Object-Level Image Segmentation with Prior Information

Von der Fakultät Informatik, Elektrotechnik und Informationstechnik
der Universität Stuttgart
zur Erlangung der Würde eines Doktor-Ingenieurs (Dr.-Ing.)
genehmigte Abhandlung

vorgelegt von
Chunlai Wang
aus Yibin, China

Hauptberichter:	Prof. Dr.-Ing. Bin Yang
Mitberichter:	Prof. Dr.-Ing. Eckehard Steinbach
Tag der mündlichen Prüfung:	05.07.2019

Institut für Signalverarbeitung und Systemtheorie
der Universität Stuttgart

2019

Acknowledgements

This dissertation would not have been possible without the help and support from many people.

First of all, I am truly indebted and thankful to Prof. Dr.-Ing Bin Yang for providing me the opportunity to work at his institute, for his patient help and smart guidance. I leaned a lot from him, not only about the technical insights and expertises, but also the way of scientific thinking.

I would like to express my deep gratitude to Prof. Dr.-Ing Eckehard Steinbach, who owns the Chair of Media Technology, TU Munich. He gave me the chance to write the master thesis within his chair in 2013 and now, accepts to be the co-referee and review my dissertation.

I sincerely acknowledge Prof. Dr.-Ing. Michael Weyrich and Prof. Dr.-Ing. Jörg Roth-Stielow (president) in my doctoral committee for taking time out of their busy schedules.

I am grateful to the colleges at ISS. It has been my great pleasure to work with them in the past few years. My special thanks go to Lukas Mauch for the co-operation of several papers and for many useful discussions and insightful comments. Many thanks to Fazeh Fallah for her proof-reading. Thanks to Dr. Thomas Kuestner, Michael Ullrich, Dr. Killian Rambach, Dr. Martin Kreissig, Dr. Christoph Zeile, Elisabeth Fellmann, Rene Troeger and Michael Bergmann for their supports and for many interesting conversations and activities.

Last but not the least, I would like to thank my family. My mother and my parents-in-law provided invaluable support during my PhD study. My wife, Dr. Yang Wang and my son, Kasi, their love are the driving force deep inside my heart and have been encouraged me to not give up. There are no words to convey how much I love them.

This thesis is dedicated to the memory of my father.

Ludwigsburg, Jan. 5th, 2019 Chunlai Wang

Contents

Abstract

Although image segmentation as a topic bridging the image data acquisition and visual perception has been evolving for a long time, the challenges ranging from feature representation to model design are still not fully resolved. This thesis deals with the task of segmenting natural images into regions corresponding to the human perception of real world objects. This is called object-level image segmentation. Comparing to traditional image segmentation, object-level image segmentation emphasizes that the segmented regions shall be semantically meaningful and thus, provides image representations which are closer to the final application.

The strategy is to consider context- and application-specific prior information and incorporate them into the development of segmentation algorithms. This is because the definition of object-level is closely related to particular contexts and applications. For instance, outdoor scene and indoor scene images often have different light conditions and contain different object classes. Even two outdoor scenes can be contextually different: urban street scenes are more complex with plentiful traffic subjects while countryside contains more flat areas such as fields or mountains. Moreover, while some tasks (e.g. semantic image segmentation) aim to understand the semantic meaning of the image regions, others (e.g. salient object detection) focus on certain properties of the objects. These application-specific prior informations provide useful regularization to image segmentation.

An overall concept is proposed to design algorithms from the machine learning perspective and with a focus on deep learning model which provides the current state-of-the-art performance. Four research objectives are derived and several methods are proposed based on the concept. (1) Deep convolutional neural networks (CNNs) are proposed for object-level image segmentation with a large training data with ground truth segmentation. The network consists

of a sequence of many convolutional, normalization, activation and pooling layers. Shortcuts can be flexibly added between layers to improve the training performance and segmentation accuracy. Two examples are used for experiments and discussion. (2) A comprehensive study of the context-awareness of the deep CNN model is conducted based on manually designed context-changes. To face changed context in the target application, context-changing data augmentation is proposed to expand the training context and to alleviate the context-sensitivity problem. (3) As the training data requirement in the deep CNN-based method is huge and the ground truth data is expensive, class extension methods are proposed to reuse old models and mitigate the requirement on data. (4) Unsupervised methods based on weak prior informations (such as declarative knowledges and assumptions) are proposed as a complementary part to deep learning and applied to the case without training data.

Several conclusions are drawn from experimental results. (1) Applying the proposed deep CNN to both semantic image segmentation and salient object segmentation achieves good segmentation performance. Supervised learning of deep CNN models shows reliable results when the context is unchanged. The drawbacks of such supervised approaches include the sensitivity to large context changes, limitation to segment new object class other than the pre-defined ones and dependency on large training data with ground truth segmentation, which is very expensive. (2) The comprehensive study on the context-awareness of the deep CNN model for semantic segmentation confirms that the model is sensitive to context changes. This leads to many limitation and potential dangers when the models are deployed in real-life applications. The proposed context-changing data augmentation method is effective for training models which are insensitive to some large context changes but with more training efforts because the training samples are increased. (3) Class extension is an effective method to adapt an old model to handle more object classes. The experiments show that up to 500 training samples are sufficient for the class extension. By fusing another binary segmentation model providing knowledge about the new object classes, the class extension does not even require manual annotation of the new training samples. (4) Incorporating weak prior informations (such as declarative knowledges and assumptions) into hand-crafted features is feasible to achieve good segmentation results without any data for training. The experiments show that the proposed purely unsupervised algorithms, UnOLIS and SGOP-SRR, both achieve state-

of-the-art unsupervised segmentation performance, indicating that object-level information is effective to reduce over-segmentation and refine the segmentation results.

Zusammenfassung

Bildsegmentierung ist ein wichtiges Thema und überbrückt Bilddatenerfassung und Visuelle Wahrnehmung. Obwohl sie hat sich seit langem entwickelt, die Herausforderung von Merkmale Extraktion bis zum Modelldesign sind nicht vollständig gelöst.

Diese Arbeit beschäftigt sich mit der Aufgabe, natürliche Bilder in Bereiche unterzuteilen, die der menschlichen Wahrnehmung von Objekten der realen Welt entsprechen. Dies wird als Bildsegmentierung auf Objektebene bezeichnet. Im Vergleich zur herkömmlichen Bildsegmentierung betont die Bildsegmentierung auf Objektebene, dass die segmentierten Bereiche semantisch sinnvoll sein sollen und liefert somit Bilddarstellungen, die näher an der Endanwendung liegen.

Die Strategie besteht darin, kontext- und anwendungsspezifische Vorinformationen zu berücksichtigen und in die Entwicklung von Segmentierungsalgorithmen zu integrieren. Der Anlass ist, dass die Definition der Objektebene ist eng mit bestimmten Kontexten und Anwendungen verbunden. Zum Beispiel, Szenen vom Freien und Innenaufnahmen haben normalerweise unterschiedliche Lichtbedingungen und enthalten unterschiedliche Objektklassen. Sogar zwei Außenszenen können kontextabhängig sein: Städtische Straßenszenen sind komplexer mit reichhaltigen Verkehrssubjekten und die Landschaft Bilden enthalten flachere Bereiche wie Felder oder Berge. Einige Aufgaben (z. B. semantische Bildsegmentierung) versuchen die semantische Bedeutung der Bildregionen zu verstehen und manchel andere (z. B. Erkennung von auffallenden Objekten) konzentrieren sich auf bestimmte Eigenschaften der Objekte. Diese aufgabenspezifischen Vorinformationen bieten eine nützliche Regularisierung für die Bildsegmentierung.

Es wird ein Gesamtkonzept vorgeschlagen, um Algorithmen aus der Perspektive des maschinellen Lernens zu entwerfen. Einer Fokus ist die tiefe neuronale Netze, was den aktuellen Stand der Technik bieten. Es werden vier For-

schungsziele abgeleitet und mehrere Methoden vorgeschlagen basierend auf dem Konzept . (1) Tiefe konvolutionelle neuronale Netze (CNNs) werden für die Bildsegmentierung auf Objektebene mit großen Trainingsdaten und Ground Truth Segmentierung vorgeschlagen. Das Netzwerk besteht aus einer Reihe von Faltungs-, Normalisierungs-, Aktivierungs- und Pooling-Schichten. Verknüpfungen zwischen den Faltungsschichten können flexibel hinzugefügt werden, um die Trainingsleistung und die Genauigkeit der Segmentierung zu verbessern. Zwei Anwendungsbeispiele werden für Experimente und Diskussionen verwendet. (2) Auf der Grundlage manuell gestalteter Kontextänderungen wird eine umfassende Untersuchung des Kontextbewusstseins des CNN-Modells durchgeführt. Um dem geänderten Kontext in der Zielanwendung zu begegnen, wird es eine kontextverändernde Datenergänzung vorgeschlagen, um den Trainingskontext zu erweitern und das Kontextsensitivitätsproblem zu lindern. (3) Da der Bedarf an Trainingsdaten in der tiefen CNN-basierten Methode groß ist und die Daten mit manuellen Segmentiereung teuer sind, werden Klassenerweiterungsmethoden vorgeschlagen, um alte trainierte Modelle wiederzuverwenden und die Datenanforderungen zu mindern. (4) Unbeaufsichtigte Methoden, die auf schwachen Vorinformationen (z. B. deklarative Kenntnisse und Annahmen) basieren , werden als ergänzender Bestandteil des Tiefenlernens vorgeschlagen und ohne Trainingsdaten auf den Fall angewendet.

Aus den experimentellen Ergebnissen werden mehrere Schlussfolgerungen gezogen. (1) Durch Anwendung des vorgeschlagenen tiefen CNN sowohl auf die Segmentierung von semantischen Bildern als auch auf die Segmentierung auffallender Objekte wird eine gute Segmentierungsleistung erzielt. Das überwachte Lernen von tiefer CNN-Modelle zeigt zuverlässige Ergebnisse, wenn der Kontext der Anwendung unverändert bleibt. Die Nachteile solcher überwachten Ansätze umfassen die Empfindlichkeit gegenüber großen Kontextänderungen, die Beschränkung auf das Segmentieren neuer Objektklassen, die nicht vordefiniert sind, und die Abhängigkeit von großen Trainingsdaten mit einer Hand-segmentierung, die sehr teuer ist. (2) Die umfassende Studie zum Kontextbewusstsein des tiefen CNN-Modells für die semantische Segmentierung bestätigt, dass das Modell für Kontextänderungen empfindlich ist, was zu zahlreichen Einschränkungen und Gefahren führen kann, wenn es auf das praktische Anwendung geht. Das vorgeschlagene Verfahren zur Erweiterung der Datenerweiterung ist geeignet für Trainingsmodelle, die gegenüber einigen

großen Kontextänderungen unempfindlich sind, jedoch mit mehr Trainingsaufwand, da die Trainingsmuster erhöht werden. Die Experimente zeigen den Erfolg des Trainings eines Modells, das gegenüber mehreren großen Kontextänderungen unempfindlich ist. (3) Die Klassenerweiterung ist eine effektive Methode zum Anpassen eines alten Modells an zusätzlichen Objektklassen. Die Eperiments Ergebnisse zeigen, dass bis zu 500 Trainingsmuster für die Klassenerweiterung ausreichend sind. Durch das Verschmelzen eines anderen binären Segmentierungsmodells, welche das Wissen über die neuen Objektklassen bereitstellt, erfordert die Klassenerweiterung nicht einmal eine manuelle Annotation der neuen Trainingsbeispiele. (4) Die Integration von schwacher vorheriger Informationen (wie deklarative Kenntnisse und Annahmen) in handwerkliche Merkmale ist möglich, um gute Segmentierungsergebnisse ohne Daten für das Training zu erzielen. Die Experimente zeigen, dass die vorgeschlagenen rein unüberwachte Algorithmen, UnOLIS und SGOP-SRR, gute Segmentierungsleistung liefern. Das bedeutet, wenn Informationen auf Objektebene wirksam sind, die entwurfte Methoden können die Übersegmentierung reduzieren und die Segmentierungsergebnisse verfeinern.

List of Symbols

Notation

$f(\cdot)$	Function (general purpose)
x	Scalar value
$\underline{x}$	Column vector
X	Integer value
$\mathbf{X}$	Matrix or tensor
$\mathbf{I}$	Identity matrix
$\mathbf{1}$	Matrix or tensor with all elements equal to 1
$\mathcal{X}$	Feature space
$\mathbb{N}, \mathbb{Z}, \mathbb{R}, \mathbb{C}$	Set of natural, integer, real and complex numbers
$x^{(i)}$	Value of x at iteration i in iterative algorithms
$\mathbf{X}^{(l)}$	Matrix or tensor $\mathbf{X}$ related to the layer l in a deep CNN
$\alpha, \beta, \gamma, \lambda$	Scalar parameters

Globally used identifiers

$b(R)$	Boundary connectivity feature value of region R
$conv(R)$	Convexity metric of region R

c_i	Saliency coverage value of a salient object proposal
d_{app}	Euclidean distance using an appearance feature
$d_{geo}(R, R')$	Geodesic distance between regions R and R'
d_{spa}	Euclidean distance based on an the spatial coordinates
$d_t(s_i, s_j)$	chi-square distances of the texture feature histogram between two regions
$d_c(s_i, s_j)$	chi-square distances of the color feature histogram between two regions
$f_{sim}(s_i, s_j)$	Similarity measurement function for superpixel s_i and s_j
$g(\cdot)$	Mapping function of a binary segmentation model
h	$h \in \{0, 1\}$, a hypothesis indicator. $h = 1$ means the hypothesis is true and $h = 0$, otherwise
l	$l \in \{0, 1\}$, label of a superpixel as background or salient foreground object
$\underline{l}$	Label vector for all superpixels of an image
p	$p \in [0, 1]$, likelihood (conditional probability)
$r_{conv}\left(R_i, R_j\right)$	Convexity value changes of merging region R_j to R_i
$r_{prox}\left(R_i, R_j\right)$	Proximity metric of two regions R_i and R_j
$r_{sim}\left(R_i, R_j\right)$	Similarity metric of two regions R_i and R_j
$r_{sym}\left(R_i, R_j\right)$	Symmetry metric of two regions R_i and R_j
$s(R)$	Saliency feature value of region R
v_i	$v_i \in V$, vertex of a graph (mathematically)
w_c	Weighting parameter for color term
w_t	Weighting parameter for texture term
$w(R)$	Weighting parameter w.r.t. region R
C	$C \in \mathbb{Z}$ and $C > 0$, output channels of OLIS
C_{FG}	Foreground region set
C_{t0}	Source context
C_t	Target context
D	$D \in \mathbb{Z}$, number of new training data for class extension
E	Edges in a Graph G (mathematically)
$E(\cdot)$	Energy function
G	Connected component (containing connected regions)
$I(\cdot)$	Indicator function

L	$L \in \mathbb{Z}$, number of network layers in a CNN
M	$M \in \mathbb{Z}$, number of images, regions or other elements
N	$N \in \mathbb{Z}$, number of training samples
O	Positive class of the binary segmentation model
$\bar{O}$	Negative class of the binary segmentation model
O_i	Compact subset of regions from the connected component G
R	A segmented region
R_i	i-th region in a set of images regions
$S(x, y)$	Saliency value of the saliency map at pixel (x,y)
S_o	Superpixel seed for salient object
V	Vertexes of a Graph G (mathematically)
$H \times W$	Spatial dimension of an image
$\mathbf{A}$	Activation of a convolutional layer
$\mathbf{B}$	Bias parameters of a convolutional layer
$\mathbf{L}$	Binary mask
$\mathbf{M}_0, \mathbf{M}_1, \mathbf{M}_2, \mathbf{M}_3$	CNN models for Semantic image segmentation
$\mathbf{P}_i$	Binary mask of the i-th proposal from SGOP-SRR
$\mathbf{S}$	Segmentation output, or saliency map
$\mathbf{S}_i$	Saliency map of the i-th proposal from SGOP-SRR
$\bar{\mathbf{S}}_i$	Average saliency map of the i-th proposal from SGOP-SRR
$\mathbf{S}'$	Refined saliency map from SGOP-SRR
$\mathbf{T}$	Segmentation output of an old CNN model
$\mathbf{V}$	Parameters for batch normalization
$\mathbf{W}$	Parameter tensor (a convolutional kernel)
$\mathbf{X}$	Input feature maps (a 3D tensor) to a CNN
$\mathbf{Y}$	Output probability maps (a 3D tensor of a CNN
$\mathcal{A}_k$	Image manipulation technique for context change
$\mathcal{C}$	Sigmoid Cross entropy loss
$\mathcal{F}$	Mapping function for object-level image segmentation
$\mathcal{G}$	Graph (mathematical)
$\mathcal{J}$	Loss function (general purpose)
$\mathcal{S}$	Test dataset
$\mathcal{S}_b$	Superpixel seed for background (a set of superpixels)

$\mathcal{T}$	Test dataset of target context
$\mathcal{X}$	Image space
$\mathcal{Y}$	Segmentation space
$\mathcal{D}$	Training dataset
α	Indicator for short cut connection
β	Weighting factor in a loss/energy function
γ	Parameter used in the F-Measure metric
ϵ	Parameter in unary energy term
λ	Weight decay parameter
$\sigma(\cdot)$	Normalization function
σ_{app}	Standard deviation parameter for color feature distance
σ_{geo}	Standard deviation parameter for geodesic distance
σ_i^2	Variance value of the saliency map for the i-th proposal
σ_{spa}	Standard deviation parameter for spatial distance
σ_{sym}	Standard deviation parameter for the symmetry feature
θ	Threshold parameter
η	Set of pairs of adjacent superpixels
τ	Temperature parameter for knowledge distillation
μ_i	Mean value of the saliency map for the i-th proposal
$\phi(\cdot)$	Activation function of neural networks
$\varphi(R_s, R)$	Strength of preventing merging region R to R_s
∂R_i	Boundary of region R_i
$\Phi(s_i, s_j, l_i, l_j)$	Pairwise energy term in a graph cut problem
$\Upsilon(s_i, l_i)$	Unary energy term of assigning a label to a superpixel
Ψ	Mapping function of an old CNN model
$\boldsymbol{\Theta}$	Vector contains all parameters of a CNN

Mathematical operations

$\|x\|$	Absolute value of a scalar x
$\|\mathcal{X}\|$	Number of elements in a set $\mathcal{X}$

$\left\|\underline{x}\right\|_2 = \sqrt{\sum_i \|x_i\|^2}$	ℓ_2 norm of vector $\underline{x}$
$(\mathbf{W} * \mathbf{X})$	Convolution of tensor $\mathbf{X}$ with tensor $\mathbf{W}$
$(\mathbf{X} \circ \mathbf{Y})$	Component-wise multiplication of tensor $\mathbf{X}$ and tensor $\mathbf{Y}$ which have the same dimension

Abbreviations

2D	Two-dimensional
3D	Three-dimensional
4D	Four-dimensional
BG	Background
BMACE	Binary Model Assisted Class Extension
CNN	Convolutional Neural Network
Cxt.	Context
DCNN	Deep Convolutional Neural Network
DF	Decision Fusion
ELU	Exponential Linear Unit
FCN	Fully Convolutional Neural Network
FG	Foreground
FN	False Negative
FP	False Positive
GAN	Generative Adversary Network
GbS	Graph-based Image Segmentation
GPU	Graphic Processing Unit
GT	Ground Truth
HTF	Hard Targets Fusion
IoU	Intersection-over-Union

LF	Loss Fusion
MAC	Mean Absolute Error
MIoU	Mean Intersection over Union value
MS	Mean Shift
ODS	Optimal Dataset Scale
OIS	Optimal Image Scale
OLIS	Object-Level Image Segmentation
POP	Perceptual Organization Principle
ReLU	Rectified Linear Units
RGB	Red, Green, Blue (A color space)
RI	Rand Index
SGD	Stochastic Gradient Descent
SGOP	Saliency-Guided Object Proposal
SLIC	Simple Linear Iterative Cluster
SRR	Salient Region Refinement
STF	Soft Targets Fusion
TN	True negative
TP	True positive
VI	Variational Information

Chapter 1.

Introduction

The human visual system can quickly recognize objects in a scene and localize the object boundaries. A comprehensive understanding of the scene allows humans to reason and to plan actions. Computer scientists and electrical engineers, together with neuroscientists have been pushing forward machines and computer systems to have the similar capability. The critical techniques essentially cover the area of computer vision and machine learning.

This thesis deals with a fundamental task in computer vision, i.e. the image segmentation, and focuses on the utilization of advanced machine learning techniques to solve the problem.

The goal of image segmentation is to partition an image into non-overlapping regions corresponding to objects contained in an image. This is widely regarded as the first step of image analysis and scene understanding and has shown promising usefulness to many applications such as

- video surveillance [8, 133, 123, 58],
- self-driving vehicles [35, 61, 17],
- smart homes [89, 129],
- robot navigation [96, 42, 77],
- content-based image retrieval from big vision data [117],
- medical image analysis for computer-aided diagnosis [137],

just to name a few.

1.1. Background

Traditional image segmentation methods can be categorized into two branches. One branch of works [154, 34, 55, 1], from the machine learning perspective, try to group pixels. Image segmentation is, from their points of view, an application of cluster analysis. In this case, low-level and local features such as the intensity values and the geometric information of pixels are used. These methods tend to produce over-segmentation, where an object is often divided into several regions. The other branch of works [172, 23, 20, 130], from the traditional computer vision perspective, treat image segmentation as a task of finding continuous surfaces (corresponding to the segmentation regions) in a limited space. They use variational and level-set techniques and optimize a certain global criterion such as the intra-region consistency and inter-region boundary lengths. These works attempt to produce regions corresponding to objects. However, their models are often hard to optimize. The energy functions are normally non-convex and sensitive to initialization [16, 23, 130]. Several assumptions and convex-relaxations are required to achieve a good solution [22, 15, 172, 138].

In recent years, another type of work [97, 164, 104, 156, 147, 69, 57, 103] treats image segmentation as a process of assigning a class label to every pixel in an image such that pixels of the same class share certain object characteristics. They use pixel- or region-based graphical models such as the Markov Random Field (MRF) or the Conditional Random Field (CRF). In these works, images were generally pre-segmented into several small regions using low-level features and then mid-level region features are extracted to calculate the energy functions for the graphical models. The segmentation effectiveness is limited to pre-defined object classes. As the modeling is complex and the learning process is not efficient using such methods, several works [99, 151, 100] researched to improve the efficiency of learning and inference. These supervised methods can learn high-level global features for image segmentation, which benefit the segmentation of perceptually meaningful objects. However, the performance is still limited.

Machine learning methods have become ubiquitous in our attempt to induce more abstract representations of visual scenes and support decisions that depend on it. A breakthrough was made by recent advances in deep learning techniques. In particular, the deep convolutional neural networks ([101, 157, 161, 79]) provide

good paralleling capability for the simultaneous processing of pixels. They allow the automatic learning of hierarchical features of an image from low-level to high-level. This is useful and effective for image classification and analysis. While the convolutional neural network goes deeper, low-level features compose to higher-level features. The high-level features contain global information and are critical for image classification.

The early works [53, 38] which apply deep neural networks for segmentation used sliding windows and patch-wise classification. The final pixel-wise decision is then made by weighting the patch-wise decision. As an image usually contains a huge number of pixels, making decisions pixel by pixel in a sequential manner is computationally expensive and time consuming. The fully convolutional neural network [118] takes an image as input and outputs the probability maps of each pixel belonging to pre-defined object classes. Several works [9, 132] then use encoder-decoder structure following the fully convolutional construction for semantic image segmentation. The idea is that a set of encoding layers project the input image to a high-level representation, capturing all important spatial dependencies needed for segmentation. A set of decoding layers are then applied to reconstruct a segmentation map from this high-level representation. The whole network is trained end-to-end using images with pixel-wise class labels in a supervised way. These methods achieve the state-of-the-art image segmentation results.

1.2. Motivation and objectives

Image segmentation is treated as an ill-posed problem in the classic computer vision community [114, 40, 163]. The correct segmentation is often indeterminate due to inconsistent understanding of objects. For instance, segmentation of a car as an entire object in an image has many application cases but the segmentation of only the tires of the car can also be correct because the tires can be treated as individual objects. Traditional image segmentation methods usually use a bottom-up process and use low-level features. They often tend to over-segmentation due to lack of high-level and global information which provides object-related knowledge.

The data-driven and learning-based segmentation methods have a more clear definition of the segmentation objective. This is because the training data contains task-specific information. For instance, the semantic image segmentation tasks in [17, 35] clearly describe the the context of the application. They define the semantic classes and provide ground truth segmentation of images. Other databases [6, 3] do not specify the semantic object classes, but also provide the subjective segmentation as reference to the correct segmentation. In another example, the salient object segmentation tasks [2, 112, 4, 155] focus on the objects which are dominant in an image and salient to the human eye. This takes the attention mechanism of the human visual system into consideration to define the segmentation goal.

In summary, the segmentation tasks are significantly related to the given information related to the definition of objects. Currently, there are many context-specific and well-defined image segmentation tasks for different applications [35, 17, 112, 155]. As the final application is different, the required image feature representation can be different. But the methods to achieve the feature representation are similar. However, there is no unified work which encapsulates the overall segmentation tasks under different conditions.

Goal 1 In this thesis, we aim to fill this gap and propose a general concept to solve the object-level image segmentation tasks.

Moreover, while the deep learning techniques are emerging [101, 157, 161, 79], the preliminary works mostly focus on the classification tasks [41, 145, 196]. Image segmentation, as a pixel-wise classification task, is more challenging and requires a different deep neural network structure. A fully convolutional neural network [118] provides a rational method for end-to-end deep learning based image segmentation while avoiding the explosive increasing of parameters when adapting classification to segmentation (imagine a segmentation is related to millions times of classification). As the networks go deeper, the gradient based back-propagation can hardly be used to train the parameters due to the vanished gradient [66]. Several works [33, 85, 79] were proposed and claimed to alleviate the problem. In particular, the residual network [79] achieved superior performance for the image classification task [145]. However, the effectiveness of migrating this technique from image classification to image segmentation

based on fully convolutional networks needs to be examined. As the end-to-end learning makes decisions for the pixels of an image simultaneously, it is in fact a multi-class joint labeling task. The impacts of this multi-class joint learning, especially the impacts on the contextual aspects, are less well-known.

Goal 2 For the reason stated above, a further goal of this thesis is to study the use of cutting edge deep learning techniques (including the design of a deep learning architecture, such as the use of residual shortcut etc.) in object-level image segmentation and study their effects regarding context-awareness and context-sensitivity.

Further, as the field of computer vision has undergone a major shift with the dominance of neural networks with a remarkable success for various problems, the vast majority of the methods employed use fully supervised learning utilizing a massive amount of curated training data from a specified context and for a specified application. This results in two problems, especially for the deep learning techniques. First, models trained by supervised learning can be only applied to a limited context (same or similar to the training data). We will never have enough annotated datasets to train models which are robust to all real world contexts. Several approaches, such as unsupervised learning [139, 184], self-supervised learning [56, 43], learning to learn [5] have been investigated. Despite a notable progress, they still make an insignificant fraction of the research in the vision community and also suffer from a considerable performance gap with supervised methods. Second, annotating data to generate necessary supervision information for pixel-wise classification tasks (such as image segmentation) is difficult and costly. For one reason, high-resolution images nowadays often contain millions of pixels. For another reason, tasks such as image segmentation are ill-defined and the annotation is subjective but the outcome of the segmentation algorithms building on it needs to be determinate.

Goal 3 This leads to the third goal of this thesis to exploit transfer learning and class extension methods, to reuse existing sources and alleviate the data requirements while fulfilling the current performance requirement.

Goal 4 The fourth goal is to exploit unsupervised computational methods for the segmentation to completely cast off the data-driven training process.

1.3. Contribution

The main contribution of this thesis contains the following aspects.

First, we propose a concept of object-level image segmentation with prior information to encapsulate many specialized image segmentation tasks and to bridge the gap between general image segmentation and the specialized image segmentation. In the concept, the context and the application related information are denoted as prior information and incorporated into the image segmentation tasks to produce critical object-level representation. The conceptual scheme provides a guideline to deal with different (object-level) image segmentation tasks.

Second, several methods are proposed under the conceptual scheme to solve particular image segmentation tasks and to illustrate the utilization of varied prior information:

- Supervised deep learning techniques for classification tasks are adapted to solve image segmentation tasks. In particular, a fully convolutional neural network (CNN) with encoder-decoder structure and with flexible integration of skip connections is proposed. The proposed CNN for object-level image segmentation applies conventional supervised learning and uses training data with corresponding ground truth segmentation as prior information. Two application cases are used as examples to show that the proposed method is effective for object-level image segmentation tasks.

- The strengths and the drawbacks, especially the context-awareness and context-sensitivity of the deep CNN models, are summarized through illustrative test and case studies. In particular, we conduct several comprehensive context-awareness tests by attentively change the context of test images and observe the effect on the outcomes of the CNN models.

- We propose context-changing data augmentation to alleviate the context-sensitivity drawback in object-level image segmentation. We experimentally show that the method is able to train models insensitive to several large context changes with corresponding context transformation techniques known in advance.

- Using class extension in semantic image segmentation as an example, the idea of reusing models is implemented and evaluated in this thesis. Transferring knowledge from another model to the target model by integrating them into extended learning processes is shown to be effective. In particular, a binary segmentation model containing knowledge of a new object class is fused into a CNN model for semantic segmentation of more object classes.

- Two unsupervised methods are proposed for class-unknown object-level image segmentation and salient object segmentation, respectively. In particular, this thesis shows the incorporation of high-level declarative prior knowledge by hand-crafting features. Better segmentation results are achieved by the proposed methods comparing to other unsupervised methods.

1.4. Organization

The rest of the thesis is organized as follows:

Chapter 2 provides a detailed description of the object-level image segmentation task for in depth understanding of the notations and terminologies used in later chapters. Meanwhile, it gives an overview of the main thread to the whole work and provides a guideline to the following chapters. The overall concept is illustrated in this chapter.

Under the conceptual scheme, Chapter 3 first examines the use of deep CNN for the object-level image segmentation with an image dataset containing ground truth segmentation as prior information. A deep encoder-decoder CNN architecture with flexible insertion of shortcuts is proposed in this chapter. The method

is applied to two object-level image segmentation tasks: semantic image segmentation and salient object segmentation. Deep CNN models are trained and several experiments are conducted to show the strengths and drawbacks of the methods.

With a focus on the first problem of context-sensitivity mentioned in Chapter 3, Chapter 4 first conducts a comprehensive case study on the context-awareness and context-sensitivity. Then, a context-changing data augmentation method is proposed to alleviate the context-sensitivity drawback. Experiments with three large context changes show the effectiveness of the proposed method.

In Chapter 5, pre-trained (old) models are treated as given prior information to reduce newly manual annotation. Methods are proposed for reusing models and transferring their knowledge to train the new model for the target image segmentation.

Chapter 6 focuses on unsupervised methods for object-level image segmentation. Two methods are proposed for two different object-level image segmentation tasks, respectively. The object related prior knowledges are used as weak prior informations and are integrated into the segmentation by designing high-quality hand-crafted features.

Chapter 7 gives a summary and concludes the whole thesis. Moreover, it gives an outlook to the future works based on the limitations of this thesis.

Chapter 2.

Definition and Concept

This chapter defines the scope of object-level image segmentation and introduces the basic idea of incorporating prior information for object-level image segmentation.

In Section 2.1, we describe the related definition of object-level image segmentation in details. We define the context referred to object-level image segmentation, discuss the typical use cases covered by object level image segmentation and introduce the prior information referred in this thesis. Section 2.1.3 discusses the implication of object-level image segmentation. Then, in Section 2.3, we introduce the methodology used to solve the object-level image segmentation, i.e. by means of multi-level feature extraction and incorporation of prior information. Section 2.4 gives an overview of the proposed methods described in the later chapters and provides a guideline to the rest of the chapters.

2.1. Object-level image segmentation

Given a digital image of a natural scene, the general image segmentation task aims to partition it into a set of non-overlapping regions whose union is the entire image [72]. In this thesis, we aim at image segmentation on the "object-level" and expect that the segmented regions correspond to meaningful objects. However, the definition of "object" is subjective and ambiguous. An "object" can refer to a "thing" (e.g. a dog, a car), a "stuff" (e.g. a building, a forest), or even a kind of texture (e.g. wood, rock). Lacking a clear definition of the "object" makes the general image segmentation a challenging and ill-posed problem.

In this thesis, we use "object-level" to emphasize that things and stuff with semantic meanings are important and a kind of texture is just a property of an object and less semantically useful within the scope of this thesis.

2.1.1. Definition

Object-level image segmentation (OLIS) is a process of grouping (clustering or classification) pixels of an image into $C \geq 2$ regions (or classes) where each region corresponds to a single or a set of semantically meaningful objects of the same kind. It is a pixel-wise classification task in the sense that each pixel must be assigned to one region. A region is not necessarily connected. It may contain, e.g. all cars in the image which are not connected.

Input Let $\mathbf{X} \in \mathcal{X} = \mathbb{R}^{H \times W \times 3}$ be an image to be segmented. It is called the input to object-level image segmentation. It is a color image with three channels. Each channel represents a component of a pre-defined color space (e.g. RGB, CIELab and YCbCr). The first two dimensions are the spatial dimensions, i.e., the image has a size of $H \times W$. Unless otherwise stated, the input images in this thesis are given in RGB color space.

An image processed by object-level image segmentation processes is taken from a certain context.

A **context** is a general designation of a scene, the objects that exist in the scene and the things that happen in the scene. These objects and things can help to explain the scene. For instance, a street scene may contain vehicles on the road (drivable surface) and pedestrians walking on the sidewalks. These objects (vehicles, pedestrians, road and sidewalks) and the things that happen (driving and walking) can be used to explain the street scene .

A **context of an image or an image dataset** is the context, from which the images are taken.

Mapping function Object-level image segmentation processes the image using a mapping function $\mathcal{F} : \mathcal{X} \rightarrow \mathcal{Y} = \mathbb{R}^{H \times W \times C}$. $\mathcal{F}$ can be either a computational and non-parametric model or a machine learning model with parameters to be tuned by a training algorithm. Moreover, $\mathcal{F}$ is not restricted to a single feed-forward computation, rather, it can be a complex combination of multiple functions with respect to the input image.

Output Let $\mathbf{Y} \in \mathcal{Y}$ be the segmentation result of $\mathbf{X}$. It is called the output. $\mathcal{Y} = \mathbb{R}^{H \times W \times C}$ is the target space or segmentation space. Again, the first two dimensions are the spatial dimensions. The third dimension indicates the segmentation channels. Note that $\mathbf{Y}$ has the same spatial size of $\mathbf{X}$. C is the number of segmented regions and is also the number of channels in $\mathcal{Y}$. Each channel indicates one segmented region. For $j \in \{1, 2, \cdot, C\}$, the j-th segmented region shall correspond to a single or a set of semantically meaningful objects.

2.1.2. Typical use cases

The object-level image segmentation defined in Section 2.1.1 is generic and contains many use cases such as semantic image segmentation [35, 17, 118, 103], FG/BG segmentation [142, 37] and salient object segmentation [112, 27]. Fig. 2.1 gives a simple illustration.

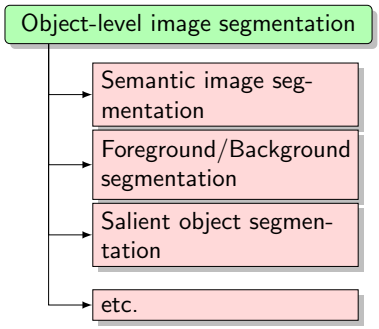


Figure 2.1.: Typical use cases covered by object-level image segmentation.

As one of the main tasks in computer vision, semantic image segmentation aims at partitioning an image into regions that delineate meaningful objects as well as labelling those regions with a predefined object class. Abundant applications are shown in different areas such as content based image search [117], image editing [50], robot navigation [42], medical image analysis [137] and street scene understanding [35].

Saliency detection is an important and challenging problem, aiming to automatically discover and locate the visually interesting regions that are consistent with human perception. It provides enhancement to traditional computer vision, computer graphics and visual communication technologies and covers a wide range of applications such as object recognition [148] and tracking [124], adaptive region-of-interest based image compression [134], content-aware image retrieval, adaptive content delivery and saliency-based image quality assessment [193]. Notice that two types of saliency detection are involved. The eye fixation-based saliency prediction [92] tries to detect salient points by modeling the human eye attention mechanisms and the salient object detection [27] focuses on highlighting the whole object uniformly and accurately. In this thesis, the second type of saliency detection is involved as an object-level image segmentation task. This is because the salient object detection often produces a score map (which is called a saliency map) and the scores are proportional to the probabilities of belonging to the salient object region. Hence, a thresholding process of the saliency map can generate the salient object segmentation.

Notice that the third dimension of the output C (defined in Section 2.1.1) is different between use cases. In many cases, C is known in advance. For instance, the binary segmentation tasks such as foreground/background (FG/BG) segmentation and salient object segmentation (segment the image into salient/non-salient regions) have $C = 2$. In semantic image segmentation, not only is C given, but also the semantic meaning of the C channels are predefined. However, image segmentation tasks with a fixed C and pre-defined object classes restrict the application. For instance, a semantic segmentation model $\mathcal{F}$ for street scene categories/classes (Road, Vehicles, Persons, etc.) cannot be applied to indoor scene segmentation, because the indoor scene contains undefined object classes like cats, chairs, desks, etc.

Although the final segmentation results always have a determined C, we allow it to be unknown before segmentation. Object-level image segmentation with unknown C is more challenging. In the case that the object classes are not specified, unsupervised methods [45, 178] are designed for segmentation in this case. Such an unsupervised object-level image segmentation can be used to discover objects in an image collection [146, 166] or to provide candidate regions for object recognition [126, 135]. Comparing to the bounding-box (e.g. sliding window) based object recognition [54, 169, 64, 198, 62], unsupervised object-level image segmentation is beneficial especially for objects that cannot be well approximated by a rectangle.

The object-level image segmentation is not applicable if the image covers only a single object, i.e. C is equal to 1. In this case, no segmentation is needed. This is not considered in this thesis.

2.1.3. Implication

Image segmentation produces simplified image representation beyond pixels. The significance of the object-level image segmentation is reflected in two ways. First, it emphasizes that the image segmentation representation shall be at the "object-level", being close to the human perception of objects in an image. Second, object-level image segmentation provides a generalized formulation for many image segmentation tasks. This reduces the gap between the general image segmentation and many task-specific image segmentation and provides a unified framework to select segmentation methods for real applications.

Superpixel and over-segmentation

A category of bottom-up segmentation methods uses low-level features (e.g. gray-scale intensity, color histogram, edges etc.) for image segmentation. The algorithms are usually unsupervised, such as clustering algorithms K-Means [1], Spectral clustering [154] and Density-based clustering [34, 152]. They often tend to segment images into small homogeneous patches and thus, most objects are over-segmented. We call them low-level segmentation or superpixel segmentation.

Figure 2.2.: Superpixel segmentation of an image from Cityscapes dataset [35] using the method proposed by Achanta et al. [1]. The algorithm applies low-level features such as color intensities and geometric positions to group local pixels.

Superpixel segmentation is essentially different from object-level segmentation in the sense that a superpixel is a small region of an image and hardly covers a complete object. The problem here is over-segmentation.

