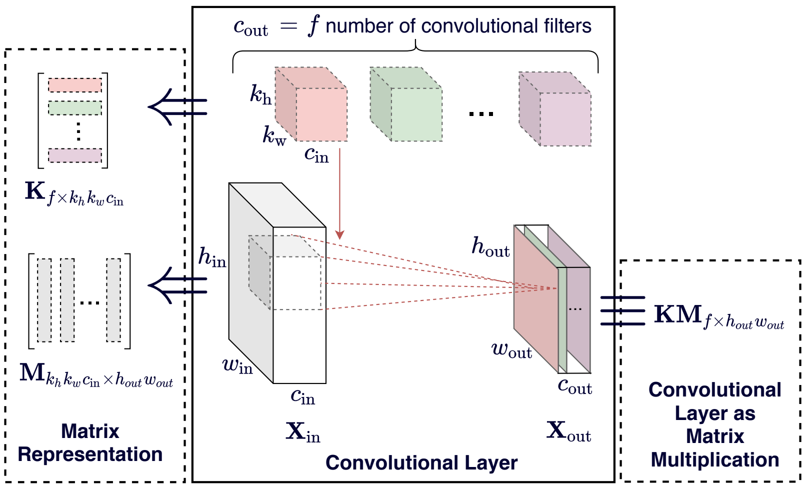

$c_{\text{out}} = f$ number of convolutional filters
k_h
k_w
c_{in}
$\mathbf{K}_{f \times k_h k_w c_{\text{in}}}$
$\mathbf{M}_{k_h k_w c_{\text{in}} \times h_{out} w_{out}}$
Matrix Representation
h_{in}
w_{in}
c_{in}
$\mathbf{X}_{\text{in}}$
h_{out}
w_{out}
c_{out}
$\mathbf{X}_{\text{out}}$
Convolutional Layer
$\mathbf{KM}_{f \times h_{out} w_{out}}$
Convolutional Layer as Matrix Multiplication

Figure 3.5: Graphical illustration of conv layer as matrix multiplication

heights, widths and channels, respectively. Assume, $\boldsymbol{\kappa} \in \mathbb{R}^{f \times k_h \times k_w \times c_{in}}$ is the set of f number of conv filters that belong to $\mathbb{R}^{k_h \times k_w \times c_{in}}$, where k_h and k_w are the height and width of a conv filter. Given that the conv operation is denoted by $*$, $\mathbf{X}_{out} = \mathbf{X}_{in} * \boldsymbol{\kappa}$. The graphical illustration is shown in the middle *Convolution Layer* block of Figure 3.5.

Assume s_w and s_h are the strides in horizontal and vertical directions respectively and $\mathrm{s} = \mathrm{s}_w = \mathrm{s}_h$. Let ψ be the number of rows or columns that are padded with zero in each side of the input in order to produce the spatial size of the output identical to the size of input. Then the shape of the output can be derived as: $h_{out} = \frac{h_{in}+2\psi-k_h}{\mathrm{s}} + 1$, $w_{out} = \frac{w_{in}+2\psi-k_w}{\mathrm{s}} + 1$, and $c_{out} = f$. Given this setting, the following subsections successively present the procedure of converting the operations in conv [Ma and Lu, 2017] and fc layers as matrix multiplications.

conv Operation as Matrix Multiplication Assume h_{in} and w_{in} are the height and width of input matrix after zero padding. The conv filters are moved across $h_{in} \times w_{in}$ space of the input by stride s. This process can be described as extracting $h_{out}w_{out}$ numbers of $k_h \times k_w \times c_{in}$ patches from the product image. Now each patch can be resized to a $k_h k_w c_{in} \times 1$ column vector. Thus the input matrix $\mathbf{X}_{in}$ (with the padded elements) can be represented as a $k_h k_w c_{in} \times h_{out} w_{out}$ matrix $\mathbf{M}$ as shown in the left *Matrix Representation* block of Figure 3.5. Subsequently, each of f number of conv filters in $\boldsymbol{\kappa}$, can be reshaped to a $1 \times k_h k_w c_{in}$ row vector. Thus, these f number of reshaped filters constitute a $f \times k_h k_w c_{in}$ filter matrix $\mathbf{K}$ (refer to the left block of Figure 3.5). Finally the conv operations in a conv layer can be written as the multiplication of two matrices as

$$\mathbf{X}_{out} = \mathbf{KM}, \tag{3.2}$$

where the dimension of $\mathbf{KM}$ is $f \times h_{out} w_{out}$ as shown in the right block of Figure 3.5. By reshaping $\mathbf{KM}$, we can get back the output of size $h_{out} \times w_{out} \times f$ of a conv layer. Similarly, the transposed conv operation [Dumoulin and Visin, 2016] can also be represented as a matrix multiplication. Next we define the operations in fc layer as matrix multiplication operations.

fc Layer as Matrix Multiplication An fc layer can also be formulated in the similar way the conv layer is defined as follows. Each channel (i.e., $h_{out} \times w_{out}$ matrix) in the conv layer (see the middle block of Figure 3.5) can be visualized as a node of a fc layer. Thus $h_{out} = w_{out} = 1$. Similarly, the weights of the connections from input vector to any node of the fc layer can be viewed as the weights of a $k_h \times k_w \times c_{in}$ conv filter, where $k_h = k_w = 1$ and c_{in} is the dimension of the input vector. Consequently the weight matrix $\mathbf{K}$ of size $f \times k_h k_w c_{in}$ for this fc layer is eventually transformed into a $f \times c_{in}$ matrix, where f number of conv filters now turns out to be the number of neurons in the fc layer.

Therefore the input matrix $\mathbf{M}$ of size $k_h k_w c_{in} \times h_{out} w_{out}$ turns out to be a $c_{in} \times 1$ column matrix that essentially represents the input vector to the fc layer. Thus for fc layers, the dimensions of $\mathbf{K}$ and $\mathbf{M}$ become $f \times c_{in}$ and $c_{in} \times 1$, respectively. Given this, the right hand side of (3.2) (i.e., $\mathbf{KM}$) provides $f \times 1$ dimensional output vector of a fc layer. Therefore, the matrix multiplication $\mathbf{KM}$ works same as the conv layer as defined in (3.2). Next we explain our proposed SCAE.

Supervised Convolutional Autoencoder (SCAE) Autoencoders (AE) reconstruct the inputs. An AE with the addition of a classification task on the representation layer is defined as a SAE. The last layer of encoder is usually considered as the representation layer from where the classification block (see Figure 3.6) is added to built a SAE. In this work, we introduce a convolutional framework for SAE which we refer to as supervised convolutional AE (SCAE). The proposed SCAE can be defined as a CNN classifier with an addition of a reconstruction block (see the decoder in Figure 3.6) for reconstructing the input. Addition of reconstruction block into the CNN helps in establishing the generalizability, which we address in the next section. The better generalizabality of SCAE tackles the illumination differences between training and test images of retail product classification problem and improves the performance

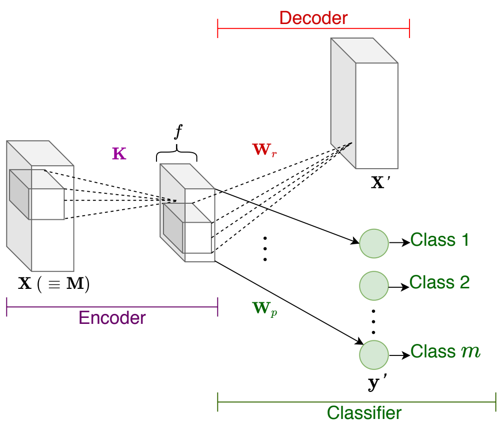


Figure 3.6: A linear supervised convolutional autoencoder (SCAE)

of the product classification. Next we formally introduce SCAE.

Let us consider a linear SCAE with a single hidden conv layer with f number of conv maps as shown in Figure 3.6. Assume the weights of the filters for the first conv layer are $\mathbf{K} \in \mathbb{R}^{f \times k_h k_w c_{in}}$ for the input image $\mathbf{X}$ as discussed in Section 3.3.1. The weights of the output conv and fc layers are $\mathbf{W}_r \in \mathbb{R}^{c_{in} \times k_h k_w f}$ (to reconstruct the product image $\mathbf{X}$) and $\mathbf{W}_p \in \mathbb{R}^{m \times k_h k_w f}$ (to predict the product label $\mathbf{y}$) respectively. Let $\mathbf{y}'$ and $\mathbf{X}'$ be the predicted product label vector and reconstructed product image. Hence, for this linear SCAE, $\mathbf{y}' = \mathbf{W}_p\mathbf{K}\mathbf{M}$ and $\mathbf{X}' = \mathbf{W}_r\mathbf{K}\mathbf{M}$, where $\mathbf{M}$ is the matrix representation of the input product image $\mathbf{X}$ and assume that $\mathbf{KM}$ includes necessary padding of rows and columns for performing transposed conv operation (in decoder, refer Figure 3.6) as discussed in Section 3.3.1. Now assume, $\mathfrak{L}_p(\cdot,\cdot)$ and $\mathfrak{L}_r(\cdot,\cdot)$ are the supervised/prediction and unsupervised/reconstruction losses respectively. Since, classification of products is the primary task of our network, $\mathfrak{L}_p(\cdot,\cdot)$ is regarded as primary loss while $\mathfrak{L}_r(\cdot,\cdot)$ is considered as secondary loss. Thus, given n number of training (product) images, our objective is to minimize the expression

$$\frac{1}{n}\sum_{i=1}^{n} \mathfrak{L}_p(\mathbf{W}_p\mathbf{K}\mathbf{M}^{(i)}, \mathbf{y}^{(i)}) + \mathfrak{L}_r(\mathbf{W}_r\mathbf{K}\mathbf{M}^{(i)}, \mathbf{X}^{(i)}). \tag{3.3}$$

We aim to compare the proposed SCAE with the CNN. First, we theoretically show that the generalization bound can be determined for SCAE. Second, we experimentally establish that SCAE performs better than CNN. Next we present the theoretical results on generalization bounds for SCAE.

Generalization Bounds for SCAE In order to provide the theoretical guarantees of generalization of linear SCAE, we show that the linear SCAE is uniformly stable. Rest of this section refers linear SCAE as SCAE. Assume $\mathtt{D}_1$ and $\mathtt{D}_2$ are the two subsets of a (training) dataset of retail products having equal number of images, where only one image in $\mathtt{D}_2$ is different from the images in $\mathtt{D}_1$.

An SCAE is said to be uniformly stable [Le et al., 2018] if the performances of product classification by the SCAE models separately trained with $\mathtt{D}_1$ and $\mathtt{D}_2$, slightly differ from one another for a fixed number of test images. In [Bousquet and Elisseeff, 2002], Bousquet *et al.* theoretically justify that the uniform stability is one of the important notions for deriving generalization performance of learning algorithms. This chapter establishes that the generalization bounds for SAE shown in [Le et al., 2018] is also satisfied for the SCAE.

Now we show that the trainable parameters $\mathbf{K}$ (referred to as forward model), which are shared by classification (supervised part) and reconstruction (unsupervised part) tasks of the SCAE, do not change significantly (i.e., bounded by a positive real number ϱ) with the change of one sample in D_1 (that constitutes D_2). Hence we can show that the SCAE is uniformly stable.

For ith training image of the dataset $\mathtt{D}_1$, $\mathbf{y}_p^{(i)}$ and $\mathbf{y}_r^{(i)}$ are the true product labels for the classification and reconstruction tasks respectively, the overall loss for the forward model $\mathbf{K}$ can be defined as

$$\mathfrak{L}(\mathbf{K}) = \frac{1}{n}\sum_{i=1}^{n} \mathfrak{L}_p(\mathbf{W}_p\mathbf{K}\mathbf{M}^{(i)}, \mathbf{y}_p^{(i)}) + \mathfrak{L}_r(\mathbf{W}_r\mathbf{K}\mathbf{M}^{(i)}, \mathbf{y}_r^{(i)}). \tag{3.4}$$

In our setting, $\mathbf{y}_p^{(i)} = \mathbf{y}^{(i)}$ and $\mathbf{y}_r^{(i)} = \mathbf{X}^{(i)}$. If we replace the t^{th} training image $(\mathbf{X}^{(t)}, \mathbf{y}^{(t)})$ by a random new instance $(\mathbf{X}'^{(t)}, \mathbf{y}'^{(t)})$, then we obtain the dataset $\mathtt{D}_2$, for which the loss can be written as

$$\mathfrak{L}_t(\mathbf{K}) = \frac{1}{n}\Big[\mathfrak{L}_p(\mathbf{W}_p\mathbf{K}\mathbf{M}'^{(t)}, \mathbf{y}_p'^{(t)}) + \mathfrak{L}_r(\mathbf{W}_r\mathbf{K}\mathbf{M}'^{(t)}, \mathbf{y}_r'^{(t)}) \\ + \sum_{i=1, i\neq t}^{n} \mathfrak{L}_p(\mathbf{W}_p\mathbf{K}\mathbf{M}^{(i)}, \mathbf{y}_p^{(i)}) + \mathfrak{L}_r(\mathbf{W}_r\mathbf{K}\mathbf{M}^{(i)}, \mathbf{y}_r^{(i)})\Big], \tag{3.5}$$

where $\mathbf{M}^{(i)}$ $\left(\text{or } \mathbf{M}'^{(i)}\right)$ is the matrix representation of $\mathbf{X}^{(i)}$ (or $\mathbf{X}'^{(i)}$) as defined earlier in Section 3.3.1.

Assume $\mathbf{K}$ (trained with $\mathtt{D}_1$) and $\mathbf{K}_t$ (trained with $\mathtt{D}_2$) correspond to the optimal forward models for the two losses $\mathfrak{L}(\cdot)$ in (3.4) and $\mathfrak{L}_t(\cdot)$ in (3.5) respectively. Then the SCAE model is said to be ϱ-uniformly stable, if the difference between the primary loss values for these two optimal models $\mathbf{K}$ and $\mathbf{K}_t$ for any test data point $(\mathbf{X}, \mathbf{y})$ is bounded by ϱ i.e.,

$$|\mathfrak{L}_p(\mathbf{W}_p\mathbf{K}_t\mathbf{M}, \mathbf{y}_p) - \mathfrak{L}_p(\mathbf{W}_p\mathbf{K}\mathbf{M}, \mathbf{y}_p)| \leq \varrho. \tag{3.6}$$

Given the matrix multiplication setting for conv layer in Section 3.3.1, without loss of generality, all the **six** assumptions referred in [Le et al., 2018], are equally valid for the input matrix $\mathbf{M}$ corresponding to input product image in the SCAE setting. The bound of generalization ϱ is determined based on these six assumptions that essentially impose certain restrictions on the input product images, targets (or class labels of the product images) and the trainable parameters of SCAE. The assumptions are formally presented in Section C.1 of Appendix C.

By the definition of uniform stability, ϱ in (3.6), which defines the uniform stability of SCAE w.r.t. the forward model (or shared parameters) $\mathbf{K}$, is essentially the generalization bound of SCAE. Now, we determine ϱ for SCAE using the following theorem.

Theorem 3.1. *Under the Assumptions C.1 to C.6, for a randomly sampled data point* $(\mathbf{X}, \mathbf{y})$,

$$|\mathfrak{L}_p(\mathbf{W}_p\mathbf{K}_t\mathbf{M}, \mathbf{y}) - \mathfrak{L}_p(\mathbf{W}_p\mathbf{K}\mathbf{M}, \mathbf{y})| \leq \frac{a(\sigma_r + \sigma_p)n'\sigma_p}{gn}\left(r + \sqrt{r^2 + \frac{4\epsilon g B_{\mathbf{W}_r} B_{\mathbf{M}} r}{a(\sigma_r + \sigma_p)n'}}\right) \\ + \frac{2\epsilon\sigma_p B_{\mathbf{W}_r} B_{\mathbf{M}}}{n}. \tag{3.7}$$

RHS of (3.7) is essentially the ϱ which is the bound on the generalization error defined for a linear SCAE in (3.6). That is, there exists a ϱ for which the classification error of SCAE lies within a closed interval. It indicates that the classification performances are consistent for SCAE. The complete *proof* of the Theorem 3.1 is provided in Section C.2 of Appendix C. Note that (3.7) is valid in a probabilistic sense. Next we find the generalization error for non-linear SCAE.

Non-linear SCAE The linear SCAE, as shown in Figure 3.6, becomes a non-linear SCAE if an activation function $\phi(\cdot)$ is applied in the hidden conv layer. Then the prediction for linear SCAE, $\mathbf{y}' = \mathbf{W}_p\mathbf{KM}$ and $\mathbf{X}' = \mathbf{W}_r\mathbf{KM}$, become $\mathbf{y}' = \mathbf{W}_p\phi(\mathbf{KM})$ and $\mathbf{X}' = \mathbf{W}_r\phi(\mathbf{KM})$ for non-linear SCAE. In that case, a generic bound on the generalization error for any non-linear SCAE cannot be drawn unlike linear SCAE. The formulation of generalization error for non-linear SCAE depends on the type of activation function $\phi(\cdot)$. The activation function $ReLU$ is a standard choice in designing a deep CNN. In fact, our RC-Net utilizes $ReLU$ in all the conv/fc layers. Therefore, we analyze the bound of generalization error for a non-linear SCAE with the $ReLU$ activation function.

For any $x \in \mathbb{R}$, the activation function $ReLU$ is defined as:

$$\phi(x) = \begin{cases} x & \text{if } x > 0, \\ 0 & \text{Otherwise.} \end{cases} \tag{3.8}$$

In that case, for any $x_1, x_2 \in \mathbb{R}$, we get

$$\phi(x_1x_2) = \begin{cases} x_1x_2 & \text{if } x_1x_2 > 0, \\ 0 & \text{Otherwise,} \end{cases} \tag{3.9}$$

and

$$x_2\phi(x_1) = \begin{cases} x_1x_2 & \text{if } x_1 > 0, \\ 0 & \text{Otherwise.} \end{cases} \tag{3.10}$$

Thus, we conclude that

$$x_2\phi(x_1) \leq \phi(x_1x_2). \tag{3.11}$$

Therefore, following (3.11), for our non-linear SCAE,

$$\phi(\mathbf{K})\mathbf{M} \leq \phi(\mathbf{KM}) \implies \boldsymbol{\Phi}_{\mathbf{K}}\mathbf{M} \leq \phi(\mathbf{KM}), \tag{3.12}$$

where $\boldsymbol{\Phi}_{\mathbf{K}}$ denotes $\phi(\mathbf{K})$. Subsequently, similar to linear SCAE, all the aforementioned six assumptions can be revised replacing $\mathbf{K}_t$ by $\boldsymbol{\Phi}_{\mathbf{K}_t}$ and $\mathbf{K}$ by $\boldsymbol{\Phi}_{\mathbf{K}}$. Thus, under these six assumptions, following (3.6) and (3.7), the non-linear SCAE model with $ReLU$ is also ϱ-uniformly stable,

$$|\mathfrak{L}_p(\mathbf{W}_p\boldsymbol{\Phi}_{\mathbf{K}_t}\mathbf{M}, \mathbf{y}_p) - \mathfrak{L}_p(\mathbf{W}_p\boldsymbol{\Phi}_{\mathbf{K}}\mathbf{M}, \mathbf{y}_p)| \leq \varrho. \tag{3.13}$$

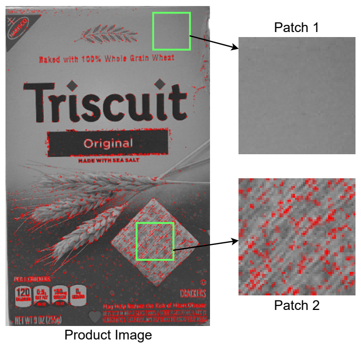


Figure 3.7: Patch 1 and Patch 2 are two arbitrary regions of the given product image. Intensity distribution within Patch 1 is almost homogeneous. Naturally, Patch 1 does not include any key-point. However, there is within-patch intensity variation in case of Patch 2. As a result, Patch 2 includes a number of key-points (marked in red).

The inequality (3.7) in Theorem 3.1 defines ϱ. Following (3.12) and (3.13), we conclude that $-\varrho$ is the *greatest lower bound* (or *infimum*) while ϱ is the *least upper bound* (or *supremum*) of the generalization error for a non-linear SCAE with $ReLU$ activation function. On the other hand, $-\varrho$ is the *lower bound* and ϱ is the *upper bound* of the generalization error for a linear SCAE. Therefore, the generalization bound for linear SCAE is tighter than that of non-linear SCAE. Note that this entire formulation is valid for the non-linear SCAE with a single hidden layer. The non-linear SCAE with multiple hidden layers needs further investigation. However, this analysis shows the benefit of a non-linear SCAE network. Moreover, the improved generalizability of the RC-Net is experimentally validated with the ablation study in Section 3.4.1. Next we present the part-level classification module of our solution.

3.3.2 Part-level Classification

The proposed part-level classification model (a) first generates the part proposals followed by (b) the selection of discriminatory parts and (c) classification of the sequence of those parts using conv-LSTM [Xingjian et al., 2015] as illustrated in Figure 3.3(b). Note that the input product image is first zoomed three times and sent to our part-level classification model as shown in Figure 3.3(b). The motivation of zooming of the input image is explained in the paragraph for *Extracting Features from the Part Proposals* of this section.

Generating Part Proposals

We introduce an unsupervised key-point based approach for generating part proposals. We use BRISK [Leutenegger et al., 2011] to find key-points on the product image I. We observe that these key-points are the most important locations on the product image which require serious attention to derive the important parts or components of the product. In a local neighborhood of the key-point, the change in intensity is higher than other regions which do not include key-points as demonstrated in Figure 3.7. In theory, local maxima of gradient of image and its scale-space versions are precursor to localization of key-points [Leutenegger et al., 2011]. Naturally, the change in intensity should be higher in and around a key-point compared to a homogeneous region. This important observation drives us to extract the part proposals around these key-points.

Assume that BRISK operator finds out η number of key-points on the image P. We extract a patch of height h and width w from P centered at each q^{th} key-point. In our implementation, only square patches (i.e., $h = w$) are considered. In our implementation, the best result is obtained for $h = w = 25$ pixels as detailed in the paragraph for *Choice of h and w in Part-level Classifier* in Section 3.4.1. The bounding box defined by the tuple (x_q, y_q, h_q, w_q) extracts the patch (i.e., sub-image) $\mathscr{P}_q$ from P , where (x_q, y_q), h_q, and w_q be the spatial location of the top left corner, height and width of the q^{th} patch respectively, where $q = 1, 2, \cdots, \eta$. The patches $\{\mathscr{P}_q\}$ are considered as part proposals.

Determining Discriminatory Parts

Given patches or part proposals $\{\mathscr{P}_q\}$, $q = 1, 2, \cdots, \eta$, selection of discriminatory parts consists of two steps, extraction of features from the part proposals followed by selection of winner proposals from the groups of part proposals. These are detailed next.

Extracting Features from the Part Proposals The layers at the end of a CNN contain more discriminatory information compared to the layers at the beginning [Huang et al., 2016]. This is an useful clue for fine-grained classification of objects.

We extract features for each q^{th} patch from the last convolution layer of encoder in the RC-Net. First of all, the patch is resized to the size of receptive field of the last convolution layer. But the size of the receptive field of the last layer of encoder in RC-Net is much larger. For example, the size of receptive field is 160×160 for 224×224 input product image. Assume, the size of a patch extracted from the 224×224 input product image is 25×25. Then resizing 25×25 patch into the size of receptive field 160×160 destroys the spatial relationship between the neighboring pixels (or local features). This spatial relationship essentially indicates the product's part-level cue. In order to avoid this problem, we increase the size of input product image during patch extraction (see *Part-level Classification* block of Figure 3.3(b)) so that the size of the patch becomes similar to that of the receptive field. Note that we obtain the best performance when the input image is zoomed three times. This way the discriminating power of the layers at the end is retained in our implementation.

Next the resized patch is forward propagated through the convolution layers of encoder in the RC-Net to derive the features. Assume, for any patch $\mathscr{P}$, we obtain a latent representation (or features) $x_{z_1 \times z_2 \times f_1}$ from the last convolution layer of encoder in the RC-Net, where f_1 is the number of 2D convolution feature maps, each with $\varkappa_1$ rows and $\varkappa_2$ columns. In other words, $\varkappa$ contains $\varkappa_1\varkappa_2$ numbers of f_1-dimensional vectors $\mathbf{v}_t$, $t = 1, 2, \cdots, \varkappa_1\varkappa_2$.

As discussed in Section 3.2, bilinear (second order) pooling [Lin et al., 2015, Lin and Maji, 2017] captures the local pairwise feature interactions in a translation invariant manner. The local pairwise feature interactions will certainly encapsulate much more discriminatory information about the objects than capturing same information without the pairwise features. The efficacy of bilinear pooling is evident in our ablation study which is carried out in Section 3.4.1. The bilinear pooling on $\varkappa$ results in A:

$$\mathrm{A} = \frac{1}{\varkappa_1\varkappa_2}\left(\sum_{t=1}^{\varkappa_1\varkappa_2} \mathbf{v}_t\mathbf{v}_t^T\right) + \mathrm{diag}(\epsilon, \epsilon, \cdots, \epsilon), \tag{3.14}$$

where A is a matrix of size $f_1 \times f_1$, $\mathrm{diag}(\epsilon, \epsilon, \cdots, \epsilon)$ is a $f_1 \times f_1$ diagonal matrix and ϵ is a small positive value. The matrix A is now flattened to form a f_1^2-dimensional vector $\mathbf{A} = (\mathrm{a}_1, \mathrm{a}_2, \cdots, \mathrm{a}_{f_1^2})$. Inspired by [Perronnin et al., 2010], we then calculate the signed square-root of each element a_i, $\mathrm{i} = 1, 2, \cdots, f_1^2$ of $\mathbf{A}$ as $sign(\mathrm{a}_\mathrm{i})\sqrt{|\mathrm{a}_\mathrm{i}|}$, where $sign(\mathrm{a}_\mathrm{i})$ and $|\mathrm{a}_\mathrm{i}|$ denote the sign and absolute value of a_i respectively. Consequently, the vector $\mathbf{A}$ is normalized into a unit vector by dividing $\mathbf{A}$ with its L2 norm $||\mathbf{A}||_2$ i.e., $\frac{\mathbf{A}}{||\mathbf{A}||_2}$. This normalized $\mathbf{A}_q$ is now the feature vector for the q^{th} patch $\mathscr{P}_q$. The procedure for selecting winner proposal is explained next.

Selection of Winner Proposal Assume β number of discriminatory parts are required to describe the product (ablation study in Section 3.4.1 suggests $\beta = 8$). First η number of part proposals are spatially clustered into β groups. The spatial clustering of part proposals essentially refers to the clustering of co-ordinates of the key-points on the product. We implement k-means clustering algorithm [Lloyd, 1982] to obtain the β group of key-points (or part proposals). Consequently, one proposal from each group is selected for representing the parts as follows.

In each group, a proposal, which is visually similar with maximum number of proposals in the group, is selected as the potential representative of a part. The visual similarity between the proposals $\mathscr{P}_{q_1}$ and $\mathscr{P}_{q_2}$ is measured by calculating the cosine similarity between their features $\mathbf{A}_{q_1}$ and $\mathbf{A}_{q_2}$.

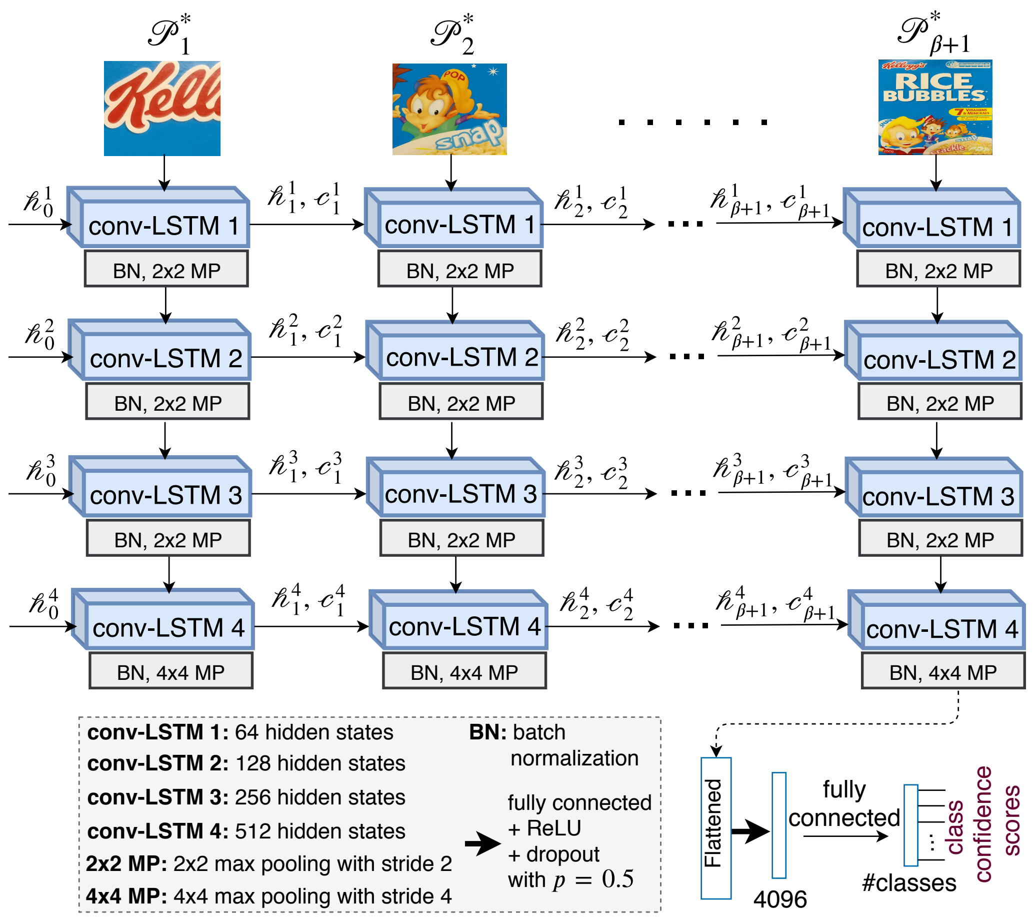
Assume there are ν proposals in the b^{th} group and $\mathrm{S} = [S_{q_1q_2}]$ is the $\nu \times \nu$ similarity matrix, where $S_{q_1q_2}$ denotes the cosine similarity value between the proposals $\mathscr{P}_{q_1}$ and $\mathscr{P}_{q_2}$. S is a symmetric matrix. Then sum of all the elements in the q_1^{th} row (or q_1^{th} column) of S represents the overall similarity between the proposal $\mathscr{P}_{q_1}$ and all other proposals in the group. The sum of elements in the q_1^{th} row (or column) is referred to as the *similarity index* of the proposal $\mathscr{P}_{q_1}$. Let us assume that the highest *similarity index* is obtained for the q_1^{th} row over all the rows in S. Hence, the proposal $\mathscr{P}_{q_1}$ is visually most similar with the other proposals in the b^{th} group. Therefore, $\mathscr{P}_{q_1}$ is selected as a discriminatory part of the product.

