Higher-order Losses and Optimization for Low-level and Deep Segmentation

by

Dmitrii Marin

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Computer Science

Waterloo, Ontario, Canada, 2021

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner:	Dr. Pascal Fua Professor, School of Computer and Communication Science, École Polytechnique Fédérale de Lausanne
Supervisor:	Dr. Yuri Boykov Professor, School of Computer Science, University of Waterloo
Internal Member:	Dr. Yaoliang Yu Professor, School of Computer Science, University of Waterloo
	Dr. Olga Veksler Professor, School of Computer Science, University of Waterloo
Internal-External Member:	Dr. Paul Fieguth Professor, Department of Systems Design Engineering, University of Waterloo

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

Dmitrii Marin has written this thesis under the supervision of Prof. Yuri Boykov. This thesis consists in part of several manuscripts written for publication as follows.

Dmitrii Marin is the sole author of Chapter 1. Parts of the thesis have been published in [184, 323, 186, 187, 183] co-authored by Dmitrii Marin. This includes the following: Chapter 2 is based on the manuscripts published in [184, 323]; Chapter 3 is based on the manuscript published in [186]; Chapter 4 is based on the manuscripts published in [187, 183].

Research presented in Chapter 2:

The research was conducted at the University of Western Ontario under the supervision of Prof. Yuri Boykov.

Dmitrii Marin is the first author and main contributor of [184]. He conducted all the experiments. Yuri Boykov provided supervision and contributed to the design and analysis of the methods and experiments. The manuscript was drafted jointly by Dmitrii Marin and Yuri Boykov. Yuchen Zhong and Yuri Boykov conducted preliminary research on curvature-based modeling of vessel tree centerlines.

Zhongwen Zhang is the first and main contributor of [323]. He conducted all the experiments. Yuri Boykov and Dmitrii Marin provided supervision and contributed to the design and analysis of the methods and experiments. The manuscript was drafted jointly by Zhongwen Zhang, Dmitrii Marin and Yuri Boykov.

Maria Drangova provided valuable feedback on the methods and manuscript drafts, she also provided raw 3D vessel data for both publications [184, 323].

Research presented in Chapter 3:

The research was conducted at the University of Western Ontario under the supervision of Prof. Yuri Boykov.

Dmitrii Marin is the first author and main contributor of [186]. He conducted all the experiments except Figure 3.2, which was obtained by Ment Tang. Yuri Boykov provided supervision and contributed to the design and analysis of the density bias theory, methods addressing the bias and experiments. The manuscript was drafted jointly by Dmitrii Marin and Yuri Boykov. Meng Tang & Ismail Ben Ayed conducted experiments for [281] where they empirically demonstrated the density bias, which sparked our interest in its theoretical explanation, presented in this thesis.

Research presented in Chapter 4:

The research was conducted at the University of Western Ontario and the University of Waterloo under the supervision of Prof. Yuri Boykov.

Dmitrii Marin is the first author and main contributor of [187, 183]. He conducted all the experiments presented in this thesis. Yuri Boykov provided supervision and contributed to the design and analysis of methods and experiments. The manuscripts were drafted jointly by Dmitrii Marin and Yuri Boykov. Meng Tang conducted some experiments for [187], which were not included in this thesis.

Abstract

Regularized objectives are common in low-level and deep segmentation. Regularization incorporates prior knowledge into objectives or losses. It represents constraints necessary to address ill-posedness, data noise, outliers, lack of supervision, *etc.* However, such constraints come at significant costs. First, regularization priors may lead to unintended biases, known or unknown. Since these can adversely affect specific applications, it is important to understand the causes & effects of these biases and to develop their solutions. Second, common regularized objectives are highly non-convex and present challenges for optimization. As known in low-level vision, first-order approaches like gradient descent are significantly weaker than more advanced algorithms. Yet, variants of the gradient descent dominate optimization of the loss functions for deep neural networks due to their size and complexity. Hence, standard segmentation networks still require an overwhelming amount of precise pixel-level supervision for training.

This thesis addresses three related problems concerning higher-order objectives and higher-order optimizers. First, we focus on a challenging application—unsupervised vascular tree extraction in large 3D volumes containing complex "entanglements" of near-capillary vessels. In the context of vasculature with unrestricted topology, we propose a new general curvature-regularizing model for arbitrarily complex one-dimensional curvilinear structures. In contrast, the standard surface regularization methods are impractical for thin vessels due to strong shrinking bias or the complexity of Gaussian/min curvature modeling for two-dimensional manifolds. In general, the shrinking bias is one well-known example of bias in the standard regularization methods. The second contribution of this thesis is a characterization of other new forms of biases in classical segmentation models that were not understood in the past. We develop new theories establishing data density biases in common pair-wise or graph-based clustering objectives, such as kernel K-means and normalized cut. This theoretical understanding inspires our new segmentation algorithms avoiding such biases. The third contribution of the thesis is a new optimization algorithm addressing the limitations of gradient descent in the context of regularized losses for deep learning. Our general trust-region algorithm can be seen as a high-order chain rule for network training. It can use many standard low-level regularizers and their powerful solvers. We improve the state-of-the-art in weakly-supervised semantic segmentation using a well-motivated low-level regularization model and its graph-cut solver.

Acknowledgements

First, I would like to express my deepest gratitude to my Ph.D. adviser Yuri Boykov. I feel lucky to be advised by a professor who has dedicated to me an unmatched amount of support and time. I am grateful for the many multi-hour meetings spent on discussing our projects as well as paper writing in an endless pursuit of clarity, simplicity and compactness.

I would like to thank prof. Olga Veksler (University of Waterloo) for the general support, constant readiness to help and insightful discussions. I thank professors Carl Olsson (Lund University) and Ismail Ben Ayed (ETS Montreal) for valuable feedback and discussions.

I thank my Ph.D. advisory and examination committees' members.

I would like to thank my co-authors, without whom my research work would not be possible. Especially, I would like to acknowledge my lab colleagues Meng Tang, Zhongwen (Rex) Zhang, Egor Chesakov, Yuchen Zhong.

I would like to thank my managers and mentors during my three research internships, especially Changlei Wu, Zijian He and Peter Vajda from Facebook, Jen-Hao Rick Chang, Anurag Ranjan, Anish Prabhu and Oncel Tuzel from Apple. While my internship research has not been ultimately included in this thesis, it has significantly broadened my expertise particularly in the area of deep learning.

I would like to acknowledge the University of Western Ontario, where I worked during my first three years of the doctorate program, and the University of Waterloo. I thank these universities' professors and other employees who contributed to my academic progress.

I thank the Society of Graduate Students and Teaching Assistants & Postdocs Union PSAC610 at the UWO, the Graduate Students Association and the University of Waterloo for the leadership opportunities to serve the graduate students community.

I thank Canada, particularly the province of Ontario, and my supervisor for the direct and indirect financial support of my work.

In addition, I would like to thank my family and friends for their wise counsel and sympathetic ear. You are always there for me.

Also, I thank my first academic advisors, Victor Lempitsky and Yuri Orechov.

Lastly, I would like to acknowledge my homeland, Russia, for an incredible gift of free and accessible education, from school to university, that provided a strong foundation for my research.

Dedication

To my parents and grandparents, who instilled the values of knowledge and education in me, for their endless support and love.

Table of Contents

List of Tables

List of Figures

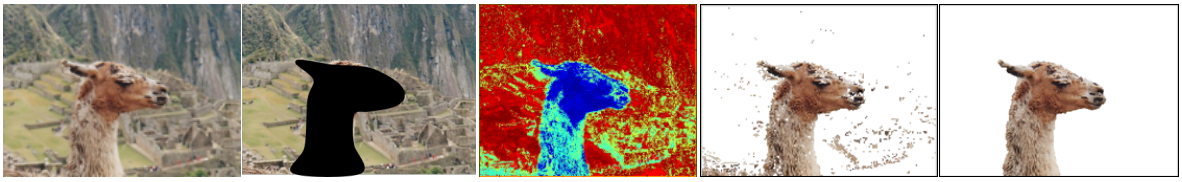
Chapter 1

Introduction into Image Segmentation

Computer vision is a field of study that enables computers to interpret images and videos. As an interdisciplinary science, computer vision [88] uses geometry, physics, decision theory, probability theory, machine learning, and other disciplines.

