IMAGE ANALYSIS FOR SHADOW DETECTION, SATELLITE IMAGE FORENSICS AND EATING SCENE SEGMENTATION AND CLUSTERING

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Sri Kalyan Yarlagadda

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

December 2020

Purdue University

West Lafayette, Indiana

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF DISSERTATION APPROVAL

Dr. Fenging M. Zhu, Chair

School of Electrical and Computer Engineering

Dr. Amy R. Reibman

School of Electrical and Computer Engineering

Dr. Jan P. Allebach

School of Electrical and Computer Engineering

Dr. Mary L. Comer

School of Electrical and Computer Engineering

Approved by:

Dr. Dimitrios Peroulis

Head of the School Graduate Program

“ The older I got, the smarter my teachers became. ” - *Ally Carter*

ACKNOWLEDGMENTS

First and foremost I would like to thank my doctoral advisor Prof. Fengqing Maggie Zhu. I am very grateful to her for choosing to guide me through my PhD. Her suggestions, encouragements and criticisms have been of immense help to me in my academic pursuits. There have been many times where I was stuck and uninspired but her guidance was what got me through. I owe her a lot. I would also like to thank Prof. Edward J. Delp for his invaluable guidance. I am very grateful to him for the opportunity to be a member of the Video and Image Processing Laboratory (VIPER). Even though I wasn't a direct student of his, he still took genuine interest in my work and gave a lot of invaluable advice. For this I am very thankful to him.

I would like to thank Prof. Amy R. Reibman, Prof. Mary L. Comer and Prof. Jan P. Allebach for being on my doctoral committee despite their busy schedules. My interactions with them have always proven to be very fruitful. I would also like to thank Prof. Paolo Bestagini for all the wonderful interactions. I learnt a lot from him. I would also like to thank Prof Edward Sazonov. My interactions with him have been always been a great learning experience. Lastly I would like to thank all the members of Video and Image Processing Laboratory (VIPER) I intereacted with for their help and support: Dr. Neeraj J. Gadgil, Dr. Khalid Tahboub, Dr. Joonsoo Kim, Dr. Yu Wang, Dr. Chichen Fu, Dr. Shaobo Fang, Dr. Javier Ribera Prat, Dr. David Joon Ho, Dr. Soonam Lee, Dr. Jeehyun Choe, Dr. Dahjung Chung,Blanca Delgado, He Li, and Chang LiuSriram Baireddy, Emily R. Bartusiak, Enyu Cai, Alain Chen,Di Chen, Qingshuang Chen, Yuhao Chen, Jiaqi Guo, Mridul Gupta, Shuo Han, Hanxiang(Hans) Hao, Jiangpeng He, Janos Horvath, Han Hu, Runyu Mao, Daniel Mas, Ruiting Shao, Zeman Shao, Changye Yang, David Güera, and Yifan Zhao. And to my family, thank you for your love and support.

And to my parents, grandma and brother, Srivinvas Rao, Sridevi, Jayalakshmi and Sri Vishnu, and all my family and friends, thanks to you for all the love and support. I am very grateful to you.

The materials in chapters 3 and 4 is based on research sponsored by DARPA and Air Force Research Laboratory (AFRL) under agreement number FA8750-16-2-0173. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes not withstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA and Air Force Research Laboratory (AFRL) or the U.S. Government.

TABLE OF CONTENTS

Page

Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

ABSTRACT

Yarlagadda Sri Kalyan Ph.D., Purdue University, December 2020. Image Analysis For Shadow Detection, Satellite Image Forensics and Eating Scene Segmentation and Clustering. Major Professor: Fengqing Maggie Zhu.

Recent advances in machine learning has enabled notable progress in many aspects of image analysis. In this thesis, we present three applications to exemplify such advancement, including shadow detection, satellite image forensics and eating scene segmentation and clustering. Shadow detection and removal are of great interest to the image processing and image forensics community. In this thesis, we study automatic shadow detection from two different perspectives. First, we propose automatic methods for detecting and removing shadows in color images. Second, we present machine learning based methods to detect if shadows have been removed in an image. In the second part of the thesis, we study image forensics for satellite images. Satellite images have been subjected to various tampering and manipulations due to easy access and the availability of image manipulation tools. In this thesis, we propose methods to automatically detect and localize spliced objects in satellite images. Extracting information from the eating scene captured by images provides new means of studying the relationship between diet and health. In the third part of the thesis, we propose a class-agnostic food segmentation method that is able to segment foods without knowing the food type and a method to cluster eating scene images based on the eating environment.

1. INTRODUCTION

Image analysis focuses on the extraction of meaningful information from digital images by means of digital image processing techniques. Over the years a vast number of digital image processing techniques have been developed for various paradigms such as media forensics [1], food image analysis [2], object detection [3], semantic segmentation [4], image and video compression [5], augmented reality and much more. These techniques sit at the heart of products and technologies that we use on a daily basis and are also enabling new generation of technologies such as self-driving cars, and augmented reality headsets. Cameras in smart phones are able to capture high quality images because of image analysis techniques developed for noise reduction and low-light photography. Many image analysis techniques are used in the manufacturing industry for automating tasks in the production lines. Today, image analysis techniques are capable of compressing high quality videos into files occupying a few mega bytes. Techniques like these form the backbone of all video streaming services in use today. Image analysis techniques developed for object detection, pose estimation and semantic segmentation are used extensively in the development of self-driving car technology.

In the past, a lot of feature engineering went into developing image analysis techniques. These features are often hand-crafted and specific to the task at hand. These features need to be customized either by manual tuning or learnt using machine learning methods. Examples of image analysis techniques based on hand crafted features include SIFT [6], SURF [7], histogram equalization, and SLIC [8]. Such approach of crafting features by hand lead to some very successful image analysis techniques. However, for many important tasks such as object detection and semantic segmentation, their performance was far below common human capability. With recent advance of deep learning [9], automatic feature engineering was made possible and

has led to the development of more effective image analysis techniques. In this thesis, we present deep learning based image analysis techniques for shadow analysis, satellite image forensics and food image analysis. Before delving into details we provide a very brief overview of deep learning.

Deep learning is a sub field of machine learning that revolves around using Artificial Neural Networks (ANN) for learning representations of data. Methods dealing with learning representations of data fall under the umbrella of representation learning. Deep learning based methods employ ANN to learn these representations. Multilayered ANN's are powerful mathematical constructs that are proven to be capable of approximating a large variety of mathematical function to arbitrary accuracy [10]. For this reason, they are nicknamed **"Universal Approximators"**. While ANNs have shown great potential, they come with a few challenges. First, it is hard to optimize them and second, they are very data hungry. Backpropogation [11] proposed in 1986 made it possible to optimize multi-layer ANNs, but processing is very slow due to low computational power offered by the then state-of-the-art machines. Also, massive amounts of data is required to train these ANNs. Since 1986, computational power of processors has risen exponentially, data storage has become very cheap and powerful graphical processing unit (GPU) can very efficiently perform backpropogation. All these technology advancements have enabled researchers to train multilayered ANNs on massive amount of data. This was first demonstrated by a group of researches led by Geoffrey E. Hinton in [12]. In [12] they trained AlexNet, a very deep Convolutional Neural Network (CNN) (a special type of ANN) to classify images into 1,000 categories. AlexNet consists of approximately 60 million parameters and was trained on ImageNet [13] dataset. ImageNet is a dataset that contains more than 1 million images belonging to 1,000 different categories. In this classification task, AlexNet outperformed its nearest competitor by a significant margin ($> 10\%$ in top-5 error) demonstrating the effectiveness of multilayered ANNs.

1.1 Shadow Detection And Removal

Shadows are common aspect of images and videos. They often contain a lot of information essential for scene lightning, scene geometry and object tracking. Because of this shadow analysis is of great interest to the image processing community. In Chapter 2 we propose a simple yet effective approach to detect and remove shadows from a single image. An image is first segmented and based on the reflectance, illumination and texture characteristics, segment pairs are identified as shadow and non-shadow pairs. The proposed method is tested on two publicly available and widely used datasets. Our method achieves higher accuracy in detecting shadows compared to previous reported methods despite requiring fewer parameters. In addition we also show results of shadow-free images obtained by relighting the pixels in the detected shadow regions.

1.2 Shadow Removal Detection And Localization For Forensics Analysis

Because shadows are so integral to images they are also of importance to image forensics community. The recent advancements in image processing and computer vision allow realistic photo manipulations. In order to avoid the distribution of fake imagery, the image forensics community is working towards the development of image authenticity verification tools. Methods based on shadow analysis are particularly reliable since they are part of the physical integrity of the scene, thus detecting forgeries is possible whenever inconsistencies are found (e.g., shadows not coherent with the light direction). An attacker can easily delete inconsistent shadows and replace them with correctly cast shadows in order to fool forensics detectors based on physical analysis. In Chapter 3, we propose a method to detect shadow removal done with state-of-the- art tools. The proposed method is based on a convolutional neural network (CNN) specifically trained for shadow removal detection using a conditional generative adversarial network (cGAN).

1.3 Detection And Localization Of Splicing In Satellite Images

In Chapter 4 we look at verifying the integrity of satellite images. Current satellite imaging technology enables shooting high-resolution pictures of the ground. As any other kind of digital images, overhead pictures can also be easily forged. However, common image forensic techniques are often developed for consumer camera images, which strongly differ in their nature from satellite ones (e.g., compression schemes, post-processing, sensors, etc. Therefore, many accurate state-of-the-art forensic algorithms are bound to fail if blindly applied to overhead image analysis. Development of novel forensic tools for satellite images is paramount to assess their authenticity and integrity. In this chapter, we propose an method for satellite image forgery detection and localization. Specifically, we consider the scenario in which pixels within a region of a satellite image are replaced to add or remove an object from the scene. Our algorithm works under the assumption that no forged images are available for training. Using a generative adversarial network (GAN), we learn a feature representation of pristine satellite images. A one-class support vector machine (SVM) is trained on these features to determine their distribution. Finally, image forgeries are detected as anomalies. The proposed algorithm is validated against different kinds of satellite images containing forgeries of different size and shape.

1.4 Learning Eating Environments Through Scene Clustering

In Chapter 5, we propose a method to cluster eating scene images based on their eating environments. It is well known that dietary habits have a significant influence on health. While many studies have been conducted to understand this relationship, little is known about the relationship between eating environments and health. Yet researchers and health agencies around the world have recognized the eating environment as a promising context for improving diet and health. In this paper, we propose an image clustering method to automatically extract the eating environments from eating occasion images captured during a community dwelling dietary study. Specif-

ically, we are interested in learning how many different environments an individual consumes food in. Our method clusters images by extracting features at both global and local scales using a pre-trained deep neural network. The variation in the number of clusters and images captured by different individual makes this a very challenging problem. Experimental results show that our method performs significantly better compared to several existing clustering approaches.

1.5 Saliency-Aware Class-Agnostic Food Image Segmentation

In Chapter 6, we propose a method to segment salient missing objects from a pair of images. Advances in image-based dietary assessment methods have allowed nutrition professionals and researchers to improve the accuracy of dietary assessment, where images of food consumed are captured using smartphones or wearable devices. These images are then analyzed using computer vision methods to estimate energy and nutrition content of the foods. Food image segmentation, which determines the regions in an image where foods are located, plays an important role in this process. Current methods are data dependent, thus cannot generalize well for different food types. To address this problem, we propose a class-agnostic food image segmentation method. Our method uses a pair of eating scene images, one before start eating and one after eating is completed. Extracting information via a pre-trained deep neural network from both the before and after eating images, we can food images by finding the salient missing objects without any prior information about the food class. We model a paradigm of top down saliency which guides the attention of the human visual system (HVS) based on a task to find the salient missing objects in a pair of images. Our method is validated on food images collected from a dietary study which showed promising results.

1.6 Contribution Of This Thesis

- We proposed a simple and effective method that uses reflectance to detect and remove shadows in color images. Although our method has very few parameters, it is able to achieve similar level of performance when compared to some of the existing deep learning and machine learning approaches.

- We analyzed shadow removal from a forensics perspective, where we designed a method for detection and localization of shadow removal from a color image using state-of-the-art shadow removal methods.We demonstrated the importance of building a custom tool for shadow removal detection as convention forensic tools fail to detect these shadow manipulations.

- We investigated the integrity of satellite images by proposing a method to detect and localize spliced objects in satellite images. Our method trains only on pristine data and does require prior knowledge of the nature of spliced objects. Our method also shows good performance in detecting and localizing spliced objects of various classes and sizes.

- We proposed a method to cluster eating scene images based on their eating environments. Dieticians can cluster images using our method for further analysis of diet quality thus reducing

- We proposed a class agnostic food image segmentation model by segmenting salient missing objects in a pair of images.Our method achieves better performance than many of the existing class agnostic approaches. Our proposed method will also be able to segment salient missing objects in other scenarios as well.

1.7 Publications From This Thesis

1. **S. K. Yarlagadda**, D.M Montserrat, D.Guera, C.J. Boushey, D. Kerr, F. Zhu "Saliency aware class agnostic food image segmentation", Submitted to ACM Transactions on Computing for Healthcare,*under review (minor revision).*

2. **S. K. Yarlagadda**, S. Baireddy, D. Güera, C. J. Boushey, Kerr DA, F. Zhu, "Learning eating environments through scene clustering", *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, May, 2020.

3. **S. K. Yarlagadda** and F. Zhu, "A Reflectance based method for shadow detection and removal", *IEEE Southwest Symposium on Image Analysis and Interpretation*, pp. 9-12. Las Vegas, NV, 2018.

4. **S. K. Yarlagadda**, D. Güera, D. Mas, P. Bestagini, F. Zhu, S. Tubaro, E. J. Delp, "Shadow removal detection and localization for forensic analysis", *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2019.

5. **S. K. Yarlagadda**, D. Güera, P. Bestagini, F. Zhu, S. Tubaro, E. J. Delp, "Satellite image forgery detection and localization using GAN and one-class classifier", *Proceedings of the IS&T Electronic Imaging*, vol. 2018, no. 7, pp. 214-1-214-9, Burlingame, CA, January 2018.

6. D. Güera, **S. K. Yarlagadda**, P. Bestagini, F. Zhu, S. Tubaro, E. J. Delp, "Reliability map estimation for CNN-based camera model attribution", *Proceedings of IEEE Winter Conference on Applications of Computer Vision*, pp. 964-973, Lake Tahoe, NV, February 2018.

7. Y. Wang, J. Ribera, C. Liu, **S. K. Yarlagadda** and F. Zhu, "Pill recognition using minimal labeled data", *Proceedings of IEEE International Conference on Multimedia Big Data*, pp. 346-353, Laguna Hills, CA, 2017.

8. S. Fang, **S. K. Yarlagadda**, Y. Wang, F. Zhu, C. Boushey, D. Kerr and E. Delp, "Image based dietary behavior and analysis using deep learning", *Mini symposium at the International Conference of the IEEE Engineering in Medicine and Biology Society*, July 2018, Honolulu, HI.

9. J Horvath, D. Güera, **S. K. Yarlagadda,** P. Bestagini, F. Zhu, S. Tubaro , E. J. Delp, "Anomaly-based manipulation detection in satellite images", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, July 2019.

10. D. Mas Montserrat, H. Hao, **S. K. Yarlagadda**, S. Baireddy, R. Shao , J. Horvath, E. Bartusiak, J. Yang ,D. Güera, F. Zhu , E. J. Delp, "Deepfakes detection with automatic face weighting", *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, July 2020.

11. D. Mas Montserrat, J. Horváth, **S. K. Yarlagadda**, F. Zhu, and E. J. Delp. "Generative autoregressive ensembles for satellite imagery manipulation detection", Submitted to IEEE International Workshop on Information Forensics and Security, 2020.

2. DETECTION AND REMOVAL OF SHADOWS FROM RGB IMAGES

2.1 Overview

Shadows are ubiquitous. They are formed when light is partially or fully occluded by objects. Shadows provide information about lighting direction [14], scene geometry and scene understanding [15] in images and are crucial for tracking objects [16] in videos. They also form an integral part of aerial images [17]. However, shadows can also complicate tasks such as object detection, feature extraction and scene parsing [18].

There have been many methods proposed to detect shadows from images and videos [16, 18–23]. In this chapter we focus on detecting shadows from color images. With the recent boom in data driven approaches, machine learning based methods have been applied to detect shadows [18, 20, 21]. In [18] Conditional Random Fields consisting of 2490 parameters are used to detect shadows in gray scale images using features such as intensity, skewness, texture, gradient similarity etc. In [20] Convolutional Neural Networks consisting of 1000's of parameters are used to detect shadows. In [19] intensity information around edges is used to detect shadow boundaries. In [21] image is first segmented and various classifiers are used to detect regions similar in color and texture by comparing different segments with each other.

In this chapter we propose a non-training based shadow detection method which requires fewer parameters, yet achieves high accuracy compared to previous methods [18, 20, 21]. We differ from [21] in the features and classifiers used for comparing regions and also in the approach of using these comparisons to obtain the shadow mask. Every surface is characterized by two features: its reflectance and its texture. When a shadow is cast on a surface its illuminance reduces, but its reflectance remains

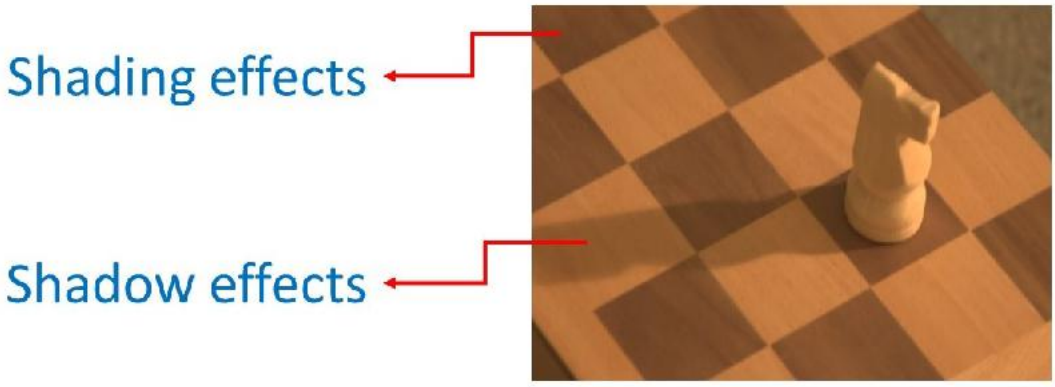
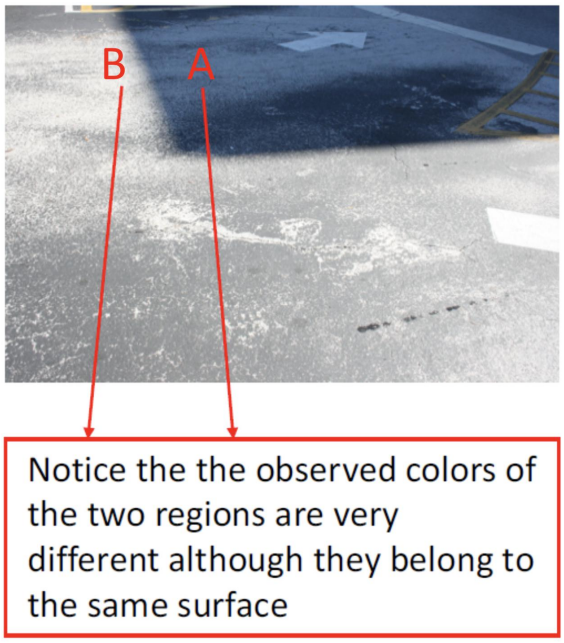


Fig. 2.1.: Note that both the surfaces are dark but one is due to shadow and another is due to shading. Such examples complicate shadow detection but can be solved using neighborhood information

the same. Due to the reduction in the illuminance, there will also be some loss in texture information. By examining a surface, it is difficult to tell whether it is dark due to the effects of shadow or shading. An example of this is given in Figure 2.1. By comparing surfaces with each other we can detect shadows with greater confidence. Hence, by pairing different regions of an image based on their reflectance, texture and illumination characteristics we can detect shadows efficiently.

2.2 Shadow Detection

Our goal is to group different regions of an image based on their reflectance, texture and illumination characteristics. To group pixels with similar properties into different regions, we first segment an image using the Quickshift method [24] with a Gaussian kernel size of 9. Our assumption is that a single segment should contain pixels with similar reflectance and illumination. An example of segmentation result is shown in Figure 2.2. In the subsections below we explain how we design the reflectance, texture and illumination classifiers to label each segment as shadow or non-shadow.



(a) Original Image

(b) Segmented Image

Fig. 2.2.: An example of a test image segmented using Quickshift with a kernel size of 9. The segmentation correctly separates the boundaries between the shadow and non-shadow regions.

2.2.1 Reflectance classifier

Consider the illumination model used in [21],

$$I_i = (t_i cos(\theta) L_d + L_e) R_i \tag{2.1}$$

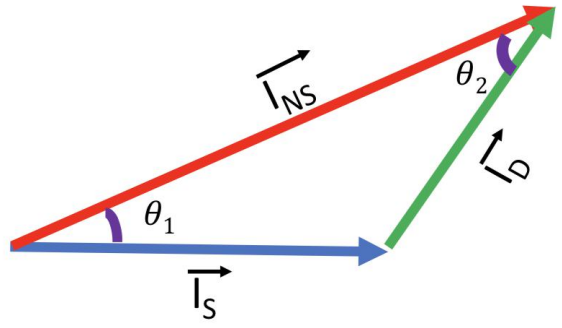
where I_i is the vector representing the i^{th} pixel in RGB space, L_d and L_e are vectors representing the direct light and reflected light from the environment, respectively. θ is the angle between direct light and surface normal and R_i is the reflectance vector. The value of t_i indicates whether the pixel belongs to shadow or non-shadow segment. When $t_i = 0$ the pixel is in the shadow segment and vice versa. Two segments belonging to the same surface but under different illumination can be modeled as $t_i = 0$ for all the pixels in the shadow segment and $t_i = 1$ for all the pixels in the non-shadow segment. Assuming that direct light and environment light are constant in magnitude and direction over the two segments, we can see that the reflectance property of the surface remains constant in both cases. Taking the respective median of all pixels in RBG color space in each of the segments, we have the following,

$$I_{NS} - \text{Median color of non-shadow segment}$$

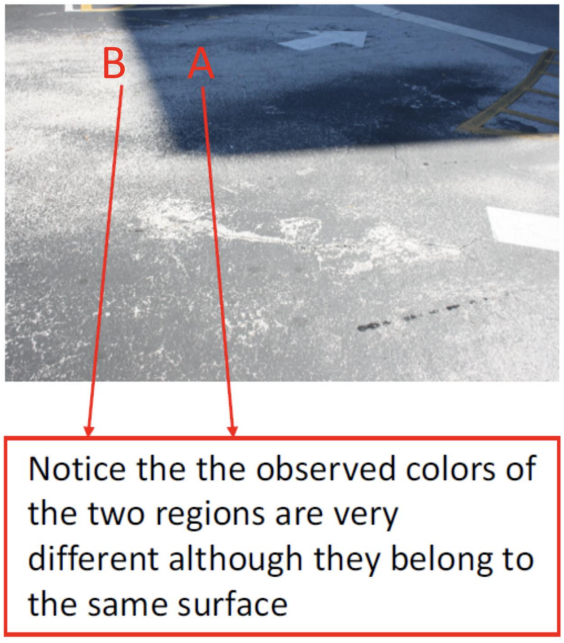
$$I_S - \text{Median color of shadow segment}$$

$$I_D = I_{NS} - I_S = (cos(\theta)L_d)R^{median}$$

In the case when L_e is similar to L_d in terms of chromaticity, the angle between the vectors I_D and I_{NS} should be zero. However, in practice L_e differs from L_d, hence these two vectors will have a small angle provided they are of the same material and a large angle if they are of different material with different reflectance properties. By thresholding the angle between the color vectors I_D and I_{NS}, we can decide whether two segments with different illumination conditions belong to the same material. We call this the "angle criterion." Notice that we don't look at the angle between I_{NS} and I_S because in the case where L_e is significantly different from L_d in terms of chromaticity, the angle between these two vectors will be very large even if they represent the same surface. We set the angle threshold to be 10°. An example of this case is illustrated in Figure 2.3.



(a)



(b)

Fig. 2.3.: An illustration of color vectors is given in 2.3a. In 2.3b visually region A (the shadow region) on the road appears blue, not gray. This is due to the large chromaticity difference in direct light L_d and the reflected light L_e leading to a large θ_1. However, the angle between I_D and I_{NS} (θ_2) will be small so that we can classify region A and region B as shadow non-shadow pairs.

2.2.2 Luminance Classifier

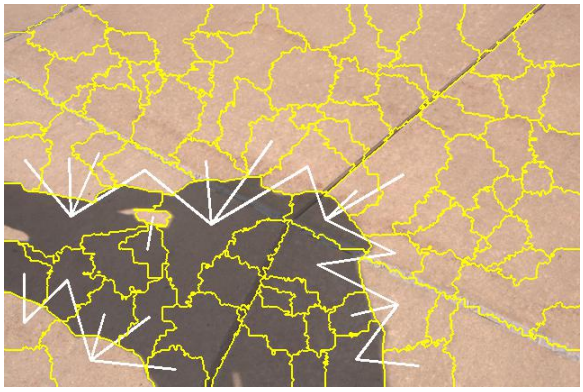
Shadows are formed when direct light is partially or fully occluded and hence have lower illumination. The decrease in illumination depends on the relative intensities

of L_d and L_e. A large decrease in illumination intensity darkens the shadow. To build an effective luminance classifier, we need to be able to detect the decrease in illumination and be able to attribute that decrease to obstruction of light and not due to some noise. In order to model this, we look at the luminance values of all pixels in the LAB color space. We compute the median luminance of all segments in the LAB space and compute the histogram of the median luminance values. The peaks of the histogram give us an estimate of the number of different illumination regions in the image.

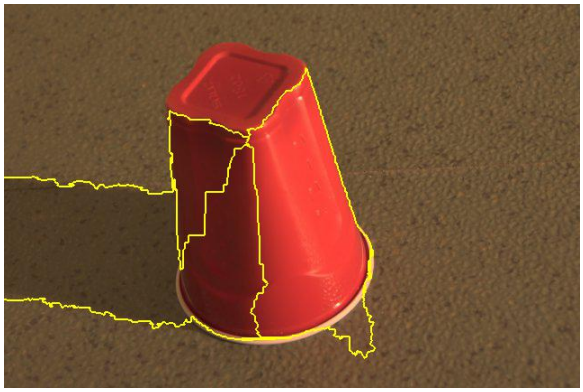
We then split the image into regions by grouping segments based on their proximity to the peaks. Segments within the same region are not compared because they have similar illumination intensity while segments from different groups are allowed for comparison to detect shadows. This step is useful because it adaptively groups segments into regions with similar illumination. An example of grouping segments into regions based on their luminance is shown in Figure 2.4. In addition to the grouping criteria, for two segments to be shadow non-shadow pairs, the ratio of their median luminance T in LAB space has to be above the threshold of 1.2 in order to avoid comparing segments with similar illumination. T can be anywhere between 1 and ∞ and the closer it is to 1 the closer the illumination intensities of the two segments are. Shadow non-shadow pairs will have a high values of T compared to segments with similar illumination intensities.