Note that the part proposals may be overlapping. However, the actual parts of a product can never be overlapping as one part always exclusively denotes a specific important region of the product. We select only one part proposal (or patch) from each group (or cluster) of part proposals. In our implementation, there are 8 (i.e., the value of β) selected proposals. They hardly overlap. Even if they overlap, it does not affect the classification accuracy at all.

Finally, we obtain β number of discriminatory parts and form a sequence, $\mathcal{P} = [\mathscr{P}_1^*, \mathscr{P}_2^*, \cdots, \mathscr{P}_\beta^*]$ ordered by the location of the top-left corner of each patch within the product image P. The proposed conv-LSTM network for encoding part-level cues of the products designed with this sequence $\mathcal{P}$ is discussed next.

Classification of Sequence of Parts using conv-LSTM Network

We present a stacked conv-LSTM [Xingjian et al., 2015] model for encoding the part-level features (see Figure 3.3(b)) and boosting the overall performance of fine-grained classification of the products. The proposed conv-LSTM network essentially works as a classifier. The block diagram of the proposed conv-LSTM network is shown in Figure 3.8. A major benefit of using conv-LSTM over LSTM [Hochreiter



$\mathscr{P}_1^*$
$\mathscr{P}_2^*$
$\mathscr{P}_{\beta+1}^*$
conv-LSTM 1
conv-LSTM 2
conv-LSTM 3
conv-LSTM 4
BN, 2x2 MP
BN, 4x4 MP
$\hbar_0^1$, $\hbar_1^1, c_1^1$, $\hbar_2^1, c_2^1$, $\hbar_{\beta+1}^1, c_{\beta+1}^1$
conv-LSTM 1: 64 hidden states
conv-LSTM 2: 128 hidden states
conv-LSTM 3: 256 hidden states
conv-LSTM 4: 512 hidden states
2x2 MP: 2x2 max pooling with stride 2
4x4 MP: 4x4 max pooling with stride 4
BN: batch normalization
fully connected + ReLU + dropout with $p = 0.5$
Flattened
4096
fully connected
#classes
class confidence scores

Figure 3.8: Architecture of proposed conv-LSTM network, $\hbar_{\eta}^{\jmath}$ and $c_{\eta}^{\jmath}$ are the hidden states and outputs of $\jmath$th conv-LSTM unit at ηth time step.

and Schmidhuber, 1997] is that the image can be directly fed to conv-LSTM without determining the features.

From the previous step, we obtain the sequence of discriminatory parts $\mathcal{P} = [\mathscr{P}_1^*, \mathscr{P}_2^*, \cdots, \mathscr{P}_\beta^*]$. In the sequence $\mathcal{P}$, the product image P is also included as the last member of the sequence to relate the parts with the product image i.e., $\mathscr{P}_{\beta+1}^* = P$. Thus the updated sequence is $\mathcal{P} = [\mathscr{P}_1^*, \mathscr{P}_2^*, \mathscr{P}_3^*, \cdots, \mathscr{P}_\beta^*, \mathscr{P}_{\beta+1}^*]$ which is the input to the proposed conv-LSTM model.

As illustrated in Figure 3.8, we design a four-layered stacked conv-LSTM network for encoding the part-level information. In the hidden states of the proposed conv-LSTM, we apply $64, 128, 256$, and 512 convolution filters in the conv-LSTM layers (or units) from start to end respectively. Notably, 3×3 padded convolution operation is applied in the proposed conv-LSTM network. The hidden state of the first conv-LSTM unit is essentially the output from the first conv-LSTM unit. The output from the first conv-LSTM unit is then forwarded through a batch normalization and a max pooling layer. Subsequently, the resultant output is set as input to the next conv-LSTM layer. Similar process is iterated for rest of the layers.

The length of the sequence $\mathcal{P}$ (which is $\beta + 1$) is the number of time steps to unroll the conv-LSTM units. The output of our conv-LSTM network at the last time step defines a feature vector $\mathbf{z}$ which is further connected to a fc layer with 4096 neurons (referred to as hidden fc layer). Consequently, conv-LSTM network performs batch normalization followed by ReLU and deterministic dropout [Santra et al., 2020a] operations after the hidden fc layer successively. However, again these 4096 neurons of the hidden fc layer are linked to another fc layer with c number of neurons equivalent to the number of classes (referred to as output layer). The initial weights of the entire network are set following [Ronneberger et al., 2015].

For the product image P, let $\tilde{\mathbf{y}}$ be the predicted vector containing class confidence scores using our conv-LSTM network. Assume $\tilde{\theta}$ denotes the weights and biases of convolution and fc layers of our conv-LSTM network. Now our intention is to train the network with n training samples to optimize the parameter $\tilde{\theta}$ for minimizing the *average classification loss* i.e.,

$$\tilde{\theta}^* = \arg\min_{\tilde{\theta}} \frac{1}{n} \sum_{j=1}^{n} \mathscr{L}_{lcl}\left(\mathbf{y}^{(j)}, \tilde{\mathbf{y}}^{(j)}; \tilde{\theta}\right), \tag{3.15}$$

where $\mathbf{y}$ is the true label of P and $\mathscr{L}_{lcl}(.,.;\tilde{\theta})$ is the traditional *cross-entropy loss* [Goodfellow et al., 2016]. The complete pipeline of classification is provided next.

3.3.3 Complete Classification Model

Both the modules (object-level and part-level) of the proposed solution perform the classification of fine-grained products (see Figure 3.3(a)). The feature reconstruction mechanism of the object-level classifier

RC-Net improves the classification performance over any standard CNN models like VGG [Simonyan and Zisserman, 2015]. Similarly, encoding of part-level features using conv-LSTM helps in accurately classifying fine-grained products. The classification potentiality of RC-Net is much higher than conv-LSTM. But the proposed conv-LSTM network significantly boosts up the classification performance of RC-Net. Therefore we combine them together for accurate classification of fine-grained products as demonstrated in Figure 3.3.

The training of object-level and part-level networks are performed separately. Once training is complete, a product image P (having label vector $\mathbf{y}$) is fed to both the object-level and part-level classification modules to obtain the predicted vectors $\mathbf{y}'$ (see Section 3.3.1) and $\tilde{\mathbf{y}}$ (see Section 3.3.2) containing class confidence scores. The final classification scores $\mathbf{y}_F$ for image P is obtained as

$$\mathbf{y}_F = \mathbf{y}' + \gamma \tilde{\mathbf{y}}, \tag{3.16}$$

where $\gamma \in [0, 1]$ is the weight of the part-level score to improve the object-level classification score for recognition of fine-grained product. Thus, the predicted class confidence scores in $\mathbf{y}_F$ carry both object-level and part-level cues of the fine-grained products. The class probabilities for the image P is then obtained by applying soft-max as

$$y'_{Fi} = \frac{\exp y_{Fi}}{\sum_{\mathfrak{a}=1}^{m} \exp y_{F\mathfrak{a}}}, \quad \forall y_{Fi} \in \mathbf{y}_F, i = 1, 2, \cdots, m. \tag{3.17}$$

The class with highest probability is the predicted label for the image P. Next we present experiments and analysis.

3.4 Experiments

For our product classification task, training data consists of the product templates and the augmented samples from these templates. On the other hand, the cropped (or extracted) products from rack images are the test data. The size of the input image for the object-level classifier is 224×224 whereas the same for part-level classifier is 672×672 after zooming the original input image by a scale of 3. This resizing procedure of all products essentially takes care of correct classification of different sizes of same product. For implementing part-level classifier, we experimentally set the number of discriminatory parts, $\beta = 8$ as detailed in the paragraph for *Ablation Studies* of this section. In our classification scheme, the weight for part-level classification score is experimentally set as $\gamma = 0.6$. Next we describe the normalization of data followed by data augmentation and learning approach of the networks.

Data Normalization The training/test images are first transformed into a fixed-size 224×224 images without altering the aspect ratio. We first resize a $\tilde{w} \times \tilde{h}$ image to $224 \times 224\frac{\tilde{h}}{\tilde{w}}$ if $\tilde{w} > \tilde{h}$, else $224\frac{\tilde{w}}{\tilde{h}} \times 224$.

We then superimpose the resized image in a white 224×224 frame in a way such that the center of the resized image must coincide with the center of the white frame. Next the transformed product image is normalized by dividing all pixel values of the image by the highest pixel value of the image. For part-level classification, we resize the normalized image to the resolution of 672×672.

Data Augmentation For the problem under discussion, only one (or very few) template image(s) is (are) available for each product. But training of deep learning model requires large amount of training data per product class. So we synthetically augment $\sim 10^4$ training samples for each product as we did in the previous chapter (see Section 2.3.1 for details). We utilize these augmented samples for training the object-level and part-level classifiers of the proposed solution.

Learning Approach The augmented product images (including the templates) and their class labels are used to train the network with the pytorch implementation of mini-batch stochastic gradient descent (SGD) [Robbins and Monro, 1951] optimization technique. The hyper-parameters of the SGD for our networks are set experimentally following training algorithms of deep learning based methods [Le et al., 2018, Santra et al., 2020a, Simonyan and Zisserman, 2015, He et al., 2016]. In our experiments, the mini-batch size is set to 2^4. SGD initially starts with a momentum of 0.99, learning rate of 0.001, and weight decay of 10^{-6}. Further the learning rate is updated after each 10 epochs dividing it by 10. Both object-level and part-level classifiers are trained for at most 60 epochs.

Competing Methods The pytorch implementation of pre-trained deep learning networks, VGG [Simonyan and Zisserman, 2015] and ResNet [He et al., 2016] are retrained with our training data for the product classification task following the protocol discussed in the previous paragraph. These networks are also trained for at most 60 epochs. Rest of the competing approaches discussed in Section 3.2 are reproduced following the respective papers.

3.4.1 Results and Analysis

Assume we have n_T number of test images out of which n'_T are correctly classified by a specific method. The accuracy of the method is defined as: $\frac{n'_T}{n_T} \times 100$ (%). Following the standard practice in reporting the performance of deep learning architectures [Santra et al., 2020a, Wan et al., 2013], we determine five-fold classification accuracy for each dataset and report the mean of these five accuracy values for each dataset. This is done in order to consider (five different) random initialization of the deep learning networks. In other words, all the experiments are repeated five times due to five different random initialization of the training data. The average performance (or *mean* accuracy) of these five trials are shown in Table 3.2.

The fine-grained product classification performance on both In-house (comprising of six categories *breakfast cereals* (BC), *deodorant* (DEO), *lip care* (LC), *oral care* (OC), *personal wash* (PW) and *mixed* (MIX)) and benchmark datasets (Grocery Products (GP), WebMarket (WM) and GroZi (GZ)) are tabulated in Table 3.2. The details of these datasets are provided in Appendix B. It is seen that our proposed

Table 3.2: Product classification accuracy (in %) on various datasets. VGG, VGG+DD and ResNet refer to VGG-19, VGG-19 with deterministic dropout (DD) [Santra et al., 2020a] and ResNet-101 respectively. The results for DD [Santra et al., 2020a] are provided from respective paper.

Methods	Categories of In-house Dataset						Benchmark Datasets		
	BC	DEO	LC	OC	PW	MIX	GP	WM	GZ
VGG [Simonyan and Zisserman, 2015]	86.03	88.75	85.19	72.01	78.12	63.29	66.05	64.48	36.17
TLA [Xiao et al., 2015]	86.82	88.12	85.40	72.69	78.46	64.00	70.32	68.21	39.92
PSCNN [Huang et al., 2016]	88.83	89.75	88.68	78.71	81.89	70.44	72.08	68.12	40.63
DRCN [Ghifary et al., 2016]	89.15	89.91	90.03	80.43	83.04	70.60	73.36	70.00	41.60
ResNet [He et al., 2016]	90.09	90.42	88.59	73.73	80.36	68.55	69.50	67.25	38.27
OPA [Peng et al., 2017]	90.13	89.91	89.91	79.58	83.85	73.64	71.77	69.52	41.57
IBP [Lin and Maji, 2017]	89.89	89.88	89.09	78.97	82.63	71.48	72.87	69.06	41.23
WSCP [Ge et al., 2019]	**92.55**	89.84	89.18	86.22	84.05	73.87	77.94	70.60	44.10
DD [Santra et al., 2020a]	-	-	-	-	-	-	**81.62**	-	45.15
VGG+DD	89.21	90.33	87.97	72.13	79.24	66.81	70.81	67.00	38.56
RC-Net	90.19	90.70	**91.93**	87.73	89.51	79.08	75.95	72.40	46.36
FGC	92.47	**90.77**	**91.93**	**90.79**	**92.62**	**84.01**	81.20	**75.71**	**48.12**

classification scheme, which captures the object and part-level cues, stands out as winner (in most of the cases) among all the methods highlighted in Table 3.2. We find that the proposed **RC-Net** (object-level classifier) and **FGC** classification scheme (object and part-level classifier) perform equally well when there are a few number of fine-grained products in a dataset like LC and DEO of In-house dataset.

Our proposed approach also outperforms other methods for benchmark datasets except for GP dataset that includes huge number of product classes (around 3K). Still, our **FGC** scheme (integrating part-level classifier with RC-Net) performs similar to state-of-the-art methods on GP dataset. This supports the potential of our **FGC** scheme that achieves similar performance like state-of-the-art approaches when the number of classes is high. Overall, our **FGC** classification model significantly enhances the fine-grained classification performance. However, Table 3.2 shows that the classification accuracy on In-house dataset is higher compared to the accuracy on benchmark datasets for all the methods. In Table 3.3, we explain why the performance on In-house datasets are impressive compared to the performance on benchmark datasets. In Figure 3.9, example classification results due to various methods for some fine-grained products are demonstrated. The superiority of our proposed scheme is also evident in these example results.

Ablation Study The ablation study is carried out on both object-level and part-level classification modules of the proposed scheme. We have performed ablation studies on (a) the influence of decoder in RC-Net for improving performance, (b) the importance of part-level classifier along with object-level classifier, (c) optimal number of discriminatory parts in building part-level classifier, and (d) effectiveness of bilinear pooling.

	True Class Label (or Ground Truth) (or Product Label) ⟶	Lakme Apricot	Lakme Cherry	NutriGrain 290g	NutriGrain 500g	NutriGrain 805g	RiceBubbles 250g	RiceBubbles 705g
	Cropped Images from Rack (or Test Images) ⟶							
Methods	VGG [Simonyan and Zisserman, 2015]	✗	✔	✗	✗	✔	✗	✔
	TLA [Xiao et al., 2015]	✔	✔	✗	✔	✗	✗	✔
	PSCNN [Huang et al., 2016]	✔	✔	✗	✗	✔	✔	✔
	DRCN [Ghifary et al., 2016]	✔	✗	✔	✗	✔	✗	✔
	ResNet [He et al., 2016]	✔	✔	✔	✔	✗	✗	✔
	OPA [Peng et al., 2017]	✔	✔	✔	✗	✔	✗	✔
	IBP [Lin and Maji, 2017]	✔	✔	✗	✔	✔	✔	✗
	WSCP [Ge et al., 2019]	✔	✔	✗	✔	✔	✗	✔
	DD [Santra et al., 2020a]	✔	✔	✔	✗	✔	✔	✗
	VGG+DD [Santra et al., 2020a]	✗	✔	✗	✔	✔	✔	✗
	RC-Net	✔	✔	✔	✗	✔	✔	✔
	FGC	✔	✔	✔	✔	✔	✔	✔

Figure 3.9: Example classification results of a few fine-grained products for which the cropped images from rack (considered as test image) are shown along with the true class labels (see top of the products). Each of these products is classified using various methods. ✔ indicates that the test image is correctly classified as the true product label while ✗ denotes the incorrect classification of test image.

Table 3.3: Analysis of performance of the proposed approach on different datasets

Dataset		Characteristics of training and test images
In-house		To generate training images, the products arranged on the racks of a supermarket, are taken out from the racks and imaged in isolation in the store environment. Thus, the product templates used in training and product test images cropped from the rack are nearly identical, captured under same lighting condition using the same camera. As a result, there is minimal intensity variation between training and test images and the classification performance for different categories of In-house dataset are roughly between 84% to 92%.
Benchmark	GP	The product templates for training are high resolution images downloaded from the web. Therefore, the intensity variations in the product templates and the cropped product images from the supermarket rack are significant. The training and test images have different resolutions and are possibly captured using different cameras. Further, the dataset includes around 3000+ product classes. As a result, the performance of our method drops to around 81%.
	WM	The products are taken out from the racks and made to lie on the floor face up to capture training images. The imaging system from the top introduces tilt and skew. Naturally, the product templates are not ideally captured. The illumination difference between the rack images and product template images is visible. We have achieved around 75% classification accuracy on this dataset.
	GZ	The resolution of both product templates and rack images are poor. The product template images are downloaded from the web and thus, most of the cases, they are not entirely identical to the products displayed on the rack images. Therefore, the test data largely differ from the training data. As a result, the accuracy on this dataset is nearly 48%.

(a) Influence of decoder in RC-Net for improving performance In our object-level classifier RC-Net, the contribution of reconstruction segment of the network is examined for classification of the products. Note that the reconstruction segment is the decoder of RC-Net (see Figure 3.4). The entire RC-Net is referred to as **Encoder+Decoder+Classifier** while the network without reconstruction segment is symbolized as **Encoder+Classifier**. Figure 3.10 presents the performance of these two networks on both In-house and benchmark datasets. It can be seen that the improvement in classification performance is 8% to 18% when the reconstruction segment is integrated with the **Encoder+Classifier** network.

(b) Importance of part-level classifier along with object-level classifier From Table 3.2, it can be clearly seen that the classification performance drops 1% to 5% for all the datasets except DEO and LC, if we do not integrate part-level module with our object-level module. In Table 3.2, **RC-Net** refers to object-level classifier while **FGC** denotes object-level with part-level classifier. Therefore, part-level classifier indeed boosts up the performance of fine-grained classification.

(c) Optimal number of discriminatory parts in building part-level classifier Our study on optimal number of discriminatory parts in building part-level classifier is performed on PW category of In-house dataset. The experiments are carried out for $\beta = 2, 4, 6, 8, 10, 12, 16,$ and 18 and the corresponding

classification accuracy are provided in Figure 3.11 keeping other parameters unchanged. We can clearly see that the optimal performance is obtained for $\beta = 8$. Moreover, if we do not consider ordered sequence of the discriminatory parts, the performance drops from 92.62% (which is the highest accuracy obtained with $\beta = 8$) to 88.93%. Thus ordered sequence of the parts is important in our model.

(d) Effectiveness of bilinear pooling Bilinear pooling shows good performance in representing fine-grained features of the objects [Lin et al., 2015, Lin and Maji, 2017]. Here we carry out an important ablation study for evaluating the effectiveness of bilinear pooling. Our experiments find that the performance improvement (in %) is 2.05, 0.02, 0.59, 1.79, 2.41, 1.88, 3.67, 1.92 and 0.81 on BC, DEO, LC, OC, PW, MIX, GP, WM, and GZ datasets respectively, if we use bilinear pooling in our part-level classifier. These results exhibit the significant contribution of bilinear pooling in our part-level classifier. Since bilinear pooling consumes insignificant processing time in GPU, this does not add any computational overhead to our part-level classifier.

Choice of h and w in Part-level Classifier As discussed in Section 3.3.2, in our implementation of part-level classification, we consider only square image patches (i.e., $h = w$) for the deep learning framework of our proposal following [Simonyan and Zisserman, 2015, He et al., 2016]. The experiments are carried out for various h (or w) such as $h = 15, 20, 25, 30, 35, 40$ on MIX category of In-house dataset (as this category includes mostly fine-grained products). The respective receiver operating characteristic (ROC) curves are drawn by plotting the true positive rate (TPR) against the false positive rate (FPR) in Figure 3.12 during training. The figure clearly depicts that the area under the curve (AUC) for the ROC curve with respect to h (or w) $= 25$ is higher than the AUCs for other ROC curves. Hence we have chosen $h = w$ =25 pixels in our implementation.

Notes on Test Time We implement the proposed algorithm in python in a computing system with the specs as follows: 64GB RAM, Intel Core i7-7700K CPU @ 4.2GHz $\times$ 8 and TITAN XP GPU. Figure 3.13 illustrates the time (in milliseconds (*ms*)) consumed by different modules of the proposed classifier for processing a single test image. Our classification algorithm recognizes an image of a retail product in 380*ms*. The object-level classifier consumes only 13*ms* to process its input image of resolution 224×224.

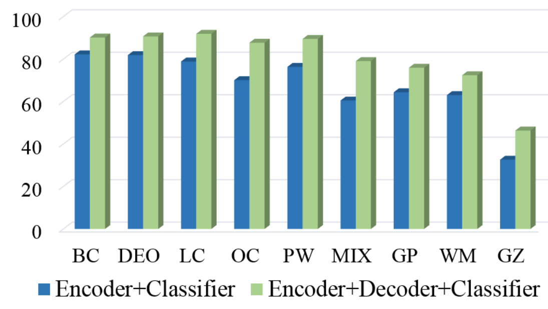


Figure 3.10: Accuracy (in %) of RC-Net with (Encoder+Decoder+Classifier) and with (Encoder+Classifier)

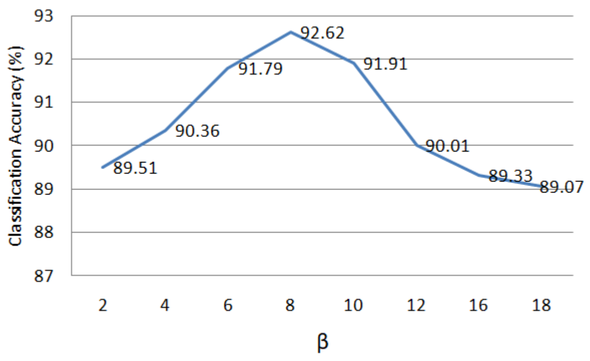


Figure 3.11: Classification accuracy (%) for different values of β

While the part-level classifier completes its execution in 367*ms*, the part detection and part encoding sub-modules take 300*ms* for processing 672×672 image and 67*ms* for processing 224×224 images respectively. Note that part detection refers to the sub-routines *generating part proposals* and *determining discriminatory parts*, and part encoding refers to *conv-LSTM classification network* explained in Section 3.3.2. Therefore, the part-level classifier improves the accuracy at least $\sim$2% (see Table 3.2) at the cost of 367*ms*. Next we present the efficacy of our fine-grained classifier (refer to as FGC) when we embed this in our system built in Chapter 2.

3.4.2 Performance Analysis with the Proposed Fine-grained Classifier

The block diagram of our product detection system (discussed in Chapter 2) is demonstrated in Figure 3.14. As discussed earlier in Chapter 2, it has three primary modules: (i) generation, (ii) classification and (iii) non-maximal suppression of the region proposals. The proposed fine-grained classification module is built inside the classification step (indicated by blue dotted rectangle in Figure 3.14) of the product detection system. Thus in this chapter, we assess the performance of our product detection system with

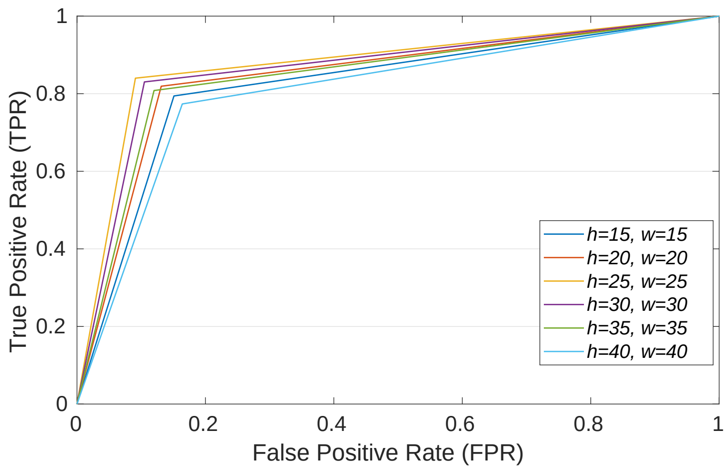


Figure 3.12: ROC curves plotting TPR against FPR for different h and w

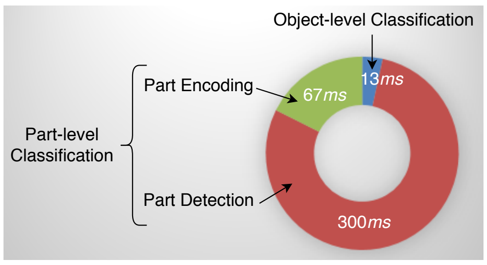


Figure 3.13: Execution time per test image (in *ms*) of various modules

our proposed fine-grained classifier (which we refer to as **ERP+FGC**) for detecting products.

We implement the proposed solution **ERP+FGC** for detecting products in a rack as follows. We first run our ERP (see Section 2.2.1) algorithm on a rack image to generate a number of proposals which are essentially the initial set of detected products. Each proposal is then fed to our FGC to determine the class label and class confidence score. Finally, the products are detected by removing the overlapping proposals using greedy non-maximal suppression (greedy-NMS) [Felzenszwalb et al., 2010] technique.

The same set of methods, which were compared with our system in Section 2.3.2 of Chapter 2, are considered in this chapter for the comparative study. Additionally, we assess the efficacy of our FGC embedding in R-CNN and hence, we implement the R-CNN [Girshick et al., 2014] by substituting its classifier with our FGC, which we refer to as **R-CNN-M**.

Like the previous chapter, we calculate F_1 score (see Appendix A) for analyzing the detection performance. Table 3.4 tabulates the performance of the methods on both In-house and benchmark datasets. Due to correct classification of products, the improvement in performance of **R-CNN-M** over R-CNN is remarkable as shown in Table 3.4. On the other hand, it is seen that **ERP+FGC** improves the performance of **ERP+CNN** by at least ∼1%. Furthermore, this is evident from the table that **R-CNN-M** outperforms other methods (except our **ERP+FGC** and **ERP+CNN**) by at least ∼3% except GDNN [Franco et al., 2017] on GroZi dataset. Our **ERP+FGC** beats all other methods and establishes its superiority.

When we embed the proposed fine-grained classifier in our system built in Chapter 2, our un-

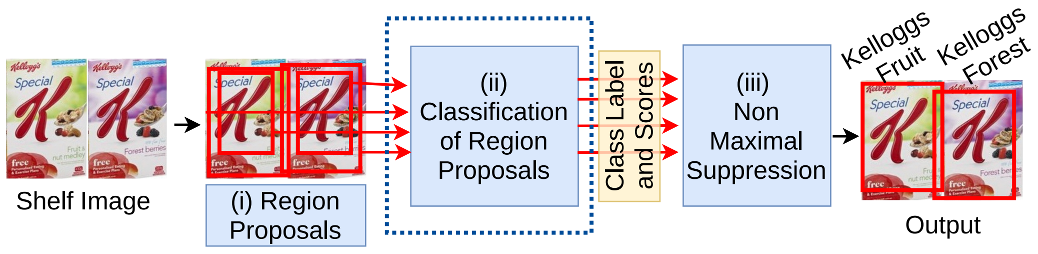


Figure 3.14: Pipeline of the proposed system (in Chapter 2, ERP is introduced for step (i) generation of region proposals) for detecting fine-grained products on the rack. Dotted rectangle highlights our contribution where the classification of products/region proposals are performed using our proposed fine-grained classifier.

Table 3.4: Product detection results (F_1 score in %) on In-house and benchmark datasets

Methods	Categories of In-house Dataset						Benchmark Datasets		
	BC	DEO	LC	OC	PW	MIX	GP	WM	GZ
S1 [Zhang et al., 2007]	41.01	45.21	47.87	48.08	54.91	49.01	58.39	49.19	31.71
CHM [Merler et al., 2007]	48.04	33.56	60.12	30.77	36.40	44.74	51.20	52.81	24.70
MLIC [George and Floerkemeier, 2014]	64.45	50.08	54.91	40.23	59.97	48.76	59.07	53.33	33.10
R-CNN [Girshick et al., 2014]	82.04	83.76	87.99	79.72	88.05	73.16	78.99	72.01	40.91
HOG [Marder et al., 2015]	62.00	28.52	49.37	28.73	44.06	50.62	58.11	43.03	28.33
BoW [Marder et al., 2015]	65.05	45.10	70.72	53.31	71.23	59.91	59.91	55.15	26.83
GBoW [Franco et al., 2017]	72.07	49.98	68.29	46.79	77.22	53.41	74.34	65.59	39.66
GDNN [Franco et al., 2017]	82.12	55.49	82.55	51.32	87.64	61.98	73.09	71.13	43.99
SET [Karlinsky et al., 2017]	83.61	83.95	88.36	81.77	88.16	74.22	79.05	72.13	43.78
U-PC [Ray et al., 2018]	84.77	52.59	86.29	55.65	81.15	65.49	76.20	67.79	40.10
R-CNN-M	87.94	83.76	89.69	85.17	90.79	77.73	79.66	74.00	43.99
ERP+CNN (see Chapter 2)	90.86	83.76	92.49	89.80	92.12	82.98	81.05	78.76	47.49
ERP+FGC	**91.26**	**85.02**	**92.83**	**90.87**	**93.02**	**85.12**	**82.45**	**80.22**	**47.84**

optimized GPU code classifies 20 region proposals in parallel in $367ms$. If our region proposal algorithm generates 500 proposals per rack, the classification time for a rack is approximately $\frac{500}{20} \times 367ms = 9.17s$. We can see that significant improvement in product detection performance (refer Table 3.4) can be obtained at the expense of $\sim$9s. Next we summarize the work done in this chapter.

3.5 Summary

In this chapter, we have addressed an important drawback of the proposed product identification system of Chapter 2, which is mis-classification of fine-grained retail products under varying illumination. To address this issue, we have designed a two-level fine-grained classification scheme considering both part and object-level cues.

The object-level and part-level cues of the product are captured with our object-level and part-level classifiers respectively. The proposed object-level classifier, the RC-Net being a SCAE has the generalization ability that somewhat resolves the illumination difference between training and test product images. Further, RC-Net efficiently encodes the overall (or global) information of a product. This chapter also theoretically shows the generalization ability of SCAE by deriving a bound of generalization error. On the other hand, our unique annotation-free part-level classifier encodes the part-level cues (i.e., local information of a product), where the discriminatory parts of the product images are first identified around the key-points. Then the ordered sequences of these discriminatory parts, encoded using convolutional LSTM, describe the products uniquely. Finally, the part-level and object-level models jointly determine the products explicitly explaining coarse to finer descriptions of the products. Our study finds that the proposed object-level classifier RC-Net generalizes better while the part-level classifier significantly boosts the overall fine-grained classification performance.