Fig. 2.2 shows an example of superpixel segmentation result, which contains a large number of superpixels. In comparison, object-level image segmentation aims at dividing the image regions into corresponding to objects aligned with the human perception. Table 2.1 gives a more detailed comparison between superpixel and object-level image segmentation. For superpixel segmentation, local and low-level features such as color, brightness of pixels and the edges are used. For object-level image segmentation, we need higher-level features (such as object shape) except the low-level features.

	Regions correspond to objects	Number of segmented Regions	Information used for segmentation
Superpixel	not necessarily	large	local, low-level features (e.g. edges)
Object-Level	yes	small	global, high-level features (e.g. object shape)

Table 2.1.: Object-level image segmentation vs. superpixel segmentation.

Although superpixel is an intermediate image representation, it is very useful. Grouping superpixels into objects has the advantage of less computational complexity in comparison to directly grouping pixels. Hence, superpixel segmentation methods are widely used as pre-segmentation before further processing. Moreover, superpixel regions are more meaningful and easier to analyse than pix-

els. Region-based features (such as texture, shape features and scene geometry) are more object-related.

Specific segmentation tasks and their gap to general image segmentation

There are many image segmentation tasks which are specified to dedicated applications. For instance, semantic image segmentation is used for street scene analysis to support driver assistance systems or autonomous driving [35, 17]. Salient object segmentation methods are commonly used as a first step of many graphics/vision applications including object tracking [124], content-aware image editing [30, 191] and retrieval [26, 83, 60]. Object instance segmentation [39, 111] accurately segments each object instance which is used for indoor robotic applications. Fig. 2.3 shows an example of these object-level image segmentation tasks.

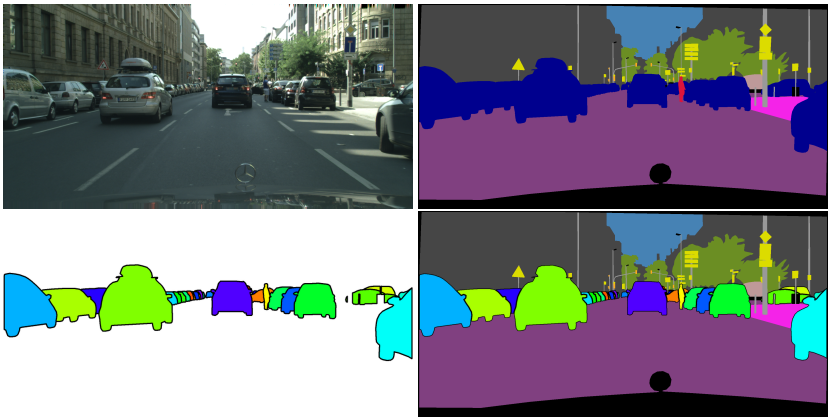


Figure 2.3.: Object image segmentation examples of an image (top left) from Cityscapes dataset [35] (Top right: semantic image segmentation. Bottom left: foreground objects (vehicles) segmentation. Bottom right: instance-aware semantic image segmentation.)

Although the applications are different, the current state-of-the-art methods to achieve the feature representation used for segmentation are similar. More specifically, the feature is either learned through feature learning technique or

manually crafted. However, there is no unified framework which encapsulates all segmentation tasks and underlines the context and the application related information.

The object-level image segmentation bridges this gap. It is generally defined and covers a bunch of segmentation tasks underlying object-level image representation outcomes. Given according prior information (introduced in the next section) specified by contexts and applications, the object-level image segmentation is achieved by incorporating the prior information in a supervised way (via a learning process) or a unsupervised way (via hand-design of features). Hence, object-level image segmentation is a more general segmentation framework.

2.2. Prior information

Object-level image segmentation restricts the image segmentation task at the object-level. However, the boundaries between object-level and other levels are blurry. We propose to address the ambiguity of "object-level" by using prior information.

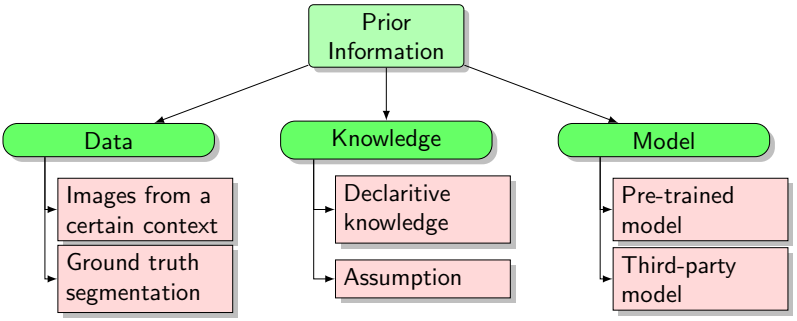


Figure 2.4.: Prior information in this thesis.

In general, prior information is the information given in advance and with any form of data regarding the context and the application. It is combined with experience and intuition. In particular, for object-level image segmentation in

this thesis, the prior information encapsulates all information which is applicable to build the segmentation model. This includes data, knowledge and models (as shown in Fig. 2.4).

- Data: It indicates a set of images with or without the ground truth segmentation. In the ideal case, an image dataset is from the same or a similar context as the target context for a certain application. However, it can also contain certain context changes with respect to the target context.

  Given an image to be segmented, a collection of images from the same context can help the object-level image segmentation task. In the machine learning community, the image collection is called a training dataset and can be used for training a segmentation model for the target application. The training images in this collection can be manually segmented and each segmented region is labeled with an object class. This ground truth segmentation is also treated as prior information.

  Although manual segmentation is helpful, it is in fact very expensive for the object-level image segmentation task. In practice, we could not expect that the training dataset from the same context contains manual segmentation. However, if there is an available training dataset from another context, with manual segmentation, it can also be used as prior information for the target segmentation task if the difference between two contexts are traceable. In particular, the context can be adapted to the target application during training.

- Knowledge: In general, knowledge is our personal map/understanding of the real-world objects. It contains our beliefs and expectations based on experience. This thesis refers to declarative knowledge of a given dataset or about the context and the application. This knowledge can be summarized from observations and noted as assumptions. They provide the basis for the design of meaningful hand-crafted features.

  For instance, the declarative (common) knowledge such as *"the dominant object in a picture is often salient and hardly connected to the image boundaries"* is based on the experience that we humans usually take pictures of an object by adjusting the camera to focus on it. This knowledge is useful for the task of a foreground/background segmentation of an image. The human

visual system has the ability to derive relevant groups and structures from an image without knowing the objects in the image. The underlying theory is known as perceptual organization principles. These principles (cf. Chapter 6.1 for more details) can be used as assumptions to group pixels or superpixels into objects when the object classes and even the number of objects are unknown.

- Model: It indicates existing statistical models learned for certain tasks. The models can be a previous well-trained old model (pre-trained model) which can be reused, or a third-party model which can be obtained from the internet. They can not be directly applied to the target task, because the output segmentation spaces are different (e.g. because the object classes are different). However, since pre-trained models often contain information of specific object classes, they can be used to design new segmentation algorithms. A CNN model trained using machine learning algorithms is a typical example.

2.3. Methodology

In this section, we introduce our general strategy of object-level image segmentation with prior information. In particular, we discuss the multi-level feature extraction methods and the methods to incorporate the prior information.

2.3.1. Overview

Fig. 2.5 gives an overview of our concept. We deal with object-level image segmentation from the machine learning perspective. The methods are demonstrated in three clusters, i.e. supervised learning, transfer learning and unsupervised learning. Within each cluster, the underlying prior information is summarized and the corresponding approaches used to incorporate the prior information are listed.

The vertical axis shows the abstraction level of the prior information. Given a training dataset of images containing their corresponding ground truth segmentation, the object classes are clearly defined. In this case, the pixel-wise annotation

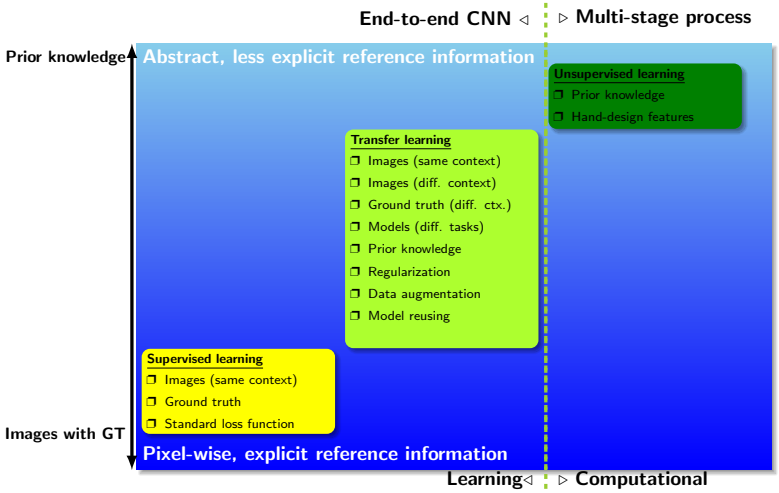


Figure 2.5.: The concept of object-level image segmentation with respect to different prior information

provides explicit reference information. On the contrary, the prior information in form of declarative prior knowledge provides abstract restriction to the objective segmentation task. In this case, the prior information is at the image level, indirect, implicit and hard to extract. In-between are the cases where there are training image data, but no corresponding ground truth segmentation or they are not directly available.

The cluster of methods varies over the horizontal direction and from supervised to unsupervised case. We highlight the powerfulness of end-to-end learning (all parameters are trained jointly), especially, with respect to the recent advances of deep learning. Given a training dataset, the deep convolutional neural network (CNN) is a powerful statistical model to automatically learn features from data and to achieve high performance segmentation. We apply CNN in both supervised learning and transfer learning clusters. In the unsupervised cluster, where no training data is available, we design multi-stage computational process.

In the next subsections, we first discuss the special requirements on the feature extractions used for object-level image segmentation. Then, we discuss the methods to incorporate the prior information for object-level image segmentation.

2.3.2. Multi-level feature extraction

Object-level image segmentation requires features from low-level to high-level. The low-level features are useful because they capture the local details of an image area and help to find the accurate boundaries. Higher-level features provide more global information. They are more semantics-related and are necessary to restrict the segmentation to be at object-level.

Two schemes are feasible to obtain such useful image feature representation for object-level image segmentation. First, the deep CNN is capable of learning such hierarchical features with a training dataset given as prior information. Second, multi-level features can be hand-designed within a multiple processing stage. The final object-level segmentation results are then achieved by clustering or classification using the features.

Automatic feature learning with deep CNN

Deep CNN has been used to solve many computer vision tasks [101, 157, 161, 79, 118, 145, 196, 35] and achieve remarkable results. Moreover, a CNN model trained with a large number of image samples for one task (e.g. image classification [101, 157, 161, 79]) can also be used as a feature extractor for another task (e.g. object detection and segmentation [195, 76, 75] and visual tracking [122]). A general understanding of the features learned in a CNN is that the first layers of CNN models learn low-level features such as edges, corners, textures etc., and they are combined in the later layers to learn more semantically meaningful and object-level features.

Such an automatic multi-level feature learning has many strengths for the object-level image segmentation task. First, the features are often very effective because the hierarchical network architecture inherits some characteristics from the human vision system and the network parameters are trained with a large number

of image samples. Second, while the network parameters are updated by an optimization algorithm, the features are automatically learned during this training. This omits the challenge of hand-designing features. Due to these advantages, in this thesis, we apply deep CNN for object-level image segmentation as long as there is an image dataset which is usable for training.

The major drawback of using deep learning is its dependence on a large training dataset. The supervised learning strategy requires the training images to be manually annotated. This is rather costly.

Multi-stage process with hand-crafted features

Feature descriptors can be manually crafted. For instance, Canny [18] designed a computational approach for edge detection; Lowe [121] designed a scale-invariant feature descriptor for object recognition; Bay et al.[10] designed a feature descriptor to increase the robustness towards geometric deformations and localization errors. In this thesis, we also manually design computational approaches for object-level image segmentation. This is especially useful in the unsupervised case, where no training data is usable as prior information.

The low-level features (such as colors, brightness, edges, etc.) are easy to extract and can be used straightforwardly. However, the high-level and object-related features are application-specific. In this case, we propose multi-stage processes and use some knowledge to extract the higher-level features. The reason for multi-stage is as follows:

- Segmentation is a pixel-wise grouping task. Knowledge is indirect/abstract and not usable for end-to-end learning.
- Grouping of pixels into objects requires information from different levels. Local information is useful for grouping pixels into small regions (superpixels), while the global information and relatedness between small regions make the construction of objects plausible.
- Instead of making decisions for each pixel, superpixels can be achieved by a pre-segmentation stage and used as the basic unit to be grouped.

- Superpixels provide detailed boundaries which attach the real object contours.

The first stage of the multi-stage is often the pre-segmentation of the image into superpixels. Grouping the superpixels into semantically meaningful object regions relies then on the higher-level features incorporated with semantic object information.

2.3.3. Incorporation of prior information

The prior information needs to be incorporated into algorithms to extract object-related features. They can be used in different ways.

Standard loss function

An image dataset from the same context and the ground truth segmentation can be incorporated using a standard loss function for supervised training. By comparing the segmentation output with the ground truth by defining the loss function, the parameters of a statistical model (e.g. a deep CNN) are tuned using an optimization algorithm. The trained model is then used for the object-level image segmentation. Chapter 3 will introduce this method in details.

Regularization

In addition to the normal pixel-wise reference information, additional prior information can be incorporated to regularize the training procedure. In this case, a regularization term is often combined with a standard loss function, leading to a non-standard loss function for training. For instance, a binary segmentation model (third-party) can be used as the additional information and is incorporated in the loss function as a regularization term (cf. Chapter 5).

Data augmentation

In many cases, the training data contains limited training samples which deliver limited context information of a certain application. Moreover, the context can change and turned to be different from the source context of the training data. In this case, we may enlarge the training data using some image manipulation techniques which are known in advance. The expectation is that such an image manipulation technique is able to map the training data to the changed context. This incorporates the prior information including the original training data and the image manipulation techniques.

Manual feature design with prior knowledge

Declarative prior knowledge from common sense or assumptions can be incorporated by hand-designing feature extractors.

2.4. Thesis guideline

As stated in Section 2.1, object-level image segmentation encapsulates many application-specific segmentation tasks. In the later chapters of this thesis, we will target three application-specific segmentation tasks, i.e. semantic image segmentation, salient object segmentation and class-unknown object-level image segmentation. These tasks have many applications and have been widely researched in recent years. We use them as examples to illustrate the proposed object-level image segmentation methods.

Table 2.2 gives a summary of the proposed methods in the next chapters. They are aligned with the concept described in this chapter.

Chapters 3, 4 and 5 cover the end-to-end learning methods using deep CNNs. DCNN-SemSeg (deep CNN for semantic segmentation) and DCNN-SalSeg (deep CNN for salient object segmentation) are two models proposed in Chapter 3. Context-changing data augmentation and the class-extension methods (HTF and BMACE) are proposed in Chapter 4 and 5, respectively. The multi-stage methods (UnOLIS and SGOP-SRR) are presented in Chapter 6.

Proposed methods		Prior information						
		Prior knowledge	Data (same ctx.)		Data (diff. ctx.)		Models	
			Images	GT	Images	GT	Multi-class (C-1)	Binary
Multi-stage	UnOLIS	√						
	SGOP-SRR	√						
End-to-end CNN	DCNN-SemSeg		√	√				
	DCNN-SalSeg		√	√				
	Context-changing Data Aug.		√		√(1)	√		
	Class extension (Hard targets fusion)		√		√	√(2)	√(3)	
	Class extension (BMACE)		√				√(3)	√(4)

Table 2.2.: The proposed methods for OLIS tasks with different prior information. (1): Images are from another context. (2): The ground truth is incomplete and contains only the class to be extended. (3): An old model outputs $C-1$ maps corresponding to $C-1$ pre-defined object classes. (4): Model outputs binary segmentation maps.

Chapter 3.

Supervised Object-Level Image Segmentation with Deep CNN

This chapter deals with supervised object-level image segmentation, where the prior information contains both a set of training images and their ground truth segmentation from the target context. The object classes are predefined along with the dataset. Supervised learning technique, in particular, the state-of-the-art deep convolutional neural networks (CNNs) are specified for object-level image segmentation.

The contribution of this chapter includes the following aspects: First, an encoder-decoder architecture of convolutional neural networks with flexible insertion of shortcuts is proposed for object-level image segmentation. Second, the proposed method is examined by applications of two object-level image tasks (semantic image segmentation and salient object segmentation). Experiments are conducted to show their segmentation results. Finally, extensive tests are conducted to illustrate the strength of the deep supervised learning and the context-aware effects.

The works of this chapter are partly reported in [175] and [182].

3.1. Introduction

Deep convolutional neural networks (CNNs) have recently shown state-of-the-art performance in many vision tasks such as image classification [101, 157, 161, 79], object detection [63, 64, 140], object tracking [181, 131], object contour

and edge detection [11, 153, 185], optical flow [46, 84], image superresolution [44, 94] as well as semantic image segmentation [9, 118, 132] and salient object segmentation [105, 194, 106]. The main benefit of deep learning techniques is their ability of efficiently learning the spatial dependencies and multi-level features.

As the CNN models for image classification and object detection generally end with two or more fully connected layers, Long et al. [118] first proposed a fully convolutional network (FCN) for semantic image segmentation. They change the fully connected layers to convolutional layers and enable the network to produce pixel-wise prediction. Badrinarayanan et al. [9] and Noh et al. [132] both used an encoder-decoder architecture FCN for semantic image segmentation. The idea is that a set of encoding layers project the input image to a high-level representation, capturing all important spatial dependencies needed for segmentation. A set of decoding layers are then applied to reconstruct a segmentation map from this high-level representation. The whole network was trained end-to-end using images with pixel-wise class labels in a supervised way. These methods achieved the state-of-the-art results in many challenging datasets [36, 51].

The training of very deep networks suffered from the problem of vanishing gradients [66, 79]. Residual learning was proposed by He et al. [79] to effectively avoid vanished gradient. He et al. [79] showed the effectiveness of such residual connection for the image classification tasks as a competitive approach and achieve the best performance in the ImageNet dataset [145].

This chapter proposes a network which introduces the shortcut techniques into the object-level image segmentation with encoder-decoder architecture of convolutional neural networks. The network is then specified for different object-level image segmentation tasks accordingly. In this chapter, we apply the network not only to semantic image segmentation, but also to the salient object segmentation task. Based on the specified task with prior information in forms of training images and pixel-wise ground truth segmentation, the CNN models are trained to achieve good segmentation results.

Although the deep learning showed success in overall image processing areas, the interpretation of its success is mainly traced for image classification tasks: shallow layers learn low-level features, and higher layers learn more abstract features and are useful for image-level decision. Nevertheless, the object-level

image segmentation tasks using CNN are different in the sense that it learns to make decision for all pixels at the same time. The joint learning of multiple classes for labeling the pixels may cause interesting effects, which are worth to study. This chapter analyses the automatically learned features by observing the activation maps of some neurons in the trained networks. A qualitative test is used to verify the context-related effect caused by decision of multiple classes for all pixels at the same time.

In the rest of this chapter, we first present the proposed deep network architecture in Section 3.2. The classic loss functions which incorporate the ground truth into learning are introduced in Section 3.3. Then, we use semantic image segmentation and salient object segmentation as two examples in Section 3.4 and 3.5, respectively, to experimentally show the application of the method. We highlight the strength of end-to-end deep supervised learning and discuss the context-aspects of the trained CNN by quantitative tests in Section 3.6. Finally, we summarize the chapter in Section 3.7.

3.2. Network architecture

This section describes the proposed CNN architecture with an encoder-decoder structure and with flexible insertion of residual shortcuts for object-level image segmentation. Such a network is used to model the mapping function $\mathcal{F}$ (cf. Chapter 2.1), which maps an input image into its segmentation representation.

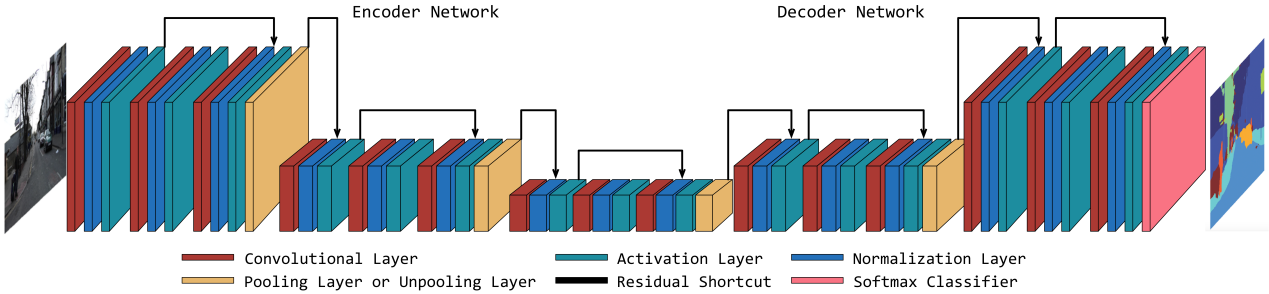


Figure 3.1.: Proposed CNN architecture (Notice this is only an example to illustrate the basic architecture. More layers are used in current DCNN and the number of encoder/decoder layers can be varied.)

Basic architecture

The basic architecture of the network is shown in Fig. 3.1. It consists of a stack of convolutional, normalization, activation, pooling and unpooling layers which are arranged in several encoders and decoders. Since a convolutional layer is always followed by one normalization and one activation layer, they are considered as a compact entity and denoted as a convolutional unit. Although the network given in Fig. 3.1 is symmetric, the numbers of encoders and decoders do not need to be equal and the number of convolutional units contained in one encoder or decoder can also be varying. Moreover, the shortcuts can be added from the shallower convolutional units to deeper convolutional units. Below, we introduce the components of the network more in details.

Let L be the number of convolutional units. The activation and output of the l-th convolutional unit are

$$\mathbf{A}^{(l)} = \mathbf{W}^{(l)} * \mathbf{X}^{(l-1)} + \mathbf{B}^{(l)}, \tag{3.1}$$

$$\mathbf{X}^{(l)} = \phi\left(\sigma(\mathbf{A}^{(l)}) + \sum_{i<l} \alpha_{il} f(\mathbf{X}^{(i)})\right), \tag{3.2}$$

where $\phi(\cdot)$ is an element-wise nonlinear activation function and $\sigma(\cdot)$ denotes the activation normalization. The three-dimensional (3D) tensors $\mathbf{A}^{(l)}, \mathbf{X}^{(l)}, \mathbf{B}^{(l)} \in \mathbb{R}^{H_l \times W_l \times O_l}$ denote the activation, the output and the bias of the convolutional unit, respectively. We call the first two dimensions of the tensors the spatial dimensions and the third dimension the feature dimension. We use $l = 0$ to denote the input layer which does not compute anything. Thus, $\mathbf{X}^{(0)}$ is the input data.

Convolutional layer Eq. (3.1) calculates the activation of a convolutional unit. $\mathbf{W}^{(l)} \in \mathbb{R}^{P_l \times Q_l \times R_l \times O_l}$ is a four-dimensional (4D) tensor containing all convolutional kernels. The first two dimensions are again spatial dimensions, the third dimension is the source feature dimension ($R_l = O_{l-1}$) and the fourth the target feature dimension. The convolution $\mathbf{W} * \mathbf{X}$ of tensor $\mathbf{X}$ with $\mathbf{W}$ is defined as

$$(\mathbf{W} * \mathbf{X})_{hwo} = \sum_{i=1}^{P} \sum_{j=1}^{Q} \sum_{k=1}^{R} \mathbf{W}_{ijko} \mathbf{X}_{h+i-1,w+j-1,k}. \tag{3.3}$$

It is a 3D convolution (or correlation) with a stride of 1 and $\mathbf{X}$ is padded with zeros as needed in the calculation.

Let $H_{l-1} \times W_{l-1}$ and $H_l \times W_l$ be the spatial sizes of the input and output of the l-th convolutional layer. They have the following relationship.

$$H_l = \frac{H_{l-1} - P_l + 2Z_l}{S} + 1, \tag{3.4}$$

$$W_l = \frac{W_{l-1} - Q_l + 2Z_l}{S} + 1. \tag{3.5}$$

Here, $P_l \times Q_l$ is the spatial size of the convolutional kernels. Generally, $Q_l = P_l$ and Q_l (or P_l) is called the kernel size. Z_l is the padding size and S is the stride size. In this thesis, $S = 1$ and Z_l is chosen as $Z_l = \frac{Q_l - 1}{2}$ unless otherwise stated. Consequently, the spatial sizes of the input and output of the convolutional layer are unchanged, i.e. $H_l = H_{l-1}$ and $W_l = W_{l-1}$.

Residual shortcuts He et al. [79] mentioned that deep networks tend to suffer from the vanishing gradient problem. Beyond a certain depth, the network can hardly be trained correctly and adding more layers will cause degradation of the network. Adding residual shortcuts can speed up the convergence and enable efficient training of deep networks. Shortcuts circumvent the vanishing gradient problem, allowing the output of one layer to flow directly to the input of a much deeper layer in the network.

In Eq. (3.2) we use α_{il} to denote a possible shortcut from unit i to unit l. $\alpha_{il} \in \{0, 1\}$ is equal to one if there is a shortcut, and zero if there is no shortcut. If the unit i and l have the same number O of feature maps, $f(\cdot)$ is simply the identity function. If $O_l \neq O_i$, $f(\cdot)$ computes a 1D linear combination over the feature dimension of $\mathbf{X}^{(i)}$

$$f(\mathbf{X}^{(i)})_{hwo} = \sum_{k=1}^{O_i} \mathbf{V}^{(l)}_{hwko} \mathbf{X}^{(i)}_{hwk}, \tag{3.6}$$

with an additional 4D weight tensor $\mathbf{V}^{(l)}$.

Normalization and activation layer $\sigma(\cdot)$ states for normalization. In this work, we choose to use batch normalization proposed by Iofffe et al. [85]. As stated by He et al. [78], such a normalization helps preventing vanishing and exploding

gradients and therefore speeds up convergence. Activation layers introduce the non-linearity to the network. We use Rectifier Linear Units (ReLU) to build our network.

Pooling and unpooling layer We use max-pooling with a window size of 2×2 for the pooling layers to keep the most significant activation and reduce the spatial resolution in favor of a high-level representation of the input image. The unpooling layers rescale the feature maps to the size of the output maps. We simply do upsampling by repeating each value once over both the spatial dimensions. Another possibility to realize the rescaling is to use transposed convolution (deconvolution or fractional-stride convolution). This introduces more parameters and increases the modelling capability of the network. However, this also increases the difficulties and efforts of training. Larger dataset and more computational power are required.

3.3. Supervised learning with standard loss function

In this chapter, the prior information contains the image data and the ground truth segmentation, which provides explicit reference information for each pixel. Therefore, we use the standard loss function as the learning objective to incorporate the prior information.

Let $\mathcal{D} = \left\{\mathbf{X}^{(0)}(n), \mathbf{Y}(n)\right\}_{n=1}^{N}$ denotes the given training dataset of N training samples. We usually build the network to compute a probability map with $O_L = C$ channels for C predefined object classes, respectively.

The ground truth segmentation $\mathbf{Y}(n) \in \mathbb{R}^{H \times W \times C}$ consists of C binary masks for C object classes, respectively.

$$\mathbf{Y}_{hwo} = \begin{cases} 1, & \text{if pixel } (h, w) \text{ belongs to class } o \\ 0, & \text{otherwise} \end{cases} . \tag{3.7}$$

We use a supervised training to determine the parameter vector $\mathbf{\Theta}$. The parameter vector $\mathbf{\Theta}$ contains all elements of $\mathbf{W}^{(l)}$, $\mathbf{V}^{(l)}$ and the bias parameters of the whole

CNN. We choose $\mathbf{\Theta}$ to minimize the loss function:

$$\hat{\mathbf{\Theta}} = \arg\min_{\mathbf{\Theta}} \frac{1}{N} \sum_{n=1}^{N} \mathcal{J}(\mathbf{X}^{(L)}(n), \mathbf{Y}(n); \mathbf{\Theta}) + \lambda ||\underline{w}||_2^2. \tag{3.8}$$

It contains 2 parts: The first term is a data term which penalizes when the network gives incorrect predictions. This term contains the main loss function $\mathcal{J}(\mathbf{X}^{(L)}, \mathbf{Y}; \mathbf{\Theta})$. The categorical cross entropy loss function (also called softmax loss) is widely used for supervised learning and to train neural networks for classification. It is adapted here for object-level image segmentation tasks by summing the loss over all pixels.

$$\mathcal{J}\left(\mathbf{X}^{(L)}, \mathbf{Y}; \mathbf{\Theta}\right) = -\sum_{h=1}^{H} \sum_{w=1}^{W} \sum_{j=1}^{C} \mathbf{Y}_{hwj} \ln \mathbf{X}_{hwj}^{(L)}. \tag{3.9}$$

The second term is a regularization term containing a ℓ_2-norm. Here, $\underline{w}$ is a column vector containing all elements of $\mathbf{W}^{(l)}$ and $\mathbf{V}^{(l)}$. λ is the weight decay parameter governing the regularization. A simple small weight decay (with the order of magnitude 10^{-4}) can improve the generalization [102]. If the value is set too high, the network does not care much about correct predictions on the training set and rather keeps the weights low. In this thesis, we follow many other works [101, 157, 118] and use the weight decay parameter $\lambda = 0.0005$.

This loss function is differentiable w.r.t. the last activation of the network. This is beneficial when applying gradient-based optimization algorithm [144] and back-propagation for the optimization problem formulated in Eq. (3.8).

3.4. Application to semantic image segmentation

In this section, we provide an example of applying the proposed deep learning network for semantic image segmentation of street scenes targeting the application of autonomous driving cars or driver assistance system.

3.4.1. CNN models

We build two deep networks based on the proposed architecture for this task: one base network without shortcuts and one with shortcuts inserted between certain layers. Conventionally, a deeper network has larger capability and would achieve better performance [101, 157, 161, 79]. However, a deeper network contains more parameters than a shallower one (given that the convolutional kernels at the same depth have the same size). Moreover, very deep networks often suffer from problems such as vanishing gradient during training. Hence, deeper networks require more training efforts: they are harder to train and require a larger training dataset.

To achieve a compromise between performance and training efforts, we decided to use a base network with 30 convolutional units. All α in Eq. (3.2) are set to 0 (no shortcuts). The base network contains approximately 44 Million parameters in total. Notice that this is rather few comparing to popular deep neural networks, such as AlexNet [101] (61 Million), VGG19 [158] (175 Million) and FCN [118] (134 Million) etc. Here, we focus on the study of the general effectiveness of inserting shortcuts and use a smaller number of parameters for efficiency. Table B.1 in Appendix B summarizes the detailed specification of the network structure. In the table, we call every three convolutional units a block.

Input The input to the network $\mathbf{X}^{(0)} \in \mathbb{R}^{224\times224\times3}$ are raw RGB images downscaled to the size 224×224 to reduce the computational complexity. The mean-subtraction is applied as preprocessing, where the mean RGB values of the entire training set are subtracted from each image.

Output Given one image, the network computes $O_L = C$ probability maps for C predefined object classes. We use a softmax function in the output layer L to normalize over the feature dimension

$$\mathbf{X}^{(L)}_{hwo} = \phi^{(L)}(\mathbf{A}^{(L)}_{hwo}) = \frac{\exp(\mathbf{A}^{(L)}_{hwo})}{\sum_{j=1}^{C}\exp(\mathbf{A}^{(L)}_{hwj})}. \tag{3.10}$$

The value of $\mathbf{X}^{(L)}_{hwo}$ can be interpreted as the probability of the pixel (h, w) belonging to class o.

For the second network, two types of shortcuts are exploited based on the base network. A rough illustration of these shortcuts is also given in Fig. 3.1. The first type of shortcuts to skip two convolutional units like these used by He et al. [79]. In this case, $\alpha_{il} = 1$, $i = l - 2$ and $f(\cdot)$ in Eq. (3.2) denotes identity mapping. We let l be the index of the last convolutional unit in one block. Thus, the first type of shortcuts performs intra-block bypath, aggregating the output of the first convolutional unit to the last one. Different from He et al. [79], where the network ends up with fully connected layers, we have designed another type of shortcuts for the encoder/decoder architecture. The second type of shortcuts performs inter-block bypath, adding the output of the last convolutional units to the activation of the first unit in the next block. In this case, $\alpha_{il} = 1$, $i = l - 1$ and $f(\cdot)$ denotes the mapping function defined in Eq. (3.6). These shortcuts skip the first convolutional unit in each block. In summary, 17 shortcuts are used for all 30 convolutional units.

Training details

Datasets For the experiments in this section, we used two road scene datasets: The CamVid dataset [17] consists of 701 pixel-wise labeled road scene images. All images were taken under good or medium weather conditions during daytime. We train on this dataset for 11 object classes: SKY, BUILDING, COLUMN POLE, ROAD, SIDEWALK, TREE, SIGN SYMBOL (e.g. traffic light, traffic signs such as speed limit sign etc.), FENCE, VEHICLE, PEDESTRIAN and BICYCLIST. The dataset is divided into a training set of 468 images and a test set of 233 images.

The Cityscape dataset [35] is a large road scene dataset for semantic understanding of urban street scenes. It contains seven scene categories treated as seven object classes in our experiments: ROAD (including road and sidewalk), CONSTRUCTION (including building, wall and fence), SMALL OBJECT (including column pole, traffic sign and traffic light), NATURE (including terrain and vegetation such as tree and grass), SKY, HUMAN and VEHICLE (including car, truck, bus, train/tram, motorcycle and bicycle). One void class is given for pixels not belonging to the seven classes. The images were taken under good or medium daytime weather conditions. The number of published images with pixel-wise

labeling is in total 3475. 2975 of them are used as training set while the other 500 images are used as test set.

Models We train both networks without and with shortcuts on the two training datasets separately. We use cross entropy loss with a ℓ_2-norm regularization as described in Eq. (3.8) and (3.9). The regularization parameter λ in (3.8) is set to 0.0005 in this example. All parameters in Θ are initialized according to the method by He et al. [78], namely, as zero biases and zero-mean Gaussian distributed weights with the standard deviation of $\sqrt{\frac{2}{M}}$, where M is the number of input elements to the current layer. More specifically, $M = H_l \times W_l \times O_l$ for the l-th layer. The learning rate is set to 0.01. The momentum value is 0.9. We use batch-wise training by stochastic gradient descent (SGD) with Nestrov momentum. One batch contains 5 images. All models are trained with more than 50 epochs to ensure convergence. Each training is performed on a single Nvidia Titan X GPU. The implementation is based on the deep learning library Keras [32] and Theano [165].

3.4.2. Experimental results

Evaluation metrics

We evaluate the methods by observing the segmentation results on the test dataset. The metrics are: pixel-wise accuracy for each class, global accuracy, mean accuracy, and mean IoU as used by Long et al. [118], Badrinarayanan et al. [9], and Noh et al. [132]. In particular, let n_{ij} be the number of pixels of class i predicted to class j. C is the total number of classes and $m_i = \sum_j n_{ij}$ is the total number of pixels of class i. Then the class accuracy is defined as n_{ii}/m_i and the global accuracy is $\sum_i n_{ii} / \sum_i m_i$. The mean accuracy is the average class accuracy of all classes. Mean IoU is the mean value of the intersection-over-union metric (IoU) $TP/(TP + FP + FN)$, where TP, FP and FN are the numbers of true positive, false positive, and false negative pixels for one object class determined over the whole test set.

Results

In the following, we show the performance of the semantic image segmentation models. We compare the results of our network with and without shortcuts to figure out the benefit of the residual shortcuts.

We record the cross entropy loss (cf. Eq. (3.9)) of a mini-batch (5 images) at each iteration during training on the Cityscapes dataset and show the results in Fig. 3.2. The network with shortcuts shows better convergence during training. It

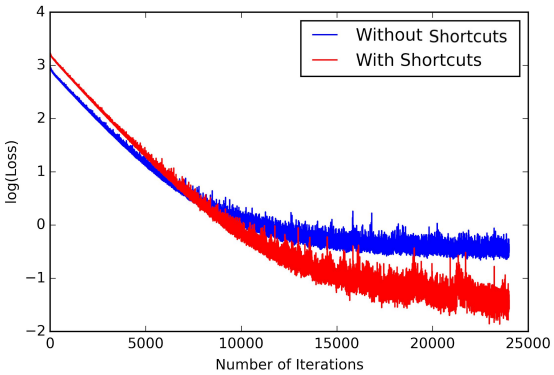


Figure 3.2.: The convergence performance during training

started with a larger loss but converged faster to a smaller value than the network without shortcuts.

We evaluate the performance of the models by testing them on the corresponding test datasets. The segmentation results with the introduced metrics are shown in Table 3.1 and Table 3.2.

Table 3.1.: Comparison of segmentation results on CamVid Dataset

Method	SKY	BUILDING	COLUMN POLE	ROAD	SIDEWALK	TREE	SIGN SYMBOL
Base Network	82.9	75.0	12.0	98.0	58.9	80.0	4.0
With Shortcuts	**93.0**	**87.0**	**22.0**	98.0	**67.0**	**81.0**	**25.0**
	FENCE	VEHICLE	PEDESTRIAN	BICYCLIST	Global	Ave.	Mean IoU
Base Network	**54.0**	46.0	23.0	69.9	77.1	54.8	39.9
With Shortcuts	50.9	**82.9**	**51.9**	**88.9**	**84.7**	**67.9**	**54.6**

Table 3.2.: Comparison of segmentation results on Cityscape Dataset

Method	ROAD	CONSTR.	OBJ.	NATURE	SKY	HUMAN	VEHICLE	Global	Ave.	Mean IoU
Base Network	94.9	88.0	9.0	**94.0**	88.9	47.0	**93.0**	89.1	73.5	63.4
With Shortcuts	**98.0**	**89.9**	**36.0**	93.0	**91.0**	**69.9**	88.9	**92.3**	**80.9**	**73.1**

The class accuracies of most classes are increased by using shortcuts in both tables. Moreover, the network with residual shortcuts shows a considerably higher global accuracy, mean accuracy (noted as Ave. in the tables) and Mean IoU. This is because adding shortcuts improves the training performance in the sense of faster convergence and converging to a lower loss (cf. Fig. 3.2). Better training performance leads to a model with better segmentation performance in general.

Few class accuracies are not improved (such as ROAD in Table 3.1), few are with small improvements (such as TREE in Table 3.1, CONSTRUCTION and SKY in Table 3.2) and few are with very minor degradation (such as NATURE in Table 3.2). Notice that these class accuracies are high (around 90%) in both cases of base network and network with shortcuts. They are hard to improve. The slight differences are tenable because both networks are trained from scratch with random initialization.

Two class accuracies (FENCE in Table 3.1 and VEHICLE in Table 3.2) are decreased. In the CamVid dataset, FENCE is a very challenging class to segment because it has some similar features to some other objects such as the tires of bicycles and vehicles. The model with shortcuts tends to improve the class accuracies of BICYCLIST and VEHICLE (see Table 3.1) and the class accuracy of FENCE is reduced. A similar reason is conjectured for the degradation of VEHICLE in Table 3.2. The model with shortcuts tends to improve the class accuracies of HUMAN and ROAD (which are often close to VEHICLE in the image) and the class accuracy of VEHICLE is reduced. Notice that the improvements of mentioned above (BICYCLIST and VEHICLE in Table 3.1 and HUMAN and ROAD in Table 3.2) are more significant than the degradation (FENCE in Table 3.1 and VEHICLE in Table 3.2). This is also reflected in the global metrics in both tables, i.e. the overall segmentation performance is improved by using shortcuts.