2.2.3 Texture classifier

Since shadow and the corresponding non-shadow segments are of the same material their texture characteristics will be similar. However, due to the reduction in illumination intensity of shadow segments, some texture information is lost. To capture this phenomenon, we look for texture similarity between the segments under comparison provided that their T is not very high, because if its high a lot of texture information would have been lost. We compute the Earth Mover Distance between



(a) Segmentation



(b) Luminance clustering

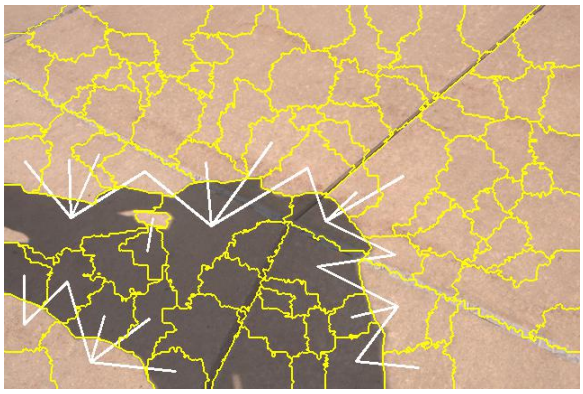
Fig. 2.4.: The segmented image in 2.4a is grouped using the luminance classifier and the result is shown in 2.4b.

the histograms of the texton maps [25] of both segments and threshold it to find whether the two segments have similar texture. However, if T is greater than 2.4 we do not compare them for texture similarity as a lot of texture information is lost in the shadow segment due to the decrease in illumination.

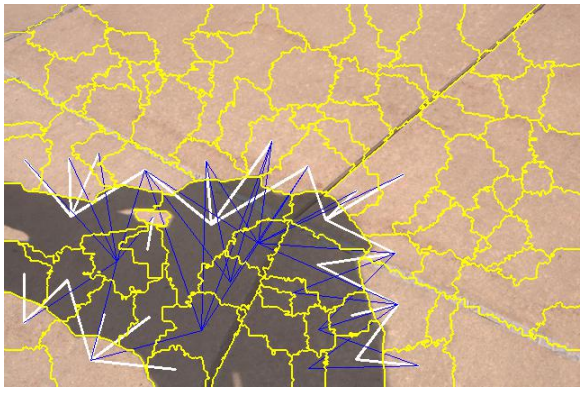
2.2.4 Implementation

In this subsection, we describe how we use the above three classifiers to detect shadow non-shadow segment pairs. Each segment is compared to its neighboring segments using the reflectance, texture and luminance classifiers discussed above. If

all the classifiers label the pair as a shadow non-shadow pair, we store that connection. We use these connections to connect more segments. For every shadow non-shadow pair, we take all the non-classified neighbors of the shadow segment and compare them to non-shadow segment using the above classifiers. We repeat this process 3 times. The reason is that some shadow segments may have neighbors which are also shadow segments themselves. Such segments will not be detected in the first iteration. In order to connect them to the already labeled shadow segments, we repeat the process by using the information obtained from the initial connections. The process is illustrated in Figure 2.5.



(a) First Iteration



(b) Second Iteration

Fig. 2.5.: The connections obtained by first iteration are marked with the white lines (2.5a) and the connections obtained by second iteration are marked by the blue lines (2.5b).

2.2.5 Refinement

The above implementation detects shadow non-shadow pairs with similar reflectance, texture and different luminance but does not put any constraint on how bright the shadow segment should be. Without such constraint, two very bright segments can be misclassified as shadow non-shadow pairs. In order to avoid this, we limit the shadow region to have a gray scale value lower than the Otsu threshold of the image. We segment the image again with a Gaussian kernel of size 3 (smaller than the Guassian kernel used in the initial segmentation) and look for segments which contain shadow pixels using the initial shadow mask. A finer segmentation mask leads to better modeling of the shadow non-shadow boundaries. Given a segment contains shadow pixels, if more than 70% of pixels in that segment have a gray scale value less than the Otsu threshold, we label the entire segment as a shadow segment and if not we label the entire segment as non-shadow.

2.3 Experimental Results

The proposed method is evaluated on two publicly available datasets, the UIUC dataset [21] and the UCF dataset [18].

2.3.1 UIUC Dataset

The UIUC dataset consists of 108 images with shadows, out of which 32 images have been used for training and 76 for testing by [21]. We have evaluated our method on the 76 test images. In addition to computing the per class accuracy, we also show the Balanced Error Rate (BER) for our method which is computed as the following,

$$\text{BER} = 1 - \frac{1}{2}\left(\frac{TP}{TP + FN} + \frac{TN}{TN + FP}\right) \tag{2.2}$$

where FP is False Positives, FN is False Negatives, TP is True Positives and TN is True Negatives. The lower the BER the better the method. BER is used because

there are fewer shadow pixels than non-shadow pixels in the images. The results of our methods and others are shown in table 2.1. We have achieved the highest accuracy detecting shadows and a very close BER compared to [20] which has the smallest BER of all three methods.

Table 2.1.: Results Our Proposed Method Compared to Other Methods On UIUC Dataset

Methods	Shadows	Non-Shadows	BER
Unary + Pair-wise([21])	.716	.952	.166
ConvNet([20])	.847	.955	**.099**
Our method	**.906**	.855	.119

2.3.2 UCF Dataset

The UCF dataset is also widely used for testing shadow detection methods. It consists of 355 images which are more diverse and complex than the UIUC dataset. In [18] 120 images were used for testing. We have tested our method on 236 images. Out of the 236 images, for 162 of them we have followed the proposed method, but for 74 images from OIRDS [26] dataset we have chosen a threshold of .35 instead of using the Otsu threshold for limiting the gray scale of the shadow pixels. This is because OIRDS dataset contains aerial images with very dark shadow regions. The results are reported in Table 2.2 and comparisons to other methods are shown in Table 2.3. In comparison to other methods, our method achieved the highest accuracy in detecting shadows and also has the best BER.

2.4 Shadow Removal

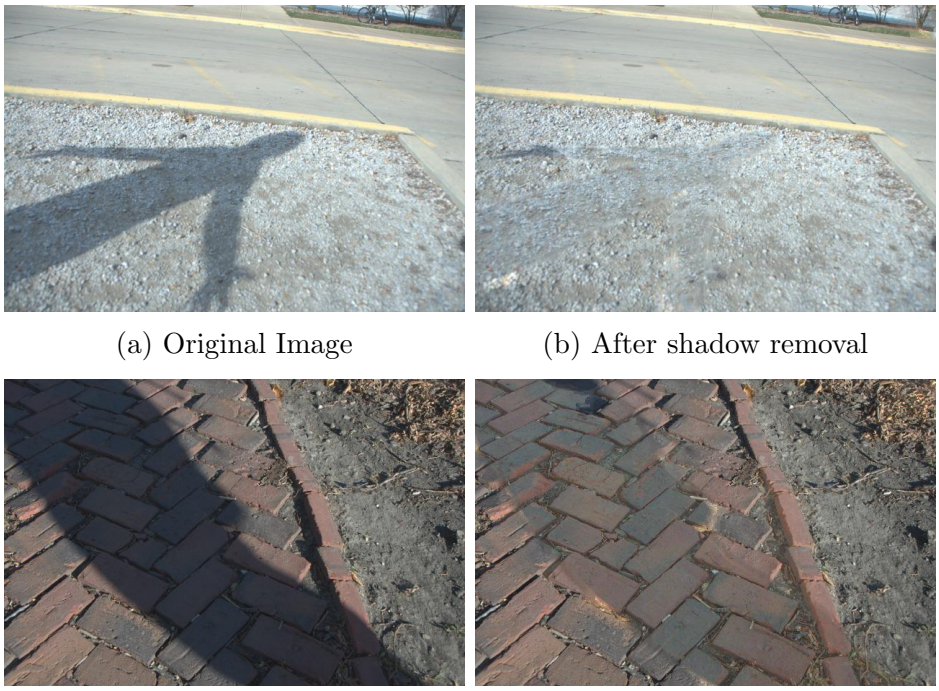
To remove shadows we follow the same approach as described in [21]. Some examples of shadow removal are shown in Figure 2.6.

Table 2.2.: Detection Confusion Matrices of Our Proposed Method On UCF Dataset

74 images from **OIRDS** dataset	Shadow	Non Shadow
Shadow	.899	.101
Non - Shadow	.116	.884
162 images from UCF dataset	Shadow	Non Shadow
Shadow	.922	.078
Non - Shadow	.191	.809

Table 2.3.: Detection Confusion Matrices of Our Proposed Method Compared to Other Methods On UCF Dataset

Methods	Shadows	Non-Shadows	BER
BDT-BCRF [18]	.639	.934	.2135
Unary + Pair-wise([21])	.733	.937	.165
ConvNet([20])	.780	.926	.147
Our method	**.920**	.827	**.1265**



(a) Original Image (b) After shadow removal

(c) Original Image (d) After shadow removal

Fig. 2.6.: Sample shadow removal results.

3. SHADOW REMOVAL DETECTION AND LOCALIZATION FOR FORENSIC ANALYSIS

3.1 Overview

Image editing tools are widely available. It is possible to download professional image manipulation tools (e.g., Photoshop), to use image editing operations directly from web interfaces (e.g., Pixlr), or even more easily to automatically forge a picture using completely unsupervised tools (e.g., FaceSwap). If maliciously edited images are shared online or distributed through broadcast channels, their impact in terms of opinion formation and fake news distribution can cause serious social consequences.

Many blind image forensic tools have been developed in the literature through years [27–29]. Among these techniques, many focus on verifying image digital integrity. These methods typically exploit statistical traces left by alterations of digital signals and can be used for a wide variety of applications (e.g., detecting the originating device or camera model [30, 31], general forgeries [32, 33], resampling [34–36] and multiple compressions [37, 38]). The main issue behind many of these methods is that they rely on a strict set of assumptions that cannot always be verified and they suffer from multiple editing operations being applied altogether. Many methods can be fooled if "laundering" operations that scramble image statistics are used to edit images (e.g., small resizing and cropping, subtle global operations, and recompressions).

Other forensics techniques rely on verifying image physical integrity. This means detecting whether an image is authentic by checking physical consistencies in reflections [39], lightning [40, 41], shadows [42–44], and other constraints that must be verified in a real-world photoshoot. As an example, by knowing the direction of light illuminating a scene, it is possible to estimate shadow directions. In the same way,

it is possible to estimate whether all objects in the scene present a shadow coherent with the other ones [42]. The drawback of this technique is that they are often semi-supervised (e.g., the analyst should manually check for shadow cast, lightning directions, etc.). However, they are innately robust against laundering. Indeed, as long as the image semantic content remains unchanged, it is still possible to verify physical inconsistencies despite resizing, rotations, or image re-compression operations. To fool these techniques, an expert manipulator must take into account the laws of physics, and retouch the picture accordingly.

In the past fooling physical integrity detection was considered a challenging task, nowadays this might be only partly true. Indeed, thanks to the increasingly advancement in machine learning and signal processing, many image editing operations can be used in an almost automatic fashion, not always requiring the hand of a professional. Among these, many methods for shadow removal have been proposed in the literature [21, 45–48]. These can be readily used to remove incorrectly cast shadows from edited images to fool physical integrity detectors leveraging shadows to assess image authenticity [43, 49].

In this chapter, we propose a method to detect whether an automatic shadow removal technique has been used to edit an image. If shadow modification is detected, we also propose a way to partly recover the location of the missing shadow. In doing so, we can help shadow-based image forensics detectors. The proposed solution is based on the use of a specific class of convolutional neural network (CNN) known as conditional generative adversarial network (cGAN). The architecture is trained on purpose for the problem under analysis on a dataset of images whose shadows have been removed with a very accurate yet easy-to-use state-of-the-art technique [48].

The rest of the chapter is structured as follows. Section 3.2 provides the reader with some background on shadow removal algorithms, and provides the formal problem definition. Section 2.2 is devoted to the explanation of the proposed methodology for shadow removal detection and localization. Section 3.4 contains all the details of the performed experimental campaign.

3.2 Background and problem statement

In this section we introduce the reader to state-of-the-art techniques for automatic shadow removal. Then, we provide the formal definition of the shadow removal detection and localization problem.

3.2.1 Shadow Removal

The problem of shadow removal involves inconspicuously relighting the shadow pixels while leaving the non shadow pixels unchanged. Over the years, many methods have been proposed to address this problem [21, 45–48]. These methods can be classified as automatic [21, 46, 47] or user-aided [45, 48], and the criterion for this classification relies solely on how shadows are detected before removal. User aided methods rely on input from humans to detect shadows and then they proceed to remove shadows automatically. Automatic shadow removal methods aim to directly go from an input image to its shadow free counterpart. Both automatic and user aided methods have their potential downsides. For automatic methods, errors in shadow detection could severely hamper the effectiveness of shadow removal, while user aided methods could prove to be tedious.

In this chapter we choose to use the shadow-removal technique proposed in [48]. This choice is driven by the following considerations. Despite being a user aided method, it requires only two rough strokes from a user as input. This makes it very easy to use for non-expert image manipulators and the visual results are very pleasant. We use the authors own implementation [48] of the shadow removal method.

3.2.2 Problem Formulation

Let us define a natural image under analysis as $\mathbf{I}$. A pixel with coordinate (x, y) is denoted as $\mathbf{I}(x, y)$. Let us also define a shadow forgery mask $\mathbf{M}$, being a matrix the same size of the image, whose entries indicate which pixels are affected by the

shadow removal algorithm. In other words the sample with coordinate (x, y) in $\mathbf{M}$ is defined as

$$\mathbf{M}(x,y) = \begin{cases} 1, & \text{if a shadow has been deleted in } \mathbf{I}(x,y). \\ 0, & \text{otherwise.} \end{cases} \tag{3.1}$$

The goal of our method is twofold. First, to detect whether any shadow has been removed in image $\mathbf{I}$. Second, if a shadow has been removed, to estimate the locations of pixels that originally contained shadow traces. In order to solve both problems, we compute $\hat{\mathbf{M}}$ being an est $\mathbf{M}$. If $\hat{\mathbf{M}} \approx \mathbf{0}$, we conclude that no shadows have been removed and the image is authentic. Conversely, if $\hat{\mathbf{M}} \not\approx \mathbf{0}$, we conclude that the image has been edited, and the original shadow was located in pixel at locations $\{(x, y) : \hat{\mathbf{M}}(x, y) = 1\}$.

3.3 Proposed Method

Our proposed method for shadow removal detection and localization is based on the following pipeline: (i) the image $\mathbf{I}$ under analysis is adapted to fit a given resolution; (ii) a convolutional neural network (CNN) trained to generate a heatmap that indicates the likelihood of shadow removal traces for each patch pixel is used; (iii) the heatmap is thresholded to estimate $\hat{\mathbf{M}}$ and to make a decision concerning shadow detection and localization. In the following, we provide a detailed description about each step of the proposed procedure.

3.3.1 Image Size Adaptation

One of the problems that arises when processing high resolution images with CNNs, is that image resolution rarely matches the CNN input size. Therefore, a common strategy consists is to split the full resolution image into smaller patches,

analyze each patch separately, and finally aggregate the results. This rationale has been already successfully used by other recently proposed forensic detectors [50, 51].

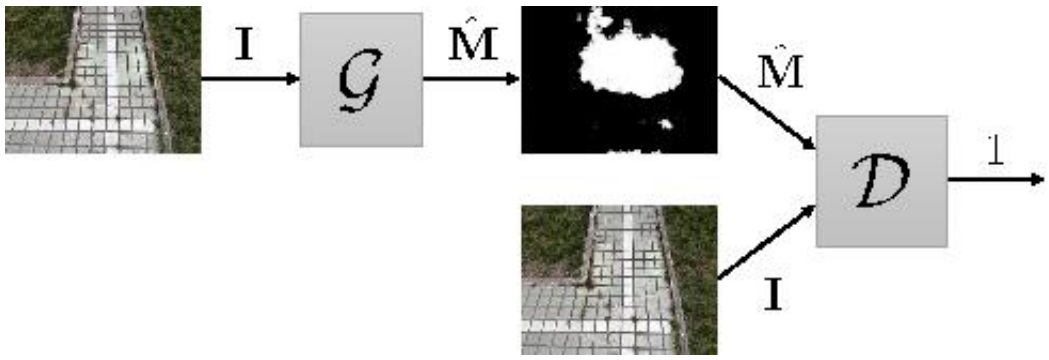
One of the issues in doing this, is the trade-off between accuracy and computational complexity. To obtain a fine-grained solution, patches must be extracted with a large overlap. This increases the number of patches to cover the whole image area, thus a higher computational time.

In order to compromise and reduce the required computational power, we propose to use a slightly different solution. As a matter of fact we train our CNN in order to work on images that have been downsized by a factor of almost 2 with respect to their original resolution. This has two major positive effects. First, the CNN becomes naturally resistant to resize laundering. Second, when a high resolution image is under analysis, the analyst can extract larger (thus less) patches, resize them for CNN analysis, and finally upsample the results back to the original image size.

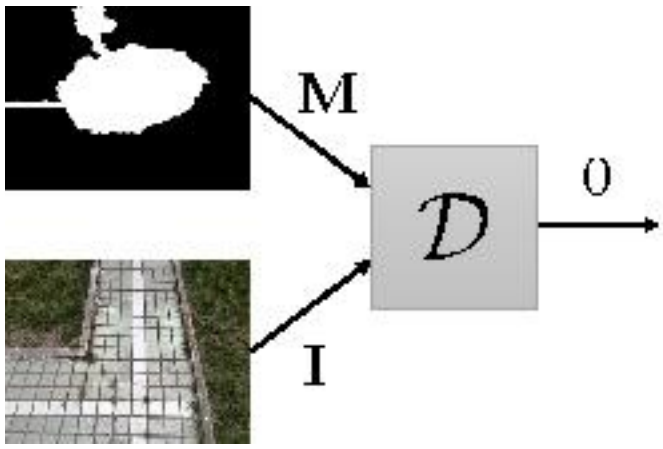
3.3.2 CNN Architecture

To estimate $\hat{\mathbf{M}}$ from an image $\mathbf{I}$ resized to the correct CNN resolution, we learn a mapping function that goes from the resized $\mathbf{I}$ to $\hat{\mathbf{M}}$ using a cGAN. This cGAN is based on pix2pix [52]. The architecture of the cGAN is composed by two different CNNs namely, the Generator G and the Discriminator D, coupled together as shown in Figure 4.3.

Generator G is a U-net [53] containing more than 10 convolutional layers with skipped connections. This network turns the input image into the estimated mask as defined $\hat{\mathbf{M}} = G(\mathbf{I})$. Discriminator D is a simpler and shallower network composed by a series of convolutional, pooling and fully connected layers. This network acts as a binary classifier on shadow masks, trying to distinguish whether they are a ground truth mask, or a mask estimated by the generator G. The discriminator is trained to output either 1 or 0 depending on the nature of the inputs, i.e, $\mathcal{D}(\mathbf{I}, \mathbf{M}) = 1$



(a) Generator Training

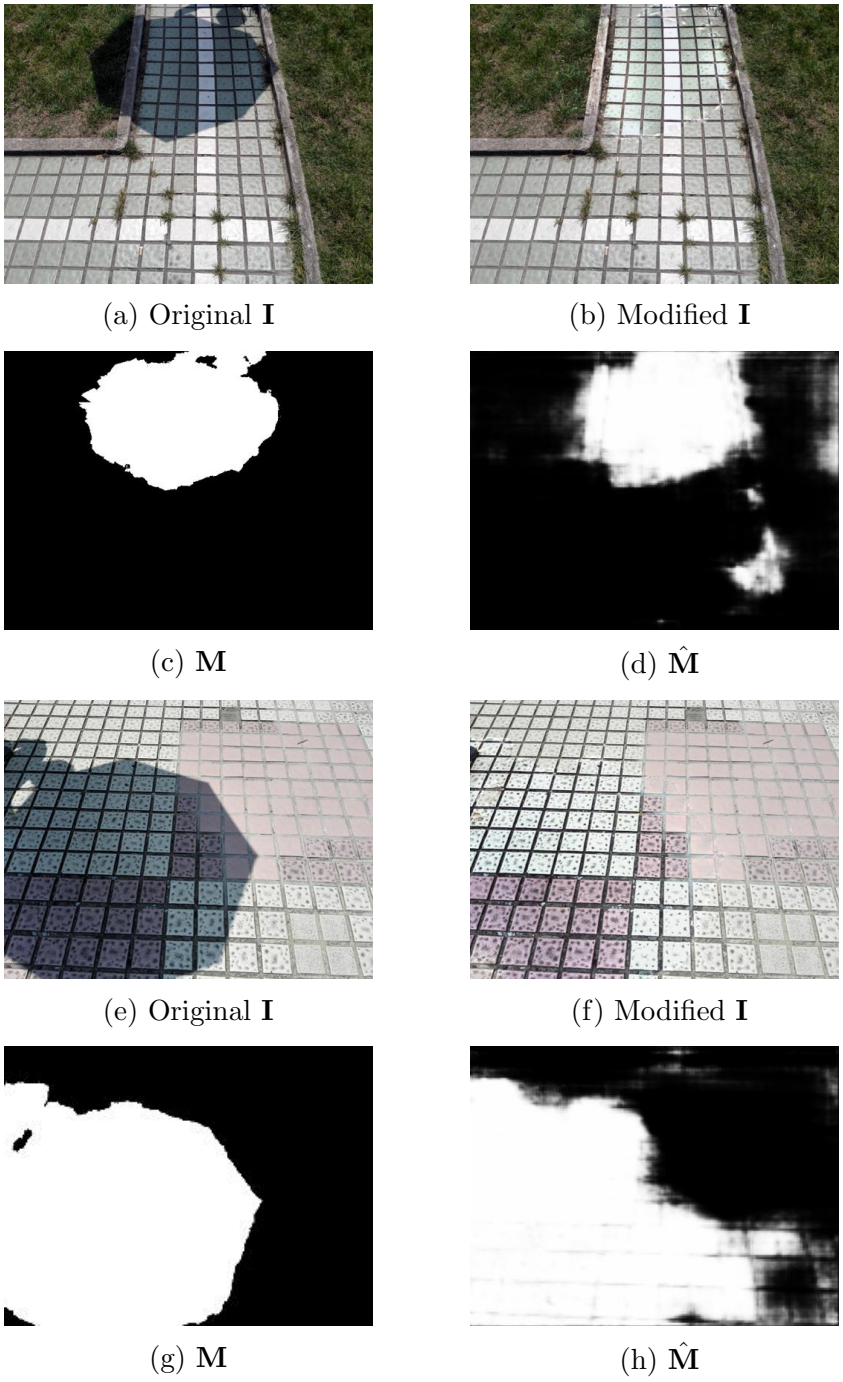


(b) Discriminator Training

Fig. 3.1.: cGAN overall architecture. The generator (a) is trained to fool the discriminator. The discriminator (b) is trained to detect ground truth and estimated masks.

and $\mathcal{D}(\mathbf{I}, \hat{\mathbf{M}}) = 0$. Both the generator and discriminator are coupled together using a loss function $\mathcal{L}_{CGAN}(G, D)$ (please refer to [52] for details on $\mathcal{L}_{CGAN}(G, D)$). In addition to $\mathcal{L}_{CGAN}(G, D)$ the generator is also trained to reduce a reconstruction loss between the predicted mask $\hat{\mathbf{M}}$ and the true mask $\mathbf{M}$, denoted as $\mathcal{L}_R(\mathbf{M}, \hat{\mathbf{M}})$. The loss function of cGAN denoted by $\mathcal{L}$, is defined as

$$\mathcal{L} = \mathcal{L}_{cGAN} + \lambda \cdot \mathcal{L}_R \tag{3.2}$$



(a) Original **I** (b) Modified **I**

(c) **M** (d) $\hat{\mathbf{M}}$

(e) Original **I** (f) Modified **I**

(g) **M** (h) $\hat{\mathbf{M}}$

Fig. 3.2.: Example of shadow removal from image **I**, ground truth shadow mask **M** and estimated shadow mask $\hat{\mathbf{M}}$.

By coupling the two loss functions as shown in Eq.3.2, we force the generator to not only generate $\hat{\mathbf{M}}$ that is close to **M** but also fool the discriminator in the process.

This additional constraint results in a G that better maps $\mathbf{I}$ to $\mathbf{M}$ as opposed to just training G to reduce $\mathcal{L}_R$ without the discriminator D.

We chose $\mathcal{L}_R$ to be the binary cross-entropy (BCE) between $\hat{\mathbf{M}} = G(\mathbf{I})$ and $\mathbf{M}$. This is different with respect to the classic pix2pix network, which makes use of $l1$-norm. However, as our goal is to estimate a binary mask, cross-entropy seems like a more natural choice (as we verify in the results presentation).

Once the network has been trained, the discriminator is not considered anymore, and the generator is used to turn new images under analysis $\mathbf{I}$ into estimated shadow masks as $\hat{\mathbf{M}} = \mathcal{G}(\mathbf{I})$.

3.3.3 Shadow Removal Detection and Localization

Depending on the image size adaptation strategy, one might need to splice together (with possible overlaps) all estimated masks $\hat{\mathbf{M}}$ coherently with the image patch extraction policy. If the image under analysis already fit the network input size, there is no need to perform additional steps and the estimated mask $\hat{\mathbf{M}}$ can be directly used. However, as we did not constrain the network output to be boolean, the mask $\hat{\mathbf{M}}$ is estimated in a real domain. To construct a binary mask, we need to threshold $\hat{\mathbf{M}}$ using a value Γ, which can be learned upon a validation set of images. An example of original image, manipulated image, ground truth mask, and estimated mask $\hat{\mathbf{M}}$ is reported in Figure 3.2.

3.4 Experiments and Results

In this section we describe our experimental evaluation. We first describe the image dataset. We then report details about the use CNN training policy. Finally, we present the achieved numerical results.

3.4.1 Image Datasets

To correctly evaluate the proposed method, we constructed a dataset containing both natural non-manipulated images and image whose shadows have been removed. We started with the publicly available Image Shadow Triplets Dataset (ISTD) proposed in [47]. ISTD consists of 1870 color image pairs from 135 different natural scenes. Each image pair is defined as $\left\{\mathbf{I}^{S}, \mathbf{I}^{SF}\right\}$, where $\mathbf{I}^{S}$ denotes an image with a shadow, and $\mathbf{I}^{SF}$ denotes a shadow-free image depicting the same scene of $\mathbf{I}^{S}$. Each image has a resolution of 640×480 pixels. A couple of examples are shown in Figure 6.1.