Computer vision includes many problems of a low and high-level understanding of digital images and video. For example, *image classification* aims to assign to an input image a semantic label from a set specified in advance. *Edge detection* aims to discover thin high-contrast regions of the image where the color changes more or less "abruptly". *Image segmentation* finds the partition of the image pixels into a few subsets that represent different objects or categories. *Motion detection* and *optical flow* aim to reconstruct the pixel correspondence for a pair of images, hence determining the motion of the objects or camera. A related *stereo* problem finds pixel correspondence for a stereo pair, hence determining the disparity (or parallax) and depth at each of the pixels. Stereo is a subtask of *3D multi-view reconstruction*, which aims to reconstruct 3D scenes from an array of images taken from different viewpoints. Computer vision also includes event detection, video tracking, objects detection, 3D pose estimation, image restoration, and many others.

In the followingSection 1.1.1, we briefly introduce the problem of interactive image segmentation. This problem is a simple and illustrative example of a much broader area of image segmentation, which is central to this thesis. This and later sections review the basic tools in segmentation and general computer vision, as well as their limitations addressed in this thesis. But first, we introduce some of our most common **mathematical notations**. Numerical sets are denoted by the blackboard bold typeface, *e.g.* $\mathbb{R}$ is the



(a) image from [240] (b) ground truth (c) log likelihood ratio (d) threshold (e) graph cut [31]

Figure 1.1: Image Segmentation: An original image **(a)** is to be partitioned into sets of the object and background pixels, as in **(b)**. The ground truth segmentation is in **(b)**. Given the color distribution for object and background, the *maximum likelihood* **(c)** gives the optimal rule (1.4) classifying pixels into the object **(d)** and background. However, due to the independent treatment of pixels, significant noise is observed in **(d)**. Methods, such as [31], exploit the correlation between pixels to produce a "smooth" result as in **(e)**.

set of real numbers. Sets are denoted by capital roman letters, such as A, B, X, *etc.* Vectors (and 1D arrays) are denoted by lower case bold letters such as $\boldsymbol{x}$, and all vectors are assumed to be column vectors. The symbol "$\top$" denotes the matrix transpose such as $\boldsymbol{x}^\top$, which is a row vector. Uppercase bold letters, such as $\boldsymbol{M}$, denote matrices or other multi-dimensional arrays such as images. An image is typically be denoted $\boldsymbol{I} \in \mathcal{I}$ where $\mathcal{I}$ is the set of all possible images. $I_p \in \mathbb{R}$ denotes scalar intensity of image pixel p. $\boldsymbol{I}_p$ denotes the multi-channel intensity vector of image pixel p. $[P]$ is the Iverson bracket, which equals 1 if the argument P is true, and equals 0 otherwise. The dot-product between vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ is denoted as $\boldsymbol{x} \cdot \boldsymbol{y}$.

1.1 Low-level Models & Regularization

First, we introduce the problem of *interactive image segmentation*, which is a classic computer vision problem. Optimization-based approaches dominate in segmentation. This thesis mainly focuses on the discrete formulation of the problem. The following subsection introduces the corresponding discrete optimization framework. Later, we briefly review alternative continuous representations in Section 1.1.2. Because of the difference in representation, they result in different numerical techniques, but the underlying models are often closely related and produce similar solutions in practice.

1.1.1 Segmentation as Discrete Optimization Problem

Image segmentation is a common computer vision problem of partitioning the set of image pixels into several subsets. In our example, we consider two subsets representing object (foreground) and background, see Figure 1.1. Suppose we are given a picture $\boldsymbol{I}$ consisting of pixels $\boldsymbol{I}_p$ for $p \in V$ where V is the set of pixels. The color intensity vector $\boldsymbol{I}_p \in \mathbb{R}^n$ where n is the dimensionallity of the color space (*e.g.* $n = 3$ for RGB images). The problem is to partition the pixels into object S and background $\bar{S}$ sets such that $V = S \cup \bar{S}$ and $S \cap \bar{S} = \emptyset$. Such partitioning in the context of computer vision is called *segmentation* and sets S and $\bar{S}$ are called *segments*. There is a label variable $x_p \in \mathcal{L}$ for each pixel $p \in V$ where the label set $\mathcal{L} = \{0, 1\}$, and $x_p = [p \in S]$. Let us define the indicator variables

$$x_p^l \quad := \quad [x_p = l]. \tag{1.1}$$

Let us also define vector $\boldsymbol{x}_p = (x_p^0, x_p^1, \ldots, x_p^{|\mathcal{L}|-1})^\top$.

As there is a multitude of different partitions, the problem requires additional constraints to be well-posed. A common way of resolving the ambiguity is providing a special function $E(S) \in \mathbb{R}$ ranking partitions, then, the solution S^* is found through optimization

$$S^* = \arg\min_S E(S). \tag{1.2}$$

Motivated by physics, such functions are often called *energies*.

The statistical decision theory studies the choice of such objectives from the probabilistic perspective. A common strategy is a maximum likelihood (ML) principle stating that the unknown parameters should be chosen such that the probability of observed data under the current model is maximized. In our case, the parameter is segmentation S.

In the following we temporarily assume the independence of pixel labels as well as the knowledge of the probability distribution of color in object pixels $\Pr(\boldsymbol{I}_p \,|\, x_p = 1)$ and background pixels $\Pr(\boldsymbol{I}_p \,|\, x_p = 0)$. The ML principle yields the rule:

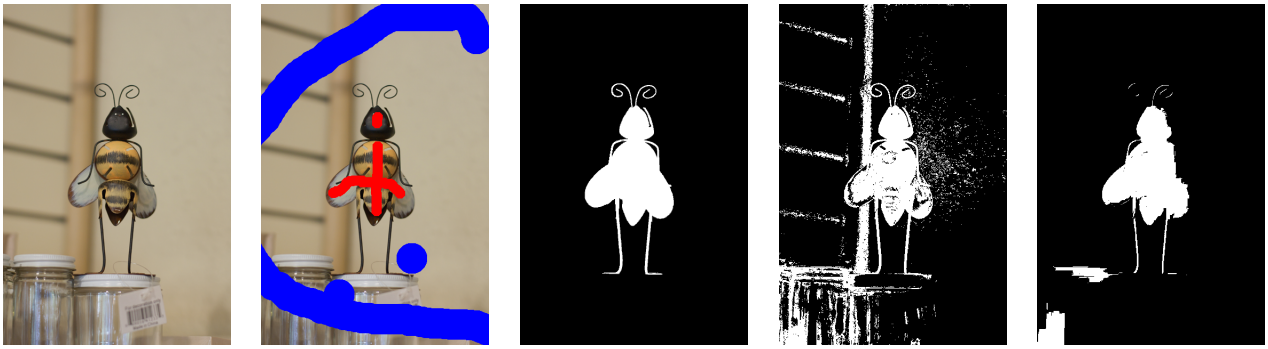
$$S^* = \arg\max_S \Pr(\boldsymbol{I} \,|\, S) \qquad \Longleftrightarrow \qquad x_p^* = \arg\max_{l \in \mathcal{L}} \Pr(\boldsymbol{I}_p \,|\, x_p = l). \tag{1.3}$$

The above is equivalent to

$$x_p^* \quad = \quad \arg\min_{x_p} \boldsymbol{u}_p \cdot \boldsymbol{x}_p \quad , \tag{1.4}$$

where

$$u_p^l \quad = \quad -\ln \Pr(\boldsymbol{I}_p \,|\, x_p = l). \tag{1.5}$$



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

Figure 1.2: Interactive Image Segmentation: The original image **(a)** is to be partitioned into the object and background. The blue and red scribbles in **(b)** denote the pixels that must belong to the background and object segments correspondingly. In **(c-e)** the black and white pixels code background and object segments correspondingly. The desired *ground-truth* segmentation is shown in **(c)**. The solution given by (1.4) is shown in **(d)** where color models are estimated from the user input **(b)**. **(e)** shows the regularized result of (1.7) [31]. Note, that most of the noise is removed. However, some thin structures like the bee's antennae almost disappear due to the shrinking bias.