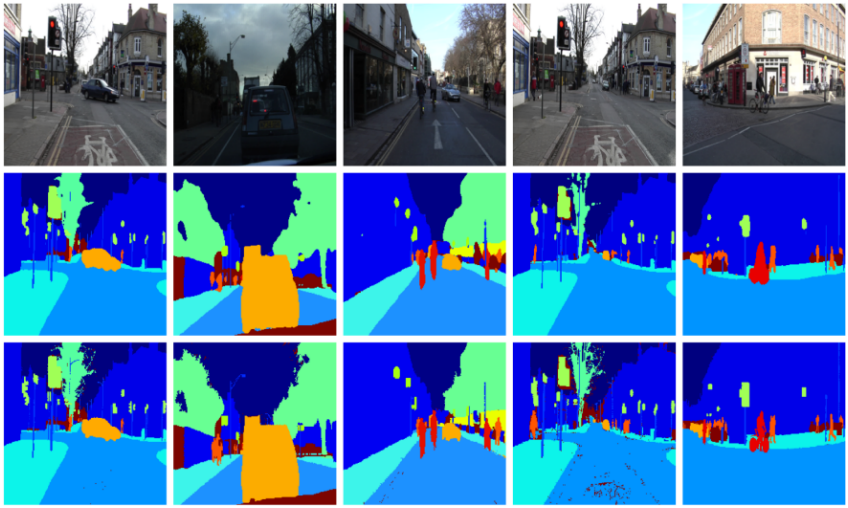


Figure 3.3.: Examples of segmentation results on Camvid dataset. First row: the original images; Second row: segmentation result of our network; Third row: the ground truth segmentation.

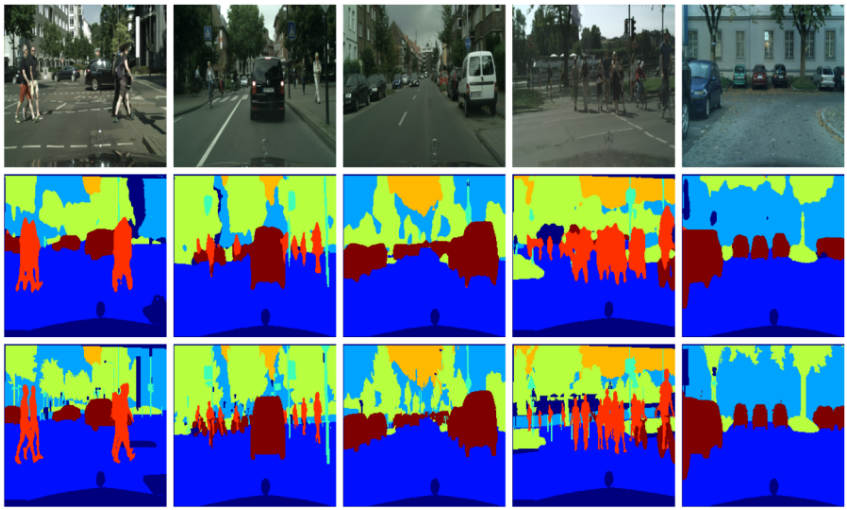


Figure 3.4.: Examples of segmentation results on Cityscapes dataset. First row: the original images; Second row: segmentation result of our network; Third row: the ground truth segmentation.

The segmentation results of several randomly selected test images are shown in Fig. 3.3 and 3.4. The segmentations of our network are close to the ground truth in general. Our segmentation results contain less details than the ground truth. The segmentation boundaries are smoother, which results in less clear object contours. One reason is that we scale down the image size (meaning lower resolution) for training. Another reason is that our encoder-decoder network architecture also downscales the feature maps. This leads to loss of information and results in less details in the segmentation.

All the objects in the images are shrunk in the horizontal direction, e.g. the pedestrians are very thin. This is again due to the rescaling of the original image for training. (We resized the CamVid images from 960×720 and the Cityscapes images from 2048×1024 to 224×224.)

Some classes (e.g. ROAD, SKY, NATURE) are segmented very well and some others (e.g. COLUME POLE, SIGN SYMBOL, PEDESTRIAN, OBJECT, HUMAN) relatively worse. Notice that the former ones often hold large area of an image, while the later ones are generally small objects. According to Eq. (3.9), our loss is summarized over all pixels but without a weight (i.e. a class balance factors) on the class of small objects. Hence, the trained network are more capable on segmentation large objects in general. A class balance factor would improve the segmentation accuracy of small objects. However, this may decrease the global accuracy. Therefore, we do not apply class balance in our training.

3.5. Application to salient object segmentation

In this section, we specify the network architecture introduced in Section 3.2 and examine its use for salient object segmentation.

3.5.1. CNN model

In this example, we use 17 convolutional units, i.e. $L = 17$. We use no shortcuts, i.e., all α in (3.2) are set to 0. With such a plain CNN (without shortcuts), we would like to highlight the benefit and significance of a large dataset in the scope of the supervised learning. Our plain network trained with large datasets with

ground truth segmentation can achieve good results on several popular test datasets in comparison to models with dedicated architecture design for salient object segmentation.

Fig. 3.5 gives an illustration of the specified network architecture. In particular, the upper part of Fig. 3.5 shows the the first 13 convolutional units structured like VGGNet [157]. They are the encoders.

The input to the network $\mathbf{X}^{(0)} \in \mathbb{R}^{500\times500\times3}$ are raw RGB images resized to 500×500. The mean-subtraction is applied as preprocessing, where the mean RGB values of the entire training set are subtracted from each image. Then the inputs are normalized to zero-mean and unit-variance over all pixels of the current mini-batch. The spatial size of the features maps is reduced gradually, and the features are getting more abstract as the network goes deeper and deeper. We use Rectified Linear Unit (ReLU) as activation function and MAX Pooling layer for down-sampling. Five Pooling layers are arranged after the second, fourth, seventh, tenth and thirteenth convolutional units, respectively. The first 13 convolutional layers all have a kernel size of 3. The kernel size of fourteenth and fifteenth convolutional layers are 7 and 1, respectively. Due to a large number of parameters, the dropout technique [159] is used in these two convolutional units to avoid overfitting. The dropout rate is set to 0.5, meaning the corresponding kernel has a probability of 0.5 to be removed from the network during a training stage. Only the reduced network is trained (i.e. parameters are updated) on that training stage. The removed kernels are then reinserted into the network with their original weights before going into the next training stage.

The fourteenth and fifteenth convolutional layers together with the 16th convolutional layer are treated as a transition block, after which a deconvolutional layer with a stride of 32 is used as a decoder. It upscales the feature maps by transposed convolution. The final feature map of the network is cropped back to 500×500. Notice that more deconvolutional layers can be added. We use a single one to restrict the number of parameters. The detailed network structure is given in Tab. B.2.

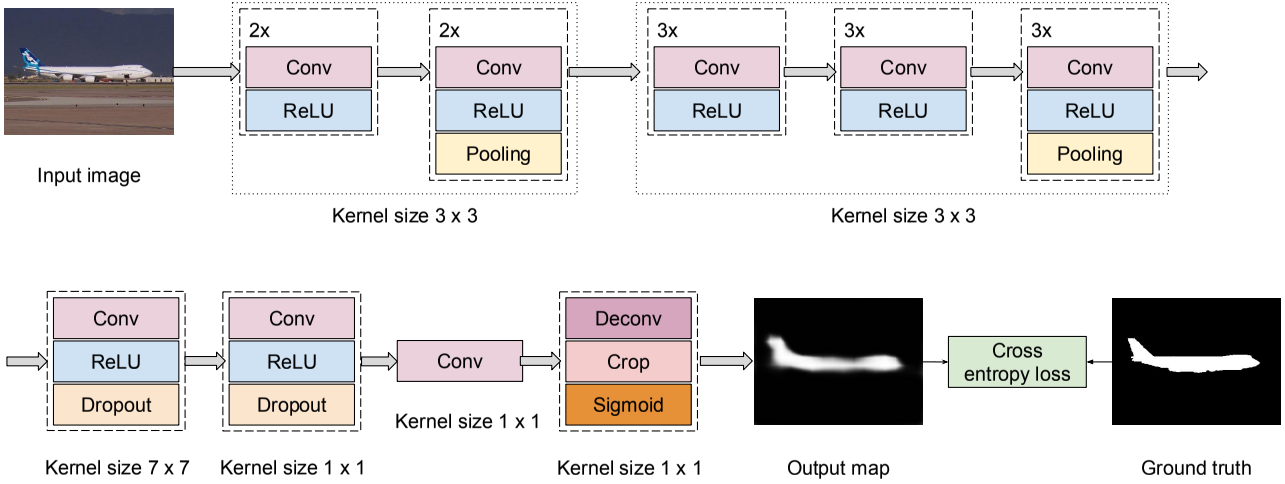


Figure 3.5.: Illustration of the CNN model for salient object segmentation

The salient object segmentation task is in fact a two-class segmentation task, i.e., $C = 2$. However, given one channel of the output map as the probability of belonging to one class, the other map is determinate. Hence, we use a sigmoid cross entropy loss (meaning the cross entropy loss is calculated based on the output of a sigmoid function). The sigmoid layer is used at the end of the network. Let $\mathbf{A}^{(L)}$ be the activation of the last convolutional unit, the output of the network is given by

$$\mathbf{X}_{hw}^{(L)} = \phi^{(L)}(\mathbf{A}_{hw}^{(L)}) = \frac{1}{1 + \exp(-\mathbf{A}_{hw}^{(L)})}. \tag{3.11}$$

Here, $\phi^{(L)}$ indicates the activation function of the last convolutional unit and it is a sigmoid function. It outputs a normalized saliency map ranges in $[0, 1]$. Each value in $\mathbf{X}_{hw}^{(L)}$ can be interpreted as the probability of the pixel belonging to the salient object.

The ground truth segmentation is also a binary map $\mathbf{Y} \in \mathbb{R}^{500\times500}$ and is given as

$$\mathbf{Y}_{hw} = \begin{cases} 1, & \text{if pixel } (h, w) \text{ belongs to salient object region} \\ 0, & \text{otherwise} \end{cases}. \tag{3.12}$$

Similar to the categorical cross entropy in Eq. (3.9), the sigmoid cross entropy is specified as

$$C\left(\mathbf{X}^{(L)}, \mathbf{Y}; \mathbf{\Theta}\right) = -\sum_{h=1}^{H}\sum_{w=1}^{W} \mathbf{Y}_{hw} \ln \mathbf{X}_{hw}^{(L)} - \sum_{h=1}^{H}\sum_{w=1}^{W} (1 - \mathbf{Y}_{hw}) \ln\left(1 - \mathbf{X}_{hw}^{(L)}\right). \tag{3.13}$$

Implementation details

The MSRA10K dataset [27], which is selected from the original MSRA database [116], is used for training. The original MSRA Salient Object Database contains more than 20,000 images with salient object annotated with bounding boxes. MSRA10K dataset contains 10,000 selected images and provides pixel-wise segmentation maps. All images contain at least one dominant object denoted as salient object. Most of the images contain only one salient object.

The weight parameter λ of the regularization term in the loss function Eq. (3.8) is set to 0.0005. To save training efforts, we do not initialize from scratch in this

example. We initialize the first 13 convolutional layers using the Salient Object Subitizing Network (SOS) [192]. It is used to count the number of salient objects in an image and the first 13 layers are same to our network structure. We believe these learned features in the SOS model can be inhered into our network and fine-tuned during training. All other parameters in Θ are initialized according to the method of He et al. [78], namely zero biases and zero-mean Gaussian distributed weights with the standard deviation $\sqrt{\frac{2}{M}}$, where M is the number of input elements to the current layer. More specifically, $M = H_l \times W_l \times O_l$ for the l-th layer. Images and ground truth maps are resized to 500×500 pixels regardless of their original size. The momentum parameter is set to 0.99: The learning rate is set to 10^{-11} because we do not train from scratch.

We use the deep learning framework Caffe [90] to build our network and for training implementation. Different from Keras [32] and Theano [165] used in Section 3.4 which are Python frameworks, Caffe is implemented in $C/C++$ and if more efficient during training. We use SGD algorithm to optimize the network parameters. We update the parameters every 50 iteration, where an iteration indicates a forward computation of the loss based on a mini-batch of 4 images. We limit the batch size to 4 images due to limited computational source. However, the losses of every 50 iterations are summarized for back-propagation and updating of the network parameters. This enlarges the "actual" batch size to 200. We stop tuning at 10000 iterations (200 epochs).

3.5.2. Experimental results

In this section, we test the trained model on different datasets to evaluate its performance.

Test datasets

Two other datasets (different from the training set MSRA10K) are used to evaluate the performance of the methods: ECSSD [155] and SED2 [4]. Both datasets are widely used for evaluation of salient object segmentation methods.

ECSSD contains 1000 images. Comparing to MSRA10K, the images in ECSSD contain more complex background structures. Similar to MSRA10K, there is in general only a single salient object in the mid area of the image. The ground truth maps were manually generated and provided within the dataset.

SED2 contains 100 images. Different to MSRA10K, most of the images in SED2 contain two or more salient objects. Moreover, the salient objects in an image can have very different size, in the sense of containing very different number of pixels. The ground truth maps were manually generated and provided within the dataset.

Evaluation metric

To evaluate the quality of the salient object segmentation results, we use the precision-recall (PR) curve against the ground truth. The precision (P) and recall (R) of a saliency map are computed by segmenting the salient region with a threshold and comparing the binary map with the ground truth. We use the threshold from 0 to 255 to generate 256 precision-recall pairs. The PR curve demonstrates the precision and recall values averaged over all images.

Moreover, we calculate the MAC (Mean Absolute Error) and the F-measure. Let M be the number of the test images, $\mathbf{Y} \in \mathbb{R}^{H \times W}$ the binary ground truth segmentation map, $\mathbf{S} \in \mathbb{R}^{H \times W}$ the output saliency map of a segmentation method,

$$MAC = \frac{1}{M} \frac{1}{HW} \sum_{m=1}^{M} \sum_{h=1}^{H} \sum_{w=1}^{W} |\mathbf{Y}_{hw}(m) - \mathbf{S}_{hw}(m)|. \tag{3.14}$$

This metric summarizes the absolute difference between a ground truth and a saliency map over all pixels and over all images in the test dataset.

The F-Measure

$$F_\gamma = \frac{(1 + \gamma^2) P * R}{\gamma^2 P + R} \tag{3.15}$$

is widely used in other methods [105, 2, 28, 27, 197]. For each pair of precision (P) and recall (R), we get a single F_γ. The largest F_γ is selected as the performance measure. Following the suggestion of Achanta et al. [2], we set $\gamma^2 = 0.3$ to emphasize the importance of precision.

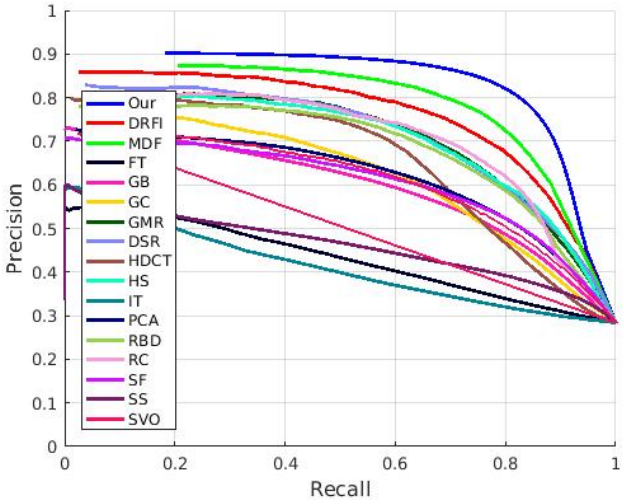
Results

We compare our CNN model with 16 methods: Multiscale deep features (MDF) [105], frequency-tuned saliency (FT) [2], regional contrast (RC) [28], semi-direct visual odometry (SVO) [24], saliency filters (SF) [136], hierarchical saliency (HS) [187], graph-based manifold ranking (GMR) [188], discriminative regional featrue integration (DRFI) [91], pacth-distinctness via PCA (PCA) [127], global cues saliency (GC) [29], dense and sparse reconstruction (DSR) [110], robust background detection (RBD) [197], hihg-dimensional color transform (HDCT) [93], the method from Itti (IT) [86], graph-based visual saliency (GB) [73], sparse salient regions (SS) [82]. These methods cover computational bottom-up, unsupervised methods and supervised learning methods as well as deep learning methods for salient object segmentation. In particular, RBD [197] introduces a hand-designed robust background measure which is then used to optimize the contrast-based saliency map. Jiang et al. [91] proposes a discriminate regional feature and utilizes supervised learning to map the regional feature vector to a saliency map. Li et al. [105] proposes to use multi-scale features extracted using deep convolutional neural networks to learn visual saliency detection model.

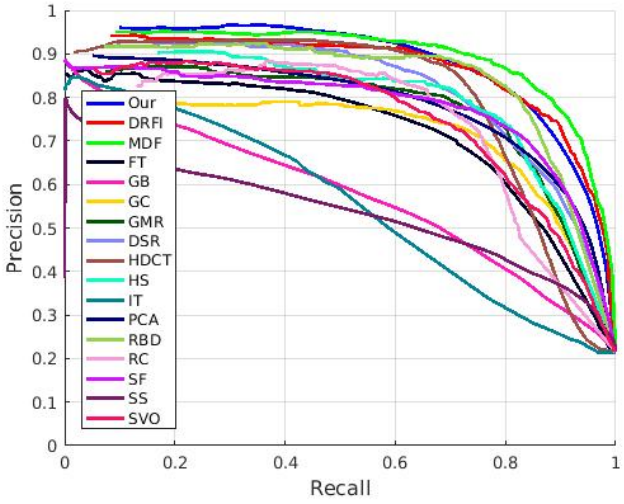
We select these methods because they are well-known and their codes or pretrained models are available online. All the provided models take a single image as input and produce a saliency output map.

Figure 3.6 shows the PR curves of the compared methods. Our model outperforms all the compared methods on the ECSSD dataset in the sense that it gives the highest Precision score at the same Recall value. It is worth to note that ECSSD contains images with more complex structured background than the training dataset MSRA10K. This shows the generalization ability of our trained model. On the SED2 dataset, our model performs comparable to several other methods and is among the best methods. In the Recall ranges under 0.5, our model produces the best Precision score. In the Recall ranges beyond 0.5, our model produces similar Precision score to those of RBD and DRFI and is slightly worse than MDF.

Table 3.3 illustrates the results evaluated using the introduced metrics (including MAE, F-measure). It indicates the similar conclusion to the PR curves. Our model shows the best performance among all compared methods in the sense of the



(a)



(b)

Figure 3.6.: PR curves on two test datasets: (a) ECSSD and (b) SED2.

largest F-Measure and the smallest MAE error value while testing on the ECSSD dataset. On the SED2 dataset, our model shows the smallest MAE error value comparing to other methods, while the F-Measure value is slightly worse than the methods RBD and MDF.

Table 3.3.: Evaluation results with MAE and F-Measure

	ECSSD Dataset		SED2 Dataset	
Method	MAE	F-Measure	MAE	F-Measure
Our	**.155**	**.768**	**.093**	.767
DRFI	.221	.725	.127	.769
MDF	.174	.760	.112	.802
FT	.328	.429	.204	.689
GB	.306	.575	.235	.521
GC	.256	.594	.183	.684
GMR	.237	.698	.163	.756
DSR	.227	.691	.140	.752
HDCT	.250	.662	.158	.717
HS	.269	.648	.155	.769
IT	.314	.340	.244	.557
PCA	.291	.604	.194	.712
RBD	.225	.680	.129	**.803**
RC	.235	.689	.146	.725
SF	.274	.570	.179	.751
SS	.373	.398	.258	.518
SVO	.421	.257	.340	.494

The reason for the above observation is that the ECSSD dataset shares many characteristics with the MSRA10K dataset, which is used to train our model. On the contrary, the SED2 dataset is less similar to the MSRA10K dataset and hence, more challenging to our model. For instance, most of the images in ECSSD and MSRA contain only one salient object locating in the middle of the image. In comparison, the images in SED2 dataset usually contains two salient objects and the salient objects varied a lot in appearance and size. We show some randomly selected image and their ground truth segmentation from MSRA10K dataset in Fig. C.1 in Appendix. Meanwhile, some test examples on ECSSD and SED2 datasets are shown in Fig. C.2 and Fig. C.3, respectively.

In general, the saliency maps achieved by our method are quite close to the ground truth because they smoothly highlight the correct salient object regions. In comparison to other methods, the object contours in our results are blurry. This is due to the downscaling of the feature maps within our encoder network and a heavy upscaling (a deconvolutional layer with a stride of 32) in our decoder network. The details are lost and the object boundaries can not be reconstructed. Other methods either fuse multi-scale image features (e.g. MDF [105], HS [187], DRFI [91]) or directly use low-level features (e.g. RC [28], GC [29], RBD [197]) and produce sharper object boundaries. The drawback of directly using low-level features is that many non-salient object regions will be detected as salient regions and the false positive rate will increase.

Two ways might be used to address the blurry contour issue of our method. First, we can apply a more gentle upscaling in the decoder network by adding more deconvolutional layers and using a small stride size at each layer. The second way is to construct multiple encoder networks with different input sizes and different number convolutional units to extract multi-scale image features [105]. Then, all the features are fused in a appropriate way to produce a saliency map. Both of them will heavily enlarge our network and increase the number of parameters and are not used in this thesis.

3.6. Discussion

Both application examples in the last two sections showed the effectiveness of the proposed network for object-level image segmentation with specified training dataset containing ground truth segmentation as given prior information.

In the first example, we examined the use of the proposed deep CNN for semantic image segmentation with supervised learning strategy. Two network specifications are exemplified: one base network without shortcuts and one with shortcuts insertion. The trained models show rational segmentation results, indicating the effectiveness of applying our method for semantic image segmentation. Residual shortcuts are preferable because the segmentation results of the model with shortcuts are generally better and the training converges faster.

Some limitations in these experiments shall be taken into consideration in order to achieve even better results. Due to computational cost, we downscaled the input images for training and used only 4 encoders and 4 decoders, leading to 30 convolutional layers. On one side, higher image resolution provides more detailed information. This can benefit the trained model and improve the segmentation performance. On the other side, 30 convolutional layers does not significantly reflect the advantages of residual shortcuts in comparison to He et al. [79], where the network consists of up to 1000 convolutional layers.

In the second example, we show the effectiveness of the supervised training of the proposed CNN for salient object region segmentation. Many works [105, 106, 194] design complicated models for salient object segmentation. We build a plain CNN (without shortcuts) and show that it can be trained to achieve comparable results with respect to the state-of-the-art deep CNN using sophisticated architecture (e.g. MDF). It is, however, to notice that we directly represent and evaluate the segmentation results of the compared methods using the pre-trained models or pre-computed results published by the authors. In comparison to all other methods, our CNN model is the only one which is trained on the large MSRA10K dataset. The performance comparison illustrated in Section 3.5.2 shows that a large training dataset is beneficial for segmentation performance, especially when applying deep supervised learning for object-level image segmentation tasks.

Two aspects of applying the deep supervised learning of CNN models for object-level image segmentation are worth to highlight. First, deep CNN architecture allows automatically learning of image features from low-level to high-level. This helps the object-level image segmentation tasks because the low-level feature is helpful for determine the local details (such as object boundaries) and the high-level features are used to make decision of object classes. Second, supervised training of deep CNN models with large datasets automatically encodes the context information. On one side, this context-awareness allows the model to infer an object class when the context of the object is well-known. On the other side, the model will fail to generate reasonable segmentation results if there are large context changes. Below, we use several test examples to illustrate these aspects.

3.6.1. Exploring saliency-related features

In the first test, we want to explore how our salient object segmentation model localizes salient objects. To do this, we examine the activation maps given an image as input and summarize the features learned by a specific convolutional kernel. We use the toolbox developed by Yosinski et al. [189]. Below, we discuss on several observed saliency-related features which were learned by the supervised training.

Saliency-related features from low-levels to high-levels

Salient object segmentation requires image information from low-level to high-level. Several methods [27, 197, 28, 86] explicitly design contrast based features for saliency detection. Through supervised learning with a large dataset, our model learned features from different levels for the salient object segmentation task automatically.

An example of an automatically learned low-level features. is shown in Fig. 3.7. The activation of a selected shallow convolutional kernel (the 25-th kernel of the first convolutional layer) shows a "edge-like" feature, which can be interpreted as the local contrast.

Comparing to the low-level feature, the higher-level features have less focus on the local details. Rather, they often reflect object-related information. Fig. 3.8 shows the activations of two selected deeper convolutional units (the 2153-th and the 2910-th kernel of the 15-th convolutional layer). The activation maps either highlight or suppress the salient object region. This shows the capability of our model to capture high-level object-related information.

Background prior

Background prior is widely used in many saliency object detection methods [27, 197, 183, 177]. Because humans often take photos by focusing on objective in the central area of the field of view, the boundary area of an image is more likely

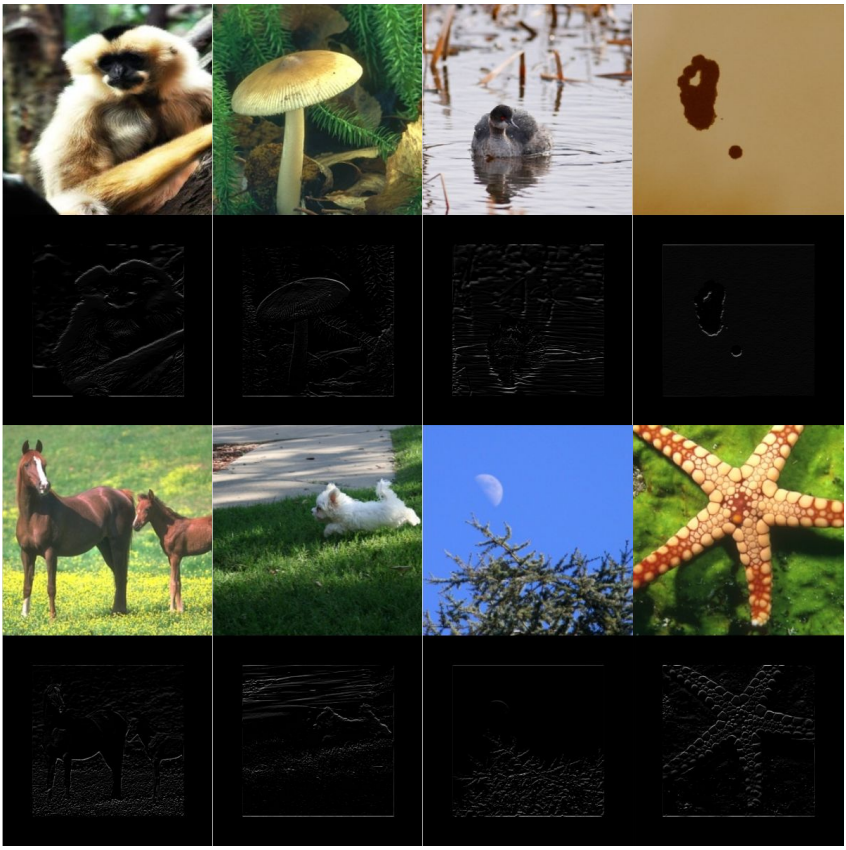


Figure 3.7.: Activation maps of the 25-th kernel in the first convolutional layer. Row 1 and 3: input images. Row 2 and 4: the corresponding activation maps.

Figure 3.8.: Activation maps of two kernels of the 15-th convolutional layer which learn high-level features. The upper part shows that the 2153-th kernel learns object contours and the lower part shows that the 2910-the kernel learns to highlight the salient object regions. Row 1 and 3: input image; Row 2 and 4: the activation maps of the 2153-th and the 2910-th kernel of the 15-th convolutional layer, respectively.

to be background area. This assumption is often denoted as a background prior and used to detect salient objects in an image.

We found that our model automatically encodes this prior. This can be conjectured from the observation shown in Fig. 3.9. A kernel (the 276-th kernel of the 15-th convolutional layer) always activates the image boundary area. This activation can be combined with other features to support the salient object segmentation.

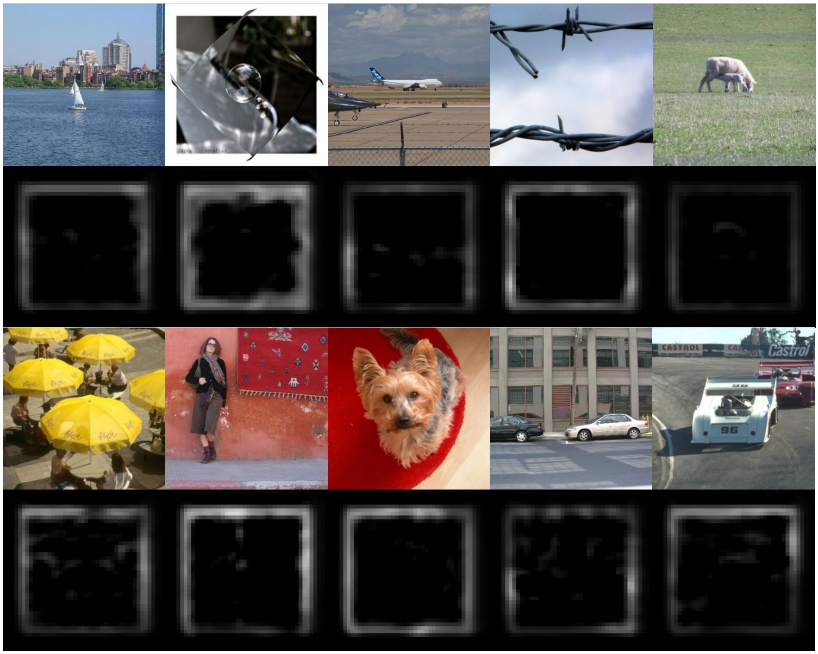


Figure 3.9.: Activation maps of the 276-th kernel in the 15-th convolutional layer. Row 1 and 3: input images. Row 2 and 4: the corresponding activation maps.

3.6.2. Automatic context encoding for semantic segmentation

In Section 3.4, we trained semantic segmentation model on the Cityscapes database. It achieves good performance on the test set of Cityscapes database.

As it learns on street scene images, we conjecture that it encodes the street scene context. To show its generalization ability to other street scene images (i.e. not from Cityscapes), we test the model with street scene pictures randomly selected from Internet.

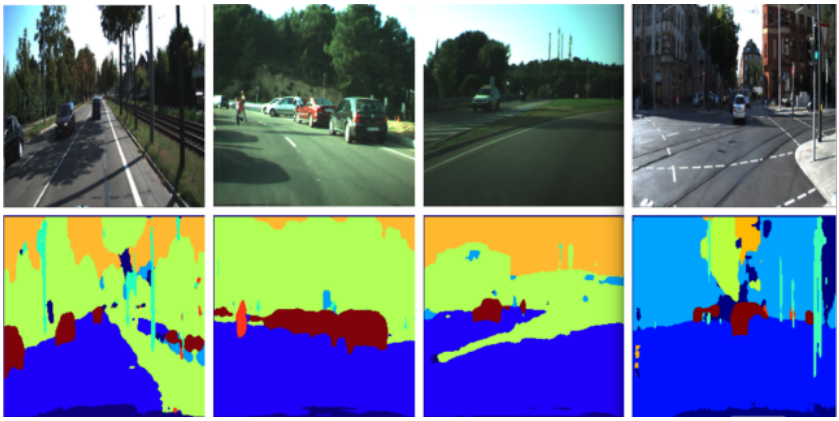


Figure 3.10.: Generalization ability: Semantic segmentation model trained on Cityscape dataset generates rational segmentation results for test images selected from internet. Row 1: test images selected from internet. Row 2: the corresponding segmentation results.

As shown in Fig. 3.10, our semantic segmentation model is well generalizable to other street scene images with the same or a similar context.

Another observation which supports the conjecture of the context-awareness is shown in Fig. 3.11. We use an image manipulation technique "Appearance replacement" (cf. Chapter 4.2.1) to modify the image content. In particular, the appearance of the object class VEHICLE is changed. Although the cars are no more like cars, our model is still able to recognize them. It can be argued that our model encodes the street scene context during learning. Hence, it recognizes

objects in the images with consideration of the street scene context information.

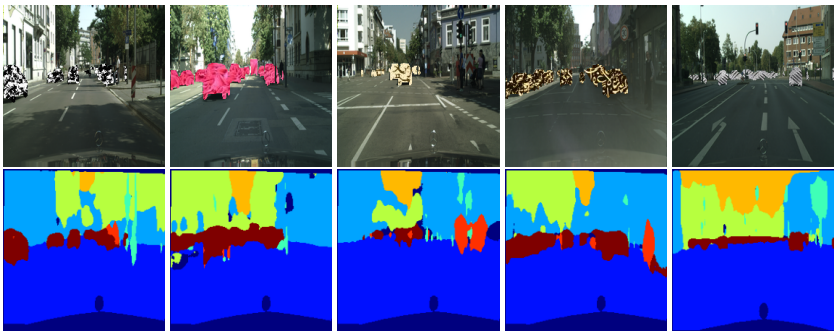


Figure 3.11.: Semantic segmentation results with abnormal VEHICLE class. Row 1: the test images generated by "appearance replacement" of cars (cf. Chapter 4.2.1). Row 2: the corresponding segmentation results.

3.6.3. Limitations and drawbacks

Although we showed that the data-driven deep supervised learning strategy is powerful and effective to solve several object-level image segmentation tasks, the drawbacks are also serious and notable.

The first issue is its dependence on labeled training data. Supervised learning requires an annotated dataset for training the model. Such a training data is hard to collect. The manual annotation of the ground-truth segmentation for object-level image segmentation tasks is extremely challenging and expensive since they are pixel-wise labeling tasks.

Second, a deep CNN contains a large number of parameters to be tuned and a deeper CNN contains more parameters. The setting of the CNN depth is often based on experience. Currently, such a deep CNN is still trained as a black box. Training such a deep CNN requires large training efforts as well as computational power.

Third, although the deep supervised learning leads to automatic context encoding and brings benefits (cf. Section 3.6.2), the context-awareness can also be an

issue in some cases. Being aware of the context may result in the failure case when unusual situations exist in the context. For instance, Fig. 3.12 shows a dangerous case when applying the semantic image segmentation model introduced in Section 3.4. We edit a test image by inserting a sheep on the street. As the training dataset does not contain any animals and the model learned only the pre-defined semantic object classes (cf. Section 3.4), a sheep is an unusual object which is not known to the model. Thus, a sheep standing on the street and in front of the driving vehicle is segmented as ROAD by the model. In conclusion, a CNN model can fail to segment objects not contained in the training set.

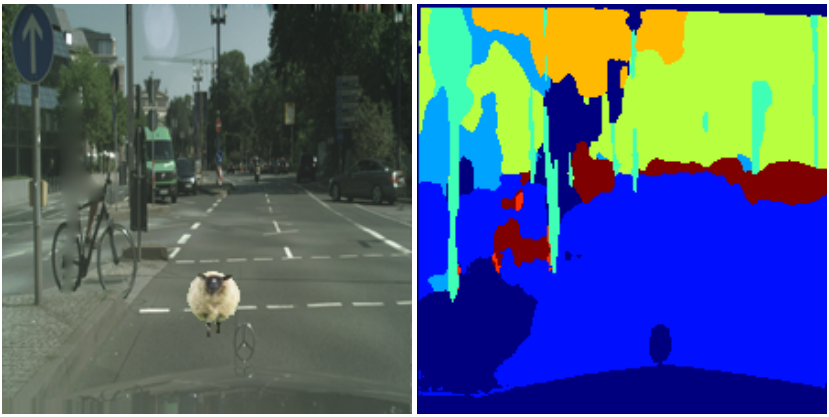


Figure 3.12.: A sheep on the road (left) is segmented as road (right) using the semantic image segmentation model trained on the Cityscapes dataset.

Moreover, encoding of the context information of the given training dataset may reduce the robustness of the model when applying it to other datasets which have different contexts. An observation is shown in Fig. 3.13. The CNN model trained on the Cityscapes dataset generates reasonable results on several urban and high-way images randomly selected from the Internet, though the model has never seen these images during training. However, it fails to segment the humans in an indoor image although HUMAN is also a known class learned during training on the street scene dataset. This is due to the context-awareness of the trained model, because the model is forced to learn several street scene object classes concurrently. Another example is the segmentation result given in Section

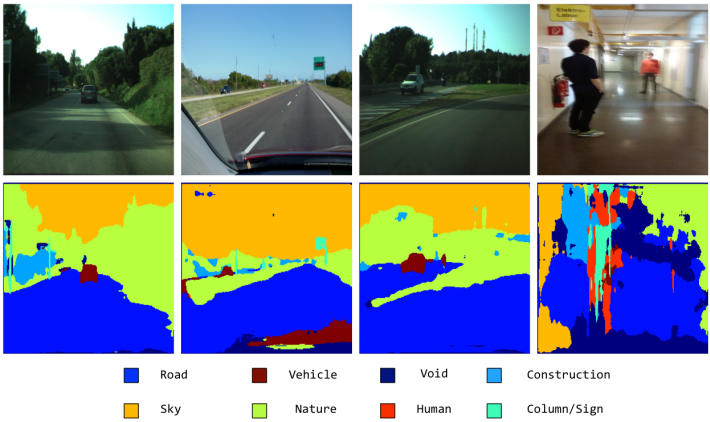


Figure 3.13.: Examples of segmentation results on external images. The segmentation of the three road scene images is fairly good, indicating that the model is quite robust to images with a similar context. However, the indoor example demonstrates that the context change disturbed the segmentation of the humans in the indoor image.

3.5. As our salient object segmentation model shows the best performance on the ECSSD dataset, it is in general slightly worse than the MDF method while testing on the SED2 dataset. This is because our model learned the context information from the training data set MSRA10K (e.g. the image contains a single salient object in the image center). This is similar in ECSSD dataset, but different in SED2 dataset.

3.7. Summary

In this chapter, an object-level image segmentation method using convolutional neural network is proposed under the framework illustrated in Chapter 2. Supervised learning strategy is used. Two application examples are used to illustrate the effectiveness of the proposed method. In particular, two semantic image segmentation models and a salient object segmentation model are trained using large training datasets with given ground truth segmentation. Using the trained models, we discussed the strength of the proposed method with super-

vised learning as well as its drawbacks due to the context aspects. In the next chapter, we study the context-awareness and the context-sensitivity by a more comprehensive case study and exploit methods to alleviate this problem.

Chapter 4.

Object-Level Image Segmentation via Context Adaptation

In the last chapter, we briefly discussed the context-awareness of the segmentation models trained with large dataset containing ground truth. Several qualitative test cases were used and showed that the CNN models are aware of the context. In this chapter, we examine the contextual aspects with a more comprehensive case study based on the semantic image segmentation model trained in Chapter 3.

Context-awareness has its positive side, namely robustness w.r.t. variations in the input data with the same context as learned by the model, see the generalization experiments in Fig. 3.10 and 3.11. However, context-awareness also has its negative side, i.e. the context-sensitivity. In real applications, a trained object-level image segmentation model may fail when the context changes, see the experiment in Fig .3.11 for indoor applications. Context adaptation is used in this case to solve the problem. In particular, this chapter proposes a context-changing data augmentation approach which utilizes known image manipulation techniques to create context changes for training and thus, enlarge the source context. Because the source context is adapted to cover more scenarios with large context changes, the trained model is expected to be more robust to these context changes.

The work of this chapter is partly reported in [176].