For each pair, we used the selected shadow removal [48] for each image $\mathbf{I}^{S}$ to obtain the manipulated shadow-free image denoted by $\hat{\mathbf{I}}^{SF}$. The binary forgery mask $\mathbf{M}^{S}$ is obtained by checking which pixels have been actually modified

$$\mathbf{M}^{S}(x,y) = \begin{cases} 0, & \text{if } \mathbf{I}^{S}(x,y) = \hat{\mathbf{I}}^{SF}(x,y), \\ 1, & \text{if } \mathbf{I}^{S}(x,y) \neq \hat{\mathbf{I}}^{SF}(x,y). \end{cases} \tag{3.3}$$

In our experimental scenario $\hat{\mathbf{I}}^{SF}$ is a forged image whose shadow has been removed, and its binary forgery mask is $\mathbf{M}^{S}$. The effectiveness of any forensic method is not only determined by how well it works on forged images but also on how effective it is on authentic images. In our scenario each non-manipulated image $\mathbf{I}^{SF}$ serves as an authentic image with a forgery mask $\mathbf{M}^{SF} = \mathbf{0}$.

In summary, our image dataset $\mathcal{D}$ consists of 1870 pairs of forged images with masks denoted by $\left\{\hat{\mathbf{I}}^{SF}, \mathbf{M}^{S}\right\}$, along with 1870 pairs of authentic images with masks denoted by $\left\{\mathbf{I}^{SF}, \mathbf{M}^{SF}\right\}$.

3.4.2 Training Strategy

Prior to training, the forged and authentic images along with their masks are resized to a resolution of 256×256 pixel to match the CNN input. The dataset

$\mathcal{D}$ is then split into training $\mathcal{D}_{train}$, validation $\mathcal{D}_{val}$ and test $\mathcal{D}_{test}$. Dataset $\mathcal{D}_{train}$ consists of 1130 forged images and the corresponding 1130 authentic images. Similarly $\mathcal{D}_{val}$ consists of 200 forged and 200 authentic images. Finally, $\mathcal{D}_{test}$ consists of the remaining 540 forged and 540 authentic images. While the entire dataset $\mathcal{D}$ contains images from 135 different scenes, the images for training and validation come from 90 of 135 different scenes, whereas images used for testing come from the remaining 45 scenes. In doing so, we ensure that the method is not merely learning how to distinguish between scenes.

In order to train the used CNN minimizing the proposed loss function, we used Adam optimizer [54] for both the discriminator and the generator. We set $\lambda = 10$, and trained the model for 200 epochs, selecting for test the model minimizing loss on $\mathcal{D}_{val}$.

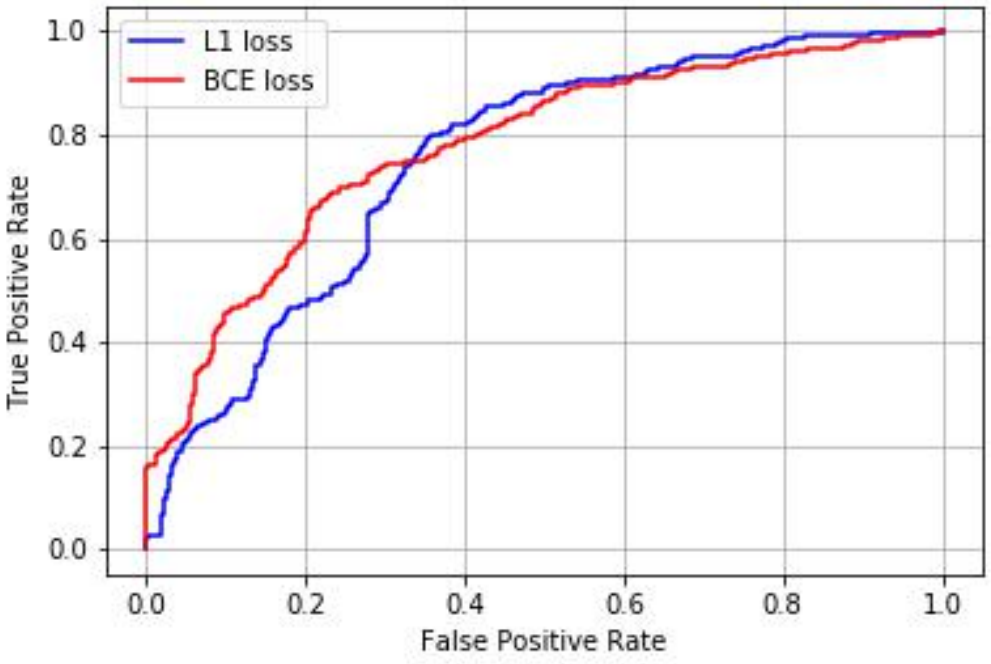


Fig. 3.3.: ROC showing shadow removal detection performance. BCE loss is the proposed one, whereas $l1$ loss is the standard pix2pix one.

3.4.3 Numerical Analysis

To show some examples of masks obtained by running our algorithm, Figure 6.1 reports two sets of images composed by the original picture with shadow, the edited picture whose shadow has been removed, the ground truth mask $\mathbf{M}$, and the estimated mask $\hat{\mathbf{M}}$. From this visual example it is possible to notice that our proposed method is able to correctly pinpoint shadow removal even on images that have been manipulated in a visually plausible manner. Moreover, the method provides information about the original shadow location, which can be helpful to perform some shadow-based forensic analysis.

To numerically evaluate our proposed method in terms of shadow-removal detection, we tested the constructed dataset. Specifically, after estimating each mask $\hat{\mathbf{M}}$, we computed its average value, and compared it against a threshold Γ to detect removed shadows. The idea is that the average value of $\hat{\mathbf{M}}$ should be zero (or approximately so) for non-manipulated pictures. Figure 3.3 reports the receiver operating characteristic (ROC) curve obtained by changing the value of the used threshold Γ. Figure 3.3 shows two curves: one obtained using the proposed binary cross-entropy (BCE) loss; one obtained using the conventional pix2pix $l1$-norm loss. It is possible to see that the proposed loss modification makes the algorithm more precise, as the achieved area under the curve (AUC) increase to 0.788 (using BCE) from 0.751 (using conventional $l1$-norm loss).

If an image has been detected as manipulated, we also want to estimate the manipulated pixels. To this purpose, we compared each thresholded estimated mask $\hat{\mathbf{M}}$ to the ground truth mask $\mathbf{M}$ in a pixel-wise fashion. By changing the threshold used to binarize $\hat{\mathbf{M}}$ we computed two ROC curves. In terms of localization, it is possible to note that AUC reaches 0.803 using the proposed BCE loss, whereas it only reached 0.701 using the conventional one.

As final experiment, we compared our method against a set of general-purpose image forensic techniques. Specifically, we considered the toolbox presented in [55],

which contains a set of more than 10 algorithms. Additionally, we also tested the technique proposed in [50], which is considered among the best splicing detection and localization tools to be used when no apriori information about the kind of manipulation are available. None one of these techniques was able to provide localization AUC > 0.6 on the proposed dataset. However, this behavior is somehow expected as none of these methods are specifically tailored to this type of manipulation and probably needs some more tuning.

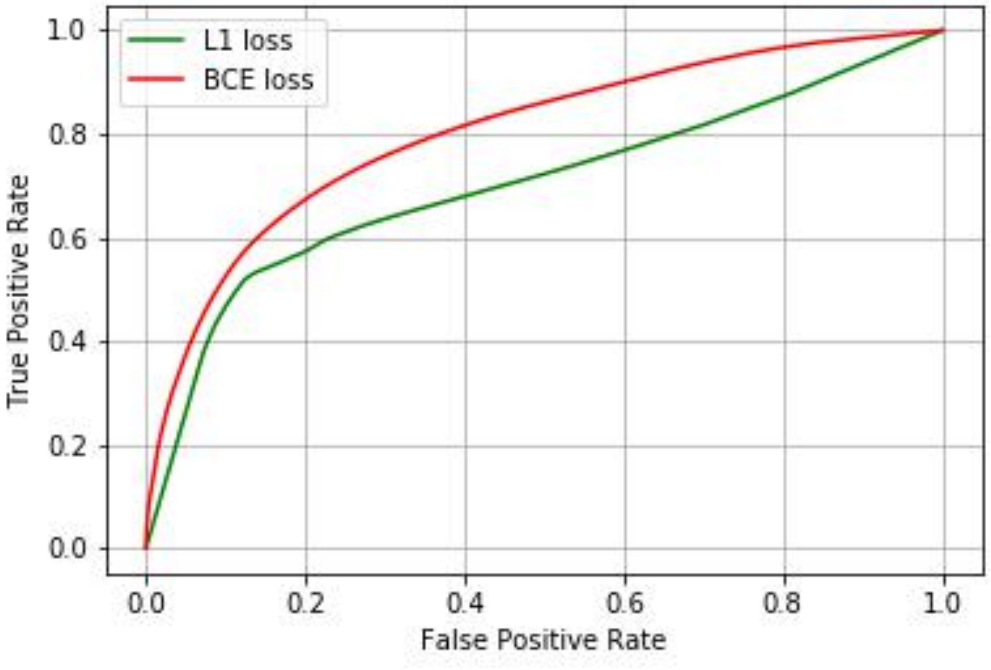


Fig. 3.4.: ROC showing shadow removal localization performance. BCE loss is the proposed one, whereas $l1$ loss is the standard pix2pix one.

4. DETECTION AND LOCALIZATION OF SPLICING IN SATELLITE IMAGES

4.1 Introduction

Ever since the birth of the Internet, the accessibility to images has become easier overtime. Internet has become an affordable and effective platform for distributing one's own images. User friendly software like Photoshop and Gimp can be used to generate a variety of image manipulations such as inpainting, copy-forge, splicing, etc. A combination of the above two scenarios is a perfect environment for producing doctored images, which when treacherously used can cause substantial damage. Therefore, it is of paramount importance to develop forensic methods to validate the integrity of an image. For this reason, over the years, the forensic community has developed several techniques for image authenticity detection and integrity assessment [27–29].

In addition to photographs captured with cameras and smartphones, other types of imagery are starting to be circulated, posing new problems for the forensic community. Indeed, current satellite imaging technology enables shooting high-resolution pictures of the ground. Due to the increased availability of satellites equipped with imaging sensors, overhead images are becoming popular. It is now possible to easily gather overhead images of the ground through public websites [56] and to buy custom image sets of specific locations and times. As any other kind of digital images, overhead pictures can also be easily forged. One question that needs to be addressed is whether these images are authentic. Cases of malicious overhead image manipulations have already been reported [57], [58]. The development of forensic methods tailored to the analysis of this type of imagery is considered to be urgent.

However, common image forensic techniques are often developed for consumer cameras, which strongly differ in their nature from satellite sensors (e.g., compression schemes, post-processing, sensors, etc.). Therefore, many accurate state-of-the-art forensic algorithms are bound to fail if blindly applied to overhead image analysis. Development of novel forensic tools for satellite images is paramount to assess their authenticity and integrity.

To fill the lack of ad-hoc forensic techniques for satellite images, the authors of [59] proposed an active method based on watermark embedding. Watermarks can then be exploited to detect possible doctored image regions. Unfortunately, this method can only be used if watermark is inserted at image inception time. More recently, the authors of [60] proposed a passive forensic method for overhead image analysis. This algorithm is based on machine learning techniques, but it can only localize image regions that have been inpainted. To the best of our knowledge, no specific algorithms for other kinds of satellite image forgeries have been proposed in the literature.

In this chapter, we propose an algorithm for satellite image forgery detection and localization. Specifically, we consider the situation in which pixels within a region of a satellite image are replaced to add or remove an object from the scene. Our algorithm works under the assumption that no forged images are available for training. Using a generative adversarial network (GAN), we learn a feature representation of pristine satellite images. A one-class support vector machine (SVM) is trained on these features to determine their distribution. Finally, image forgeries are detected as anomalies.

To validate the proposed method, we built a custom dataset of forged satellite images using different forgery sizes. Results in terms of forgery detection and localization are presented. Moreover, as the proposed algorithm works by analyzing images patch-wise, it is possible to strongly parallelize it to keep processing time at bay.

4.2 Problem Definition and Background

In this section we describe the problem formulation and notation used throughout the entire chapter. Following this, we provide some background concepts on autoencoders and convolutional neural networks.

4.2.1 Problem formulation

Consider an image $\mathbf{I}$ coming from a satellite. We can represent the pixel integrity associated with the image $\mathbf{I}$, as a binary mask $\mathbf{M}$ of the same size as the image in pixels. Each entry of $\mathbf{M}$ is a binary label 0 or 1, such that a pixel belonging to a forged area is assigned the label 0 and a pixel from an untampered area is assigned a 1. As forgery, in this chapter we consider an object insertion / removal through a copy-paste operation from a different source. This means that forged pixels do not belong to a satellite image but come from a different device (e.g., the picture of a plane acquired with a normal camera). Figure 4.1 shows an example of a pristine satellite image and a completely white (i.e., label 1) mask, as well as a forged image with the respective black and white mask localizing the forgery. Within this setup, our goal is twofold:

- *Tampering Detection*: given an image, detect whether it is pristine or forged.
- *Tampering Localization*: given a forged image, detect which are the forged pixels.

These two tasks can be accomplished by computing $\hat{\mathbf{M}}$ (i.e., an estimate of $\mathbf{M}$). If $\hat{\mathbf{M}}$ contains any entry different from 1, the image is detected as forged. Entries of $\hat{\mathbf{M}}$ whose values are 0 represent forged pixel positions.

4.2.2 Related Work

In this section we present a brief summary of autoencoders that are needed to follow this chapter. For a thorough review, we recommend the readers to refer to Chapter 14 of [61].

Autoencoders are neural networks that are trained to attempt to obtain an output equal to the input through a set of linear and non-linear operations that expand or reduce data dimensionality at some point in the network. They consist of two parts: the encoder A_e and decoder A_d. The output of the encoder is called feature vector or hidden representation, and we represent it as $\mathbf{h}$. In this chapter we work with autoencoders where the dimensionality of $\mathbf{h}$ is lower than the dimensionality of the input. This kind of autoencoders are known as undercomplete autoencoders. From now on, whenever we refer to autoencoders we refer to undercomplete autoencoders. Such an architecture forces the autoencoder to capture a salient representation of the input in a reduced dimensionality space.

Autoencoders are trained by minimizing through iterative procedures a loss value defined as

$$L = \mathcal{L}(x,\ A_d(A_e(x))), \tag{4.1}$$

where $\mathcal{L}(\cdot,\ \cdot)$ is the loss function computing some distance between its two arguments, x is the autoencoder input, and $A_d(A_e(x))$ is the output. In the special case where $\mathcal{L}$ is the mean squared error (MSE) loss, the autoencoder learns to perform a generalized non linear principal component analysis (PCA).

In this chapter we design our autoencoders using Convolutional Neural Networks (CNNs). CNNs have proven to be very successful in a variety of computer vision tasks such as object recognition [62], object detection [63], etc. They came to limelight in 2012 [64] when they produced stunning results in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [65]. Since then there has been an explosion in the application of CNNs to various other computer vision tasks and often resulting in new state of the art results.

(a) Pristine image (b) Pristine mask

(c) Forged mask (d) Forged Mask

Fig. 4.1.: Example of pristine (a) and forged (c) images **I** associated to their binary forgery masks **M** (b) and (d), respectively

When it comes to image forensics, the use of CNNs has been on the rise. Many forensic problems deal with non-linear and often difficult to model pipelines. Therefore, CNNs have proven to be successful in this area. The first works using CNNs in this area were focused on steganalysis [66–69]. Strictly concerning multimedia forensics, many other tasks have been considered. As a few examples, [70] deals with median filtering detection, [71] proposes the use of a constrained convolutional layer for forgery detection. In [51, 72–74], the problem of camera model identification

and its possible forgeries is explored. Double JPEG compression is also considered in [38, 75].

Convolutional neural networks usually consist of operations such as convolutions, batch normalization [76], local pooling, thresholding and non linear activations. These operations are stacked together and are tuned by minimizing a cost function at the output. Following, we describe some of the most commonly used layers:

- Convolutional: the input of this layer is convolved with a bank of filters whose response is learned through training. The input is typically a 3D structure, i.e., it has two spatial coordinates plus depth (e.g., an RGB image). The output is known as feature map.

- Max pooling: given an input x, a sliding window is used to extract the maximum value over it.

- Batch Normalization: given an input x, this layer normalizes x by imposing zero mean and unit variance. Details about this are explained in [76].

- Deconvolutional: this layer is the transpose of a convolutional layer. The output is obtained by convolving a zero-padded version of the input with a filter bank learned through training. The spatial dimensions of the output are greater than that of the input.

4.3 Method

In this section we elaborate on the details of our method to detect object insertion / deletion attacks in satellite images. In particular, the pipeline of our method is reported in Figure 4.2, and it is composed by the following steps:

- The color image under analysis is split into patches (either overlapping or not) of size 64×64 pixels.

- A adversarially trained autoencoder encodes the patches into a low dimensional representation called feature vector $\mathbf{h}$.

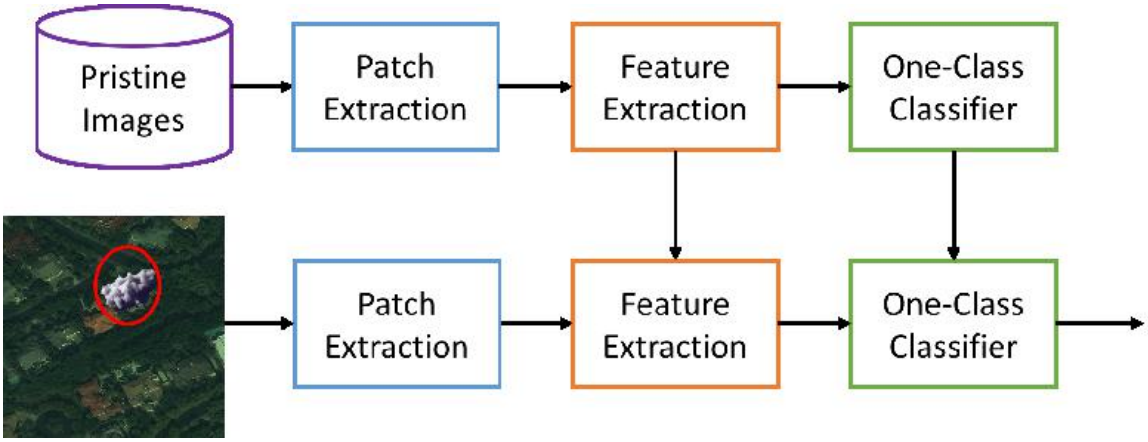


Fig. 4.2.: Pipeline of the proposed method. At training time, the feature extractor and one-class SVM learn their models from pristine images only. At testing time, forged areas are detected as anomaly with respect to the learned model.

- A one-class SVM fed with **h** is used to detect forged patches as anomalies with respect to features distribution learned from pristine patches.
- Once all patches are classified, a label mask for the entire image is obtained by grouping together all the patch labels.

The rationale behind the proposed solution is that autoencoders are able to capture a reduced dimensionality representation of the input data, still retaining important characteristic information, as shown in [77] for forensic purposes. Therefore, by training an autoencoder only on pristine data, we expect it to learn to extract features specific of original satellite images. Conversely, when it is tested on forged data, the extracted features should be strongly different from those obtained from pristine images. A one-class SVM trained on pristine features only can then be used to discriminate between features coming from pristine and forged images. Following, we report a detailed explanation of each step of the proposed pipeline.

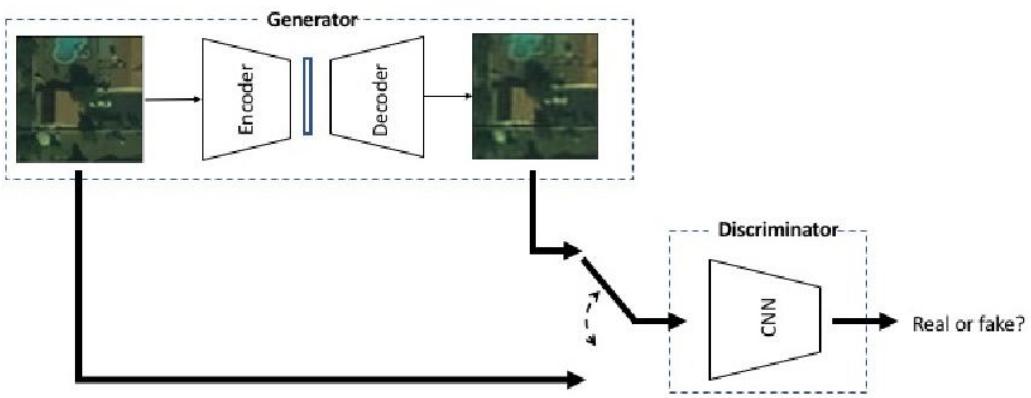


Fig. 4.3.: Architecture of the used GAN.

4.3.1 Patch Extraction

The given image $\mathbf{I}$ is split into regular patches $\mathbf{P}_k$, where $k \in [1, K]$ is the patch index, and K is the total amount of patches. Patches can be either overlapped or not depending on the selected trade-off between detection accuracy and computational complexity (i.e., overlapping leads to more patches to analyze but more accurate results).

4.3.2 Feature Extraction

Every patch $\mathbf{P}_k$ is fed to the autoencoder A which consists of two parts: the encoder A_e and decoder A_d. Both A_e and A_d are made of convolutional and deconvolutional neural networks respectively. They are symmetric in terms of the number of layers. The architecture of A_e has been selected following the same rationale of [78] and it is as follows:

- *conv1*: convolution layer with 16 filters each of size (6,6) with stride 1.
- *conv2*: convolution layer with 16 filters each of size (5,5) with stride 2.
- *conv3*: convolution layer with 32 filters each of size (4,4) with stride 2.
- *conv4*: convolution layer with 64 filters each of size (3,3) with stride 2.
- *conv5*: convolution layer with 128 filters each of size (2,2) with stride 2.

All convolutional layers except *conv5* are followed by batch normalization. All the convolution layers are activated using a linear function. The output of *conv5* is the feature vector $\mathbf{h}$, a 2048 dimensional vector and has a much lower dimension than that of the input which is 12288 dimensional. The architecture of A_d is as follows:

- *dconv1*: deconvolution layer with 64 filters each of size (2,2) with stride 2.
- *dconv2*: deconvolution layer with 32 filters each of size (3,3) with stride 2.
- *dconv3*: deconvolution layer with 16 filters each of size (4,4) with stride 2.
- *dconv4*: deconvolution layer with 16 filters each of size (5,5) with stride 2.
- *dconv5*: deconvolution layer with 3 filters each of size (6,6) with stride 1.

Every deconvolutional layer is followed by batch normalization except *deconv5*. *Deconv5* has a hyperbolic tangent activation where all other deconvolution layers have linear activations. The output of *deconv5* is the output of the autoencoder. Once the autoencoder is trained on pristine image patches, we use it as feature extractor to compute the feature vector $\mathbf{h}_k = A_e(\mathbf{P}_k)$ from each image patch $\mathbf{P}_k$.

Conventionally A can be trained using stochastic gradient descent to minimize mean squared loss between input (i.e., $\mathbf{P}_k$) and output (i.e., $A_d(A_e(\mathbf{P}_k))$) . However better results can be achieved when we follow an adversarial framework for training the autoencoder. In [79] the authors established a framework of min-max adversarial game between two neural networks, namely the generator and discriminator, and

such networks are called Generative Adversarial Networks. As shown in Figure 4.3, the discriminator aims to accurately discriminate between patches from real satellite images and patches created by the generator. The generator on the other hand aims to mislead the discriminator by trying to generate data closer and closer to the real one. Such frameworks have proven to be extremely effective.

The architecture of the discriminator D we use is as follows:

- *conv1*: convolution layer with 16 filters each of size (5,5) with stride 1 followed by Leaky ReLU and batch normalization.
- *conv2*: convolution layer with 16 filters each of size (2,2) with stride 2.
- *conv3*: convolution layer with 32 filters each of size (4,4) with stride 1 followed by Leaky ReLU and batch normalization.
- *conv4*: convolution layer with 32 filters each of size (2,2) with stride 2.
- *conv5*: convolution layer with 64 filters each of size (3,3) with stride 1 followed by Leaky ReLU and batch normalization.
- *conv6*: convolution layer with 64 filters each of size (2,2) with stride 2 followed by Leaky ReLU and batch normalization.
- *fc1*: A 128-neuron fully connected dense layer followed by a Leaky ReLU activation.
- *fc2* : A single neuron followed by a sigmoid activation.

After training the autoencoder using the GAN strategy on pristine images only, the encoder A_e is used to extract the feature vector $\mathbf{h}_k$ from each patch $\mathbf{P}_k$ under analysis.

4.3.3 One-Class SVM

The autoencoder A is trained only on pristine patches and hence it learns to encode them very well. So when A sees a patch containing a forgery, it encodes it quite differently. In order to capture this difference without any knowledge on forged data, we use a one-class SVM trained on feature vectors $\mathbf{h}$ extracted from pristine images only. The used one class SVM learns pristine feature distribution. It then outputs a soft value which represents the likelihood of the feature vector $\mathbf{h}$ under analysis being pristine. We define the soft mask as a matrix the same size of the image, where each entry contains the soft SVM output relative to the image patch in the same position. This soft mask can be used to obtain the final detection binary mask $\hat{\mathbf{M}}$ by simply thresholding.

4.4 Experimental Validation

In this section we report the experimental validation of the proposed technique. We first discuss how we built the used dataset. We then provide details about the considered experimental setup for reproducible research. Finally, we show the achieved numerical results.

4.4.1 Dataset

We tested our algorithm using overhead images obtained from the Landsat Science program [80,81]. The Landsat Science is a program run jointly by NASA [82] and the US Geological Survey(USGS) [83]. It was first launched in 1972 and has produced the longest, continuous record of Earth's land surface as seen from Space. NASA is responsible for the remote sensing equipment, launching satellites and validating their performance. USGS operates the satellites and manages data reception, archiving and distribution. Since late 2008 these images have been made available free of charge. The Landsat Program obtains overhead images from a series of satellites. We have

created our dataset $\mathcal{D}$ using images from one satellite. $\mathcal{D}$ consists of 130 color images each cropped at a resolution of 650×650 pixels. This dataset is further divided into three parts namely training $\mathcal{D}_{\text{train}}$, validation $\mathcal{D}_{\text{val}}$ and testing $\mathcal{D}_{\text{test}}$.