The corresponding objective w.r.t. (1.2) is the sum of log-likelihoods

$$E(S) \quad = \quad \sum_{p \in V} - \ln \Pr(\boldsymbol{I}_p \,|\, x_p) \quad = \quad \sum_{p \in V} \sum_{l \in \mathcal{L}} u_p^l x_p^l. \tag{1.6}$$

Such energies that linearly depend on the segment indicator variables $\{x_p^l\}$ are called *unary*. These terms measuring adherence of a data point (pixel) to a specific label are called *data terms* or *appearance models*. We address the practical problem of unknown appearance models momentarily, but let us first consider one common extension of unary energy (1.6).

Due to complex overlapping appearance models, the resulting segmentation tends to have many mistakes as in Figure 1.1(d). The segmentation in Figure 1.1(d) is noisy, *i.e.* it has many disconnected parts, each consisting of few pixels. Intuitively, the photographs of typical objects should not result in such artifacts. This intuition, if incorporated into decision rule (1.6), could avoid noisy solutions. For example, such noisy solutions have many "discontinuities", *i.e.* transitions from object to background and vice versa. The following energy [97, 26], extending (1.6), accounts for the number of discontinuities:

$$E(S) \quad = \quad \sum_{p \in V} \sum_{l \in \mathcal{L}} u_p^l x_p^l + \gamma \sum_{q \in G_p} w_{pq} \, [x_p \neq x_q] \tag{1.7}$$

where $\gamma > 0$, G_p is the set of neighboring pixels to p, and each discontinuity is weighted by $w_{pq} = \exp(-\beta\|\boldsymbol{I}_p - \boldsymbol{I}_q\|^2)$ based on color difference $\|\boldsymbol{I}_p - \boldsymbol{I}_q\|$ and $\beta^{-1} = \frac{1}{\sum_p |G_p|} \sum_{p \in V} \sum_{q \in G_p} \|\boldsymbol{I}_p - \boldsymbol{I}_q\|^2$. So, solutions with many discontinuities will be assigned a higher energy value and likely not be chosen by optimization. Indeed, the optimal solution is much improved, see Figure 1.1(e). In essence, we introduced a *bias* to "smooth" solutions. Weights w_{pq} penalize less if boundary between object and background lies in the area of high contrast (large color difference $\|\boldsymbol{I}_p - \boldsymbol{I}_q\|$). Thus, the model (1.7) also encourages *edge alignment*.

In Bayesian statistics, approaches like (1.7) correspond to a *maximum a posteriori probability* (MAP) estimate for parameter S [97]. MAP is closely related to ML but incorporates a *prior distribution* $g_{prior}(S)$ over the unknown parameter S:

$$S^* \;=\; \arg\max_S \Pr(S|\boldsymbol{I}) \;=\; \arg\max_S \Pr(\boldsymbol{I}|S)\, g_{prior}(S). \tag{1.8}$$

In (1.7), $g_{prior}(S)$ is a second order *Markov Random Field* (MRF):

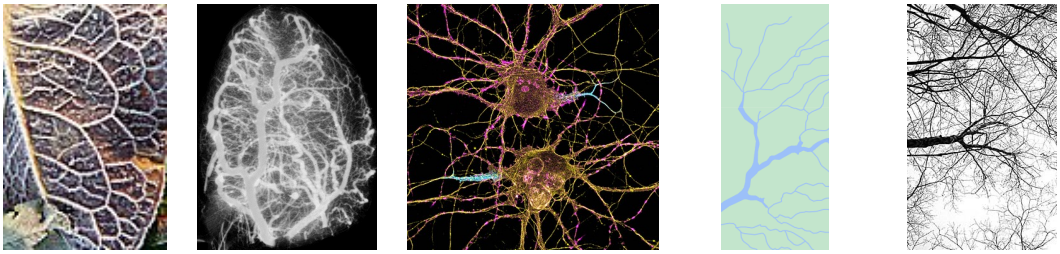
$$g_{prior}(S) = \frac{1}{Z} \prod_{(p,q) \in G} \psi_{pq}(x_p, x_q) \qquad \text{and} \qquad \psi_{pq}(x_p, x_q) \propto \exp(-\gamma\, w_{pq}\, [x_p \neq x_q]) \tag{1.9}$$

where Z is a normalization constant. In the literature, such approaches are called *Markov Random Field (MRF) regularization* [97] or *Conditional Random Field (CRF) regularization* [160], which is formally introduced in Section 1.2.1.

While MRF regularization is an important tool dealing with deficiencies of the appearance models, it must be used with caution as it inevitably introduces assumptions about the segment shape. For example, pairwise prior in (1.7) is known to be proportional to the length of segments boundary (assuming $w_{pq} = 1$) and thus has a *shrinking bias* [26], see Figure 1.2(e). While this prior may be appropriate for compact objects with relatively short boundaries, it is not suited for other common types of objects such as thin structures in Figure 1.2 and 1.3. The prevalence and practical significance of objects of such "non-compact" shapes motivate research in developing new tailored MRF/CRF regularization models incorporating the knowledge about such objects. Also, understanding the biases in the new and old regularization models is equally important to understand their limitations.

1.1.2 Continuous Formulations of Segmentation

The discrete/continuous classification of segmentation methods is based on the types of two important sets defining the segmentation problem, namely the set of segmentation variables



leaf vessels* blood vessels neurons** river bed*** tree branches

Figure 1.3: Examples of thin structures in nature. *Daniela_deGol, CC BY 4.0, via Wikimedia Commons; **Leterrier, NeuroCyto Lab, INP, Marseille, France CC BY 2.0, via flickr.com; ***Google Map Data ©2021.

and the set of their possible values. For example, in Section 1.1.1 both the set of variables $X = \{x_p \in \mathcal{L} \,|\, p \in V\}$ and the set of their values $\mathcal{L}$ (labels) are discrete (and finite). Such formulations are often referred to as *discrete-discrete* or simply *discrete*. Besides that, some formulations use discrete X & continuous labels $\mathcal{L}$ and continuous X & continuous $\mathcal{L}$. They are called *discrete-continuous* and *continuous-continuous* correspondingly. Below, we briefly introduce two types of representation, namely active contours and level sets. Both are initially formulated as continuous-continuous. However, to devise a practical optimization, the domain is often discretized yielding discrete-continuous formulations.

Active Contours (discrete-continuous)

There are well-known examples of discrete-continuous methodologies in segmentation including *active contours* (a.k.a. *snakes*) [138, 41, 313] and *random walker* [108, 63]. For shortness, we review the first one as the earlier of the two.

In Section 1.1.1, we described segmentation in terms of pixel-level discrete segmentation variables. Another approach is to *explicitly* model the segments' boundaries. Such explicit boundary model and the corresponding optimization are often called "*active contours*", also known as *snakes* [138, 41, 313]. The snake model is motivated by the standard continuous contour representation in 2D via function $f : [0,1] \to \mathbb{R}^2$ parameterized by argument $s \in [0,1]$: $f(s) = (x(s), y(s))$ where x and y correspond to image coordinates. The optimization functional $E(f \,|\, \boldsymbol{I})$ can be designed to attract a smooth or regular contour to image edges. The functional typically consists of two components, *i.e.* the image-based term $E_{image}(f|\boldsymbol{I})$ ensuring that the contour is aligned with the salient features of image $\boldsymbol{I}$

such as lines and edges; and the regularization term $E_{int}(f)$ ensuring the spline is smooth:

$$E(f \mid \boldsymbol{I}) = E_{edge}(f \mid \boldsymbol{I}) + E_{int}(f) \tag{1.10}$$

$$E_{image}(f \mid \boldsymbol{I}) = \int_0^1 -\|\nabla \boldsymbol{I}(f(s))\|^2 \,\mathrm{d}s \tag{1.11}$$

$$E_{int}(f) = \int_0^1 \alpha(s)\|f_s(s)\|^2 + \beta(s)\|f_{ss}(s)\|^2 \,\mathrm{d}s \tag{1.12}$$

where $\nabla I(f(s))$ is the image gradient at point $f(s)$, f_s and f_{ss} are the first and second order derivatives with respect to s, $\alpha(s)$ and $\beta(s)$ control the relative weight between the first and second order smoothness.