4.1. Introduction

In classical supervised learning, an image dataset (containing the ground truth segmentation) from a certain context is given as prior information. We call this context of the labelled training data the source context. Models are trained based on the training samples from the source context and make predictions on unseen data from the target context. It is, however, to notice that an implicit assumption is made in the classical supervised learning, i.e. the source context and the target context are the same.

Chapter 3 shows that supervised learning of deep convolutional neural networks (CNNs) is able to train models for object-level image segmentation tasks with good performance. Meanwhile, it was shown by several qualitative tests that these models are weak to solve the segmentation tasks when the context changes.

In practice, the context often changes. For example, in the application of autonomous driving or driving assistance system, a CNN model trained on images of sunny days is desired to also work for images of rainy days or of nighttime. This requires the CNN model to be insensitive to these context changes. One solution might be labelling the images from all possible contexts for the application and use them to train a new model. This involves, however, tremendous efforts in recollecting labelled data and retraining, especially for the object-level image segmentation tasks, because pixel-wise labelling is required.

This chapter contributes in two aspects. First, it continues the study of the context-awareness. The conjecture is, while the network learns to segment different objects and makes decisions to all pixels simultaneously, it automatically encodes the context information. While the last chapter only discussed the context-awareness of the model with a few words, this chapter examines the contextual aspects with a more comprehensive study based on a semantic image segmentation model trained in Chapter 3.4. Several experiments are conducted in a case study and the context of the input images is modified using different image manipulation techniques. In this way, we examine the context-awareness of the trained model by seeing how does it react to the context changes.

Second, this chapter exploits methods to solve the object-level image segmentation task with the prior information that a training dataset with ground truth segmentation is given while the application involves different contexts than the training data. In general, the model trained on the source context is not directly effective to be used to segment the images from the target context. However, if there are determined similarities between both source and target context, we are able to incorporate additional information to train a model which is effective for the target segmentation task. This reduces the efforts of recollecting labelled data to train a new model, while adapting the context to the target application. We call such an approach context adaptation.

Two approaches are possible for context adaptation:

Incorporation of the prior information by adversarial training. Both training data, the one from the target context and without ground truth and the other from the source context and with ground truth, are considered as the given prior information. Incorporation of the prior information is realized by regularization using adversarial network and adversarial training. The goal of the adversarial training is to learn features which are invariant to both the source and target context, but still discriminative to distinguish between the object classes. For this purpose, an adversarial network along with the segmentation network is needed. During training, there is a standard segmentation loss which aims at preserving the discriminative ability of the final feature maps, and there is the regularization adversarial loss which aims to achieve context invariant features. The training procedure is often done like playing a mini-max game, and the final training status is to achieve the Nash Equilibrium.

Although many research works [167, 59, 120, 168] inspired by Goodfellow [68] were emerging in recent years, applying adversarial training to solve the context adaptation problem introduced in this section for object-level image segmentation tasks is still unsuccessful. The reasons are, on one side, due to the challenge of pixel-wise segmentation task, namely, such features are hard to learn. On the other side, there are still uncertainties of training an adversarial network in success due to problems such as model collapse [150]. It is always hard to achieve the theoretical Nash Equilibrium [68]. Many tricks are required during training

such as adding noise [7], using gradient penalty [70], and selecting optimiztion algorithms and activation layers [139].

In the rest of this chapter, we focus on the study of the second solution for context adaptation (introduced below). Further in-depth research of the adversarial training based context adaptation for object-level image segmentation is a future work and outside the scope of the thesis.

Incorporation of the prior information by data augmentation. Data augmentation is widely used in data-driven learning methods. On one hand, it is used to address the problem of lack of training data. On the other hand, it is also useful to enlarge the variability of the training data and in a certain degree, enhance the context-insensitivity of the trained model.

In most cases, the applier just involves as many as possible data augmentation techniques. However, too many training data also aggravated the training difficulty, because the more data requires more training time and training efforts, and the variation of the data may cause the problem of underfitting and non-convergence.

In this chapter, we consider the following prior information given in advance: (1) A training data with the ground truth from the source context; (2) An image-to-image transformation function which shifts the data distribution from the source context to (a subset of) the target context. We propose training with context-changing data augmentation for context adaptation. We map the training data to an extended data which is expected to be similar to the target context.

In the rest of this chapter, we first present a comprehensive case study on the context-awareness and context-sensitivity in Section 4.2. Towards context-insensitive models, Section 4.3 introduces a context-changing data-augmentation method. Section 4.4 discusses the methods for the single object class segmentation task in multiple-context application. Finally, the whole chapter is summarized in Section 4.5.

4.2. Case study on context-awareness

In this section, we continue the study of the context-awareness from the last chapter and examine the conjecture that the CNN automatically encodes the context information with a more comprehensive case study. Through the case study, we use the semantic image segmentation example illustrated in Section 3.4.

We use hand-designed context changes and conduct several experiments on the Cityscapes street scene images [36] to study the context-awareness of the semantic image segmentation model.

4.2.1. Context changes

The context changes can be myriad. In this chapter, we change the context by intentionally modifying the image content. We classify the context changes into three types based on the target object class(es) to be modified: global context changes, local context changes and leave-one-out context changes.

The terms "global" and "local" indicate that the changes are conducted across all and only for one object class, respectively, but not spatially. For instance, the global context changes can be brightness change (e.g. histogram equalization), filtering (e.g. smoothing, sharpening) and geometric transformations (e.g. rotation, flipping) of the whole image. The local context changes can be appearance replacement (replacing the appearance of one object class with another appearance), removal of one object class by region interpolation or by setting the pixels in the corresponding regions to zero. The leave-one-out context changes mean that all object classes are changed except for a predefined foreground object class. To obtain leave-one-out context changes, we use a sequence of local context changes.

We aim to study the context-awareness of our model by observing how does it react to specific changes. For instance, the segmentation results of rotated images could reflect the context-awareness w.r.t. the general location and the orientation of an object class. The relatedness between object classes (if their presence are correlated) can be reflected by comparing the segmentation results with and

without local context changes. For example, the pose of a human riding a bicycle helps the recognition of the bicycle and avoids misclassifying the bicycle as a motorcycle. Moreover, the context-awareness regarding a foreground object class against its background is also interesting because it reflects the transfer ability of the model for segmenting a specific object class in another context. This can be studied using leave-one-out context changes.

The used image manipulation techniques The image manipulation techniques are plentiful. We only select the following ones for our study. They are conventional and easy to implement.

Image rotation. Rotation in clockwise direction and around its center by a certain angle. The rotated image is cropped to be the same size as the original image and the values of pixels outside the rotated image are set to 0.

Image flipping. The image is flipped in the horizontal or vertical direction.

Region interpolation. Given an image region specified by a binary mask, we smoothly interpolate the value of all pixels inside the region from the pixel values on the boundary of the region by solving Laplace's equation (see Appendix A.1). The region is specified by the ground truth mask of an object class.

Appearance replacement. Given an image region specified by the ground truth binary mask of an object class, we replace the region by filling it with an appearance pattern (color and texture), which is abnormal to the object class and hence does not fit to the context learned by the model.

In this paper, we use 10 abnormal appearance patterns shown in Fig. 4.1.

We show the examples of the resulting context changes in Fig. 4.2. Worth to mention here is that the region interpolation generates relatively gentle context changes (the second and 4th columns in Fig. 4.2) because we only smooth the inter region and the contour of an object is preserved. In comparison, the appearance replacement generates stronger context changes (the 3rd and last columns in Fig. 4.2) due to the abnormality of the appearance patterns.

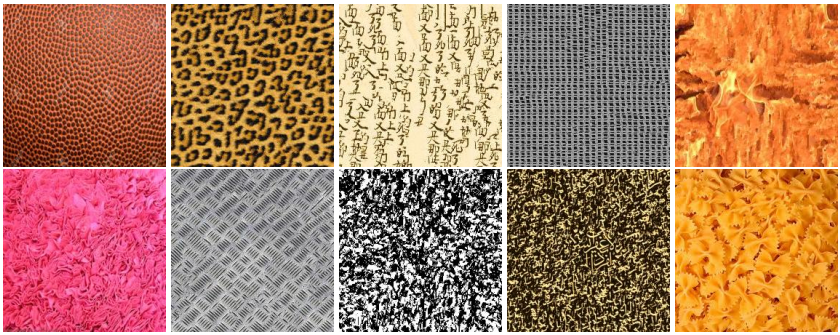


Figure 4.1.: 10 abnormal appearance patterns for replacement of specific object classes in the Cityscapes dataset. The texture synthesis method presented in [47] is used.

4.2.2. Experimental results

In the following experiments, we change the context of the test images from the Cityscapes dataset [36] accordingly, and study how does the trained model react to these context changes. We organize the experiments according to the types of context changes.

To evaluate the segmentation results, we use the same evaluation metrics introduced in Chapter 3.4 except that the mean accuracy is not presented here to save space. It can be simply derived by averaging the given class accuracies over all classes.

Global context changes In this experiment, we study the awareness of the CNN model to global context changes. We use the 500 test images from the test set of Cityscapes dataset.

In the first experiment, we rotate the images with an angle $\vartheta \in \Lambda$, respectively, where

$$\vartheta \in \Lambda = \{15^\circ, 30^\circ, 45^\circ, 60^\circ, 75^\circ, 90^\circ, 105^\circ, 120^\circ, 135^\circ, 150^\circ, 165^\circ\}.$$

We use $\vartheta = 0^\circ$ to indicate the original test image. The ground truth masks are rotated in the same way. Table 4.1 shows the class accuracies for all 7 object

Figure 4.2.: Examples of context changes. The first column shows the original images, the second and 3rd column shows the corresponding images after interpolation and appearance replacement of a random object class. The last two columns show two leave-HUMAN-out context changes using region interpolation and appearance replacement, respectively.

classes plus the void class, the global accuracy of all classes and the mean IoU, all in percentage.

Table 4.1.: Class accuracies, global accuracy and mean IoU for rotated images

ϑ	ROAD	CONSTR.	OBJ.	NATURE	SKY	HUMAN	VEHICLE	Void	Global	MIoU
0°	98.09	90.35	36.34	93.13	91.34	69.76	89.15	90.76	92.35	74.12
45°	82.28	21.33	0.46	83.65	31.05	10.29	24.77	33.57	49.48	23.12
90°	36.57	31.30	3.55	79.93	38.65	1.33	5.1	23.23	37.42	18.61
135°	26.54	8.13	0.09	69.08	1.86	2.85	3.22	27.07	26.06	11.16
180°	16.87	35.65	14.71	70.71	0.33	40.00	6.73	32.63	30.22	16.05
225°	24.23	6.13	0.18	76.21	2.01	13.30	6.43	28.12	26.53	12.03
270°	31.38	28.49	3.42	81.10	41.45	2.32	10.65	21.78	35.36	18.45
315°	86.94	23.14	0.33	86.19	62.91	2.15	24.72	24.56	52.93	26.88

All classes experience degradation of accuracy and thus, the global accuracy and mean IoU reduce significantly. This indicates that our model is sensitive to this kind of context changes. This is reasonable since our model learns to recognize sky on the top, road on the bottom and constructions on two sides. Hence, rotating by 45° and 315° leads to a smaller accuracy reduction for ROAD. Sky can no more be recognized when it is rotated to the bottom of the image (rotation by 135°, 180°, 225°). Similarly, construction can no more be recognized when the image is rotated by 135° and 225°. These observations illustrate that our model is aware of the general location of object classes.

NATURE has in this case the smallest accuracy loss in comparison to other classes because this class includes both trees and grasses, which could be everywhere. Hence, the location awareness is less important for NATURE. The relative high segmentation accuracy of this class despite of image rotation confirms this expectation.

Moreover, our model shows the orientation-awareness. HUMAN experiences a relative small accuracy degradation if the images are rotated by 180°. In comparison, a rotation of 90° and 270° leads to very bad HUMAN segmentation because our model only learned walking or riding humans and no lying persons from the context of a street scene dataset. Similarly, VEHICLE can still be recognized in some cases when they are not rotated by 90°, 135° and 180°, which makes the vehicles inverted. These results illustrate that our model learned the general shape and the orientation of the objects.

In the next experiment, we flip the images in both vertical and horizontal directions. The results are shown in Table 4.2.

Table 4.2.: Class accuracies, global accuracy and mean IoU for flipped images

Flip	ROAD	CONSTR.	OBJ.	NATURE	SKY	HUMAN	VEHICLE	Void	Global	MIoU
Non	98.09	90.35	36.34	93.13	91.34	69.76	89.15	90.76	92.35	74.12
Vert.	15.63	37.18	20.65	72.83	0.29	43.66	7.71	37.90	31.27	17.34
Horiz.	95.59	86.08	18.32	89.24	87.60	57.88	80.16	57.52	84.62	60.64

As expected, the horizontal flipping does not cause a serious degradation of accuracy, since the context in the sense of object appearance, shape, location and orientation remains unchanged. In comparison, the vertical flipping generates similar results as a rotation by 180°.

Local context changes We study the contextual aspects by investigating if the model is aware of the relationship between object classes. Although removing one object class makes the ground truth of that class unreasonable, we intentionally keep the evaluation of that class using the original ground truth in order to additionally show the appearance and the location awareness of the model.

First, we do region interpolation for each of the object classes and evaluate the semantic segmentation on the modified images. The results are shown in Table 4.3. Each row shows the corresponding performance metrics when the region of the object class shown in the first column are interpolated.

Table 4.3.: Class accuracies, global accuracy and mean IoU for region interpolation.

	ROAD	CONSTR.	OBJ.	NATURE	SKY	HUMAN	VEHICLE	Void	Global	MIoU
Non	98.09	90.35	36.34	93.13	91.34	69.76	89.15	90.76	92.35	74.12
Road	98.04	89.69	32.35	92.65	91.18	75.37	88.20	90.03	91.96	72.81
Constr.	97.97	67.03	37.14	94.58	91.18	72.13	88.16	89.35	87.55	64.46
Object	98.10	90.45	21.77	93.21	91.18	70.06	89.17	90.63	92.07	72.45
Nature	98.22	90.86	35.40	30.46	91.39	71.60	88.63	90.50	82.52	59.87
Sky	98.08	90.24	36.23	93.16	91.27	69.79	89.13	90.74	92.33	74.05
Human	98.07	90.16	36.30	93.12	91.35	45.24	89.30	90.68	91.98	71.76
Vehicle	98.08	89.64	35.10	92.75	91.10	71.03	52.98	90.06	89.47	68.25

We notice that the performance variation is small in general. The region interpolation of one object class does not influence the segmentation of other classes significantly. This is due to the gentle context change generated by this manipulation technique. Moreover, the region interpolation of ROAD and SKY does not even effect the segmentation of themselves because the interpolation does not change their appearance very much.

While most classes experience a slight variation, HUMAN shows in several cases relative large accuracy ascension. This happens when the region interpolation is conducted for ROAD and CONSTRUCTION. Since humans are generally walking on the road and near the buildings, the increased accuracies are probably caused by the correction of some former false decisions of the boundary pixels.

In a second experiment, we do the appearance replacement for each object class, respectively. The results on the modified images are shown in Table 4.4. We notice that the appearance replacement of one object class causes accuracy degradation of other object classes except for OBJECT. Note this class has a low segmentation

Table 4.4.: Class accuracies, global accuracy and mean IoU for appearance replacement applied to each object class.

	ROAD	CONSTR.	OBJ.	NATURE	SKY	HUMAN	VEHICLE	Void	Global	MIoU
Non	98.09	90.35	36.34	93.13	91.34	69.76	89.15	90.76	92.35	74.12
Road	25.74	89.74	33.28	90.03	91.06	51.13	60.00	75.55	59.92	43.47
Constr.	97.78	46.25	40.64	83.74	83.92	55.51	81.27	82.20	79.78	55.47
Object	97.58	87.22	47.27	84.79	88.82	61.12	85.79	88.42	89.71	69.08
Nature	97.99	88.84	31.29	8.35	79.06	61.54	82.62	86.80	77.06	51.04
Sky	98.00	89.64	36.30	91.10	0.85	70.03	88.80	90.16	89.16	60.56
Human	97.64	90.03	36.51	91.81	91.14	44.34	87.26	89.99	91.35	70.33
Vehicle	96.88	89.67	35.45	90.75	90.64	63.54	47.67	88.77	88.06	63.82

accuracy in general and contains small regions such as column pole, traffic sign and traffic light. Applying this context change does not alter their appearance very much.

The overall results show the context-awareness regarding correlated object classes. We see strong dependencies of HUMAN and VEHICLE w.r.t. ROAD and CONSTRUCTION, because the biggest accuracy degradation of HUMAN and VEHICLE is caused by abnormal appearance of ROAD and CONSTRUCTION. More such dependencies are found on NATURE w.r.t. CONSTRUCTION and OBJECT and SKY w.r.t. CONSTRUCTION and NATURE. All of them align with the facts that nature often overlaps with construction and objects as well as sky is surrounded by construction and nature.

In both local context changes, we showed an evidence for the appearance awareness of our model, since the local context changes to one object class mostly cause a significant accuracy reduction of that class (see the main diagonal of Table 4.3 and 4.4). This is not the case for region interpolation of ROAD and SKY and appearance replacement to OBJECT due to reasons mentioned above.

Leave-one-out context changes In this experiment, we treat one of the object classes as the foreground (FG) object class and others as background (BG) in order to see how does the model react to context changes to the FG, namely in its BG.

We use both region interpolation and appearance replacement for the leave-one-out context changes w.r.t. each object class. In Table 4.5, we only show the

segmentation accuracy of each object class, when it is treated as FG and all other object classes are modified.

Table 4.5.: Class accuracies for leave-one-out context changes.

Manipulation	ROAD	CONSTR.	OBJ.	NATURE	SKY	HUMAN	VEHICLE
Non	98.09	90.35	36.34	93.13	91.34	69.76	89.15
Region interpolation	92.00	88.60	27.73	93.75	86.33	73.73	88.02
Appearance replacement	94.24	80.89	30.70	64.95	42.27	24.49	33.37

The results demonstrate the awareness of the FG against BG changes. Again, region interpolation does not affect the segmentation of the FG class significantly because the BG classes are smoothed and the context contained in the BG is simplified. Some classes (NATURE and HUMAN) become even more salient than before leading to an increased segmentation accuracy. On the contrary, the abnormal appearance of the BG object classes causes notable context changes to FG, so that the FG class can no more be recognized.

4.2.3. Discussion

The experiments show that the model is less sensitive to some context changes and quite sensitive to others. Comparing the modified image data with the original one, we classify the context changes into two categories:

- **Small context changes** indicate that the modification of the image data does not lead to a different interpretation of the context (cf. definition in Chapter 2.1.1). This means the scene as well as the object and the things in the scene are the same as those in the original image data. This includes the global context change of flipping in the horizontal direction and the local and leave-one-out context changes using region interpolation.

- **Large context changes** indicate that the modification of the image data leads to a different interpretation of the context (cf. definition in Chapter 2.1.1). This means either one or more objects or things can not be interpreted as those from the original image data, or the whole scene can not be interpreted as the same as the original data. This includes the global context change of rotation by non-zero angles, flipping in the vertical direction and

the local and leave-one-out context changes using abnormal appearance replacement.

Such a categorization is based on the human interpretation of the real world scenes. Hence, the small and large context changes also reflect the human perception of objects and related to realistic and non-realistic context changes. With small context changes, the content in the modified images is still semantically meaningful to a certain degree and many small context changes are realistic changes. Non-realistic changes in an image often look "strange" and harm the interpretation of the original image. Thus, they are large context changes.

Fig. 4.3 and 4.4 show some examples of the small and large context changes, respectively, and the corresponding segmentation results. In the first row of Fig. 4.3, all three context changes (column 2, 3, and 4) of the original image (column 1) are small. They do not significantly shock the human perception of the street scene. In particular, the horizontal flipping (row 1, column 2) does not cause any appearance (color and texture) change of the objects in the image. The structure of the semantic object classes are still the same interpretation as the training data: sky is on the top, building is on the side, the pedestrians and the cars are crossing the street (even though in the other direction due to horizontal flipping). The region interpolation of ROAD (row 1, column 3) gently smoothes the street but it still looks similar to street. The region interpolation of VEHICLE (row 1, column 4) smoothes the cars on the street and they are submerged by the street. These small context changes lead to segmentation results (row 2, column 2,3 and 4) with little degradation comparing to the original image. Notice that the cars in the last column are smoothed and segmented as ROAD and not VEHICLE. This is good because it aligns with the human perception.

The first row of Fig. 4.4 shows some large context changes of the same original image as in Fig. 4.3. These context changes lead to different interpretation comparing to training data and result in images which are unconventional to the human perception of a street scene. For instance, all standing objects (e.g. trees, buildings and pedestrians) are "lying" based on the conventional perpendicular view of human perception due to the image rotation of 270 degree (row 1, column 1), while the training data is in align with the normal perpendicular view. In the second example (row 1, column 2), street is on the top and sky on the bottom due the vertical flipping. In the third example (row 1, column 3), the pedestrians

are not like normal human because of the abnormal appearance replacement operation in the HUMAN regions. In the last example (row 1, column 4), all classes except HUMAN are replaced with some strange appearances. This significantly effects the human perception of the street scene. All these large context changes have essential influences on the segmentation results. As shown in the second row of Fig. 4.4, the conventional structure of the street scene as achieved in Fig. 4.3 is hardly recognizable for the first, second and the last example. In the first column, the pedestrians are hardly detected and segmented because such "lying " persons are non-realistic. A large part of the street (especially that in the upper area of the image) is not segmented as ROAD because the street is normally flat but not perpendicular and it is often located in the lower area of the image. In the second column, the CNN model recognizes the street on the upper part of the image mostly as building because building is often located in the upper part of the image but street does not. In the forth column, the background can hardly be segmented by our model. Even the unchanged class (HUMAN) is harder to recognize than before due to the large context changes of the background. Although the third column shows a similar segmentation structure as those in Fig. 4.3, the segmentation of HUMAN suffers from a large degradation because the appearance replacement is non-realistic.

In summary, the small and realistic context changes lead to a tolerable segmentation quality degradation, while the large and non-realistic context changes make the segmentation unsatisfactory.

The core reasons underlying the context-awareness of deep CNN based semantic image segmentation model can be summarized from several perspectives. The source context is determined by the training dataset and the network learns this context by seeing the ensemble of all images during training. More specifically, semantic image segmentation uses a deep CNN which realizes the extraction of different levels of features of the input data and the context information are encoded in these features. Some of these features represent the information about certain objects and some others represent the information about the whole scene. All these information are hidden behind the digital pixels of an image. The ability of extracting such a kind of hidden context information is one of the major advantages of deep CNN compared to traditional computer vision methods.

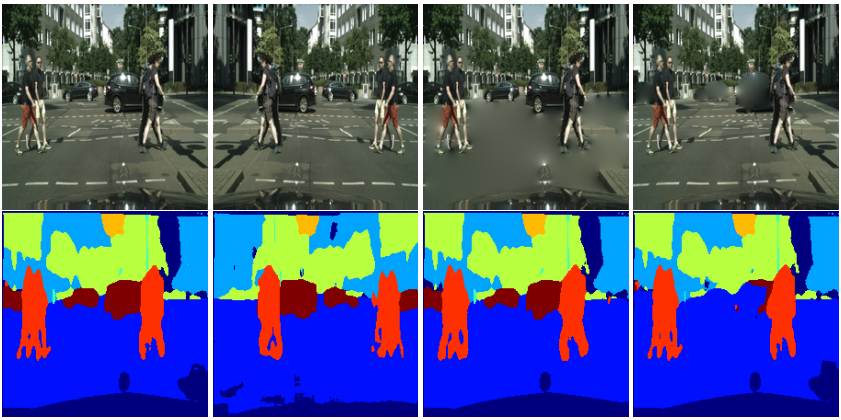


Figure 4.3.: Examples of segmentation results with small context changes. The first row shows the original images and the modified images with small context changes (horizontal flipping, region interpolation of ROAD, region interpolation of VEHICLE). The second row shows the corresponding segmentation results.

Moreover, due to the end-to-end fully convolutional structure, the model was forced to learn multiple semantic classes simultaneously and to label all pixels at the same time. This leads to the automatic encoding of the relationships between the semantic classes, which is also a kind of context information. Hence, the model described above can be carefully used for other street scene datasets for achieving reasonable segmentation results, but it is not suitable for datasets from another context, e.g. an indoor dataset. Moreover, it is worth to note that there are potential dangers when applying our model to the street scene if unknown objects exist in the scene. For instance, our model will fail to segment a lying human on the road because it has never seen this during training. These situations can happen in practice, but they are relatively large context changes compared to the training context. In the next section, we propose context-changing data augmentation to train a model to enhance its robustness.

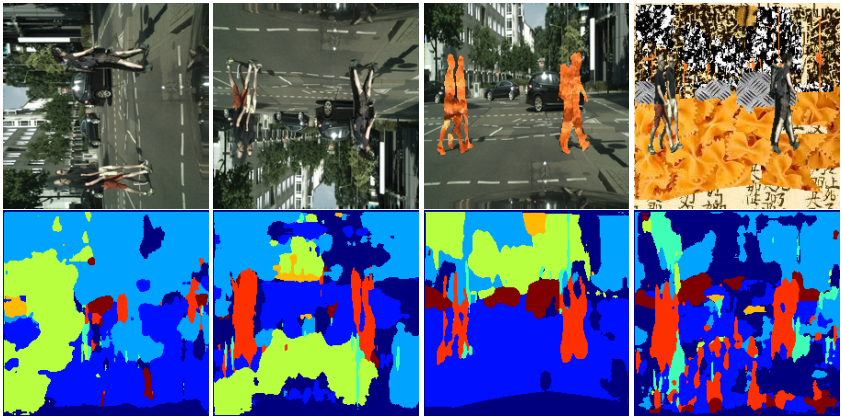


Figure 4.4.: Examples of segmentation results with large context changes. The first row shows the modified images using the same test image as in Fig. 4.3, but with large context changes (rotation of 270 degree, vertical flipping, appearance replacement of HUMAN, leave-HUMAN-out appearance replacement). The second row shows the corresponding segmentation results.

4.3. Toward context-insensitive models

The study in the last section shows that the semantic segmentation model is sensitive to large context changes. In real applications, we often hope to get a model which is trained on limited data to be robust to even relative large context changes. For instance, we want the model trained on day light images to work for images with bad light condition, e.g. images captured in the night. A second example could be that the training is conducted on images from non-raining/non-hazed scenarios while the test images are from raining or hazed scenarios. Due to context-sensitivity, the model could not perform well because the context change between the training and target images is large.

Several works proposed domain adaptation [59, 167, 119] to learn features which are invariant to context changes. Then, the same classifier used for the first context is desired to work in another context. These methods do not necessarily require the labels from all contexts. Instead, domain confusion and adversarial training techniques can be used to transfer the knowledge from one context

to another. While these methods ensure context-invariant feature learning, the learned features are not ensured to be discriminative. Moreover, those methods are generally for image classification, where the output of the network is only a one-dimensional vector containing the probabilities of the whole image belonging to different classes. For semantic image segmentation, these techniques (especially adversarial training methods such as Ganin and Lempitsky [59] and Tzeng et al. [167]) achieve less success since the large-scale two-dimensional pixel-wise classification is more challenging. In addition, the training of adversarial networks itself is tricky, since it often suffers from model collapse problem [67].

Towards a context-insensitive model for multiple context applications, we propose a simple yet effective approach towards training context-insensitive models. Our approach assumes prior knowledge of the training and target context, from which an image manipulation technique for context-change can be derived. For instance, with prior knowledge that the training context are non-raining/non-hazed images, and the target context are raining/hazed images, we can use rendering algorithms [143, 160] to simulate rains/hazes and generate images similar to the target context.

In the following, we assume this image manipulation technique to be known and validate the proposed context-changing data augmentation technique for training context-insensitive models. We leave the research of a generic formulation of such an image manipulation technique between different contexts as a future work.

4.3.1. Problem description

We have an original data set $\mathcal{D} = \{\mathbf{X}(i), \mathbf{Y}(i)\}_{i=1}^{N}$ from the source context Ct_0. $\mathbf{X}(i)$ and $\mathbf{Y}(i)$ are the i-th image and the corresponding ground truth segmentation, respectively. A test data set $\mathcal{T}$ contains images from one or multiple contexts, denoted as a set of context $Ct = \{Ct_k\}_{k=1}^{K}$. For each context $Ct_k \in Ct$, we assume that it is able to find a projection function $\mathcal{A}_k : \mathcal{X} \to \mathcal{X}$ which modifies an image from Ct_0 to another image with the context Ct_k. The corresponding ground truth remains unmodified if the location and the semantic meaning of all classes do not change (e.g., if $\mathcal{A}_k$ is a rain rendering algorithm). Otherwise, it can be obtained

through the same geometrical transformation as $\mathcal{A}_k$ or through semantic class modification according to $\mathcal{A}_k$. The total number of changed contexts in $\mathcal{T}$ with respect to $\mathcal{D}$ is K. We aim at training a CNN for semantic segmentation using only $\mathcal{D}$ which is, however, generalizable to all contexts in $\mathcal{T}$.

4.3.2. Context-changing data augmentation

Our solution to the above problem is context-changing data augmentation. This is simple and straightforward. Comparing to the conventional data augmentation techniques, our context-changing data augmentation is more objective oriented, since it directly incorporates the prior information and thus, is expected to be applicable to all the target contexts. For each image in $\mathcal{D}$, we generate K augmented images using $\mathcal{A}_1, \cdots, \mathcal{A}_K$, respectively. The ground truth of this image is modified accordingly if necessary. Then we train a model using all $(1 + K) \times N$ image pairs in the same way as described in the chapter 3.4.

4.3.3. Experiments

In the next experiments, we aim at validating the effectiveness of our context-changing data augmentation for training context-insensitive models.

Experimental settings

We use the same street scene dataset Cityscapes $\mathcal{D}$ containing 2975 training images. The original test set $\mathcal{S}$ contains 500 images from the same context of $\mathcal{D}$. We generate new test sets $\mathcal{S}_1$, $\mathcal{S}_2$, $\mathcal{S}_3$, which are contextually different from $\mathcal{D}$. In this experiment, we use vertical flipping, rotation with a random angle

$$\vartheta \in \Lambda = \{15^\circ, 30^\circ, 45^\circ, 60^\circ, 75^\circ, 90^\circ, 105^\circ, 120^\circ, 135^\circ, 150^\circ, 165^\circ\}$$

and appearance replacements of a random number (< 3) of semantic classes to simulate $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}_3$ from $\mathcal{S}$, respectively. As examined in last section, a model trained using just $\mathcal{D}$ would not work well on $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}_3$.

We trained four models $\mathbf{M}_0$, $\mathbf{M}_1$, $\mathbf{M}_2$ and $\mathbf{M}_3$ using $\mathcal{D}_0 = \mathcal{D}$, $\mathcal{D}_1 = \mathcal{D}_0 \cup \mathcal{A}_1(\mathcal{D})$, $\mathcal{D}_2 = \mathcal{D}_1 \cup \mathcal{A}_2(\mathcal{D})$ and $\mathcal{D}_3 = \mathcal{D}_2 \cup \mathcal{A}_3(\mathcal{D})$, respectively. Here, $\mathcal{A}_1$, $\mathcal{A}_2$ and $\mathcal{A}_3$ are the image manipulation techniques vertical flipping, rotation and appearance replacement, corresponding to the techniques used to generate the simulated test data. We use the same training strategy as in Section 3.4. We train each model maximally 90 epochs but stop earlier during training if the validation accuracy stops increasing. At the end, the four models are trained 40 epochs, 40 epochs, 50 epochs and 90 epochs, respectively. We evaluate the results on the test datasets $T_0 = S$, $\mathcal{T}_1 = \mathcal{T}_0 \cup \mathcal{S}_1$, $\mathcal{T}_2 = \mathcal{T}_1 \cup \mathcal{S}_2$ and $\mathcal{T}_3 = \mathcal{T}_2 \cup \mathcal{S}_3$. We use only the global accuracy metrics, i.e. global accuracy and mean IoU in this case.

Table 4.6.: Global accuracy and Mean IoU for multiple context datasets.

	Global Acuracy				Mean IoU			
	$\mathcal{T}_0$	$\mathcal{T}_1$	$\mathcal{T}_2$	$\mathcal{T}_3$	$\mathcal{T}_0$	$\mathcal{T}_1$	$\mathcal{T}_2$	$\mathcal{T}_3$
$\mathbf{M}_0$	0.8823	0.7066	0.6295	0.6383	0.6909	0.4579	0.3796	0.4012
$\mathbf{M}_1$	0.8842	0.8850	0.7649	0.7367	0.6918	0.6926	0.5379	0.5199
$\mathbf{M}_2$	0.8803	0.8897	0.8722	0.8138	0.6910	0.6914	0.6824	0.6094
$\mathbf{M}_3$	0.8783	0.8770	0.8711	0.8774	0.6915	0.6924	0.6838	0.6960

Results

Table 4.6 shows the evaluation results of all four models tested on all four datasets. Model $\mathbf{M}_0$ does not work well on $\mathcal{T}_1$ and even worse on $\mathcal{T}_2$ and $\mathcal{T}_3$. Similar results can also be found in other upper triangular values in the table. This shows the context-sensitivity of the model again: A model trained on one context is not robust against large context changes.

All lower triangular values and the values on the diagonal (in red) are relatively high and stable. This denotes high segmentation accuracies. This illustrates the success of our context-changing data augmentation technique for training context-insensitive model. Specifically, $\mathbf{M}_1$ trained with vertical flipping augmented data now works on both $\mathcal{T}_0$ and $\mathcal{T}_1$. $\mathbf{M}_2$ is now insensitive to both flipping and random angle rotation and $\mathbf{M}_3$ is now insensitive to all three context changes. Some examples of testing $\{\mathbf{M}_i\}_{i=0}^3$ on images from different target contexts are shown in Fig. C.4.

4.3.4. Limitation

Our approach has limitations. First, data augmentation enlarges not only the training data, but also the training context, i.e., the variation of the training images. This, on one side, requires the network to be large enough to capture the whole data distribution and, on the other side, requires more time and efforts for training until convergence. In our experiments, we show that we can obtain models being insensitive to several additional contexts. Nevertheless, this assumes to know effective image manipulation techniques derived from prior knowledge of context changes. Moreover, the context insensitivity of our models is limited to the context changes they have ever "seen". As we test these models on images containing other context changes, the models will all perform bad.

4.4. Single-context or multi-context model

In the previous sections, we assume that the semantic object classes are invariant through different contexts and we train semantic segmentation models jointly for multiple object classes. In practice, there are many applications, where we are interested in only one (or a few) target object class, but the context of an application may change. For example, we want to train a robust single-class human model which is able to segment humans regardless of the context, such as walking humans on road, seating humans in office and lying humans on beach. In these contexts, it is no longer feasible and even possible to design effective context-changing functions for data augmentation because the multi-class assumption (roads, humans, vehicles, sky etc.) as for the Cityscapes dataset implies a certain context (of a street scene) and these object classes will not be all available in office or on beach. Naturally, a multi-class model for one context is expected to perform worse on images from another context. The question is how to train a context-free single-class model for the segmentation of one target object class (e.g. HUMAN) independent of the context. This is a different task than semantic segmentation for multiple classes.

Two approaches are possible. In the first approach called cross-context single-class model, a single-class model is trained using datasets from a large diversity

of contexts. Here, the collection of sufficient training data for a single target class in different contexts plays an important role.

In the second approach called context-specific multi-class model, one multi-class model is trained by the dataset of a particular context. Then all multi-class models of different contexts are applied to the same test image of unknown context for the segmentation of the target class available in all contexts. The results of all multi-class models are then combined in a suitable way. If the unknown context is identical to one of the trained contexts, there is a high probability that the corresponding multi-class model will provide a satisfying segmentation result. The disadvantage is that this approach will fail abruptly if the unknown context is new and none of trained multi-class models fits to the test image. Nevertheless, the context-specific multi-class model has the strength that it automatically encodes context-specific information. Since the contextual relationship between different object classes helps the simultaneous segmentation of all classes, a better segmentation of the target object class can be expected than the single-class model.

It is an open question which approach is more robust and efficient for a cross-context single-class segmentation. Due to the lack of labelled datasets for different well-defined contexts, an experimental study could not be conducted yet. This is one future research challenge. Nevertheless, the decision to use a single-class or multi-class model and a single-context or multi-context model depends heavily on the underlying application.

4.5. Summary

This chapter studies the contextual aspects of deep CNN models for object-level image segmentation using a comprehensive case study and proposes a context-changing data-augmentation method for context adaption. Semantic image segmentation is assumed as the target segmentation task.

We systematically modify the context of the input images using different image manipulation techniques and examine the context information encoded in the network by seeing how does the model react to these changes. We conduct several experiments on the Cityscapes street scene dataset and illustrate different

kinds of context-awareness of the CNN model. We conclude that the model is insensitive to some small and realistic context changes but sensitive to large and non-realistic context changes.

Given effective image manipulation techniques for context changes, we show the success of our context-changing data augmentation. We summarize the limitations of our work, discuss approaches for training context-free single-class segmentation models and point out several future challenges.

Chapter 5.

Class Extension in Semantic Image Segmentation

In this chapter, we continue the study of using deep convolutional neural networks for object-level image segmentation. We consider semantic image segmentation in the street scene context as the specified task and investigate how to efficiently extend an already trained deep CNN model to new object classes. In particular, to enable learning with less training and manual annotation effort, we assume a small training image dataset (up to 500 images) with only incompletely labelled or even without ground truth. Meanwhile, we assume that the old CNN model or a binary segmentation model can be used as prior information. Class extension is feasible by reusing the models and incorporating them within the training process on the new training images.

Section 5.1 introduces more background and motivation of our work in this chapter. In Section 5.2, we introduce a basic method which requires manual annotation of only the new classes to be extended. In Section 5.3, a more general method is proposed, which avoids the manual annotation of the new classes by using a binary segmentation model to support the class extension. A new objective function is proposed to incorporate this prior information into training. Section 5.4 summarizes the chapter and discusses the limitations and conclusions.

The works of this chapter are partly reported in [175] and [179].

5.1. Introduction

As stated in the last chapters, training deep CNN models for semantic image segmentation in a supervised manner requires a large number of training samples. The pixel-wise labeling of the input images to obtain the ground truth is very expensive.

On the other hand, a training dataset can only contain a limited number of object classes and a trained model can only handle pre-defined object classes (closed set). However, semantic image segmentation in a real world setup is an open set problem, meaning that we have very large number of object classes, which might grow over time.

Fig. 3.12 in Chapter 3 shows a failure case of applying a semantic image segmentation model. The sheep on the street is segmented as a part of the road by the model, which leads to potential dangers when applying this model for real applications such as camera-based driver-assistance systems. In fact, there are many practical situations, where a well-trained deep CNN model for semantic segmentation of several pre-defined semantic classes is available, but the application changes in the sense that more object classes need to be handled. This motivates our work in this chapter: we investigate how to efficiently extend a well-trained model to some new object classes while preserving the effectiveness of segmenting the old classes. This is called class extension in semantic image segmentation.

5.1.1. Problem formulation

The class extension task referred in this thesis can be formulated as follows.

Given a well-trained model as a mapping function $\Psi : \mathbb{R}^{H \times W \times 3} \rightarrow \mathbb{R}^{H \times W \times C}$, which maps color images to segmentation maps with C object classes. It is typically a deep CNN [106, 132, 9, 175]. $H \times W$ is the image size. The output of the model is a probability map for C object classes.