Out of the 130 images, 30 of them are used to create patches for training and validation. Patches of size $64 \times 64 \times 3$ are extracted from every image with a patch stride of 32×32 generating a total of 10830 patches. Out of these patches, 20 percent have been used for validation and the remaining for training. So $\mathcal{D}_{\text{train}}$ consists of 8664 patches and $\mathcal{D}_{\text{val}}$ 2166 patches.

The remaining 100 images are used for creating $\mathcal{D}_{\text{test}}$. Half of $\mathcal{D}_{\text{test}}$ is used to generate forgeries, the remaining half is kept as pristine testing data. In order to create forged images, credible objects such as airplanes, clouds, etc. are spliced at random positions onto the 50 selected images from $\mathcal{D}_{\text{test}}$. During the splicing operation, the size of spliced objects relative to the used analysis patch size is controlled. Therefore, we define three sizes namely:

- Small - Object size is smaller than the patch size (approximately 32 pixel per side).
- Medium - Object size is comparable to patch size (approximately 64 pixel per side).
- Large - Object size is larger than patch size (approximately 128 pixel per side).

Objects of each size are forged onto the 50 images at random positions to create 150 forged images, i.e., we have 50 images with Small objects spliced onto them ($\mathcal{D}_{32}$), 50 images with Medium objects spliced onto them ($\mathcal{D}_{64}$), and at last 50 images with Large objects spliced onto them ($\mathcal{D}_{128}$). Examples of pristine images and forged images with different size forgeries are shown in Figure 4.4.

4.4.2 Experimental Setup

Our model consists of two important components, namely the Autoencoder A and the one-class SVM. In this section we describe the policies we used to choose the best model and the various hyper parameters for each of these.

Autoencoder

The autoencoder is fed with patches from pristine images and its task is to capture their distribution. An autoencoder can be trained on its own but we choose to couple it with a Discriminator D to further push its training. The job of D is to be able to discriminate between patches produced by A and the actual patches from pristine images. By coupling A with D we form a Generative Adversarial Network (GAN). We can judge the performance of A using the mean squared error metric and the performance of D by binary cross entropy. Both A and D are CNNs and, in order to choose the right CNN architecture, we use the following approach:

- The architecture of D is fixed and it is described in the *Feature Extraction* section.
- For A, a variety of CNN architectures are tested and the one with the lowest mean squared error loss is chosen as the best model.

The various architectures tested to choose the best model for A are detailed in Table 4.1 and Table 4.2

Training Strategy

We adopt two different training strategies, *With GAN* and *Without GAN*

- **Without GAN**: we train the autoencoder with patches from $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{val}}$. We use the Adam optimizer for a total of 100 epochs. The model weights with

the lowest MSE loss on $\mathcal{D}_{\text{val}}$ over the 100 epochs are chosen as the final model weights.

- **With GAN**: we first train the autoencoder using patches from $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{val}}$ using the Adam optimizer for 100 epochs. The weights corresponding to the lowest loss on $\mathcal{D}_{\text{val}}$ over the 100 epochs are locked. We then use these weights to initialize the weights of the generator in the GAN. The GAN is then trained for 100 epochs with the SGD optimizer and a learning rate of 0.001 for the discriminator, and the Adam optimizer with a learning rate of 0.001 for the generator. The GAN training is carried out on batches of 128 patches. The weights with the lowest MSE loss are chosen for the final generator.

For the SVM, we used a radial basis function with $\gamma = 1/2048$ as kernel, and small value of nu-parameter ($\nu = 0.00001$).

4.4.3 Results

Using the **With GAN** training strategy, the mean squared error (MSE) loss over $\mathcal{D}_{\text{val}}$ for the various architectures of the autoencoder are reported in Table 4.3. Architectures A_2, A_3 and A_4 have similar number of parameters (about 100k) while A_1 has almost a million parameters. Note that A_1 , A_2 and A_3 perform similarly despite the huge difference in the number of parameters. Among all of them A_4 provides the lowest MSE and hence better patch reconstruction. Therefore, we decided to select architecture A_4 for our system.

In order to visualize the forged-vs-pristine discriminability power of feature vectors $\mathbf{h}$ extracted from the proposed autoencoder, we applied the t-SNE algorithm [84]. T-SNE can be used for unsupervised feature dimensionality reduction in order to check whether it is possible to cluster some data. In our particular case, the feature vector $\mathbf{h}$ is a high dimensional vector that is very difficult to visualize, whereas, by applying t-SNE we have some visual clues on feature behavior. Figure 4.5 shows the distribution of features in a reduced dimensionality space of three dimensions using t-SNE. It is

possible to notice that patches not containing forged pixels have features that cluster together (i.e., red dots). Conversely, features belonging to patches containing forged pixels (i.e., blue dots) are spread in the three-dimensional space far from the pristine cluster. This confirms that the proposed feature vector is able to capture forged-vs-pristine information.

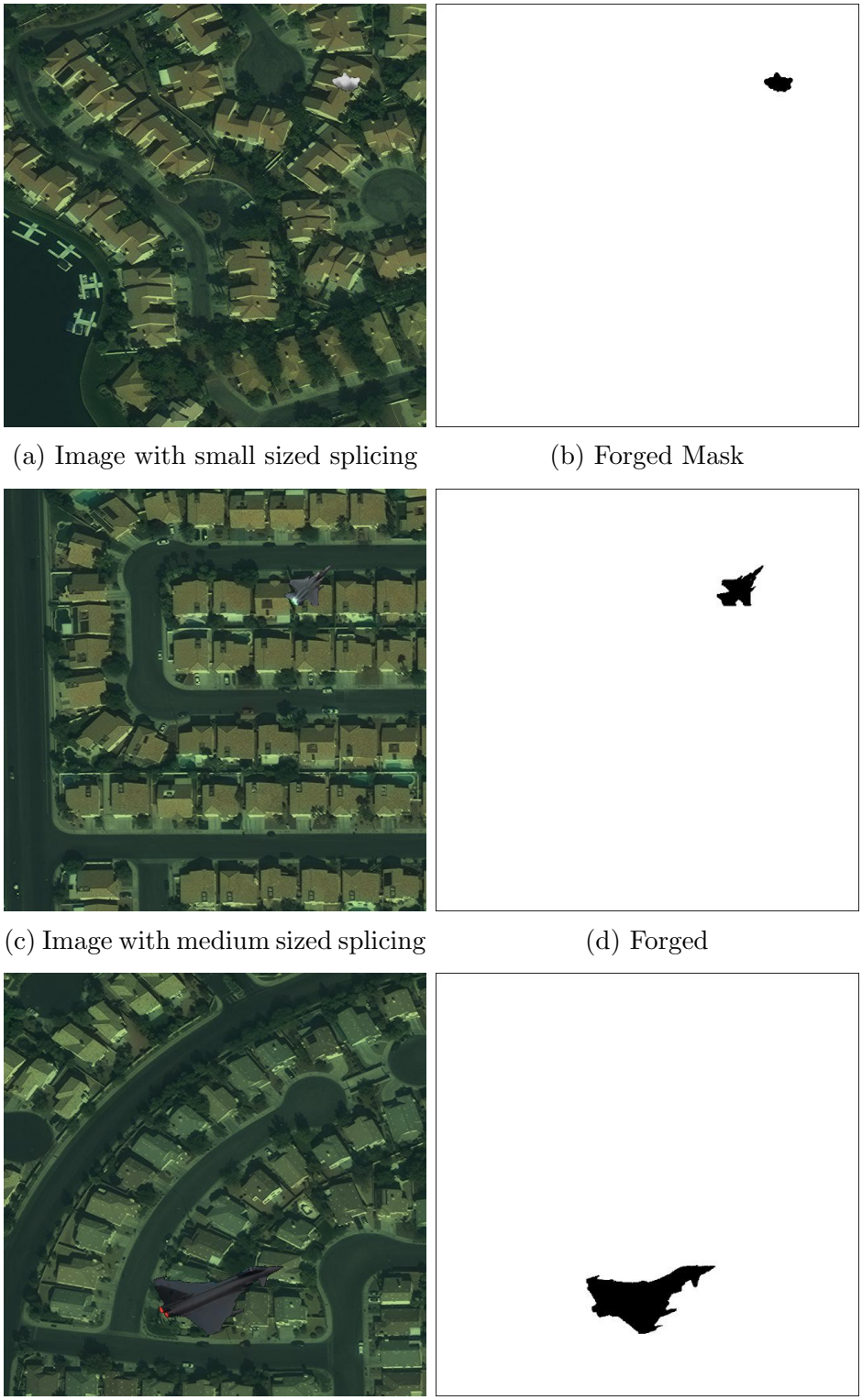
In order to evaluate forgery detection performance, we estimated the soft mask $\tilde{\mathbf{M}}$ for each image in the dataset. For each mask, we selected as pristine confidence the minimum $\tilde{\mathbf{M}}$ value (i.e., the SVM output associated to the least probable pristine patch). By thresholding this confidence score, we obtained a receiver operating characteristic (ROC) curve. Figure 4.6 shows some examples of forged images, groundtruth masks, and estimated soft mask $\tilde{\mathbf{M}}$. Figure 4.7 shows ROC curves split for datasets containing forgeries of different average size. Clearly, the bigger the forgery (i.e., 128 pixel per side), the better the performance (area under the curve around 0.97). However, even when forgeries are smaller than the analysis block (i.e., 32 pixel per side on 64×64 blocks), the area under the curve (AUC) is almost 0.80.

Additional results are reported in Table 4.4. This table reports AUC for each different size dataset depending on the used autoencoder training strategy. More precisely, it is possible to notice that by training the autoencoder without the GAN, detection results are always slightly worse. This motivates the use of the GAN training paradigm for forgery detection in this scenario.

In order to validate the proposed method in terms of localization, we computed a soft mask $\tilde{\mathbf{M}}$ for each image in the dataset. We then thresholded each soft mask $\tilde{\mathbf{M}}$ to obtain a binary mask soft mask $\hat{\mathbf{M}}$. For each image and used threshold, we computed: the true positive rate as the percentage of forged pixels correctly detected; false positive rate as the percentage of pristine pixels detected as forged. Based on these two values, we drew ROC curves. Figure 4.8 shows ROC curves obtained with our proposed GAN on datasets with forgeries of different size. Specifically, it is possible to notice that AUC is always greater than 0.90. In particular, if the forgeries are twice the size of the analysis patch, the AUC is higher than 0.97.

Additional results are reported in Table 4.5. We show AUC values for the different datasets (according to forgery sizes), comparing the effect of training the autoencoder with or without the GAN. Notice that, for localization purposes, it is slightly better to avoid the GAN.

A final consideration is devoted to computational time. We tested the proposed algorithm on a workstation equipped with an Intel Core i7-5930K CPU, 128 GB of RAM and a NVIDIA GeForce Titan X GPU. The processing time needed for a 64×64 pixel patch (considering both the autoencoder and the SVM) was around 500μs for testing. As each patch processing is independent, the algorithm allows for strong parallelization, thus making processing of high resolution images not an issue.



(a) Image with small sized splicing (b) Forged Mask

(c) Image with medium sized splicing (d) Forged

(e) Image with large sized splicing (f) Forged

Fig. 4.4.: Examples of forged images with forgeries of different sizes. Ground truth masks **M** are also reported.

Table 4.1.: The four proposed encoder architectures (shown in columns). The number of filters for each convolutional layer, its size and the used stride is shown in parenthesis followed by the activation function. If no activation is specified for any layer its assumed to be linear activation

Encoder Architectures			
A_{1e}	A_{2e}	A_{3e}	A_{4e}
conv1 (16, (6,6), Stride 1) + ReLU	conv1 (16, (6,6), Stride 1)	conv1 (16, (6,6), Stride 1)	conv1 (16, (6,6), Stride 1)
conv2 (32, (5,5), Stride 2) + ReLU	conv2 (16, (5,5), Stride 2)	conv2 (16, (5,5), Stride 2)	conv2 (16, (5,5), Stride 2)
conv3 (64, (4,4), Stride 2) + ReLU	conv3 (32, (4,4), Stride 2)	conv3 (32, (4,4), Stride 2)	conv3 (32, (4,4), Stride 2)
conv4 (128, (3,3), Stride 2) + ReLU	conv4 (32, (3,3), Stride 2)	conv4 (32, (3,3), Stride 2)	conv4 (64, (3,3), Stride 2)
conv5 (256, (2,2), Stride 2) + ReLU	conv5 (128, (2,2), Stride 2)	conv5 (128, (2,2), Stride 2)	conv5 (128, (2,2), Stride 2)
BN after each convolutional layer except conv5			

Table 4.2.: The four proposed decoder architectures (shown in columns). The number of filters for each convolutional layer, its size and the used stride is shown in parenthesis followed by the activation function. If no activation is specified for any layer its assumed to be linear activation

Decoder Architectures			
A_{1d}	A_{2d}	A_{3d}	A_{4d}
deconv1 (256, (2,2), Stride 2) + ReLU	deconv1 (32, (2,2), Stride 2)	deconv1 (64, (2,2), Stride 2)	deconv1 (64, (2,2), Stride 2)
deconv2 (128, (3,3), Stride 2) + ReLU	deconv2 (32, (3,3), Stride 2)	deconv2 (32, (3,3), Stride 2)	deconv2 (32, (3,3), Stride 2)
deconv3 (64, (4,4), Stride 2) + ReLU	deconv3 (16, (4,4), Stride 2)	deconv3 (32, (4,4), Stride 2)	deconv3 (16, (4,4), Stride 2)
deconv4 (32, (5,5), Stride 2) + ReLU	deconv4 (16, (5,5), Stride 2)	deconv4 (16, (5,5), Stride 2)	deconv4 (16, (5,5), Stride 2)
deconv5 (3, (6,6), Stride 1) + tanh	deconv5 (3, (6,6), Stride 1) + tanh	deconv5 (3, (6,6), Stride 1) + tanh	deconv5 (3, (6,6), Stride 1) + tanh
BN after each deconvolutional layer except deconv5			

Table 4.3.: MSE of the various autoencoder architectures. The one with the lowest MSE loss is selected for the proposed method.

Architecture	**Trainable Parameters**	**MSE loss**
A_1	997299	0.00131671
A_2	84547	0.00131675
A_3	124883	0.00130047
A_4	135939	0.00125511

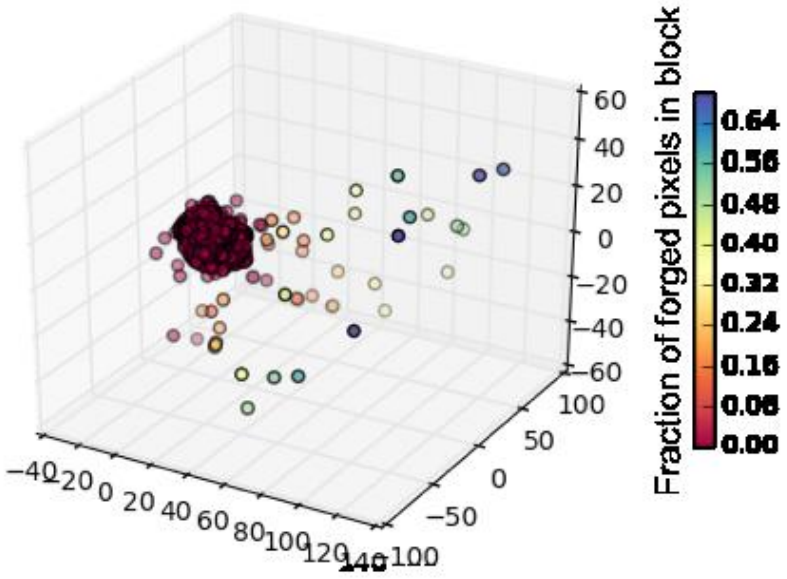


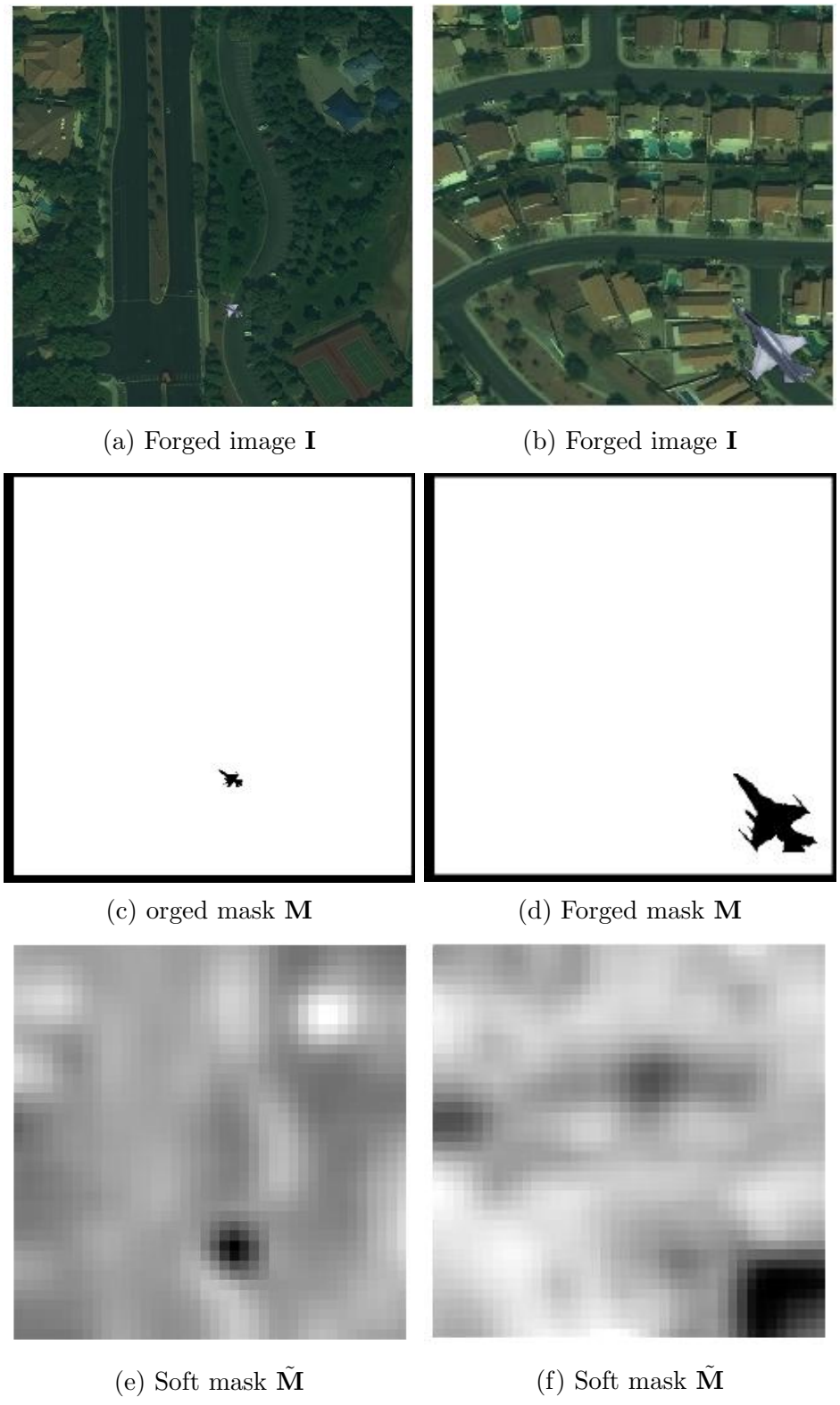
Fig. 4.5.: Example of t-SNE representation of the feature vectors extracted from a forged image. Features from pristine patches (i.e., red dots) cluster together, whereas features from forged patches (i.e., blue dots) are more distant.

Table 4.4.: Detection results in terms of AUC for the different datasets. AUCs are reported in two different cases: autoencoder trained with or without the GAN. Best results are reported in italics.

Forgery Size	**AUC (without Gan)**	**AUC (with GAN)**	**AUC Difference**
Small	0.784	*0.797*	+0.013
Medium	0.904	*0.920*	+0.016
Large	0.950	*0.972*	+0.022

Table 4.5.: Localization results in terms of AUC for the different datasets. AUCs are reported in two different cases: autoencoder trained with or without the GAN. Best results are reported in italics.

Forgery Size	**AUC (without Gan)**	**AUC (with GAN)**	**AUC Difference**
Small	*0.913*	0.902	-0.009
Medium	*0.963*	0.961	-0.002
Large	*0.970*	0.974	-0.004



(a) Forged image **I**

(b) Forged image **I**

(c) orged mask **M**

(d) Forged mask **M**

(e) Soft mask $\tilde{\mathbf{M}}$

(f) Soft mask $\tilde{\mathbf{M}}$

Fig. 4.6.: Examples of forged images with ground truth forged mask **M** and estimated soft mask $\tilde{\mathbf{M}}$. It is possible to notice the correlation between ground truth and estimated soft mask.

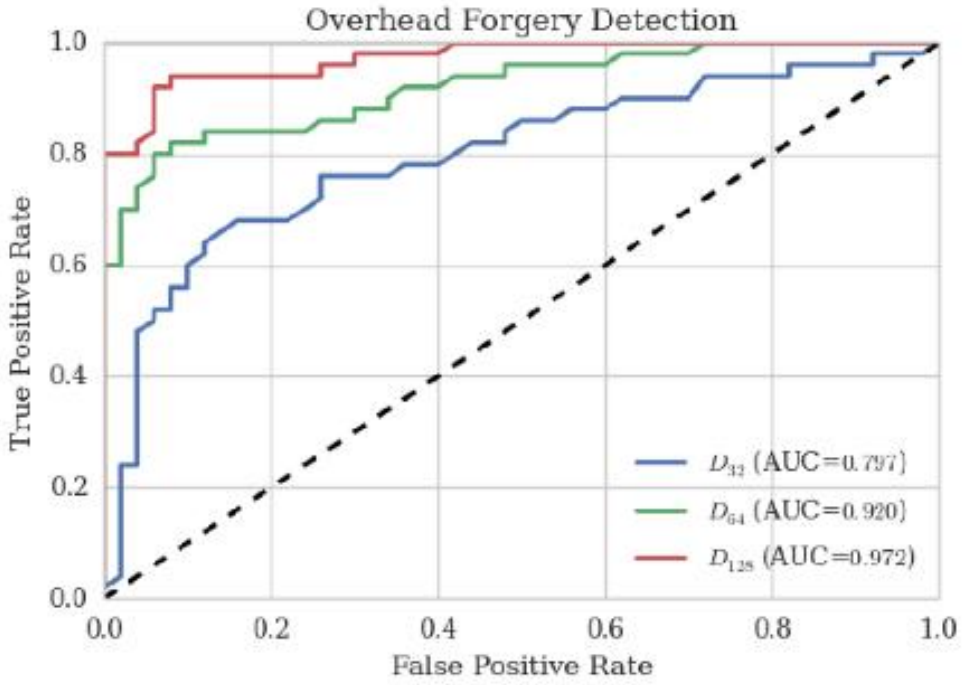


Fig. 4.7.: Forgery detection ROC curves. Each curve represents results on a different dataset according to the forgery average size.

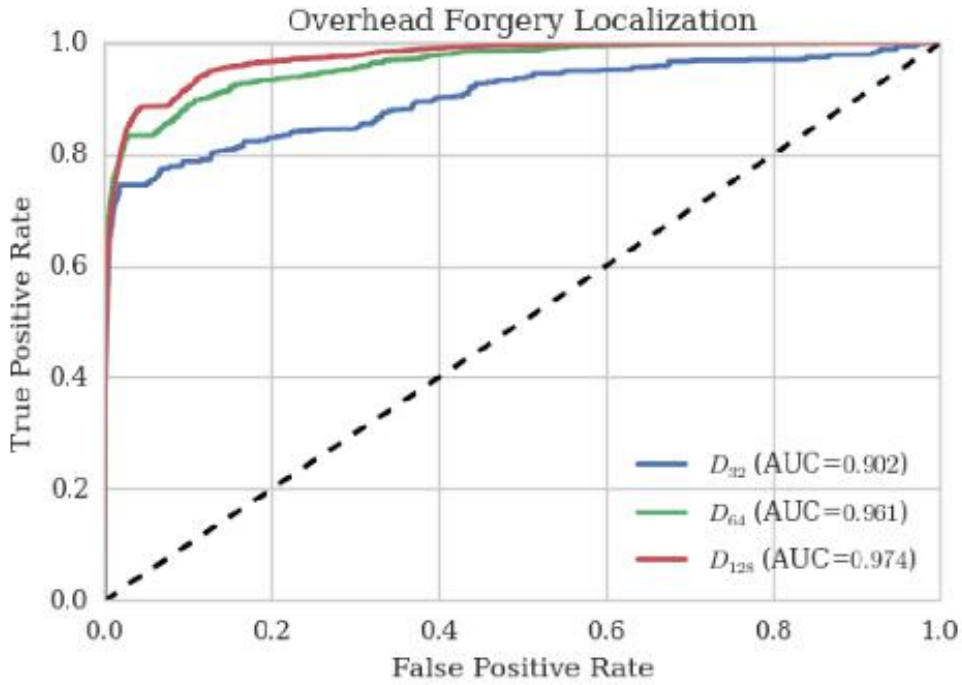


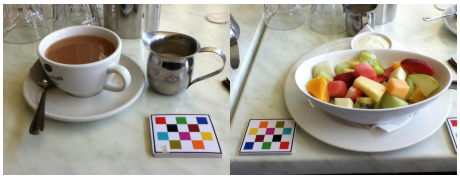
Fig. 4.8.: Forgery localization ROC curves. Each curve represents results on a different dataset according to the forgery average size.

5. LEARNING EATING ENVIRONMENTS THROUGH SCENE CLUSTERING

5.1 Overview

In 2016, $7.5 trillion was spent on healthcare worldwide, which is approximately 10% of the world GDP [85]. While there are many factors that influence health, dietary habits have a significant impact [86, 87]. To understand the complex relationship between dietary habits and health, nutrition practitioners and researchers often conduct dietary studies to subjectively assess dietary intake of children and adults. Participants in these studies are asked to report the foods and drinks they consumed on a daily basis for a period of time. This data is then analyzed by researchers to understand the impact of certain dietary behaviors on health. For example, studies have shown that frequent consumption of fast food [88], skipping breakfast [89], and absence of home food [90] contribute to the increasing risk of obesity and overweight.