The question is how to optimize such continuous functionals. The actual active contour models or snakes discretize the domain of the contour parameter s resulting in a discrete-continuous optimization problem over $\boldsymbol{p} : V \to \mathbb{R}^2$:

$$E_{image}(\{\boldsymbol{p}_i\} \mid \boldsymbol{I}) = \sum_{i \in V} -\|\nabla \boldsymbol{I}(\boldsymbol{p}_i)\|^2 \tag{1.13}$$

$$E_{int}(\{\boldsymbol{p}_i\}) = \sum_i \alpha(i)\|\boldsymbol{p}_{i+1} - \boldsymbol{p}_p\|^2/h^2 \;+\; \beta(i)\|\boldsymbol{p}_{i+1} - 2\boldsymbol{p}_i + \boldsymbol{p}_{i-1}\|^2/h^4 \tag{1.14}$$

where $V = \{1, \ldots, N\}$ and $\boldsymbol{p}_i = f(s_i)$ for uniformly sampled $0 = s_1 < s_2 < \ldots < s_N = 1$, and $h = s_i - s_{i-1}$. There are many extensions to the model above that incorporate different application-specific terms in (1.13) or propose a better optimization [272].

As before, the regularization prior $E_{int}(\{\boldsymbol{p}_i\})$ corresponds to an MRF ensuring that the adjacent snake control points $\{\boldsymbol{p}_i\}$ do not vary too much. This MRF model incorporates the first and second-order priors corresponding to the *length* and *curvature* of the contour. Such a combination is called *elastic* prior. It is known that simple snake models tend to shrink, so it is recommended to initialize the snake outside of an object [272].

It is possible to define complex snakes. For example, a snake can consist of several disconnected splines which could be either open or closed. One can also incorporate additional constraints requiring intersection or touching of the splines, which can be used to model vessel bifurcations. However, a significant limitation of snakes is the fixed topology. In the example of vessels, this would require knowing all the bifurcations of the tree, which is impractical in large-scale problems.

Level-set Methods (continuous-continuous)

The explicit contour representation has its drawbacks, for example, it is challenging to change the topology of the curve or new reparameterization may be required if the shape changes dramatically [272]. *Level sets* [294, 285, 66] is an alternative representation of a closed contour where the curve is defined as zero crossing of a special *embedding* function $\phi(\boldsymbol{p}, t) : \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R}$ where $\boldsymbol{p} \in \mathbb{R}^2$ is the spatial coordinates and $t \in \mathbb{R}$ is time. For many desired contour properties (objective functions), one can straightforwardly express them in terms of ϕ. For example, the gradient descend for the standard contour length regularization objective, known as *curvature flow*, gives the following PDE for the implicit embedding function ϕ [272]:

$$\frac{\mathrm{d}\phi}{\mathrm{d}t} = g(\boldsymbol{I})|\nabla\phi| \operatorname{div}\left(\frac{\nabla\phi}{\|\nabla\phi\|}\right) + \nabla g(\boldsymbol{I}) \cdot \nabla\phi \tag{1.15}$$

where $g(\boldsymbol{I})$ is a generalized image edge potential such as (1.11). The first term (regularization) pushes the curve in the direction of its curvature. The second term pushes the boundary to the minimum of potential $g(\boldsymbol{I})$. Though level sets can easily change the topology of the contour, they still tend to suffer from local minima as they are based on local measures such as gradients, texture, color, *etc.*

1.1.3 Types of Image Segmentation

Definitions, such as (1.6) and (1.7), are incomplete as the color probability distributions $\Pr(\boldsymbol{I}_p \,|\, x_p)$ or $\Pr(x_p \,|\, \boldsymbol{I}_p)$ are unknown in practice. To make the problem well defined, additional constraints are required. Depending on the type of these constraints, the image segmentation can be further classified as follows.

Interactive image segmentation: For any particular image that needs to be segmented, the user provides a supervision input, such as partial labeling [172] as in Figure 1.2(b) or bounding boxes [240]. Labeled pixels allow statistical estimation of probabilities $\Pr(\boldsymbol{I}_p \,|\, x_p)$. We denote such estimates (models) as $P(\boldsymbol{I}_p \,|\, x_p)$. Such estimates can be used in (1.6) as in [31].

Unsupervised segmentation: In unsupervised segmentation, the user's input is minimized. For example, the user may only specify the number of segments. The goal of the system then is to partition the pixels of the image(s) into several segments. Like clustering,

these methods rely on carefully designed objective functions incorporating low-level cues, such as colors and edge contrast [97, 195, 296, 253, 6].

Specialized methods detecting certain object types are common. For example, the objects of interest in medical images can often be characterized by the brightness or intensity of the pixels, *e.g.* contrast CT. In these cases, simple thresholding methods are often employed [219, 304]. Another example is vessel segmentation, where the shape of vessels is a key feature in various detection methods. For example, Frangi et al. [90] proposed an unsupervised method of tubular structure detection based on hessian eigenvector analysis.

Since there is no access to supervision about segments (like in the interactive segmentation) for generic images, the unsupervised methods rely on the image data and low-level cues to group pixels into segments. In this setting, the segmentation is cast as a clustering problem. For example, the standard K-means can be formulated in terms of (1.6):

$$u_p^l = u_p^l(S) = \|\boldsymbol{I}_p - \boldsymbol{\mu}_l\|^2$$

where the cluster mean for label l is $\boldsymbol{\mu}_l = \sum_q \boldsymbol{I}_q x_q^l \Big/ \sum_q x_q^l$. Note, the corresponding energy as a function of segments is not unary as coefficients u_p^l depend on the current segmentation S.

Semantic segmentation (fully-supervised): The goal is to train a system that would assign a semantic label from a fixed set to any pixel of an unseen image. The set of semantic classes is application-specific. For example, for common images, it can include classes such as *person*, *tv*, *char*, *dog*, *etc.* In medical images, it can be *cancerous* vs *normal tissue*.

In order to build such a system, it is often assumed that a dataset of labeled images is provided such that for each pixel $p \in V$ of each image $\boldsymbol{I}$ of the dataset, a ground truth semantic label $y_p(\boldsymbol{I})$ is provided, see Figure 1.4. Given such a dataset, one is to create a system $\sigma_p : \mathcal{I} \to \Delta_{\mathcal{L}}$, via machine learning [157, 7, 176], that for each pixel on a new image $\boldsymbol{I} \in \mathcal{I}$ outputs a semantic label. We assume that such systems output a categorical distribution $\sigma_p(\boldsymbol{I}) \in \Delta_{|\mathcal{L}|}$ over labels at each pixel $p \in V$. Such distributions could be though as probability estimates $\sigma_p^l(\boldsymbol{I}) \approx \Pr(y_p(\boldsymbol{I}) = l \,|\, \boldsymbol{I})$ corresponding[1] to $u_p^l = -\log \sigma_p^l(\boldsymbol{I})$ in (1.5). The common cross-entropy (or negative log-likelihood) loss is:

$$\sum_p \sum_l -y_p^l \log \sigma_p^l \quad \to \quad \min_{\sigma \in \mathcal{F}} \tag{1.16}$$

[1] Unlike (1.5) and (1.8), here we make a different assumption: $\Pr(S|I) = \frac{1}{Z} \prod_{p \in V} \Pr(x_p|\boldsymbol{I}) g_{prior}(S)$.

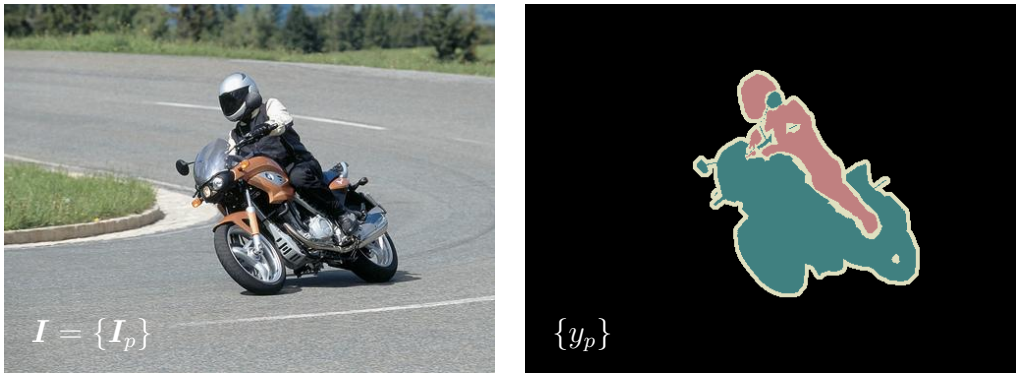


Figure 1.4: Example of semantic segmentation data [84]. The original image on the left is to be partitioned into semantic segments on the right. The semantic labels are color-coded. Black corresponds to the background, pink represents person label, cyan is motorbike.

where for simplicity we omitted explicit dependence of x_p^l and σ_p^l on $\boldsymbol{I}$ and only consider single image $\boldsymbol{I}$. In theory, the above equation is summed over all images in the dataset. Minimization is often taken over a parametric family $\mathcal{F} = \{\sigma(\theta)\}$ where θ is a set of parameters resulting in the following optimization problem:

$$\sum_p \sum_l -y_p^l \log \sigma_p^l(\theta) \quad \rightarrow \quad \min_\theta . \tag{1.17}$$

Having found the optimal θ^*, one segments any new image $\boldsymbol{I} \in \mathcal{I}$ via

$$x_p \;=\; \arg\max_l \; \sigma_p^l(\boldsymbol{I}, \theta^*). \tag{1.18}$$

See the details on the standard optimization methods (for neural networks) in Section 1.5.3.