Given a new training dataset $\mathcal{D} = \{\mathbf{X}(i)\}_{i=1}^{D}$ containing D images from the same or a similar context as the trained model Ψ. The semantic segmentation task is now extended to handle more classes. An overview of such a task is illustrated

in Fig. 5.1. Here, we illustrate the principle for the case of only one new class. An extension to multiple new classes is discussed in Section 5.3.4. The objective of class extension is to find a mapping function $\mathcal{F} : \mathbb{R}^{H \times W \times 3} \rightarrow \mathbb{R}^{H \times W \times (C+1)}$, where the new probability map contains one additional class.

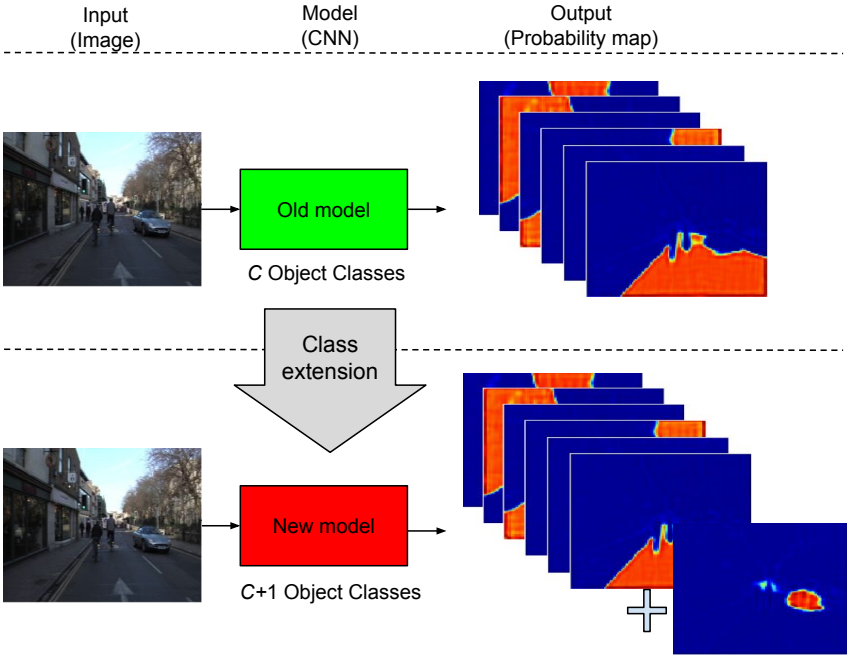


Figure 5.1.: Illustration of the class extension task

5.1.2. Preliminary solution discussion

Several ways are possible for the class extension introduced above.

1. First, we can ignore the old model, build a new one and train it from scratch. This requires both images and ground truth for all $C + 1$ semantic classes. The data requirement is high, meaning that the number N of the new training images shall be large. The annotation task is very expensive.

This is not efficient. Hence, this way is not advisable, especially for the applications, where the old model is still available, but the old training set is often not available or already deleted (e.g. due to privacy reasons).

2. An easier way is to extend the network by adding one channel in the last convolutional unit, inherit the parameters of the old model and fine-tune the new model. This requires in general less training samples. Since the new segmentation task considers $C + 1$ semantic classes, the ground truth segmentation needs to provide pixel-wise labels of $C + 1$ classes. Hence, the manual labelling effort is still large.

3. The manual labelling can be further reduced by reusing the old model to generate approximate ground truth segmentation. As a well trained old model learned the old classes, applying it to the new training images (from the same application domain) can generate good segmentation of the old classes. Hence, manual annotation is only required for the new object class. An approximated ground truth segmentation for all classes is then obtained by overlaying the manual annotation of the new class upon the output from the old model. We show the effectiveness of this idea in the Section 5.2.

4. Further, even the annotation of the new classes can be avoided, given not only the old semantic segmentation model but also a binary segmentation model as prior information. The binary segmentation model contains information about the new class to be extended. Extending the segmentation task is feasible by fusing the information from both old and binary segmentation model. We illustrate this method in details in Section 5.3.

Research works for class extension in CNN-based semantic image segmentation are rare. One reason may be that the semantic image segmentation with fully convolutional network was first proposed three years ago [118]. Although, it has been widely exploited in recent years, it is still at an early stage. Currently, industrial companies apply the idea (1) and (2) for class extension, even though they rely on a large number of new training samples and are too expensive. The idea (3) was shown to achieve success [175].

5.2. Basic class extension strategy

As the first two approaches discussed in the last section involve large efforts of manually labeling, they are out of the scope of this chapter. In this section, we focus on the third solution and show its effectiveness by an experimental study. In Section 5.3, the fourth solution will be introduced in details.

For all experiments, we use the CNN architecture proposed in Chapter 3.

5.2.1. Method

Our basic class extension strategy contains two main steps. First, we need to extend the network. Second, the extended network is trained for the extended segmentation task.

Extension of the network

Our fully convolutional architecture from Chapter 3 can be easily extended with a new object class. We simply add one more channel to the last convolutional unit whose probability map indicates the probability of each pixel belonging to the new object class.

More precisely, the convolutional kernels $\mathbf{W}^{(L)} \in \mathbb{R}^{P_L \times Q_L \times R_L \times C}$ of the last convolutional unit L (as introduced in Chapter 3.2) are enlarged to $\tilde{\mathbf{W}}^{(L)} \in \mathbb{R}^{P_L \times Q_L \times R_L \times (C+1)}$, where $\tilde{\mathbf{W}}^{(L)}_{ijko}$ for $o = C + 1$ is initialized as before (i.e., as described in Chapter 3.4). while other elements of $\tilde{\mathbf{W}}^{(L)}$ are inherited from $\mathbf{W}^{(L)}$.

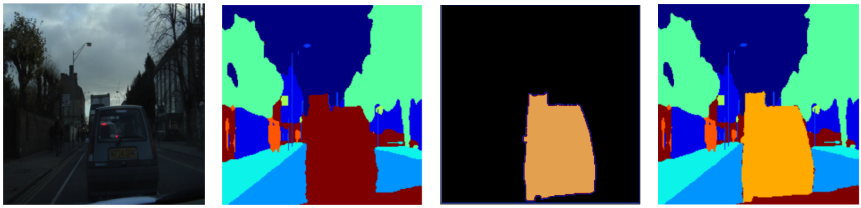
Training of the extended network

We aim to train the above extended network with less training data and annotation effort. We use a small number of $D \in \{100, 200, 300, 400, 500\}$ training images

in our experiments. The ground truth contains only a binary mask $\mathbf{L} \in \mathbb{R}^{224\times224}$ for the new object class, i.e.

$$\mathbf{L}_{hw} = \begin{cases} 1, & \text{for the class } C+1 \\ 0, & \text{for others} \end{cases}, \tag{5.1}$$

and no labels for the old C classes are available. Our training strategy of class extension contains several steps. First, we apply the well-trained C-class model to each of the new training images and get C probability maps $\mathbf{T} \in \mathbb{R}^{224\times224\times C}$ for the old object classes. Then, we generate the new ground truth $\mathbf{Y} \in \mathbb{R}^{224\times224\times(C+1)}$ by concatenating the new object class label $\mathbf{L}$ with $\mathbf{T}$. This process is illustrated in Fig. 5.2.



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

Figure 5.2.: An example of generating the new ground truth for $C+1$ classes (d) for input image (a) with the C-class prediction (b) of the pretrained model and the new label (c) for the new class.

It is, however, to note that we throw away the class predictions of the old model where $\mathbf{L}_{hw} = 1$ and keep the predictions where $\mathbf{L}_{hw} = 0$, i.e.,

$$\mathbf{Y}_{hwo} = \begin{cases} \mathbf{T}_{hwo}(1 - \mathbf{L}_{hw}), & o \in \{1, ..., C\} \\ \mathbf{L}_{hw}, & o = C+1. \end{cases} . \tag{5.2}$$

Finally, we train the extended network using the new ground truth $\mathbf{Y}$ with the same training process presented in Chapter 3.4. Notice that the extended network inherits all parameters from the old model except for the new channel of the last convolutional layer which is initialized randomly. Hence, we use a lower learning

rate 0.001. This fine-tuning strategy preserves the segmentation accuracy of the old object classes while adapting the model to predict the new object class.

5.2.2. Experimental results

In this experiment, we again use the Cityscapes dataset [35] to test our class extension method because it is larger and more challenging than the other street scene dataset CamVid [17]. We assume VEHICLE to be the new class and not known to the old model. The dataset is divided into three parts: 2475 images for the pretraining before class extension (with vehicle class labeled as void), 500 images for training of the new model (with the ground truth segmentation of only the vehicle class, i.e. VEHICLE/non-VEHICLE) and the original test set with 500 images for testing.

In order to examine the amount of training data required to train the extended model, we first construct experiments of class extension with different numbers of training images. We vary this number from 100 to 500. Fig. 5.3 illustrates the results using two global metrics.

Generally, more new training images lead to a better performance. However, the test performance stops increasing roughly after 400 images, a quite small amount of training data in deep learning.

We also compare the test performance of our extended model with the pretrained one. Impressively, the new model performs well and shows good generalization ability. Table 5.1 shows that the new model achieved a class accuracy of 82.9% for the new vehicle class, while the class accuracy of the other classes generally remains constant. The SMALL OBJECT and HUMAN classes experience a degradation of accuracy. Notice that both of them are small object classes containing a low number of pixels in the training data. Since we have not done any class balancing, both classes have a relatively low accuracy before the class extension. Their segmentation accuracy is even worse after the class extension because during training of the new model, we integrate a new class VEHICLE, which is again a relative large object class and contains more pixels in the new training set than SMALL OBJECT and HUMAN. Both classes are even more underrepresented.

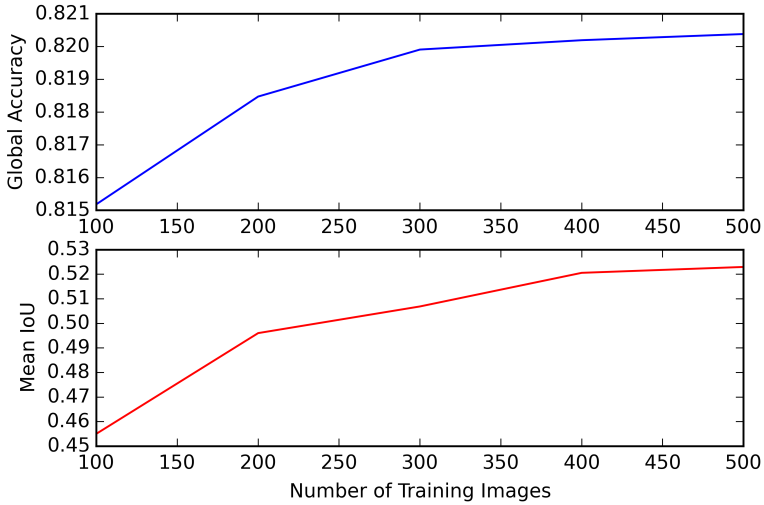


Figure 5.3.: Test accuracy with respect to the number of new training images.

Table 5.1.: Class extension results (Cityscapes Dataset)

	Before extension	After extension
VOID	0.720	0.690
ROAD	0.990	0.990
CONSTRUCTION	0.880	0.889
SMALL OBJECT	0.230	0.160
NATURE	0.880	0.880
SKY	0.949	0.949
HUMAN	0.449	0.300
VEHICLE	-	**0.829**
Global Accuracy	0.872	0.874
Averaged Accuracy	0.727	0.709
Mean IoU	0.532	0.542

5.3. Binary segmentation based class extension

In Section 5.2, the ground truth of the new object class are still needed. In this section, instead of manually annotated segmentation of the D new training images,

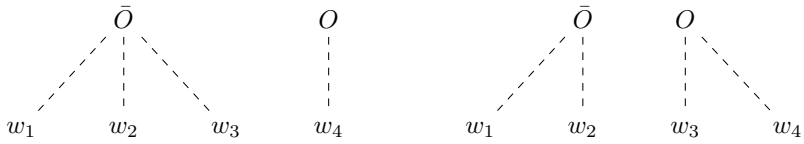


Figure 5.4.: Two typical relationships between the classes.

we propose to use an existing binary segmentation model for class extension. Our method is called BMACE (Binary Model Assisted Class Extension).

Our idea is to use publicly available or pre-trained binary segmentation models instead of the ground truth of new training images. For instance, Jain et al. [88] published a generic foreground object segmentation model for Foreground (FG)/Background (BG) segmentation. Other researchers published salient object segmentation models for salient/non-salient segmentation [107, 194, 108]. These binary classes can be treated as higher-level classes which involve the semantic classes, because semantic objects can often be categorized to FG or BG and salient or non-salient. By fusing the information from both old and binary segmentation model, we are able to extend the segmentation task without the need of manually annotating the new training images.

5.3.1. Binary segmentation model

The binary segmentation model $g : \mathbb{R}^{H\times W\times 3} \rightarrow \mathbb{R}^{H\times W}$ maps an image to a probability map for the class O. The contrary class is notified as $\bar{O}$.

Not all binary segmentation models can be used with our BMACE method. Rather, there are the following conditions on the binary segmentation model.

First, it should contain some knowledge about the new class. In the ideal case, the binary segmentation model is specified for the segmentation of the new class. However, the class O can also contain multiple semantic classes, including the new class to be extended. Fig. 5.4 shows two examples to illustrate the class relationship. Let $\left\{w_j\right\}_{j=1}^{C}$ correspond to the set of old classes and w_{C+1} to the new one. Fig. 5.4 (left) shows an ideal case, where O and w_{C+1} denote the same new class. The right diagram shows another case, where O contains multiple semantic

classes including w_{C+1}. For instance, the four semantic classes in Fig. 5.4 are Sky (w_1), Grass (w_2), Animal (w_3) and Human (w_4). The binary segmentation model can be derived from a human detection task and outputs a probability map for Human (O). It is then identical to the new class w_4. But it is also possible that the binary segmentation model distinguishes between foreground (FG, O) and background (BG, $\bar{O}$) objects. FG (O) contains then Animal (w_3) and Human (w_4) while FG ($\bar{O}$) contains the classes Sky (w_1) and Grass (w_2).

In addition, the object classes are organized in such a way that there is an integer a ($1 < a \leq C$), which separates the object classes into $\left\{w_j\right\}_{j=1}^{a}$ and $\left\{w_j\right\}_{j=a+1}^{C+1}$, corresponding to $\bar{O}$ and O, respectively. In Fig. 5.4, $a = 3$ for the case left and $a = 2$ for the right. This means, all $C + 1$ old and new semantic classes are covered by the binary segmentation task, either in O or in $\bar{O}$.

5.3.2. Decision fusion and hard targets fusion

With a binary segmentation model illustrated in the last section as additional information, the extended segmentation task with a new object class can be achieved in several ways. In this section, we introduce Decision Fusion (DF) and Hard Targets Fusion (HTF).

Decision fusion (DF)

Given a test image, fusing the decisions from the old and the binary segmentation model is able to obtain the target segmentation. Fig. 5.5 shows the basic strategy of decision fusion. It needs no training of a new model. However, two models (old and binary) need to be stored in the memory.

There are many strategies for decision fusion [186, 115, 81, 95]. For class extension in semantic segmentation, a straightforward strategy was used in [175] to generate combined segmentation maps. Let $\mathbf{T} \in \mathbb{R}^{H \times W \times C}$ and $\mathbf{S} \in \mathbb{R}^{H \times W}$ be the probability outputs of the old and the binary segmentation model, respectively.

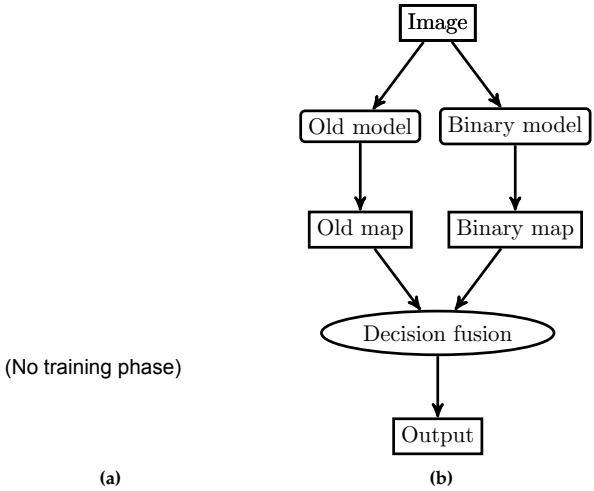


Figure 5.5.: Decision fusion. (a) Training phase (not needed) (b) Deployment phase

The final segmentation map $\mathbf{Y}$ is computed as

$$\mathbf{Y}_{klj} = \begin{cases} \mathbf{T}_{klj}\left(1 - I\left(h\right)\right), & j \in \{1, ..., C\}. \\ I(h), & j = C + 1. \end{cases} \tag{5.3}$$

h is a hypothesis which is true if the binary segmentation model segments the pixel (k, l) into O but the old model classifies it as a class from $\left\{w_j\right\}_{j=1}^{a}$ and false otherwise. Here, the old model classifies by picking the largest probability and the binary segmentation model makes decision by thresholding the probability map $\mathbf{S}$. The threshold is set to 0.5 in this work. $I(h)$ is an indicator function which gives 1 if the hypothesis h is true and zero, otherwise. This decision fusion expects the binary segmentation model to perform well on the target images.

Notice that Eq. (5.3) is generalizable to the computation introduced in Section 5.2. It can be simply interpreted as overlaying the binary mask upon the output maps from the old model in the case of Section 5.2, because it uses the manual annotated binary segmentation.

Hard targets fusion (HTF)

Rather than simply fusing the decisions from two models, it is feasible to train a new model for class extension. In Section 5.2, the ground truth segmentation of the new object class can be interpreted as output from a perfect binary segmentation model, i.e., it is designed for the new class as shown in Fig. 5.4 (left) and performs a binary $O/\bar{O}$ segmentation with 100% accuracy. Here, we provide an extended formulation of this method to handle both cases in Fig. 5.4 and call it hard targets fusion (HTF).

Again, the old model and its parameters are inherited to alleviate the training effort. In particular, the structure of the new CNN differs from the old one by an additional channel in the last convolutional layer. This new channel indicates the probability of the new class. All parameters except for those in the last convolutional layer are copied to the new model for initialization.

Fig. 5.6 illustrate the basic strategy of hard targets fusion.

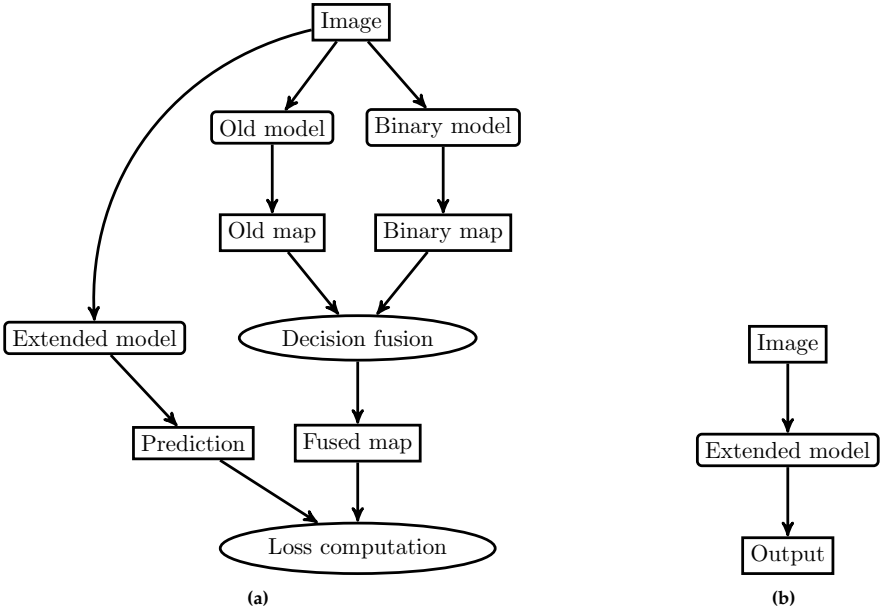


Figure 5.6.: Hard targets fusion. (a) Training phase (b) Deployment phase

Let $\boldsymbol{\Theta}$ be the vector contains all parameters in the new model. It is trained by minimizing the following objective function:

$$\hat{\boldsymbol{\Theta}} = \arg\min_{\boldsymbol{\Theta}} \frac{1}{D} \sum_{i=1}^{D} \mathcal{J}\left(\mathbf{Y}(i), \mathbf{Y}^*(i); \boldsymbol{\Theta}\right) + \lambda \|\underline{w}\|_2^2. \tag{5.4}$$

This function has the similar interpretation as Eq. (3.8). Different from Eq. (3.8) where the main loss function $\mathcal{J}(\mathbf{X}, \mathbf{Y}; \Theta)$ is based on the ground truth segmentation, this objective function use the approximated reference information $\mathbf{Y}^*$. Here, $\mathbf{Y}^* \in \mathbb{R}^{H \times W \times (C+1)}$ is a probability map obtained by decision fusion (Eq. (5.3)) and

$$\mathcal{J}(\mathbf{Y}, \mathbf{Y}^*) = -\sum_{h=1}^{H} \sum_{w=1}^{W} \sum_{j=1}^{C+1} \mathbf{Y}^*_{hwj} \ln \mathbf{Y}_{hwj}. \tag{5.5}$$

This means, HTF trains an extended version of the old model by adding one output channel corresponding to the new class and using the decision fusion as ground truth.

The second term is the ℓ_2-norm regularization. $\underline{w}$ is, again, a column vector containing all elements of $\mathbf{W}^{(l)}$ and $\mathbf{V}^{(l)}$ as given in Eq. (3.1) and (3.6), respectively. λ is set to 0.0005 due to the same consideration as before (cf. introduction to Eq. (3.8) in Chapter 3.3).

5.3.3. BMACE method

In this section, we describe the BMACE (Binary Model Assisted Class Extension) method in detail. It fuses the information from the old model and the binary segmentation model for class extension.

Compared to the basic class extension strategy (Section 5.2) and the HTF method (Section 5.3.2) introduced before, BMACE has three novelties: (1) It does not derives an approximated ground truth for all $C + 1$ classes as before (cf. Section 5.3.2 but directly use the probability outputs from the old model and the binary segmentation model as soft targets. (2) It preserves the distribution of the old classes and distills the knowledge of the old classes using softened probabilities (see the introduction to the first part of the new loss function in this section). (3) It incorporates the information from the binary segmentation model by adding a

regularization term to the loss function (see the introduction to the second part of the new loss function in this section). Fig. 5.7 illustrates the basic strategy of our BMACE method.

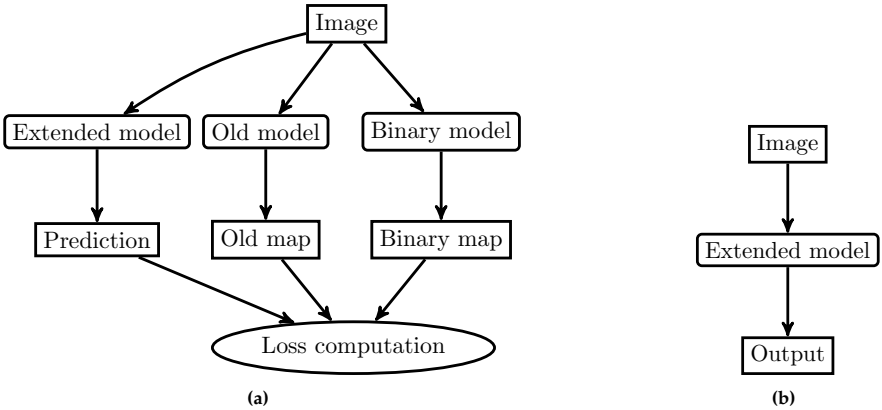


Figure 5.7.: BMACE method. (a) Training phase (b) Deployment phase

Training phase In the training phase, we use two independent softmax layers at the end of the new (extended) CNN (cf. Fig. 5.8):

$$\mathbf{Q}_j = \frac{\exp(\mathbf{Z}_j/\tau)}{\sum_{r=1}^{C} \exp(\mathbf{Z}_r/\tau)}, \quad 1 \le j \le C. \tag{5.6}$$

$$\mathbf{P}_j = \frac{\exp(\mathbf{Z}_j)}{\sum_{r=1}^{C+1} \exp(\mathbf{Z}_r)}, \quad 1 \le j \le C+1. \tag{5.7}$$

This results in two probability maps $\mathbf{Q} \in \mathbb{R}^{H\times W\times C}$ and $\mathbf{P} \in \mathbb{R}^{H\times W\times (C+1)}$ for a single feed-forward computation of the deep CNN. j is the class index (the third dimension). $\mathbf{Q}_j \in \mathbb{H} \times \mathbb{W}$ and $\mathbf{P}_j \in \mathbb{H} \times \mathbb{W}$ are the maps for class j. $\mathbf{Z} \in \mathbb{R}^{H\times W\times (C+1)}$ is the activation of the last convolutional layer of the new CNN. Its channels are then selectively fed into the two softmax layers. Note that the probabilities in $\mathbf{Q}$ are normalized over the old C classes and those in $\mathbf{P}$ are normalized over all $C + 1$ classes. Moreover, τ is a parameter called temperature, which is used only in Eq. (5.6) for the purpose of knowledge distillation (introduced below).

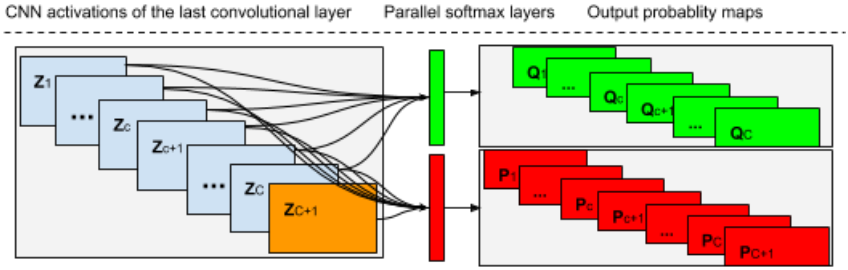


Figure 5.8.: Illustration of the use of two softmax layers during training

To train the new model, we use the loss function

$$\mathcal{J}_{new} = -\sum_{k=1}^{H}\sum_{l=1}^{W}\sum_{j=1}^{C}\mathbf{T}'_{klj}\ln\mathbf{Q}_{klj} - \beta\sum_{k=1}^{H}\sum_{l=1}^{W}\left(\mathbf{S}_{kl}\ln\sum_{j=a+1}^{C+1}\mathbf{P}_{klj} + (1-\mathbf{S}_{kl})\ln\sum_{j=1}^{a}\mathbf{P}_{klj}\right). \tag{5.8}$$

Comparing to the one used in Eq. (5.5), the new loss function contains two parts.

The first part is used to preserve the segmentation of the old classes and to distill the knowledge of the old model. A well-trained model Ψ contains useful information of the old classes. This information is not only shown in the final class decision of a pixel, but also in the probabilities assigned to other classes. Hence, we use the probability map $\mathbf{T}' \in \mathbb{R}^{H\times W\times C}$ as the reference information (soft targets) for the old C classes. This penalizes when the predication of the old C classes do not agree with the old model. Notice that both HTF (cf. Section 5.3.2) and the new BMACE method use soft targets (probabilities), while the basic class extension method (cf. Section 5.2) uses one-hot encoded hard targets.

A CNN model often contains a softmax layer at the end of the network which produces a probability map $\mathbf{X}^{(L)} \in \mathbb{R}^{H\times W\times C}$:

$$\mathbf{X}^{(L)}_{hwj} = \frac{\exp\left(\mathbf{A}^{(L)}_{hwj}\right)}{\sum_{r=1}^{C}\exp\left(\mathbf{A}^{(L)}_{hwr}\right)}. \tag{5.9}$$

Here, $\mathbf{A}^{(L)}$ is the activation of the last convolutional layer of the CNN. To produce the probability map $\mathbf{T}'$ to train the new model for class extension, we

slightly modify the softmax layer of the old model Ψ by adding the temperature parameter τ:

$$\mathbf{T}'_{hwj} = \frac{\exp\left(\mathbf{A}^{(L)}_{hwj}/\tau\right)}{\sum_{r=1}^{C} \exp\left(\mathbf{A}^{(L)}_{hwr}/\tau\right)}. \tag{5.10}$$

Given a training image $\mathbf{X} \in \mathcal{D}$, the corresponding $\mathbf{T}'$ is computed by feeding it into the old model with the above softmax layer. τ is the parameter also presented in Eq. (5.6). It is set to 1 in conventional softmax activation. Setting τ to a value larger than 1 will result in softened probabilities.

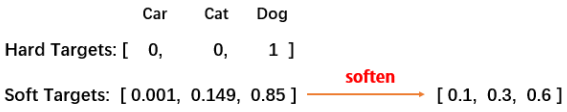


Figure 5.9.: Hard targets, soft targets and softening the targets

Fig. 5.9 gives an example of image classification task with three classes (Car, Cat and Dog) to illustrate the soft targets and the influence of the temperature parameter. Let p_1, p_2 and p_3 be the predicted class labels or probabilities of the class Car, Cat and Dog, respectively. The hard targets show the final one-hot encoded class label of the image, namely $p_1 = 0$, $p_2 = 0$ and $p_3 = 1$. The soft targets with conventional softmax output ($\tau = 1$) give the predicted probability of each class, namely $p_1 = 0.001$, $p_2 = 0.149$ and $p_3 = 0.850$. The soft targets contain more information than the hard targets. They tell us that Cat is semantically closer to Dog than Car because its probability ($p_2 = 0.149$) is higher than Car ($p_1 = 0.001$), though they are both lower than Dog ($p_3 = 0.850$). This information is not contained in the hard targets because both p_1 and p_2 are 0 in that case. A higher temperature for the softmax activation produce a softer probability distribution over the classes, e.g. $p_1 = 0.1$, $p_2 = 0.3$ and $p_3 = 0.6$ shown in the figure. Now it is even more clear that Cat is closer to Dog than Car because the difference between p_1 and p_2 becomes larger while the final decision remains the same, i.e. the class Dog.

Hinton et al. [80] first proposed the distillation technique to exploit the knowledge for model compression. The idea is to use a τ larger than 1 to fuse several existing models and train the final model (compressed model). They showed

promising results for image classification and speech recognition tasks. Here, we apply the knowledge distillation for class extension in semantic image segmentation. We distill the knowledge from the old model Ψ. by setting τ in Eq. (5.6) and (5.10) to a value larger than 1. This results in softened probabilities and allows the implicit knowledge of the object classes to be exploited. It is, however, worth to note that larger τ will decrease the difference between the probabilities of the predicted class and the other classes. Setting $\tau \to \infty$ will results in a uniform distribution over the classes and decrease the discriminative ability of the model. More discussion about setting the parameter τ is given in Section 5.3.4.

The second part in Eq. (5.8) is used to fuse the information of the binary model. $\mathbf{S} \in \mathbb{R}^{H \times W}$ is the probability map produced by the binary model. Let $\mathbf{X} \in \mathcal{D}$ be a training image, it is fed into the binary segmentation model $g : \mathbb{R}^{H \times W \times 3} \to \mathbb{R}^{H \times W}$ to compute the corresponding $\mathbf{S}$. The new model containing $C+1$ classes shall agree with the binary segmentation. Hence, the $O/\bar{O}$ probabilities are used as targets for the summarized probabilities of corresponding semantic classes involved in $O/\bar{O}$.

β is a weight parameter which controls the balance of both terms in the loss function. In the experiment in Section 5.3.4, we set β to 1 (i.e. equally weighted terms) to avoid bias in favour of the old model or the binary segmentation model because we often don't known which model is more reliable.

Deployment phase In the deployment phase, the first softmax layer which outputs $\mathbf{Q}$ is removed from the trained CNN. Given a test image, the extended model outputs $\mathbf{P}$ as the probability map for each pixel. An *argmax* operation is applied to label a pixel to the class with the largest probability.

5.3.4. Experiments

We conduct several experiments to evaluate the proposed method. Again, we choose Cityscapes dataset [35] We evaluate the methods by observing the segmentation results on the test dataset. The metrics are again: pixel-wise accuracy for each class, global accuracy, mean accuracy and mean IoU as introduced in Chapter 3.4.

Setting up

We train an old model for 7 classes only, i.e., ROAD, CONSTRUCTION, SMALL OBJECT, NATURE, SKY, HUMAN and VOID. VEHICLE is not a predefined class and hence belongs to VOID. It is then treated as the new class to be extended. 2475 images were randomly selected from the training set and used to pre-train a model severed as an old model. We use the same network architecture as before, i.e., an encoder-decoder structure with shortcuts for semantic image segmentation. The parameters are initialized using the method of He et al. [78]. The standard cross-entropy is used. The old model was trained for 50 epochs with a fixed learning rate of 0.01.

We use the rest 500 images from the training set for class extension. Note that the ground truth is NOT used here. The new model is initialized by copying the old parameters from the old model to the unchanged layers of the network. The parameters in the last convolutional layer are initialized again according to He et al. [78]. During class extension, we fine-tune the whole model for 30 epochs with a fixed learning rate of 0.003. The temperature parameter τ is set to 5 unless otherwise stated. A discussion about the selection of τ is given at the end of the experiments.

Experiment I

In the first experiment, we assume a perfect binary segmentation model which can provide 100% segmentation accuracy of the VEHICLE class. In this case, the binary segmentation model is an index function which projects a training image to its corresponding manual segmentation of VEHICLE.

Table 5.2 shows our results. With a perfect binary segmentation model, both HTF and BMACE are effective since they preserve the class accuracy of old classes while achieve a high accuracy (over 80%) of the new class VEHICLE. BMACE is better than HTF because it achieves a much higher accuracy for the class SMALL OBJECT and HUMAN. These two class accuracies are even slightly improved comparing to the old model. This is because of knowledge distillation. We use $\tau = 5$ to generate softened probability for training. This contains more information of the semantic classes (as illustrated in Fig. 5.9 and

Table 5.2.: Segmentation accuracy achieved by class extension with a perfect binary segmentation model

	Old model	HTF	BMACE
VOID	0.786	0.716	0.719
ROAD	0.981	0.982	0.978
CONSTRUCTION	0.892	0.911	0.892
SMALL OBJECT	0.294	0.220	0.306
NATURE	0.891	0.875	0.887
SKY	0.928	0.922	0.929
HUMAN	0.576	0.379	0.583
VEHICLE	-	**0.815**	**0.830**
Global Accuracy	0.890	0.881	0.886
Mean Accuracy	0.764	0.727	0.766
Mean IoU	0.668	0.647	0.665

introduced in Section 5.3.3 and allows the model to learn more details of the object classes. In this case, SMALL OBJECTS and HUMAN are most beneficial because segmentation of these small classes requires more detailed information.

Fig. 5.10 shows an example of the segmentation results. Both HTF (column 3) and BMACE (column 4) can segment the cars (shown in yellow) in the test image (column 1), while the old model (column 2) can not. The BMACE segmentation is better than HTF because it is obviously closer to the ground truth (column 5). This observation agrees with our expectation, because the BMACE uses soft targets and applies knowledge distillation for training the new model it outperforms HTF which applies hard targets.

Figure 5.10.: Examples of the segmentation results using our methods. From left to right: input image, segmentation by old model, class extended model (HTF), class extended model (BMACE) and ground truth. The new class to be extended is VEHICLE in yellow.

Experiment II

In conventional supervised deep learning, a large number of training samples with ground truth is required. Our methods have the strength of requiring no ground truth for class extension. Moreover, since we reuse the old model, less training images are needed.

In this experiment, we examine the number of new training images required to extend a new class using our methods. We use the experimental settings as in Experiment I and vary the number of new training images (namely D in Eq. (5.4)) for class extension. D varies over $\{20, 50, 100, 300, 500\}$. Five new models are trained using the same old model and for HTF, BMACE with $\tau = 1$ and BMACE with $\tau = 5$, respectively. 15 models are trained in total for this experiment.

Fig. 5.11 illustrates the results with two global metrics, i.e. mean IoU and global accuracy. The segmentation accuracies of all new models converge very fast with respect to the number of new training images D and tend to saturate when $D = 500$. This means 500 images without any further labelling task is almost sufficient for our class extension methods to achieve success.

Experiment III

In the third experiment, we use a third party model pixel-objectness [87] as the binary segmentation model. It is publicly available. The model was trained on PASCAL-VOC 2012 [52] together with additional samples from Hariharan et al. [74]. All foreground object classes are expected to be highlighted in the objectness map. We normalize the objectness score to $[0, 1]$ and treat it as a probability of O.

Comparing to the perfect model used in the first experiment, this binary segmentation model performs much worse on the Cityscapes dataset because the PASCAL dataset contains images from plentiful contexts and many of them (e.g. the indoor context) are very different to Cityscapes. Nevertheless, by using this binary segmentation model on Cityscapes dataset, we observe that the binary mask roughly highlights VEHICLE and HUMAN areas (Fig. 5.12). We hence

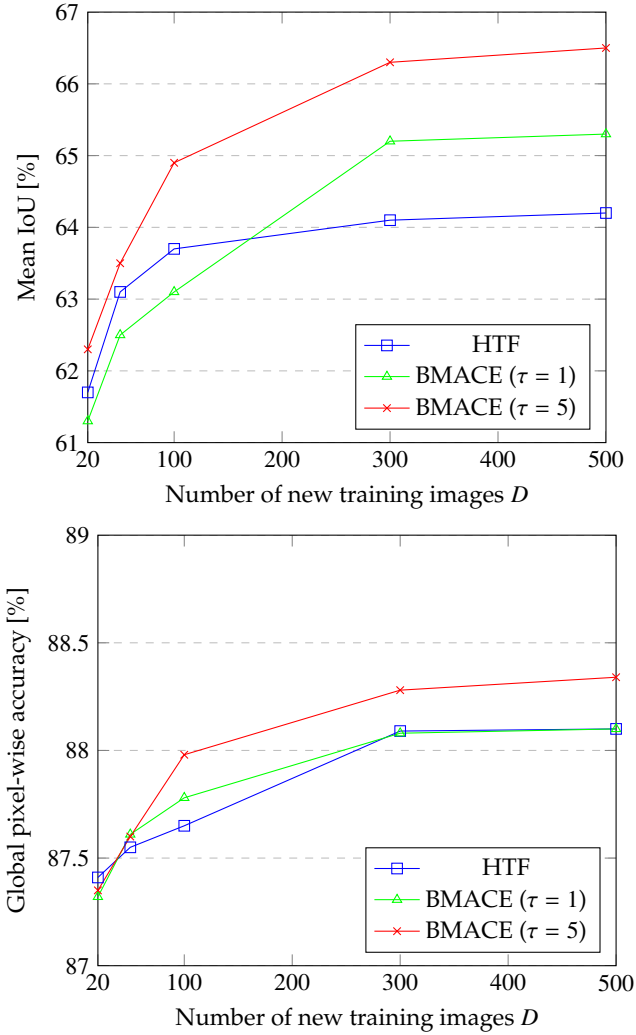


Figure 5.11.: The influence of the number of training images used for class extension.

assume that VEHICLE and HUMAN correspond to O (FG) and the rest semantic classes correspond to $\bar{O}$ (BG).

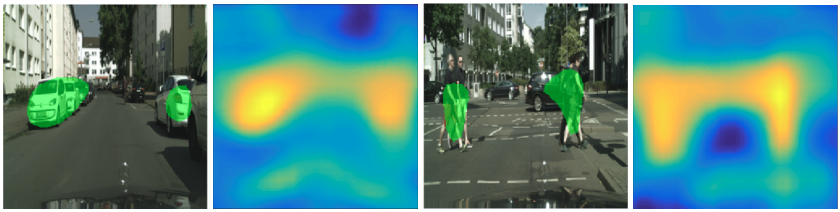


Figure 5.12.: The left column shows the original images from Cityscapes, overlaid with the binary segmentation result (in green) using pixel-objectness [87]. The right column visualizes the corresponding objectness map. The brighter area covers pixels with higher probability of belonging to foreground object.