We have seen that the overall product classification accuracy is in the range of 90% for In-house dataset. The accuracy is less for benchmark datasets due to large variations between training and test data. However, at the end, we embed this fine-grained classifier in our product detection system (presented in Chapter 2) and we have witnessed the performance improvement of our system. Our system implements greedy-NMS as its last subroutine. It is noticed that there exists some confusion between the choice of best (geometrically) fitted region proposal versus region proposal with higher classification score due to greedy nature of non-maximal suppression (greedy-NMS). As a result, the proposed system sometimes, either misses the best (geometrically) fitted region proposal resulting inaccurate identification of all the (vertically) stacked products, or assigns incorrect class label to a product. In the next chapter, we resolve this issue of greedy-NMS introducing our graph-based NMS posing the problem as a cost optimization problem of a directed acyclic graph.

CHAPTER 4

Graph-based Non-maximal Suppression of Region Proposals

4.1 Introduction

Figure 4.1 illustrates the block diagram of the product detection framework proposed in Chapter 2 and modified in Chapter 3. In the previous two chapters, we have addressed two important issues of detecting retail products: (a) successful generation of region proposals, and (b) classification of fine-grained products. There exists yet another important issue of selecting the right region proposal. The correct choice between the best (geometrically) fitted region proposal versus region proposal with higher classification score, determines the right region proposal. We address this issue of selection of right region proposal in this chapter. After successful selection of region proposal, we are able to (i) accurately identify all the

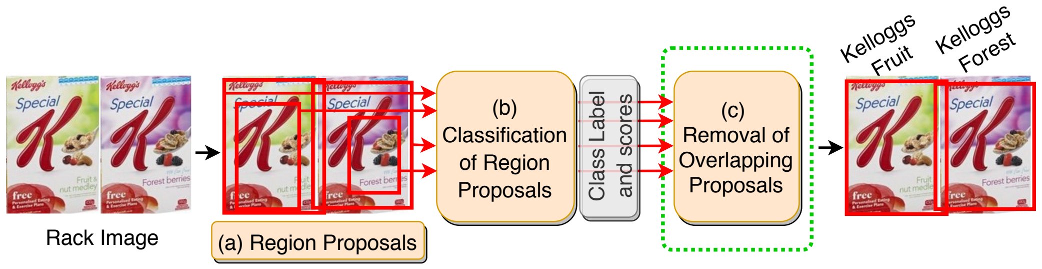


Figure 4.1: Flow chart of our system proposed in Chapter 2 for detecting products. This chapter addresses the step *(c) Removal of Overlapping Proposals* of our system (highlighted by green dotted rectangle) while the steps *(a) Region Proposals* and *(b) Classification of Region proposals* are respectively addressed in Chapters 2 and 3.

(vertically) stacked products and (ii) assign the appropriate class labels to the products present in a rack.

The system for product identification is divided into two phases. In the first phase, a number of potential region proposals (where the potentiality is measured through classification scores) are generated to capture products on the rack. In the second phase, the region proposals, which are unlikely to be a product, are eliminated. Our previous attempts in Chapters 2 and 3 looked into the first phase while we take care of the second phase in this chapter. We have introduced a graph based technique to eliminate the region proposals which are unlikely to be a product. This second phase is the major contribution of this chapter.

As shown in Figure 4.1, our system has three steps (a) generation, (b) classification, and (c) removal of (overlapping) region proposals. Step (a) finds potential regions around the products on a rack image using ERP (see Section 2.2.1). Subsequently step (b) assigns the classification (or confidence) scores and product classes to the proposals using our FGC (see Section 3.3.1). In step (c), our system implements a greedy non-maximal suppression (greedy-NMS) [Felzenszwalb et al., 2010] approach for removing the overlapping proposals in Chapters 2 and 3. The classification scores of proposals are non-maximally suppressed. The contribution of our method in this chapter is to devise a better graph-based non-maximal suppression (G-NMS) strategy to select best potential region to describe a retail product sitting on the rack.

The working principles of greedy-NMS and G-NMS are different. The advantages of G-NMS over greedy-NMS are introduced in this paragraph. Greedy-NMS always eliminates the regions (with lower scores) which are overlapped with the regions with higher scores. This greedy procedure often discards a (geometrically) better fitted proposal (with lower score) which is overlapped with a proposal with higher score. As a result, greedy-NMS does not provide satisfactory result in most cases. We introduce a novel graph-based non-maximal suppression (G-NMS) technique using both classification scores and labels of the proposals for removal of overlapping proposals. G-NMS first determines the *potential confidence score (pc-score)* of a proposal H by finding out the set of proposals (referred to as overlapping group) that are overlapped with H. Subsequently a directed acyclic graph (DAG) is constructed with the proposals using their *pc-scores* and overlapping groups. Finally, we find out the maximum weighted path of the DAG to find the products on the rack.

In [Ray et al., 2018], the authors eliminate overlapping detection by finding out the maximum weighted path of a DAG constructed with the potential regions. But their DAG fails to detect vertically stacked products while the DAG in our G-NMS can detect both horizontally and vertically stacked products as explained and demonstrated in Section 4.2. With respect to state-of-the-art approaches, the contributions of the proposed scheme are listed below:

(a) An unique approach to calculate *potential confidence scores* of the automatic identification of stacked products in the rack images is proposed for modelling a directed acyclic graph.

(b) The graph-based non-maximal suppression scheme is introduced for addressing a serious bottleneck of greedy non-maximal suppression technique.

The remainder of the chapter is structured as follows. Section 4.2 presents the proposed method. The experimental study is carried out in Section 4.3. Finally, Section 4.4 summarizes the chapter.

4.2 Method M3: Graph-based Non-maximal Suppression

Let m be the number of individual products (i.e., product classes) in the database of product templates $\mathbb{D}$. Assume that the individual product be denoted as $\mathbb{D}_t$, $t = 1, 2, \cdots, m$. Let I be the given rack image which displays multiple products. Given this setting, we aim to localize the products $\mathbb{D}_t$ in the rack I.

As illustrated in Figure 4.1, we first run region proposal algorithm, ERP to obtain χ number of region proposals H_z, $z = 1, 2, \cdots, \chi$. These H_z are the initial set of detection, some of which are overlapping. Each H_z is then fed into our classifier, FGC for determining the classification score and product class of the proposal H_z. Next we introduce the G-NMS for removing overlapping/ambiguous detection.

The proposed G-NMS takes region proposals, scores and labels of those proposals and provides the final output with detected products as shown in Figure 4.2. The goal of our G-NMS is to retain at most one detection per group of overlapping detection. The *intersection-over-union* (IoU) [Girshick et al., 2014] between two regions defines the overlap amount (between those two regions). The overlapping group of any proposal H_z is a set of proposals which overlap with H_z by an amount more than a preset IoU value $IoUthresh$. Example group of overlapping detection can be seen in Figures 4.2 and 4.3(a) (refer blue arrow). The greedy non-maximal suppression (greedy-NMS) [Felzenszwalb et al., 2010] retains at most one detection (having highest class confidence score) per overlapping group of regions.

Bottleneck of greedy-NMS In the procedure of greedy-NMS, eliminating all the regions with lower scores that are overlapping with a region with the highest score may sometime remove a potential region

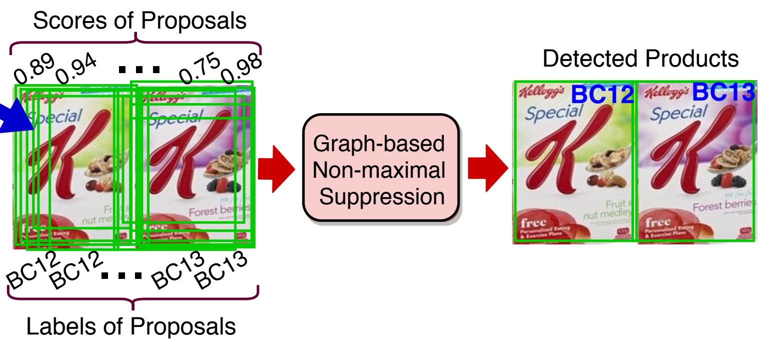
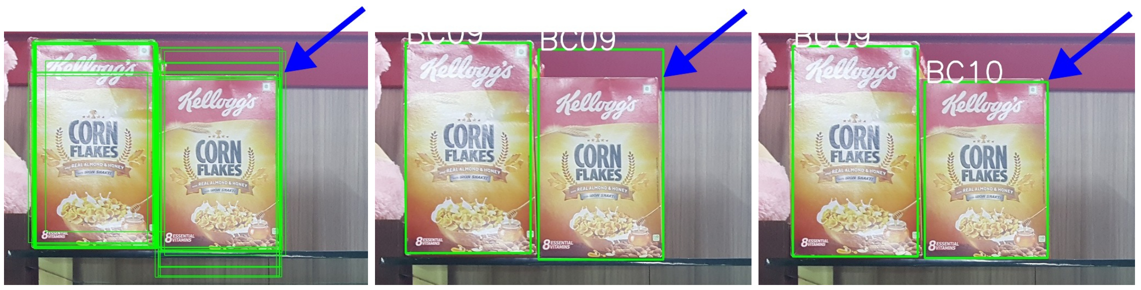


Figure 4.2: Block diagram of the proposed G-NMS scheme. BC12, BC13 and 0.89, $\cdots$,0.98 are the product classes and scores of the proposals respectively obtained from the classifier.



(a) Region proposals (b) greedy-NMS (c) **Proposed G-NMS**

Figure 4.3: For an example rack image, (a) region proposals, and results of (b) greedy-NMS and (c) our proposed **G-NMS**. Product classes are on the top-left corner of the rectangles.

that provides a better geometric fit for the true object. An example of this can be seen in Figure 4.3(b). The detected region using greedy-NMS for the second product (from left) is incorrect even though the region proposals (see Figure 4.3(a)) have provided a better geometric fit (by BC10) for the product. In Figure 4.3(a), BC09 and BC10 are the similar yet non-identical products that are placed on the second positions (from left) on the rack. The only difference between BC09 and BC10 is the higher dimension of BC09. Due to this minor variation between products, the classifier often gets confused and assigns the higher class confidence score to BC09 in the region marked by blue arrow in Figure 4.3(b) over all other regions in the group shown in Figure 4.3(a). As a result, greedy-NMS wrongly detects the product as BC09 (see Figure 4.3(b)).

Why G-NMS over greedy-NMS? The bottleneck of greedy-NMS (explained in the above paragraph and shown in Figure 4.3(b)) is resolved by our G-NMS technique whose result can be seen in Figure 4.3(c). Let there be a region proposal H classified as a product $\mathbb{D}_1$. There are other region proposals overlapping with H. If these other proposals are also identified as $\mathbb{D}_1$, then the classification scores of these proposals are very close to that of H. Further, the variations of overlap amount between H and these proposals should be low. With these two important findings, we first calculate a potential confidence score (or *pc-score*) for each of the region proposals. The *pc-score* is derived by adding two penalty terms with the classification scores of the proposals. The penalty terms essentially characterize previously discussed two findings. Therefore, the *pc-scores* are the rectified classification scores of the proposals. Subsequently, we model a directed acyclic graph (DAG) with the region proposals as the nodes of the DAG. The maximum weighted path of the DAG provides the detected products (see Figure 4.3(c)).

Why G-NMS over DAG in [Ray et al., 2018]? The DAG in [Ray et al., 2018] always selects at most one detection per group of overlapping detection aligned vertically. Their DAG approach fails in detecting all stacked products (refer Figure 4.4(a)). In their method, DAG is designed considering proposals at each and every locations of the rack horizontally. As a result the approach of [Ray et al., 2018] is constrained

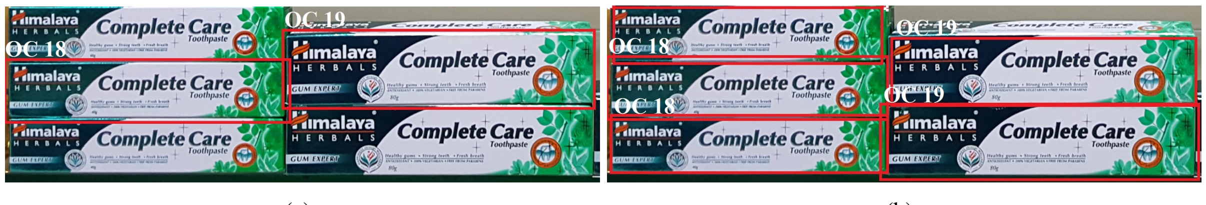


Figure 4.4: Output of (a) DAG in [Ray et al., 2018] and (b) **proposed DAG**. Red boxes show the detections. Product classes are on top-left corner of the boxes.

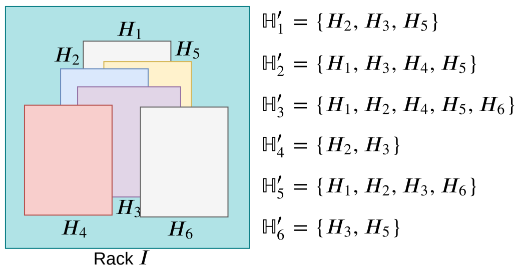
by the horizontal alignment of proposals. In contrast, the present approach prepares DAG considering overlapping groups of proposals where alignment of proposals does not matter. An example using DAG modelled in [Ray et al., 2018] is shown in Figure 4.4(a). The result using our G-NMS approach, which correctly identifies all the products present in the rack, is shown in Figure 4.4(b).

4.2.1 G-NMS with An Example

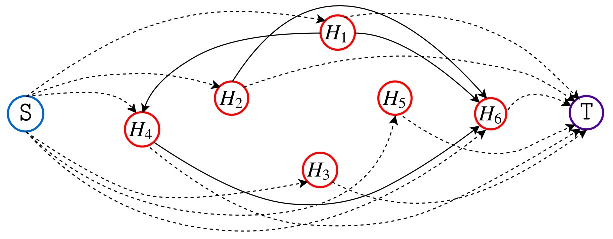
The proposed G-NMS is a two-step process. First, our G-NMS calculates *potential confidence scores* (*pc-scores*) of the proposals. Next G-NMS models a DAG of the regions with the *pc-scores* and finds out the maximum weighted path of the DAG. We now describe these steps using an example presented in Figure 4.5.

In Figure 4.5(a), six region proposals are shown for a rack image I. The proposals are first ordered by sorting their scores in descending order. Let $H_1, H_2, \cdots, H_6$ be the ordered proposals with the scores $s_1, s_2, \cdots, s_6$ respectively, obtained from CNN as explained in the second paragraph of Section 4.2. Thus $s_1 > s_2 > \cdots > s_6$. For each $H_z, z = 1, 2, \cdots, 6$, we first find out the overlapped regions of H_z. As shown in Figure 4.5(a), let H_1, H_3, H_4, H_5 be the overlapped regions of H_2 with the overlap amounts o_1, o_3, o_4, o_5 respectively, and $\mathbb{H}'_2 = \{H_1, H_3, H_4, H_5\}$.

Let the product class of H_2 be $\mathbb{D}_1$. Now we find out other proposals which are also recognized as


H_1
H_2
H_5
H_3
H_4
H_6
Rack I
$\mathbb{H}'_1 = \{H_2, H_3, H_5\}$
$\mathbb{H}'_2 = \{H_1, H_3, H_4, H_5\}$
$\mathbb{H}'_3 = \{H_1, H_2, H_4, H_5, H_6\}$
$\mathbb{H}'_4 = \{H_2, H_3\}$
$\mathbb{H}'_5 = \{H_1, H_2, H_3, H_6\}$
$\mathbb{H}'_6 = \{H_3, H_5\}$

(a)



(b)

Figure 4.5: (a) $\{H_q\}, q = 1, 2, \cdots 6$ are the ordered proposals. $\mathbb{H}'_z$ includes the overlapped regions of H_z s.t. $\forall H_q$, $IoU(H_q, H_z) > IoUthresh$. (b) Graph $\mathtt{G}$ constructed with all H_q as shown in (a). Continuous arrow ($\longrightarrow$): edges between regions, dashed arrow ($\dashrightarrow$): edges between source/sink and region.

$\mathbb{D}_1$ in $\mathbb{H}'_2$. Let H_1 and H_5 be also classified as $\mathbb{D}_1$. Hence out of the four regions in the overlapping group $\mathbb{H}'_2$, two regions $\{H_1, H_5\}$ are classified as the same product and $\mathbb{H}'_2 - \{H_1, H_5\}$ $(\equiv \{H_3, H_4\})$ are the regions classified as the products other than $\mathbb{D}_1$. Assume H_2 better fits a true product (i.e., IoU value between H_2 and the ground truth bounding box (GTBB) for the true product is greater than the IoU values between other three proposals and the GTBB).

As mentioned earlier, the deviation of the classification scores s_1, s_5 of $\{H_1, H_5\}$ in $\mathbb{H}'_2$ from the classification score s_2 of H_2 is less than the deviation of the classification scores of $\{H_3, H_4\}$ in $\mathbb{H}'_2$ from the classification score of H_2. And the standard deviation of the overlap amounts o_1, o_5 of the same is less than that of $\{H_3, H_4\}$. These properties are taken into account when defining the *pc-score* for H_2 (say ps_{H_2}) as

$$ps_{H_2} = s_2 - \frac{|s_1 - s_2| + |s_5 - s_2|}{2} - std\_dev(\{o_1, o_5\}), \tag{4.1}$$

where $std\_dev(\cdot)$ computes the standard deviation of a set. The ps_{H_2} represents not only the class confidence score but also the relative importance of the proposal in an overlapping group.

Using the potential confidence scores and overlapping groups of the regions, we now model a directed acyclic graph (DAG) (say G) as follows. As demonstrated in Figure 4.5(b), the set of vertices in G includes the proposals $H_1, H_2, \cdots, H_6$ and two dummy nodes: source S and sink T. First we draw the outgoing edges from source S to all other vertices except T and assign a small number ϵ as the weights to all these edges. Similarly, the edges are drawn from all the vertices except the source S (i.e., from all the proposals) to the sink T and the weights of the edges are the *pc-scores* of the corresponding proposals. For example, the weight of the edge from H_1 to T is ps_{H_1}. All the connections from/to source/sink are shown using dashed line in the graph in Figure 4.5(b). Rest of the vertices $H_1, H_2, \cdots, H_6$ in G are now sequentially (starting from H_1) selected to make connections between them (highlighted using continuous arrow in Figure 4.5(b)). The weight of any such connection from H_z to $H_{z'}$ is the *pc-score* of H_z i.e., ps_{H_z}. However, the connections between the proposals in G are established in such a way that the graph G does not contain any cycle. Any path of G includes at most one detection per overlapping group.

Therefore, by design, any edge of G should be from one region with higher *pc-score* to another region having lower *pc-score*. This criterion enforces non-existence of cycles in G. The graph G must have an outgoing edge from a region H_z to $H_{z'}$ only when the intersection of the overlapping groups of H_z and $H_{z'}$ (i.e., $\mathbb{H}'_z$ and $\mathbb{H}'_{z'}$) does not include any of these two proposals H_z and $H_{z'}$. Further, the outgoing edge connects H_z to $H_{z'}$ if any predecessor of H_z does not belong to the overlapping group of $H_{z'}$. The graph in Figure 4.5(b) satisfies these properties given the overlapping groups of Figure 4.5(a). The two steps of the proposed G-NMS, determining potential confidence scores of proposals and construction of directed acyclic graph (DAG) (the maximum weighted path of which essentially generates the output) are formally described in the following two successive sections.

4.2.2 Determining Potential Confidence Scores of Proposals:

Let $\mathbb{H} = \{H_z\}, z = 1, 2, \cdots, \chi$ be the sets of ordered proposals. For each $H_z \in \mathbb{H}$, we first find out the set of overlapped regions $\mathbb{H}'_z = \{H_q\}, q = 1, 2, \cdots, \chi' \leq \chi$. However out of χ' number of elements in $\mathbb{H}'_z$, let χ'' number of regions and H_z are recognized as same product. Assume the scores and overlap amount of these identically labelled χ'' number of regions are defined by the sets $\mathbb{S}'_z$ and $\mathbb{O}_z$ respectively. For each H_z in $\mathbb{H}$, we define a *potential confidence score (pc-score)*, ps_{H_z} as follows:

$$ps_{H_z} = s_z - \frac{\sum_{s \in \mathbb{S}'_z} |s - s_z|}{|\mathbb{S}'_z|} - std\_dev(\mathbb{O}_z), \tag{4.2}$$

where $|\cdot|$ denotes the cardinality of a set. If $|\mathbb{O}_z| = 0$ or $|\mathbb{O}_z| = 1$, we set $std\_dev(\mathbb{O}_z) = 0$. If $|\mathbb{S}'_z| = 0$, we set $\frac{\sum_{s \in \mathbb{S}'_z} |s - s_z|}{|\mathbb{S}'_z|} = 0$. In (4.2), the second and third terms of the right hand side define the consistency of scores and closeness of overlaps of the proposals of same class in a group. Thus the proposal with higher *pc-score* in an overlapping group provide a better fit for the true product. Next we present construction of DAG of proposals with the *pc-score*.

4.2.3 Construction of Directed Acyclic Graph (DAG):

The DAG is defined as $\mathtt{G}(\mathbb{V}, \mathbb{E})$ of χ number of ordered region proposals $H_z \in \mathbb{H}$. Here $\mathbb{V}$ and $\mathbb{E}$ define the set of vertices and edges of $\mathtt{G}$ respectively, where $\mathbb{V} = \{\mathtt{S}, \mathtt{T}\} \cup \mathbb{H}$ and $|\mathbb{V}| = \chi + 2$.

The connections between the vertices must be made in a way such that (a) $\mathtt{G}$ is acyclic and (b) each path of $\mathtt{G}$ includes at most one proposal per overlapping group of proposals. Thus the edges between the vertices in $\mathtt{G}$ are drawn as follows.

(P1) Source vertex $\mathtt{S}$ has outgoing edges to each vertex in $\mathbb{V} - \{\mathtt{S}, \mathtt{T}\}$ ($\equiv \mathbb{H}$) with the edge weights $\mathtt{e}_{\overrightarrow{\mathtt{S}H_z}} = \epsilon, \forall H_z \in \mathbb{H}$.

(P2) Sink vertex $\mathtt{T}$ has incoming edges from each vertex in $\mathbb{V} - \{\mathtt{S}, \mathtt{T}\}$ ($\equiv \mathbb{H}$) with the edge weights $\mathtt{e}_{\overrightarrow{H_z\mathtt{T}}} = ps_{H_z}, \forall H_z \in \mathbb{H}$.

(P3) For rest of the vertices H_z, a vertex is sequentially selected (starting from H_1) and the outgoing edges from the selected vertex are drawn as follows. For any two vertices $H_z, H_{z'} \in \mathbb{V} - \{\mathtt{S}, \mathtt{T}\}$ ($\equiv \mathbb{H}$), there exists an edge between them with the associated edge weight $\mathtt{e}_{\overrightarrow{H_zH_{z'}}} = ps_{H_z}$ iff all the following properties are satisfied.

   (i) $z' > z$

   (ii) $H_z, H_{z'} \notin \mathbb{H}'_z \cap \mathbb{H}'_{z'}$

   (iii) $\mathbb{H}'_z \cap \mathbb{P}_z = \emptyset$, where $\mathbb{P}_z$ be the set of predecessors of H_z in $\mathtt{G}$

In the graph $\mathtt{G}$, we determine the maximum weighted path which provides us the detected products in the rack I. The maximum weighted path defines the maximum sum of the *pc-scores* of the region proposals selected from each of the overlapping group of proposals. However, in any graph, determining a maximum weighted path is an NP-hard problem. As $\mathtt{G}$ is a DAG, after negating the weights of the edges, we determine the minimum weighted path in $\mathtt{G}$ using Bellman-Ford algorithm [Bellman, 1958]. The entire procedure of G-NMS is provided in Algorithm 2. Figure 4.3(c) shows correct output using G-NMS compared to the result of Figure 4.3(b). The detail experiment with our proposed scheme is presented next.

Algorithm 2 Graph-based Non-maximal Suppression

```
    Input: Region proposals $\{H_z\}$ (or $\mathbb{H}$) with their classification scores and product classes
    Output: Final detections determined from $\{H_z\}$
 1: procedure IoU ($H_z$, $H_{z'}$)
 2:     return intersection-over-union between $H_z$ and $H_{z'}$
 3: end procedure
 4: procedure OVERLAPPING-GROUP ($H_z$)
 5:     $\mathbb{H}'_z \leftarrow \emptyset$
 6:     for each $H_{z'} \in \mathbb{H}$ do
 7:         if $\text{IoU}(H_z, H_{z'}) > IoUthresh$ then
 8:             $\mathbb{H}'_z \leftarrow \mathbb{H}'_z \cup \{H_{z'}\}$
 9:         end if
10:     end for
11:     return $\mathbb{H}'_z$
12: end procedure
13: procedure PROPOSAL-WITH-IDENTICAL-LABEL ($H_z$)
14:     $\mathbb{O}_z \leftarrow \emptyset$, $\mathbb{S}'_z \leftarrow \emptyset$, $\mathbb{H}'_z \leftarrow$ OVERLAPPING-GROUP($H_z$)
15:     for each $H_{z'} \in \mathbb{H}'_z$ do
16:         if product classes of $H_z$ and $H_{z'}$ are equal then
17:             $\mathbb{O}_z \leftarrow \mathbb{O}_z \cup \{\text{IoU}(H_z, H_{z'})\}$
18:             $\mathbb{S}'_z \leftarrow \mathbb{S}'_z \cup \{\text{classification score of } H_{z'}\}$
19:         end if
20:     end for
21:     return $[\mathbb{O}_z, \mathbb{S}'_z]$
22: end procedure
23: procedure POTENTIAL-CONFIDENCE-SCORE ($H_z$)
24:     $s_z \leftarrow$ classification score of $H_z$
25:     $[\mathbb{O}_z, \mathbb{S}'_z] \leftarrow$ PROPOSAL-WITH-IDENTICAL-LABEL($H_z$)
26:     $ps_{H_z} \leftarrow s_z - \frac{\sum_{s \in \mathbb{S}'_z} |s - s_z|}{|\mathbb{S}'_z|} - std\_dev(\mathbb{O}_z)$        ▷ [using (4.2)]
27:     return $ps_{H_z}$
28: end procedure
29: for each $H_z \in \mathbb{H}$ do
30:     $\mathbb{H}'_z \leftarrow$ OVERLAPPING-GROUP($H_z$)
31:     $ps_{H_z} \leftarrow$ POTENTIAL-CONFIDENCE-SCORE($H_z$)
32: end for
33: Using $ps_{H_z}$ and $\mathbb{H}'_z$, $\forall H_z \in \mathbb{H}$, construct a graph $\mathtt{G}(\mathbb{V}, \mathbb{E})$, $\mathbb{V} = \{\mathtt{S}, \mathtt{T}\} \cup \mathbb{H}$ with source $\mathtt{S}$, sink $\mathtt{T}$,
    and edges $\mathtt{e} \in \mathbb{E}$ drawn following (P1), (P2), and (P3) s.t. $\mathtt{G}$ becomes DAG.
34: Find out the proposals in the minimum weighted path of the negated $\mathtt{G}$ using [Bellman, 1958].
```

4.3 Experiments

For a test rack image, ERP generates the region proposals as described in Chapter 2 and subsequently the proposals are send to our FGC as described in Chapter 3 for obtaining the classification scores and class labels. The proposals with scores less than 0.5 are discarded. Rest of the proposals alongwith their scores and labels are passed to our G-NMS. This entire procedure is referred to as **ERP+FGC+G-NMS**. The experiments for the selection of $IoUthresh$ parameter of G-NMS is presented in the paragraph *Choice of* $IoUthresh$.

The methods, that were considered for our comparative study in Chapters 2 and 3 (see Section 2.3.2 for more details), are also compared with our method in this chapter. In order to evaluate efficacy of G-NMS in other methods, we substitute greedy-NMS in R-CNN [Girshick et al., 2014] with our proposed **G-NMS** technique, which we refer to as **R-CNN-G**. Note that $IoUthresh$ parameter of greedy-NMS is equal to that of our G-NMS. We also consider soft-NMS [Bodla et al., 2017] in our comparative study. The soft-NMS is embedded within our proposed ERP+FGC for evaluating the performance of soft-NMS. This procedure is referred to as ERP+FGC+soft-NMS.

The results of all the methods are evaluated calculating F_1 score, that we have explained in Appendix A. Experiments are carried out on the six categories (*breakfast cereals* (BC), *deodorant* (DEO), *lip care* (LC), *oral care* (OC), *personal wash* (PW) and *mixed* (MIX)) of one In-house and three benchmark datasets Grocery Products (GP) [George and Floerkemeier, 2014], WebMarket (WM) [Zhang et al., 2007] and GroZi (GZ) [Merler et al., 2007], which are detailed in Appendix B.

4.3.1 Results and Analysis

The performances of various schemes on the above mentioned datasets are tabulated in Table 4.1, where best F_1 score for each dataset is highlighted in bold. The proposed **R-CNN-G** (i.e., R-CNN with G-NMS) outperforms all the methods for all the datasets except our methods **ERP+CNN**, **ERP+FGC**, and **ERP+FGC+G-NMS**. As expected, **ERP+FGC+G-NMS** outperforms all other methods by at least $\sim$1% on all the datasets except on LC and PW categories of In-house and GZ datasets. **ERP+FGC+G-NMS** shows similar performance like **ERP+FGC** on PW category of In-house dataset and marginal improvement on LC and GZ from the performance of **ERP+FGC**. Maximum margin with respect to the closest competitor is 3.31% achieved on MIX category of In-house dataset. However, Table 4.1 clearly infers that the ERP module (proposed in Chapter 2) of our system contributes most in improving product detection performance (w.r.t. competing methods) compared to other two modules FGC (proposed in Chapter 3) and G-NMS (introduced in this chapter).

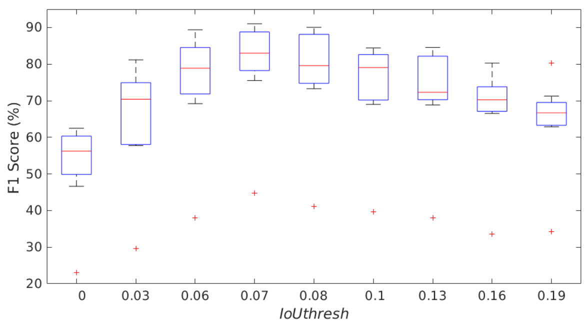
Further, in Table 4.1, we can clearly see that the proposed **G-NMS** (see row for **ERP+FGC+G-NMS**) has established its superiority over greedy-NMS (see row for **ERP+FGC**) in almost all the cases

Table 4.1: Product detection results (F_1 score in %). R-CNN, **ERP+CNN**, **R-CNN-M**, and **ERP+FGC** apply greedy-NMS while **R-CNN-G** and **ERP+FGC+G-NMS** implement our **G-NMS** for removing overlapping proposals.