While many studies have been conducted to understand how diet affects health [91], fewer work has been done to study the relationship between eating environments and health. However, researchers [92, 93] and organizations such as the World Heath Organization and International Obesity Task Force have recognized the vital role of eating environments on health and diet. Studies have shown that some factors of the environment such as screen viewing during meals, family mealtime [94], and meal preparation time [95] influence health. For instance, family mealtime is shown to positively affect nutrient intake and meal preparation time is inversely related to Body Mass Index (BMI). While such patterns have been found, the relationship between eating environments and health is still poorly understood, partly due to the lack of valid, reliable measures of environmental factors. In this chapter, we focus on understanding eating environments of an individual using digital images captured during



(a) Scene 1

(b) Scene 2

Fig. 5.1.: Here are images of two different eating environments, captured by a single participant. The colored checkerboard in all the images is the FM.

eating occasions. In particular, we are interested in learning how many different environments an individual consumes food in.

Dietary Recall, 24-hr recall, and Food Frequency Questionnaire (FFQ) [96] are well-known dietary assessment methods used in most dietary studies. These methods require participants to manually enter details of their diet information through a web interface, or an in-person or phone interview. These methods are known to be time-consuming and prone to errors because participants may not recall all foods and beverages they consumed or cannot accurately estimate the food portion [96]. To overcome these limitations, researchers have leveraged advances in mobile technology to develop image-based dietary assessment methods to record daily food intake. Some examples of image based dietary assessment tools are TADA™ [97] and FoodLog [98].In these approaches, participants record their diet by capturing images of foods and beverages consumed using mobile cameras. These images can then be analyzed by trained analysts [99] or using image analysis methods [100–102] to extract nutrient information. In this chapter, we leverage eating occasion images captured using the TADA™ system to cluster eating scenes based on their visual appearance to

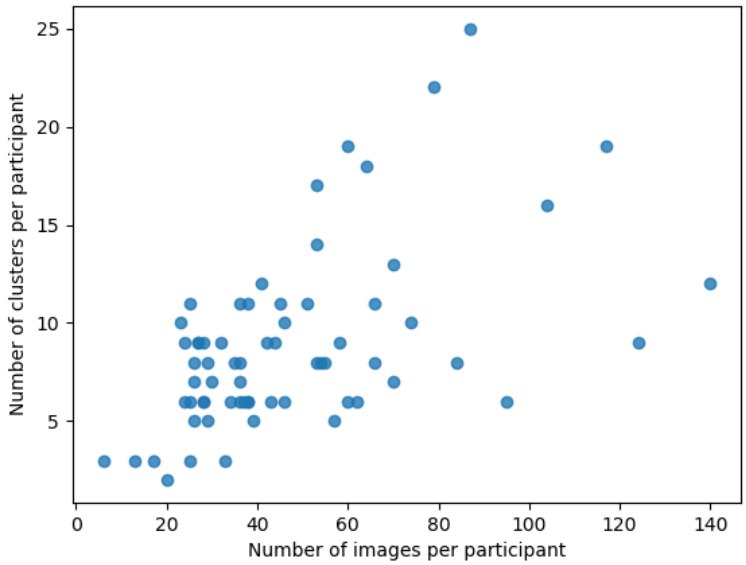


Fig. 5.2.: Scatter plot of the number of images captured per participant versus the number of eating scene clusters per participant.

help understand the relationship between a person's eating environment and dietary quality.

5.2 Dataset and Related Work

We used a dataset $\mathcal{D}$ that consists of 3137 images from 66 participants collected in a community dwelling dietary study [103]. In this study, participants were asked to take pictures of their foods and to place a colored checkerboard with known dimensions, called the Fiducial Marker (FM), in the scene. An example of two pairs of images belonging to two different eating environments are shown in Fig. 5.1. The FM serves two purposes: to aid in color correction, and more importantly, to provide a reference scale for food portion estimation, so the nutrient content of the foods can be computed. One of the challenges of this dataset is the large variance in the number of images and the number of eating environments for different participant, as shown in Fig. 5.2.

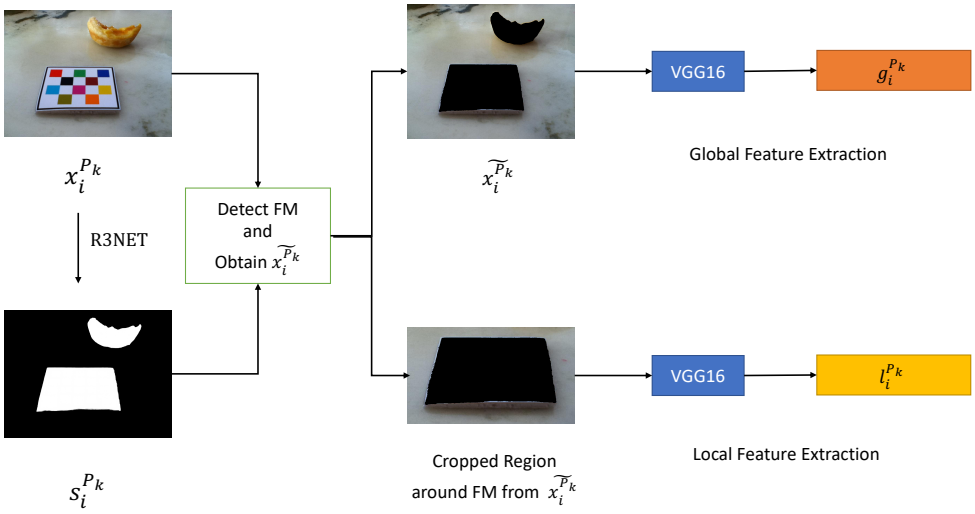


Fig. 5.3.: Overview of feature extraction

Our goal is to cluster food images of a participant based on eating environments. Clustering is not a new concept. Several classical [104] and deep learning-based [105] methods have been proposed. Classical clustering approaches are applied on relevant features extracted from the data [106]. However, the problem of clustering food images based on the eating environments captured has not been studied before, so the relevant features are not defined. On the other hand, deep learning-based methods, given sufficient data, are capable of simultaneously learning to extract relevant features and cluster images [105]. A common assumption of such approaches is that the number of clusters is known *a priori* [105]. However, in our case, we do not know *a priori* the number of eating environments for each participant. In addition, the number of images collected by each participant is not sufficient to apply deep learning-based methods directly, as they usually require hundreds of images per cluster for training a good model. As both classical and deep learning-based approaches have their shortcomings for clustering eating environments from images, new techniques need to be developed to solve this problem.

5.3 Method

In this section, we describe the details of the proposed method. First, we introduce the notation used throughout the chapter. P_k denotes the k^{th} participant in the dataset. The i^{th} image captured by participant P_k is denoted by $x_i^{P_k}$. Our goal is to cluster food images based on eating environment. We do this by extracting features at the local and global level from relevant pixels of $x_i^{P_k}$ and then applying a clustering method on said features.

5.3.1 Global Feature Extraction

The image $x_i^{P_k}$ contains many salient objects such as the food, drinks, FM, and silverware. However, pixels belonging to the salient objects do not contain any credible information regarding the eating environment because a person can eat different food

with different plates in the same eating environment. Pixels belonging to the FM are also not relevant because the FM is common to all food images, irrespective of the eating environment. Instead, we are interested in pixels belonging to the non-salient regions. To extract these relevant pixels in $x_i^{P_k}$, we first extract its binary saliency map denoted by $s_i^{P_k}$ using a state-of-the-art saliency estimator R3NET [107]. The salient pixels of $x_i^{P_k}$ have a value of 1 in $s_i^{P_k}$ and the rest have a value of 0. The relevant pixels of $x_i^{P_k}$ are captured by $\tilde{x}_i^{P_k}$, and is defined as

$$\tilde{x}_i^{P_k} = (1 - s_i^{P_k}) * x_i^{P_k} \tag{5.1}$$

Features are extracted from $\tilde{x}_i^{P_k}$ using a Convolutional Neural Network (CNN) VGG16 [108] pre-trained on the ImageNet dataset [13]. We use a pre-trained VGG16 for feature extraction because these features are robust to artifacts such as noise, lighting changes, and differing viewpoints.

The 2^{nd} convolutional layer is chosen for feature extraction and the reasoning behind this is explained in Section 5.4.1. Global Average Pooling (GAP) is applied to the output of the 2^{nd} layer to spatially summarize the information into a 64-dimensional vector. This vector is denoted as $g_i^{P_k}$ and we refer to it as the global feature.

5.3.2 Local Feature Extraction

Since the FM is always placed in close proximity to the foods, we assume it is on the same surface as the foods. Examples of such surfaces may include desk, dining table, and kitchen counter, to name a few. These surfaces give us a lot of information about the eating environment because it is very likely that a person uses the same surface during an eating event in a particular eating environment. Therefore, we extract features from this surface and they are denoted by $l_i^{P_k}$.

By identifying the FM in the image, we can extract information about the eating surface. The FM is detected by finding the salient object with the highest number of

interest points. An interest point is a pixel that ORB (Oriented FAST and Rotated BRIEF) [109] finds useful for image registration. The FM has a lot of interest points because of the colored checkerboard pattern and it is unlikely for other salient objects such as foods to have as many distinct interest points. The salient objects of the image can be found by performing connected component analysis on $s_i^{P_k}$ and treating each connected component as a salient object. Once we have located the FM, we extract a region around it from $\tilde{x}_i^{P_k}$. $l_i^{P_k}$ is obtained from this region using the pre-trained VGG16 in the same way as done in 5.3.1. Fig. 5.3 illustrates the the global and local feature extraction process.

5.3.3 Feature Fusion and Clustering

We fuse the local and global features using their distance matrices. The distance matrices G^{P_k} and L^{P_k} are defined as follows

$$\begin{aligned} L_{(i,j)}^{P_k} &= ||l_i^{P_k} - l_j^{P_k}||_2 \\ G_{(i,j)}^{P_k} &= ||g_i^{P_k} - g_j^{P_k}||_2 \end{aligned} \tag{5.2}$$

The fused distance matrix D^{P_k} is defined as follows

$$D^{P_k} = \alpha * L^{P_k} + (1 - \alpha) * G^{P_k} \tag{5.3}$$

Here $\alpha \in [0, 1]$ and controls the relative importance of the local and global features. A higher value of α indicates local features are more important and vice versa. In our case , $\alpha = 0.44$. The reason behind this is explained in Section 5.4.1. Affinity Propagation (AP) [110] is applied to D^{P_k} to obtain the final clusters.

5.4 Experiments

In this section we describe all the experiments conducted and compare our method to four existing clustering methods, namely: DBSCAN [111], MeanShift [112], OP-

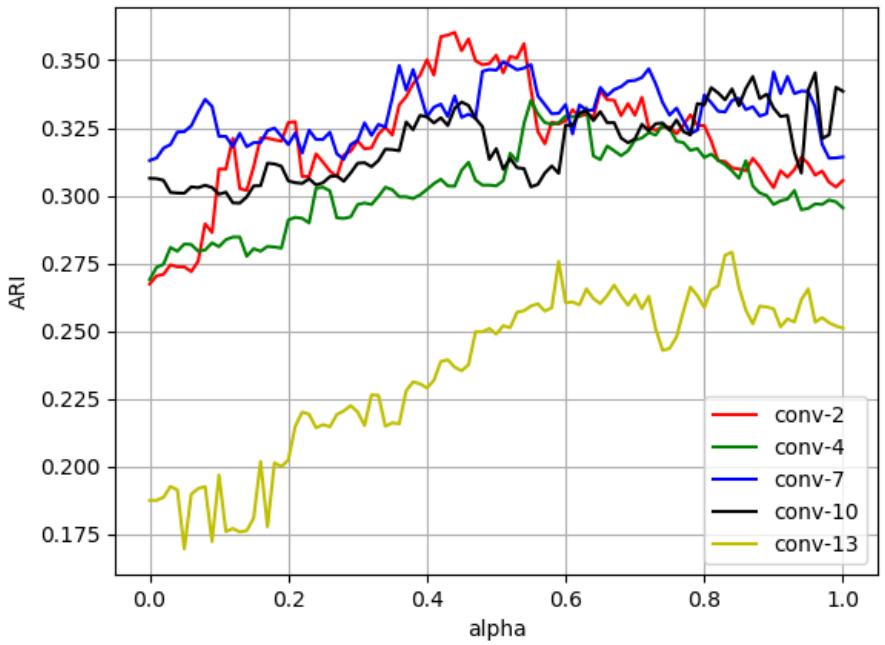


Fig. 5.4.: ARI scores for different α and convolutional layer m on $\mathcal{D}_{\text{val}}$

TICS [113], and AP [110]. We use the Adjusted Rand Index (ARI) and Normalized Mutual Information (NMI) to measure the accuracy of our clusters. ARI ranges from -1.0 to 1.0 while NMI ranges from 0 to 1.0. Higher values indicate better clustering for both ARI and NMI. Accuracy over a dataset is reported as the average accuracy among all participants.

5.4.1 Hyperparameter Tuning

Our method has two hyperparameters: the weighting factor for feature fusion α and the convolutional layer of VGG16 for feature extraction m. Our dataset $\mathcal{D}$ is split into $\mathcal{D}_{\text{val}}$ and $\mathcal{D}_{\text{test}}$. $\mathcal{D}_{\text{val}}$ consists of images from 10 participants and $\mathcal{D}_{\text{test}}$ contain images from 56 participants. Our method is evaluated on $\mathcal{D}_{\text{val}}$ by varying α and m. α ranges from 0 to 1 in steps of 0.01. To find the optimal m for our dataset, we extract features from convolutional layers preceding a max-pooling layer. There are five such layers in VGG16. ARI is used to select the optimal values. Fig. 5.4 shows the ARI scores for different values of α and convolutional layer m, and the optimal value for α is 0.44 and for m is 2.

From equation 5.3, we can infer that as the value of α increases, more weight is given to local features and vice-versa. An optimal value of 0.44 for α suggests that our method performs best when approximately equal weight is given to both local and global features. This shows that using only one of these features is less optimal. In Fig. 5.4, we can see that the performance of our method degrades once the chosen feature extraction layer gets very deep. Later layers, like conv-13, extract abstract features and completely lose information about edges, colors, and textures. We suspect this loss of information is the reason for decrease in performance.

As discussed above, we chose VGG16 as the CNN architecture and varied α and m over $\mathcal{D}_{\text{val}}$ to find the optimal solution. In principle, we could also optimize for the right architecture by implementing our method on other CNN architectures such as ResNet [114], Inception [115] *etc.* in place of VGG16 for feature extraction. Opti-

mizing across the architecture space would significantly increase the complexity of the experiments. We chose VGG16 over ResNet and Inception [115] because of the simplicity of its structure for easier analysis. VGG16 has only 13 convolutional layers and they are all stacked one after another in a linear fashion. In contrast, Resnet can have anywhere between 34 to 152 convolutional layers and for Inception, convolutional layers are stacked in a non-linear fashion. Comparison between different network architectures could be a future direction to explore when implementing for a specific application with target requirements such as model complexity and computation resource.

5.4.2 Testing

We evaluated the performance of our method on $\mathcal{D}_{\text{test}}$ after selecting the optimal values for α and m using the validation set $\mathcal{D}_{\text{val}}$. We choose four well-known clustering methods for comparison, namely DBSCAN [111], MeanShift [112], OPTICS [113], and AP [110] using the eating scene image as the input. ARI and NMI for the five methods are reported in Table 5.1, where our method achieves the best performance. It is worth noting that although we use AP [110] for clustering after feature fusion, our method performs significantly better than AP [110] when meaningful features are not known. This indicates that our feature extraction strategy is highly relevant and very important to our problem.

Table 5.1.: ARI and NMI scores for methods tested on $\mathcal{D}_{\text{test}}$. The best results are reported in bold.

Methods	ARI	NMI
Ours	**0.39**	**0.68**
DBSCAN [111]	0.24	0.47
MeanShift [112]	0.12	0.35
OPTICS [113]	0.08	0.39
AP [110]	0.2	0.49

6. SALIENCY-AWARE CLASS-AGNOSTIC FOOD IMAGE SEGMENTATION

6.1 Introduction

It is well-known that dietary habits have profound impacts on the quality of one's health and well-being [86, 87]. While a nutritionally sound diet is essential to good health [116], it has been established through various studies that poor dietary habits can lead to many diseases and health complications. For example, studies from the World Health Organization (WHO) [116] have shown that poor diet is a key modifiable risk factor for the development of various non-communicable diseases such as heart disease, diabetes and cancers, which are the leading causes of death globally [116]. In addition, studies have shown that poor dietary habits such as frequent consumption of fast food [88], diets containing large portion size of energy-dense foods [117], absence of home food [90] and skipping breakfast [89] all contribute to the increasing risk of overweight and obesity. Because of the many popular diseases affecting humans are related to dietary habits, there is a need to study the relationship between our dietary habits and their effect on our health.

Understanding the complex relationship between dietary habits and human health is extremely important as it can help us mount intervention programs to prevent these diet related diseases [118]. To better understand the relationship between our dietary habits and human health, nutrition practitioners and researchers often conduct dietary studies in which participants are asked to subjectively assess their dietary intake. In these studies, participants are asked to report foods and drinks they consumed on a daily basis over a period of time. Traditionally, self-reporting methods such as 24-hr recall, dietary records and food frequency questionnaire (FFQ) are popular for conducting dietary assessment studies [119]. However, these methods have several

drawbacks. For example, both the 24-hr recall and FFQ rely on the participants' ability to recall foods they have consumed in the past. In addition, they are also very time-consuming. For dietary records, participants are asked to record details of the meals they consumed. Although this approach is less reliant on the participants' memory, it requires motivated and trained participants to accurately report their diet [119]. Another issue that affects the accuracy of these methods is that of under-reporting due to incorrect estimation of food portion sizes. Under-reporting has also been associated with factors such as obesity, gender, social desirability, restrained eating and hunger, education, literacy, perceived health status, age, and race/ethnicity [97]. Therefore, there is an urgent need to develop new dietary assessment methods that can overcome these limitations.

In the past decade, experts from the nutrition and engineering field have combined forces to develop new dietary assessment methods by leveraging technologies such as the Internet and mobile phones. Among the various new approaches, some of them use images captured at the eating scene to extract dietary information. These are called image-based dietary assessment methods. Examples of such methods include TADA™ [97], FoodLog [98] , FoodCam [120], Snap-n-Eat [121], GoCARB [122, 123] and DietCam [124], to name a few. In these methods, participants are asked to capture images of foods and drinks consumed via a mobile phone. These images are then analyzed to estimate the nutrient content. Estimating the nutrient content of foods in an image is commonly performed by trained dietitians, which can be time consuming, costly and laborious. More recently, automated methods have been developed to extract nutrient information of the foods from images [125–127]. The process of extracting nutrient information from images generally involves three subtasks, food segmentation, food classification and portion size estimation [97]. Food image segmentation is the task of grouping pixels in an image representing foods. Food classification can then identify the food types. Portion size estimation [126] is the task of estimating the volume/energy of the foods in the image. Each of these tasks is essential for building an automated system to accurately extract nutrient information

from food in images. In this chapter, we focus on the task of food segmentation. In particular, we propose a food segmentation method that does not require information of the food types.

Food segmentation plays a crucial role in estimating nutrient information as the image segmentation masks are often used to estimate food portion sizes [124–126, 128, 129]. Food segmentation from a single image is a challenging problem as there is a large inter- and intra-class variance among different food types. Because of this variation, techniques developed for segmenting a particular class of foods will not be effective on other food classes. Despite these drawbacks, several learning based food segmentation methods [100, 123, 130, 131] have been proposed in recent years. One of the constraints of learning based methods is data dependency. They are only effective on the food categories they trained on. For instance in [130], class activation maps are used to segment food images. The Food-101 dataset [132] is used to train the model and the method is tested on a subset of another dataset that have common food categories with Food-101. This is a clear indication that their method [130] is only effective on food classes that have been trained on. Similarly, the learning based method proposed in [131] is trained and tested only on UEC-FOOD100 [133]. The UEC-FOOD100 dataset has a total of 12,740 images with 100 different food categories, out of which 1,174 have multiple foods in a single image. In their method, the dataset is partitioned into training and testing subsets, each contains all the food categories. The authors of [131] split this dataset into training and testing in the following way. All the images containing a single food category were used for training and images containing multiple food categories were used for testing. This way the training set contained 11,566 images and the testing set contains 1,174 images. Splitting the dataset in this fashion does not guarantee that the training and testing subsets contain images belonging to different food categories. In fact this would mean they contain common food categories. Furthermore, the authors in [131] did not conduct any cross dataset evaluation. Thus the learning based method in [131] is also only effective on food categories it has been trained on. In [123], a semi automatic method is proposed

to segment foods. The authors of [123] assume that foods are always present in a circular region. In addition they also assume information about the number of different food categories is known. They conducted experiments on a dataset of 821 images. While they achieve promising results, their approach is not designed for the real world scenario where their assumptions don't hold. In [134] a food segmentation technique is proposed that exploits saliency information. However, the success of this approach relies on successfully detecting the food container. In [134] the food container is assumed to be a circular plate. Experimental results were reported using a dataset consisting of only 60 images. While the assumptions in [134] are valid in some cases, its very likely they will not hold in often in the real world.

In addition, there are also constraints imposed by the available datasets. Publicly available food image datasets such as UECFOOD-100 [133], Food-101 [132] and UECFOOD-256 [135] are biased towards a particular cuisine and also do not provide pixel level labelling. Pixel level labelling is crucial because it forms the necessary ground truth for training and evaluating learning based food segmentation methods. To overcome the limitations posed by learning based methods and the availability of public datasets with ground truth information, we proposed to develop a food segmentation method that is class-agnostic. In particular, our class-agnostic food segmentation method uses information from two images, the before eating and after eating image to segment the foods consumed during the meal.

Our data is collected from a community dwelling dietary study [136] using the TADA™ platform. In this study, participants were asked to take two pictures of their eating scene, one before they start eating which we call the before eating image and one immediately after they finished eating which we call the after eating image. The before eating and after eating image represent the same eating scene, however for the purpose of this work, we only select image pairs where the after eating image does not contain any food. Our goal is to segment the foods in the before eating image using information from both before and after eating images. To illustrated this problem in a more general scenario, lets consider an experimental setup in which a person is

(a) Before eating image.

(b) After eating image.

Fig. 6.1.: A pair of eating scene images, taken before and after a meal is consumed. The salient missing object in figure a is the food in the container.

given a pair of images shown in Fig.6.1 and is asked the following question, "Can you spot the salient objects in Fig. 6.1a that are missing in Fig. 6.1b?". We refer to these as the *salient missing objects*. To find salient missing objects, the Human Vision System (HVS) compares regions that are salient in both images. In this example, the food, container and color checkerboard in Fig. 6.1a are the salient objects and in Fig. 6.1b, the color checkerboard, spoon and container are the salient objects. Comparing the salient objects in both of these images, HVS can identify the food as the salient missing object. In this chapter, our goal is to build a model to answer this question. By looking for salient missing objects in the before eating image using the after eating image as the reference we can then segment the foods without additional information such as the food classes. As the above approach does not require information about the food class, we are able to build a class-agnostic food segmentation method by segmenting only the salient missing objects.

The above question does not bear significance for just any pair of random images. It only becomes relevant when the image pairs are related. For example, in Fig. 6.1, both images have many regions/objects with same semantic labels such as color checkerboard, container and the black background. However, the relative positions of these regions/objects are different in both images due to camera pose and different time of capturing the images. Because of similarity at the level of semantics between both images, it is plausible to define the notion of salient missing objects. Notice that we are not interested in pixel-level differences due to changes in illumination, poses and angles.

In this experimental scenario, the visual attention of HVS is guided via a task, hence it falls under the category of top down saliency. Visual attention [137, 138] is defined as the process that capacitates a biological or artificial vision system to identify relevant regions in a scene [137]. Relevance of every region in a scene is attributed through two different mechanisms, namely top down saliency and bottom up saliency. In top down saliency, attention is directed by a task. An example of this mechanism in action is how a human driver's HVS identifies relevant regions on the road for a safe journey. Other examples where top down saliency have been studied are sandwich making [139] and interactive game playing [140]. In bottom up saliency, attention is directed towards those regions that are the most conspicuous. Bottom up saliency is also known as visual saliency. In the real world, visual attention of HVS is guided by a combination of top down saliency and bottom up saliency. In the above question of finding salient missing objects, visual attention is guided by a task and hence it falls under the category of top down saliency. Top down saliency has not been studied as extensively as visual saliency because of its complexity [137].

In this chapter, we propose an unsupervised method to find the salient missing objects between a pair of images for the purpose of designing a class agnostic food segmentation method. We use the after eating image as the background to find the contrast of every pixel in the before eating image. We then fuse the contrast map along with saliency maps to obtain the final segmentation mask of the salient

missing objects in the before eating image. We also compare our method to other class-agnostic methods. Since food is a salient object in the before eating image, by detecting salient objects in the before eating image we are able to segment the food. We compared our method to four state-of-the-art salient object detection methods, namely R3NET [107], NLDF [141], UCF [142] and Amulet [143].

The chapter is organized as follows. In section 2, we formulate our problem and discuss related work. We describe our proposed method in detail in Section 3. In section 4, we discuss dataset and experiment design. In section 5, we discuss experimental results and compare our method with other salient object detection methods. Conclusions are provided in Section 6.

6.2 Problem Formulation and Related Work

In this section, we first introduce common notations used throughout the chapter. We then discuss related works on modeling top down saliency and change detection.

6.2.1 Problem Formulation

Consider a pair of images $\{I^b, I^a\}$ captured from an eating scene.

- I^b : We refer to it as the "before eating image." This is the meal image captured before consumption.
- I^a : We refer to it as the "after eating image." This is the meal image captured immediately after consumption.

Our goal is to obtain a binary mask M^b, that labels the salient missing objects in I^b as foreground (with a binary label of 1) and rest of I^b as background (with a binary label of 0).

6.2.2 Related Work

Our goal is to find salient missing objects in a pair of images. Since the visual attention of the HVS is guided by a task, it falls under the category of top down saliency. Top down saliency is much more complex than visual saliency and hence has not been studied extensively. Some of the recent works modeling top down saliency paradigms are [144, 145]. In [145], given an image or video and an associated caption, the authors proposed a model to selectively highlight different regions based on words in the caption. Our work is related in the sense that we also try to highlight and segment objects/regions based on a description, except that the description in our case is a much more generic question of finding the salient missing objects in a pair of images without specific details.

Another related problem is modeling change detection [146–149]. In change detection, the objective is to detect all relevant changes between a pair of images that are aligned or can be potentially aligned via image registration. Examples of such changes may include object motion, missing objects, structural changes [147] and changes in vegetation [146]. One of the key differences between change detection and our proposed problem is that in change detection, the pair of images are aligned or can be potentially aligned via image registration [150] which is not true in the case of salient missing objects. In the case of finding salient missing objects, we cannot guarantee that I^b and I^a can be registered, as often there is relative motion between objects of interest as shown in Fig. 6.1 and also in Fig. 6.6.

The problem of finding salient missing objects can be thought of as a change detection problem in a more complex environment than those that have been previously considered. Hence, we need to develop new methods to solve this problem.