Table 5.3.: Segmentation accuracy achieved by class extension with an external binary segmentation model for pixel-objectness [87]

Method	DF	HTF	BMACE
VOID	0.596	0.585	0.685
ROAD	0.756	0.757	0.928
CONSTRUCTION	0.698	0.705	0.729
SMALL OBJECT	0.208	0.198	0.153
NATURE	0.715	0.709	0.744
SKY	0.882	0.883	0.917
HUMAN	0.576	0.428	0.546
VEHICLE	0.452	0.345	0.445
Global Accuracy	0.686	0.623	**0.773**
Mean Accuracy	0.610	0.569	**0.643**
Mean IoU	0.494	0.412	**0.525**

Table 5.3 shows the results. Impressively, our method outperforms both decision fusion (DF) and HTF regarding the global metrics and the class accuracies of most classes. This means that our model better tolerates bad binary segmentation models due to a better information fusion and knowledge distillation of the old model. In comparison, decision fusion and HTF rely more on the performance of the binary segmentation model. This shows again the strength of our method and the advantage of using soft targets.

There is a small degradation in the class accuracies of SMALL OBJECT, HUMAN and VEHICLE. This is related to our assumption of mapping the HUMAN and

VEHICLE to FG and the rest to BG. This inaccurate projection causes confusion during the training of the new model.

Fig. C.5 in the Appendix shows some segmentation examples for this experiment. The new class to be extended is VEHICLE in yellow. DF (column 2) produces bad segmentation of the new class because the binary segmentation model performs bad on the Cityscapes dataset. Due to the same reason, both HTF (column 3) and BMACE (column 4) show worse segmentation of the new class VEHICLE in this experiment than before (in experiment I). BMACE segmentation is better than DF and HTF because it shows a higher accuracy of the new class VEHICLE while preserving the old classes.

The influence of the temperature parameter. In our method, τ is used to control the softening of the information. In the experiments above, we set $\tau = 5$ and it shows benefits. To further study the effects of this temperature parameter on our method, we vary τ over $\{1, 2, 5, 10\}$ in the first experiment and observe the segmentation performance.

Table 5.4.: Results of using different temperature values.

τ	1	2	5	10
VOID	0.717	0.720	0.719	0.703
ROAD	0.981	0.981	0.978	0.850
CONSTRUCTION	0.911	0.896	0.892	0.801
SMALL OBJECT	0.191	0.246	0.306	0.556
NATURE	0.890	0.896	0.887	0.845
SKY	0.923	0.943	0.929	0.962
HUMAN	0.451	0.515	0.583	0.743
VEHICLE	0.813	0.823	0.830	0.837
Global Accuracy	0.878	**0.885**	**0.886**	0.816
Average Accuracy	0.735	0.752	0.766	0.787
Mean IoU	0.653	**0.663**	**0.665**	0.597

Table 5.4 shows the results. Both $\tau = 2$ and $\tau = 5$ perform similarly and outperform $\tau = 1$ (without knowledge distillation). Enlarging τ to 10 does not improve but degrades the overall performance regarding global accuracy and mean IoU. The small object classes benefit from large τ, as shown for SMALL OBJECT and HUMAN. However, a too large value of τ degrades the segmentation of large

object classes such as ROAD, CONSTRUCTION and NATURE. This is due to the fact that larger τ softens the targets more significantly (cf. Fig. 5.9) and increases the relative weights of the small objects in the overall loss but too large τ would reduce the discriminative ability of the model. Our method is expected to be robust with $\tau \in [2, 5]$.

Discussion on the extension of more classes. In this paper, we illustrate only the results of class extension of one class due to limited space. A straightforward way of extending more classes is a sequence of our class extension. Moreover, it is worth to note that our method can be easily adapted to extend multiple classes simultaneously given that binary segmentation models are available for each new class. To do this, the loss function needs to be adapted with multiple regularization terms, each incorporating the information of a binary segmentation model.

5.4. Summary

This chapter deals with easy and efficient extension of a well trained segmentation model in the sense of adapting it to more object classes. We notify this as a basic problem of model reuse. With respect to transfer learning, we propose a strategy to adapt our segmentation model to new object classes, while the already trained parameters are preserved. Class extension methods with focus on semantic image segmentation using deep CNNs are proposed. While ground truth is extremely expensive in such pixel-wise dense labeling tasks, we reduce the manual annotation efforts by reusing the old model and moreover, by using an extra binary segmentation model to support the class extension.

In Section 5.2, a basic strategy is used to easily extend a trained network to new object classes without training it from scratch. Benefiting from our fully convolutional architecture, the class extension turns out to be simple and straightforward. We experimentally illustrate that very little data are required for the class extension and thus, the annotation task for ground truth is significantly alleviated.

In Section 5.3, incorporation of binary segmentation models is applied for class extension. First, we propose a baseline method called hard targets fusion (HTF).

This method use the direct fusion of the outputs from the binary segmentation model and the old model to generate approximate reference information. The new model is then trained use this approximated reference information. In the second proposed method called BMACE, we use "soft targets", distill knowledge from the old model and use the binary segmentation model to regularize the training of the new model. Using the Cityscapes dataset, we experimentally show the effectiveness of our methods under multiple settings and with few training samples (up to 500 images). BMACE outperforms HTF by a large margin. The BMACE method can achieve comparable or even better results in comparison with directly fusing the outputs of the old and the binary segmentation model.

Compared to the basic strategy in Section 5.2, the BMACE method proposed in Section 5.3 has the advantage of without annotation effort. However, it requires an effective binary segmentation model as prior information.

Limitations There are three major limitations of the methods proposed in this chapter: First, in the BMACE method, although we claimed that there was no need to manually label the new classes , the binary segmentation model is trained with manually labelled data. Second, in the experiment, we show that our method improved the accuracies of small object classes. But the improvement of small object classes sacrifices a certain degree of accuracy of other classes. Third, due to lack of public datasets, limited experiments are done to illustrate the real use of the proposed method. Experiment I assumed a perfect model for the new object class, which never exists in practice.

Chapter 6.

Unsupervised Object-Level Image Segmentation

The previous chapters deal with learning-based object-level image segmentation. All methods proposed in these chapters rely on a training data and pixel-wise reference information.

In this chapter, we consider unsupervised segmentation, i.e., no training dataset is available. In this case, we are not able to train a CNN model with lots of parameters to be optimized. However, given declarative prior knowledge (as defined in Chapter 2) of the specified task and context, we are able to design features for object-level image segmentation. In this case, the declarative prior knowledge is a kind of weak prior information and is incorporated in feature engineering. Hand-crafted features based on this prior information are designed to exploit the object-level information.

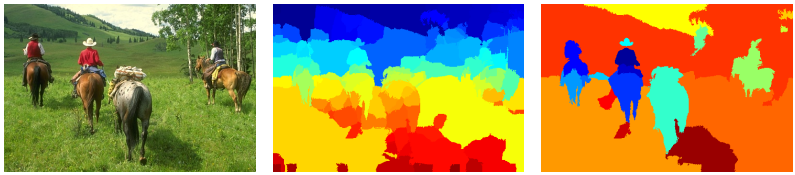
We target two object-level image segmentation tasks and propose two methods to show the design of effective features, respectively. In Section 6.1, we propose a method for generic object-level image segmentation given no pre-defined object classes. We use foreground and background priors, cooperating with several criteria derived from perception organization principles to generate unsupervised object level segmentation. This work is partly reported in [178]. In Section 6.2 we propose a method for unsupervised salient object segmentation which incorporates the object-level information based on bottom-up saliency model. This work is partly reported in [177]. Section 6.3 gives a short summary to this chapter.

6.1. UnOLIS: A region growing based method

Conventional unsupervised image segmentation methods such as mean shift [34], graph-based segmentation (GbS) [55], SLIC [1] and normalized cut [154] often return many superpixels or object parts. They are simple to use but tend to over-segmentation (as shown in the second, fourth and sixth columns of Fig. C.6). Several other methods were proposed to generate image segments in a hierarchical manner [141, 6] so that compact object regions can be possibly derived from a certain high level by cutting the dendrogram or thresholding the contour probability map. For instance, Ren and Shakhnarovich [141] proposed a cascaded region agglomeration method called ISCRA to generate hierarchical segmentation. Starting with a low-level segmentation, they gradually merge regions and train boundary classifiers for every stage of the cascade. Arbelaez et al. [6] proposed the gPb-OWT-UCM method based on their gPb boundary detector [125]. It combines local and global edge features by learning the weight parameters with a logistic regression using subjectively drawn boundaries as ground truth. These methods are supervised and require training data. This restricts their application in practice.

In this section, we present a novel post-processing approach for **un**supervised **o**bject-**l**evel **i**mage **s**egmentation (UnOLIS). As shown in Fig. 6.1, we start with the results of a conventional unsupervised segmentation method. Then, we incorporate prior knowledge to group the pre-segmented regions. We design object-related features such as region saliency, boundary connectivity, and compactness, convexity, as well as other perceptual organization principles of object regions. The proposed UnOLIS method is a multi-stage process, producing final segmentation results which are more semantically meaningful than the conventional unsupervised segmentation.

The following sections first give a brief introduction to the related prior knowledge. Then, a detailed description of the proposed method, in particular, the designed features based on the prior knowledge, is given in Section 6.1. We test our method on the Berkeley Segmentation Dataset (BSDS500) [6] and compare our method with conventional unsupervised segmentation methods. Strength of our method in the sense of creating semantically more meaningful results and avoiding over-segmentation is shown.



(a) (b) (c)

Figure 6.1.: Illustration of the proposed method: For an input image (a), we use a conventional unsupervised method MS [34] to generate the pre-segmentation (b). (c) shows the final segmentation of the proposed method.

6.1.1. Used prior knowledge

Unsupervised grouping of pixels into objects is challenging. Without any reference information (e.g., how does an object look like), segmentation is an ill-posed problem and therefore not solvable. Instead, exploitation of prior knowledge (a kind of weak prior information) is mandatory to provide guidance for grouping pixels or superpixels.

Two kinds of assumptions (prior information used in this section) based on human vision perception principles are introduced below. Using these priors to guide the segmentation, object parts are grouped together in an unsupervised way.

Saliency and boundary connectivity

The knowledge of generic foreground objects are taken into consideration here. In general, the foreground objects in an image are more salient than the background. This is because the foreground objects are often structurally more complex than the background in an image. On the other hand, the background often contains more smooth features (e.g., color and texture). For instance, an airplane is more rational to be interpreted as a foreground object when it is flying in the sky (background). As sky is perceptually flat and smooth, the airplane is perceptually salient. This is called the saliency feature of the foreground objects.

Moreover, the objects located in the center of the image tend to attract more attention and are more likely to be interpreted as foreground. This is because humans often take pictures of an object focusing the central area or near the central area of the view. In real applications such as for autonomous driving, the objects in front (in the center) of the driving direction are critical and need to be detected. This property is interpreted later in this section as the image boundary connectivity, which robustly describes that the foreground objects are less connected to the image boundary.

Perception organization principle

The perception organization principle is also known as the Gestalt theory [71]. It summarizes descriptive rules from psychology that aim to explain how do humans segregate an object from its background, especially, which visual properties of objects support our perception [174]. In this section, we show that these principles can also be used to help designing features for unsupervised object-level image segmentation. The considered principles and the design details are illustrated in Section 6.1.2.

6.1.2. Proposed method

Our method consists of several steps as shown in Fig. 6.2. We pre-segment the image into regions using any conventional unsupervised segmentation method, e.g., Mean shift [34], Graph-based segmentation [55] and SLIC [1]. Then, we combine a global region-based saliency and a robust background feature (boundary connectivity) to cluster the pre-segmented regions into foreground (FG) and background (BG). Afterwards, we treat foreground and background regions in different ways because foreground objects and background often have different properties. In particular, the shape and appearance of a foreground object is often more complicated than the background, e.g. persons as normal foreground objects in an image can have different poses and different wearing while the ground as a normal background object in an image is often flat and grey. Hence, to group foreground object regions, we encode several object priors (compactness, convexity and other perceptual organization principle features) with a region

growing process. In parallel, we group the BG regions into different background objects by Mean Shift clustering algorithm [31] and simply use the color feature. The object-level segmentation is produced by bringing together the results of both foreground objects and background objects.

Below, we describe the main steps after pre-segmentation, in particular, the FG/BG clustering and foreground region grouping processes. Both steps incorporate prior knowledge by crating the feature extraction.

FG/BG clustering

We process the foreground and background regions in different manner because they have different properties. To cluster the pre-segmented regions into FG/BG, we use hand-crafted features. In our experiments, region saliency and boundary connectivity features are designed.

Region saliency Region-based color contrast is a simple but effective way to measure saliency. Here, we calculate the saliency value of a region R by summing up the weighted color contrasts between R and all other regions $R_i \neq R$, leading to a global region-based contrast:

$$s(R) = \sum_{R_i \neq R} w(R_i) \exp\left(-\frac{d^2_{spa}(R, R_i)}{2\sigma^2_{spa}}\right) d_{app}(R, R_i). \tag{6.1}$$

R_i is the i-th region obtained by a pre-segmentation method. $d_{app}(R, R_i)$ indicates the color contrast of R_i to the target region R and is the base to calculate the global saliency value $s(R)$.

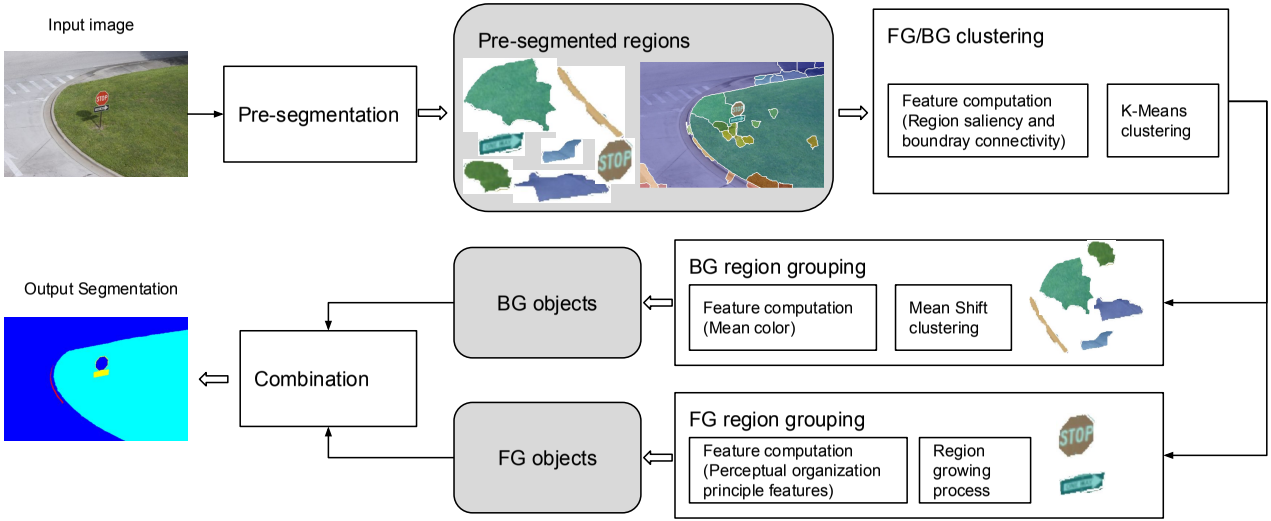


Figure 6.2.: Overview of the method: For an input image, we generate FG objects and BG objects which are combined as the final object-level segmentation.

Given an image $\mathbf{X} \in \mathbb{R}^{n\times 3}$ to be segmented, it contains n pixels and three color channels indicating the three coordinates of a color space, such as RGB, YUV and CIE-Lab. We use the mean color values as the region feature and thus, the feature dimension is 3. More specifically, let $\underline{x} = (x_1, x_2, x_3)^\top$ and $\underline{y} = (y_1, y_2, y_3)^\top$ be the color feature vectors of region R and R_i, respectively,

$$x_j = \frac{\sum_{p\in R} \mathbf{X}_{pj}}{|\{p|p \in R\}|}, \quad 1 \le j \le 3, \tag{6.2}$$

$$y_j = \frac{\sum_{p\in R_i} \mathbf{X}_{pj}}{|\{p|p \in R_i\}|}, \quad 1 \le j \le 3. \tag{6.3}$$

p indicates the p-th pixel in the image. We compute $d_{app}(R, R_i)$ as the Euclidean distance between the two mean color values.

$$d_{app}(R, R_i) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + (x_3 - y_3)^2}. \tag{6.4}$$

We use the CIE-Lab color space because it is designed to approximate human vision. Given an image in RGB, it can be transformed to CIE-Lab as described in the Appendix A.2.

d_{spa} is the (spatial) Euclidean distance between region centroids of R and R_i. Large $d_{spa}(R, R_i)$ will reduce the contribution of the color contrast of region R_i. σ_{spa} is a standard deviation parameter and is set to 0.4 in our experiment. $w(R_i)$ is a weight to stress the color contrast for large R_i. We use

$$w(R_i) = \frac{|\{p|p \in R_i\}|}{n}. \tag{6.5}$$

It is in the range of $(0, 1)$. This is simple and straightforward.

Boundary connectivity In addition to the global contrast based region saliency value, we design a background feature to enhance the accuracy of FG/BG clustering. We consider the background prior which assumes that regions on the image boundary are more likely to be part of background. This is called boundary connectivity in [197]. It provides a measure to the probability that one region R

belongs to the background. Originally, the boundary connectivity is given as

$$b(R) = \frac{|\{p|p \in R, p \in bnd\}|}{\sqrt{|\{p|p \in R\}|}}, \tag{6.6}$$

where p and R are pixel and region in the image and bnd is the set of pixels at the image boundary. $b(R)$ is the ratio of the region perimeter (i.e., the number of pixels which belong to region R and on the image boundary) to the square root of its area.

The intuition is that larger $b(R)$ indicates a region R which is more tightly connected to the image boundary and has a higher probability of belonging to background and vice verse. In fact, all regions which are not touching the image boundary will have $b(R) = 0$ based on Eq. (6.6). This is not good because a presegmentation method often generates many regions (superpixels) which do not touch the image boundary but belong to the background. Using such a boundary connectivity as a feature will fail to classify those regions. A new feature based on the background prior information needs to be designed.

Below, we propose a soft (weighted) version of the boundary connectivity feature. To ensure the discriminative ability on the regions which are not touching the image boundary, we decide to have a boundary connectivity which is always larger than 0. This is intuitive because for any region in the image, you can always find a path with connected regions which ends up at the image boundary. Moreover, we argue that the boundary connectivity shall depend on all regions because the path to the image boundary is not unique and it can cover any region in the image. The proposed feature is computed as

$$b(R) = \frac{bndLen_{soft}(R)}{\sqrt{Area_{soft}(R)}}, \tag{6.7}$$

where

$$Area_{soft}(R) = \sum_{i=1}^{M} w_{geo}(R, R_i) Area(R_i), \tag{6.8}$$

$$bndLen_{soft}(R) = \sum_{i=1}^{M} w_{geo}(R, R_i) bndLen(R_i). \tag{6.9}$$

M is the total number of regions. $Area(R_i)$ is the area of region R_i and is computed as the number of pixels in R_i. $bndLen(R_i)$ is the number of image boundary pixels in R_i. $Area_{soft}(R)$ and $bndLen_{soft}(R)$ are weighted sums of $Area$ and $bndLen$ over all regions. The weight is chosen as

$$w_{geo}(R, R_i) = \exp\left(-\frac{d^2_{geo}(R, R_i)}{2\sigma^2_{geo}}\right), \tag{6.10}$$

where $d_{geo}(R, R_i)$ is the geodesic distance (see below) between R and R_i. $\sigma_{geo} = 15$ is another standard deviation parameter.

In order to define the geodesic distance, we consider each pre-segmented region of an image as a node in an undirected graph. Geometrically connected regions are also connected in the graph by an edge. The edge weight is d_{app} as used in Eq. (6.1) and defined in Eq. (6.4), i.e the Euclidean distance between the mean color values of both regions. The geodesic distance $d_{geo}(R, R')$ between the regions R and R' is the sum of edge weights along the shortest path between R and R', i.e.

$$d_{geo}(R, R') = \min_{R_1=R, R_2, ..., R_n=R'} \sum_{i=1}^{n-1} d_{app}(R_i, R_{i+1}). \tag{6.11}$$

$\{R_1, ..., R_n\}$ is a series of sequential nodes (regions) on the path from R to R'. Comparing to d_{app}, the geodesic distance d_{geo} contains extra distance information, i.e. the length of the path. A region R_i can have a very small d_{app} to R, but it shall not contribute significantly (meaning $w(R, R_i)$ is large) to the soft area and and soft bndLen of R (as defined in Eq. (6.8) and (6.9)) if there is a pair of neighbour regions on the path which have a large d_{app}.

This weighted version of the boundary connectivity $b(R)$ is a more reliable feature based on the background prior knowledge. For all regions, $b(R) > 0$. Larger $b(R)$ indicates that region R is more tightly connected to the image boundary and hence, less probable to belong to a foreground object.

Clustering We do a simple k-means clustering of the two dimensional region descriptors $[s(R) \quad b(R)]$. The cluster with a higher value of s and a lower value of b contains foreground regions and the other cluster contains background regions.

Foreground region grouping

The foreground objects and background objects will be generated based on the clustered superpixels regions. In this subsection, we group foreground superpixel regions to perceptually meaningful foreground objects based on some object priors. We define the set of candidate foreground regions as C_{FG}. As a pre-processing step, we first divide C_{FG} into several components, where each component G is a group of connected regions. Then we try to derive one or more objects from each component. The former step is done because we assume that objects are compactly clustered in spatial space. In the later step, each component can contain one or several objects.

We generate the foreground object list from each component $G = \{R_1, ..., R_n\} \subseteq C_{FG}$. The problem can be formulated as follows:

Given a component $G \subseteq C_{FG}$, there is an energy function $E(O)$ of a *compact* subset $O \subseteq G$.
The goal is to find all non-overlapping compact subsets (objects) $\{O_1, ..., O_m\}$ by

$$\{O_1, ..., O_m\} = \arg\min_{\{O_i\}} \sum_{i=1}^{m} E(O_i)$$
$$s.t. \quad G = \bigcup_{i=1}^{m} O_i, O_i \cap O_j = \emptyset, 1 \leq i < j \leq m.$$

Instead of solving this discrete optimization problem, we apply a region growing approach encoded with some perceptual organization principles (POP). Doing this, we avoid handcrafting the energy function $E(O)$ and training the parameters contained in the function.

We take the following principles into consideration and utilize them as foreground object priors to group foreground regions into objects.

Similarity Similar regions are more likely to belong to a whole. We consider the color similarity to be

$$r_{sim}(R_i, R_j) = \exp\left(-\frac{d_{app}^2(R_i, R_j)}{2\sigma_{app}^2}\right) \tag{6.12}$$

.

Symmetry For simplicity, we assume that objects are more likely to be vertically symmetric. This is true in many cases. For instance, a front standing person can be divided into several parts (head, upper body, lower body), which are all symmetric along the vertical axis. Hence, we measure the mirror symmetry of two regions along a vertical axis by

$$r_{sym}(R_i, R_j) = \exp\left(-\frac{(y_i - y_j)^2}{2\sigma_{sym}^2}\right), \tag{6.13}$$

where y_i and y_j are the column coordinates of the centroids of region R_i and R_j. If $r_{sym}(i, j)$ is large, R_i and R_j are more likely to be grouped together.

Proximity We consider the common border ratio of two regions R_i and R_j to measure their attachment strength. It is defined as

$$r_{prox}(R_i, R_j) = \max\left(\frac{Len_{R_i|R_j}}{Len_{\partial R_i}}, \frac{Len_{R_i|R_j}}{Len_{\partial R_j}}\right), \tag{6.14}$$

where $Len_{\partial R_i}$ and $Len_{\partial R_j}$ are the perimeters of region R_i and R_j. $Len_{R_i|R_j} = Len_{R_j|R_i}$ is the length of their common border. Accordingly, two regions with a higher common border ratio are encouraged to be grouped together.

Convexity The convexity is a global property of the grouped region. We use the perimeter $L(R)$ of region R divided by the squared root of its area $A(R)$ to measure the convexity of region R:

$$conv(R) = \frac{Len(\partial R)}{\sqrt{Area(R)}}. \tag{6.15}$$

Notice that a smaller value of $conv(R)$ indicates a higher convexity. In practice, we measure the change of $conv$ when we merge a region R_j to R_i, i.e.

$$r_{conv}(R_i, R_j) = conv(R_{ij}) - conv(R_i), \tag{6.16}$$

where R_{ij} is the output region of merging R_j to R_i.

Based on the perceptual organization principle (POP) priors, we define a metric to measure the likelihood of merging two regions. At each step, we seed from the most salient region $R_s \in G \subset C_{FG}$, which is considered as a part of an unknown

object. Given neighbor region $R \in G$ connected to R_s, we use a sigmoid function to model the conditional probability that R should be grouped to R_s,

$$p(h = 1|\varphi(R_s, R)) = (1 + \exp(-\varphi(R_s, R))^{-1}, \tag{6.17}$$

where $h = 1$ indicates that R belongs to R_s and $h = 0$ vise versa. $\varphi(R_s, R)$ is a function to measure the strength that prevents merging R to R_s. It is defined as

$$\begin{aligned} \varphi(R_s, R) =& r_{sim}(R_s, R) + r_{sym}(R_s, R) \\ &+ r_{prox}(R_s, R) + \beta r_{conv}(R_s, R), \end{aligned} \tag{6.18}$$

where β is a parameter set to 1. Given a threshold θ, R is grouped to R_s and removed from G, if $\frac{p(h=1|\varphi(R_s,R))}{p(h=0|\varphi(R_s,R))} > \theta$. Otherwise, the current region group is considered as an object and a new region growing is started from a new R_s. We repeat this process until no R_s can be chosen. We set θ to 1.

Performing the region growing process inside each connected component, we get the final foreground object list.

Background region grouping

We apply mean shift algorithm to cluster the background regions from Section 6.1.2. The used features are the mean color values of each region in the CIE-Lab color space. Notice that other clustering methods and other features (e.g. texture) may be used. Their effects are not evaluated here since we focus on the foreground object region grouping.

6.1.3. Experiments

Dataset

We tested our method on the Berkeley Segmentation Data Set (BSDS500) [6]. It contains 300 images for training and validation and 200 images for testing. For each image, BSDS500 provides human-annotated segmentation as ground truth by five different subjects on average. This results in multiple ground truth for a single image. An example of the multiple ground truth is shown in Fig. 6.3. All

three human subjects generate different segmentations of the same image. The first segmentation (figure (b)) contains more details of the background area and the second segmentation (figure (c)) contains more details of the lizard. The third segmentation (figure (d)) contains the lowest number of segmented regions.

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

Figure 6.3.: Exmaple of the multiple ground truth segmentation. (a): Original image, (b): annotation of subject 1, (c): annotation of subject 1, (d): annotation of subject 1.

All 500 images are used in our experiments to compare conventional unsupervised methods of Mean shift (MS) [34], Graph-based segmentation (GbS) [55] and SLIC [1] to our UnOLIS methods $\text{UnOLIS}_{\text{MS}}$, $\text{UnOLIS}_{\text{GbS}}$ and $\text{UnOLIS}_{\text{SLIC}}$, which use the unsupervised methods as pre-segmentation, correspondingly. In addition, we also compare our approach to two state-of-the-art supervised image segmentation methods gPb-OWT-UCM [6] and ISCRA [141]. Notice that for ISCRA only the released results of the 200 test images are available and used for comparison.

Evaluation metrics

Following Arbelaez et al. [6], we use the performance metrics segmentation covering, rand index (RI) and variational information (VI) for evaluation.

The covering of a segmentation S by a segmentation S' is defined as:

$$C(S' \rightarrow S) = \frac{1}{N} \sum_{R \in S} |R| \max_{R' \in S'} O(R, R') \tag{6.19}$$

where N denotes the total number of pixels in the image and $O(R, R')$ is the conventional overlap ratio between two regions R and R':

$$O(R, R') = \frac{|R \cap R'|}{|R \cup R'|}. \tag{6.20}$$

The covering of a computer segmentation S by a multiple ground truth $\{G_i\}$ is defined by first covering S separately with each human segmentation G_i and then averaging over the different humans.

The Rand Index (RI) between a computer segmentation S and the ground truth G is given by the sum of the number of pairs of pixels that have the same label (meaning belong to the same segmented region) in S and G and those that have different labels in S and G, divided by the total number of pairs of pixels. Given a set of ground truth segmentation G_k, the Probabilistic Rand Index (PRI) is used as the RI presented in this section.

$$PRI(S, \{G_k\}) = \frac{1}{T} \sum_{i<j} [c_{ij} p_{ij} + (1 - c_{ij})(1 - p_{ij})] \tag{6.21}$$

where c_{ij} stands for pixels i and j have the same label and p_{ij} is its probability. T is the total number of pixel pairs. Using the sample mean to estimate p_{ij}, the above equation amounts to averaging the Rand Index among different ground-truth segmentations.

The Variation of Information (VI) measures the distance between two segmentation according to their average conditional entropy.

$$VI(S, G) = H(S) + H(G) - 2I(S, G) \tag{6.22}$$

where H and I represent the entropies and mutual information between the computer segmentation S and ground truth G.

Higher covering and RI values indicate a better segmentation quality, while VI performs conversely. For all methods, there are parameters to be set during deployment. This includes the spatial and range kernel bandwidth for MS [34], the threhold parameter in GbS [55] and the number of expected superpixels in SLIC [1]. For each compared method, we use at least 20 different parameter settings to reduce their influence on the evaluation. While different parameter settings will result in different segmentation, we consider both an optimal parameter setting

per image (OIS) and an optimal parameter setting for the entire dataset (ODS). More specifically, OIS means choosing the best parameter setting for each image and ODS means using a fixed parameter for all images in the dataset which is calibrated to provide optimal performance on the training set.

Results

Table 6.1.: Results using multiple ground truths

	Covering		RI		VI	
	OIS	ODS	OIS	ODS	OIS	ODS
MS	0.4848	0.4065	0.7966	0.7629	2.4926	2.7624
GbS	0.5124	0.4371	0.7800	0.7416	2.098	2.3055
SLIC	0.4276	0.4068	0.7555	0.7352	2.7525	2.8010
$\text{UnOLIS}_{\text{MS}}$	**0.6173**	**0.5193**	0.8274	0.7479	1.6137	1.9848
$\text{UnOLIS}_{\text{GbS}}$	0.5744	0.4922	0.7675	0.7105	1.8146	2.1014
$\text{UnOLIS}_{\text{SLIC}}$	0.5525	0.4912	0.7881	0.7386	1.8885	2.1977
ISCRA	**0.6606**	**0.5923**	0.8491	0.7847	1.2001	1.3793
gPb-OWT-UCM	**0.6479**	**0.5907**	0.8545	0.8223	1.4839	1.6400

We first compare different segmentation methods using multiple ground truths in Table 6.1. In this case, the average of Covering, RI and VI of all ground truths is calculated. Obviously, our methods significantly outperform all traditional unsupervised methods except for the case RI (in red). The reason for the small performance degradation is a large variation among the ground truths. While some subjects tend to create coarse segmentation, others tend to catch more details and create finer segmentation. In comparison, our approach aims at an object-level segmentation, which is possibly regarded as under-segmentation by some detailed ground truths. This could lead to low RI scores, since RI tolerates under-segmentation less than the other metrics. The lower part of Table 6.1 shows the results of the state-of-the-art supervised methods. They outperform all unsupervised methods as expected, but our approach reduces considerably the performance gap between the unsupervised and supervised image segmentation methods.

In Table 6.2, the same experiment is repeated, but using a single ground truth, which is the coarsest one among all subjects (meaning the one with lowest

Table 6.2.: Results using the single ground truth

	Covering		RI		VI	
	OIS	ODS	OIS	ODS	OIS	ODS
$\text{UnOLIS}_{\text{MS}}$	**0.6914**	**0.5725**	0.8362	0.7351	1.3115	1.8068
$\text{UnOLIS}_{\text{GbS}}$	0.6508	0.5836	0.7824	0.7066	1.4653	1.8052
$\text{UnOLIS}_{\text{SLIC}}$	0.6310	0.5355	0.7911	0.7245	1.6018	1.9286
ISCRA	**0.6985**	**0.6208**	0.8491	0.7847	1.2001	1.3793
gPb-OWT-UCM	**0.6809**	**0.6146**	0.8365	0.7790	1.3232	1.4693

number of segmentation regions). In comparison to Table 6.1, the performance of our method is significantly improved due to the use of a single coarse object-level ground truth. Moreover, we see that our unsupervised methods achieve results which are comparable to state-of-the-art supervised methods.

Figure C.6 demonstrates some examples of unsupervised image segmentation. Worth to mention is that our method outputs roughly 13 segments per image in average with a standard deviation of 4.73, regardless of the utilized pre-segmentation methods and parameter settings. This result agrees quite well with the total number of objects in an image. On the contrary, traditional unsupervised methods generate unstable numbers of segments from a few to several hundreds depending on the parameter settings. Also, the segmentation accuracy experiences a big variation.

6.2. SGOP-SRR: Salient object segmentation with refinement

Salient object segmentation provides enhancement to traditional computer vision, computer graphics and visual communication technologies and covers a wide range of applications such as object recognition [148] and tracking [124], adaptive region-of-interest based image compression [134], content-aware image retrieval, adaptive content delivery and saliency-based image quality assessment [193]. The salient object segmentation methods often produce a score map (which is called a saliency map) and the scores are proportional to the probabilities of belonging to the salient object region. Hence, a thresholding process of the saliency

map can generate the salient object segmentation. This section addresses this salient object segmentation given no training data to learn statistical models.

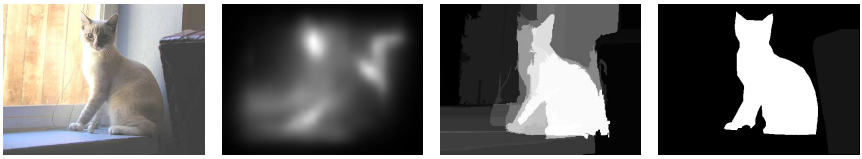
6.2.1. Ideas and overview

Salient object region is related to image features from different levels. Traditional methods predict salient areas in a bottom-up fashion and use low-level features. Itti et al. [86] proposed one of the oldest but effective model in this manner. They used hand-crafted features including intensity, color and orientation based on neuroscience principles. Their work inspired lots of research in this field which typically formulate salient region detection as a problem of image contrast analysis in a center-surround manner. Hence, they lack the capability of effectively modeling the perception of a whole object.

Meanwhile, high-level object information was proved to influence the human perception of visual saliency [48, 49]. Several works try to incorporate high-level features into saliency models. Cerf et al. [21] combined existing saliency map models with a face detector. Later, Borji and Ali [12] combined bottom-up features with more top-down object information from humans, faces, cars, texts, and animals. These methods achieve better performance than traditional models. However, the high-level information used by their models was generally gained by supervised learning strategies from a limited number of specific objects. Hence, they could fail to detect new categories of salient objects.

In this thesis, we propose a new method for integrating high-level information and bottom-up saliency models to generate a saliency map. Unlike the previous works which only consider the high-level information of several objects relying on supervised learning, we use a global search of object proposals [19] in an unsupervised manner. Since the object proposal generates category-independent object candidates, we naturally take arbitrary objects into consideration.

Saliency detection and object proposal are both related to visual attention. While saliency highlights the regions, which are more visually attractive, object proposals are assigned high scores if they are visually attended as objects. This inspired our idea of saliency-guided object proposal and refined salient region detection using the object proposals. Fig. 6.4 illustrates our approach with an ex-



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

Figure 6.4.: Overview of our approach: (a) Input image, (b) saliency map generated by a bottom-up model [73], (c) improved saliency map by our method, (d) ground truth

ample. Notice that the initial saliency map (Fig. 6.4(b)) generated by a bottom-up model includes noisy salient regions. The improved saliency map (Fig. 6.4(c)) suppresses the noisy salient regions and highlights the true salient object regions uniformly.

Two prior informations are used in our approach. The first one is a declarative knowledge (which is a kind of weak prior information) saying that the most salient areas in an image are often covered by some salient objects. This inspired the first stage of our approach (i.e. saliency-guided object proposal, see the next section for more details) which generates many salient object candidates. The second prior information contains two assumptions (a kind of prior information, cf. Chapter 2.2) according to observations on the salient object proposals from the first stage. First, the true salient object proposals strongly overlap with each other while the noisy proposals barely overlap with other proposals. Second, good salient object proposals have higher and smoother saliency values than the bad ones in general. These two assumptions (prior informations) are incorporated in the second stage of our approach (cf. Section 6.2).

In the following, we introduce our method in details in Section 6.2. In Section 6.2.3, we test our approach on two datasets and compare its performance with several other methods.

6.2.2. Proposed method

Our approach consists of a preprocessing stage and two main stages. Given an image, we generate the initial saliency map using a conventional bottom-up

model [73] and pre-segment the image into superpixels [55] which are used as the smallest units for generating object proposals. The two main stages are saliency-guided object proposal (SGOP) and salient region refinement (SRR). SGOP takes inputs from the initial saliency map and image superpixels to generate a pool of salient object proposals, which are then used in SRR for salient region refinement. In the following, we describe both main stages in details.

SGOP: Saliency-guided object proposal

While the bottom-up model provides local estimation of the saliency value for each pixel, the SGOP stage focuses on the global search of object-level information. Our goal in this stage is to generate a pool of salient object proposals. Notice that conventional object proposal methods (e.g. [19]) try to generate all possible object regions in order to ensure a high recall for object detection, while we deal with salient region detection where only salient objects are important.

Each of our salient object proposal is generated by a binary segmentation of an image. We use the graph cut algorithm [98] for this purpose because it provides a global optimal solution. Notice that graph cut is primarily designed for interactive segmentation [14] which requires seed placement and the parameter settings in the algorithm also influence the segmentation. We vary the seeds and parameter settings to generate a pool of salient object proposals. Below, we describe how to generate a single proposal and then how to vary the seeds and parameters to obtain a pool of them.

Graph representation Given a superpixel segmentation of an image and a superpixel seed S_o for salient object and a background seed $\mathcal{S}_b$, we construct a graph $\mathcal{G} = (V, E)$ where S_o, $\mathcal{S}_b$ and the remaining superpixels are represented by the vertices v_o, v_b and v_i, respectively. Notice that the background seed $\mathcal{S}_b$ is a region contains a set of superpixels (see the section about seed placement and parameter setting in 6.2). The vertices v_o and v_b corresponding to superpixels are called terminals in the graph. Hence, the edges from v_o to v_i and from v_i to v_b are called t-links (link to terminals). The edges between two internal adjacent vertices v_i and v_j (neither v_o or v_b) are called n-links (link between neighbouring superpixels). Two vertices v_i and v_j are adjacent if the corresponding superpixels

s_i and s_j contain pixels that are direct neighbours. We use 4-connected neighbourhood and a pixel has 2, 3 or 4 direct neighbours, if it is located at the image corner, image boundary or none of them, respectively. Each link has an assigned weight describing the similarity between the two connected vertices.