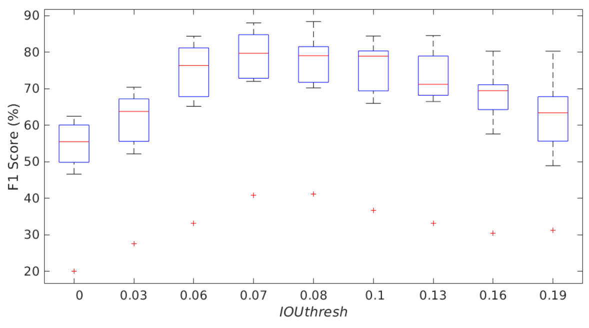
Methods	Categories of In-house Dataset						Benchmark Datasets		
	BC	DEO	LC	OC	PW	MIX	GP	WM	GZ
S1 [Zhang et al., 2007]	41.01	45.21	47.87	48.08	54.91	49.01	58.39	49.19	31.71
CHM [Merler et al., 2007]	48.04	33.56	60.12	30.77	36.40	44.74	51.20	52.81	24.70
MLIC [George and Floerkemeier, 2014]	64.45	50.08	54.91	40.23	59.97	48.76	59.07	53.33	33.10
R-CNN [Girshick et al., 2014]	82.04	83.76	87.99	79.72	88.05	73.16	78.99	72.01	40.91
HOG [Marder et al., 2015]	62.00	28.52	49.37	28.73	44.06	50.62	58.11	43.03	28.33
BoW [Marder et al., 2015]	65.05	45.10	70.72	53.31	71.23	59.91	59.91	55.15	26.83
GBoW [Franco et al., 2017]	72.07	49.98	68.29	46.79	77.22	53.41	74.34	65.59	39.66
GDNN [Franco et al., 2017]	82.12	55.49	82.55	51.32	87.64	61.98	73.09	71.13	43.99
SET [Karlinsky et al., 2017]	83.61	83.95	88.36	81.77	88.16	74.22	79.05	72.13	43.78
U-PC [Ray et al., 2018]	84.77	52.59	86.29	55.65	81.15	65.49	76.20	67.79	40.10
ERP+CNN (see Chapter 2)	90.86	83.76	92.49	89.80	92.12	82.98	81.05	78.76	47.49
R-CNN-M (see Chapter 3)	87.94	83.76	89.69	85.17	90.79	77.73	79.66	74.00	43.99
ERP+FGC (see Chapter 3)	91.26	85.02	92.83	90.87	**93.02**	85.12	82.45	80.22	47.84
R-CNN-G	88.43	83.96	89.91	86.70	91.03	79.17	80.21	75.50	44.81
ERP+FGC+soft-NMS	91.61	85.81	92.88	91.23	92.78	86.13	82.79	81.03	47.67
ERP+FGC+G-NMS	**92.29**	**87.31**	**92.98**	**92.13**	**93.02**	**88.43**	**83.87**	**82.30**	**48.68**

for removing overlapping proposals which is our focus in this chapter. Similar observation can be noticed when we embed our G-NMS with R-CNN. **R-CNN-G** outperforms both R-CNN and **R-CNN-M**. Figure 4.3 shows an example of such improvement. The incorrect identification of product (see blue arrow in Figure 4.3(a)) with greedy-NMS is corrected (see blue arrow in Figure 4.3(b)) with our proposed **G-NMS**.

Choice of *IoUthresh* For choosing the optimal value of *IoUthresh* in our **G-NMS** algorithm, we draw a box-and-whisker diagram (see Figure 4.6(a)) for a set of *IoUthresh* values in between 0.0 and



(a)



(b)

Figure 4.6: Box plot of F_1 scores for different *IoUthresh* values used in (a) **G-NMS** and (b) greedy-NMS

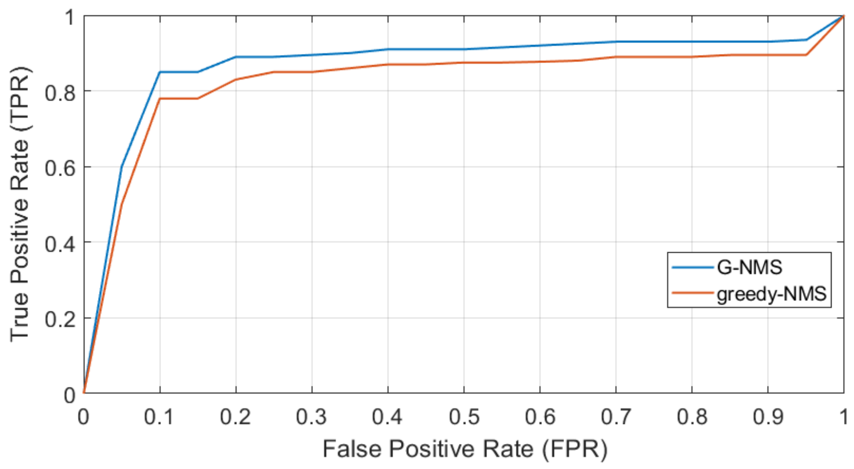


Figure 4.7: ROC plot for the proposed **G-NMS** and greedy-NMS approaches

0.19. Our experiment finds that the overall performance rapidly deteriorates when $IoUthresh \geq 0.2$. So 0.19 is chosen as the upper bound of $IoUthresh$ in our experiment. This experiment is carried out on all the datasets of retail products. In Figure 4.6(a), each box visually represents F_1 scores of **G-NMS** with the specified $IoUthresh$ on all the datasets. On each box, the central red line, bottom edge and top edge respectively indicate the *median*, *first quartile* and *third quartile* of the F_1 scores. The '+' symbol denotes the outliers and the whiskers expand extreme F_1 scores without considering outliers. In Figure 4.6(a), notice that the central red line, bottom edge, and top edge of the box for $IoUthresh = 0.07$ are on the top of central red lines, bottom edges, and top edges of other boxes respectively. In other words, *median*, *first quartile* and *third quartile* of F_1 scores for **G-NMS** with $IoUthresh = 0.07$ are greater than that of F_1 scores for other $IoUthresh$ values. From Figure 4.6(b), similar observations can also be seen for greedy-NMS. We obtain the best performances for **greedy-NMS** also with $IoUthresh = 0.07$.

The superiority of our G-NMS approach over greedy-NMS is also established through the ROC plot in Figure 4.7. ROC plot is drawn by plotting true positive rate (TPR) = $\frac{\text{TP}}{\text{TP+FN}}$ vs. false positive rate (FPR) = $\frac{\text{FP}}{\text{FP+TN}}$ for both the approaches. TP, TN, FP, and FN are defined in Appendix A. TPR and FPR are calculated for 457 rack images of In-house dataset by varying the $IoUthresh$ parameter in the interval $[0, 0.19]$. Figure 4.7 confirms that area under the ROC curve (AUC) for **G-NMS** is significantly higher than AUC for greedy-NMS.

Notes on Test Time All the methods (including the proposed one) are implemented in python and tested in a system with 64 GB RAM, Intel Core i7-7700K CPU @ 4.2GHz$\times$8 and TITAN XP GPU. The computational costs of our proposed G-NMS and greedy-NMS are analyzed in terms of execution time of the respective algorithm. This study is carried out considering both our **ERP** and the region proposal scheme in R-CNN/**R-CNN-G** in the following two ways. (a) For processing a rack image during test, our **ERP** (in **ERP+FGC+G-NMS**) or the other region proposal algorithm (in both R-CNN and **R-CNN-G**)

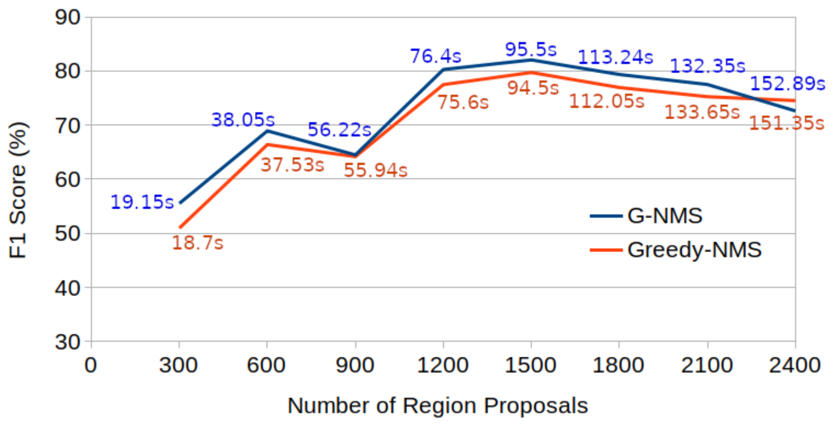


Figure 4.8: F_1 Score vs. number of region proposals plot highlighting the execution time (s) for the proposed **G-NMS** embedded with R-CNN and greedy-NMS embedded with R-CNN

takes almost similar time, ∼40s for generating ∼1500 proposals. However, our **ERP** generates much less number of region proposals than R-CNN/**R-CNN-G**. The classification procedure takes ∼52s in case of our **ERP+FGC+G-NMS** and ∼54s in case of R-CNN/**R-CNN-G**. The proposed **G-NMS** is executed in ∼1.5s while greedy-NMS is completed in ∼0.5s in R-CNN for processing ∼1500 proposals. (b) The second analysis is carried out only for R-CNN vs. **R-CNN-G**. Using the region proposal algorithm in R-CNN or **R-CNN-G**, we can generate proposals as many as we want. We wish to examine the effect of number of region proposals in the product detection performance for our **G-NMS** and greedy-NMS. This is shown in the F_1 score vs. number of region proposals plot in Figure 4.8, which also highlights the execution time in seconds. F_1 score and execution time are determined by varying the number of proposals (changing the parameters of region proposal algorithm in R-CNN or **R-CNN-G**) for a rack image. The above two experiments infer that the slight increase in time comes with the benefit of significant improvement in product detection performance (see Table 4.1). Next we present the summary of this chapter.

4.4 Summary

This chapter resolves a key bottleneck of the greedy-NMS approach that was implemented in Chapter 2. Greedy-NMS discards the proposals (with lower classification scores) that are overlapped with the proposal with higher classification score. This greedy approach often eliminates the (geometrically) better fitted region proposals with (marginally) lower classification scores. Due to this confusion, we often fails to accurately identify all the (vertically) stacked products displayed on the rack and we sometimes miss to recognize the correct label of a product present in the rack. This chapter addresses these issues by presenting a viable graph-based non-maximal suppression (G-NMS) technique.

Greedy-NMS always looks only at the classification scores of the proposals for removing overlapping proposals. Instead of looking only at classification scores, our method elegantly disambiguates overlapping region proposals based on a combination of classification score, class label and extent of overlap. Our proposal has the potential to handle object recognition in a crowded scene. G-NMS first determines the *potential confidence scores* (*pc-scores*) of the region proposals by defining the groups of overlapping regions. Subsequently, a directed acyclic graph (DAG) is strategically constructed with the proposals utilizing their *pc-scores* and overlapping groups. Eventually the maximum weighted path of the DAG provides the products that are present in the rack.

So far we have concentrated on designing a potential solution (or machine vision system) for the automatic identification of retail products on the racks of supermarkets. The efficacy of our methods are evident in our experiments for identification of products. However, we need to concentrate on yet another important aspect of the product identification problem, the identification of empty spaces between products or in the shelves of the racks. We address this challenge in the next chapter by segmenting empty and non-empty spaces with our novel graph-based model.

CHAPTER 5

Identification of Empty Spaces on Shelves

5.1 Introduction

The state-of-the-art methods of Chapter 1 mostly address techniques to identify products. A key direction of the application that needs attention is the automatic identification of gaps (between products) or empty spaces anywhere on the shelves. This automatic identification of empty spaces on the shelves is the focus of this chapter. In order to identify the gaps on the shelves, we can take the complement of product detection results obtained in the previous three chapters. However, that complemented result should detect promotional stickers, colorful or textured background, horizontal and vertical bars of the shelf, etc. along with the gaps. Instead our gap identification approach in this chapter recognizes gap, just as our approaches in previous chapters recognize products.

For planogram compliance, the store associate needs to manually identify the empty spaces. This is time consuming and error prone. Delays in identifying out-of-stock (OOS) conditions lead to significant loss in revenue, especially when the product is present in the inventory [Moorthy et al., 2015] but not displayed on the rack. There exists several automated OOS detection systems like radio-frequency identification (RFID) tags [Michael and McCathie, 2005] and weighted sensor shelves [Moorthy et al., 2015]. They are expensive and difficult to reconfigure for fast changing retail product line. On the other hand, there exists only one known published work [Yılmazer and Birant, 2021] that uses computer vision to detect gaps between products automatically. The use of a camera, either hand-held by a store associate or fixed on the facing rack, seems to be a feasible economic option for continuous monitoring of gap and planogram compliance. In this chapter, we introduce a solution for detecting gaps in the images of shelves. Figure 5.1 illustrates empty regions in an example shelf image.

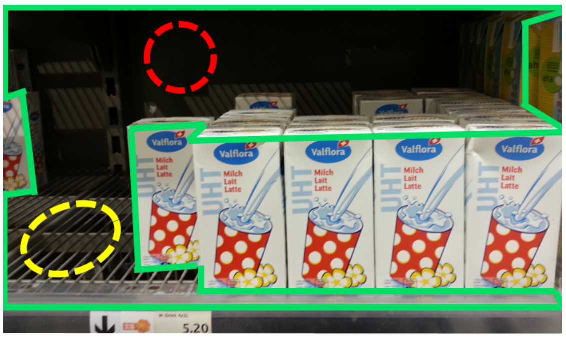


Figure 5.1: In an example shelf image, the regions covered with the green polygon represent the empty spaces. Red and yellow dotted circles highlight different textures for empty space.

We define a gap as an empty space (or a region) created on the shelf after a product is picked up from the shelf. Spaces on the shelf not covered by products (intentionally) are also considered empty spaces. For example, the region covered by the green boundary in Figure 5.1 indicates that a (or few) product(s) is (are) missing. Note that different empty regions may exist in the same image having different textures, colors or features. For example, in Figure 5.1, the regions highlighted by the red and yellow dotted circles on the shelf present different textures for gap. The absence of unique inherent characteristics of empty regions amplify the challenges to solve gap detection problem. Note that, gap, empty region and empty space are interchangeably used in this chapter.

More precisely, we have posed our task of identifying gaps in the shelf images as a segmentation problem. The objective is to estimate the binary mask identifying the empty regions as white and remaining as black. Flow chart of our proposal is shown in Figure 5.2. Our scheme first over-segments the entire image into superpixel regions and construct a graph of superpixels (say, $\widetilde{\mathtt{G}}$), where the edges of $\widetilde{\mathtt{G}}$ captures the association between two superpixels. Subsequently, the features of each superpixel (or each node of $\widetilde{\mathtt{G}}$) is extracted by feeding this $\widetilde{\mathtt{G}}$ as input to a graph convolutional network (GCN) [Kipf and Welling, 2016] that imbibes the neighbourhood information of superpixels within the feature embedding (or feature vector) by graph convolution. Further, the representation of the association/similarity between superpixels i.e., the weights of the edges in $\widetilde{\mathtt{G}}$ (referred to as edge features) are encoded with a Siamese

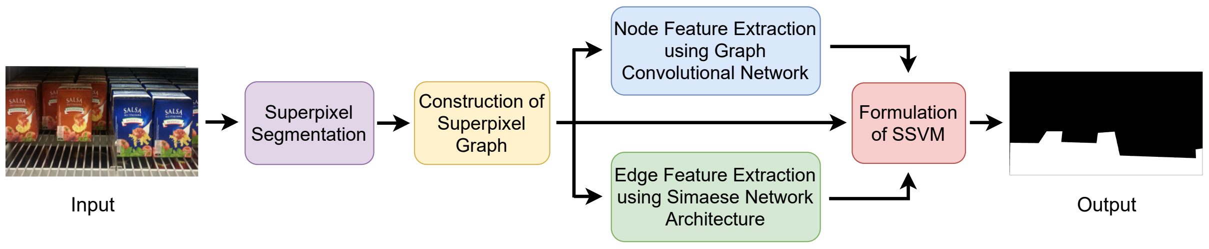

Input
Superpixel Segmentation
Construction of Superpixel Graph
Node Feature Extraction using Graph Convolutional Network
Edge Feature Extraction using Simaese Network Architecture
Formulation of SSVM
Output

Figure 5.2: Process flow of our proposed scheme for identification of empty spaces on the shelves

network [Koch et al., 2015]. This way, our proposal establishes the relationship between the adjacent superpixels being part of a empty or non-empty region in a shelf image using these novel node and edge features of $\widetilde{\mathtt{G}}$. However, using these sets of features for the nodes and edges in $\widetilde{\mathtt{G}}$, we formulate a structural support vector machine to generate a binary mask that classifies the empty region.

There exists many recent state-of-the-art deep learning based image segmentation approaches like U-Net [Ronneberger et al., 2015], DeepLabV3 [Chen et al., 2017], LinkNet [Chaurasia and Culurciello, 2017], FPN [Lin et al., 2017], PSPNet [Zhao et al., 2017], DeepLabV3+ [Chen et al., 2018], PAN [Li et al., 2018], MA-Net [Fan et al., 2020]. These approaches may be used for identification of empty regions on the shelves. We compare these methods with our method in our comparative study in Section 5.4.2. These methods require a large number of training (shelf) images, which is difficult to obtain due to fast changing line of products and display plan in retail stores. Contributions of our proposed scheme compared to all these state-of-the-art approaches are two-folds:

1) We utilize a graph convolutional neural network and a Siamese network in the formulation of a structural support vector machine for detection of empty and non-empty spaces on the shelves with the limited set of training (shelf) images.
2) To the best of our knowledge, we are the first to release the benchmark datasets in GitHub [Santra et al., 2021a] for identification of gaps in the shelves. The identified and annotated gaps are marked in the publicly available datasets Grocery Products [George and Floerkemeier, 2014], WebMarket [Zhang et al., 2007], and GroZi [Merler et al., 2007].

Rest of the chapter is organized as follows. The benchmarking procedure of the shelf images from publicly available datasets are described in Section 5.2. Section 5.3 explains our proposed method. The experiments are carried out in Section 5.4. Finally we summarize the chapter in Section 5.5.

5.2 Benchmark Datasets for Identification of Gaps

For designing and validating a model for automatic identification of gap, we require ground truth for each shelf image specifying the empty and non-empty spaces. In order to generate ground truth, we manually annotate the images of shelves by labelling the empty and non-empty spaces with the polygons.

We follow a certain convention and annotate the images with the two different labels: gap and non-gap polygons drawn on the image. The empty space is defined by (a) the locations of shelf where background of racks are visible and (b) the dark regions where the objects are invisible. In the end, we obtain a binary mask (which we refer to as ground truth) for each shelf image with pixel value 1 as presence of a gap and 0 as presence of a non-gap region (product/non-usable parts of shelf). A binary mask corresponding to an example shelf image is shown in Figure 5.3(b). The regions other than gaps on the shelf image are referred to as non-gap in this thesis.

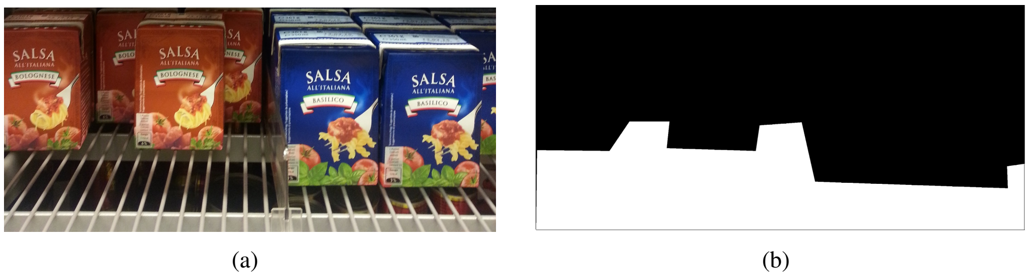


Figure 5.3: (a) An example shelf image, and (b) the ground truth i.e., binary mask for the shelf image (a) where white and black regions denote/highlight gap and non-gap respectively.

The images of shelves for manual annotation are from the publicly available datasets Grocery Products [George and Floerkemeier, 2014], WebMarket [Zhang et al., 2007], and GroZi [Merler et al., 2007], the descriptions of which are provided in Appendix B. We have used the graphical annotation tool *labelme* [Wada, 2016]. Grocery Products dataset includes 680 images of shelves captured from different supermarkets. We select 305 images where the empty regions are present. Similarly, we annotate 98 and 50 images of shelves from the WebMarket and GroZi datasets respectively. A part of these annotations is made public as benchmark datasets for further studies on identification of gaps. The annotations are available at GitHub [Santra et al., 2021a]. Next we present the proposed methodology.

5.3 Method M4: Identification of Empty Spaces on Shelves

We approach the problem under discussion as a binary segmentation problem in which pixels are classified into either of the two classes, 1 (gap) and 0 (non-gap). The overall block diagram of our proposed scheme is illustrated in Figure 5.2. The steps of our scheme are explained in the following subsections.

5.3.1 Superpixel Segmentation

In order to identify the empty regions on the shelves, the proposed scheme aims to label each pixel of the images of shelves. In our case, the number of pixels in the image of a shelf is in the order of 10^5. The procedure for extracting features from each of these many pixels and labeling them is computationally expensive. In order to reduce the complexity, all the images are initially over-segmented into a few regions consisting of a group of pixels, called superpixels. Assume we obtain N number of superpixels in the images of shelves and each superpixel is denoted as x_i, $\forall\, i = \{1, 2, \ldots, \mathrm{N}\}$. In our implementation, we utilize the *simple linear iterative clustering* (SLIC) algorithm [Achanta et al., 2010] for generating superpixels for a shelf image. Next we construct the graph of superpixels for each image.

5.3.2 Construction of Graph of Superpixels

For every shelf image, we construct a graph with the superpixels as the nodes. The edges are connected for the pairs of adjacent superpixels. It is evident that the graph will be a connected-graph as each superpixel is adjacent to at least one other superpixel. We refer this graph of superpixels as *superpixel graph* (SG) in rest of the chapter.

Assume, we have a shelf image I for which we have four superpixels $\mathtt{x}_1$, $\mathtt{x}_2$, $\mathtt{x}_3$, and $\mathtt{x}_4$ as illustrated in Figure 5.4(a). Further assume, $\widetilde{\mathtt{G}}$ be the SG for the image I. Hence, the set of nodes of $\widetilde{\mathtt{G}}$ is $\widetilde{\mathbb{V}} = \{\mathtt{x}_1, \mathtt{x}_2, \mathtt{x}_3, \mathtt{x}_4\}$ and the set of edges is $\widetilde{\mathbb{E}} = \{e_1 = (\mathtt{x}_1, \mathtt{x}_2), e_2 = (\mathtt{x}_1, \mathtt{x}_3), e_3 = (\mathtt{x}_2, \mathtt{x}_3), e_4 = (\mathtt{x}_3, \mathtt{x}_4), e_5 = (\mathtt{x}_2, \mathtt{x}_4)\}$ as shown in Figure 5.4(b). In this work, this SG is an equivalent representation of the superpixels in I. Thus in rest of the chapter, the node $\mathtt{x}_i$ of $\widetilde{\mathtt{G}}$ essentially refers to the superpixel $\mathtt{x}_i$ of I.

In $\widetilde{\mathtt{G}}$, each node $\mathtt{x}_i$ is characterized by the unary feature embedding $\mathbf{u}(\mathtt{x}_i)$, and each edge $(\mathtt{x}_i, \mathtt{x}_j)$ is characterized by the pairwise feature embedding $\mathbf{p}(\mathtt{x}_i, \mathtt{x}_j)$ for the adjacent superpixels $\mathtt{x}_i$ and $\mathtt{x}_j$. Thus, the unary feature embedding refers to the feature vector of a superpixel i.e., a node of $\widetilde{\mathtt{G}}$. The pairwise feature embedding represents a feature vector for two adjacent superpixels i.e., an edge of $\widetilde{\mathtt{G}}$. In rest of the chapter, node and unary features, and edge and pairwise features are interchangeably used. Given these, for the example shown in Figure 5.4(a), we define a structured data $\mathcal{X}$ which consists of:

(a) Adjacency matrix $\mathcal{A}$ of $\widetilde{\mathtt{G}}$,

(b) Pairwise feature embeddings for the edges e_1, e_2, e_3, e_4, e_5 and,

(c) Unary feature embeddings for the nodes $\mathtt{x}_1, \mathtt{x}_2, \mathtt{x}_3, \mathtt{x}_4$.

This $\mathcal{X}$ is essentially the input to Structural Support Vector Machine (SSVM) for identification of gap/non-gap in I. But before discussing SSVM, we define the unary and pairwise feature embeddings for nodes and edges using graph convolutional network (GCN) and a Siamese network respectively.

In our proposal, GCN or Siamese network learn the labels of the nodes (gap being labeled as 1, and non-gap as 0) of $\widetilde{\mathtt{G}}$ (as described in the beginning of Section 5.3) for all the shelf images in the train-set. We extract the feature embedding for nodes and edges of $\widetilde{\mathtt{G}}$ from learnt GCN and Siamese network

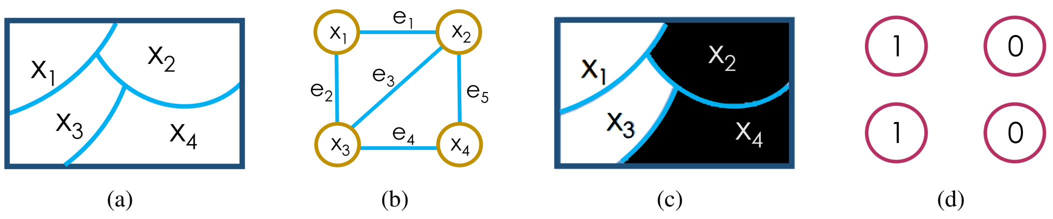


Figure 5.4: (a) Image I segmented in superpixels. (b) Superpixel graph of (a). (c) Binary mask (i.e., ground truth) for (a) labelling each superpixel, where the white and black regions indicate empty and non-empty spaces, respectively. (d) Label for the nodes of superpixel graph (b).

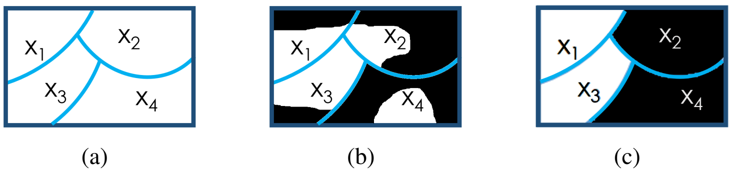


Figure 5.5: (a) Image I segmented in superpixels as shown in Figure 5.4(a). (b) Ground truth pixels overlayed on (a). (c) Label of each superpixel of (a), where the white and black regions indicate empty and non-empty spaces, respectively. Majority of pixels in superpixel x_1 is white as shown in (b). Therefore, the label of superpixel x_1 is given as white as shown in (c). Similarly, labels of superpixels x_2, x_3 and x_4 are determined.

respectively. However, in order to learn GCN or Siamese network, an SG and the labels of superpixels are required. We have already explained the procedure for constructing SG in the beginning of this section. Next we describe the procedure for defining labels to the superpixels of a shelf image before extracting features for the nodes and edges of $\widetilde{G}$.

5.3.3 Labelling of Superpixels

The labelling of superpixels is demonstrated using an example in Figure 5.5. Figure 5.5(a) is a shelf image I segmented in four superpixels x_1, x_2, x_3 and x_4. Figure 5.5(b) is a ground truth image (or the binary mask I_{gt}) with each pixel marked either as white or black representing empty or non-empty regions respectively. Figure 5.5(b) shows that superpixel x_1 has majority of white pixels. Therefore, the label of superpixel x_1 is assigned as white as shown in Figure 5.5(c). Similarly, based on majority voting of white or black pixels, superpixels x_2, x_3 and x_4 are labelled as black, white and black, respectively as shown in Figure 5.5(c). This way the binary mask B labeling each superpixel of the shelf image I is determined. In rest of this chapter, mask B is refereed to as structured ground truth mask. Next we extract features for the nodes of $\widetilde{G}$.

5.3.4 Feature Representation of the Nodes in SG

The unary feature embedding in the structured data $\mathcal{X}$ of $\widetilde{G}$ is essentially the feature representation for the nodes (i.e., superpixels) of $\widetilde{G}$. Since we look for a novel feature representation for each superpixel of I considering not only the superpixel itself but also its neighboring superpixels, we design a GCN [Kipf and Welling, 2016] that principally accumulates the local neighborhood information of each superpixel utilizing graph convolution. Naturally, the SG $\widetilde{G}$ is input to the GCN.

The block diagram of the proposed GCN model is presented in Figure 5.6. The architecture of the proposed three layered GCN is detailed in our implementation Section 5.4. However, a GCN usually takes adjacency matrix of the graph $\widetilde{G}$ and an initial feature vector for each node of $\widetilde{G}$. In this work, the

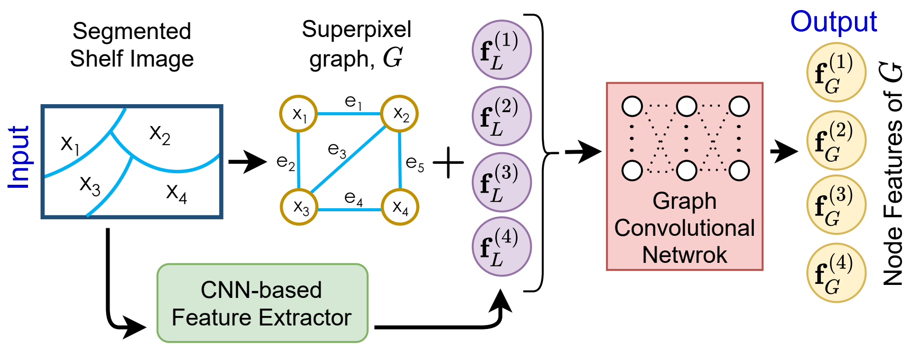


Figure 5.6: Flowchart of the proposed node feature extraction scheme. For the segmented image in Figure 5.4, superpixels x_1, x_2, x_3, and x_4 are sent to CNN-based feature extractor for determining respective initial feature vectors $\mathbf{f}_L^{(1)}$, $\mathbf{f}_L^{(2)}$, $\mathbf{f}_L^{(2)}$, and $\mathbf{f}_L^{(4)}$. These initial feature vectors and the SG are passed through the graph convolutional network for obtaining the features $\mathbf{f}_G^{(1)}$, $\mathbf{f}_G^{(2)}$, $\mathbf{f}_G^{(3)}$, and $\mathbf{f}_G^{(4)}$ of the nodes x_1, x_2, x_3, and x_4 respectively.

initial feature vectors for the nodes or superpixels are determined using a CNN based feature extractor (referred to as initial feature extractor or IFE) designed on top of the pre-trained VGG-19 [Simonyan and Zisserman, 2015] (see Figure 5.6). The architectural details of this IFE are also provided in the implementation details in Section 5.4. Assume, IFE returns $\mathfrak{d}_1$-dimensional feature vector $\mathbf{f}_L^{(i)}$ for each i^{th} node of $\widetilde{\mathtt{G}}$. In our implementation, $\mathfrak{d}_1 = 128$.