6.3 Method

In this section, we describe the details of our proposed method to segment salient missing objects in I^b. Our method consists of three parts, *segmentation and feature*

extraction, *contrast map generation* and *saliency fusion*. An overview of our proposed method is described in Fig. 6.2.

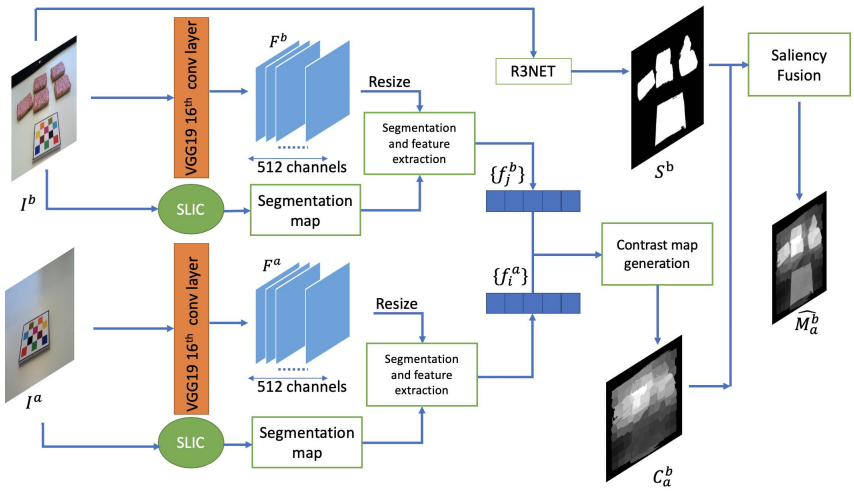


Fig. 6.2.: Overview of proposed method.

6.3.1 Segmentation And Feature Extraction

We first segment the pair of images I^a and I^b using SLIC [8] to group pixels into perceptually similar superpixels. Let $\mathcal{A} = \{a_i\}$ denote the superpixels of the after eating image I^a and $\mathcal{B} = \{b_j\}$ for superpixels of the before eating image I^b. Let $\mathcal{A}^e \subset \mathcal{A}$ denote the set of superpixels that are located along the image boundaries of I^a. Similarly $\mathcal{B}^e \subset \mathcal{B}$ denote the superpixels located along the image boundaries of I^b.

We extract features from each superpixel. We use these features to compute the contrast map. The contrast map C_a^b gives an estimate of the probability of pixels belonging to objects/regions present in I^b but missing in I^a. This will be explained in detail in section 6.3.2. To compute an accurate contrast map, pixels belonging to similar regions in I^b and I^a should have similar feature representation and vice versa. Going from I^b to I^a we can expect changes in scene lightning, changes in noise levels and changes in segmentation boundaries because of relative object motion. To

compute an accurate contrast map, its important that feature representation of pixels are robust to these artifacts.For this reason, we extract features using a pretrained Convolutional Neural Network (CNN) instead of using hand-crafted features. We use the VGG19 [108] pretrained on the ImageNet dataset [13]. ImageNet is a large dataset consisting of more than a million images belonging to 1000 different classes. It captures the distribution of natural images very well. Hence, features from a model pre-trained on Imagenet are able to better represent many objects, surface, textures *etc.* [147,151–153]. In contrast, features extracted from a model pre-trained on a food dataset are only able to represent food objects. As our goal is to extract features that are able to effectively represent both foods and other objects/surfaces, we chose to use a model pre-trained on ImageNet for generalizability.

We use the pretrained VGG19 for both I^b and I^a. The output of 16^{th} convolutional layer in VGG19 is extracted as the feature map. The reasoning behind this choice is explained in section 6.4.3. According to Table 1 in [108], VGG19 has a total of 16 convolutional layers. The dimensionality of the output of the 16^{th} convolutinal layer of VGG19 is $14 \times 14 \times 512$ where 14×14 is the spatial resolution. The input (I^b or I^a) to VGG19 has a spatial resolution of 224×224. We spatially upscale the output of the 16^{th} convolution layers by a factor of 16. We denote these upscaled feature maps of I^b and I^a as F^b and F^a, respectively. The dimensionality of F^b and F^a is then $224 \times 224 \times 512$. Thus every pixel will be represented by a 512 dimensional vector in the feature space. For each superpixel, we denote the extracted features as $\{f_j^b\}$ for the before eating image and $\{f_i^a\}$ for the after eating image. Using these extracted feature maps, f_i^a and f_j^b are computed as described in Eq. 6.1.

$$\begin{aligned} f_i^a &= \frac{1}{m_i^a} \sum_{k \in r_i^a} F^a(k) \\ f_j^b &= \frac{1}{m_j^b} \sum_{k \in r_j^b} F^b(k) \end{aligned} \tag{6.1}$$

where r_i^a denotes the set of pixels belongs to superpixel a_i and m_i^a is its cardinality. r_j^b and m_j^b are similarly defined.

6.3.2 Contrast Map Generation

Contrast is a term often associated with salient object detection methods. Contrast of a region in an image refers to its overall dissimilarity with other regions in the same image. It is generally assumed that regions with high contrast demand more visual attention [154]. In the context of our problem, visual attention is guided by trying to find objects in I^b that are missing in I^a. Therefore, our contrast map C_a^b of I^b is an estimate of the probability of each pixel belonging to an object missing in I^a. C_a^b is computed as shown in Eq. 6.2.

$$C_a^b = \frac{C_a^{b,\text{local}} + C_a^{b,\text{neigh}}}{max(C_a^{b,\text{local}} + C_a^{b,\text{neigh}})} \tag{6.2}$$

In $C_a^{b,\text{local}}$, contrast values of a superpixel b_j is computed using information from b_j and I^a, while in $C_a^{b,\text{neigh}}$ contrast value of b_j is computed using information from b_j, its neighboring superpixels and I^a. $max(C_a^{b,\text{local}} + C_a^{b,\text{neigh}})$, which is the maximum value in the contrast map, is used to normalize C_a^b to $[0, 1]$. To compute the contrast map $C_a^{b,\text{local}}$ or $C_a^{b,\text{neigh}}$, contrast values are computed for each superpixel and then these values are assigned to the associated individual pixels. However, if b_j is a superpixel along the image boundaries, that is $b_j \in \mathcal{B}^e$, we assign b_j a contrast value of zero. We assume that the salient missing objects are unlikely to be present along the image boundaries.

The contrast value of a superpixel $b_j \notin \mathcal{B}^e$ is denoted by $c_j^{b,\text{local}}$, and is computed as:

$$c_j^{b,\text{local}} = \min_{\forall i \text{ such that } a_i \in \mathcal{A}} ||f_j^b - f_i^a||_2 \tag{6.3}$$

If $b_j \in \mathcal{B}^e$ then $c_j^{b,\text{local}} = 0$. $c_j^{b,\text{local}}$ is the minimum Euclidean distance between the feature vector f_j^b and the closest feature vector of a superpixel in the after eating image. A superpixel b_j belonging to objects/regions that are common to both I^b and I^a will have lower value of $c_j^{b,\text{local}}$, while b_j belonging to objects/regions present in I^b but missing in I^a will likely have higher value of $c_j^{b,\text{local}}$.

Before describing how we compute $C_a^{b,\text{neigh}}$, we need to introduce a few more notations. For a given superpixel b_j, let $\mathcal{N}(b_j)$ denote the set of all neighboring superpixels of b_j. Similarly, for any superpixel a_i, $\mathcal{N}(a_i)$ is the set of neighboring superpixels. Consider a complete bipartite graph over the two sets of superpixels $\{a_i, \mathcal{N}(a_i)\}$ and $\{b_j, \mathcal{N}(b_j)\}$ denoted by

$$G_{a_i,b_j} = (\{a_i, \mathcal{N}(a_i)\} \cup \{b_j, \mathcal{N}(b_j)\},\ \mathcal{E}_{a_i,b_j}) \tag{6.4}$$

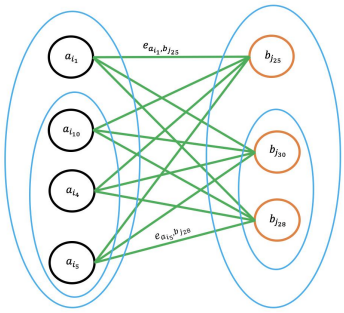
where $\mathcal{E}_{a_i,b_j}$ is the set of edges in G_{a_i,b_j}. An example is shown in Fig. 6.3.

In G_{a_i,b_j}, consider an edge $e_{a_{i_1},b_{j_1}}$ between the two superpixels $a_{i_1} \in \{a_i, \mathcal{N}(a_i)\}$ and $b_{j_1} \in \{b_j, \mathcal{N}(b_j)\}$, the edge weight is evaluated by the Euclidean norm $w(\cdot)$ defined as:

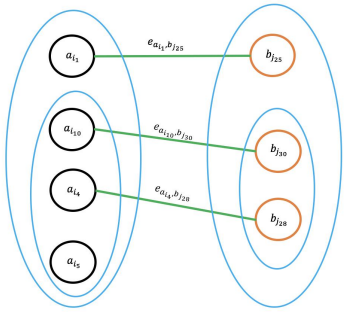
$$w(e_{a_{i_1},b_{j_1}}) = ||f_{a_{i_1}} - f_{b_{j_1}}||_2 \tag{6.5}$$

A matching over G_{a_i,b_j} is a set of edges $\mathcal{S} \subset \mathcal{E}_{a_i,b_j}$ such that no two edges in $\mathcal{S}$ share the same nodes. A maximum matching over G_{a_i,b_j}, denoted by $\mathcal{S}_k^{\mathcal{E}_{a_i,b_j}} \subset \mathcal{E}_{a_i,b_j}$, is a matching of maximum cardinality. There can be many possible maximum matchings over G_{a_i,b_j}, hence we use subscript k in $\mathcal{S}_k^{\mathcal{E}_{a_i,b_j}}$ to denote one such possibility. The cost of a given $\mathcal{S}_k^{\mathcal{E}_{a_i,b_j}}$ is denoted by $D(\mathcal{S}_k^{\mathcal{E}_{a_i,b_j}})$ and is defined as:

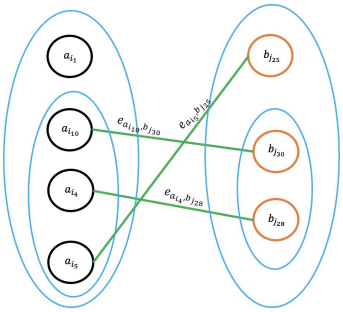
$$D(\mathcal{S}_k^{\mathcal{E}_{a_i,b_j}}) = \sum_{\forall e \in S_k^{\varepsilon_{a_i,b_j}}} w(e) \tag{6.6}$$



(a) $G_{a_{i_1},b_{j_{25}}}$



(b) $\mathcal{S}_1^{\mathcal{E}_{a_{i_1},b_{j_{25}}}} = \{e_{a_{i_4},b_{j_{28}}}, e_{a_{i_{10}},b_{j_{30}}}, e_{a_1,b_{j_{25}}}\}$



(c) $\mathcal{S}_2^{\mathcal{E}_{a_{i_1},b_{j_{25}}}} = \{e_{a_{i_{10}},b_{j_{30}}}, e_{a_{i_4},b_{j_{28}}}, e_{a_5,b_{j_{25}}}\}$

Fig. 6.3.: Consider 2 hypothetical nodes $a_{i_1} \in \mathcal{A}$ with $\mathcal{N}(a_{i_1}) = \{a_{i_{10}}, a_{i_4}, a_{i_5}\}$ and $b_{j_{25}} \in \mathcal{B}$ with $\mathcal{N}(b_{j_{25}}) = \{b_{j_{30}}, b_{j_{28}}\}$. In Fig. 6.3a, we illustrate how $G_{a_{i_1},b_{j_{25}}}$ is constructed. Note that because $G_{a_{i_1},b_{j_{25}}}$ is a complete bipartite graph there is an edge from every node in $\{a_{i_1}, \mathcal{N}(a_{i_1})\}$ to every node in $\{b_{j_{25}}, \mathcal{N}(b_{j_{25}})\}$. In Fig. 6.3b and Fig. 6.3c, examples of plausible maximum matching are shown. The value of $D(\mathcal{S}_1^{\mathcal{E}_{a_{i_1},b_{j_{25}}}}) = w(e_{a_1,b_{j_{25}}}) + w(e_{a_{i_{10}},b_{j_{30}}}) + w(e_{a_{i_4},b_{j_{28}}})$ and $D(\mathcal{S}_2^{\mathcal{E}_{a_{i_1},b_{j_{25}}}})$ can be computed in a similar manner.

Given a G_{a_i,b_j}, we want to find the maximum matching with the minimum cost. We refer to this minimum cost as $\hat{D}_{\min}(G_{a_i,b_j})$ and it is computed as:

$$\hat{D}_{\min}(G_{a_i,b_j}) = \min_{\forall k \text{ such that } \exists\ \mathcal{S}_k^{\mathcal{E}_{a_i,b_j}}} D(\mathcal{S}_k^{\mathcal{E}_{a_i,b_j}}) \tag{6.7}$$

For two superpixels a_i and b_j, $\hat{D}_{\min}(G_{a_i,b_j})$ measures the similarity between the two superpixels and the similarity between their neighborhoods. The lower the value of $\hat{D}_{\min}(G_{a_i,b_j})$, the more similar the two superpixels are both in terms of their individual characteristics and their neighboring superpixels. The contrast value of superpixel $b_j \notin \mathcal{B}^e$ in $C_a^{b,\text{neigh}}$ is denoted by $c_j^{b,\text{neigh}}$ and is computed as:

$$c_j^{b,\text{neigh}} = \min_{\forall i \text{ such that } a_i \in \mathcal{A}} \frac{\hat{D}_{\min}(G_{a_i,b_j})}{l_{a_i,b_j}} \tag{6.8}$$

In Eq. 6.8, $l_{a_i,b_j} = \min(|\{a_i, \mathcal{N}(a_i)\}|, |\{b_j, \mathcal{N}(b_j)\}|)$ where $|\{.\}|$ denotes the cardinality of the set $\{.\}$. If $b_j \in \mathcal{B}^e$ then $c_j^{b,\text{neigh}} = 0$. $\hat{D}_{\min}(G_{a_i,b_j})$ is likely to increase as l_{a_i,b_j} increases because there are more edges in maximum matching. In order to compensate this effect, we divide $\hat{D}_{\min}(G_{a_i,b_j})$ by l_{a_i,b_j} in Eq. 6.8.

6.3.3 Saliency Fusion

The contrast map C_a^b gives an estimate of the probability of pixels belonging to objects/regions present in I^b but missing in I^a. However, we would like to segment salient missing objects. As explained in Section 6.1, to find the salient missing objects, the HVS compares objects/regions in I^b that have a high value of visual saliency. Therefore, we are interested in identifying regions in the contrast map C_a^b which correspond to high visual saliency. The visual saliency information of I^b needs to be incorporated into C_a^b to obtain our final estimate $\hat{M}_a^b$, where $\hat{M}_a^b$ is the probability of each pixel in I^b belonging to the salient missing objects. We can then obtain the final

binary label M^b, by thresholding $\hat{M}_a^b$ with $T \in [0, 1]$. If S^b is the visual saliency map of I^b, then $\hat{M}_a^b$ is computed as:

$$\hat{M}_a^b = \frac{\alpha * S^b + C_a^b}{max(\alpha * S^b + C_a^b)} \tag{6.9}$$

where $max(\alpha * S^b + C_a^b)$ is the normalization term. In Eq. 6.9, α is a weighting factor between $[0, 1]$ that varies the relative contributions of S^b and C_a^b towards $\hat{M}_a^b$. The value of α is empirically computed and will be explained in Section 6.4.3. To compute S^b, we use the state-of-the-art salient object detection method R3NET [107]. We also compared our method to other deep learning based salient object detection methods such as Amulet [143], UCF [142] and NLDF [141].

6.4 Experimental Results

6.4.1 Dataset

The dataset $\mathcal{D}$ we use for evaluating our method contains 566 pairs of before eating and after eating images. Along with image pairs, ground truth masks of the salient missing objects in the before eating images (which in this case are foods) are also provided. These images are a subset of images collected from a community dwelling dietary study [136]. The images in $\mathcal{D}$ exhibit a wide variety of foods and eating scenes. Participants in this dietary study are asked to capture a pair of before and after eating scene images, denoted as I^b and I^a. A typical participant takes about 3 to 5 pairs of images per day depending on his/her eating habits. These image pairs are then sent to a cloud based server to analyze nutrient contents. $\mathcal{D}$ is split randomly into $\mathcal{D}_{val}$ (49 image pairs) and $\mathcal{D}_{test}$ (517 image pairs). $\mathcal{D}_{val}$ is used for choosing the optimal hypyerparameters namely α and the convolutional layer. More details are explained in section 6.4.3. $\mathcal{D}_{test}$ is used to evaluate the accuracy of our method compared to other methods. Examples of image pairs from $\mathcal{D}_{test}$ along with the predicted masks obtained by our method and the salient object detection methods

are shown in Fig. 6.6. $\mathcal{D}_{test}$ and $\mathcal{D}_{val}$ have very different food classes. In addition, the background of the images in $\mathcal{D}_{val}$ is very different from those in $\mathcal{D}_{test}$. This makes $\mathcal{D}$ very apt for our experiments, because $\mathcal{D}_{val}$ does not give any information about the food classes present in $\mathcal{D}_{test}$. Thus if a model tuned on $\mathcal{D}_{val}$ performs well on $\mathcal{D}_{test}$, it signifies that the model is able to segment foods without requiring information about the food class.

6.4.2 Evaluation Metrics

We use two standard metrics for evaluating the performance of the proposed method. These metrics are commonly used to assess the quality of salient object detection methods [155].

- **Precision and Recall** Consider $t = \{I^b, I^a, G^b\}$ in $\mathcal{D}$. In t, G^b represents the ground truth mask of the salient missing objects in I^b. Pixels belonging to the salient missing objects in G^b have a value of 1 and the rest have a value of 0. Our proposed method outputs $\hat{M}_a^b$ which has a range between $[0, 1]$. We can then generate a segmentation mask M^b using a threshold $T \in [0, 1]$. Given M^b and G^b, precision (P) and recall (R) are computed over $\mathcal{D}$ as:

$$\text{P} \ : \frac{\sum_{\forall t \in \mathcal{D}} |M^b \cap G^b|}{\sum_{\forall t \in \mathcal{D}} |M^b|} \ , \ \text{R} \ : \frac{\sum_{\forall t \in \mathcal{D}} |M^b \cap G^b|}{\sum_{\forall t \in \mathcal{D}} |G^b|} \tag{6.10}$$

  For a binary mask, $|\cdot|$ denotes the number of non-zero entries in it. By varying T between 0 and 1, we have different pairs of precision and recall values. When precision and recall values are plotted against each other, we obtain the precision recall (PR) curve. The information provided by precision and recall can be condensed into their weighted harmonic mean denoted by F_β, where F_β is computed as:

$$F_\beta = \frac{(1 + \beta^2) * Precision * Recall}{\beta^2 * Precision + Recall} \tag{6.11}$$

The value of lies between $[0, 1]$. A higher values of F_β indicates better performance. The value of β^2 is chosen to be 0.3 similar to other works [155]. β is a control parameter that emphasizes the importance of precision over recall. The value F_β varies as we move along the PR curve. The entire information of PR curve can be summarized by the maximal F_β denoted by $F_\beta^{\max}$, as discussed in [155, 156].

- **Receiver Operator Characteristics (ROC)** Similar to the PR curve, ROC curve is a plot of the true positive rate (TPR) against the false positive rate (FPR). TPR and FPR are defined as:

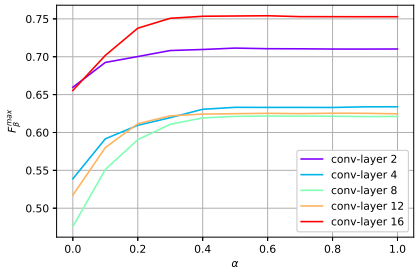
$$\text{TPR:}\frac{\sum_{\forall t \in \mathcal{D}} |M^b \cap G^b|}{\sum_{\forall t \in \mathcal{D}} |G^b|} \ , \ \text{FPR:}\frac{\sum_{\forall t \in \mathcal{D}} |M^b \cap (1 - G^b)|}{\sum_{\forall t \in \mathcal{D}} |(1 - G^b)|} \tag{6.12}$$

Similar to F_β^{max}, the entire information provided by ROC curve can be condensed into one metric called AUC, which is the area under the ROC curve. Higher values of AUC indicate better performance. A perfect method will have an AUC of 1 and a method that randomly guesses values in M^b will have an AUC of 0.5.

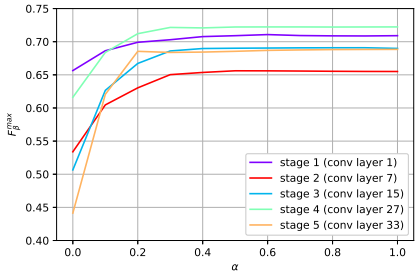
6.4.3 Experiments

Hyperparameter selection

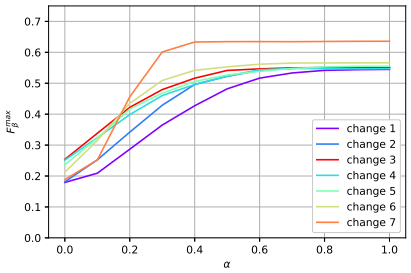
The method described in Section 3 requires 2 hyperparameters, namely α in Eq. 6.9 and the convolutional layer of VGG19 for feature extraction. To justify the use of a pre-trained VGG19 for feature extraction, we have also conducted experiments by extracting features from ResNet34 [114] and Inception-v3 [115], pre-trained on ImageNet. These experiments are conducted on $\mathcal{D}_{val}$ to find the best F_β which gives us a set of optimal hyperparameters.



(a) VGG19 $F_\beta^{\max}$ vs α



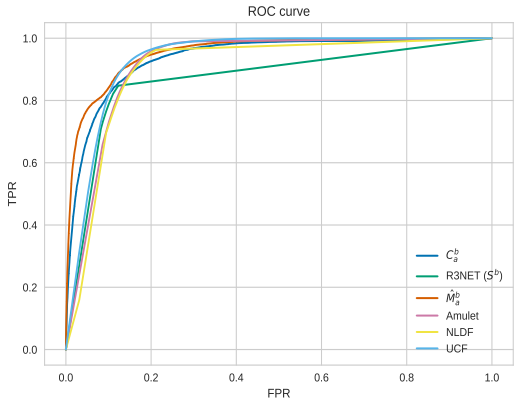
(b) ResNet34 $F_\beta^{\max}$ vs α



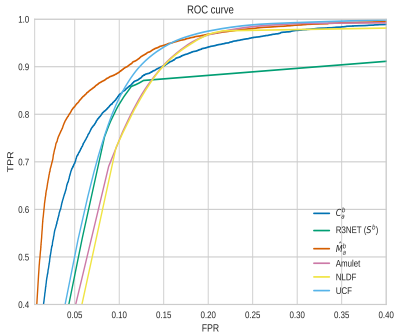
(c) Inception-v3 $F_\beta^{\max}$ vs α

Fig. 6.4.: $F_\beta^{\max}$ of $\hat{M}_a^b$ on $\mathcal{D}_{val}$ are plotted as α varies. (a) For VGG19, $F_\beta^{\max}$ is reported using features from all convolutional layers that precede a max polling layer. (b) For ResNet34, features were extracted from the output of each stage. (c) For Inception-V3, features were extracted from each layer whenever the output spatial dimensions do not match the input spatial dimensions.

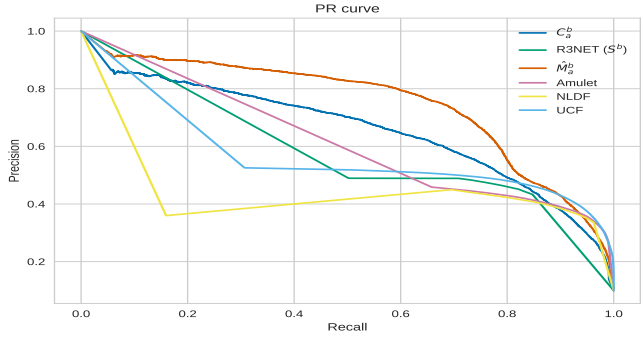
To choose the best convolutional layer, we evaluate $\hat{M}_a^b$ using features from every convolutional layer of VGG19 that precedes a max pooling layer. There are 5 such convolutional layers in VGG19. The architecture of ResNet34 can be divided into 5 stages [114]. To find the optimal layer in ResNet34, we extracted features from the



(a) ROC curve



(b) ROC curve (Zoomed)



(c) PR curve

Fig. 6.5.: ROC and PR curves of R3NET [107] (also S^b), NLDF [141], Amulet [143], UCF [142], C_a^b and $\hat{M}_a^b$ are shown in the above plots. Fig 6.5b is a zoomed in version of ROC curve in Fig 6.5a

output of each stage. The architecture of Inception-v3 is very different from those of ResNet34 and VGG19. To find the optimal layer in Inception-v3, we extract features whenever there is a change in spatial dimension as the inputs propagate through the network. There are 7 such changes occur in Inception-v3 before the average pooling operation. Please refer to architecture of Inception-v3 provided in PyTorch [157] for more details. In addition to extracting features from various convolutional layers, we also vary α from 0 to 1 in steps of 0.1. We plot F_β^{max} as α varies for every convolutional layer. The result is shown in Fig. 6.4. From Fig. 6.4, its quite evident that features from the 16^{th} convolutional layer gives the best performance compared to features from other layers. In addition it's also evident that features from VGG19 achieve better performance than features from ResNet34. For features from VGG19, the value of F_β^{max} attains its maximum value of 0.754 for $\alpha = 0.6$.

As we go deeper into the convolutional layers of VGG19, the features extracted become increasingly abstract, but suffer from decrease in resolution. Abstract features are less prone to changes in illumination, noise and pose which suits our task well. We noticed in Figure 6.4, as we go deeper into the convolutional layers, we first observe a degradation in the quality of features extracted (conv-layer 2 to conv-layer 8). This trend is reversed from conv-layer 8 to conv-layer 16 with a significant improvement of $F_\beta^{\max}$. We suspect this is because at first the negative effect of decreased resolution outweighs the benefit of abstract features. However, this trend quickly reverses from conv-layer 8 and beyond.