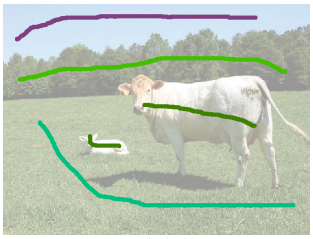
Weakly supervised semantic segmentation: In this setting, instead of labels for each pixel in a dataset, partial labeling [172, 279, 187] or even image-level tags [259, 327, 14, 217, 148] are only available, see Figure 1.5. With less supervision, systems trained on such datasets typically underperform their fully supervised counterparts. An advantage, however, is that the dataset acquisition process is much cheaper.

1.1.4 Objectives: Energies, Losses, Criteria, *etc.*

The literature uses a variety of terms referring to various objectives. These terms include *energy, loss, criterion, cost, error, fitness measure, quality, accuracy*, *etc.* In essence, all


original image


full annotation


scribbles

{cattle,
forest,
grass,
sky}

tags

Figure 1.5: Example of a weakly supervised semantic segmentation data. Figures from [172] © 2016 IEEE.

of these refer to an objective function or functional of some optimization problem. The particular use of the terminology is mostly historical and determined by the common practice of the respective scientific fields. For example, in MRF/CRF optimization such objectives are often called "energies" due to the link to Gibbs distribution and physics [97]. In machine learning, such objectives are often called "losses", which have roots in statistical theory [301]. Clustering analysis often refers to "criteria".

The contributions of this thesis are placed in the intersection of several fields, hence we use these terms interchangeably.

1.2 MRF/CRF Regularization Objectives

The body of research on regularization and its applications is vast. In the early days of computer vision (the 1960s – 1980s), approaches relied on raw data processing without any regularization. For example, in the optical flow and stereo problems, the research used simple techniques such as matching patches of an image [234], analyzing the image gradients [178] and phase correlation [156]. Simple filtering methods were developed for edge detection including Sobel [261], Prewitt [233] filters, and Canny [39].

In the context of image segmentation, motivated by biological visual systems, many early image segmentation approaches were based on contrast edges. The fundamental problem with the edges is that they rarely form closed boundaries. Thus, the continuation of the edges was required to obtain a segmentation. In addition, edges often led to spurious boundaries that needed to be removed. Both edge continuation and false boundaries elimination were addressed via heuristic search and dynamic programming [94], relaxations techniques [238] or curve fitting via extensions of the Hough transform [79].

Another group of early approaches to segmentation relied on thresholding techniques, such as (1.4), with application in medical [304, 219, 94] and natural scenes [208, 94]. Clustering of pixels can be seen as a generalization of thresholding methods. In the context of image segmentation, data points are pixels and feature vectors are multiple image channels that can include pixels' intensity, color, texture, *etc.* [299, 59, 94]. Such methods used heuristic post-processing ensuring robustness to noise and adherence of the object or segment boundaries to the image edges.

The researchers realized early that the object segmentation would "*be impossible ... without considerable analysis of shape and surface properties*" [222]. These methods were based on region grouping heuristics where regions with similar properties merged [196, 35, 80]. However, these methods assumed simplistic scenes composed of only combinations of a small number of simple geometric shapes, such as polyhedra and cylinders.

The next period (the 1980s – 2012) is characterized by using MRF/CRF regularization as a formal mathematical framework incorporating prior knowledge about the application domain. In the following period, the approaches based on deep artificial neural networks have gradually begun to dominate the computer vision field. The next section reviews a generic concept of MRF/CRF regularization as well as its several applications.

1.2.1 Markov Random Fields Basics

An influential work of Geman and Geman [97] showed that many computer vision objectives, including (1.7) as well as spline/mesh fitting, texture modeling, images restoration and segmentation, have natural probabilistic interpretation via Gibbs distribution

$$\Pr \quad \propto \quad \exp(-E/T) \tag{1.19}$$

where E is an energy function, such as (1.7), and constant T is a temperature parameter of the Gibbs distribution. Thus, Geman and Geman showed that by optimizing such models one inevitably makes certain probabilistic assumptions.

Specifically, let $V = \{1, 2, \dots, N\}$ be a set of *sites*. Depending on the application, sites V can refer to indices of pixels, voxels, or data points in general. We assume that there is a symmetric irreflexive neighborhood relationship between sites $N \subset V \times V$. Let $G = \{G_p \,|\, p \in V\}$ such that $G_p = \{q \mid (p, q) \in N\}$. Note that the pair (V, N) (or equivalently (V, G)) is an undirected graph.

Let $X = \{X_p \,|\, p \in V\}$ denote a set of random variables with values $X_p \in \mathcal{L}$. In this thesis, we focus on the case of discrete $\mathcal{L}$. In that case, $\mathcal{L}$ is called the set of labels. Let $\Omega = \{\boldsymbol{x} = (x_1, \dots, x_N) \mid x_p \in \mathcal{L}, p \in V\}$ be a set of all configurations of random variables X. Such configuration $x \in \Omega$ is called labeling. A set $C \subset V$ is called a *clique* if all pairs of sites in C are neighbors in G. Let $\mathcal{C}$ denote the set of all cliques.

Definition: X is an MRF with respect to G if

$$\Pr(X = \boldsymbol{x}) > 0 \quad \text{for all } x \in \Omega, \text{ and} \tag{1.20}$$

$$\Pr(X_p = x_p \,|\, X_q = x_q, q \neq p) = \Pr(X_p = x_p \,|\, X_q = x_q, q \in G_p) \quad \text{for all } p \in V. \tag{1.21}$$

The functions $\Pr(X_p = x_p \,|\, X_q = x_q, q \in G_p)$ are called *local characteristics* of the MRF, which uniquely define joint probability $\Pr(X = \boldsymbol{x})$. Equation 1.21 describes local property of the MRF. It informally states that any site depends on its neighbors only.

Now, the Gibbs distribution relative to G is a probability measure π on Ω such that

$$\pi(\boldsymbol{x}) = \frac{1}{Z} \exp(-E(\boldsymbol{x})/T) \quad \text{and} \quad E(\boldsymbol{x}) = \sum_{C \in \mathcal{C}} E_C(\boldsymbol{x}_C) \tag{1.22}$$

where Z is the normalization constant, $E(\boldsymbol{x})$ is the energy function corresponding to distribution π, and $\boldsymbol{x}_C = (x_p \,|\, p \in C)$ is a restriction of $\boldsymbol{x}$ on clique $C \subset V$, and E_C is a general MRF energy *potential* defined on clique C.

Theorem 1.1 (Clifford and Hammersley [57]). *X is a Markov Random Field with respect to G if and only if $\pi(x) = \Pr(X = w)$ is a Gibbs distribution with respect to G.*

The order of the MRF X is defined as the maximal clique size $\max_{C \in \mathcal{C}} |C|$.

Many problems in compute vision are formulated as the estimation of distribution parameters. For example, given observed variables $\boldsymbol{I}$ (*e.g.* RGB image pixels or voxel intensities in computer tomography), the *maximum likelihood estimator* (MLE) states that the unknown parameters X should be chosen to maximize the likelihood function

$$\boldsymbol{x}^* = \arg\max_{\boldsymbol{x}} \Pr(\boldsymbol{I} \mid X = \boldsymbol{x}) \tag{1.23}$$

where X are parameters of distribution (*e.g.* pixel labeling in case of segmentation).