Then $\widetilde{\mathtt{G}}$ along with the initial feature vectors for all its nodes $\mathbf{f}_L^{(i)}, i = 1, 2, \ldots, \mathrm{N}$, are send to GCN and layer-wise propagated in GCN [Kipf and Welling, 2016] as follows:

$$\mathbf{H}^{(\ell+1)} = \phi\left(\widetilde{\mathbf{D}}^{-\frac{1}{2}}\widetilde{\mathcal{A}}\widetilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{H}^{(\ell)}\mathbf{W}^{(\ell)}\right) \tag{5.1}$$

where $\widetilde{\mathcal{A}} = \mathcal{A} + \mathrm{I}_\mathrm{N}$ is the adjacency matrix of $\widetilde{\mathtt{G}}$ considering self loops, $\mathcal{A}$ is the adjacency matrix of $\widetilde{\mathtt{G}}$, and I_N is the identity matrix of size N. Here $\widetilde{\mathbf{D}}_{ii} = \sum_j \widetilde{\mathcal{A}}_{ij}$ and $\widetilde{\mathbf{D}}_{ij} = 0,\ \forall i \neq j$. $\mathbf{H}^{(\ell)}$ is the input to the ℓ^{th} layer of GCN and hence $\mathbf{H}^{(0)}$ is the initial input to GCN. $\mathbf{H}^{(0)}$ is the input matrix containing the initial feature vectors $\mathbf{f}_L^{(i)}, i = 1, 2, \cdots, \mathrm{N}$. $\mathbf{W}^{(\ell)}$ is the trainable weight matrix for layer ℓ and $\phi(\cdot)$ is the activation function. For extracting features of the nodes in SG (or superpixels), we build a three-layered GCN. The last layer of GCN classifies the input superpixel (or the node of SG) to gap or non-gap. Once GCN learns its parameters, the output of the penultimate layer can be considered as the feature representation of a superpixel. This way GCN defines a $\mathfrak{d}_2$-dimensional feature vector $\mathbf{f}_G^{(i)}$ for each i^{th} node of $\widetilde{\mathtt{G}}$ aggregating the local neighborhood information of a superpixel. The aggregation of local information is ensured by the inclusion of $\widetilde{\mathcal{A}}$ in (5.1). In our implementation, $\mathfrak{d}_2 = 16$. In other words, the unary feature embedding for each i^{th} node x_i of $\widetilde{\mathtt{G}}$ can be defined as $\mathbf{u}(x_i) = \mathbf{f}_G^{(i)}$. Next we present our method for finding out features of the edges of $\widetilde{\mathtt{G}}$.

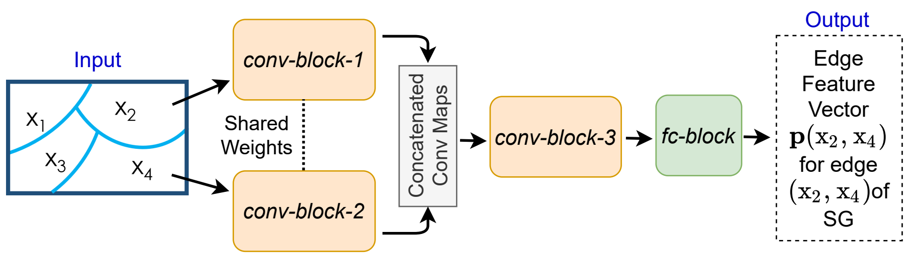


Figure 5.7: Schematic of the proposed Siamese network architecture for extraction of edge feature. For the segmented image example in Figure 5.4, we illustrate the extraction of pairwise feature $\mathbf{p}(x_2, x_4)$ for the edge (x_2, x_4) of SG, $\widetilde{\mathtt{G}}$.

5.3.5 Feature Representation of the Edges in SG

As mentioned earlier, there exists an edge between two nodes x_i and x_j in $\widetilde{\mathtt{G}}$, if the superpixels x_i and x_j in I are adjacent. In this work, we aim to define a feature for the edge (x_i, x_j) that encodes the similarity (or dissimilarity) between the two neighboring superpixels x_i and x_j in I. We can pose this as a two class (similar or dissimilar) classification problem that takes two superpixels (or sub-images) as input. We build a Siamese network architecture (SNA) [Koch et al., 2015] to classify a pair of superpixels as similar or dissimilar. Once the SNA is learnt with the training examples, the features for the edges in $\widetilde{\mathtt{G}}$ are extracted from the SNA.

Each pair of adjacent superpixels (x_i, x_j) in I is the input to our SNA for finding out the feature for the edge (x_i, x_j) in $\widetilde{\mathtt{G}}$. The schematic of our SNA is provided in Figure 5.7 and its architecture is detailed in our implementation in Section 5.4. However, for any pair of adjacent superpixels (x_i, x_j), the superpixel x_i is fed to the first convolutional block, *conv-block*-1 while x_j is sent to the second convolutional block, *conv-block*-2. A convolutional block refers to a stack of convolutional layers. According to the principle of Siamese network, the learnable weights of the blocks *conv-block*-1 and *conv-block*-2 are shared (see Figure 5.7). The weights (i.e., convolutional filters) are shared in order to project both the input superpixels into a common feature space. Both the input superpixels are first transformed to their respective covolutional maps using the shared weights. The outputs (i.e., convolutional (conv) maps) of both the blocks are then concatenated and fed to the third block *conv-block*-3 of our SNA. The output of *conv-block*-3 is finally passed through a *fc-block* comprising of three consecutive fully connected (fc) layers. Last fc layer defines the classification score of similarity and dissimilarity between two input superpixels x_i and x_j. Once the SNA learns to classify a pair of superpixels as similar or dissimilar, the output of penultimate layer of *fc-block* of SNA represents the features for the input superpixels. This is how we obtain the feature representation for the edge between x_i and x_j. The SNA provides $\mathfrak{d}_3$-dimensional pairwise feature vector/embedding $\mathbf{p}(x_i, x_j)$ for each edge (x_i, x_j) in $\widetilde{\mathtt{G}}$. In our implementation, $\mathfrak{d}_3 = 16$. Next we present the SSVM for identification of gaps/non-gaps in the shelf images.

5.3.6 SSVM for Identification of Empty Spaces

SSVM [Xue et al., 2008] is a classifier which predicts the labels for the nodes of $\widetilde{\mathtt{G}}$ minimizing the loss between the predicted and true labels. In our problem, we utilize SSVM to learn the labels of the nodes (i.e., 0 or 1) of the $\widetilde{\mathtt{G}}$ (as described in Section 5.3.2) for all the shelf images in the train-set. Once the SSVM is learnt, we obtain the labels of the nodes which we assign to the corresponding superpixels of a shelf image. In this way, the entire image is segmented into gaps and non-gaps (empty space being labeled as 1, else 0). Next we formulate the SSVM with the structured data $\mathcal{X}$ (see Section 5.3.2) and the labels of superpixels (see Section 5.3.3).

Formulation of SSVM

Assume, we have ϑ_{trn} number shelf images $I^{(\mathrm{k})}$, $\mathrm{k} = 1, 2, \ldots, \vartheta_{trn}$ in the train-set. Corresponding to each training image $I^{(\mathrm{k})}$, we obtain the structured data $\mathcal{X}^{(\mathrm{k})}$ and the true labels $Y^{(\mathrm{k})}$ for the superpixels. Given these, the gap identification problem can be posed as follows.

Each of the superpixels of any shelf image I can be interpreted as a discrete random variable taking values from the set $\Omega = \{0, 1\}$, where 1 signifies gap and 0 denotes non-gap. Let us assume that $Y = \{\mathrm{y}_1, \mathrm{y}_2, \ldots, \mathrm{y}_\mathrm{N}\} \in \mathscr{Y} = \Omega^\mathrm{N}$ be a feasible label vector for the N number of nodes in $\widetilde{\mathtt{G}}$. In that case, there exists 2^N possible label vectors (or feasible labeling) for the $\widetilde{\mathtt{G}}$ with N number of nodes i.e., $|\mathscr{Y}| = 2^\mathrm{N}$. Thus, the set of 2^N feasible label vectors must include the true label (Boolean) vector (for the SG). Now the gap identification problem boils down to finding out the true label vector from 2^N possible label vectors for the $\widetilde{\mathtt{G}}$. For example, the possible label vectors for the graph shown in Figure 5.4(b) is $Y = (\mathrm{y}_1, \mathrm{y}_2, \mathrm{y}_3, \mathrm{y}_4)$, where $\mathrm{y}_i, i = 1, 2, 3, 4$, can be 0 or 1. Then the number of possible label vectors for this graph is 2^4 out of which we aim to find out the true label vector $(1, 0, 1, 0)$ as shown in Figure 5.4(d).

In order to obtain the true label vector $Y^{(\mathrm{k})}$ for any input $\mathcal{X}^{(\mathrm{k})}$, we define a potential function $\mathrm{E}(\mathcal{X}, Y)$ which will be maximized when $Y = Y^{(\mathrm{k})}$ for the given $\mathcal{X} = \mathcal{X}^{(\mathrm{k})}$ i.e.,

$$Y^{(\mathrm{k})} = \arg\max_{Y} \mathrm{E}(\mathcal{X}^{(\mathrm{k})}, Y), \tag{5.2}$$

and the potential function $\mathrm{E}(\mathcal{X}, Y)$ is formulated as:

$$\mathrm{E}(\mathcal{X}, Y) = \mathbf{w}^\mathrm{T} \boldsymbol{\omega}(\mathcal{X}, Y), \tag{5.3}$$

where $\mathbf{w}$ is the weight vector and $\boldsymbol{\omega}$ is the joint feature vector for an input $\mathcal{X}$ and its possible label vector Y. Our target is to learn this weight vector $\mathbf{w}$ with the train-set such that the potential function E is maximized for the true label vector.

The potential function $\mathrm{E}(\mathcal{X}, Y)$ can be defined as the sum of the potential functions $\mathrm{E}_\mathbf{u}(\mathrm{x}_i, \mathrm{y}_i)$ and

$E_{\mathbf{p}}(x_i, x_j, y_i, y_j)$ contributed by the unary and pairwise features as:

$$\arg\max_{Y} E(\mathcal{X}, Y) = \arg\max_{Y} \left(\sum_{i=1}^{N} E_{\mathbf{u}}(x_i, y_i) + \sum_{(x_i, x_j) \in \widetilde{\mathbb{E}}} E_{\mathbf{p}}(x_i, x_j, y_i, y_j) \right). \tag{5.4}$$

Again $E_{\mathbf{u}}$ can be written as:

$$E_{\mathbf{u}}(x_i, y_i) = \mathbf{w}_{\mathbf{u}}^{T} \boldsymbol{\omega}_{\mathbf{u}}(x_i, y_i), \tag{5.5}$$

where $\boldsymbol{\omega}_{\mathbf{u}}$ is the joint feature vector of the node x_i and its label y_i. $\boldsymbol{\omega}_{\mathbf{u}}$ associates the unary features of the nodes with their labels. $\boldsymbol{\omega}_{\mathbf{u}}$ is defined as:

$$\boldsymbol{\omega}_{\mathbf{u}}(x_i, y_i) = \Big(\mathbb{I}(y_i = 0)\mathbf{u}(x_i),\ \mathbb{I}(y_i = 1)\mathbf{u}(x_i) \Big), \tag{5.6}$$

where $\mathbb{I}(\cdot)$ is the indicator function that checks the condition and returns 1 if the condition holds and 0 otherwise. Again the potential function $E_{\mathbf{p}}$ contributed by pairwise features can be written as:

$$E_{\mathbf{p}}(x_i, x_j, y_i, y_j) = \mathbf{w}_{\mathbf{p}}^{T} \boldsymbol{\omega}_{\mathbf{p}}(x_i, x_j, y_i, y_j), \tag{5.7}$$

where $\boldsymbol{\omega}_{\mathbf{p}}$, which is the joint feature vector of the nodes x_i, x_j and their labels y_i, y_j for pairwise features, is defined as:

$$\begin{aligned} \boldsymbol{\omega}_{\mathbf{p}}(x_i, x_j, y_i, y_j) \quad = \quad & \Big(\mathbb{I}'(y_i = 0, y_j = 0)\mathbf{p}(x_i, x_j), \mathbb{I}'(y_i = 0, y_j = 1)\mathbf{p}(x_i, x_j), \\ & \mathbb{I}'(y_i = 1, y_j = 0)\mathbf{p}(x_i, x_j), \mathbb{I}'(y_i = 1, y_j = 1)\mathbf{p}(x_i, x_j) \Big), \end{aligned} \tag{5.8}$$

where $\mathbb{I}'(\cdot, \cdot)$ is an indicator function that returns 1 if both the conditions (in its arguments) are true and return 0 otherwise. As given in (5.6), $\boldsymbol{\omega}_{\mathbf{u}}$, which is a function of a node x_i and its corresponding label y_i in SG, returns a vector of length $2\mathfrak{d}_2$ (as $\mathbf{u}(x_i)$ is a $\mathfrak{d}_2$-dimensional vector, see Section 5.3.4). Similarly from (5.8), we can see that $\boldsymbol{\omega}_{\mathbf{p}}$ is a function of a pair of adjacent nodes x_i, x_j and their corresponding labels y_i, y_j in $\widetilde{\mathbb{G}}$ returning a vector of length $4\mathfrak{d}_3$ (as $\mathbf{p}(x_i)$ is $\mathfrak{d}_3$-dimensional vector, see Section 5.3.5). Thus, the joint feature vector $\boldsymbol{\omega}$ in (5.3) can be derived as:

$$\boldsymbol{\omega}(\mathcal{X}, Y) = \left(\sum_{i=1}^{N} \boldsymbol{\omega}_{\mathbf{u}}(x_i, y_i), \sum_{(x_i, x_j) \in \widetilde{\mathbb{E}}} \boldsymbol{\omega}_{\mathbf{p}}(x_i, x_j, y_i, y_j) \right).$$

Given these, we aim to learn $\mathbf{w}$ in (5.3) with the SSVM [Xue et al., 2008]. However, before we train the SSVM for determining $\mathbf{w}$, we have to define the loss between the true and predicted label vectors for any input $\mathcal{X}$ by the SSVM model. In order to do that, we calculate the Hamming loss between the true and predicted label vectors as:

$$\Delta(Y^{(k)}, Y) = \sum_{i=1}^{N} \mathbb{I}(y_i^{(k)} \neq y_i), \tag{5.9}$$

where Y and $Y^{(k)}$ are (any) feasible and true label vectors of an input $\mathcal{X}^{(k)}$ respectively. Since Y and $Y^{(k)}$ are the Boolean vectors (as mentioned in the second paragraph of this subsection) i.e., elements of which are either 0 or 1, the Hamming loss is calculated. In (5.3), $E(\mathcal{X}, Y)$ eventually is the function which maps each label vector Y of the SG in $\mathcal{X}$ to a scalar value (or score). Hence, SSVM is learnt in a way such that (a) the true label vector has the highest score and (b) the score is lower when the Hamming loss defined in (5.9) is higher. Keeping all these in mind, we define an SSVM with the formulation of one slack SVM [Joachims et al., 2009] as:

$$\min_{\mathbf{w}} \quad \lambda||\mathbf{w}||^2 + \frac{1}{\vartheta_{trn}} \sum_{k=1}^{\vartheta_{trn}} \varepsilon_k, \tag{5.10}$$

such that,

$$\sum_{k=1}^{\vartheta_{trn}} \Big(\Delta(Y^{(k)}, \hat{Y}^{(k)}) - \mathbf{w}^T \boldsymbol{\omega}(\mathcal{X}^{(k)}, Y^{(k)}) + \mathbf{w}^T \boldsymbol{\omega}(\mathcal{X}^{(k)}, \hat{Y}^{(k)}) \Big) \leq \sum_{k=1}^{\vartheta_{trn}} \varepsilon_k,$$
$$\forall (\hat{Y}^{(1)}, \hat{Y}^{(2)}, \ldots, \hat{Y}^{(\vartheta_{trn})}) \in \mathcal{Y} \times \mathcal{Y} \times \ldots \times \mathcal{Y}, \tag{5.11}$$

where ε_k are the slack variables, λ is a positive regularization constant, $\mathcal{Y}$ is the set of all possible label vectors $\hat{Y}^{(k)}$ for $\mathcal{X}^{(k)}$. The equation (5.10) is a convex quadratic optimization problem with $|\mathcal{Y}|^{\vartheta_{trn}}$ number of constraints, where (5.11) represents all the constraints. Finally, we obtain the weight vector $\mathbf{w}$ which is learnt using the structured input data (as defined in the third paragraph of Section 5.3.2) of the images of shelves from the train-set following the approach in [Joachims et al., 2009].

For any test shelf image I, we predict the empty/non-empty regions by creating the structured data $\mathcal{X}$ as explained in Section 5.3.2 and using the trained SSVM as:

$$\hat{Y} = F(\mathcal{X}) = \arg\max_{Y \in \mathcal{Y}} \mathbf{w}^T \boldsymbol{\omega}(\mathcal{X}, Y). \tag{5.12}$$

We solve (5.12) using AD^3 algorithm [Martins et al., 2015]. Hence $\hat{Y} = F(\mathcal{X})$ is the predicted label vector for the nodes of SG in $\mathcal{X}$ i.e., the labels of the superpixels in the shelf image I. Thus $\hat{Y}$ produces the predicted binary mask $\hat{B}$ for any shelf image. Subsequently, the predicted label in $\hat{B}$ for each superpixel x_i is assigned to all the pixels contained within the superpixel x_i and thus finally the binary mask is obtained with all pixels labeled either 0 (for non-gap) or 1 (for gap). Next we present the experiments, results and analysis.

5.4 Experiments

5.4.1 Implementation Details

First of all, any shelf image is resized into a fixed-size 700×460 image. Next SLIC segmentation method [Achanta et al., 2010] is run for $\mathrm{N} = 1000$ number of superpixels. The experiments for choosing N is provided in the paragraph *Choice of Number of Superpixels (*N*)* of Section 5.4.2. The compactness [Achanta et al., 2010] of SLIC method is experimentally set to 50 in this work.

The superpixels being of irregular shape cannot be send to a CNN based model (IFE and Siamese network architecture in our scheme) directly as input. So in our implementation, a patch of shape 32×32 is cropped out for each superpixel centred at the centroid of the superpixel. First, all these patches of superpixels are fed to IFE for extracting initial feature vectors to fit into GCN.

The architecture of IFE is composed of a convolutional (conv) block (comprising of a number conv and pooling layers) and two fc layers, fc-1 and fc-2 on top of the convolutional block. The architecture of the conv block is identical to the entire conv block of VGG-19 [Simonyan and Zisserman, 2015]. fc-1 and fc-2 layers have 128 and 2 nodes respectively. fc-2 classifies each superpixel patch to either gap or non-gap. Further, we perform ReLU and dropout (with dropout probability 0.5) operations just after fc-1 and before fc-2. However, the learnable weights of the conv block of IFE is initialized with the pre-trained weights of pytorch [Paszke et al., 2017] implementation of VGG-19 while the weights of fc-1 and fc-2 layers are randomly initialized with the values in $[-1, 1]$ drawn following normal distribution. In our implementation, the input to fc-2 defines the 128-dimensional feature vectors for each of the superpixel.

The adjacency matrix of the SG and the 128-dimensional feature vectors of each superpixel are the input to our GCN. The proposed GCN is comprised of 3 graph convolutional (gc) layers, namely gc-1, gc-2 and gc-3 as shown in Figure 5.6. gc-1, gc-2 and gc-3 layers include 64, 16 and 2 number of nodes. After each of gc-1 and gc-2 layers, we execute ReLU and dropout (with dropout probability 0.5) operations. Weights of all these gc layers are randomly initialized with the values in $[-1, 1]$ drawn following normal distribution. gc-3 layer eventually classifies each node of SG i.e., each superpixel to gap or non-gap aggregating the features of adjacent superpixels in gc layers. In this work, the output from gc-2 is the (unary) feature vector for any node of SG.

The feature vector for two adjacent superpixels or an edge of the SG is determined by sending their patches to our SNA. As shown in Figure 5.7, SNA consists of 3 conv blocks: *conv-block-1*, *conv-block-2*, *conv-block-3* and 1 fc block: *fc-block*. Since, *conv-block-1* and *conv-block-2* share their weights, they are essentially treated as one block whose architecture is identical to the conv block of VGG-16. The outputs from *conv-block-1* (which takes one superpixel as input) and *conv-block-2* (which receives another superpixel as its input) are concatenated and fed to *conv-block-3*. The structure of *conv-block-3* is identical to the chunk of 10-th to 16-th conv layers (including intermediate maxpool layer) of VGG-19. The weights of these blocks are initialized with weights of respective blocks of the VGG pre-trained models. The *fc-block* (consisting of 3 fc layers having 64, 16 and 2 nodes respectively) takes the output of *conv-block-3* and classifies the adjacent superpixels to be identically labeled or not (refer Section 5.3.5). *fc-block* also performs ReLU and dropout (with dropout probability 0.5) operations after each of first two fc layers. Weights of all these fc layers are randomly initialized with the values in $[-1,1]$ drawn following normal distribution. In our model, the output from penultimate layer of *fc-block* is eventually the (pairwise) feature vector for the edge between the adjacent superpixels in SG. The SG along with these unary and pairwise feature vectors are sent to SSVM for identification of gaps and non-gaps in shelf images.

All the above deep learning based models, IFE, GCN and SNA are implemented with pytorch library [Paszke et al., 2017] while SSVM is designed with pystruct library [Müller and Behnke, 2014] of python using Titan XP GPU. Next we explain the training strategies of the above models under discussion.

Training During training of the deep learning based models IFE, GCN, and SNA of our proposed approach, approximately 80% images of the train-set are used for training while rest 20% images are utilized for validation of the network. All the networks are trained by applying softmax function on output (referred to as softmax output) and then calculating the cross-entropy loss between the softmax output and one hot label vector. Adam optimizer at learning rate of 0.0001, weight decay of 5e-4, and mini-batch (of shelf images) of 1 are used to learn the networks. IFE and SNA are optimized upto 150 epochs while GCN is trained for at most 400 epochs. On the other hand, for training of SSVM, the chosen tunable parameters are maximum iteration of 100, a regularization parameter of 0.1 and a convergence tolerance of 0.1. Next we present the competing methods.

Competing Methods We compare our proposal with the competing methods: U-Net [Ronneberger et al., 2015], DeepLabV3 [Chen et al., 2017], LinkNet [Chaurasia and Culurciello, 2017], FPN [Lin et al., 2017], PSPNet [Zhao et al., 2017], DeepLabV3+ [Chen et al., 2018], PAN [Li et al., 2018], MA-Net [Fan et al., 2020]. We have used pytorch implementation of all these methods with default setup available in GitHub [Yakubovskiy, 2020]. All these networks are trained for 100 epochs. Next we present the results and analysis.

5.4.2 Results and Analysis

Performance Measure The methods are evaluated using the measure, *intersection-over-union* (IoU) [Dice, 1945] used in evaluating the performances of the methods in semantic segmentation. The IoU essentially determines the similarity between the predicted binary mask $\hat{\text{B}}$ and true binary mask I_{gt} of an image of a shelf. Subsequently, the gap identification performance for the ϑ_{tst} number of test images in a dataset is defined by the *mean* IoU $(mIoU)$ as :

$$mIoU = \frac{1}{\vartheta_{tst}} \left(\frac{|I_{gt} \cap \hat{\mathrm{B}}|}{|I_{gt} \cup \hat{\mathrm{B}}|} \right). \tag{5.13}$$

The experiments are carried out on three publicly available datasets Grocery Products (GP) [George and Floerkemeier, 2014], WebMarket (WM) [Zhang et al., 2007], and GroZi (GZ) [Merler et al., 2007] (see Appendix B for more details on these datasets) from which we select 305, 98, and 50 number of shelf images and create ground truth specifying empty/non-empty regions as explained in Section 5.2. For each dataset, we randomly choose approximately 60% of these shelf images as train-set and rest 40% as test-set. The train-set includes 184, 59, and 30 number of shelf images in GP, WM, and GZ respectively while test-set contains 121, 39, and 20 number of images in the respective datasets. These train-set and test-set containing images and their ground truths are made public in GitHub [Santra et al., 2021a].

Table 5.1 presents the gap identification accuracy in terms of $mIoU$ (%) of the proposed approach including the competing ones. The proposed scheme outperforms deep learning based state-of-the-art methods in all the evaluations by at least ∼1%. The maximum performance improvement from the nearest competitor is ∼3% (see right most column for GZ in Table 5.1) which is indeed remarkable. In fact, our method achieves the higher accuracy for GZ dataset. Our method performs equally well for the datasets having larger or smaller number of training images (for GP there are 184 training images while for GZ there are only 30 training images). On the contrary, the performance of purely deep learning based methods deteriorates with the decreasing size of train-set, which is why, the margin of $mIoU$ for our method w.r.t. others becomes higher when train-set is smaller as witnessed for GZ. However, the results for all the methods are inferior on the GP and WM datasets compared to GZ due to large variation

Table 5.1: Gap identification results of various methods on benchmark datasets. DLV3 and DLV3+ represents DeepLabV3 and DeepLabV3+ respectively.

Methods	$mIoU$ (%)		
	GP	WM	GZ
U-Net [Ronneberger et al., 2015]	69.36	66.83	81.76
DLV3 [Chen et al., 2017]	67.82	64.89	78.47
LinkNet [Chaurasia and Culurciello, 2017]	68.73	66.28	79.25
FPN [Lin et al., 2017]	67.36	66.19	77.60
PSPNet [Zhao et al., 2017]	66.19	64.55	72.30
DLV3+ [Chen et al., 2018]	68.98	64.75	75.70
PAN [Li et al., 2018]	68.95	64.93	77.31
MA-Net [Fan et al., 2020]	69.64	63.60	81.66
Proposed	**70.62**	**69.20**	**84.58**

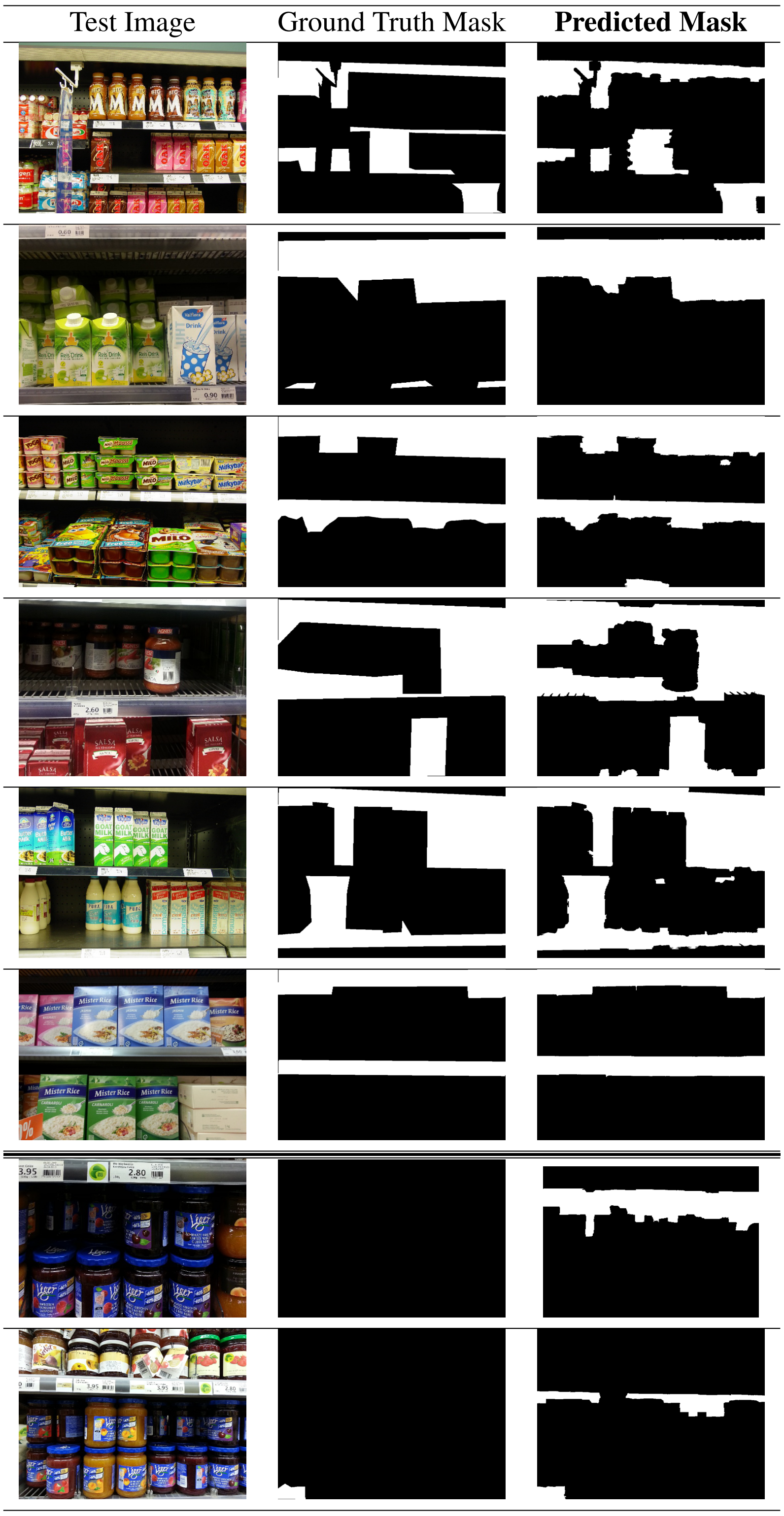


Figure 5.8: A few qualitative results from the test-set of various datasets. Top six rows (starting from second row) show the efficacy of the proposed scheme while the last two rows present the failure cases of our solution when the products with darker packaging appear like a gap.

Table 5.2: Performances of our method removing or adding different components of it on test-set of our benchmark datasets

Components in our scheme	$mIoU$ (%)		
	GP	WM	GZ
(i) NFE	65.98	60.06	79.11
(ii) NFE + SSVM	66.75	63.76	79.89
(iii) **Proposal:** NFE + EFE + SSVM	70.62	69.20	84.58

in the texture of gaps and in the packaging of products.