Graph cut The task is to find a binary segmentation of the image as a salient object proposal by assigning a label $l_i = 1$ (salient object) or $l_i = 0$ (background) to each super-pixel s_i. Our objective is to obtain an optimal labeling $\underline{l} = \{l_1, ..., l_M\}$ for all M superpixels in terms of an energy function

$$E(\underline{l}) = \sum_i \Upsilon(s_i, l_i) + \beta \sum_{i,j \in \eta} \Phi(s_i, s_j, l_i, l_j). \tag{6.23}$$

η is the set of pairs of adjacent superpixels, Υ and Φ are unary and pairwise energy defined later and β is a trade-off parameter between these two energy terms.

The *unary energy* is also noted as a data term. It models the weights of t-links, indicating the cost of assigning the label l_i to the superpixel s_i:

$$\Upsilon(s_i, l_i) = \begin{cases} f_{sim}(s_i, \mathcal{S}_b): & l_i = 1 \\ \epsilon f_{sim}(s_i, S_o): & l_i = 0 \end{cases} \tag{6.24}$$

This function shows that the weights of t-links are determined by a function $0 \leq f_{sim} \leq 1$ measuring the similarity between s_i and S_o or $\mathcal{S}_b$. We use one group of 150 texture features [171] and one group of 150 color features [170], all normalized to $[0, 1]$, to compute the similarity f_{sim} between two image regions (superpixel or superpixel set) s_i and s_j:

$$f_{sim}(s_i, s_j) = 1 - \frac{1}{w_t + w_c} \left[w_t d_t(s_i, s_j) + w_c d_c(s_i, s_j) \right]. \tag{6.25}$$

w_t and w_c are two weighting parameters, both set to 1 in our experiments. d_t and d_c are the chi-square distances of the texture and color feature histogram, respectively. Given two feature histograms $\mathbf{x} = [x_1, .., x_n]$ and $\mathbf{y} = [y_1, ..., y_n]$ each having n bins. The chi-square distance between the two histograms is defined

as

$$d(\mathbf{x}, \mathbf{y}) = \frac{1}{2} \sum_{i=1}^{n} \frac{(x_i - y_i)^2}{(x_i + y_i)} \tag{6.26}$$

In our case, $\mathbf{x}$ and $\mathbf{y}$ are either both texture features or both color features.

ϵ is a parameter which controls the penalty associated to labelling superpixels similar to the salient object seed S_o as background. A large value of ϵ reduces the probability of assigning a superpixel to the background and increases the size of the salient object proposal.

The *pairwise energy* Φ is also known as a smoothness term. It models the weights of n-links and is defined by

$$\Phi(s_i, s_j, l_i, l_j) = \mathbf{1}(i, j \in \eta, l_i \neq l_j) f_{sim}(s_i, s_j). \tag{6.27}$$

It penalizes the assignment of different labels to similar adjacent superpixels.

In summary, the above problem is a combinatorial optimization problem. Kolmogorov and Zabin [98] showed that minimizing the energy function in Eq. (6.23) is equivalent to find the minimum cut in the corresponding graph, which can be efficiently solved by the max-flow/min-cut algorithm [13].

Seed placement and parameter setting For a particular choice of S_o, $\mathcal{S}_b$, α, λ, we create a unique global optimal binary image segmentation as a single salient object proposal. By varying these parameters, we generate a pool of salient object proposals.

The salient object seeds are determined by finding superpixels with a maximum saliency value over a threshold, which is adaptively set to be twice the mean of the initial saliency map. This results in a pool of salient superpixels. We vary S_o over each of them.

The background seeds are determined by the superpixels on the image boundaries. However, instead of simply choosing $\mathcal{S}_b$ as the set of superpixels along all 4 image boundaries (left, right, top, bottom), we consider different combinations of them. We choose the seed $\mathcal{S}_b$ to contain superpixels along one of the following image boundary combinations: only left, only right, only top, left and right, left and top, right and top, left and right and top, all 4 image boundaries.

The superpixels along the bottom image boundary are taken only once (in the combination of all 4 image boundaries) because we found that they are often contained in a salient object. This strategy of background seed increases the diversity of the salient object proposals and reduces their dependence on the background assumptions. Notice that we remove a superpixel from $\mathcal{S}_b$ if it is set as S_o.

We vary β from 0.1 to 0.9 with a step size of 0.2 and from 1 to 5 with a step size of 1 and ϵ from 0.5 to 3 with a step size of 0.25 to produce a more diverse set of proposals.

Post-processing The salient object proposals generated above can be multiplicated. We remove duplicated ones and keep each proposal unique.

SRR: Salient region refinement

The initial saliency map generated by a conventional bottom-up saliency model is sensitive to high frequency background noise. In this stage, we aim at refining the saliency map based on two properties of the salient object proposals from SGOP. First, the object proposals generated by true salient superpixels strongly overlap while those generated by noisy salient superpixels rarely overlap with other proposals. This inspired a voting-like strategy to remove noise and refine the saliency map. Second, good salient object proposals have higher and smoother saliency values than noisy proposals. This motivated the use of both the saliency value and its smoothness for the SRR process.

Let $\mathbf{S} \in \mathbb{R}^{H \times W}$ be the initial saliency map generated by a conventional saliency model (e.g. [73]). $H \times W$ is the spatial size. $S(x, y) \in [0, 1]$ is the saliency value at pixel (x, y). Here, (x, y) is the spatial coordinates. Let $\mathbf{P}_i \in \mathbb{R}^{H \times W}$ be the i-th salient object proposal from SGOP stage. $P_i(x, y) \in \{0, 1\}$ is the binary indicator at pixel (x, y). $P_i(x, y) = 1$ and $P_i(x, y) = 0$ indicate that the pixel (x, y) belongs to the i-th proposal and background, respectively.

We define $\mathbf{S}_i \in \mathbb{R}^{H \times W}$ as the initial saliency map for the region of the i-th proposal.

$$\mathbf{S}_i = \mathbf{S} \circ \mathbf{P}_i. \tag{6.28}$$

Here, $\circ$ indicates component-wise multiplication of two matrices. Then, we calculate the mean μ_i and variance σ_i^2 of $\mathbf{S}_i$. σ_i^2 reflects the smoothness and quality of $\mathbf{S}_i$. The average saliency map $\bar{\mathbf{S}}_i$ for the image derived from the i-th proposal is defined as

$$\bar{S}_i(x,y) = \begin{cases} \mu_i & P_i(x,y) = 1 \\ 0 & P_i(x,y) = 0 \end{cases} . \tag{6.29}$$

In addition, we calculate the saliency coverage c_i of the i-th proposal by

$$c_i = \frac{\sum_{x,y} S_i(x,y)}{\sum_{x,y} S(x,y)}. \tag{6.30}$$

The refined saliency map $\mathbf{S}'$ is then calculated by a voting-like strategy based on all salient object proposals.

$$\begin{aligned} \tilde{S}(x,y) &= \sum_i c_i e^{-\sigma_i^2} \bar{S}_i(x,y), \\ S'(x,y) &= \frac{\tilde{S}(x,y)}{\max_{(x,y)} \tilde{S}(x,y)}, \end{aligned} \tag{6.31}$$

Each saliency map of a object proposal contribute to the final saliency map with a weighting factor. Smoother (i.e. smaller σ_i^2) and with a higher saliency coverage (i.e. larger c_i) the saliency map is, the corresponding object proposal contribute more to the refined saliency map.

6.2.3. Experiments

In this section, we select two benchmark databases to evaluate the performance of our proposed method.

MSRA-1000 MSRA-1000 [2] includes 1000 images sampled from the MSRA Salient Object Database [116]. This database is widely used for saliency detection. It covers a large variety of image contents, but most of the images include only one salient object centered in the image and with high contrast to the background.

This allows several methods [183, 197, 27] to achieve good performance by using these information as priors.

PASCAL-S PASCAL-S [112] is a new dataset developed for saliency detection benchmarking. It contains 850 natural images with multiple complex objects and cluttered backgrounds. This dataset is one of the most challenging saliency datasets because the salient objects can be located in the image center or on the image boundaries. The average contrast between salient objects and background is smaller than MSRA.

Both datasets provide ground truth saliency maps.

To evaluate the quality of the salient region detection, we use the precision-recall (PR) curve and the F-measure as introduced in Chapter 3.5.2. We use the threshold from 0 to 255 to generate 256 precision and recall pairs. The PR curve demonstrates the precision and recall averaged over all images. For each pair of precision-recall, we get a single F_γ. The largest F_γ is selected as the performance measure. Following the suggestion in [2], we set $\gamma^2 = 0.3$ to emphasize the importance of precision.

Results

In this section, we compare our approach with 5 recent models for salient region detection FT [2], GC [27], MR [188], GS [183], wCtr [197]. As baselines, we also show the results achieved by two bottom-up saliency models Itti [86], GB [73]. We use either the pre-computed saliency maps or codes provided by other authors to generate their saliency maps. For our approch, we use the bottom-up saliency model [73] to generate the initial saliency map and use [55] to generate the superpixels because they are computationally efficient and with publicly available code in Internet. Notice that several other recent works also present excellent results [113, 180, 194, 106]. However, they use part of the datasets for supervised training. The models based on deep neural networks [180, 194, 106] rely on very large datasets for deep learning. Therefore, these methods are not compared in this paper due to a fairness in consideration.

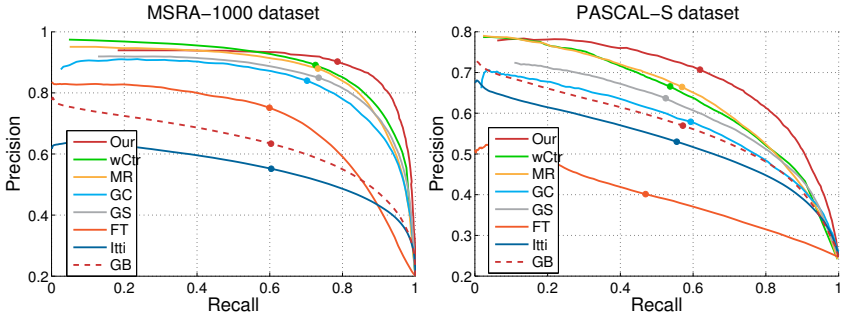


Figure 6.5.: PR curves of saliency detection methods on two datasets

The PR curves are shown in Fig. 6.5 and the F-measure scores are given in Table 6.3.

Table 6.3.: F-measures of the benchmarking methods on two datasets

Data Set	Itti	GB	FT	GC	wCtr	GS	MR	Our
MSRA1000	0.5629	0.6265	0.7097	0.8037	0.8468	0.8200	0.8406	**0.8627**
PASCAL-S	0.5357	0.5702	0.4154	0.5824	0.6379	0.6169	0.6425	**0.6846**

Our method outperforms all other methods. Notice that MR and wCtr provide a higher precision than our method in the range of low recall for the MSRA-1000 dataset. Nevertheless, we achieve a higher F-measure than them. For the more challenging PASCAL-S dataset, our approach clearly outperforms other methods by a large margin.

Fig. C.7 shows several selected examples of challenging situations. The saliency maps generated by our method agrees well with the ground truth.

Discussion

We propose a new unsupervised approach to combine bottom-up saliency model with global object segmentation. By using saliency-guided object proposals, we update the saliency map by a refinement process.

Our method has two main strengths. First, the saliency guides the object candidate segmentation and thus restricts the search area of objects. This avoids search over entire image and saves computation cost. Second, while the bottom-up saliency detection usually generates noisy salient regions due to the use of low-level local features, the global property of the object proposals helps to reduce the noise.

The final saliency map shows a higher accuracy in the sense of reduced noisy saliency in the background region and homogeneous, uniform salient regions within objects. Our approach outperforms other unsupervised methods and achieves the state-of-the-art performance for two benchmarking datasets. Since our approach relies on the preprocessing steps of superpixel segmentation and an initial saliency map, even better performance is expected by further improvement of the preprocessing steps.

6.3. Summary

In this chapter, two examples are introduced to show the success of incorporating prior knowledge by hand-designing features for object-level image segmentation. Both methods rely on no training dataset and are unsupervised.

In the first example, we present a novel post-processing approach for unsupervised object-level image segmentation (UnOLIS). Starting with the results of any conventional unsupervised segmentation method, we first combine a global region-based saliency and a robust background feature to cluster the pre-segmented regions into foreground and background. We then design a region growing process, encoded with several object priors, to generate a high quality foreground object segmentation. In parallel, we group the background regions into different objects by clustering. We test our method on the Berkeley Segmentation Dataset (BSDS500). Our approach significantly improves conventional unsupervised segmentation methods and achieves almost comparable results as the state-of-the-art supervised image segmentation methods.

In the second example, a method is proposed to incorporate the object-level information based on the bottom-up saliency model. By using saliency-guided object proposals, we implicitly remove the noisy salient regions and produce a refined

saliency map. Experiments show that this method boosted the performance of bottom-up saliency models and performs favorably against other unsupervised salient object segmentation methods.

Chapter 7.

Conclusion and Future Work

7.1. Summary

The ambiguity of the general image segmentation task is due to unsatisfied specification of the target application. This thesis deals with the task of segmenting natural images into regions corresponding to human perception of real world objects. This is called object-level image segmentation.

The implications of object-level image segmentation can be summarized from two sides. First, in contrast to traditional image segmentation, object-level image segmentation emphasizes that the segmented regions shall be semantically meaningful and thus, provides image representations which are closer to the perceptual goal. Second, the definition of object-level image segmentation encapsulates many application-specific segmentation tasks such as semantic image segmentation, salient object segmentation and generic foreground/background segmentation. Object-level image segmentation with prior information provides a technical guidance to deal with different image segmentation tasks.

In this thesis, context- and application-specific information is denoted as prior information and is incorporated into the segmentation algorithms. In particular, this thesis handles the object-level image segmentation from the machine learning perspective. Several novel methods (either supervised or unsupervised) are proposed to deal with different segmentation tasks, yet, being aligned with the general concept introduced in Chapter 2.

Deep learning is highlighted in this thesis. As long as a set of training images and the pixel-wise reference information are usable, deep CNN is selected to solve

the object-level image segmentation problem. In Chapter 3, we propose a deep convolutional neural network architecture for object-level image segmentation. The network consists of a sequence of convolutional, normalization, activation and pooling layers. Skip connections (residual shortcuts) can be flexibly added between layers to improve the training performance and segmentation accuracy. We examine the strengths of the proposed method using two application examples, i.e., semantic image segmentation and salient object segmentation, and also study its drawbacks such as context-sensitivity and requirements on data.

In Chapter 4, we focus on the context-sensitivity problem. We examine the context-awareness with a comprehensive case study. The results show that the CNN model trained with pixel-wise reference information and in a supervised manner is sensitive to many context changes. Then, we propose context-changing data augmentation as a straightforward method to train context-insensitive models. In this method, the training data (containing both images and ground truth segmentation) and context transformation techniques are used as prior information. This prior information is incorporated into the generation of new images similar to the target context. The proposed method enlarges the source context for training and trains models which are insensitive to the context changes known in advance.

In Chapter 5, class extension methods are proposed to reuse models and alleviate the requirements on data. In particular, the basic class extension method reuses the old model by copying its parameters to initialize the new model and using its output probability map to generate the pixel-wise reference information. To train the new model, only a small number (<500) of training images are required and only the new object class needs to be manually annotated. Given an additional binary segmentation model as prior information, the BMACE method is proposed to fuse the old and the binary segmentation model and to train the new model. The prior information is incorporated during training by using a non-standard loss function with a regularization term. In this case, even the manually annotation of the new object class is avoided.

Complementary to the deep learning methods, Chapter 6 focuses on unsupervised computational methods. No image dataset and ground truth segmentation are given as prior information and thus, it is not able to train a model with a

large number of parameters. Instead, declarative prior knowledge and assumptions are taken as prior information. We propose hand-crafted features based on such weak prior information to achieve the object-level segmentation goal. Two novel methods, UnOLIS (unsupervised object-level image segmentation) and SGOP-SRR (saliency-guided object proposal for salient region refinement), are proposed. The prior knowledge with object-level information is incorporated into the process by designing segmentation features. More specifically, the UnOLIS method combines a background prior with visual saliency and perceptual organization principles to generate an unsupervised object-level image segmentation. The SGOP-SRR method combines a bottom-up saliency model with global object segmentation. It updates the saliency map by a refinement process based on saliency-guided object proposals.

7.2. Limitations and future works

In this dissertation, we present a general concept of object-level image segmentation with prior information, which contains many segmentation tasks as special cases. The idea is to handle different application cases by incorporating the prior information. Two branches of methods, the learning-based methods and the unsupervised computational methods, can be exploited, depending on that if the prior information contains training data. Yet, the algorithms and methods proposed in Chapters 3, 4, 5 and 6 are dedicated to the application cases and independent to each other. A common algorithmic framework containing all the proposed algorithms as special cases would be nice. A future work would be studying the feasibility of such a general formulation of the methods and its application to different cases.

Another limitation of the methodology is that we focus on using deep neural network methods for the learning-based branch, because deep learning is the state-of-the-art machine learning method. It makes breakthroughs in many computer vision tasks and has been widely applied in recent years. However, it is worth noticing that there are other classic machine learning algorithms which can be applied. It is not discussed in this thesis how these algorithms can be adapted to our concept and what their advantages and disadvantages are compared to deep CNN. These can be researched in the future.

Moreover, we focus on using convolutional neural networks because this type of neural networks has been proved to be suitable to handle many high-level image processing tasks such as image classification [101, 157, 162, 79] and object detection [140, 65, 63]. Yet, there are other types of deep neural networks, such as Capsule Network [149], Recurrent Nerual Network (RNN) [173, 109] and the combination of CNN and RNN [25]. Study and evaluation of these neural networks for object-level image segmentation is another future work.

Chapter 3 proposed a deep CNN architecture for object-level image segmentation. Two application examples are presented and experiments are conducted which show good segmentation results. By examining the several activation maps of the convolutional layers, we illustrate that the learned features are meaningful. Shallow layers learn low-level features and deep layers learn global and object-related features. However, the networks are treated as black boxes during training. They are trained with a large number of images and the corresponding ground truth segmentation and the parameters in the network are optimized by back-propagation according to the loss function. The plausibility of the learned features are not traceable. Many visualization methods [190, 128] are proposed to exploit the interior of the deep networks. They help us to gain valuable insights into CNN models but do not change the way of training. In the future, it is worth delving into explainable deep learning strategy for object-level image segmentation.

To solve the context-sensitivity problem of a deep CNN model, Chapter 4 proposes context adaptation by context-changing data-augmentation. Experiments show the success of training models which are insensitive to several additional contexts. This is, however, limited to the availability of effective image manipulation techniques to enlarge the context. Moreover, the context insensitivity of the models is limited to the context changes they have ever "seen". Training models which are insensitive to all context changes in the supervised manner is challenging and even impossible because the real world context changes are incessant. However, it is worth exploring methods to train a model for a single class segmentation and being applicable to multiple contexts (as discussed in Chapter 4.4). This is regarded as a future work.

In Chapter 5, class-extension is proposed to reuse old CNN models and alleviate the data requirement. Experimental results show that the proposed BMACE

method is effective to fuse the old and the binary segmentation model for class extension. Yet, the application of the BMACE method relies on an effective binary segmentation model. Moreover, the work of this chapter only focuses on the semantic image segmentation tasks. A more general class-extension method is worth of a further study.

A limitation to our unsupervised object-level image segmentation methods proposed in Chapter 6 is that they strongly rely on hand-designed features. The effectiveness of the features is the crucial point to the effectiveness of the methods. However, designing feature is challenging and depends on experience. The effectiveness is not guaranteed. Although the pixel-wise reference information is costly, a collection of training images is in nowadays, less expensive as before. It is a future work to explore the feasibility of applying unsupervised learning method for object-level image segmentation. This is a rather difficult task because it requires the algorithms to learn object classes by themselves.

Appendix A.

Mathematical Description

A.1. Region interpolation

The Laplace's equation is given as follows

$$\Delta u = \nabla^2 u = u_{xx} + u_{yy} = 0. \tag{A.1}$$

It is a second-order partial differential equation (PDE). $u(x, y)$ is a scaler function which interprets the pixel value of a channel (R or G or B) of an image w.r.t. the spatial coordinates x and y in the column and row direction, respectively. Δ is the Laplace operator. $\Delta = \nabla^2$. It indicates the second-order partial differential operation.

$$u_{xx} = \frac{\partial^2 u}{\partial x^2} \tag{A.2}$$

$$u_{yy} = \frac{\partial^2 u}{\partial y^2} \tag{A.3}$$

For a given PDE in domain $x \in [a, b]$, we define

$$\triangle x = \frac{b - a}{N}, \tag{A.4}$$

where N is the width of the image (i.e. the number of columns). The derivatives of "u" leas to N equations for u_i, $i = 1, 2, 3, \ldots, N$.

$$u_i \equiv u(i\triangle x) \tag{A.5}$$

$$x_i = i\triangle x \tag{A.6}$$

$$y_i = i\triangle y \tag{A.7}$$

We approximate the second partial derivatives in the PDE suing the central difference scheme, i.e.

$$\frac{\partial^2 u}{\partial x^2}_{i,j} \approx \frac{u_{i-1,j} - 2u_{i,j} + u_{i+1,j}}{(\triangle x)^2} \tag{A.8}$$

$$\frac{\partial^2 u}{\partial y^2}_{i,j} \approx \frac{u_{i,j-1} - 2u_{i,j} + u_{i,j+1}}{(\triangle y)^2} \tag{A.9}$$

Putting the double partial derivatives in the Laplace's equation, we have

$$\frac{u_{i-1,j} - 2u_{i,j} + u_{i+1,j}}{(\triangle x)^2} + \frac{u_{i,j-1} - 2u_{i,j} + u_{i,j+1}}{(\triangle y)^2} = 0 \tag{A.10}$$

for pixel (i, j). Since the pixels are spaced apart equally in both spatial dimensions, we have $\triangle x = \triangle y$. The equation can further be rearranged as

$$u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1} - 4u_{i,j} = 0 \tag{A.11}$$

for pixel (i, j). This is a discretized form of the Laplace equation.

The region marked by a binary mask serves as the hole which is supposed to be filled by the diffused surrounding pixels. The surronding pixels have fixed pixel values. They serve as the as the Dirichlet boundary conditions. As shown in Eq. (A.11), the intensity of each pixel (i, j) in the hole region is governed by the intensity of pixels on the immediate left, right, top and bottom. Using the boundary conditions, we proceed with the discretized Laplace equation for processing of the subsequent pixels. This interpolates inward from the pixel values on the outer boundary of the regions until the region is completely filled.

A.2. Color transform

A color in the RGB space consists of red, green and blue intensities. A color in the CIE-Lab space consists of lightness L and two color-opponent dimensions a and b. The transformation of a color in RGB space to CIE-Lab requires a reference white point (X_r, Y_r, Z_r). The standard white point is the CIE standard illuminant D65, which simulates noon daylight with correlated color temperature of 6504 K.

Two steps are needed for the transformation.

A.2.1. RGB to XYZ

First, the companded RGB channels (denoted with upper case (R, G, B), or generically V) are made linear with respect to energy (denoted with lower case (r, g, b) or generically v).

The components of (R, G, B) are in the nominal range $[0, 1]$. They are converted to XYZ by the following computation:

$$v = \begin{cases} \frac{V}{12.92} & \text{if} \quad V \leq 0.04045 \\ \left(\frac{V+0.055}{1.055}\right)^{2.4} & \text{otherwise} \end{cases} \tag{A.12}$$

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = [M] \begin{bmatrix} r \\ g \\ b \end{bmatrix} \tag{A.13}$$

The transformation matrix $[M]$ is calculated from the RGB reference primaries and is given as

$$[M] = \begin{bmatrix} 0.412453 & 0.357580 & 0.180423 \\ 0.212671 & 0.715160 & 0.072169 \\ 0.019334 & 0.119193 & 0.950227 \end{bmatrix}. \tag{A.14}$$

A.2.2. XYZ to Lab

This conversion conversion requires the reference white which is fixed at $(X_r, Y_r, Z_r) = (0.9642, 1, 0.8249)$. The three components of the CIE-Lab color channels are computed as follows:

$$L = 116 f_y - 16 \tag{A.15}$$

$$a = 500(f_x - f_y) \tag{A.16}$$

$$b = 200(f_y - f_z) \tag{A.17}$$

where

$$f_x = \begin{cases} \sqrt[3]{x_r} & if\, x_r > \epsilon \\ \frac{\kappa x_r + 16}{116} & otherwise \end{cases} \tag{A.18}$$

$$f_y = \begin{cases} \sqrt[3]{y_r} & if\, y_r > \epsilon \\ \frac{\kappa y_r + 16}{116} & otherwise \end{cases} \tag{A.19}$$

$$f_z = \begin{cases} \sqrt[3]{z_r} & if\, z_r > \epsilon \\ \frac{\kappa z_r + 16}{116} & otherwise \end{cases} \tag{A.20}$$

$$x_r = \frac{X}{X_r} \tag{A.21}$$

$$y_r = \frac{Y}{Y_r} \tag{A.22}$$

$$z_r = \frac{Z}{Z_r} \tag{A.23}$$

$\epsilon = 0.008856$ and $\kappa = 903.3$.

Appendix B.

CNN Structure

Table B.1.: Configuration of the semantic image segmentation network (For simplicity, only convolutional layers and pooling/unpooling layers are shown.)

Type	Layers	Filter size	Stride	Feature dimension
	Input	–	–	$224 \times 224 \times 3$
Encoder−1	Conv ×3	$3 \times 3 \times 64$	1	$224 \times 224 \times 64$
	Pool−1	2×2	2	$112 \times 112 \times 64$
Encoder−2	Conv ×3	$3 \times 3 \times 128$	1	$112 \times 112 \times 128$
	Pool−2	2×2	2	$56 \times 56 \times 128$
Encoder−3	Conv ×3	$3 \times 3 \times 256$	1	$56 \times 56 \times 256$
	Pool−3	2×2	2	$28 \times 28 \times 256$
Encoder−4	Conv ×3	$3 \times 3 \times 512$	1	$28 \times 28 \times 512$
	Pool−4	2×2	2	$14 \times 14 \times 512$
Transition	Conv ×3	$3 \times 3 \times 1024$	1	$14 \times 14 \times 1024$
Decoder−1	Conv ×3	$3 \times 3 \times 512$	1	$14 \times 14 \times 512$
	UnPool−1	2×2	2	$28 \times 28 \times 512$
Decoder−2	Conv ×3	$3 \times 3 \times 256$	1	$28 \times 28 \times 256$
	UnPool−2	2×2	2	$56 \times 56 \times 256$
Decoder−3	Conv ×3	$3 \times 3 \times 128$	1	$56 \times 56 \times 128$
	UnPool−3	2×2	2	$112 \times 112 \times 128$
Decoder−4	Conv ×3	$3 \times 3 \times 64$	1	$112 \times 112 \times 64$
	UnPool−4	2×2	2	$224 \times 224 \times 64$
	Conv ×2	$3 \times 3 \times 64$	1	$224 \times 224 \times 64$
Classifer	Conv ×1	$3 \times 3 \times C$	1	$224 \times 224 \times C$
	Softmax	–	–	$224 \times 224 \times C$

Table B.2.: Configuration of the salient object segmentation network

Layer identifier	Padding	Filter size	Stride	Feature dimension
data	-	-	-	$500 \times 500 \times 3$
conv1-1	100	$3 \times 3 \times 64$	1	$698 \times 698 \times 64$
ReLU1-1	-	-	-	-
conv1-2	1	$3 \times 3 \times 64$	1	$698 \times 698 \times 64$
ReLU1-2	-	-	-	-
pool1	1	2×2	2	$349 \times 349 \times 64$
conv2-1	1	$3 \times 3 \times 64$	1	$349 \times 349 \times 128$
ReLU2-1	-	-	-	-
conv2-2	1	$3 \times 3 \times 64$	1	$349 \times 349 \times 128$
ReLU2-2	-	-	-	-
pool2	1	2×2	2	$175 \times 175 \times 128$
conv3-1	1	$3 \times 3 \times 256$	1	$175 \times 175 \times 256$
ReLU3-1	-	-	-	-
conv3-2	1	$3 \times 3 \times 256$	1	$175 \times 175 \times 256$
ReLU3-2	-	-	-	-
conv3-3	1	$3 \times 3 \times 256$	1	$175 \times 175 \times 256$
ReLU3-3	-	-	-	-
pool3	1	2×2	2	$88 \times 88 \times 256$
conv4-1	1	$3 \times 3 \times 512$	1	$88 \times 88 \times 512$
ReLU4-1	-	-	-	-
conv4-2	1	$3 \times 3 \times 512$	1	$88 \times 88 \times 512$
ReLU4-2	-	-	-	-
conv4-3	1	$3 \times 3 \times 512$	1	$88 \times 88 \times 512$
ReLU4-3	-	-	-	-
pool4	1	2×2	2	$44 \times 44 \times 512$
conv5-1	1	$3 \times 3 \times 512$	1	$44 \times 44 \times 512$
ReLU5-1	-	-	-	-
conv5-2	1	$3 \times 3 \times 512$	1	$44 \times 44 \times 512$
ReLU5-2	-	-	-	-
conv5-3	1	$3 \times 3 \times 512$	1	$44 \times 44 \times 512$
ReLU5-3	-	-	-	-
pool5	1	2×2	2	$22 \times 22 \times 512$
conv6	1	$7 \times 7 \times 4096$	1	$16 \times 16 \times 4096$
ReLU6	-	-	-	-
Drop6	-	-	-	-
conv7	1	$7 \times 7 \times 4096$	1	$16 \times 16 \times 4096$
ReLU7	-	-	-	-
Drop7	-	-	-	-
score	1	$1 \times 1 \times 1$	1	$16 \times 16 \times 1$
deconv	14	$63 \times 63 \times 4096$	31	$500 \times 500 \times 1$

Appendix C.

Segmentation Examples

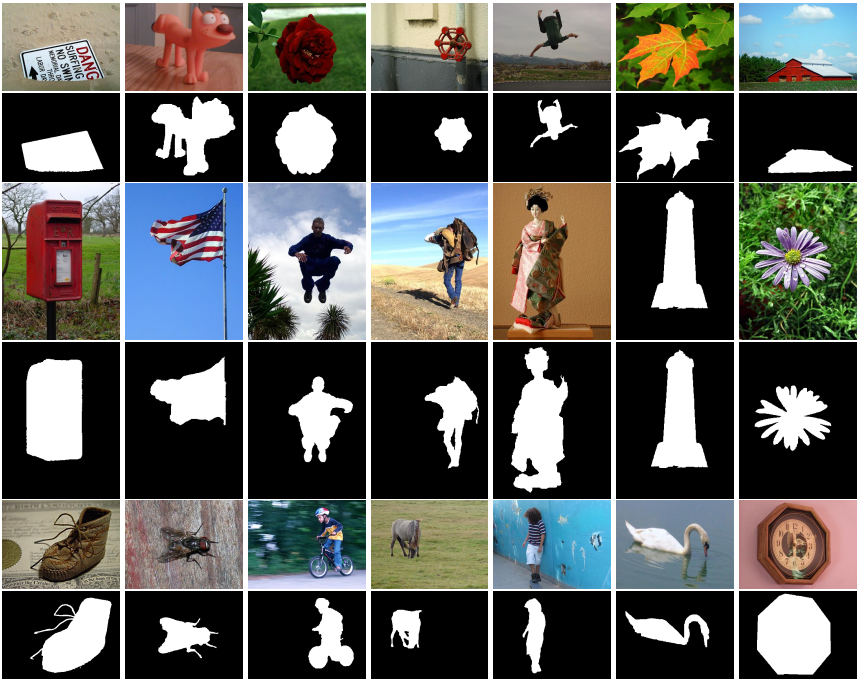


Figure C.1.: Randomly selected images and their corresponding ground truth salient object segmentation from MSRA10K dataset

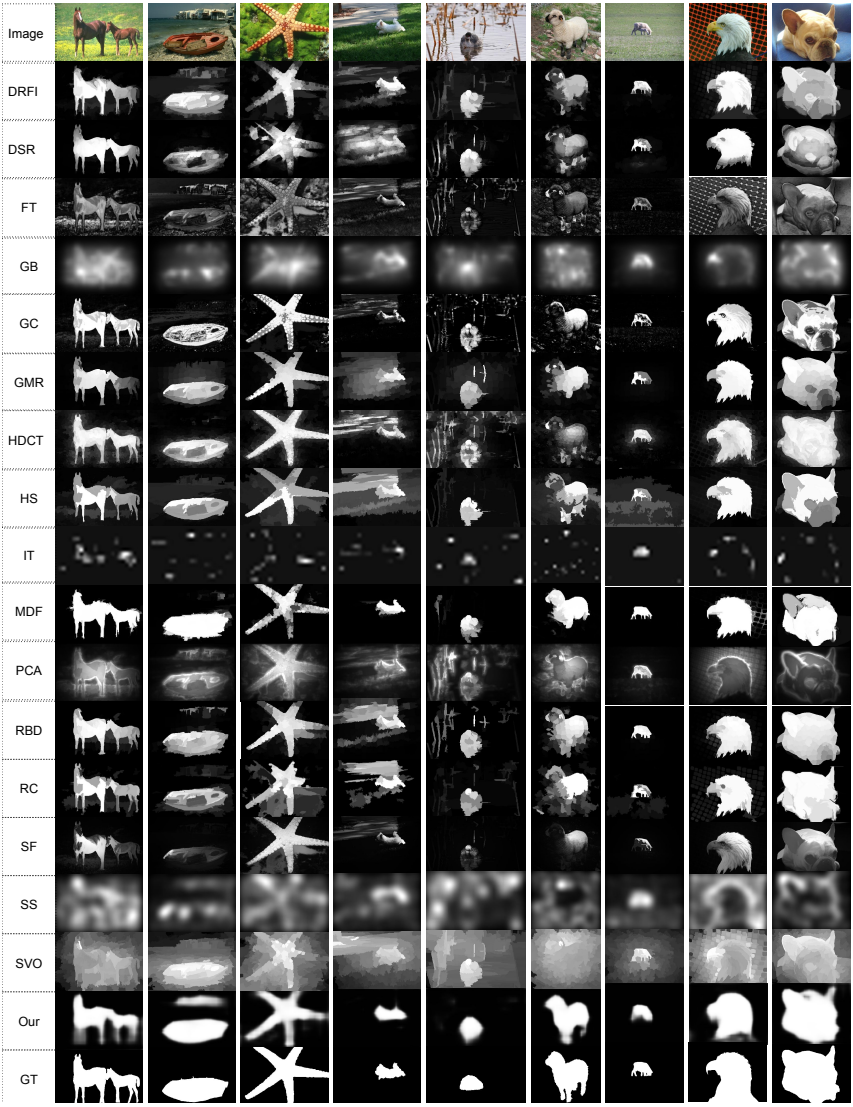


Figure C.2.: Saliency maps. Row 1: selected images from ECSSD dataset. Row 2 - 18: results of our salient object segmentation CNN and compared with other 16 approaches. Last row: ground truth segmentation.

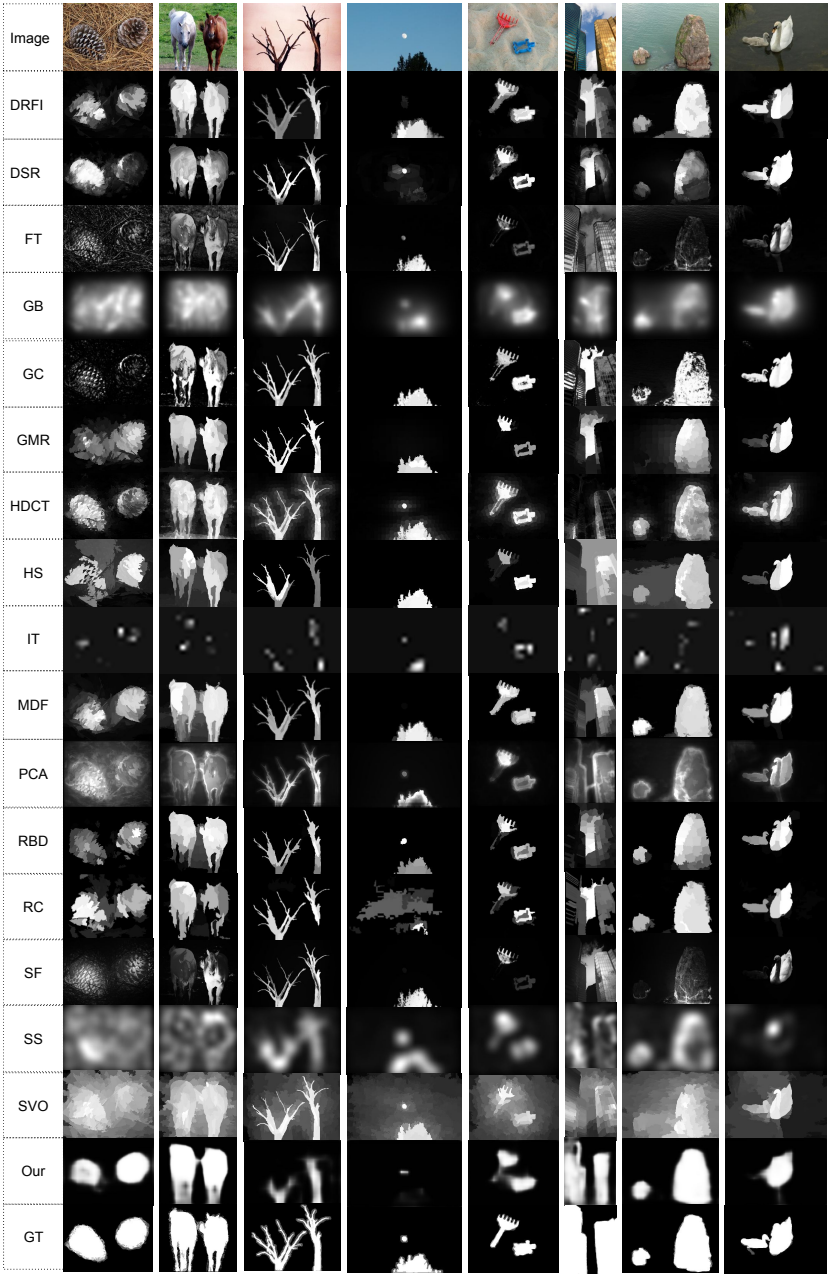


Figure C.3.: Saliency maps. Row 1: selected images from SED2 dataset. Row 2 - 18: results of our salient object segmentation CNN and compared with other 16 approaches. Last row: ground truth segmentation.