Testing

After obtaining the optimal hyperparameters as described in section 6.4.3, we evaluated our method on $\mathcal{D}_{test}$. $\hat{M}_a^b$ is computed for every image pair in $\mathcal{D}_{test}$ and the ROC and PR curves are computed on $\mathcal{D}_{test}$. Since our goal is to develop a class-agnostic food segmentation method, we compared the proposed method to 4 state-of-the-art salient object detection techniques, namely R3NET [107], NLDF [141],

Amulet [143] and UCF [142]. Salient object detection methods are class-agnostic and are applicable in this scenario as food is always a salient object in I_b. Since these are deep learning based methods, we use their respective pre-trained models to compute the saliency maps of I_b. The ROC and PR curves of various methods are shown in Fig. 6.5. The F_β^{max} and AUC values are reported in Table 6.1.

Table 6.1.: AUC and $F_\beta^{\max}$ values of various maps and methods.

Maps	AUC	$F_\beta^{\max}$
C_a^b	0.937	0.645
R3NET [107] (S^b)	0.871	0.527
$\hat{\mathbf{M}}_\mathbf{a}^\mathbf{b}$ (ours)	**0.954**	**0.741**
Amulet [143]	0.919	0.499
NLDF [141]	0.909	0.493
UCF [142]	0.934	0.536

6.5 Discussion

The goal of our method is to segment the salient missing objects in I^b using information from a pair of images I^a and I^b. In the contrast map generation step as described in Section 6.3.2, we provide an estimate of the probability of pixels belonging to objects/regions in I^b but missing in I^a. In the saliency fusion step as described in Section 6.3.3, saliency information of pixels in I^b is fused into the contrast map C_a^b so as to emphasize that we are looking for salient missing objects. In order to show that the various steps of our proposed method achieve their individual objectives, we plotted the PR and ROC curves of the contrast map C_a^b, the visual saliency map S^b from R3NET [107] and the estimated salient missing objects probability map $\hat{M}_a^b$ in Fig. 6.5c and Fig. 6.5a. In addition, we also plot PR and ROC curves for the 3 other salient object detection methods. From these plots, we can see that combining S^b and C_a^b as described in Section 6.3.3 improves the overall performance. This is also

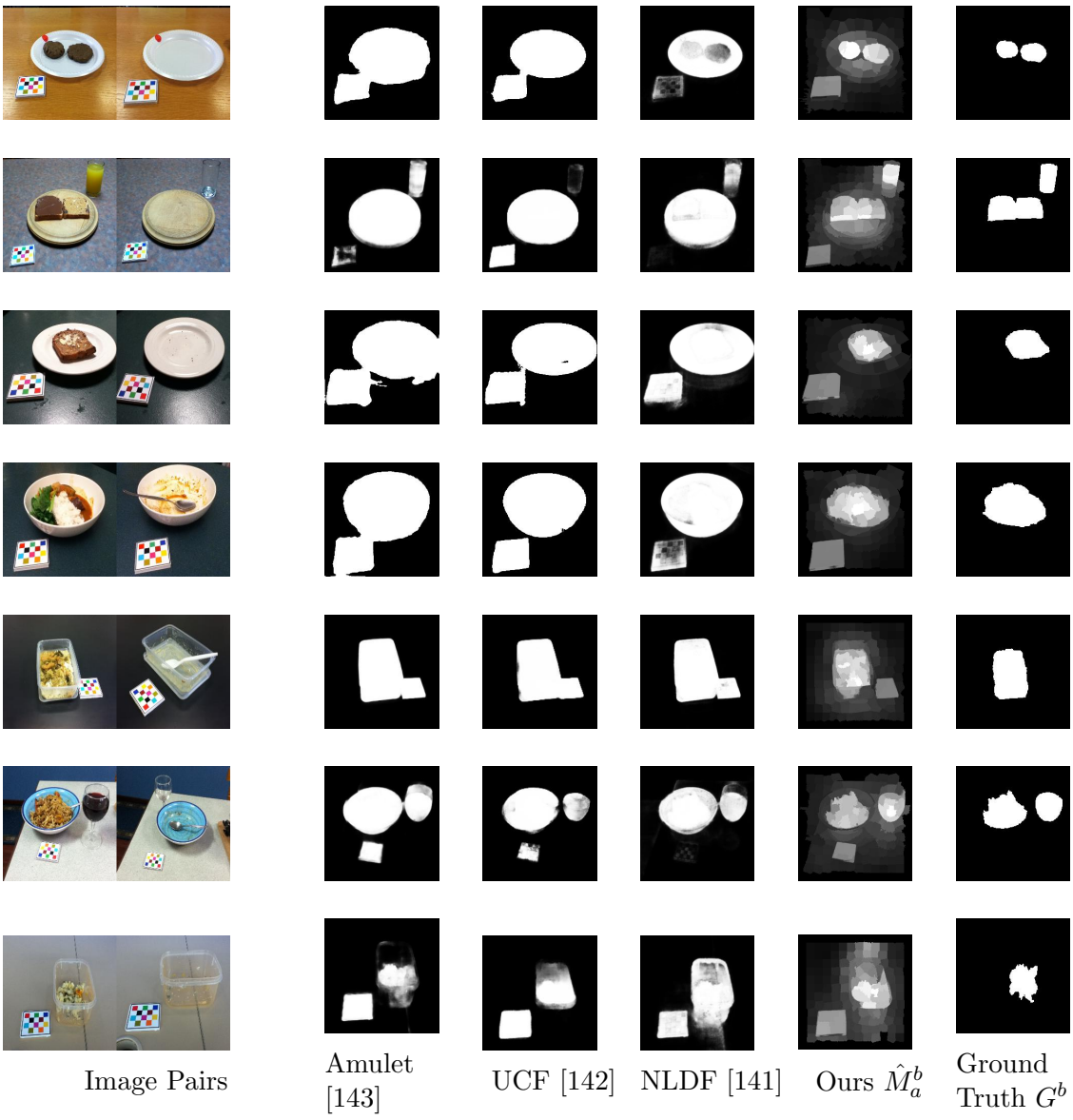


Fig. 6.6.: Sample image pairs from $\mathcal{D}_{test}$ along with various maps are shown. For every row, the first group of two images are the original before and after eating images, respectively. The second group of images are the saliency maps generated by Amulet [143], UCF [142], NLDF [141], R3NET [107] , M_a^b (our method) followed by ground truth mask G^b. The ground truth images are binary maps with pixels of value 1 representing foods and pixels of value 0 representing background. All the others are probability maps with pixels having values between 0 and 1.

illustrated in Table 6.1, where both AUC and $F_\beta^{\max}$ of $\hat{M}_a^b$ are higher than C_a^b. This is because the contrast map C_a^b by itself models all the missing objects/regions while

the probability map $\hat{M}_a^b$ also takes into account the visual saliency map S^b, which can more accurately model the salient missing objects. We can also observe from the PR and ROC curves in Fig. 6.5 and values in Table6.1 that our method achieved better performance than the state-of-the-art salient object detection methods such as R3NET [107], NLDF [141], Amulet [143] and UCF [142]. We also visually verify the performance of our method as illustrated in Fig. 6.6. The salient object detection methods Amulet [143], UCF [142] and NLDF [141] failed to detect only foods in these images, while R3NET [107] succeeded in detecting the foods but also placed equal importance to other salient objects such as the color checkerboard. Our method gave higher probability to the foods which are the salient missing objects compared to other salient objects in the scene. It must also be noted that our method did not have access to information about food classes in $\mathcal{D}_{test}$. This is because $\mathcal{D}_{val}$ and $\mathcal{D}_{test}$ have very few food classes in common. By tuning the parameters on $\mathcal{D}_{val}$, our method will not have access to information about the food classes in $\mathcal{D}_{test}$. Hence the performance of our method on $\mathcal{D}_{test}$ is indicative of its effectiveness of segmenting foods in a class-agnostic manner. These unique characteristics of $\mathcal{D}$ are also explained in section 4.1. Hence, by modeling the foods as salient missing objects, we are able to build a better class-agnostic food segmentation method compared to existing methods.

7. SUMMARY AND FUTURE WORK

7.1 Overview

In Chapter 2 we proposed a simple yet effective shadow detection method requiring few parameters. Each image was first segmented and segment pairs were identified as shadow non-shadow pairs based on their reflectance, illumination and texture characteristics. Experimental results showed that our method was effective for detecting shadows but had a lower accuracy in identifying non-shadows. The connections between the detected shadow and non-shadow pairs were used to successfully remove shadows in test images.

In Chapter 3 we proposed a shadow removal detector for forensic image analysis. Given an input image, the proposed CNN outputs a mask showing: (i) whether the image was manipulated by means of a shadow removal technique; (ii) the pixel locations where a shadow was possibly present before removal. This detector can be used as additional tool in an analyst's asset in order to counter anti-forensic attacks tailored to shadow-based forensics detectors. The proposed solution has been tested against a shadow removal method that has good performance despite been very easy to use by non-experts. As a byproduct of our investigation, we noticed that many digital integrity detectors that appear to be extremely accurate in many situations, did not achieve the same performance in our analysis. It is possible that manipulation traces left by more unconventional image processing methods (as shadow removal) are different in nature by more classical and well-studied image editing operations. In the future we will study other state-of-the-art shadow removal methods.

In Chapter 4 we proposed a solution for satellite imagery forgery detection and localization. The rationale behind the proposed method is that it is possible to train an autoencoder to obtain a compact representation of image patches coming from

pristine satellite pictures. This autoencoder can than be used as a feature extractor for image patches. During testing, a one-class SVM is used to detect whether feature vectors come from pristine images or not, thus representing forgeries. The solution proposed in Chapter 4 makes use of generative adversarial networks to train the autoencoder for the forgery detection task. Moreover, it is worth noting that the whole system is trained only on pristine data. This means that no prior knowledge on the forgeries is assumed to be available. Tests on copy-paste attacked images with different forgery size show promising accuracy in both detection and localization. Future work will be devoted to study system robustness to different kinds of forgeries as well.

In Chapter 5, we proposed a method to cluster food images based on their eating environment. Our method extracts features from a pre-trained CNN at multiple levels. These features are fused using their distance matrices and a clustering algorithm is applied after feature fusion. Our method is evaluated on a dataset containing 3137 eating scene images collected from a dietary study with a total of 585 clusters. We compared our method to state-of-the-art clustering methods and showed improved performance.

In Chapter 6, we propose a class-agnostic food segmentation method by segmenting the salient missing objects in a before eating image I^b using information from a pair of before and after eating images, I^b and I^a. We treat this problem as a paradigm of top down saliency detection where visual attention of HVS is guided by a task. Our proposed method uses I^a as background to obtain a contrast map that is an estimate of the probability of pixels of I^b belonging to objects/regions missing in I^a. The contrast map is then fused with the saliency information of I^b to obtain a probability map $\hat{M}_a^b$ for salient missing objects. Our experimental results validated that our approach achieves better performance both quantitatively and visually when compared to state-of-the-art salient object detection methods such as R3NET [107], NLDF [141], Amulet [143] and UCF [142].

7.2 Complete List of Publications

Following is an exhaustive list of the publications in which I have been personally involved during my Ph.D. studies at Purdue University :

1. **S. K. Yarlagadda**, D.M Montserrat, D.Guera, C.J. Boushey, D. Kerr, F. Zhu "Saliency aware class agnostic food image segmentation", Submitted to ACM Transactions on Computing for Healthcare,*under review (minor revision).*

2. **S. K. Yarlagadda**, S. Baireddy, D. Güera, C. J. Boushey, Kerr DA, F. Zhu, "Learning eating environments through scene clustering", *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, May, 2020.

3. **S. K. Yarlagadda** and F. Zhu, "A Reflectance based method for shadow detection and removal", *IEEE Southwest Symposium on Image Analysis and Interpretation*, pp. 9-12. Las Vegas, NV, 2018.

4. **S. K. Yarlagadda**, D. Güera, D. Mas, P. Bestagini, F. Zhu, S. Tubaro, E. J. Delp, "Shadow removal detection and localization for forensic analysis", *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2019.

5. **S. K. Yarlagadda**, D. Güera, P. Bestagini, F. Zhu, S. Tubaro, E. J. Delp, "Satellite image forgery detection and localization using GAN and one-class classifier", *Proceedings of the IS&T Electronic Imaging*, vol. 2018, no. 7, pp. 214-1-214-9, Burlingame, CA, January 2018.

6. D. Güera, **S. K. Yarlagadda**, P. Bestagini, F. Zhu, S. Tubaro, E. J. Delp, "Reliability map estimation for CNN-based camera model attribution", *Proceedings of IEEE Winter Conference on Applications of Computer Vision*, pp. 964-973, Lake Tahoe, NV, February 2018.

7. Y. Wang, J. Ribera, C. Liu, **S. K. Yarlagadda** and F. Zhu, "Pill recognition using minimal labeled data", *Proceedings of IEEE International Conference on Multimedia Big Data*, pp. 346-353, Laguna Hills, CA, 2017.

8. S. Fang, **S. K. Yarlagadda**, Y. Wang, F. Zhu, C. Boushey, D. Kerr and E. Delp, "Image based dietary behavior and analysis using deep learning", *Mini symposium at the International Conference of the IEEE Engineering in Medicine and Biology Society*, July 2018, Honolulu, HI.

9. J Horvath, D. Güera, **S. K. Yarlagadda,** P. Bestagini, F. Zhu, S. Tubaro , E. J. Delp, "Anomaly-based manipulation detection in satellite images", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, July 2019.

10. D. Mas Montserrat, H. Hao, **S. K. Yarlagadda**, S. Baireddy, R. Shao , J. Horvath, E. Bartusiak, J. Yang ,D. Güera, F. Zhu , E. J. Delp, "Deepfakes detection with automatic face weighting", *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, July 2020.

11. D. Mas Montserrat, J. Horváth, **S. K. Yarlagadda**, F. Zhu, and E. J. Delp. "Generative autoregressive ensembles for satellite imagery manipulation detection", Submitted to IEEE International Workshop on Information Forensics and Security, 2020.

REFERENCES

REFERENCES

[1] L. Verdoliva, "Media forensics and deepfakes: an overview," *arXiv preprint arXiv:2001.06564*, 2020.

[2] Y. Wang, S. Fang, C. Liu, F. Zhu, D. A. Kerr, C. J. Boushey, and E. J. Delp, "Food image analysis: The big data problem you can eat!" *50th Asilomar Conference on Signals, Systems and Computers*, pp. 1263–1267, Nov 2016. [Online]. Available: https://doi.org/10.1109/ACSSC.2016.7869576

[3] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969. [Online]. Available: https://doi.org/10.1109/TPAMI.2018.2844175

[4] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz, and D. Terzopoulos, "Image segmentation using deep learning: A survey," *arXiv preprint arXiv:2001.05566*, 2020.

[5] S. Ma, X. Zhang, C. Jia, Z. Zhao, S. Wang, and S. Wanga, "Image and video compression with neural networks: A review," *IEEE Transactions on Circuits and Systems for Video Technology*, 2019. [Online]. Available: https://doi.org/10.1109/TCSVT.2019.2910119

[6] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004. [Online]. Available: ttps://doi.org/10.1023/B:VISI.0000029664.99615.94

[7] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," *European conference on computer vision*, pp. 404–417, 2006. [Online]. Available: https://doi.org/10.1007/11744023_32

[8] A. Radhakrishna, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC Superpixels Compared to State-of-the-Art Superpixel Methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, Nov 2012. [Online]. Available: http:/doi.org/10.1109/TPAMI.2012.120

[9] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015. [Online]. Available: https://doi.org/10.1038/nature14539

[10] K. Hornik, M. Stinchcombe, H. White *et al.*, "Multilayer feedforward networks are universal approximators." *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989. [Online]. Available: https://doi.org/10.1016/0893-6080(89)90020-8

[11] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986. [Online]. Available: https://doi.org/10.1038/323533a0

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

[13] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, June 2009. [Online]. Available: http:/doi.org/10.1109/CVPR.2009.5206848

[14] J. Lalonde, A. A. Efros, and S. G. Narasimhan, "Estimating natural illumination from a single outdoor image," *IEEE International Conference on Computer Vision*, pp. 183–190, Sept 2009. [Online]. Available: https://doi.org/10.1109/ICCV.2009.5459163

[15] S. Wehrwein, K. Bala, and N. Snavely, "Shadow detection and sun direction in photo collections," *International Conference on 3D Vision*, pp. 460–468, Oct 2015. [Online]. Available: https://doi.org/10.1109/3DV.2015.58

[16] J. Gao, J. Dai, and P. Zhang, "Region-based moving shadow detection using watershed algorithm," *International Symposium on Computer, Consumer and Control (IS3C)*, pp. 846–849, July 2016. [Online]. Available: https://doi.org/10.1109/IS3C.2016.215

[17] Y. I. Shedlovska and V. V. Hnatushenko, "Shadow detection and removal using a shadow formation model," *IEEE International Conference on Data Stream Mining Processing (DSMP)*, pp. 187–190, Aug 2016. [Online]. Available: https://doi.org/10.1109/DSMP.2016.7583537

[18] J. Zhu, K. G. G. Samuel, S. Z. Masood, and M. F. Tappen, "Learning to recognize shadows in monochromatic natural images," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 223–230, June 2010. [Online]. Available: https://doi.org/10.1109/CVPR.2010.5540209

[19] J.-F. Lalonde, A. A. Efros, and S. G. Narasimhan, "Detecting ground shadows in outdoor consumer photographs," *European Conference on Computer Vision: Part II*, pp. 322–335, 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1888028.1888053

[20] S. H. Khan, M. Bennamoun, F. Sohel, and R. Togneri, "Automatic shadow detection and removal from a single image," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 3, pp. 431–446, March 2016. [Online]. Available: https://doi.org/10.1109/TPAMI.2015.2462355

[21] R. Guo, Q. Dai, and D. Hoiem, "Paired regions for shadow detection and removal," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 12, pp. 2956–2967, Dec 2013. [Online]. Available: https://doi.org/10.1109/TPAMI.2012.214

[22] K. Chung, Y. Lin, and Y. Huang, "Efficient shadow detection of color aerial images based on successive thresholding scheme," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 47, no. 2, pp. 671–682, Feb 2009. [Online]. Available: https://doi.org/10.1109/TGRS.2008.2004629

[23] Q. Wu, W. Zhang, and B. V. K. V. Kumar, "Strong shadow removal via patch-based shadow edge detection," *IEEE International Conference on Robotics and Automation*, pp. 2177–2182, May 2012. [Online]. Available: https://doi.org/10.1109/ICRA.2012.6224561

[24] A. Vedaldi and S. Soatto, "Quick shift and kernel methods for mode seeking," *European Conference on Computer Vision*, pp. 705–718, 2008. [Online]. Available: https://doi.org/10.1007/978-3-540-88693-8_52

[25] D. R. Martin, C. C. Fowlkes, and J. Malik, "Learning to detect natural image boundaries using local brightness, color, and texture cues," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 5, pp. 530–549, May 2004. [Online]. Available: https://doi.org/10.1109/TPAMI.2004.1273918

[26] F. Tanner, B. Colder, C. Pullen, D. Heagy, C. Oertel, and P. Sallee, "Overhead imagery research data set (OIRDS) an annotated dat a library and tools to aid in the development of computer vision algorithms, 2009."

[27] A. Rocha, W. Scheirer, T. Boult, and S. Goldenstein, "Vision of the unseen: Current trends and challenges in digital image and video forensics," *ACM Computing Surveys*, vol. 43, no. 4, pp. 26:1–26:42, Oct. 2011. [Online]. Available: http://doi.acm.org/10.1145/1978802.1978805

[28] A. Piva, "An overview on image forensics," *ISRN Signal Processing*, vol. 2013, p. 22, November 2013. [Online]. Available: http://dx.doi.org/10.1155/2013/496701

[29] M. C. Stamm, Min Wu, and K. J. R. Liu, "Information forensics: An overview of the first decade," *IEEE Access*, vol. 1, pp. 167–200, May 2013. [Online]. Available: http://dx.doi.org/10.1109/ACCESS.2013.2260814

[30] M. Kirchner and T. Gloe, "Forensic Camera Model Identification," in *Handbook of Digital Forensics of Multimedia Data and Devices*. John Wiley & Sons, Ltd, 2015. [Online]. Available: http://dx.doi.org/10.1002/9781118705773.ch9

[31] A. Tuama, F. Comby, and M. Chaumont, "Camera model identification with the use of deep convolutional neural networks," *IEEE International Workshop on Information Forensics and Security*, pp. 1–6, Dec 2016. [Online]. Available: http://dx.doi.org/10.1109/WIFS.2016.7823908

[32] L. Verdoliva, D. Cozzolino, and G. Poggi, "A feature-based approach for image tampering detection and localization," *IEEE International Workshop on Information Forensics and Security*, pp. 149–154, Dec 2014, Atlanta, GA. [Online]. Available: http://dx.doi.org/10.1109/WIFS.2014.7084319

[33] L. Gaborini, P. Bestagini, S. Milani, M. Tagliasacchi, and S. Tubaro, "Multi-clue image tampering localization," pp. 125–130, Dec 2014. [Online]. Available: http://dx.doi.org/10.1109/WIFS.2014.7084315

[34] A. C. Popescu and H. Farid, "Exposing digital forgeries by detecting traces of resampling," *IEEE Transactions on Signal Processing*, vol. 53, no. 2, pp. 758–767, Feb 2005. [Online]. Available: http://dx.doi.org/10.1109/TSP.2004.839932

[35] M. Kirchner, "Fast and reliable resampling detection by spectral analysis of fixed linear predictor residue," *ACM Workshop on Multimedia and Security*, pp. 11–20, 2008. [Online]. Available: http://doi.acm.org/10.1145/1411328.1411333

[36] D. Vázquez-Padín and F. Pérez-González, "Prefilter design for forensic resampling estimation," *IEEE International Workshop on Information Forensics and Security*, pp. 1–6, Nov 2011. [Online]. Available: http://dx.doi.org/10.1109/WIFS.2011.6123133

[37] T. Bianchi and A. Piva, "Image forgery localization via block-grained analysis of jpeg artifacts," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 3, pp. 1003–1017, June 2012. [Online]. Available: http://dx.doi.org/10.1109/TIFS.2012.2187516

[38] M. Barni, L. Bondi, N. Bonettini, P. Bestagini, A. Costanzo, M. Maggini, B. Tondi, and S. Tubaro, "Aligned and non-aligned double JPEG detection using convolutional neural networks," *Journal of Visual Communication and Image Representation*, vol. 49, no. Supplement C, pp. 153–163, November 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S104732031730175X

[39] E. Wengrowski, Z. H. Sun, and A. Hoogs, "Reflection correspondence for exposing photograph manipulation," *IEEE International Conference on Image Processing*, pp. 4317–4321, Sep. 2017, Beijing, China. [Online]. Available: http://dx.doi.org/10.1109/ICIP.2017.8297097

[40] M. K. Johnson and H. Farid, "Exposing digital forgeries by detecting inconsistencies in lighting," *ACM Workshop on Multimedia and Security*, pp. 1–10, 2005, New York, NY. [Online]. Available: http://doi.acm.org/10.1145/1073170.1073171

[41] T. J. d. Carvalho *et al.*, "Exposing digital image forgeries by illumination color classification," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 7, pp. 1182–1194, Jul. 2013. [Online]. Available: http://dx.doi.org/10.1109/TIFS.2013.2265677

[42] Q. Liu, X. Cao, C. Deng, and X. Guo, "Identifying image composites through shadow matte consistency," *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 3, pp. 1111–1122, sep 2011. [Online]. Available: http://dx.doi.org/10.1109/TIFS.2011.2139209

[43] E. Kee, J. F. O'Brien, and H. Farid, "Exposing photo manipulation with inconsistent shadows," *ACM Transactions on Graphics*, vol. 32, no. 3, pp. 28:1–28:12, July 2013. [Online]. Available: http://doi.acm.org/10.1145/2487228.2487236

[44] V. Tuba, R. Jovanovic, and M. Tuba, "Digital image forgery detection based on shadow hsv inconsistency," *IEEE International Symposium on Digital Forensic and Security*, pp. 1–6, Apr 2017, Tirgu Mures, Romania. [Online]. Available: http://dx.doi.org/10.1109/ISDFS.2017.7916505

[45] H. Gong, D. Cosker, C. Li, and M. Brown, "User-aided single image shadow removal," *Proceedings of the IEEE International Conference on Multimedia and Expo*, pp. 1–6, Jul. 2013, San Jose, CA. [Online]. Available: http://dx.doi.org/10.1109/ICME.2013.6607463

[46] Q. Yang, K. Tan, and N. Ahuja, "Shadow removal using bilateral filtering," *IEEE Transactions on Image Processing*, vol. 21, no. 10, pp. 4361–4368, Oct. 2012. [Online]. Available: http://dx.doi.org/10.1109/TIP.2012.2208976

[47] J. Wang, X. Li, and J. Yang, "Stacked conditional generative adversarial networks for jointly learning shadow detection and shadow removal," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1788–1797, Jul. 2018, Salt Lake City, UT.

[48] H. Gong and D. Cosker, "Interactive shadow removal and ground truth for variable scene categories," *Proceedings of the British Machine Vision Conference*, Sep. 2014, Nottingham, UK. [Online]. Available: http://dx.doi.org/10.5244/C.28.36

[49] Y. Ke, F. Qin, W. Min, and G. Zhang, "Exposing image forgery by detecting consistency of shadow," *The Scientific World Journal*, vol. 2014, p. 9, Mar 2014. [Online]. Available: https://doi.org/10.1155/2014/364501

[50] D. Cozzolino, G. Poggi, and L. Verdoliva, "Splicebuster: A new blind image splicing detector," *Proceedings of the IEEE International Workshop on Information Forensics and Security*, pp. 1–6, Nov. 2015. [Online]. Available: http://dx.doi.org/10.1109/WIFS.2015.7368565

[51] L. Bondi, S. Lameri, D. Güera, P. Bestagini, E. J. Delp, and S. Tubaro, "Tampering detection and localization through clustering of camera-based CNN features," *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 1855–1864, July 2017, Honolulu, HI. [Online]. Available: http://dx.doi.org/10.1109/CVPRW.2017.232

[52] P. Isola, J. Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5967–5976, Jul 2017, Honolulu, HI. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2017.632

[53] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Oct. 2015, Munich, Germany. [Online]. Available: http://arxiv.org/abs/1505.04597

[54] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *International Conference on Learning Representations*, May 2015, San Diego, CA. [Online]. Available: http://arxiv.org/abs/1412.6980

[55] M. Zampoglou, S. Papadopoulos, and Y. Kompatsiaris, "Large-scale evaluation of splicing localization algorithms for web images," *Multimedia Tools and Applications*, vol. 76, no. 4, pp. 4801–4834, Feb. 2017. [Online]. Available: https://doi.org/10.1007/s11042-016-3795-2

[56] G. Geography, *15 Free Satellite Imagery Data Sources*, August 2017 (accessed January 1, 2018), *http://gisgeography.com/free-satellite-imagery-data-list*.