A few example qualitative results of our method are provided in Figure 5.8. The efficiency of our method is established in the eight example results shown in top six rows (displaying images) of Figure 5.8, where the predicted binary masks are almost similar to the true binary masks. The bottom two rows of Figure 5.8 illustrate two notable failure cases. Our analysis finds that the non-gap is misidentified as gap in both the images due to darkness in packaging of the product. Next we perform the ablation study.

Ablation Study The ablation study is carried out on all benchmark datasets for investigating the contributions of different components of the proposed scheme. Our proposal has three primary components such as: node feature extractor (NFE) i.e., IFE + GCN, edge feature extractor (EFE) i.e., GCN, and SSVM i.e., our classifier. In the proposed scheme, NFE is the basic component without which SSVM cannot be executed. Therefore, next we provide the efficacy of the remaining two components EFE and SSVM of our proposal to identify the gaps or non-gaps in the shelf images.

Contribution of EFE SSVM can be modeled using the SG and its node features NFE, without explicitly extracting edge features of SG. In that case, SSVM considers the adjacency value (1 or 0) in the adjacency matrix of SG as edge feature. This setup, which can be denoted as NFE + SSVM, examines the necessity of EFE i.e., SNA in our proposal. If we remove EFE module from our proposal, the performance drops at least $\sim$5% (compare rows (ii) and (iii) of Table 5.2) that clearly shows the importance of EFE. This happens because the Siamese network considers each pair of adjacent superpixels and efficiently captures the discriminatory characteristics between them as the edge feature.

Contribution of SSVM We can use only NFE for obtaining the gap identification results. The results of this model essentially illustrate the contribution of SSVM. If we look at the performances of our proposal (row (iii)) and NFE (row (i)) in Table 5.2, the difference is at least $\sim$5% and at most $\sim$10%. Thus the necessity of SSVM can be clearly noticed.

Hence, our ablation study suggests that all three components of our proposal are significant in accurately identifying gaps and non-gaps in shelf images. To be specific, this study infers that SSVM contributes most in achieving higher gap identification performance with respect to competing approaches.Next we analyze the test time (or inference time) of the proposed method.

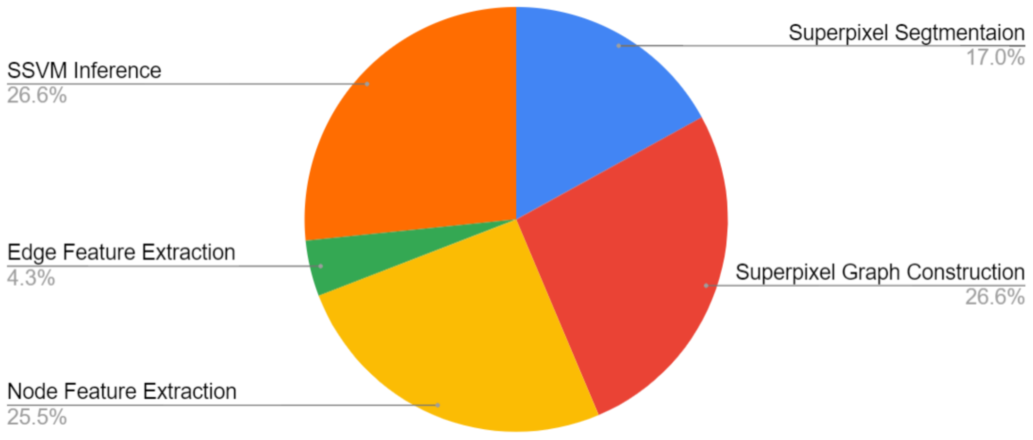


Figure 5.9: The pie-chart representing the distribution of the execution time consumed by the different building blocks of the proposed approach for identifying gaps in a shelf image

Notes on Inference Time The proposed algorithm is implemented in python and tested in a computing system with the following specifications: 96GB RAM, Intel Core i9-9820X CPU 3.30GHz$\times$20 and 24GB TITAN RTX GPU. The modules of the proposed approach involved during inference are: (a) Superpixel Segmentation (Section 5.3.1), (b) Superpixel Graph Construction (Section 5.3.2), (c) Node Feature Extraction (Section 5.3.4), (d) Edge Feature Extraction (Section 5.3.5) and (e) SSVM Inference (Section 5.3.6). For identifying the gaps in a (test) shelf image, the time consumed by each of these modules of our scheme is presented using a pie-chart in Fig. 5.9. The total time taken by the un-optimized code of the proposed approach is $\sim$0.93 seconds. Among all the modules, as expected, Node Feature Extraction consumes the highest time due to the Graph Convolution process explained in Section 5.3.4. Further, the CPU implementations of graph manipulation (Superpixel Segmentation and Superpixel Graph Construction) and SSVM inference process have increased the overall execution time. A shelf image is analyzed in less than a second with the un-optimized implementation of our algorithm. However, the (deep learning based) competing methods take $\sim$0.45 seconds for identifying the gaps in a (test) shelf image. All the competitive methods are end-to-end deep learning based methods, which are entirely implemented in GPU. On the contrary, our current implementation of the proposed approach involves CPU alongside the GPU, which essentially increases the test time. Our analysis finds that the test time of our scheme should be close to that of the competitors if we implement the graph manipulation and SSVM inference in the GPU. However, with this fraction of seconds increase of test time w.r.t. the competing approaches, the proposed scheme yields a significantly better performances in almost all the cases. Next we carry out the experiment for choosing the number of superpixels for creating the superpixel graph.

Choice of Number of Superpixels (N) An example shelf image I, segmented into four superpixels is

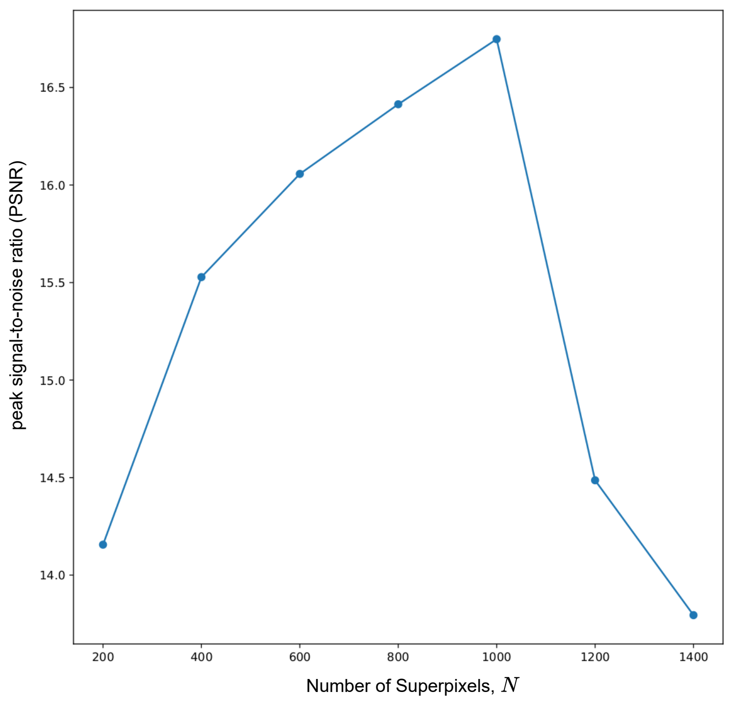


Figure 5.10: Peak signal-to-noise ratio (PSNR) (in dB) values between I_{gt} and B for different numbers of superpixels N generated by SLIC superpixel segmentation algorithm.

shown in Figure 5.4(a). The pixel-level ground truth of I is I_{gt} as shown in Figure 5.5(b). In order to train the SSVM, we label each superpixel by majority voting of white or black pixels to create the structured ground truth mask B as explained in Section 5.3.3 and as shown in Figure 5.5.

The choice of number of superpixels (N) should be such that the (pixel-wise) difference between the pixel-level ground truth mask $I_{gt}^{(\mathrm{k})}$ and structured ground truth mask $\mathrm{B}^{(\mathrm{k})}$ is minimum for the training image $I^{(\mathrm{k})}$. That is, Figures 5.5(b) and 5.5(c) become almost identical. In order to ensure that, we choose N in a way such that the peak signal-to-noise ratio (PSNR) [Welstead, 1999] (in db) between $I_{gt}^{(\mathrm{k})}$ and $\mathrm{B}^{(\mathrm{k})}$ is maximum. Thus, we compute the mean PSNR for the images in the training set of the WM dataset varying N from 200 to 1400 in intervals of 200. The mean PSNR for various N is plotted in Figure 5.10. It can be seen that till about $N = 1000$, the PSNR increases. This means that more the granularity in segmentation, more (pixel-wise) similar are $I_{gt}^{(\mathrm{k})}$ and $\mathrm{B}^{(\mathrm{k})}$. However, for $\mathrm{N} > 1000$, PSNR starts to fall due to the inconsistent superpixel boundaries determined by the SLIC algorithm. Therefore, we set $\mathrm{N} = 1000$ in our implementation. Next we discuss the importance of our method in context of retail stores.

5.4.3 Suitability of the Proposal for Retail Store Environment

The deep learning based approaches usually require enormous training data. Limited training images result into over-fitting of the model during training and hence poor generalized performance. Due to availability of limited training data for the application under consideration, we have used structural support vector machine that learns lesser number of parameters ($2 \times$ number of node features $+ 4 \times$ number of edge features) compared to competing deep learning based method.

The deep learning models, that we have utilized in this work, are GCN to extract the features of the superpixels and a Siamese network to extract the features of a pair of superpixels. In order to train these networks, a minimal set of labeled data (i.e., annotated shelf images) is good enough.

Assume there are 30 self images in the train-set, each of which has 1000 superpixels as set in our implementation. In that case, the training data, which is used for training the node feature extractor (GCN), contain 30×1000 samples. For training the edge feature extractor (Siamese network), we have $30 \times$ number of edges in each SG (obviously more than 1000) training samples.

Such a training scheme is large enough to train the proposed GCN or Siamese network. On the contrary, all other deep learning based segmentation methods considered in our comparative study (see Table 5.1), require to train millions of parameters. As a result, the proposed scheme outperforms all these methods as evident in Table 5.1. Hence, in the context of retail store with limited number of training (shelf) images, given that the product display plan in supermarkets changes frequently, the proposed scheme is expected to be a better choice for an application like identification of empty spaces. Next the chapter is summarized.

5.5 Summary

This chapter looks into designing a pragmatic solution for automatically identifying gaps in the images of shelves/racks captured in supermarkets. We have posed this gap identification problem as an image segmentation problem. Capturing a large number of training images of shelves and annotating them is next to impossible particularly with the fast changing line of products. Keeping this in mind, the proposed method is built in a way such that it performs significantly well when it is trained with a minimal set of images unlike widely used deep learning based image segmentation techniques.

The shelf image is first over-segmented into a number of superpixels to construct a graph of superpixels (SG). Subsequently, a graph convolutional network and a Siamese network are built to process the SG. The method presented in this chapter uses graph convolutional network (GCN) for feature extraction of the superpixels (to be used as node feature of SG) while Siamese network architecture (SNA) captures the similarity of the neighbouring superpixels in a feature embedding (to be used as edge feature of the

SG). Finally, an SSVM based inference model is formulated to segment the empty and non-empty regions from the shelf image. Utilizing GCN and SNA to obtain the node and edge features of a superpixel graph for training SSVM has never been attempted. We have shown their importance in classification of gaps using SSVM. We consider this to be the key contribution of our proposal.

In order to validate our proposal and other competing approaches, we manually annotate the images of shelves labeling empty and non-empty regions. We release a part of these annotated data at https://github.com/gapDetection/gapDetectionDatasets for the publicly available benchmark image datasets of retail products, namely Grocery Products, WebMarket and GroZi. We believe that the release of this dataset for gap detection is an important milestone for application-driven computer vision research community. The results obtained in this chapter indicate that the problem needs much more attention from the community.

The previous three Chapters 2, 3, and 4 present our attempts for the automatic identification of products in the retail stores while this chapter proposes the first known attempt for segmentation of gaps/non-gaps in the images of shelves of supermarkets. Finally, in the next chapter, we conclude the thesis.

CHAPTER 6

Conclusions

In this thesis, we have looked into two important aspects related to the products displayed on the racks of supermarkets:

(a) identification and localization of the products, and

(b) identification of empty spaces on the shelves for automatic planogram compliance.

The proposed annotation-free machine vision system has three modules in sequence:

(i) generation

(ii) classification, and finally

(iii) non-maximal suppression of region proposals.

The region proposals originate from ideal marketing images of the products. While recognizing the products, we take care of minor orientations of products on the shelves and variation of store level illumination to certain extent. Finer distinctions between product packages and products vertically stacked on the shelves, are also identified.

To identify empty spaces on the shelves automatically, we introduce a graph-based image segmentation approach. This is the first known attempt for identification of empty spaces on the shelves. In this context, we release annotated data [Santra et al., 2021a] for some of the images from three benchmark datasets detailed in Appendix B. This thesis introduces a few deep neural network models for some specific tasks related to the problem under discussion. Further, we have utilized a few state-of-the-art deep neural architectures for solving the problem. All these deep learning architectures and their objectives are tabulated in Table 6.1.

Based on the research conducted in this thesis, we reach to the following conclusions:

Table 6.1: The deep learning models/architectures introduced or used in this thesis

Chapters	Deep Learning Architectures	Objectives	Comments
Chapter 2, Chapter 3, Chapter 4	ResNet-101 [He et al., 2016]	Classification of products	Reused
Chapter 3, Chapter 4	RC-Net	Object-level or coarse-grained classification of products	Newly introduced
	conv-LSTM [Xingjian et al., 2015]	Classification of sequence of image patches for part-level or fine-grained classification of products	Introduced a new architecture of conv-LSTM
Chapter 5	VGG-19 [Simonyan and Zisserman, 2015]	Classification of a superpixel as a gap or non-gap	Reused
	GCN [Kipf and Welling, 2016]	Classification of superpixel associating neighboring superpixels as a gap or non-gap	Introduced a new architecture of GCN
	Siamese Network [Koch et al., 2015]	Classification of a pair of superpixels as a dissimilar or similar superpixels	Introduced a new architecture of Siamese Network

- The proposed exemplar-driven region proposal algorithm has proved its efficacy by generating fewer number of proposals compared to an exemplar-independent region proposal scheme. Further, our region proposal algorithm generates proposals around the products with the help of elegant key-point matching based approach. As a result, our scheme generates very few false proposals away from the products. In contrast, exemplar-independent approaches are responsible for higher number of false positives.

- In generating exemplar-driven region proposals, we estimate scale between product templates and rack. The quality of scale estimation depends on the availability of physical dimensions of product templates.

- Automatic identification of stacked products is always a challenge. We manage region proposals using a directed acyclic graph compensating classification score of a region proposal with the quality of geometric fit of the proposals with the templates. Selection of region proposals using graph based approaches shows improved result compared to greedy selection of region proposals.

- The proposed fine-grained classifier has looked at both object-level and part-level cues. As a result, this bi-level approach enables us to efficiently differentiate the fine-grained products. Moreover, our object-level classifier RC-Net (due to its enhanced generalization ability) resolves the effect of illumination difference between training and test images. When we embed this in our product detection system, we see a noticeable improvement in product detection performance.

- In this thesis, we have developed an end-to-end system for automatic product identification. Each of the three solutions ERP, FGC, and G-NMS contribute sufficiently well in the final result. In particular, ERP plays the lead role in achieving significantly better results for our system than other competing approaches.

- Identification of empty spaces on the shelves is an interesting image segmentation problem. We have shown that the structural support vector machine is a viable option for achieving desired image segmentation with the limited number of labelled data. Both the graph convolutional and Siamese networks are integrated to obtain attractive empty space segmentation results.

Next we list a few challenges as possible future directions of research.

6.1 Future Directions of Research

We conceive the following set of challenges based on the work done in this thesis on product identification system.

- Simultaneous detection of products and empty spaces in between could be an interesting research problem. Fast changing line of products and unavailability of annotated shelf images restrict use of efficient deep learning approaches. However, it would be an interesting challenge to recognize products and empty spaces in one go using a minimal set of annotated images of racks/shelves (as we did for identification of empty spaces). This challenge may be addressed using techniques of semantic segmentation.

- Estimation of scale between the product templates and rack needs further attention. In many occasions, the availability of physical dimensions of a product template is a serious restriction. In that case, we may use involved context information. A potential direction of research could be exploitation of texture based context information.

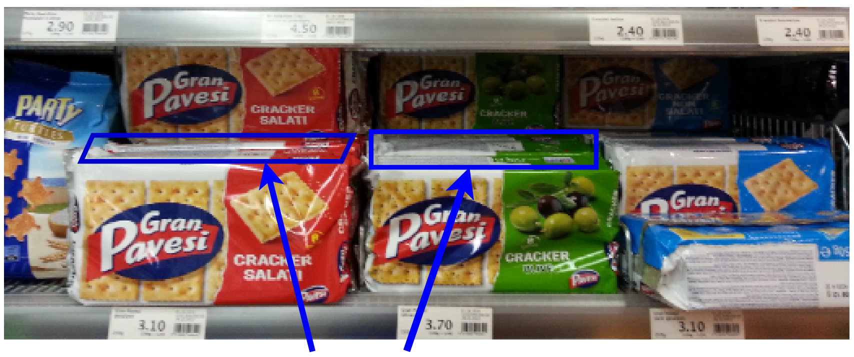


Figure 6.1: Empty spaces on stacked products marked by blue quadrilateral and arrow

- The proposed G-NMS runs in quadratic time. The improvement in time complexity of the G-NMS can be explored from different perspectives.
- Product packages change quickly. Number of products increases rapidly. Frequent retraining of machine learning system is a serious bottleneck. These issues present interesting research problems in context of product identification challenge. A related difficult research problem is novelty detection when a very similar but new product needs to be identified as novelty compared to the existing product templates.
- The structural support vector machine along with graph convolutional and Siamese networks have shown outstanding performance in identifying empty spaces using minimal set of labelled images. However, these three machine learning models are trained separately in the proposal. An integrated training framework for these three models in an end-to-end setup could be an interesting theoretical research direction.
- The empty space identification in shelves, will always remain a difficult research problem due to variation in definition of empty spaces on the shelves. Assume, a product is picked from the top of a vertical stack of products. This results in an empty space (for example, see the regions covered by blue quadrilateral and marked by blue arrow in Figure 6.1). Identification of such an empty space is indeed difficult. One possible way to deal with this situation is to look at the depth of the stack from temporal image frames. In such a scenario, depth estimation for stacked products on the shelf images and continuous monitoring of sequences of images could be an exciting image processing problem that may be explored.
- The application-driven research project that we have demonstrated in this thesis, has a number of system management issues. Hardware and software management for real time implementation, challenges in mapping and scaling of algorithm in GPU, and implementation strategy partially in edge or hand held device and balance in backbone or in cloud, all these are important future direction of research for this application.

Overall, these challenges suggest that the retail product identification system will continue to be an exciting problem for research and development in computer vision with sufficient room for improvement in the years to come.

APPENDIX A

Evaluation Indicators for Assessing Product Detection Performance

In [Santra and Mukherjee, 2019], we notice that different evaluation indicators are used in different state-of-the-art methods (see Table 1.3) for validating the solutions. Keeping retail context in mind, in this thesis, the efficiency of the methods are evaluated by calculating F_1 score for measuring the performance of detecting products on the shelves of the supermarkets. This score implicitly encapsulates the standard measures, *recall*, *precision*, or *intersection-over-union* (IoU). For the problem under discussion, each product sitting on a rack is defined using a rectangular bounding box and the class label of the product. Thus TP (true positives), FN (false negatives), and FP (false positives) are defined for each labeled product displayed on the rack. Subsequently, the F_1 score is calculated for each rack in the test set.

Let a product $\mathbb{D}_t$ (ground truth) is present on the rack image I. Figure A.1 shows an example rack I displaying two products $\mathbb{D}_1$ and $\mathbb{D}_2$ (see red boxes), for which we obtain three detected bounding boxes (see green, blue, and yellow boxes detected using an algorithm). Given this, we find the counts of TP, FP, and FN for the rack image I as follows.

If the center of any detected bounding box (i.e., detected by an algorithm) lies within $\mathbb{D}_t$ on the rack and the label of the detected box is same as the label of $\mathbb{D}_t$, the count of TP of the rack I is increased by 1. For example, the detected green box $\mathbb{D}_{\mathbb{1}}$ for the red product $\mathbb{D}_{\mathbb{1}}$ is the TP in Figure A.1.

If the center of the detection lies within $\mathbb{D}_t$ on the rack but its label is different from that of $\mathbb{D}_t$, the count of FP of the rack I is increased by 1. As an example, the detected blue box $\mathbb{D}_3$ for the red colored product $\mathbb{D}_{\mathbb{1}}$ is the FP in Figure A.1. If the center of a detection does not lie within any ground truth

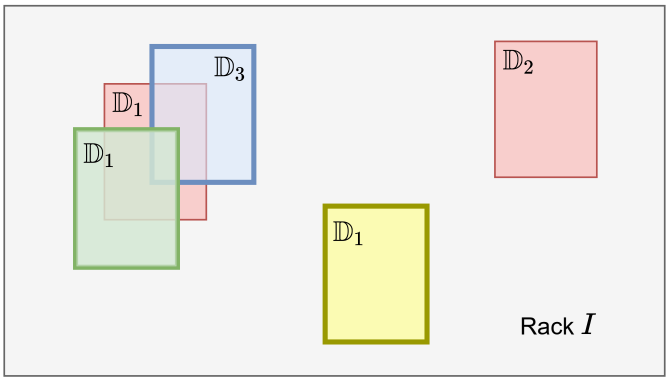


Figure A.1: Example rack image displaying two products $\mathbb{D}_1$ and $\mathbb{D}_2$ (see the red boxes), for which the green detected box labeled $\mathbb{D}_1$ is TP, the blue detected box labeled $\mathbb{D}_3$ is FP, and the yellow detected box labeled $\mathbb{D}_1$ is also an FP.

product in I, the count of FP of the rack I is increased by 1, like the detected yellow box $\mathbb{D}_{\mathbb{1}}$ in Figure A.1.

If the center of any detection does not lie within the $\mathbb{D}_t$ in the rack (for example, see the red colored product $\mathbb{D}_2$ in Figure A.1), the count of FN of the rack I is increased by 1. Given this, for the rack I, the F_1 score is derived as

$$F_1 \text{ score} = \frac{2(recall \times precision)}{recall + precision}, \tag{A.1}$$

where

$$recall = \frac{\text{TP}}{\text{TP} + \text{FN}}, \text{ and } precision = \frac{\text{TP}}{\text{TP} + \text{FP}}.$$

APPENDIX B

Datasets of Retail Products

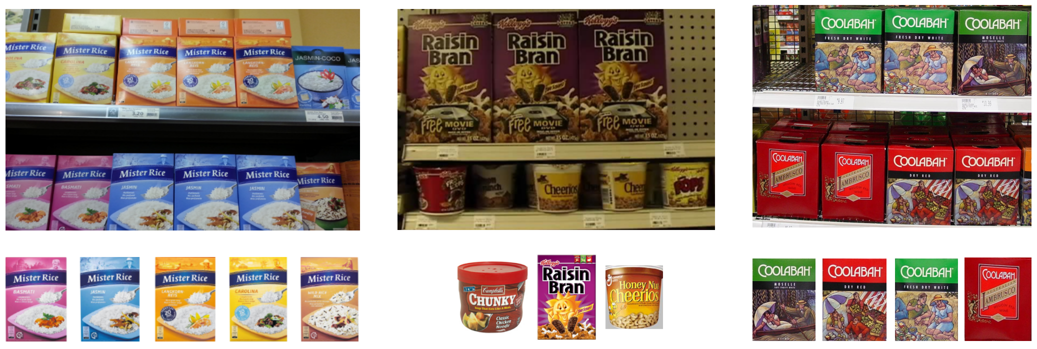
B.1 In-house Dataset

We capture 457 rack images of 352 products in various supermarkets to create **In-house** dataset. This collection of rack images display many fine-grained products. We capture one template image for each product in a controlled scenario. The products belong to six larger classes of products such as *breakfast cereals* (BC) (72 products and 151 racks), *deodorant* (DEO) (55 products and 100 racks), *lip care* (LC) (20 products and 80 racks), *oral care* (OC) (51 products and 30 racks), *personal wash* (PW) (82 products and 36 racks) and *mixed* (MIX) (72 products and 60 racks). Note that, *mixed* or MIX category refers to a collection of products from other five categories of the dataset. For creating ground truth, all the racks are manually annotated by labeling the products using tight rectangular bounding boxes. In addition, we have used following three benchmark datasets.

B.2 Benchmark Datasets

In this thesis, we have evaluated various approaches on three benchmark datasets: Grocery Products[1] [George and Floerkemeier, 2014], WebMarket[2] [Zhang et al., 2007], and GroZi[3] [Merler et al., 2007], which are detailed in the following sections.

[1]http://people.inf.ethz.ch/mageorge/ accessed as on Dec, 2017
[2]http://yuhang.rsise.anu.edu.au/ accessed as on Jul, 2021
[3]http://grozi.calit2.net/grozi.html accessed as on Jul, 2021



(a) (b) (c)

Figure B.1: Examples of rack (top-row) and product templates (bottom-row) from (a) Grocery Products, (b) GroZi-120, and (c) WebMarket datasets

B.2.1 Grocery Products

The Grocery Products (GP) dataset consists of 680 rack images that exhibit 3235 products. The products are collected from 27 categories. The rack images display 6 to 30 number of products. The product templates, which are imaged in a controlled environment, are downloaded from the web. While the rack images are clicked in the supermarket environment from various viewing angles with non-identical magnification levels and lighting conditions. The dataset also presents many similar yet non-identical (i.e., fine-grained) products (see Figure B.1(a)) which are displayed on the rack images. The classification of fine-grained products is a challenge for this dataset. The dataset also includes ground truth for all the rack images indicating similar group of products with tight bounding boxes.

B.2.2 WebMarket

The WebMarket (WM) dataset includes 3153 rack images captured from 18 shelves in a supermarket. The dataset contains template images of 100 products which are present only in 402 (out of 3153) rack images. There exists three instances for each of the products in the dataset. This dataset also provides a few fine-grained products. The rack images are clicked when the products are on the shelf. On the contrary, the products are imaged off the shelf. So scale, pose and illumination are not identical for rack and product images. Examples of rack and a few products are shown in Figure B.1(c). The annotations, for the bounding boxes of the products in the rack images, are not bundled with the dataset. We have manually annotated the racks for evaluation of the methods.

B.2.3 GroZi

The GroZi (GZ) dataset provides 29 videos of racks, which display 120 products captured in a supermarket. The frames after each 5 consecutive frames (e.g., 1st, 6th, 11th, ... frames) are extracted from the videos and annotation for these frames are included in the dataset. The dataset distributes 2 to 14 template images for each product. In this dataset, each product is displayed in 14 to 814 rack images. The dataset also provides (separately) the products cropped from the racks using the annotations for the bounding boxes of the products in the rack images. Most of the products in rack differ from the templates as shown in Figure B.1(b). This poses additional challenge for the dataset.

APPENDIX C

Proof of the Theorem 3.1

C.1 Assumptions

The assumptions in proving the Theorem 3.1 are listed below.

Assumption C.1. *Predicting class labels of the products is our primary target while reconstruction of the input product images is our secondary (or auxiliary) target. The spaces containing product images and primary targets are bounded. In other words, the input and primary target must satisfy $\|\mathbf{M}\|_F \leq B_{\mathbf{M}}$ and $\|\mathbf{y}_p\|_2 \leq B_{\mathbf{y}}$ respectively. Similarly the spaces $\mathscr{W}_p, \mathscr{W}_r$, and $\mathscr{K}$, where the learnable parameters $\mathbf{W}_p, \mathbf{W}_r$, and $\mathbf{K}$ exist, respectively, are also bounded i.e.*

$$\mathscr{W}_p = \{\mathbf{W}_p \in \mathbb{R}^{m \times k_h k_w f} : \|\mathbf{W}_p\|_F \leq B_{\mathbf{W}_p}\},$$
$$\mathscr{W}_r = \{\mathbf{W}_r \in \mathbb{R}^{c_{in} \times k_h k_w f} : \|\mathbf{W}_r\|_F \leq B_{\mathbf{W}_r}\},$$
$$\mathscr{K} = \{\mathbf{K} \in \mathbb{R}^{f \times k_h k_w c_{in}} : \|\mathbf{K}\|_F \leq B_{\mathbf{K}}\},$$

where $B_{\mathbf{M}}, B_{\mathbf{y}}, B_{\mathbf{K}}, B_{\mathbf{W}_p}, B_{\mathbf{W}_r}$ are the positive constants and $\|\cdot\|_F$ represents the Frobenius norm (which is also referred to as ℓ^2 norm) of a matrix. In rest of the thesis, $\|\cdot\|_F$ and $\|\cdot\|_2$ are interchangeably used for denoting the ℓ^2 norm of a matrix.

Assumption C.2. *The reconstruction error $\mathfrak{L}_r(\cdot, \mathbf{y}_r)$ is Lipschitz i.e., $\mathfrak{L}_r(\cdot, \mathbf{y}_r)$ is σ_r-admissible. Formally, for any two possible predictions $\hat{\mathbf{y}}_r$ and $\hat{\mathbf{y}}_r'$ of $\mathbf{y}_r$ (i.e. the input product image X), there exists a* *real positive constant σ_r such that*

$$|\mathfrak{L}_r(\hat{\mathbf{y}}_r, \mathbf{y}_r) - \mathfrak{L}_r(\hat{\mathbf{y}}_r', \mathbf{y}_r)| \leq \sigma_r \|\hat{\mathbf{y}}_r - \hat{\mathbf{y}}_r'\|_2.$$

Assumption C.3. *The reconstruction error $\mathfrak{L}_r(\cdot, \mathbf{y}_r)$ is also g-strongly-convex i.e. for a real positive constant g,*

$$\langle \hat{\mathbf{y}}_r - \hat{\mathbf{y}}_r', \nabla \mathfrak{L}_r(\hat{\mathbf{y}}_r, \mathbf{y}_r) - \nabla \mathfrak{L}_r(\hat{\mathbf{y}}_r', \mathbf{y}_r) \rangle \geq g \|\hat{\mathbf{y}}_r - \hat{\mathbf{y}}_r'\|_2^2.$$

Assumption C.4. *There must be a bound for the growth of the primary loss $\mathfrak{L}_p(., \mathbf{y}_p)$ i.e., for some positive constant $\sigma_p > 0$ and for any $\mathbf{K}, \mathbf{K}_t \in \mathscr{K}$,*

$$|\mathfrak{L}_p(\mathbf{W}_p \mathbf{K}_t \mathbf{M}, \mathbf{y}_p) - \mathfrak{L}_p(\mathbf{W}_p \mathbf{K} \mathbf{M}, \mathbf{y}_p)| \leq \sigma_p \|\mathbf{W}_r (\mathbf{K}_t - \mathbf{K}) \mathbf{M}\|_2.$$

Assumption C.5. *We must have a representative set $\mathtt{E}$ of the feature maps of the training images. Formally, there exists a subset*

$$\begin{aligned} \mathtt{E} &= \{\mathbf{E}^{(1)}, \mathbf{E}^{(2)}, \cdots, \mathbf{E}^{(n')}\} \subset \{\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \cdots, \mathbf{X}^{(n)}\} \\ &\equiv \{\mathbf{M}^{(\mathbf{E}^{(1)})}, \mathbf{M}^{(\mathbf{E}^{(2)})}, \cdots, \mathbf{M}^{(\mathbf{E}^{(n')})}\} \subset \{\mathbf{M}^{(1)}, \mathbf{M}^{(2)}, \cdots, \mathbf{M}^{(n)}\}, \end{aligned}$$

such that the feature maps of any product image $\mathbf{M}$ that belongs to the training image set, can be reconstructed by $\mathtt{E}$ with a small error i.e., $\mathbf{M} = \sum_{i=1}^{n'} \alpha^{(i)} \mathbf{M}^{(\mathbf{E}^{(i)})} + \omega$, where $\alpha^{(i)} \in \mathbb{R}$, $\sum_{i=1}^{n'} {\alpha^{(i)}}^2 \leq r$, $\|\omega\|_2 \leq \frac{\epsilon}{n}$, ϵ is a positive constant.