In Bayesian statistics the *maximum a posteriori* (MAP) parameter estimates are the parameters maximizing the a posteriori distribution of the parameters

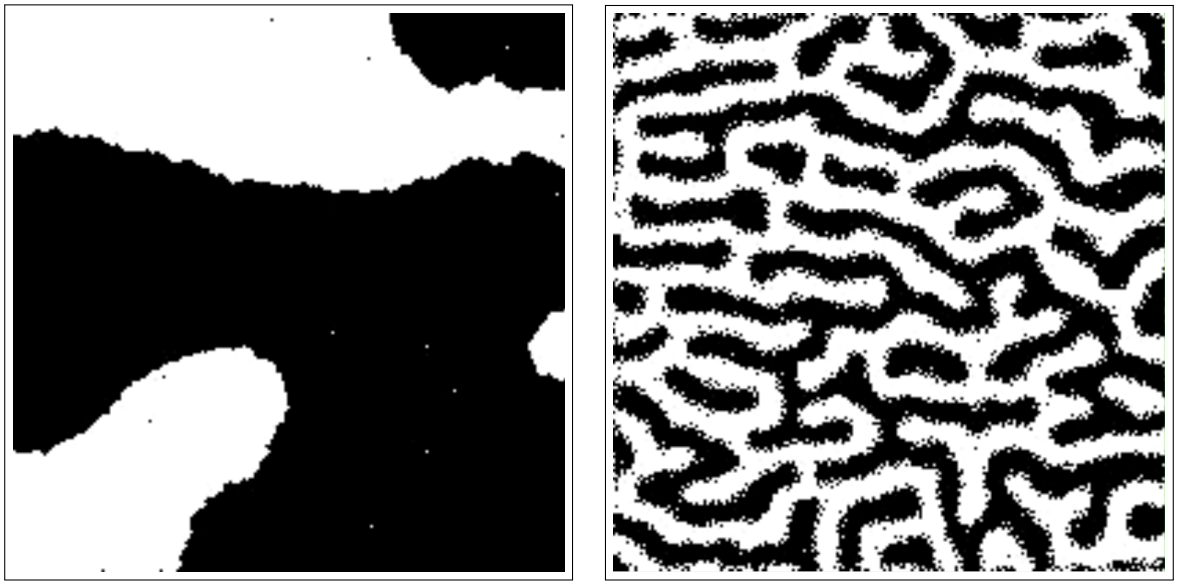
$$\arg\max_{\boldsymbol{x}} \Pr(X = \boldsymbol{x} \mid I) = \arg\max_{\boldsymbol{x}} \Pr(\boldsymbol{I} \mid X = \boldsymbol{x}) \Pr(X = \boldsymbol{x}) \tag{1.24}$$

where a prior distribution $\Pr(X = \boldsymbol{x})$ is often chosen to be an MRF.

The term $\Pr(\boldsymbol{I} \mid X = \boldsymbol{x})$ measures the adherence of the observed data to their model. In the following, we refer to $\Pr(\boldsymbol{I} \mid X = \boldsymbol{x})$ and corresponding Gibbs energy term as the *data (fidelity) model or term*. The distribution $\Pr(X = \boldsymbol{x})$ and the corresponding Gibbs energy terms reflect the prior knowledge about labeling x and are referred to as *regularization*.

As noted above, an MRF regularization imposes certain probabilistic assumptions on the hidden variables. Such assumptions significantly affect the result of energy minimization and may or may not be appropriate for a particular application. It is important that users of MRF regularization understand these assumptions and associated biases. It is possible to theoretically describe these biases in some cases. Additionally, one may generate samples from the corresponding prior distribution to "intuit" the effect of the MRF prior. Figure 1.6 shows samples from two different priors demonstrating drastically different behavior affecting the choice of regularization, depending on the application.

Conditional Random Fields: The term *Conditional Random Field* (CRF) [160] accounts for the fact that the exact values of unary u_p and pairwise w_{pq} in (1.7) depend on the image $\boldsymbol{I}$ itself. If the image $\boldsymbol{I}$ is treated as observed random variables and the labeling X are considered as hidden random variables, then the exact formulation of a markovian property requires conditional probability. Formally, the pair of variables $(\boldsymbol{I}, X)$ is called a CRF if the conditional probability $\Pr(X \mid \boldsymbol{I})$ is an MRF. While the difference between MRF and CRF may be technically important, we use those terms interchangeably in this thesis. In all such cases, the exact meaning of the field will be clear from the context.



(a) Potts model of [31] (b) curvature model of [206]

Figure 1.6: Samples from different segmentation shape priors obtained via the Gibbs sampler [97] from the Gibbs distribution (1.22). The Potts model [97, 31] shown in (a) tends to produce compact objects with short boundaries. The curvature model [206] shown in (b) produces elongated objects with low curvature of the boundary.

1.2.2 Applications of MRF Regularization

Image Segmentation

Boykov and Jolly [31] demonstrated that the interactive image segmentation problem, see Figure 1.2, with Potts potential (1.37) and $L = 2$ classes can be solved exactly via *graph cut*. They assumed that a subset of pixels $V_l \subset V$ is labeled (by a user). We denote these pixel labels as $l_p \in \mathcal{L}$ for $p \in V_l$. Their optimization objective (energy) is

$$\begin{cases} \min\limits_{\boldsymbol{x}} \sum\limits_{p \in V} \sum\limits_{l \in \mathcal{L}} -x_p^l \ln \tilde{q}(\boldsymbol{I}_p \,|\, X_p = l) \;+\; \sum\limits_{q \in G_p} w_{pq}[x_p \neq x_q] & \text{such that} \\ x_p \;=\; l_p & \text{for all } p \in V_l \end{cases} \tag{1.25}$$

where $w_{pq} = \exp(\|\boldsymbol{I}_p - \boldsymbol{I}_q\|^2 / \tilde{\sigma}^2)$ with $\tilde{\sigma}^2 = \frac{1}{|V|} \sum_{p \in V} \|\boldsymbol{I}_p - \boldsymbol{I}_q\|^2$, and $\tilde{q}(\boldsymbol{c} \,|\, l)$ is the estimate of the distribution of color intensities $\boldsymbol{c} \in \mathbb{R}^3$ inside segment with label $l \in \mathcal{L}$, which is estimated from the labeled pixels set V_l. In case of the multiple labels $L > 2$, one can use α-expansion [29], TRW-S [152], *etc.*, see Section 1.2.4.

The popularity of the Potts prior in the literature necessitates understanding its properties and, hence, possible shortcoming or limitations on the applications. As noted in the Section 1.1.1, the term $\sum_{q \in G_p} w_{pq}[x_p \neq x_q]$ with positive $w_{pq} > 0$ penalizes the number of discontinuities, *i.e.* places where neighboring sites have different labels, effectively shifting the preference toward "smoother" solutions. In other words, the Potts prior has a bias to compact objects with short boundaries, also known as shrinking bias [152] demonstrated in Figure 1.2(e). While in many cases such property is appropriate, there are many vision applications where Potts prior is ill-suited, particularly in the area of thin structures detection such as neurons, blood vessels, edges detection, *etc.*, see Figure 1.3.

Instead of the Potts model, the works of [246, 265, 117, 82, 213, 252, 206] used the curvature of the boundary as regularization for binary segmentation problems. Employing curvature helped to alleviate the shrinking bias and enable interesting applications such as thin objects segmentation and inpainting. However, these methods often require computationally expensive optimization and/or suffer from discretization artifacts.

Optical Flow

In the optical flow, the method of Horn and Schunck [124] introduced a smoothness term R, which can be considered as a (continuous) MRF regularization, into their objective. Assuming $I : \mathbb{R}^3 \to \mathbb{R}$ is a (continuous) 3D image with the third dimension corresponding to

time, the unknown optical flow $\vec{V} = [u(x, y, t), v(x, y, t)]$ is defined for each image location (x, y) at time t. Specifically, their objective is

$$E(u, v) = \ell(u, v, I) + \alpha R(u, v), \tag{1.26}$$

$$\ell(u, v, I) = \iint \left(\frac{\partial I}{\partial x} u + \frac{\partial I}{\partial y} v + \frac{\partial I}{\partial t} \right)^2 \mathrm{d}x \, \mathrm{d}y, \tag{1.27}$$

$$R(u, v) = \iint \|\nabla u\|^2 + \|\nabla v\|^2 \, \mathrm{d}x \, \mathrm{d}y, \tag{1.28}$$

where α is a smoothness parameter and $\nabla = \partial/\partial x + \partial/\partial y$ is the spatial gradient operator. Term $R(u, v)$ encourages a "smooth" optical flow that does not change much between neighbor points. This assumption is reasonable for compact objects with relatively short boundaries and moving in almost rigid fashion. Thus, the term $R(u, v)$ incorporates the prior knowledge of the particular application domain. The algorithm solves the minimization problem via Euler-Lagrange equations and then devises a discretized iterative scheme.