[57] B. News, *Conspiracy Files: Who shot down MH17?*, April 2016 (accessed January 1, 2018), *http://www.bbc.com/news/magazine-35706048*.

[58] Mashable, *Satellite images show clearly that Russia faked its MH17 report*, May 2015 (accessed January 1, 2018), *http://mashable.com/2015/05/31/russia-fake-mh17-report*.

[59] A. T. S. Ho, X. Zhu, and W. M. Woon, "A semi-fragile pinned sine transform watermarking system for content authentication of satellite images," *IEEE International Geoscience and Remote Sensing Symposium*, January 2005, Seoul, Korea. [Online]. Available: http://dx.doi.org/10.1109/IGARSS.2005.1525212

[60] L. Ali, T. Kasetkasem, F. G. Khan, T. Chanwimaluang, and H. Nakahara, "Identification of inpainted satellite images using evalutionary artificial neural network (EANN) and k-nearest neighbor (KNN) algorithm," *Proceedings of the IEEE International Conference of Information and Communication Technology for Embedded Systems*, May 2017, Chonburi, Thailand. [Online]. Available: http://dx.doi.org/10.1109/ICTEmSys.2017.7958765

[61] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA: MIT Press, 2016.

[62] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *arXiv:1512.03385*, December 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[63] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," *arXiv:1703.06870v2*, April 2017. [Online]. Available: http://arxiv.org/abs/1703.06870

[64] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Proceedings of the Neural Information Processing Systems Conference*, pp. 1097–1105, December 2012, lake Tahoe, NV. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[65] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge," *arXiv:1409.0575v3*, January 2015. [Online]. Available: http://arxiv.org/abs/1409.0575

[66] Y. Qian, J. Dong, W. Wang, and T. Tan, "Deep learning for steganalysis via convolutional neural networks," *Proceedings of the SPIE/IS&T Electronic Imaging Conference*, vol. 9409, p. 10, January 2015, san Francisco, CA. [Online]. Available: http://proceedings.spiedigitallibrary.org/proceeding.aspx?doi=10.1117/12.2083479http://www.scopus.com/inward/record.url?eid=2-s2.0-84932146885{&}partnerID=tZOtx3y1

[67] L. Pibre, P. Jérôme, D. Ienco, and M. Chaumont, "Deep learning for steganalysis is better than a rich model with an ensemble classifier, and is natively robust to the cover source-mismatch," *IS&T International Symposium on Electronic Imaging*, vol. 2016, no. 8, February 2016, san Francisco, CA. [Online]. Available: http://arxiv.org/abs/1511.04855

[68] G. Xu, H. Z. Wu, and Y. Q. Shi, "Structural design of convolutional neural networks for steganalysis," *IEEE Signal Processing Letters*, vol. 23, no. 5, pp. 708–712, May 2016. [Online]. Available: http://dx.doi.org/10.1109/LSP.2016.2548421

[69] V. Sedighi and J. Fridrich, "Histogram layer, moving convolutional neural networks towards feature-based steganalysis," *Proceedings of the IS&T International Symposium on Electronic Imaging*, vol. 2017, no. 7, January 2017, burlingame, CA.

[70] C. Jiansheng, K. Xiangui, L. Ye, and Z. J. Wang, "Median filtering forensics based on convolutional neural networks," *IEEE Signal Processing Letters*, vol. 22, no. 11, pp. 1849–1853, June 2015. [Online]. Available: http://dx.doi.org/10.1109/LSP.2015.2438008

[71] B. Bayar and M. C. Stamm, "A deep learning approach to universal image manipulation detection using a new convolutional layer," *ACM Workshop on Information Hiding and Multimedia Security*, pp. 5–10, June 2016, Vigo, Spain. [Online]. Available: http://doi.acm.org/10.1145/2909827.2930786

[72] A. Tuama, F. Comby, and M. Chaumont, "Camera model identification based on machine learning approach with high order statistics features," *IEEE European Signal Processing Conference*, pp. 1183–1187, August 2016, Budapest, Hungary. [Online]. Available: http://dx.doi.org/10.1109/EUSIPCO.2016.7760435

[73] L. Bondi, L. Baroffio, D. Güera, P. Bestagini, E. J. Delp, and S. Tubaro, "First steps toward camera model identification with convolutional neural networks," *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 259–263, March 2017. [Online]. Available: http://dx.doi.org/10.1109/LSP.2016.2641006

[74] D. Güera, Y. Wang, L. Bondi, P. Bestagini, S. Tubaro, and E. J. Delp, "A counter-forensic method for CNN-based camera model identification," *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 1840–1847, July 2017, Honolulu, HI. [Online]. Available: dx.doi.org/10.1109/CVPRW.2017.230

[75] Q. Wang and R. Zhang, "Double JPEG compression forensics based on a convolutional neural network," *EURASIP Journal on Information Security*, vol. 2016, no. 1, p. 23, December 2016. [Online]. Available: https://doi.org/10.1186/s13635-016-0047-y

[76] S.Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv:1502.03167v3*, March 2015. [Online]. Available: http://arxiv.org/abs/1502.03167

[77] D. Cozzolino and L. Verdoliva, "Single-image splicing localization through autoencoder-based anomaly detection," *IEEE International Workshop on Information Forensics and Security*, pp. 1–6, December 2016, abu Dhabi, United Arab Emirates. [Online]. Available: http://doi.org/10.1109/WIFS.2016.7823921

[78] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2536–2544, June 2016, las Vegas, NV. [Online]. Available: http://doi.org/10.1109/CVPR.2016.278

[79] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Neural Information Processing Systems Conference*, pp. 2672–2680, December 2014, montréal, Canada.

[80] A. W. S. Inc., *Landsat on AWS*, Accessed January 1, 2018, *https://aws.amazon.com/public-datasets/landsat/*.

[81] N. Aeronautics and S. Administration, *Landsat Science*, Accessed January 1, 2018, *https://landsat.gsfc.nasa.gov/*.

[82] ——, *NASA*, Accessed January 1, 2018, *https://www.nasa.gov/*.

[83] U. G. Survey, *USGS.gov — Science for a changing world*, Accessed January 1, 2018, *https://www.usgs.gov/*.

[84] L. van der Maaten and G. Hinton, "Visualizing high-dimensional data using t-SNE," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, November 2008.

[85] W. H. Organization *et al.*, "Public spending on health: a closer look at global trends," World Health Organization, Tech. Rep., 2018.

[86] A. E. Mesas, M. Muñoz-Pareja, E. López-García, and F. Rodríguez-Artalejo, "Selected eating behaviours and excess body weight: a systematic review," *Obes Rev*, vol. 13, no. 2, pp. 106–135, Feb 2012, 21955734[pmid]. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/21955734

[87] K. Nordström, C. Coff, H. Jönsson, L. Nordenfelt, and U. Görman, "Food and health: individual, cultural, or scientific matters?" *Genes Nutr*, vol. 8, no. 4, pp. 357–363, Jul 2013, 23494484[pmid]. [Online]. Available: http://doi.org/10.1007/s12263-013-0336-8

[88] G. Garcia, T. S. Sunil, and P. Hinojosa, "The fast food and obesity link: consumption patterns and severity of obesity," *Obes Surg*, vol. 22, no. 5, pp. 810–818, May 2012. [Online]. Available: http://doi.org/10.1007/s11695-012-0601-8

[89] P. R. Deshmukh-Taskar, T. A. Nicklas, C. E. O'Neil, D. R. Keast, J. D. Radcliffe, and S. Cho, "The relationship of breakfast skipping and type of breakfast consumption with nutrient intake and weight status in children and adolescents: the National Health and Nutrition Examination Survey 1999-2006," *J Am Diet Assoc*, vol. 110, no. 6, pp. 869–878, Jun 2010. [Online]. Available: http:/doi.org/10.1016/j.jada.2010.03.023

[90] A. J. Hammons and B. H. Fiese, "Is frequency of shared family meals related to the nutritional health of children and adolescents?" *Pediatrics*, vol. 127, no. 6, pp. e1565–1574, Jun 2011. [Online]. Available: http:/doi.org/10.1542/peds.2010-1440

[91] W. Min, S. Jiang, L. Liu, Y. Rui, and R. Jain, "A survey on food computing," *ACM Comput. Surv.*, vol. 52, no. 5, pp. 92:1–92:36, Sep. 2019. [Online]. Available: http://doi.acm.org/10.1145/3329168

[92] S. L. Booth, J. F. Sallis, C. Ritenbaugh, J. O. Hill, L. L. Birch, L. D. Frank, K. Glanz, D. A. Himmelgreen, M. Mudd, B. M. Popkin *et al.*, "Environmental and societal factors affect food choice and physical activity: Rationale, influences, and leverage points," *Nutrition Reviews*, vol. 59, no. 3, pp. S21–S36, 2001. [Online]. Available: http:/doi.org/10.1111/j.1753-4887.2001.tb06983.x

[93] J. O. Hill, H. R. Wyatt, G. W. Reed, and J. C. Peters, "Obesity and the environment: Where do we go from here?" *Science*, vol. 299, no. 5608, pp. 853–855, 2003. [Online]. Available: http://doi.org/10.1126/science.1079857

[94] B. Wansink and E. Van Kleef, "Dinner rituals that correlate with child and adult bmi," *Obesity (Silver Spring, Md.)*, vol. 22, 05 2014. [Online]. Available: http:/doi.org/10.1002/oby.20629

[95] B. M. Appelhans, E. Segawa, I. Janssen, L. M. Nackers, R. Kazlauskaite, A. Baylin, J. W. Burns, L. H. Powell, and H. M. Kravitz, "Meal preparation and cleanup time and cardiometabolic risk over 14years in the study of women's health across the nation (swan)," *Preventive medicine*, vol. 71, 12 2014. [Online]. Available: http:/doi.org/10.1016/j.ypmed.2014.11.025

[96] J.-S. Shim, K. Oh, and H. C. Kim, "Dietary assessment methods in epidemiologic studies," *Epidemiol Health*, vol. 36, pp. e2 014 009–e2 014 009, Jul 2014, 25078382[pmid]. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/25078382

[97] F. Zhu, M. Bosch, I. Woo, S. Kim, C. J. Boushey, D. S. Ebert, and E. J. Delp, "The use of mobile devices in aiding dietary assessment and evaluation," *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 4, pp. 756–766, Aug 2010.

[98] K. Aizawa and M. Ogawa, "Foodlog: Multimedia tool for healthcare applications," *IEEE MultiMedia*, vol. 22, no. 2, pp. 4–8, Apr 2015. [Online]. Available: http://doi.org/10.1109/MMUL.2015.39

[99] E. Howes, C. J. Boushey, D. A. Kerr, E. J. Tomayko, and M. Cluskey, "Image-Based Dietary Assessment Ability of Dietetics Students and Interns," *Nutrients*, vol. 9, no. 2, p. 114, feb 2017. [Online]. Available: http:/doi.org/10.3390/nu9020114

[100] F. Zhu, M. Bosch, N. Khanna, C. J. Boushey, and E. J. Delp, "Multiple Hypotheses Image Segmentation and Classification With Application to Dietary Assessment," *IEEE Journal of Biomedical and Health Informatics*, vol. 19, no. 1, pp. 377–388, jan 2015. [Online]. Available: http://doi.org/10.1109/JBHI.2014.2304925

[101] Fang, Shao, Kerr, Boushey, and Zhu, "An End-to-End Image-Based Automatic Food Energy Estimation Technique Based on Learned Energy Distribution Images: Protocol and Methodology," *Nutrients*, vol. 11, no. 4, p. 877, apr 2019. [Online]. Available: http:/doi.org/10.3390/nu11040877

[102] Y. Wang, Y. He, C. J. Boushey, F. Zhu, and E. J. Delp, "Context based image analysis with application in dietary assessment and evaluation," *Multimedia Tools and Applications*, vol. 77, no. 15, pp. 19 769–19 794, aug 2018. [Online]. Available: http:/doi.org/10.1007/s11042-017-5346-x

[103] D. A. Kerr, A. J. Harray, C. M. Pollard, S. S. Dhaliwal, E. J. Delp, P. A. Howat, M. R. Pickering, Z. Ahmad, X. Meng, I. S. Pratt *et al.*, "The connecting health and technology study: a 6-month randomized controlled trial to improve nutrition behaviours using a mobile food record and text messaging support in young adults," *The international journal of behavioral nutrition and physical activity.*, vol. 13, no. 1, 2016.

[104] Rui Xu and D. Wunsch, "Survey of clustering algorithms," *IEEE Transactions on Neural Networks*, vol. 16, no. 3, pp. 645–678, May 2005. [Online]. Available: http:/doi.org/10.1109/TNN.2005.845141

[105] E. Min, X. Guo, Q. Liu, G. Zhang, J. Cui, and J. Long, "A survey of clustering with deep learning: From the perspective of network architecture," *IEEE Access*, vol. 6, pp. 39 501–39 514, 2018. [Online]. Available: http:/doi.org/10.1109/ACCESS.2018.2855437

[106] S. Wazarkar and B. N. Keshavamurthy, "A survey on image data analysis through clustering techniques for real world applications," *Journal of Visual Communication and Image Representation*, vol. 55, pp. 596–626, 2018. [Online]. Available: https://doi.org/10.1016/j.jvcir.2018.07.009

[107] Z. Deng, X. Hu, L. Zhu, X. Xu, J. Qin, G. Han, and P.-A. Heng, "R3net: Recurrent residual refinement network for saliency detection," *International Joint Conference on Artificial Intelligence*, pp. 684–690, July 2018. [Online]. Available: http://dl.acm.org/citation.cfm?id=3304415.3304513

[108] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *International Conference on Learning Representations*, May 2015, San Diego, CA. [Online]. Available: https://arxiv.org/abs/1409.1556

[109] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International Conference on Computer Vision*, 2011, pp. 2564–2571. [Online]. Available: http:/doi.org/10.1109/ICCV.2011.6126544

[110] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *science*, vol. 315, no. 5814, pp. 972–976, 2007. [Online]. Available: http:/doi.org/10.1126/science.1136800

[111] 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*, vol. 96, no. 34, 1996, pp. 226–231.

[112] D. Comaniciu and P. Meer, "Mean shift: a robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, May 2002.

[113] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, "Optics: ordering points to identify the clustering structure," in *ACM Sigmod record*, vol. 28, no. 2. ACM, 1999, pp. 49–60. [Online]. Available: https://doi.org/10.1145/304181.304187

[114] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *The IEEE Conference on Computer Vision and Pattern Recognition*, June 2016. [Online]. Available: https://arxiv.org/abs/1512.03385

[115] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," pp. 2818–2826, June 2016. [Online]. Available: https://arxiv.org/abs/1512.00567

[116] W. H. Organization, *Global Health Risks Mortality and Burden of Disease Attributable to Selected Major Risks*. World Health Organization, 2009. [Online]. Available: https://apps.who.int/iris/handle/10665/44203

[117] C. Piernas and B. M. Popkin, "Food portion patterns and trends among u.s. children and the relationship to total eating occasion size, 1977-2006," *J Nutr*, vol. 141, no. 6, pp. 1159–1164, Jun 2011. [Online]. Available: http:/doi.org/10.3945/jn.111.138727

[118] B. L. Daugherty, T. E. Schap, R. Ettienne-Gittens, F. M. Zhu, M. Bosch, E. J. Delp, D. S. Ebert, D. A. Kerr, and C. J. Boushey, "Novel Technologies for Assessing Dietary Intake: Evaluating the Usability of a Mobile Telephone Food Record Among Adults and Adolescents," *Journal of Medical Internet Research*, vol. 14, no. 2, p. e58, Apr 2012. [Online]. Available: http:/doi.org/10.2196/jmir.1967

[119] J.-S. Shim, K. Oh, and H. C. Kim, "Dietary assessment methods in epidemiologic studies," *Epidemiol Health*, vol. 36, pp. e2 014 009–e2 014 009, Jul 2014, 25078382[pmid]. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/25078382

[120] Y. Kawano and K. Yanai, "Foodcam: A real-time food recognition system on a smartphone," *Multimedia Tools and Applications*, vol. 74, no. 14, pp. 5263–5287, Jul 2015. [Online]. Available: http:/doi.org/10.1007/978-3-319-04117-9_38

[121] W. Zhang, Q. Yu, B. Siddiquie, A. Divakaran, , and H. Sawhney, ""Snap-n-Eat": Food Recognition and Nutrition Estimation on a Smartphone," *J Diabetes Sci Technol*, vol. 9, no. 3, pp. 525–533, May 2015. [Online]. Available: http:/doi.org/10.1177/1932296815582222

[122] M. F. Vasiloglou, S. Mougiakakou, E. Aubry, A. Bokelmann, R. Fricker, F. Gomes, C. Guntermann, A. Meyer, D. Studerus, and Z. Stanga, "A comparative study on carbohydrate estimation: Gocarb vs. dietitians," *Nutrients*, vol. 10, no. 6, p. 741, 2018. [Online]. Available: http:/doi.org/10.3390/nu10060741

[123] J. Dehais, M. Anthimopoulos, and S. Mougiakakou, "Food image segmentation for dietary assessment," *Proceedings of the 2nd International Workshop on Multimedia Assisted Dietary Management*, pp. 23–28, 2016. [Online]. Available: http:/doi.org/10.1145/2986035.2986047

[124] F. Kong and J. Tan, "Dietcam: Automatic dietary assessment with mobile camera phones," *Pervasive and Mobile Computing*, vol. 8, no. 1, pp. 147 – 163, 2012.

[125] S. Fang, F. Zhu, C. J. Boushey, and E. J. Delp, "The use of co-occurrence patterns in single image based food portion estimation," *IEEE Global Conference on Signal and Information Processing*, pp. 462–466, Nov 2017. [Online]. Available: http:/doi.org/10.1109/GlobalSIP.2017.8308685

[126] S. Fang, C. Liu, F. Zhu, E. J. Delp, and C. J. Boushey, "Single-view food portion estimation based on geometric models," *IEEE International Symposium on Multimedia*, pp. 385–390, Dec 2015. [Online]. Available: http:/doi.org/10.1109/ISM.2015.67

[127] S. Fang, Z. Shao, R. Mao, C. Fu, E. J. Delp, F. Zhu, D. A. Kerr, and C. J. Boushey, "Single-view food portion estimation: Learning image-to-energy mappings using generative adversarial networks," *2018 25th IEEE International Conference on Image Processing (ICIP)*, pp. 251–255, Oct 2018. [Online]. Available: http:/doi.org/10.1109/ICIP.2018.8451461

[128] K. Okamoto and K. Yanai, "An automatic calorie estimation system of food images on a smartphone," *International Workshop on Multimedia Assisted Dietary Management*, pp. 63–70, 2016.

[129] S. S. Pouladzadeh, Parisa and R. Al-Maghrabi, "Measuring calorie and nutrition from food image," *IEEE Transactions on Instrumentation and Measurement*, vol. 63, no. 8, pp. 1947–1956, Aug 2014.

[130] Y. Wang, F. Zhu, C. J. Boushey, and E. J. Delp, "Weakly supervised food image segmentation using class activation maps," *IEEE International Conference on Image Processing*, pp. 1277–1281, Sep 2017.

[131] W. Shimoda and K. Yanai, "Cnn-based food image segmentation without pixel-wise annotation," *New Trends in Image Analysis and Processing – ICIAP Workshops*, 2015.

[132] L. Bossard, M. Guillaumin, and L. Van Gool, "Food-101 – mining discriminative components with random forests," *European Conference on Computer Vision*, pp. 446–461, 2014.

[133] Y. Matsuda, H. Hoashi, and K. Yana, "Recognition of multiple-food images by detecting candidate regions," *IEEE International Conference on Multimedia and Expo*, pp. 25–30, July 2012.

[134] H.-C. Chen, W. Jia, X. Sun, Z. Li, Y. Li, J. D. Fernstrom, L. E. Burke, T. Baranowski, and M. Sun, "Saliency-aware food image segmentation for personal dietary assessment using a wearable computer," *Measurement Science and Technology*, vol. 26, no. 2, p. 025702, 2015.

[135] Y. Kawano and K. Yanai, "Automatic expansion of a food image dataset leveraging existing categories with domain adaptation," *European Conference on Computer Vision 2014 Workshops*, pp. 3–17, 2015.

[136] D. A. Kerr, A. J. Harray, C. M. Pollard, S. S. Dhaliwal, E. J. Delp, P. A. Howat, M. R. Pickering, Z. Ahmad, X. Meng, I. S. Pratt, J. L. Wright, K. R. Kerr, and C. J. Boushey, "The connecting health and technology study: a 6-month randomized controlled trial to improve nutrition behaviours using a mobile food record and text messaging support in young adults," *The international journal of behavioral nutrition and physical activity.*, vol. 13, no. 1, 2016. [Online]. Available: http:/doi.org/10.1186/s12966-016-0376-8

[137] A. Borji, D. N. Sihite, and L. Itti, "Quantitative analysis of human-model agreement in visual saliency modeling: A comparative study," *IEEE Transactions on Image Processing*, vol. 22, no. 1, pp. 55–69, Jan 2013. [Online]. Available: http:/doi.org/10.1109/TIP.2012.2210727

[138] A. Borji and L. Itti, "State-of-the-art in visual attention modeling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 185–207, Jan 2013. [Online]. Available: http:/doi.org/10.1109/TPAMI.2012.89

[139] D. H. Ballard, M. M. Hayhoe, and J. B. Pelz, "Memory representations in natural tasks," *Journal of cognitive neuroscience.*, vol. 7, no. 1, pp. 66–80, 1995. [Online]. Available: http:/doi.org/10.1162/jocn.1995.7.1.66

[140] R. J. Peters and L. Itti, "Beyond bottom-up: Incorporating task-dependent influences into a computational model of spatial attention," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, June 2007. [Online]. Available: http:/doi.org/10.1109/CVPR.2007.383337

[141] Z. Luo, A. Mishra, A. Achka, J. Eichel, S. Li, and P.-M. Jodoin, "Non-local deep features for salient object detection," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6593–6601, July 2017. [Online]. Available: http:/doi.o10.1109/CVPR.2017.698

[142] P. Zhang, D. Wang, H. Lu, H. Wang, and B. Yi, "Learning uncertain convolutional features for accurate saliency detection," *IEEE International Conference on Computer Vision*, pp. 212–221, Oct 2017. [Online]. Available: http:/doi.org/10.1109/ICCV.2017.32

[143] P. Zhang, D. Wang, H. Lu, H. Wang, and X. Ruan, "Amulet: Aggregating multi-level convolutional features for salient object detection," *IEEE International Conference on Computer Vision*, pp. 202–211, Oct 2017. [Online]. Available: http:/doi.org/10.1109/ICCV.2017.31

[144] C. Cao, X. Liu, Y. Yang, Y. Yu, J. Wang, Z. Wang, Y. Huang, L. Wang, C. Huang, W. Xu, D. Ramanan, and T. S. Huang, "Look and think twice: Capturing top-down visual attention with feedback convolutional neural networks," *IEEE International Conference on Computer Vision*, pp. 2956–2964, Dec 2015. [Online]. Available: http:/doi.org/10.1109/ICCV.2015.338

[145] V. Ramanishka, A. Das, and J. Z. K. Saenko, "Top-down visual saliency guided by captions," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3135–3144, July 2017. [Online]. Available: http:/doi.org/10.1109/CVPR.2017.334

[146] S. Khan, X. He, F. Porikli, M. Bennamoun, F. Sohel, and R. Togneri, "Learning deep structured network for weakly supervised change detection," *International Joint Conference on Artificial Intelligence*, pp. 2008–2015, 2017. [Online]. Available: http://dl.acm.org/citation.cfm?id=3172077.3172167

[147] K. Sakurada and T. Okatani, "Change detection from a street image pair using cnn features and superpixel segmentation," *British Machine Vision Conference*, pp. 61.1–61.12, September 2015. [Online]. Available: http:/doi.org/10.5244/C.29.61

[148] S. H. Khan, X. He, F. Porikli, and M. Bennamoun, "Forest change detection in incomplete satellite images with deep neural networks," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 9, pp. 5407–5423, Sept 2017. [Online]. Available: http:/doi.org/10.1109/TGRS.2017.2707528

[149] A. P. A. N. R. Rengarajan, Vijay and G. Seetharaman, "Efficient change detection for very large motion blurred images," *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 315–322, June 2014. [Online]. Available: http:/doi.org/10.1109/CVPRW.2014.55

[150] R. Szeliski, "Image alignment and stitching: A tutorial," Tech. Rep. MSR-TR-2004-92, October 2004. [Online]. Available: https://www.microsoft.com/en-us/research/publication/image-alignment-and-stitching-a-tutorial/

[151] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition," *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, p. I–647–I–655, 2014. [Online]. Available: https://arxiv.org/abs/1310.1531

[152] F. Anselmi, J. Z. Leibo, L. Rosasco, J. Mutch, A. Tacchetti, and T. Poggio, "Unsupervised learning of invariant representations," *Theoretical Computer Science*, vol. 633, pp. 112 – 121, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0304397515005587

[153] G. Li and Y. Yu, "Visual saliency based on multiscale deep features," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5455–5463, June 2015. [Online]. Available: http:/doi.org/10.1109/CVPR.2015.7299184

[154] F. Perazzi, P. Krähenbühl, Y. Pritch, and A. Hornung, "Saliency filters: Contrast based filtering for salient region detection," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 733–740, June 2012. [Online]. Available: http:/doi.org/10.1109/CVPR.2012.6247743

[155] A. Borji, M.-M. Cheng, H. Jiang, and J. Li, "Salient object detection: A benchmark," *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 5706–5722, Dec 2015. [Online]. Available: http:/doi.org/10.1109/TIP.2015.2487833

[156] C. C. F. Martin, David R. and J. Malik, "Learning to detect natural image boundaries using local brightness, color, and texture cues," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 5, pp. 530–549, May 2004. [Online]. Available: http:/doi.org/10.1109/TPAMI.2004.1273918

[157] 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," *NIPS-W*, 2017.

VITA

VITA

Sri Kalyan Yarlgadda was born in Andhra Pradesh, India in 1993. In 2015, he received his bachelor of technology in Electrical Engineering from Indian Institute of Technology Madras (IITM). Dr Yarlagadda joined the Ph.D. program at the school of Electrical and Computer Engineering at Purdue University, West Lafayette, Indiana under the supervision of Prof Fengqing Maggie Zhu. During his Ph.D he primarily worked on projects sponsored by National Science Foundation(NSF). Air Force Research Laboratory (AFRL), the Defense Advanced Research Projects Agency (DARPA), and the National Geospatial Intelligence Agency (NGA). During his Ph.D. he also gained experience at Google Inc. His current research interests are video/image processing, medical image analysis and deep learning. He is a student member of the IEEE and the IEEE Signal Processing Society.