Assumption C.6. *The representative set $\mathtt{E}$ of the training images also have to be representative in terms of reconstruction error i.e., the average reconstruction error of the images in $\mathtt{E}$ needs to be upper bounded by some constant factor of the average reconstruction error. Consider any two datasets $\mathtt{D}_1$ and $\mathtt{D}_2$. The dataset $\mathtt{D}_2$ is formed by replacing t^{th} image in $\mathtt{D}_1$ with a random new instance. Let $\mathbf{K}$ and $\mathbf{K}_t$ be the optimal forward models trained with $\mathtt{D}_1$ and $\mathtt{D}_2$ respectively. Let $N(\cdot)$ be the average reconstruction error for the model $\mathbf{K}$, without considering the image that is replaced,*

$$N(\mathbf{K}) = \frac{1}{n} \sum_{i=1, i \neq t}^{n} \mathfrak{L}_r(\mathbf{W}_r \mathbf{K} \mathbf{M}^{(i)}, \mathbf{y}_r^{(i)}),$$

and similarly the average reconstruction error for the representative images

$$N_E(\mathbf{K}) = \frac{1}{n'} \sum_{i=1}^{n'} \mathfrak{L}_r(\mathbf{W}_r \mathbf{K} \mathbf{M}^{(\mathbf{E}^{(i)})}, \mathbf{y}_r^{(\mathbf{E}^{(i)})}),$$

where $\mathbf{y}_r^{(\mathbf{E}^{(i)})}$ is the reconstruction target for the representative image $\mathbf{M}^{(\mathbf{E}^{(i)})} \in \mathtt{E}$. Then, for any small

$\alpha > 0$, there exists $a > 0$ such that

$$\begin{aligned}&[N_E(\mathbf{K}) - N_E((1-\alpha)\mathbf{K} + \alpha\mathbf{K}_t)] + [N_E(\mathbf{K}_t) - N_E((1-\alpha)\mathbf{K}_t + \alpha\mathbf{K})]\\ &\leq a[N(\mathbf{K}) - N((1-\alpha)\mathbf{K} + \alpha\mathbf{K}_t)] + [N(\mathbf{K}_t) - N((1-\alpha)\mathbf{K}_t + \alpha\mathbf{K})].\end{aligned}$$

This ensures that the increase or decrease in error for the forward models $\mathbf{K}_t$ and $\mathbf{K}$ are similar for $N(\cdot)$ and $N_E(\cdot)$.

C.2 Proof of the Theorem 1

We prove Theorem 3.1 under the Assumptions C.1 to C.6 as explained in Section 3.3.1 for the conv operation defined in Section 3.3.1 [Le et al., 2018].

Proof. Assumption C.4 states that

$$|\mathfrak{L}_p(\mathbf{W}_p\mathbf{K}_t\mathbf{M}, \mathbf{y}_p) - \mathfrak{L}_p(\mathbf{W}_p\mathbf{K}\mathbf{M}, \mathbf{y}_p)| \leq \sigma_p \|\mathbf{W}_r(\mathbf{K}_t - \mathbf{K})\mathbf{M}\|_2. \tag{C.1}$$

For any arbitrary data point $(\mathbf{X}, \mathbf{y})$, in order to bounding the LHS of (C.1), we first have to determine the upper bound of LHS of (C.1) in terms of the representative data points, $\sum_{i=1}^{n'} \|\mathbf{W}_r(\mathbf{K}_t - \mathbf{K})\mathbf{M}^{(\mathbf{E}^{(i)})}\|_2$. Note that, $\mathbf{M}$ and $\mathbf{M}^{(\mathbf{E}^{(i)})}$ are the matrix representations of the product images $\mathbf{X}$ and $\mathbf{E}^{(i)}$ respectively. This is the first part of the proof which we derive next.

Part 1: Upper bound of LHS of (C.1) in terms of representative points. From Assumption C.5, any product image $\mathbf{X}$ (or $\mathbf{M}$) can be represented as $\sum_{i=1}^{n'} \alpha^{(i)}\mathbf{M}^{(\mathbf{E}^{(i)})} + \eta$ $(\equiv \sum_{i=1}^{n'} \alpha^{(i)}\mathbf{E}^{(i)} + \eta)$. Then

$$\|\mathbf{W}_r(\mathbf{K}_t - \mathbf{K})\mathbf{M}\|_2 = \left\| \sum_{i=1}^{n'} \alpha^{(i)} \mathbf{W}_r(\mathbf{K}_t - \mathbf{K})\mathbf{M}^{(\mathbf{E}^{(i)})} + \mathbf{W}_r(\mathbf{K}_t - \mathbf{K})\eta \right\|_2. \tag{C.2}$$

Using triangle inequality, $\|C_1 + C_2\| \leq \|C_1\| + \|C_2\|$, we further obtain,

$$\|\mathbf{W}_r(\mathbf{K}_t - \mathbf{K})\mathbf{M}\|_2 \leq \left\| \sum_{i=1}^{n'} \alpha^{(i)} \mathbf{W}_r(\mathbf{K}_t - \mathbf{K})\mathbf{M}^{(\mathbf{E}^{(i)})} \right\|_2 + \|\mathbf{W}_r(\mathbf{K}_t - \mathbf{K})\eta\|_2.$$

Now using Cauchy-Schwarz inequality and the inequality, $\|C_1C_2\| \leq \|C_1\|\|C_2\|$, we get

$$\|\mathbf{W}_r(\mathbf{K}_t - \mathbf{K})\mathbf{M}\|_2 \leq \sqrt{\sum_{i=1}^{n'} \alpha^{(i)^2}} \left\| \sqrt{\sum_{i=1}^{n'} \left(\mathbf{W}_r(\mathbf{K}_t - \mathbf{K})\mathbf{M}^{(\mathbf{E}^{(i)})}\right)^2} \right\|_2 + \|\mathbf{W}_r\|_2 \|(\mathbf{K}_t - \mathbf{K})\|_2 \|\eta\|_2. \tag{C.3}$$

From Assumption C.5, we get, $\sum_{i=1}^{n'} \alpha^{(i)^2} \leq r$ and $\|\eta\|_2 \leq \frac{\epsilon}{n}$ while from Assumption C.1, we obtain,

$\|\mathbf{W}_r\|_2 \le B_{\mathbf{W}_r}$ and $\|\mathbf{K}\|_2 \le B_{\mathbf{K}}$, then

$$\|\mathbf{K}_t - \mathbf{K}\|_2 = \|\mathbf{K}_t + (-\mathbf{K})\|_2 \le \|\mathbf{K}_t\|_2 + \|-\mathbf{K}\|_2 \le B_{\mathbf{K}} + B_{\mathbf{K}} = 2B_{\mathbf{K}}.$$

Therefore, (C.3) can be written as:

$$\begin{aligned}\|\mathbf{W}_r(\mathbf{K}_t - \mathbf{K})\mathbf{M}\|_2 &\le \sqrt{r}\sqrt{\sum_{i=1}^{n'} \left\|\mathbf{W}_r(\mathbf{K}_t - \mathbf{K})\mathbf{M}^{(\mathbf{E}^{(i)})}\right\|_2^2} + \frac{2B_{\mathbf{W}_r}B_{\mathbf{K}}\epsilon}{n} \\ &\le \sqrt{r}\Gamma + \frac{2B_{\mathbf{W}_r}B_{\mathbf{K}}\epsilon}{n}, \end{aligned} \tag{C.4}$$

where assume, $\Gamma = \sqrt{\sum_{i=1}^{n'} \|\mathbf{W}_r(\mathbf{K}_t - \mathbf{K})\mathbf{M}^{(\mathbf{E}^{(i)})}\|_2^2}$. Now, we aim to bound this Γ which is the second part of our proof.

Part 2: Bounding $\Gamma = \sqrt{\sum_{i=1}^{n'} \|\mathbf{W}_r(\mathbf{K}_t - \mathbf{K})\mathbf{M}^{(\mathbf{E}^{(i)})}\|_2^2}$. Assume, for the overall loss $\mathfrak{L}(\mathbf{K})$ for the forward model $\mathbf{K}$, the Bregman divergence is represented by $D_{\mathfrak{L}}(\mathbf{K}_t\|\mathbf{K})$ and defined as:

$$D_{\mathfrak{L}}(\mathbf{K}_t\|\mathbf{K}) = \mathfrak{L}(\mathbf{K}_t) - \mathfrak{L}(\mathbf{K}) - \langle \mathbf{K}_t - \mathbf{K}, \nabla\mathfrak{L}(\mathbf{K})\rangle, \tag{C.5}$$

where the inner product notation refers to summation of element-wise product for the matrices. Now, we use following three bounds:

$$D_N(\mathbf{K}_t\|\mathbf{K}) \le D_{\mathfrak{L}}(\mathbf{K_t}\|\mathbf{K}), \tag{C.6}$$

$$D_N(\mathbf{K}\|\mathbf{K}_t) \le D_{\mathfrak{L}_t}(\mathbf{K}\|\mathbf{K}_t), \tag{C.7}$$

$$D_{N_E}(\mathbf{K}_t\|\mathbf{K}) + D_{N_E}(\mathbf{K}\|\mathbf{K}_t) \le a[D_N(\mathbf{K}_t\|\mathbf{K}) + D_N(\mathbf{K}\|\mathbf{K}_t)]. \tag{C.8}$$

Before utilizing the above three inequalities, we need to first prove above three inequalities.

Determining the inequalities (C.6) and (C.7): Since the sum of two strictly convex functions for N is a strict subset of sum of strictly convex functions for $\mathfrak{L}$, $\mathfrak{L} - N$ is strictly convex. Hence $\mathfrak{L} - N$ provides a valid potential for the Bregman divergence. Then, applying the properties of Bregman divergence, we obtain

$$0 \le D_{\mathfrak{L}-N}(\mathbf{K}_t\|\mathbf{K}) \implies D_N(\mathbf{K}_t\|\mathbf{K}) \le D_{\mathfrak{L}}(\mathbf{K}_t\|\mathbf{K}).$$

Hence the inequality (C.6) is proved and using the similar reasoning, the inequality (C.7) can also be proved that

$$D_N(\mathbf{K}\|\mathbf{K}_t) \le D_{\mathfrak{L}_t}(\mathbf{K}\|\mathbf{K}_t).$$

Determining the inequality (C.8): The LHS $D_{N_E}(\mathbf{K}_t\|\mathbf{K}) + D_{N_E}(\mathbf{K}\|\mathbf{K}_t)$ of (C.8) can be simplified using the Bregman divergence (as defined in (C.5)) for N_E (or N) as:

$$D_{N_E}(\mathbf{K}_t\|\mathbf{K}) + D_{N_E}(\mathbf{K}\|\mathbf{K}_t) = \big\langle \mathbf{K}_t - \mathbf{K}, \nabla N_E(\mathbf{K}_t) - \nabla N_E(\mathbf{K})\big\rangle. \tag{C.9}$$

Now, by the definition of directional derivative, we obtain

$$\big\langle \mathbf{K}_t - \mathbf{K}, \nabla N_E(\mathbf{K})\big\rangle = \lim_{\alpha\to 0} \frac{N_E\Big((1-\alpha)\mathbf{K} - \alpha\mathbf{K}_t\Big) - N_E(\mathbf{K})}{\alpha} \tag{C.10}$$

Using property of the inner product and since both the limit exists, we get

$$\begin{aligned}
\big\langle \mathbf{K}_t - \mathbf{K}, \nabla N_E(\mathbf{K}_t) - \nabla N_E(\mathbf{K})\big\rangle &= -\big\langle \mathbf{K}_t - \mathbf{K}, \nabla N_E(\mathbf{K})\big\rangle - \big\langle \mathbf{K} - \mathbf{K}_t, \nabla N_E(\mathbf{K}_t)\big\rangle \\
&= \lim_{\alpha\to 0^+}\left[\frac{N_E(\mathbf{K}) - N_E\Big((1-\alpha)\mathbf{K} + \alpha\mathbf{K}_t\Big)}{\alpha}\right. \\
&\qquad \left. + \frac{N_E(\mathbf{K}_t) - N_E\Big((1-\alpha)\mathbf{K}_t + \alpha\mathbf{K}\Big)}{\alpha}\right] \\
&\le \lim_{\alpha\to 0^+} a\left[\frac{N(\mathbf{K}) - N\Big((1-\alpha)\mathbf{K} + \alpha\mathbf{K}_t\Big)}{\alpha}\right. \\
&\qquad \left. + \frac{N(\mathbf{K}_t) - N\Big((1-\alpha)\mathbf{K}_t + \alpha\mathbf{K}\Big)}{\alpha}\right] \text{(Assumption C.6)} \\
&= -\lim_{\alpha\to 0} a\left[\frac{N\Big((1-\alpha)\mathbf{K} + \alpha\mathbf{K}_t\Big) - N(\mathbf{K})}{\alpha}\right] \\
&\qquad - \lim_{\alpha\to 0} a\left[\frac{N\Big((1-\alpha)\mathbf{K}_t + \alpha\mathbf{K}\Big) - N(\mathbf{K}_t)}{\alpha}\right] \\
&= -a\big\langle \mathbf{K}_t - \mathbf{K}, \nabla N(\mathbf{K})\big\rangle - a\big\langle \mathbf{K} - \mathbf{K}_t, \nabla N(\mathbf{K}_t)\big\rangle \text{(using (C.10))} \\
&= a\big[D_N(\mathbf{K}_t\|\mathbf{K}) + D_N(\mathbf{K}\|\mathbf{K}_t)\big].\text{(using (C.9))}
\end{aligned}$$

Hence, (C.9) now implies

$$D_{N_E}(\mathbf{K}_t\|\mathbf{K}) + D_{N_E}(\mathbf{K}\|\mathbf{K}_t) = \big\langle \mathbf{K}_t - \mathbf{K}, \nabla N_E(\mathbf{K}_t) - \nabla N_E(\mathbf{K})\big\rangle \le a\big\langle \mathbf{K}_t - \mathbf{K}, \nabla N(\mathbf{K}_t) - \nabla N(\mathbf{K})\big\rangle$$

$$\text{i.e. } D_{N_E}(\mathbf{K}_t\|\mathbf{K}) + D_{N_E}(\mathbf{K}\|\mathbf{K}_t) \le a\big[D_N(\mathbf{K}_t\|\mathbf{K}) + D_N(\mathbf{K}\|\mathbf{K}_t)\big] \tag{C.11}$$

Hence, the inequality (C.8) is proved.

Bounding Γ using (C.6), (C.7), (C.8) and Assumptions C.2 and C.3: (C.11) gives

$$\begin{aligned}
a\big[D_{\mathfrak{L}}(\mathbf{K}_t\|\mathbf{K}) + D_{\mathfrak{L}_t}(\mathbf{K}\|\mathbf{K}_t)\big] &\ge D_{N_E}(\mathbf{K}_t\|\mathbf{K}) + D_{N_E}(\mathbf{K}\|\mathbf{K}_t) \\
&= \big\langle \mathbf{K}_t - \mathbf{K}, \nabla N_E(\mathbf{K}_t) - \nabla N_E(\mathbf{K})\big\rangle \text{ (using (C.9))}
\end{aligned}$$

$$
\begin{aligned}
&= \big\langle \mathbf{K} - \mathbf{K}_t, \nabla \frac{1}{n'} \sum_{i=1}^{n'} \mathfrak{L}_r(\mathbf{W}_r \mathbf{K} \mathbf{M}^{(\mathbf{E}^{(i)})}, \mathbf{y}_r^{(\mathbf{E}^{(i)})}) \big\rangle \\
&\quad - \big\langle \mathbf{K} - \mathbf{K}_t, \nabla \frac{1}{n'} \sum_{i=1}^{n'} \mathfrak{L}_r(\mathbf{W}_r \mathbf{K}_t \mathbf{M}^{(\mathbf{E}^{(i)})}, \mathbf{y}_r^{(\mathbf{E}^{(i)})}) \big\rangle \\
&= \frac{1}{n'} \sum_{i=1}^{n'} \Big\langle \mathbf{K} - \mathbf{K}_t, \nabla \mathfrak{L}_r(\mathbf{W}_r \mathbf{K} \mathbf{M}^{(\mathbf{E}^{(i)})}, \mathbf{y}_r^{(\mathbf{E}^{(i)})}) \Big\rangle \\
&\quad - \frac{1}{n'} \sum_{i=1}^{n'} \Big\langle \mathbf{K} - \mathbf{K}_t, \nabla \mathfrak{L}_r(\mathbf{W}_r \mathbf{K}_t \mathbf{M}^{(\mathbf{E}^{(i)})}, \mathbf{y}_r^{(\mathbf{E}^{(i)})}) \Big\rangle \\
&= \frac{1}{n'} \sum_{i=1}^{n'} \Big\langle \mathbf{W}_r (\mathbf{K} - \mathbf{K}_t) \mathbf{E}^{(i)}, \nabla \mathfrak{L}_r(\mathbf{W}_r \mathbf{K} \mathbf{M}^{(\mathbf{E}^{(i)})}, \mathbf{y}_r^{(\mathbf{E}^{(i)})}) \\
&\qquad\qquad - \nabla \mathfrak{L}_r(\mathbf{W}_r \mathbf{K}_t \mathbf{M}^{(\mathbf{E}^{(i)})}, \mathbf{y}_r^{(\mathbf{E}^{(i)})}) \Big\rangle \\
&\geq \frac{g}{n'} \sum_{i=1}^{n'} \|\mathbf{W}_r \mathbf{K} \mathbf{M}^{(\mathbf{E}^{(i)})} - \mathbf{W}_r \mathbf{K}_t \mathbf{M}^{(\mathbf{E}^{(i)})}\|_2^2 \text{ (Assumption C.3)} \\
&= \frac{g}{n'} \sum_{i=1}^{n'} \|\mathbf{W}_r (\mathbf{K} - \mathbf{K}_t) \mathbf{M}^{(\mathbf{E}^{(i)})}\|_2^2.
\end{aligned} \tag{C.12}
$$

Since $\mathbf{K}$ and $\mathbf{K}_t$ are the optimal forward models, $\nabla \mathfrak{L}(\mathbf{K}) = 0$ and $\nabla \mathfrak{L}_t(\mathbf{K}_t) = 0$. In that case, using (C.5), we get

$$
D_{\mathfrak{L}}(\mathbf{K}_t \| \mathbf{K}) + D_{\mathfrak{L}_t}(\mathbf{K} \| \mathbf{K}_t) = \big(\mathfrak{L}(\mathbf{K}_t) - \mathfrak{L}_t(\mathbf{K}_t)\big) + \big(\mathfrak{L}_t(\mathbf{K}) - \mathfrak{L}(\mathbf{K})\big). \tag{C.13}
$$

Using (3.4) and (3.5), we derive

$$
\begin{aligned}
\mathfrak{L}(\mathbf{K}_t) - \mathfrak{L}_t(\mathbf{K}_t) = \frac{1}{n} \Big[\mathfrak{L}_p(\mathbf{W}_p \mathbf{K} \mathbf{M}^{(t)}, \mathbf{y}_p^{(t)}) - \mathfrak{L}_p(\mathbf{W}_p \mathbf{K} \mathbf{M}'^{(t)}, \mathbf{y}_p'^{(t)}) \Big] \\
+ \frac{1}{n} \Big[\mathfrak{L}_r(\mathbf{W}_r \mathbf{K} \mathbf{M}^{(t)}, \mathbf{y}_r^{(t)}) - \mathfrak{L}_r(\mathbf{W}_r \mathbf{K} \mathbf{M}'^{(t)}, \mathbf{y}_r'^{(t)}) \Big].
\end{aligned} \tag{C.14}
$$

Similarly,

$$
\begin{aligned}
\mathfrak{L}_t(\mathbf{K}) - \mathfrak{L}(\mathbf{K}) = \frac{1}{n} \Big[- \mathfrak{L}_p(\mathbf{W}_p \mathbf{K} \mathbf{M}^{(t)}, \mathbf{y}_p^{(t)}) + \mathfrak{L}_p(\mathbf{W}_p \mathbf{K} \mathbf{M}'^{(t)}, \mathbf{y}_p'^{(t)}) \Big] \\
+ \frac{1}{n} \Big[- \mathfrak{L}_r(\mathbf{W}_r \mathbf{K} \mathbf{M}^{(t)}, \mathbf{y}_r^{(t)}) + \mathfrak{L}_r(\mathbf{W}_r \mathbf{K} \mathbf{M}'^{(t)}, \mathbf{y}_r'^{(t)}) \Big].
\end{aligned} \tag{C.15}
$$

Then (C.13) becomes

$$
\begin{aligned}
D_{\mathfrak{L}}(\mathbf{K}_t \| \mathbf{K}) + D_{\mathfrak{L}_t}(\mathbf{K} \| \mathbf{K}_t) = \frac{1}{n} \Big[\mathfrak{L}_p(\mathbf{W}_p \mathbf{K} \mathbf{M}^{(t)}, \mathbf{y}_p^{(t)}) - \mathfrak{L}_p(\mathbf{W}_p \mathbf{K} \mathbf{M}'^{(t)}, \mathbf{y}_p'^{(t)}) \Big] \\
+ \frac{1}{n} \Big[\mathfrak{L}_r(\mathbf{W}_r \mathbf{K} \mathbf{M}^{(t)}, \mathbf{y}_r^{(t)}) - \mathfrak{L}_r(\mathbf{W}_r \mathbf{K} \mathbf{M}'^{(t)}, \mathbf{y}_r'^{(t)}) \Big] \\
+ \frac{1}{n} \Big[- \mathfrak{L}_p(\mathbf{W}_p \mathbf{K} \mathbf{M}^{(t)}, \mathbf{y}_p^{(t)}) + \mathfrak{L}_p(\mathbf{W}_p \mathbf{K} \mathbf{M}'^{(t)}, \mathbf{y}_p'^{(t)}) \Big] \\
+ \frac{1}{n} \Big[- \mathfrak{L}_r(\mathbf{W}_r \mathbf{K} \mathbf{M}^{(t)}, \mathbf{y}_r^{(t)}) + \mathfrak{L}_r(\mathbf{W}_r \mathbf{K} \mathbf{M}'^{(t)}, \mathbf{y}_r'^{(t)}) \Big]
\end{aligned} \tag{C.16}
$$

By Assumption C.2, since $\mathfrak{L}_r$ is σ_r-admissible, then we have

$$|\mathfrak{L}_r(\mathbf{W}_r\mathbf{K}_t\mathbf{M}^{(t)},\mathbf{y}_r^{(t)}) - \mathfrak{L}_r(\mathbf{W}_r\mathbf{K}\mathbf{M}^{(t)},\mathbf{y}_r^{(t)})| \leq \sigma_r\|\mathbf{W}_r(\mathbf{K}_t-\mathbf{K})\mathbf{M}^{(t)}\|_2 \tag{C.17}$$

Similar result is obtained for $\mathfrak{L}_r$ but with σ_p by Assumption C.4,

$$|\mathfrak{L}_p(\mathbf{W}_p\mathbf{K}_t\mathbf{M}^{(t)},\mathbf{y}_p^{(t)}) - \mathfrak{L}_p(\mathbf{W}_p\mathbf{K}\mathbf{M}^{(t)},\mathbf{y}_p^{(t)})| \leq \sigma_p\|\mathbf{W}_r(\mathbf{K}_t-\mathbf{K})\mathbf{M}^{(t)}\|_2 \tag{C.18}$$

Now from (C.12), we have

$$\begin{aligned}
\frac{g}{n'}\sum_{i=1}^{n'}\|\mathbf{W}_r(\mathbf{K}-\mathbf{K}_t)\mathbf{M}^{(\mathrm{E}^{(i)})}\|_2^2 &\leq a\Big[\big(\mathfrak{L}(\mathbf{K}_t)-\mathfrak{L}_t(\mathbf{K}_t)\big)+\big(\mathfrak{L}_t(\mathbf{K})-\mathfrak{L}(\mathbf{K})\big)\Big] \text{ (using (C.13))}\\
&= \frac{a}{n}\Bigg[\Big[\mathfrak{L}_p(\mathbf{W}_p\mathbf{K}\mathbf{M}^{(t)},\mathbf{y}_p^{(t)}) - \mathfrak{L}_p(\mathbf{W}_p\mathbf{K}\mathbf{M}'^{(t)},\mathbf{y}_p'^{(t)})\Big]\\
&\quad - \Big[\mathfrak{L}_p(\mathbf{W}_p\mathbf{K}\mathbf{M}^{(t)},\mathbf{y}_p^{(t)}) - \mathfrak{L}_p(\mathbf{W}_p\mathbf{K}\mathbf{M}'^{(t)},\mathbf{y}_p'^{(t)})\Big]\\
&\quad + \Big[\mathfrak{L}_r(\mathbf{W}_r\mathbf{K}\mathbf{M}^{(t)},\mathbf{y}_r^{(t)}) - \mathfrak{L}_r(\mathbf{W}_r\mathbf{K}\mathbf{M}'^{(t)},\mathbf{y}_r'^{(t)})\Big]\\
&\quad - \Big[-\mathfrak{L}_r(\mathbf{W}_r\mathbf{K}\mathbf{M}^{(t)},\mathbf{y}_r^{(t)}) - \mathfrak{L}_r(\mathbf{W}_r\mathbf{K}\mathbf{M}'^{(t)},\mathbf{y}_r'^{(t)})\Big]\Bigg]\\
&\qquad\qquad \text{(using (C.16))}\\
&\leq \frac{a}{n}\Big[\sigma_p\|\mathbf{W}_r(\mathbf{K}_t-\mathbf{K})\mathbf{M}^{(t)}\|_2 - \sigma_p\|\mathbf{W}_r(\mathbf{K}_t-\mathbf{K})\mathbf{M}'^{(t)}\|_2\\
&\quad + \sigma_r\|\mathbf{W}_r(\mathbf{K}_t-\mathbf{K})\mathbf{M}^{(t)}\|_2 - \sigma_r\|\mathbf{W}_r(\mathbf{K}_t-\mathbf{K})\mathbf{M}'^{(t)}\|_2\Big]\\
&\qquad\qquad \text{(using (C.17) and (C.18))}\\
&= \frac{a(\sigma_p+\sigma_r)}{n}\Big[\|\mathbf{W}_r(\mathbf{K}_t-\mathbf{K})\mathbf{M}^{(t)}\|_2 - \|\mathbf{W}_r(\mathbf{K}_t-\mathbf{K})\mathbf{M}'^{(t)}\|_2\Big]
\end{aligned} \tag{C.19}$$

Now the upper bound of

$$\Gamma = \sqrt{\sum_{i=1}^{n'}\|\mathbf{W}_r(\mathbf{K}_t-\mathbf{K})\mathbf{M}^{(\mathrm{E}^{(i)})}\|_2^2}$$

is obtained by putting all these together as follows

$$\begin{aligned}
\frac{g}{n'}\Gamma^2 &\leq \frac{a(\sigma_p+\sigma_r)}{n}\Big[\|\mathbf{W}_r(\mathbf{K}_t-\mathbf{K})\mathbf{M}^{(t)}\|_2 - \|\mathbf{W}_r(\mathbf{K}_t-\mathbf{K})\mathbf{M}'^{(t)}\|_2\Big]\\
&\leq \frac{a(\sigma_p+\sigma_r)}{n}\Big[\|\mathbf{W}_r(\mathbf{K}_t-\mathbf{K})\mathbf{M}^{(t)}\|_2 + \|\mathbf{W}_r(\mathbf{K}_t-\mathbf{K})\mathbf{M}'^{(t)}\|_2\Big]\\
&\leq \frac{a(\sigma_p+\sigma_r)}{n}\Big[2\Big(\sqrt{r}\Gamma + \frac{2B_{\mathbf{W}_r}B_{\mathbf{K}}\epsilon}{n}\Big)\Big]
\end{aligned}$$

Therefore,

$$\begin{aligned}
&\Gamma^2 \leq \frac{a(\sigma_p+\sigma_r)n'}{gn}\left[2\left(\sqrt{r}\Gamma + \frac{2B_{\mathbf{W}_r}B_{\mathbf{K}}\epsilon}{n}\right)\right] \\
\implies &\Gamma^2 - \frac{2a(\sigma_p+\sigma_r)n'\sqrt{r}\Gamma}{gn} \leq \frac{2a(\sigma_p+\sigma_r)n'}{gn}\cdot\frac{2B_{\mathbf{W}_r}B_{\mathbf{K}}\epsilon}{n} \\
\implies &\Gamma^2 - 2\cdot\Gamma\cdot\frac{a(\sigma_p+\sigma_r)n'\sqrt{r}}{gn} + \left(\frac{a(\sigma_p+\sigma_r)n'\sqrt{r}}{gn}\right)^2 \leq \frac{4a(\sigma_p+\sigma_r)n'B_{\mathbf{W}_r}B_{\mathbf{K}}\epsilon}{gn^2} + \left(\frac{a(\sigma_p+\sigma_r)n'\sqrt{r}}{gn}\right)^2 \\
\implies &\left(\Gamma - \frac{a(\sigma_p+\sigma_r)n'\sqrt{r}}{gn}\right)^2 \leq \frac{a^2(\sigma_p+\sigma_r)^2n'^2r}{g^2n^2} + \frac{4a(\sigma_p+\sigma_r)n'B_{\mathbf{W}_r}B_{\mathbf{K}}\epsilon}{gn^2} \\
\implies &\left(\Gamma - \frac{a(\sigma_p+\sigma_r)n'\sqrt{r}}{gn}\right)^2 \leq \frac{a^2(\sigma_p+\sigma_r)^2n'^2}{g^2n^2}\left(r + \frac{4gB_{\mathbf{W}_r}B_{\mathbf{K}}\epsilon}{a(\sigma_p+\sigma_r)n}\right) \\
\implies &\Gamma - \frac{a(\sigma_p+\sigma_r)n'\sqrt{r}}{gn} \leq \frac{a(\sigma_p+\sigma_r)n'}{gn}\sqrt{\left(r + \frac{4gB_{\mathbf{W}_r}B_{\mathbf{K}}\epsilon}{a(\sigma_p+\sigma_r)n}\right)} \\
\implies &\Gamma \leq \frac{a(\sigma_p+\sigma_r)n'}{gn}\left(\sqrt{r} + \sqrt{r + \frac{4gB_{\mathbf{W}_r}B_{\mathbf{K}}\epsilon}{a(\sigma_p+\sigma_r)n}}\right) \qquad \text{(C.20)}
\end{aligned}$$

Thus, finally, from Assumption C.4, we obtain

$$\begin{aligned}
|\mathfrak{L}_p(\mathbf{W}_p\mathbf{K}_t\mathbf{M},\mathbf{y}_p) - \mathfrak{L}_p(\mathbf{W}_p\mathbf{K}\mathbf{M},\mathbf{y}_p)| &\leq \sigma_p\|\mathbf{W}_r(\mathbf{K}_t-\mathbf{K})\mathbf{M}\|_2 \\
&\leq \sigma_p\sqrt{r}\Gamma + \sigma_p\frac{2B_{\mathbf{W}_r}B_{\mathbf{K}}\epsilon}{n} \text{ (using (C.4))} \\
&\leq \sigma_p\sqrt{r}\frac{a(\sigma_p+\sigma_r)n'}{gn}\left(\sqrt{r} + \sqrt{r + \frac{4gB_{\mathbf{W}_r}B_{\mathbf{K}}\epsilon}{a(\sigma_p+\sigma_r)n}}\right) + \sigma_p\frac{2B_{\mathbf{W}_r}B_{\mathbf{K}}\epsilon}{n} \\
&\qquad \text{(using (C.20))} \\
&= \frac{a(\sigma_p+\sigma_r)n'\sigma_p}{gn}\left(r + \sqrt{r^2 + \frac{4\epsilon gB_{\mathbf{W}_r}B_{\mathbf{K}}r}{a(\sigma_p+\sigma_r)n}}\right) + \frac{2\epsilon\sigma_pB_{\mathbf{W}_r}B_{\mathbf{K}}}{n}
\end{aligned}$$

Hence, the theorem is proved.