The smoothness prior R above tends to "over-smooth" the result, particularly at discontinuities where moving objects have a boundary. Instead of desired sharp change in optical flow value, the method yields smooth "slow" transitions. To address this smoothing bias, a simple modification was proposed [21, 20] where the regularization (1.28) becomes

$$\iint \rho(\|\nabla u\|) + \rho(\|\nabla v\|) \, \mathrm{d}x \, \mathrm{d}y \tag{1.29}$$

such that ρ is a robust potential, *e.g.* $\rho(\tau) = \min(\tau^2, \beta)$ for $\beta > 0$. Black and Anandan [20] show that such modification corresponds to the line process of Geman and Geman [97] for discontinuity detection. Once the change of the optical flow reaches a plateau of robust potential, it is considered a discontinuity with a fixed penalty. Thus, the robust potential does not over-penalize the discontinuities and does not force the solution where the data strongly suggests otherwise.

Stereo Correspondence and 3D Surface Estimation

Stereo correspondence is similar to the optical flow problem. In stereo, we are given a pair of images obtained by cameras from two different viewpoints. Because of the camera displacement, the objects on the image are also displaced such that close objects are displaced more than distant ones. This effect is called parallax. The task is to determine the displacement of corresponding pixels in the image pair and, hence, determine the distance to the objects. It is often assumed that the displacement is restricted to the horizontal

direction. Stereo correspondence mimics the way humans perceive depth. Stereo is an important building block in 3D reconstruction.

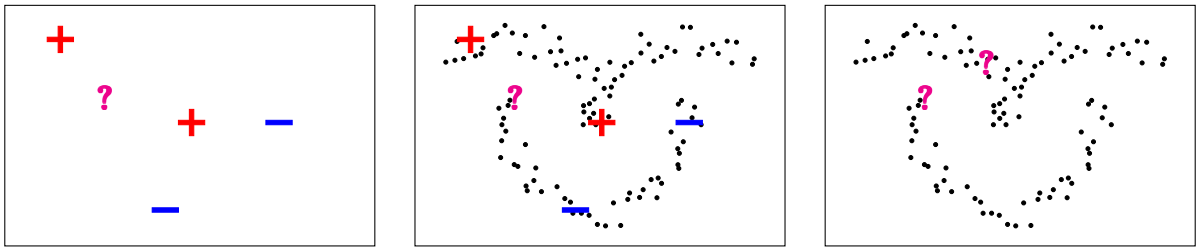
As the other problems we discussed above, stereo is typically formulated as an optimization problem as well. The objective is similarly often composed of two terms, *i.e.* the data term and regularization. The data term relies on the assumption that corresponding pixels (patches) in the two images have a similar appearance (color, texture, *etc.*). The regularization term ensures that neighboring pixels are displaced similarly. The displacement or disparity is denoted by $\boldsymbol{d}_p \in \mathbb{Z}^2$ for each pixel $p \in V$, and the coordinates of pixel p are denoted by boldface $\boldsymbol{p} \in \mathbb{Z}^2$. Then, the basic energy can be formulated as follows

$$E(\{\boldsymbol{d}_p\} \,|\, \boldsymbol{I}_L, \boldsymbol{I}_R) \;\; = \;\; \sum_p (\boldsymbol{I}_L(\boldsymbol{p}) - \boldsymbol{I}_R(\boldsymbol{p} + \boldsymbol{d}_p))^2 \;\; + \;\; \sum_{q \in G_p} w_{pq} \, [\boldsymbol{d}_p \neq \boldsymbol{d}_q] \tag{1.30}$$

where $\boldsymbol{I}_L$ and $\boldsymbol{I}_R$ are the stereo pair of images. The extensions include handling of occlusions, providing symmetric constraints to both left and right images, using a convex regularization term (*e.g.* total variation) [131], or robust regularization [20], *etc.*, see [272].

If the neighborhood structure G only contains horizontal edges (scanlines), that is the pixels in different rows do not interact with each other, then a simple dynamic programming approach can globally minimize the energy [127] such as (1.30). Moreover, if G is a tree (a graph without loops) then the dynamic programming can also give the global minimum [292]. In general, the problem of (1.30) is NP-hard and approximate methods are typically used, for example, graph cuts [28, 150].

Solving stereo correspondence allows perceiving the depth at each pixel on the image. This is a form of 3D vision or reconstruction. One of the applications of (multi-view) stereo is estimating globally consistent 3D models from images obtained from known viewpoints. This is an old computer vision problem with a rich body of research, see [272] for a review. One important aspect of those reconstruction algorithms is the employed shape prior, which allows obtaining smooth and denoised surfaces in 3D. One popular optimization technique used in [260, 297, 118, *etc.*], graph cut, results in object surfaces that tend to shrink inwards [272]. Due to the equivalence of graph cut to the binary Potts model and close relation to the minimal surfaces [151, 152], this is an example of the shrinking bias phenomenon. Olsson and Boykov [211] in the context of *tangential approximation of surfaces* from a point cloud proposed a new optimization framework that includes accurate integral absolute and squared curvature estimation and regularization. The advantage of this approach is the absence of shrinking bias.



(a) supervised learning (b) semi-supervised learning (c) unsupervised learning

Figure 1.7: Three Levels of Supervision: In the supervised setting (a), the data point marked with the question mark would be classified as negative class (red). On the other hand, the presence of unlabeled data points (black dots) in (b) reveals a non-linear structure of data, which may change the prediction for the questioned point. In an unsupervised setting (c) one is to separate the data points into disjoint clusters based on some grouping criteria. Hence, the problem is to determine if the two questioned points belong to the same cluster.

Semi-supervised Learning

Often, it is convenient to cast computer vision problems in terms of machine learning.

Let us consider a *semi-supervised* classification problem. Given a labeled data set $F_l = \{\boldsymbol{f}_p \mid p \in V_l\}$, $Y = \{y_p \mid p \in V_l\}$ and an unlabelled dataset $F_u = \{\boldsymbol{f}_p \mid p \in V_u\}$ where $\boldsymbol{f}_p \in \mathbb{R}^n$ is the feature vector of data point $p \in V_l \cup V_u$ and $y_p \in \mathcal{L}$ is the ground truth label of data point $p \in V_l$, the problem is to find a decision function $g \in \mathcal{G}$ from a *hypothesis space* $\mathcal{G} \subset \{g : \mathbb{R}^n \to \mathbb{R}^{|\mathcal{L}|}\}$ according to a given *loss function* ℓ_{semi}:

$$\arg\min_{g \in \mathcal{G}} \ell_{\text{semi}}\big(g, F_l, L, F_u\big) \tag{1.31}$$

For a hypothesis g, also known as *scoring* function, a prediction $\hat{y}$ for a feature vector $\boldsymbol{f}$ is

$$\hat{y} \;=\; \arg\max_{l \in \mathcal{L}} \left[g(\boldsymbol{f})\right]_l \tag{1.32}$$

where $[\boldsymbol{v}]_l$ is the l-th component of some vector $\boldsymbol{v} \in \mathbb{R}^{|\mathcal{L}|}$.

Note, that if $T_u = \emptyset$ the problem reduces to *supervised* learning. On the other hand, if $T_l = \emptyset$ the problem reduces to *unsupervised* learning. The comparison of different learning settings is illustrated in Figure 1.7.

The design of the loss function ensures that the optimal scoring function evaluated on training set inputs F_l predicts consistently with the training set labels Y. In addi-

tion, the availability of unlabeled dataset F_u may provide new information about the data distribution that can improve the performance of the resulting classifier.