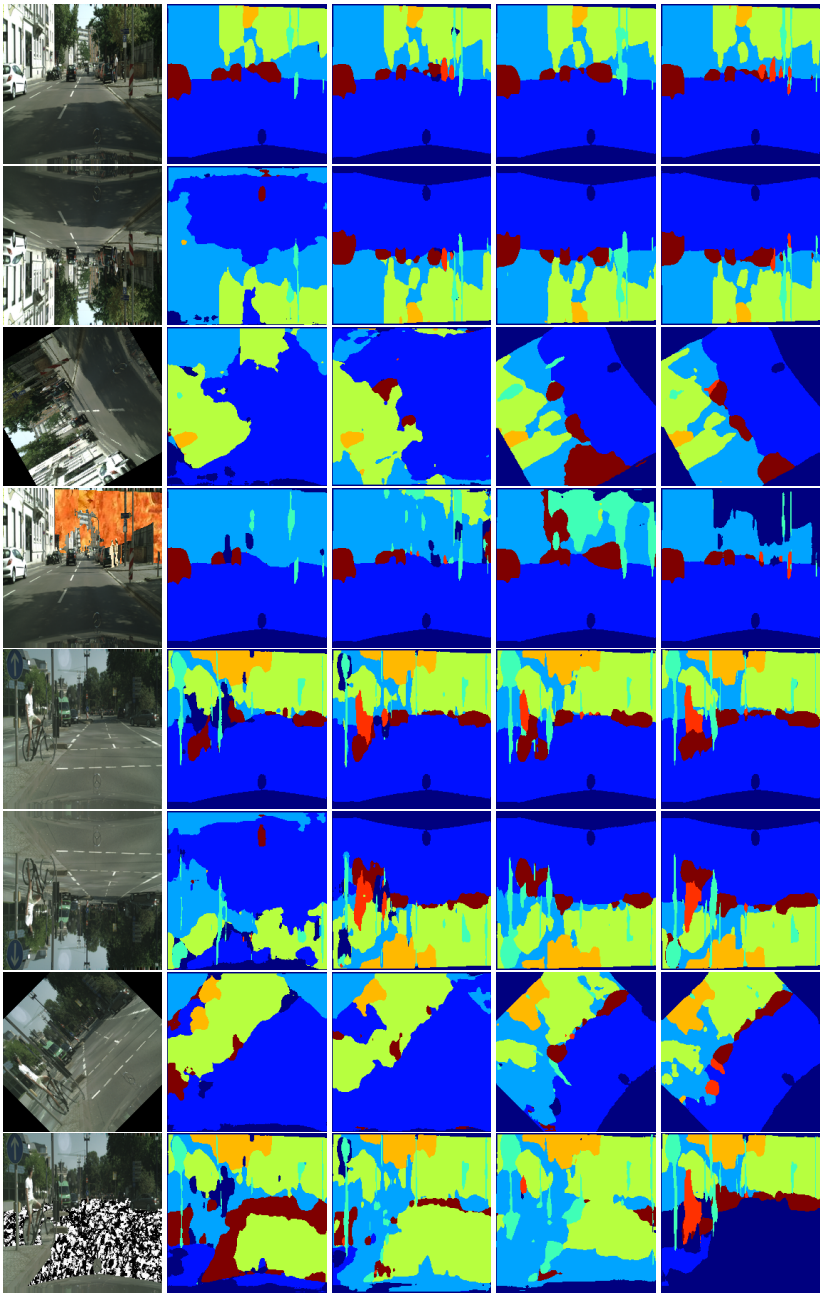


Figure C.4.: Examples of segmentation results for context-changing images. First column shows the images with context changes and the second to fifth the segmentation results of $\mathbf{M}_0$, $\mathbf{M}_1$, $\mathbf{M}_2$ and $\mathbf{M}_3$, respectively.

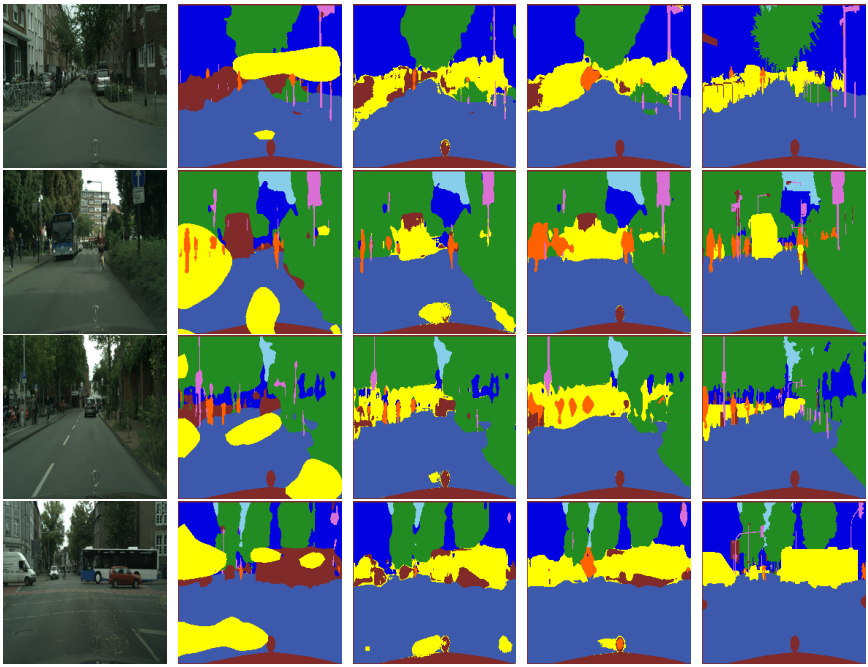


Figure C.5.: Examples of the segmentation results using the external binary segmentation model. From left to right: input image, direct decision fusion (DF), class extended model (HTF), class extended model (BMACE) and ground truth. The new class to be extended is expected to be VEHICLE in yellow.

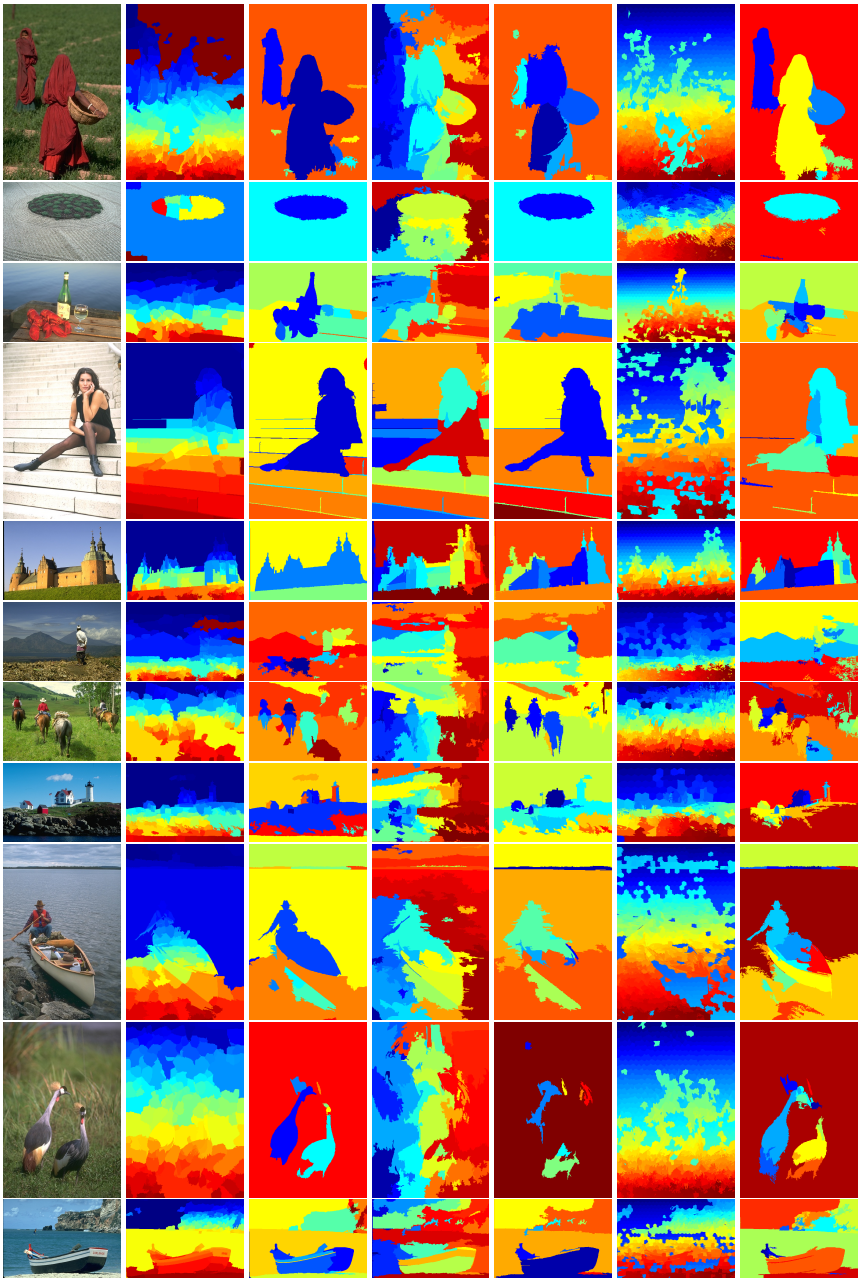


Figure C.6.: Segmentation examples on BSD500. From left to right: Orignal image, MS, UnOLIS$_{\text{MS}}$, GbS, UnOLIS$_{\text{GbS}}$, SLIC, UnOLIS$_{\text{SLIC}}$.

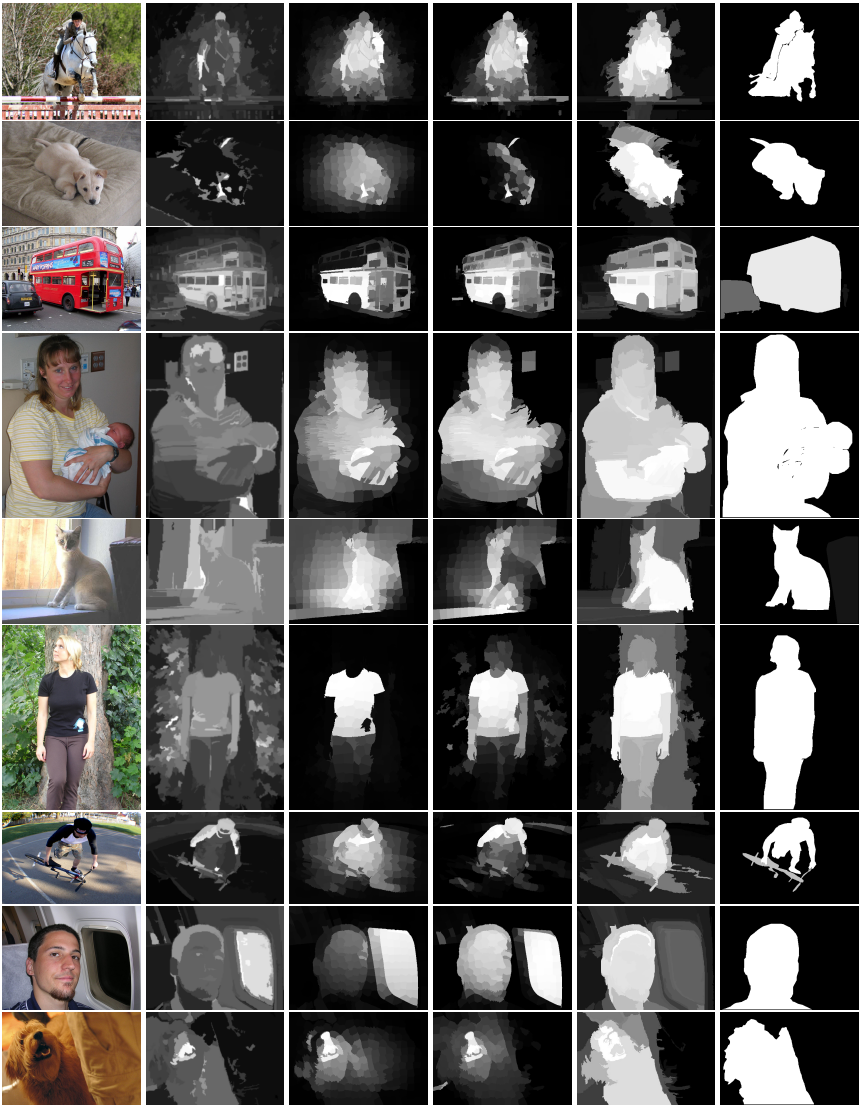


Figure C.7.: Challenging examples selected from the PASCAL-S dataset. First and last columns contain the original images and their ground truth saliency maps. Second to fifth columns contain the saliency maps generated by GC ([27]), MR ([188]), wCtr ([197]) and our method.

Bibliography

[1] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Susstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, 2012.

[2] Ravi Achanta, Sheila Hemami, Francisco Estrada, and Sabine Susstrunk. Frequency-tuned salient region detection. In *Proc. CVPR*, pages 1597–1604, 2009.

[3] Sharon Alpert, Meirav Galun, Achi Brandt, and Ronen Basri. Image segmentation by probabilistic bottom-up aggregation and cue integration. *IEEE transactions on pattern analysis and machine intelligence*, 34(2):315–327, 2012.

[4] Sharon Alpert, Meirav Galun, Achi Brandt, and Ronen Basri. Image segmentation by probabilistic bottom-up aggregation and cue integration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(2):315–327, 2012.

[5] Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W Hoffman, David Pfau, Tom Schaul, and Nando de Freitas. Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems*, pages 3981–3989, 2016.

[6] Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 33(5):898–916, 2011.

[7] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pages 214–223, 2017.

[8] Roberto Arroyo, J Javier Yebes, Luis M Bergasa, Iván G Daza, and Javier Almazán. Expert video-surveillance system for real-time detection of suspicious behaviors in shopping malls. *Expert systems with Applications*, 42(21):7991–8005, 2015.

[9] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv preprint arXiv:1511.00561*, 2015.

[10] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *European conference on computer vision*, pages 404–417. Springer, 2006.

[11] Gedas Bertasius, Jianbo Shi, and Lorenzo Torresani. Deepedge: A multi-scale bifurcated deep network for top-down contour detection. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pages 4380–4389. IEEE, 2015.

[12] Ali Borji. Boosting bottom-up and top-down visual features for saliency estimation. In *Proc. CVPR*, pages 438–445, 2012.

[13] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, 2004.

[14] Yuri Y Boykov and Marie-Pierre Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in nd images. In *Proc. ICCV*, volume 1, pages 105–112, 2001.

[15] Andrea Braides and Gianni Dal Maso. Non-local approximation of the mumford-shah functional. *Calculus of Variations and Partial Differential Equations*, 5(4):293–322, 1997.

[16] Xavier Bresson, Selim Esedoá¸lu, Pierre Vandergheynst, Jean-Philippe Thiran, and Stanley Osher. Fast global minimization of the active contour/snake model. *Journal of Mathematical Imaging and vision*, 28(2):151–167, 2007.

[17] Gabriel J Brostow, Julien Fauqueur, and Roberto Cipolla. Semantic object classes in video: A high-definition ground truth database. *Pattern Recognition Letters*, 30(2):88–97, 2009.

[18] John Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698, 1986.

[19] Joao Carreira and Cristian Sminchisescu. Constrained parametric min-cuts for automatic object segmentation. In *Proc. CVPR*, pages 3241–3248, 2010.

[20] Vicent Caselles, Ron Kimmel, and Guillermo Sapiro. Geodesic active contours. *International journal of computer vision*, 22(1):61–79, 1997.

[21] Moran Cerf, Jonathan Harel, Wolfgang Einhäuser, and Christof Koch. Predicting human gaze using low-level saliency combined with face detection. In *Advances in neural information processing systems*, pages 241–248, 2008.

[22] Antonin Chambolle. Image segmentation by variational methods: Mumford and shah functional and the discrete approximations. *SIAM Journal on Applied Mathematics*, 55(3):827–863, 1995.

[23] Tony F Chan and Luminita A Vese. Active contours without edges. *IEEE Transactions on image processing*, 10(2):266–277, 2001.

[24] Kai-Yueh Chang, Tyng-Luh Liu, Hwann-Tzong Chen, and Shang-Hong Lai. Fusing generic objectness and visual saliency for salient object detection. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 914–921. IEEE, 2011.

[25] Jianxu Chen, Lin Yang, Yizhe Zhang, Mark Alber, and Danny Z Chen. Combining fully convolutional and recurrent neural networks for 3d biomedical image segmentation. In *Advances in Neural Information Processing Systems*, pages 3036–3044, 2016.

[26] Tao Chen, Ming-Ming Cheng, Ping Tan, Ariel Shamir, and Shi-Min Hu. Sketch2photo: Internet image montage. *ACM Transactions on Graphics (TOG)*, 28(5):124, 2009.

[27] Ming Cheng, Niloy J Mitra, Xumin Huang, Philip HS Torr, and Song Hu. Global contrast based salient region detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):569–582, 2015.

[28] Ming-Ming Cheng, Niloy J Mitra, Xiaolei Huang, Philip HS Torr, and Shi-Min Hu. Salient object detection and segmentation. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (1):1–1, 2014.

[29] Ming-Ming Cheng, Jonathan Warrell, Wen-Yan Lin, Shuai Zheng, Vibhav Vineet, and Nigel Crook. Efficient salient region detection with soft image abstraction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1529–1536, 2013.

[30] Ming-Ming Cheng, Fang-Lue Zhang, Niloy J Mitra, Xiaolei Huang, and Shi-Min Hu. Repfinder: Finding approximately repeated scene elements for image editing. In *ACM Transactions on Graphics (TOG)*, volume 29, page 83. ACM, 2010.

[31] Yizong Cheng. Mean shift, mode seeking, and clustering. *IEEE transactions on pattern analysis and machine intelligence*, 17(8):790–799, 1995.

[32] François Chollet. Keras. `https://github.com/fchollet/keras`, 2015.

[33] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). In *4th International Conference on Learning Representations (ICLR)*, May 2016.

[34] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(5):603–619, 2002.

[35] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[36] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3213–3223, 2016.

[37] Antonio Criminisi, Toby Sharp, and Andrew Blake. Geos: Geodesic image segmentation. In *European Conference on Computer Vision*, pages 99–112. Springer, 2008.

[38] Angel Cruzroa, Ajay Basavanhally, Hannah Gilmore, Michael Feldman, Shridar Ganesan, Natalie Shih, John Tomaszewski, and Anant Madabhushi. Automatic detection of invasive ductal carcinoma in whole slide images with convolutional neural networks. *Proceedings of SPIE - The International Society for Optical Engineering*, 9041(2):139–144, 2014.

[39] Jifeng Dai, Kaiming He, and Jian Sun. Instance-aware semantic segmentation via multi-task network cascades. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3150–3158, 2016.

[40] A David and Ponce Jean. Computer vision: a modern approach. *Prentice Hall*, pages 654–659, 2002.

[41] Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[42] Guilherme N DeSouza and Avinash C Kak. Vision for mobile robot navigation: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 24(2):237–267, 2002.

[43] Carl Doersch and Andrew Zisserman. Multi-task self-supervised visual learning. In *The IEEE International Conference on Computer Vision (ICCV)*, 2017.

[44] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):295–307, 2016.

[45] Michael Donoser, Martin Urschler, Martin Hirzer, and Horst Bischof. Saliency driven total variation segmentation. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 817–824. IEEE, 2009.

[46] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2758–2766, 2015.

[47] Alexei A Efros and Thomas K Leung. Texture synthesis by non-parametric sampling. In *IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 1033–1038, 1999.

[48] Wolfgang Einhäuser, Merrielle Spain, and Pietro Perona. Objects predict fixations better than early saliency. *Journal of Vision*, 8(14):18–18, 2008.

[49] Lior Elazary and Laurent Itti. Interesting objects are visually salient. *Journal of vision*, 8(3):3–3, 2008.

[50] Martin Evening and Jeff Schewe. *Adobe Photoshop CS5 for photographers: the ultimate workshop*. Focal Press, 2012.

[51] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International journal of computer vision*, 111(1):98–136, 2015.

[52] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[53] Clement Farabet, Camille Couprie, Laurent Najman, and Yann LeCun. Learning hierarchical features for scene labeling. *IEEE Transactions on pattern analysis and machine intelligence*, 35(8):1915–1929, 2013.

[54] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2010.

[55] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.

[56] Basura Fernando, Hakan Bilen, Efstratios Gavves, and Stephen Gould. Self-supervised video representation learning with odd-one-out networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5729–5738. IEEE, 2017.

[57] Alexander Fix, Aritanan Gruber, Endre Boros, and Ramin Zabih. A graph cut algorithm for higher-order markov random fields. In *2011 IEEE International Conference on Computer Vision (ICCV)*, pages 1020–1027. IEEE, 2011.

[58] Gian Luca Foresti, Christian Micheloni, Lauro Snidaro, Paolo Remagnino, and Tim Ellis. Active video-based surveillance system: the low-level image and video processing techniques needed for implementation. *IEEE Signal Processing Magazine*, 22(2):25–37, 2005.

[59] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International Conference on Machine Learning*, pages 1180–1189, 2015.

[60] Yue Gao, Meng Wang, Zheng-Jun Zha, Jialie Shen, Xuelong Li, and Xindong Wu. Visual-textual joint relevance learning for tag-based social image search. *IEEE Transactions on Image Processing*, 22(1):363–376, 2013.

[61] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[62] Ross Girshick. Fast r-cnn. In *International Conference on Computer Vision (ICCV)*, 2015.

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

[64] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jagannath Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 580–587. IEEE, 2014.

[65] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jagannath Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 580–587, 2014.

[66] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

[67] Ian Goodfellow. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016.

[68] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[69] Stephen Gould, Richard Fulton, and Daphne Koller. Decomposing a scene into geometric and semantically consistent regions. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 1–8. IEEE, 2009.

[70] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pages 5767–5777, 2017.

[71] David Walter Hamlyn. *The psychology of perception: A philosophical examination of Gestalt theory and derivative theories of perception*. Routledge, 2017.

[72] Robert M Haralick and Linda G Shapiro. *Computer and robot vision*. Addison-wesley, 1992.

[73] Jonathan Harel, Christof Koch, and Pietro Perona. Graph-based visual saliency. In *Advances in neural information processing systems*, pages 545–552, 2006.

[74] Bharath Hariharan, Pablo Arbeláez, Lubomir Bourdev, Subhransu Maji, and Jitendra Malik. Semantic contours from inverse detectors. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 991–998. IEEE, 2011.

[75] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Simultaneous detection and segmentation. In *European Conference on Computer Vision*, pages 297–312. Springer, 2014.

[76] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Hypercolumns for object segmentation and fine-grained localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 447–456, 2015.

[77] Jean-Bernard Hayet, Frédéric Lerasle, and Michel Devy. A visual landmark framework for indoor mobile robot navigation. In *IEEE International Conference on Robotics and Automation*, volume 4, pages 3942–3947. IEEE, 2002.

[78] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034, 2015.

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

[80] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *stat*, 1050:9, 2015.

[81] Tin Kam Ho, Jonathan J. Hull, and Sargur N. Srihari. Decision combination in multiple classifier systems. *IEEE transactions on pattern analysis and machine intelligence*, 16(1):66–75, 1994.

[82] Xiaodi Hou, Jonathan Harel, and Christof Koch. Image signature: Highlighting sparse salient regions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(1):194–201, 2012.

[83] Shi-Min Hu, Tao Chen, Kun Xu, Ming-Ming Cheng, and Ralph R Martin. Internet visual media processing: a survey with graphics and vision applications. *The Visual Computer*, 29(5):393–405, 2013.

[84] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, 2017.

[85] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, 2015.

[86] Laurent Itti, Christof Koch, and Ernst Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (11):1254–1259, 1998.

[87] Suyog Jain, Bo Xiong, and Kristen Grauman. Pixel objectness. *arXiv preprint arXiv:1701.05349*, 2017.

[88] Suyog Dutt Jain, Bo Xiong, and Kristen Grauman. Pixel objectness. *arXiv preprint arXiv:1701.05349*, 2017.

[89] Ahmad Jalal and Shaharyar Kamal. Real-time life logging via a depth silhouette-based human activity recognition system for smart home services. In *2014 11th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 74–80. IEEE, 2014.

[90] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678. ACM, 2014.

[91] Huaizu Jiang, Jingdong Wang, Zejian Yuan, Yang Wu, Nanning Zheng, and Shipeng Li. Salient object detection: A discriminative regional feature integration approach. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 2083–2090. IEEE, 2013.

[92] Tilke Judd, Frédo Durand, and Antonio Torralba. A benchmark of computational models of saliency to predict human fixations. In *MIT Technical Report*, 2012.

[93] Jiwhan Kim, Dongyoon Han, Yu-Wing Tai, and Junmo Kim. Salient region detection via high-dimensional color transform. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 883–890, 2014.

[94] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1646–1654, 2016.

[95] Josef Kittler, Mohamad Hatef, Robert PW Duin, and Jiri Matas. On combining classifiers. *IEEE transactions on pattern analysis and machine intelligence*, 20(3):226–239, 1998.

[96] KI Kiy. Segmentation and detection of contrast objects and their application in robot navigation. *Pattern Recognition and Image Analysis*, 25(2):338–346, 2015.

[97] Pushmeet Kohli and Philip HS Torr. Dynamic graph cuts for efficient inference in markov random fields. *IEEE transactions on pattern analysis and machine intelligence*, 29(12):2079–2088, 2007.

[98] Vladimir Kolmogorov and Ramin Zabin. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):147–159, 2004.

[99] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In *Advances in Neural Information Processing Systems (NIPS)*, pages 109–117, 2011.

[100] Philipp Krähenbühl and Vladlen Koltun. Parameter learning and convergent inference for dense random fields. In *International Conference on Machine Learning (ICML)*, 2013.

[101] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1097–1105, 2012.

[102] Anders Krogh and John A Hertz. A simple weight decay can improve generalization. In *Advances in neural information processing systems*, pages 950–957, 1992.

[103] M Pawan Kumar, Haithem Turki, Dan Preston, and Daphne Koller. Parameter estimation and energy minimization for region-based semantic segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 37(7):1373–1386, 2015.

[104] Lubor Ladicky, Chris Russell, Pushmeet Kohli, and Philip HS Torr. Graph cut based inference with co-occurrence statistics. In *European Conference on Computer Vision*, pages 239–253. Springer, 2010.

[105] Guanbin Li and Yizhou Yu. Visual saliency based on multiscale deep features. In *Proc. CVPR*, pages 5455–5463, 2015.

[106] Guanbin Li and Yizhou Yu. Deep contrast learning for salient object detection. In *Proc. CVPR*, 2016.

[107] Guanbin Li and Yizhou Yu. Deep contrast learning for salient object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[108] Guanbin Li and Yizhou Yu. Visual saliency detection based on multiscale deep cnn features. *IEEE Transactions on Image Processing*, 25(11):5012–5024, 2016.

[109] Ruiyu Li, Kaican Li, Yi-Chun Kuo, Michelle Shu, Xiaojuan Qi, Xiaoyong Shen, and Jiaya Jia. Referring image segmentation via recurrent refinement networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5745–5753, 2018.

[110] Xiaohui Li, Huchuan Lu, Lihe Zhang, Xiang Ruan, and Ming-Hsuan Yang. Saliency detection via dense and sparse reconstruction. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 2976–2983. IEEE, 2013.

[111] Yi Li, Haozhi Qi, Jifeng Dai, Xiangyang Ji, and Yichen Wei. Fully convolutional instance-aware semantic segmentation. *arXiv preprint arXiv:1611.07709*, 2016.

[112] Yin Li, Xiaodi Hou, Christian Koch, James M Rehg, and Alan L Yuille. The secrets of salient object segmentation. In *Proc. CVPR*, pages 280–287, 2014.

[113] Yin Li, Xiaodi Hou, Christof Koch, James M Rehg, and Alan L Yuille. The secrets of salient object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 280–287, 2014.

[114] G Linda and C George Shapiro. Stockman, computer vision, 2001.

[115] Nick Littlestone and Manfred K Warmuth. The weighted majority algorithm. *Information and computation*, 108(2):212–261, 1994.

[116] Tie Liu, Zejian Yuan, Jian Sun, Jingdong Wang, Nanning Zheng, Xiaoou Tang, and Heung-Yeung Shum. Learning to detect a salient object. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(2):353–367, 2011.

[117] Ying Liu, Dengsheng Zhang, Guojun Lu, and Wei-Ying Ma. A survey of content-based image retrieval with high-level semantics. *Pattern recognition*, 40(1):262–282, 2007.

[118] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440, 2015.

[119] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *International Conference on Machine Learning*, pages 97–105, 2015.

[120] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Unsupervised domain adaptation with residual transfer networks. In *Advances in Neural Information Processing Systems*, pages 136–144, 2016.

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

[122] Chao Ma, Jia-Bin Huang, Xiaokang Yang, and Ming-Hsuan Yang. Hierarchical convolutional features for visual tracking. In *Proceedings of the IEEE international conference on computer vision*, pages 3074–3082, 2015.

[123] Lucia Maddalena and Alfredo Petrosino. A self-organizing approach to background subtraction for visual surveillance applications. *IEEE Transactions on Image Processing*, 17(7):1168–1177, 2008.

[124] Vijay Mahadevan and Nuno Vasconcelos. Saliency-based discriminant tracking. In *Proc. CVPR*, pages 1007–1013, 2009.

[125] Michael Maire, Pablo Arbelaez, Charless Fowlkes, and Jitendra Malik. Using contours to detect and localize junctions in natural images. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[126] Tomasz Malisiewicz and Alexei A Efros. Improving spatial support for objects via multiple segmentations. 2007.

[127] Ran Margolin, Ayellet Tal, and Lihi Zelnik-Manor. What makes a patch distinct? In *Proc. CVPR*, pages 1139–1146, 2013.

[128] Lukas Mauch, Chunlai Wang, and Bin Yang. Subset selection for visualization of relevant image fractions for deep learning based semantic image segmentation. *Journal of the Franklin Institute*, 355(4):1931–1944, 2018.

[129] Alex Mihailidis, Brent Carmichael, and Jennifer Boger. The use of computer vision in an intelligent environment to support aging-in-place, safety, and independence in the home. *IEEE Transactions on information technology in biomedicine*, 8(3):238–247, 2004.

[130] David Mumford and Jayant Shah. Optimal approximations by piecewise smooth functions and associated variational problems. *Communications on pure and applied mathematics*, 42(5):577–685, 1989.

[131] Hyeonseob Nam and Bohyung Han. Learning multi-domain convolutional neural networks for visual tracking. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4293–4302. IEEE, 2016.

[132] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1520–1528, 2015.

[133] Shipra Ojha and Sachin Sakhare. Image processing techniques for object tracking in video surveillance-a survey. In *2015 International Conference on Pervasive Computing (ICPC)*, pages 1–6. IEEE, 2015.

[134] Nabil Ouerhani, Javier Bracamonte, Heinz Hügli, Michael Ansorge, and Fausto Pellandini. Adaptive color image compression based on visual attention. In *Proc. 11th IEEE International Conference on Image Analysis and Processing*, pages 416–421, 2001.

[135] Caroline Pantofaru, Cordelia Schmid, and Martial Hebert. Object recognition by integrating multiple image segmentations. *Computer Vision–ECCV 2008*, pages 481–494, 2008.

[136] Federico Perazzi, Philipp Krähenbühl, Yael Pritch, and Alexander Hornung. Saliency filters: Contrast based filtering for salient region detection. In *Proc. CVPR*, pages 733–740, 2012.

[137] Dzung L Pham, Chenyang Xu, and Jerry L Prince. Current methods in medical image segmentation. *Annual review of biomedical engineering*, 2(1):315–337, 2000.

[138] Thomas Pock, Antonin Chambolle, Daniel Cremers, and Horst Bischof. A convex relaxation approach for computing minimal partitions. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 810–817. IEEE, 2009.

[139] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[140] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 91–99, 2015.

[141] Zhile Ren and Gregory Shakhnarovich. Image segmentation by cascaded region agglomeration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2011–2018, 2013.

[142] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. In *ACM transactions on graphics (TOG)*, volume 23, pages 309–314. ACM, 2004.

[143] Pierre Rousseau, Vincent Jolivet, and Djamchid Ghazanfarpour. Realistic real-time rain rendering. *Computers & Graphics*, 30(4):507–518, 2006.

[144] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533, 1986.

[145] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[146] Bryan C Russell, William T Freeman, Alexei A Efros, Josef Sivic, and Andrew Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 1605–1614. IEEE, 2006.

[147] Chris Russell, Pushmeet Kohli, Philip HS Torr, et al. Associative hierarchical crfs for object class image segmentation. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 739–746. IEEE, 2009.

[148] Ueli Rutishauser, Dirk Walther, Christof Koch, and Pietro Perona. Is bottom-up attention useful for object recognition? In *Proc. CVPR*, volume 2, pages II–37, 2004.

[149] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. In *Advances in Neural Information Processing Systems*, pages 3856–3866, 2017.

[150] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242, 2016.

[151] A. G. Schwing, T. Hazan, M. Pollefeys, and R. Urtasun. Efficient structured prediction with latent variables for general graphical models. In *International Conference on Machine Learning (ICML)*, 2012.

[152] Jianbing Shen, Xiaopeng Hao, Zhiyuan Liang, Yu Liu, Wenguan Wang, and Ling Shao. Real-time superpixel segmentation by dbscan clustering algorithm. *IEEE Transactions on Image Processing*, 25(12):5933–5942, 2016.

[153] Wei Shen, Xinggang Wang, Yan Wang, Xiang Bai, and Zhijiang Zhang. Deepcontour: A deep convolutional feature learned by positive-sharing loss for contour detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3982–3991, 2015.

[154] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):888–905, 2000.

[155] Jianping Shi, Qiong Yan, Li Xu, and Jiaya Jia. Hierarchical image saliency detection on extended cssd. *IEEE transactions on pattern analysis and machine intelligence*, 38(4):717–729, 2016.

[156] Jamie Shotton, John Winn, Carsten Rother, and Antonio Criminisi. Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *European Conference on Computer Vision (ECCV)*, pages 1–15. Springer, 2006.

[157] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations (ICLR)*, May 2015.

[158] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[159] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[160] Sonia Starik and Michael Werman. Simulation of rain in videos. In *Texture Workshop, IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 406–409, 2003.

[161] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[162] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[163] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[164] Martin Szummer, Pushmeet Kohli, and Derek Hoiem. Learning crfs using graph cuts. In *European conference on computer vision*, pages 582–595. Springer, 2008.

[165] The Theano Development Team, Rami Al-Rfou, Guillaume Alain, Amjad Almahairi, Christof Angermueller, Dzmitry Bahdanau, Nicolas Ballas, Frédéric Bastien, Justin Bayer, Anatoly Belikov, et al. Theano: A python framework for fast computation of mathematical expressions. *arXiv preprint arXiv:1605.02688*, 2016.

[166] Tinne Tuytelaars, Christoph H Lampert, Matthew B Blaschko, and Wray Buntine. Unsupervised object discovery: A comparison. *International journal of computer vision*, 88(2):284–302, 2010.

[167] Eric Tzeng, Judy Hoffman, Trevor Darrell, and Kate Saenko. Simultaneous deep transfer across domains and tasks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4068–4076, 2015.

[168] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Computer Vision and Pattern Recognition (CVPR)*, volume 1, page 4, 2017.

[169] Jasper RR Uijlings, Koen EA van de Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013.

[170] Koen EA Van De Sande, Theo Gevers, and Cees GM Snoek. Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1582–1596, 2010.

[171] Manik Varma and Andrew Zisserman. A statistical approach to texture classification from single images. *International Journal of Computer Vision*, 62(1-2):61–81, 2005.

[172] Luminita A Vese and Tony F Chan. A multiphase level set framework for image segmentation using the mumford and shah model. *International journal of computer vision*, 50(3):271–293, 2002.

[173] Francesco Visin, Marco Ciccone, Adriana Romero, Kyle Kastner, Kyunghyun Cho, Yoshua Bengio, Matteo Matteucci, and Aaron Courville. Reseg: A recurrent neural network-based model for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 41–48, 2016.

[174] Johan Wagemans, James H Elder, Michael Kubovy, Stephen E Palmer, Mary A Peterson, Manish Singh, and Rüdiger von der Heydt. A century of gestalt psychology in visual perception: I. perceptual grouping and figure–ground organization. *Psychological bulletin*, 138(6):1172, 2012.

[175] Chunlai Wang, Lukas Mauch, Ze Guo, and Bin Yang. On semantic image segmentation using deep convolutional neural network with shortcuts and easy class extension. In *Image Processing Theory Tools and Applications (IPTA), 2016 6th International Conference on*, pages 1–6. IEEE, 2016.

[176] Chunlai Wang, Lukas Mauch, Mehul Manoj Saxena, and Bin Yang. On the contextual aspects of using deep convolutional neural network for semantic image segmentation. *Journal of Electronic Imaging*, 27(5):051223, 2018.

[177] Chunlai Wang and Bin Yang. Saliency-guided object proposal for refined salient region detection. In *Visual Communications and Image Processing (VCIP), 2016*, pages 1–4. IEEE, 2016.

[178] Chunlai Wang and Bin Yang. An unsupervised object-level image segmentation method based on foreground and background priors. In *Image Analysis and Interpretation (SSIAI), 2016 IEEE Southwest Symposium on*, pages 141–144. IEEE, 2016.

[179] Chunlai Wang, Jiawei Yu, Lukas Mauch, and Bin Yang. Binary segmentation based class extension in semantic image segmentation using convolutional neural networks. In *2018 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2018.

[180] Lijun Wang, Huchuan Lu, Xiang Ruan, and Ming-Hsuan Yang. Deep networks for saliency detection via local estimation and global search. In *Proc. CVPR*, pages 3183–3192, 2015.

[181] Lijun Wang, Wanli Ouyang, Xiaogang Wang, and Huchuan Lu. Visual tracking with fully convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3119–3127, 2015.

[182] Xuya Wang. Salient object segmentation using deep convolutional neural network. *Master Thesis. University of Stuttgart*, 2016.

[183] Yichen Wei, Fang Wen, Wangjiang Zhu, and Jian Sun. Geodesic saliency using background priors. In *Computer Vision–ECCV 2012*, pages 29–42. Springer, 2012.

[184] Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487, 2016.

[185] Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *Proceedings of the IEEE international conference on computer vision*, pages 1395–1403, 2015.

[186] Lei Xu, Adam Krzyzak, and Ching Y Suen. Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE transactions on systems, man, and cybernetics*, 22(3):418–435, 1992.

[187] Qiong Yan, Li Xu, Jianping Shi, and Jiaya Jia. Hierarchical saliency detection. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 1155–1162. IEEE, 2013.

[188] Chuan Yang, Lihe Zhang, Huchuan Lu, Xiang Ruan, and Ming-Hsuan Yang. Saliency detection via graph-based manifold ranking. In *Proc. CVPR*, pages 3166–3173, 2013.

[189] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. In *Deep Learning Workshop, International Conference on Machine Learning (ICML)*, 2015.

[190] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[191] Guo-Xin Zhang, Ming-Ming Cheng, Shi-Min Hu, and Ralph R Martin. A shape-preserving approach to image resizing. In *Computer Graphics Forum*, volume 28, pages 1897–1906. Wiley Online Library, 2009.

[192] Jianming Zhang, Shugao Ma, Mehrnoosh Sameki, Stan Sclaroff, Margrit Betke, Zhe Lin, Xiaohui Shen, Brian Price, and Radomir Mech. Salient object subitizing. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[193] Wei Zhang, Yi Tian, Xiaojie Zha, and Hantao Liu. Benchmarking state-of-the-art visual saliency models for image quality assessment. In *Proc. ICASSP*, 2016.

[194] Rui Zhao, Wanli Ouyang, Hongsheng Li, and Xiaogang Wang. Saliency detection by multi-context deep learning. In *Proc. CVPR*, pages 1265–1274, 2015.

[195] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Object detectors emerge in deep scene cnns. In *3rd International Conference on Learning Representations (ICLR)*, 2015.

[196] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 2017.

[197] Wangjiang Zhu, Shuang Liang, Yichen Wei, and Jian Sun. Saliency optimization from robust background detection. In *Proc. CVPR*, pages 2814–2821, 2014.

[198] C Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In *Computer Vision–ECCV 2014*, pages 391–405. Springer, 2014.