References

R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. Slic superpixels. Technical report, 2010.

S. Advani, B. Smith, Y. Tanabe, K. Irick, M. Cotter, J. Sampson, and V. Narayanan. Visual co-occurrence network: using context for large-scale object recognition in retail. In *Embedded Systems For Real-time Multimedia (ESTIMedia), 2015 13th IEEE Symposium on*, pages 1–10. IEEE, 2015.

W. Alhalabi and D. Attas. Toward device assisted identification of grocery store sections and items for the visually impaired. In *Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV)*, page 49. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2016.

A. Andreopoulos and J. K. Tsotsos. 50 years of object recognition: Directions forward. *Computer Vision and Image Understanding*, 117(8):827–891, 2013.

G. Aragon-Camarasa and J. P. Siebert. Unsupervised clustering in hough space for recognition of multiple instances of the same object in a cluttered scene. *Pattern Recognition Letters*, 31(11):1274–1284, 2010.

A. Auclair, L. D. Cohen, and N. Vincent. How to use sift vectors to analyze an image with database templates. In *International Workshop on Adaptive Multimedia Retrieval*, pages 224–236. Springer, 2007.

V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12): 2481–2495, 2017.

R. Bao, K. Higa, and K. Iwamoto. Local feature based multiple object instance identification using scale and rotation invariant implicit shape model. In *Asian Conference on Computer Vision*, pages 600–614. Springer, 2014.

H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. *Computer vision–ECCV 2006*, pages 404–417, 2006.

H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.

I. Baz, E. Yoruk, and M. Cetin. Context-aware hybrid classification system for fine-grained retail product recognition. In *Image, Video, and Multidimensional Signal Processing Workshop (IVMSP), 2016 IEEE 12th*, pages 1–5. IEEE, 2016.

R. Bellman. On a routing problem. *Quarterly of applied mathematics*, 16(1):87–90, 1958.

Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

J. P. Bigham, C. Jayant, A. Miller, B. White, and T. Yeh. Vizwiz:: Locateit-enabling blind people to locate objects in their environment. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pages 65–72. IEEE, 2010.

T. Bishop. How 'amazon go'works: the technology behind the online retailer's groundbreaking new grocery store. *GeekWire. Extraído de https://www. geekwire. com/2016/amazon-go-works-technology-behind-online-retailersgroundbreaking-new-grocery-store*, 2016.

N. Bodla, B. Singh, R. Chellappa, and L. S. Davis. Soft-nms–improving object detection with one line of code. In *Proceedings of the IEEE international conference on computer vision*, pages 5561–5569, 2017.

A. Bosch, A. Zisserman, and X. Munoz. Image classification using random forests and ferns. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

O. Bousquet and A. Elisseeff. Stability and generalization. *Journal of machine learning research*, 2 (Mar):499–526, 2002.

R. Brenner, J. Priyadarshi, and L. Itti. Perfect accuracy with human-in-the-loop object detection. In *European Conference on Computer Vision*, pages 360–374. Springer, 2016.

N. Bruce and J. Tsotsos. An information theoretic model of saliency and visual search. *Attention in cognitive systems. Theories and systems from an interdisciplinary viewpoint*, pages 171–183, 2007.

J. Canny. A computational approach to edge detection. In *Readings in Computer Vision*, pages 184–203. Elsevier, 1987.

K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*, 2014.

A. Chaurasia and E. Culurciello. Linknet: Exploiting encoder representations for efficient semantic segmentation. In *2017 IEEE Visual Communications and Image Processing (VCIP)*, pages 1–4. IEEE, 2017.

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

L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018.

T. Chong, I. Bustan, and M. Wee. Deep learning approach to planogram compliance in retail stores.

J. Cleveland, D. Thakur, P. Dames, C. Phillips, T. Kientz, K. Daniilidis, J. Bergstrom, and V. Kumar. Automated system for semantic object labeling with soft-object recognition and dynamic programming segmentation. *IEEE Transactions on Automation Science and Engineering*, 2016.

N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, pages 248–255. Ieee, 2009.

L. R. Dice. Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302, 1945.

A. Dingli and I. Mercieca. Multimedia interfaces for people visually impaired. In *Advances in Design for Inclusion*, pages 487–495. Springer, 2016.

A. Diplaros, T. Gevers, I. Patras, et al. Color-shape context for object recognition. In *IEEE Workshop on Color and Photometric Methods in Computer Vision*, pages 1–8, 2003.

A. Diplaros, T. Gevers, and I. Patras. Combining color and shape information for illumination-viewpoint invariant object recognition. *IEEE Transactions on Image Processing*, 15(1):1–11, 2006.

E. R. Dougherty. An introduction to morphological image processing. *Tutorial texts in optical engineering*, 1992.

V. Dumoulin and F. Visin. A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv:1603.07285*, 2016.

M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.

T. Fan, G. Wang, Y. Li, and H. Wang. Ma-net: A multi-scale attention network for liver and tumor segmentation. *IEEE Access*, 8:179656–179665, 2020.

P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32 (9):1627–1645, 2010.

C. F. Flores, A. Gonzalez-Garcia, J. van de Weijer, and B. Raducanu. Saliency for fine-grained object recognition in domains with scarce training data. *Pattern Recognition*, 94:62–73, 2019.

G. L. Foresti and C. Regazzoni. A change-detection method for multiple object localization in real scenes. In *Industrial Electronics, Control and Instrumentation, 1994. IECON'94., 20th International Conference on*, volume 2, pages 984–987. IEEE, 1994.

A. Franco, D. Maltoni, and S. Papi. Grocery product detection and recognition. *Expert Systems with Applications*, 81:163–176, 2017.

Y. Freund, R. Schapire, and N. Abe. A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, 14(771-780):1612, 1999.

E. Frontoni, M. Contigiani, and G. Ribighini. A heuristic approach to evaluate occurrences of products for the planogram maintenance. In *Mechatronic and Embedded Systems and Applications (MESA), 2014 IEEE/ASME 10th International Conference on*, pages 1–6. IEEE, 2014.

W. Ge, X. Lin, and Y. Yu. Weakly supervised complementary parts models for fine-grained image classification from the bottom up. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3034–3043, 2019.

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

M. George and C. Floerkemeier. Recognizing products: A per-exemplar multi-label image classification approach. In *European Conference on Computer Vision*, pages 440–455. Springer, 2014.

M. George, D. Mircic, G. Soros, C. Floerkemeier, and F. Mattern. Fine-grained product class recognition for assisted shopping. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 154–162, 2015.

T. Gevers. Robust histogram construction from color invariants. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference On*, volume 1, pages 615–620. IEEE, 2001.

T. Gevers and A. W. Smeulders. Color-based object recognition. *Pattern recognition*, 32(3):453–464, 1999a.

T. Gevers and A. W. Smeulders. Content-based image retrieval by viewpoint-invariant color indexing. *Image and vision computing*, 17(7):475–488, 1999b.

T. Gevers and A. W. Smeulders. Pictoseek: Combining color and shape invariant features for image retrieval. *IEEE transactions on Image Processing*, 9(1):102–119, 2000.

T. Gevers and H. Stokman. Robust histogram construction from color invariants for object recognition. *IEEE transactions on pattern analysis and machine intelligence*, 26(1):113–118, 2004.

Y. A. Ghassabeh and H. A. Moghaddam. Adaptive linear discriminant analysis for online feature extraction. *Machine vision and applications*, 24(4):777–794, 2013.

M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, and W. Li. Deep reconstruction-classification networks for unsupervised domain adaptation. In *European Conference on Computer Vision*, pages 597–613. Springer, 2016.

R. Girshick. Fast r-cnn. In *Computer Vision (ICCV)*, pages 1440–1448. IEEE International Conference on, 2015.

R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587. IEEE Conference on, 2014.

E. Goldman and J. Goldberger. Large-scale classification of structured image classification from conditional random field with deep class embedding. *CoRR*, abs/1705.07420, 2017.

I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.

T. W. Gruen, D. Corsten, and S. Bharadwaj. Retail out of stocks: A worldwide examination of causes, rates, and consumer responses. *Grocery Manufacturers of America, Washington, DC*, 2002.

Z. Haladová and E. Šikudová. Multiple instances detection in rgbd images. In *International Conference on Computer Vision and Graphics*, pages 246–253. Springer, 2014.

H. He and S. Chen. Imorl: Incremental multiple-object recognition and localization. *IEEE Transactions on Neural Networks*, 19(10):1727–1738, 2008.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778. IEEE Conference on, 2016.

K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

K. Higa, K. Iwamoto, and T. Nomura. Multiple object identification using grid voting of object center estimated from keypoint matches. In *Image Processing (ICIP), 2013 20th IEEE International Conference on*, pages 2973–2977. IEEE, 2013.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Y. Hu, M. Lu, and X. Lu. Driving behaviour recognition from still images by using multi-stream fusion cnn. *Machine Vision and Applications*, 30(5):851–865, 2019.

D. Huang, R. Zhang, Y. Yin, Y. Wang, and Y. Wang. Local feature approach to dorsal hand vein recognition by centroid-based circular key-point grid and fine-grained matching. *Image and Vision Computing*, 58:266–277, 2017.

S. Huang, Z. Xu, D. Tao, and Y. Zhang. Part-stacked cnn for fine-grained visual categorization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1173–1182, 2016.

L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 20(11):1254–1259, 1998.

K. Iwamoto, R. Mase, and T. Nomura. Bright: A scalable and compact binary descriptor for low-latency and high accuracy object identification. In *Image Processing (ICIP), 2013 20th IEEE International Conference on*, pages 2915–2919. IEEE, 2013.

D. Jameson and L. M. Hurvich. Essay concerning color constancy. *Annual review of psychology*, 40(1): 1–24, 1989.

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 *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678. ACM, 2014.

T. Joachims, T. Finley, and C.-N. Yu. Cutting-plane training of structural svms. *Machine Learning*, 77: 27–59, 10 2009. doi: 10.1007/s10994-009-5108-8.

P. Jund, N. Abdo, A. Eitel, and W. Burgard. The freiburg groceries dataset. *arXiv preprint arXiv:1611.05799*, 2016.

L. Karlinsky, J. Shtok, Y. Tzur, and A. Tzadok. Fine-grained recognition of thousands of object categories with single-example training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4113–4122. IEEE Conference on, 2017.

N. Kejriwal, S. Garg, and S. Kumar. Product counting using images with application to robot-based retail stock assessment. In *Technologies for Practical Robot Applications (TePRA), 2015 IEEE International Conference on*, pages 1–6. IEEE, 2015.

T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

G. Koch, R. Zemel, and R. Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2. Lille, 2015.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

V. Kulyukin and A. Kutiyanawala. Accessible shopping systems for blind and visually impaired individuals: Design requirements and the state of the art. *The Open Rehabilitation Journal*, 3:158–168, 2010.

L. Le, A. Patterson, and M. White. Supervised autoencoders: Improving generalization performance with unsupervised regularizers. In *Advances in Neural Information Processing Systems*, pages 107–117, 2018.

Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pages 9–48. Springer, 2012.

S. Leutenegger, M. Chli, and R. Y. Siegwart. Brisk: Binary robust invariant scalable keypoints. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2548–2555. IEEE, 2011.

K. Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly of applied mathematics*, 2(2):164–168, 1944.

H. Li, P. Xiong, J. An, and L. Wang. Pyramid attention network for semantic segmentation. In *BMVC*, 2018.

T.-Y. Lin and S. Maji. Improved Bilinear Pooling with CNNs. In *British Machine Vision Conference (BMVC)*, 2017.

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

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

J. Liu and Y. Liu. Grasp recurring patterns from a single view. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2003–2010, 2013.

S. Liu and H. Tian. Planogram compliance checking using recurring patterns. In *Multimedia (ISM), 2015 IEEE International Symposium on*, pages 27–32. IEEE, 2015.

S. Liu, W. Li, S. Davis, C. Ritz, and H. Tian. Planogram compliance checking based on detection of recurring patterns. *IEEE MultiMedia*, 23(2):54–63, 2016a.

T. Liu, D. Tao, M. Song, and S. J. Maybank. Algorithm-dependent generalization bounds for multi-task learning. *IEEE transactions on pattern analysis and machine intelligence*, 39(2):227–241, 2016b.

W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016c.

S. Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

D. López-de Ipiña, T. Lorido, and U. López. Indoor navigation and product recognition for blind people assisted shopping. In *International Workshop on Ambient Assisted Living*, pages 33–40. Springer, 2011.

D. G. Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.

D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

S. Luan, C. Chen, B. Zhang, J. Han, and J. Liu. Gabor convolutional networks. *IEEE Transactions on Image Processing*, 27(9):4357–4366, 2018.

K. Lyu, Y. Li, and Z. Zhang. Attention-aware multi-task convolutional neural networks. *IEEE Transactions on Image Processing*, 29:1867–1878, 2019.

W. Ma and J. Lu. An equivalence of fully connected layer and convolutional layer. *arXiv preprint arXiv:1712.01252*, 2017.

M. Marder, S. Harary, A. Ribak, Y. Tzur, S. Alpert, and A. Tzadok. Using image analytics to monitor retail store shelves. *IBM Journal of Research and Development*, 59(2/3):3–1, 2015.

D. W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the society for Industrial and Applied Mathematics*, 11(2):431–441, 1963.

A. Martins, M. Figueiredo, P. Aguiar, N. Smith, and E. Xing. Ad3: Alternating directions dual decomposition for map inference in graphical models. *Journal of Machine Learning Research*, 16:495–545, 03 2015.

A. Maurer, M. Pontil, and B. Romera-Paredes. The benefit of multitask representation learning. *The Journal of Machine Learning Research*, 17(1):2853–2884, 2016.

M. O. M. Medina, Z. Fan, T. Ranatunga, D. T. Barry, U. Sinha, S. Kaza, and V. Krishna. Customer service robot and related systems and methods, Oct. 23 2015. US Patent App. 14/921,899.

M. Merler, C. Galleguillos, and S. Belongie. Recognizing groceries in situ using in vitro training data. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

C. P. Metzger. *High fidelity shelf stock monitoring*. PhD thesis, ETH Zurich, Zurich, Switzerland, 2008.

K. Michael and L. McCathie. The pros and cons of rfid in supply chain management. In *International Conference on Mobile Business (ICMB'05)*, pages 623–629, 2005. doi: 10.1109/ICMB.2005.103.

M. Mohri, A. Rostamizadeh, and D. Storcheus. Generalization bounds for supervised dimensionality reduction. In *Feature Extraction: Modern Questions and Challenges*, pages 226–241, 2015.

R. Moorthy, S. Behera, and S. Verma. On-shelf availability in retailing. *International Journal of Computer Applications*, 115:47–51, 04 2015. doi: 10.5120/20296-2811.

D. Mukherjee, Q. J. Wu, and G. Wang. A comparative experimental study of image feature detectors and descriptors. *Machine Vision and Applications*, 26(4):443–466, 2015.

A. C. Müller and S. Behnke. Pystruct: learning structured prediction in python. *J. Mach. Learn. Res.*, 15 (1):2055–2060, 2014.

J. Nicholson, V. Kulyukin, and D. Coster. Shoptalk: independent blind shopping through verbal route directions and barcode scans. *The Open Rehabilitation Journal*, 2(1):11–23, 2009.

C. L. Novak and S. A. Shafer. Anatomy of a color histogram. In *Computer Vision and Pattern Recognition, 1992. Proceedings CVPR'92., 1992 IEEE Computer Society Conference on*, pages 599–605. IEEE, 1992.

K. Oh, M. Lee, G. Kim, and S. Kim. Detection of multiple salient objects through the integration of estimated foreground clues. *Image and Vision Computing*, 54:31–44, 2016.

C. Papageorgiou and T. Poggio. Trainable pedestrian detection. In *Image Processing, 1999. ICIP 99. Proceedings. 1999 International Conference on*, volume 4, pages 35–39. IEEE, 1999.

C. P. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. In *Computer vision, 1998. sixth international conference on*, pages 555–562. IEEE, 1998.

A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

P. Pawara, E. Okafor, M. Groefsema, S. He, L. R. Schomaker, and M. A. Wiering. One-vs-one classification for deep neural networks. *Pattern Recognition*, 108:107528, 2020.

Y. Peng, X. He, and J. Zhao. Object-part attention model for fine-grained image classification. *IEEE Transactions on Image Processing*, 27(3):1487–1500, 2017.

F. Perronnin, J. Sánchez, and T. Mensink. Improving the fisher kernel for large-scale image classification. In *European conference on computer vision*, pages 143–156. Springer, 2010.

R. Pietrini, D. Manco, M. Paolanti, V. Placidi, E. Frontoni, and P. Zingaretti. An iot edge-fog-cloud architecture for vision based planogram integrity. In *Proceedings of the 13th International Conference on Distributed Smart Cameras*, pages 1–5, 2019.

A. Ray, N. Kumar, A. Shaw, and D. Prasad Mukherjee. U-pc: Unsupervised planogram compliance. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 586–600, 2018.

J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 779–788. IEEE Conference on, 2016.

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

H. Robbins and S. Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

B. Santra and D. P. Mukherjee. A comprehensive survey on computer vision based approaches for automatic identification of products in retail store. *Image and Vision Computing*, 86:45–63, 2019.

B. Santra, A. Paul, and D. P. Mukherjee. Deterministic dropout for deep neural networks using composite random forest. *Pattern Recognition Letters*, 131:205 – 212, 2020a. ISSN 0167-8655. doi: https://doi.org/10.1016/j.patrec.2019.12.023.

B. Santra, A. K. Shaw, and D. P. Mukherjee. Graph-based non-maximal suppression for detecting products on the rack. *Pattern Recognition Letters*, 140:73 – 80, 2020b. ISSN 0167-8655. doi: https://doi.org/10.1016/j.patrec.2020.09.023.

B. Santra, U. Ghosh, and D. P. Mukherjee. Datasets for identification of gaps in the images of shleves in supermarkets. `https://github.com/gapDetection/gapDetectionDatasets`, 2021a.

B. Santra, A. K. Shaw, and D. P. Mukherjee. An end-to-end annotation-free machine vision system for detection of products on the rack. *Machine Vision and Applications*, 32(3):1–13, 2021b.

A. Saran, E. Hassan, and A. K. Maurya. Robust visual analysis for planogram compliance problem. In *Machine Vision Applications (MVA), 2015 14th IAPR International Conference on*, pages 576–579. IEEE, 2015.

M. Shapiro. Executing the best planogram. In *Professional Candy Buyer*, Norwalk, CT, USA, Nov./Dec. 2009.

J. Shen, X. Hao, Z. Liang, Y. Liu, W. Wang, and L. Shao. Real-time superpixel segmentation by dbscan clustering algorithm. *IEEE Transactions on Image Processing*, 25(12):5933–5942, 2016.

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

I. Sobel. History and definition of the sobel operator. *Retrieved from the World Wide Web*, 2014.

H. Sun, J. Zhang, and T. Akashi. Templatefree: product detection on retail store shelves. *IEEJ Transactions on Electrical and Electronic Engineering*, 15(2):242–251, 2020.

T. Sun, L. Sun, and D.-Y. Yeung. Fine-grained categorization via cnn-based automatic extraction and integration of object-level and part-level features. *Image and Vision Computing*, 64:47–66, 2017.

C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.

R. Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

K. A. Thakoor, S. Marat, P. J. Nasiatka, B. P. McIntosh, F. E. Sahin, A. R. Tanguay, J. D. Weiland, and L. Itti. Attention biased speeded up robust features (ab-surf): a neurally-inspired object recognition algorithm for a wearable aid for the visually-impaired. In *Multimedia and Expo Workshops (ICMEW), 2013 IEEE International Conference on*, pages 1–6. IEEE, 2013.

A. Tonioni and L. Di Stefano. Product recognition in store shelves as a sub-graph isomorphism problem. In *International Conference on Image Analysis and Processing*, pages 682–693. Springer, 2017.

A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing visual features for multiclass and multiview object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(5):854–869, 2007.

A. M. Treisman and G. Gelade. A feature-integration theory of attention. *Cognitive psychology*, 12(1): 97–136, 1980.

J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013.

G. Varol and R. S. Kuzu. Toward retail product recognition on grocery shelves. In *Sixth International Conference on Graphic and Image Processing (ICGIP 2014)*, pages 944309–944309. International Society for Optics and Photonics, 2015.

G. Varol, R. S. Kuzu, and Y. S. Akgiil. Product placement detection based on image processing. In *Signal Processing and Communications Applications Conference (SIU), 2014 22nd*, pages 1031–1034. IEEE, 2014.

M. Villamizar, A. Garrell, A. Sanfeliu, and F. Moreno-Noguer. Interactive multiple object learning with scanty human supervision. *Computer Vision and Image Understanding*, 149:51–64, 2016.

P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–I. IEEE, 2001.

B.-N. Vo, B.-T. Vo, N.-T. Pham, and D. Suter. Joint detection and estimation of multiple objects from image observations. *IEEE Transactions on Signal Processing*, 58(10):5129–5141, 2010.

K. Wada. labelme: Image Polygonal Annotation with Python. https://github.com/wkentaro/labelme, 2016.

L. Wan, M. Zeiler, S. Zhang, Y. Le Cun, and R. Fergus. Regularization of neural networks using dropconnect. In *International conference on machine learning*, pages 1058–1066, 2013.

S. T. Welstead. *Fractal and wavelet image compression techniques*, volume 40. Spie Press, 1999.

WHO. World health organization fact sheet, N° 282 (2014), 2014. URL http://www.who.int/mediacentre/factsheets/fs282/en/. accessed on 24-Jun-2017.

T. Winlock, E. Christiansen, and S. Belongie. Toward real-time grocery detection for the visually impaired. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pages 49–56. IEEE, 2010.

T. Xiao, Y. Xu, K. Yang, J. Zhang, Y. Peng, and Z. Zhang. The application of two-level attention models in deep convolutional neural network for fine-grained image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 842–850, 2015.

G.-S. Xie, X.-Y. Zhang, W. Yang, M. Xu, S. Yan, and C.-L. Liu. Lg-cnn: From local parts to global discrimination for fine-grained recognition. *Pattern Recognition*, 71:118–131, 2017.

S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pages 802–810, 2015.

H. Xue, S. Chen, and Q. Yang. Structural support vector machine. In F. Sun, J. Zhang, Y. Tan, J. Cao, and W. Yu, editors, *Advances in Neural Networks - ISNN 2008*, pages 501–511, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg. ISBN 978-3-540-87732-5.

P. Yakubovskiy. Segmentation models pytorch. https://github.com/qubvel/segmentation_models.pytorch, 2020.

H. Yao, D. Zhang, J. Li, J. Zhou, S. Zhang, and Y. Zhang. Dsp: Discriminative spatial part modeling for fine-grained visual categorization. *Image and Vision Computing*, 63:24–37, 2017.

E. Yörük, K. T. Öner, and C. B. Akgül. An efficient hough transform for multi-instance object recognition and pose estimation. In *Pattern Recognition (ICPR), 2016 23rd International Conference on*, pages 1352–1357. IEEE, 2016.

J. Yu, K. C. Yow, and M. Jeon. Joint representation learning of appearance and motion for abnormal event detection. *Machine Vision and Applications*, 29(7):1157–1170, 2018.

R. Yılmazer and D. Birant. Shelf auditing based on image classification using semi-supervised deep learning to increase on-shelf availability in grocery stores. *Sensors*, 21:327, 01 2021. doi: 10.3390/s21020327.

Q. Zhang, D. Qu, F. Xu, K. Jia, N. Jiang, and F. Zou. An improved method for object instance detection based on object center estimation and convex quadrilateral verification. In *Information Technology, Networking, Electronic and Automation Control Conference, IEEE*, pages 174–177. IEEE, 2016a.

Q. Zhang, D. Qu, F. Xu, K. Jia, and X. Sun. Dual-layer density estimation for multiple object instance detection. *Journal of Sensors*, 2016, 2016b.

T. Zhang. Covering number bounds of certain regularized linear function classes. *Journal of Machine Learning Research*, 2(Mar):527–550, 2002.

Y. Zhang, L. Wang, R. Hartley, and H. Li. Where's the weet-bix? In *Asian Conference on Computer Vision*, pages 800–810. Springer, 2007.

Y. Zhang, L. Wang, R. Hartley, and H. Li. Handling significant scale difference for object retrieval in a supermarket. In *Digital Image Computing: Techniques and Applications, 2009. DICTA'09.*, pages 468–475. IEEE, 2009.

H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890, 2017.

P. Zientara, S. Advani, N. Shukla, I. Okafor, K. Irick, J. Sampson, S. Datta, and V. Narayanan. A multitask grocery assistance system for the visually impaired smart glasses, gloves, and shopping carts provide auditory and tactile feedback. *IEEE CONSUMER ELECTRONICS MAGAZINE*, 6(1):73–81, 2017a.

P. A. Zientara, S. Lee, G. H. Smith, R. Brenner, L. Itti, M. B. Rosson, J. M. Carroll, K. M. Irick, and V. Narayanan. Third eye: A shopping assistant for the visually impaired. *Computer*, 50(2):16–24, 2017b.

List of Publications by the Author

Patents Pending

A. K. Shaw, S. Y. Rao, P. Hari, D. P. Mukherjee, and **B. Santra**. Fine-grained classification of retail products. 2020. [Referred in the thesis].

Patents

A. K. Shaw, R. Ramakrishnan, S. Y. Rao, P. Hari, D. P. Mukherjee, and **B. Santra**. Method and system for region proposal based object recognition for estimating planogram compliance. *Australia Patent: 2020205301*. [Referred in the thesis].

Journal Articles

S. Agarwal, **B. Santra**, and D. P. Mukherjee. Anubhav: recognizing emotions through facial expression. *The Visual Computer*, 34(2):177–191, 2018.

B. Santra and D. P. Mukherjee. A comprehensive survey on computer vision based approaches for automatic identification of products in retail store. *Image and Vision Computing*, 86:45–63, 2019. [Referred in the thesis].

B. Santra, A. Paul, and D. P. Mukherjee. Deterministic dropout for deep neural networks using composite random forest. *Pattern Recognition Letters*, 131:205 – 212, 2020a. ISSN 0167-8655. [Referred in the thesis].

B. Santra, A. K. Shaw, and D. P. Mukherjee. Graph-based non-maximal suppression for detecting products on the rack. *Pattern Recognition Letters*, 140:73 – 80, 2020b. ISSN 0167-8655. [Referred in the thesis].

B. Santra, U. Ghosh, and D. P. Mukherjee. Graph-based modelling of superpixels for automatic identification of empty shelves in supermarkets. *Pattern Recognition*, 2021a. [Under revision] [Referred in the thesis].

B. Santra, A. K. Shaw, and D. P. Mukherjee. An end-to-end annotation-free machine vision system for detection of products on the rack. *Machine Vision and Applications*, 32:56, 2021b. ISSN 1432-1769. [Referred in the thesis].

B. Santra, A. K. Shaw, and D. P. Mukherjee. Generalization ability of supervised convolutional autoencoder in classifying retail product images. *IEEE Transactions on Image Processing*, 2021c. [Under the process of re-submission] [Referred in the thesis].

B. Santra, A. K. Shaw, and D. P. Mukherjee. Part-based annotation-free fine-grained classification of images of retail products. *Pattern Recognition*, 121:108257, 2022. [Referred in the thesis].

Conference Papers

B. Santra and D. P. Mukherjee. Local dominant binary patterns for recognition of multi-view facial expressions. In *Proceedings of the Tenth Indian Conference on Computer Vision, Graphics and Image Processing*, page 25. ACM, 2016a.

B. Santra and D. P. Mukherjee. Local saliency-inspired binary patterns for automatic recognition of multi-view facial expression. In *Image Processing (ICIP), 2016 IEEE International Conference on*, pages 624–628. IEEE, 2016b.

B. Santra, D. P. Mukherjee, and D. Chakrabarti. A non-invasive approach for estimation of hemoglobin analyzing blood flow in palm. In *Biomedical Imaging (ISBI 2017), 2017 IEEE 14th International Symposium on*, pages 1100–1103. IEEE, 2017.