Often, the semi-supervised loss can be decomposed into supervised part ℓ_{s} dealing with the labeled set T and the unsupervised part ℓ_{u} dealing with unlabeled data $F = F_l \cup F_{\mathrm{u}}$:

$$\ell_{\mathrm{semi}}\big(g, F_l, L, F_u\big) \;=\; \ell_{\mathrm{s}}(g, F_l, Y) \;+\; \ell_{\mathrm{u}}(g, F) \tag{1.33}$$

This approach closely resembles the maximum *a posteriori* principle discussed in Section 1.1.1. Therefore, one can use an MRF probability prior π to construct the unsupervised term of the loss:

$$\ell_{\mathrm{u}}(g, F) \;\propto\; -\ln \Pr(X = g(F)) \;=\; -\ln \pi(g(F)) \tag{1.34}$$

where X is the set of random variables denoting the labeling of data points.

1.2.3 Properties of Pair-wise and Higher-order MRF Models

Potts Model

Let us consider a few typical examples of MRFs. The Ising [132] model is common in physics and explains the properties of ferromagnetism. The energy of the system is

$$\sum_{p \in V} \sum_{q \in G_p} J_{pq} x_p x_q \;\; + \;\; \sum_{p \in V} u_p x_p \tag{1.35}$$

where each site p has an unknown spin label $x_p \in \{+1, -1\}$, the sign of $J_{pq} \in \mathbb{R}$ corresponds to magnetic properties of the material, and $u_p \in \mathbb{R}$ describes the interaction of the spin with an external magnetic field at site $p \in V$. G_p corresponds to nearest neighbors; *e.g.* in 2D a site with coordinates (p, q) may have neighbors $\{(p+1, q), (p-1, q), (p, q+1), (p, q-1)\}$.

A general *pair-wise* energy can be define as follows:

$$\sum_{p \in V} \sum_{q \in G_p} \psi_{pq}(l, k) x_p^l x_q^k \;\; + \;\; \sum_{p \in V} u_p^l x_p^l \tag{1.36}$$

where each site $p \in V$ has a label $x_p \in \mathcal{L} = \{0, 1, \ldots, L-1\}$, unary potentials $u_p^l \in \mathbb{R}$, pair-wise potentials $\psi_{pq}(l, k) \in \mathbb{R}$, and indicator variables $x_p^l = [x_p = l]$. Such energy is non-submodular and the corresponding minimization problem is NP-hard.

The Potts model is a special case of pair-wise energy and a generalization of the Ising model to multiple spin labels. It corresponds to $\psi_{pq}(l,k) = w_{pq}[l \neq k]$. The energy is

$$\sum_{p \in V} \sum_{q \in G_p} w_{pq}[x_p \neq x_q] \quad + \quad \sum_{p \in V} \sum_{l \in \mathcal{L}} u_p^l x_p^l \tag{1.37}$$

where $w_{pq} \geq 0$ is the constant penalty (weight) for discontinuity of labeling o f the pair (x_p, x_q). The Potts energy (1.37) is still non-submodular and the optimization is NP-hard. However, in the case of $|\mathcal{L}| = 2$ the Potts model is submodular (and can be optimized by graph cuts) [159, 31, 152]. This follows from the equivalent submodularity criterion [152]:

$$\forall p \in V \;\; \forall q \in G_p : \qquad \underbrace{E_{pq}(0,0) + E_{pq}(1,1)}_{=0} \;\; \leq \;\; \underbrace{E_{pq}(0,1) + E_{pq}(1,0)}_{=2w_{pq} \geq 0} \tag{1.38}$$

where we defined $E_{pq}(x_p, x_q) = w_{pq}[x_p \neq x_q]$.

The Potts model has many applications in signal processing and computer vision, including image segmentation [31, 240] as in (1.7), correspondence [28, 29], semantic segmentation [153, 52], image restoration [97], geometric model fitting [129], *etc.*

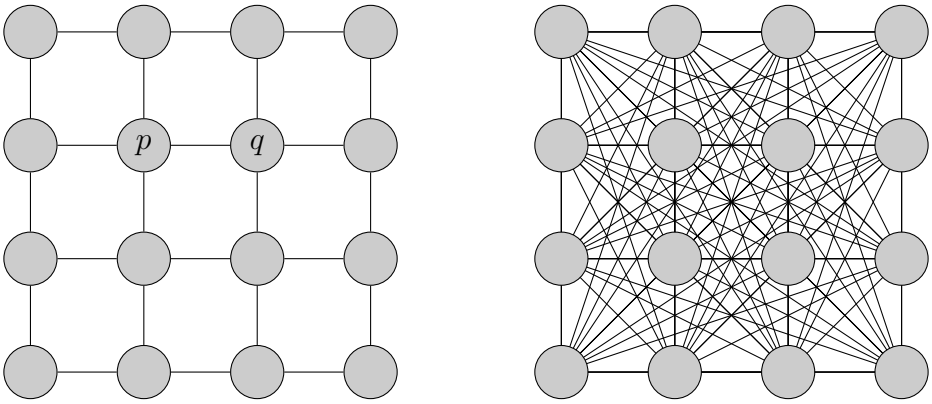
The Potts model has an insightful interpretation in the context of segmentation problems. Let *segment* S_l be the set of pixels that have label l: $S_l = \{p \in V \,|\, x_p = l\}$. Assuming the standard grid nearest neighborhood system G, the Potts term can be seen as an approximation of the length of the boundary between segments $\{S_l\}$ [26]:

$$\sum_{p \in V} \sum_{q \in G_p} [x_p \neq x_q] \;\; \approx \;\; \sum_{l \in \mathcal{L}} |\partial S_l| \tag{1.39}$$

where ∂S_l is the set of boundary points of segment S_l. To establish this connection, Boykov and Kolmogorov [26] used the Cauchy-Crofton formula from integral geometry relating the length of a curve with the number of its intersections by a random line. An immediate consequence is that Potts energy prefers segments with shorter boundaries. This phenomenon is called *shrinking bias* as it often treats thin details as noise and tends to shrink object boundary, see illustrations in Figure 1.2(e) and Figure 1.6.

Dense Potts Model

One feature of the Potts model—and MRF models in general—is its locality, *i.e.* conditional independence of non-neighboring pixels, see Figure 1.8(a). This property makes evaluation



(a) 4-Grid Connectivity (b) Dense Connectivity

Figure 1.8: Graphical model illustration. The sites are represented by circles/nodes. The edge between too nodes connects neighbors in the corresponding CRF/MRF. Each edge connecting nodes p and $q \in G_p$ corresponds to $w_{pq}[x_p \neq x_q]$ in (1.37). **(a)**: Grid CRF such as in [31, 29, 97, 240, etc]. **(b)** Dense CRF from [153, 53].

and (approximate) inference in such models tractable. In *dense* CRF Potts models each site is interacting with all other pixels, see Figure 1.8(b):

$$\sum_{(p,q)\in V\times V} w_{pq}[x_p \neq x_q] \quad + \quad \sum_{p\in V}\sum_{l\in\mathcal{L}} u_p^l x_p^l. \tag{1.40}$$

If the pair-wise potential w_{pq} has a certain property, then the efficient approximate inference, *i.e.* mean-field approximation, is possible in dense MRF models [153]. In particular, the pair-wise potential is required to be a Gaussian potential of the form

$$w_{pq} \quad = \quad \sum_{m=1}^{K} w^{(m)} k^{(m)}(\boldsymbol{f}_p, \boldsymbol{f}_q) \tag{1.41}$$

where $w^{(m)}$ are linear combination weights, $k^{(m)}(\boldsymbol{f}_p, \boldsymbol{f}_q) = \exp(-\frac{1}{2}(\boldsymbol{f}_p - \boldsymbol{f}_q)^\top \boldsymbol{\Lambda}^{(m)}(\boldsymbol{f}_p - \boldsymbol{f}_q))$ is a Gaussian kernel, $\boldsymbol{f}_p$ is a feature vector for each pixel $p \in V$, and $\boldsymbol{\Lambda}^{(m)}$ is a symmetric, positive-definite precision matrix, which defines the shape of kernel $k^{(m)}$. In [153], the feature vectors $\boldsymbol{f}_p$ are composed of RGB color channels $\boldsymbol{c}_p$ and 2D coordinates $\boldsymbol{p}_p$.

Despite the "dense" nature of dense CRFs, one may approximate local property by including the pixel coordinates into the feature vectors and designing matrices $\boldsymbol{\Lambda}^{(m)}$ such that the strength of the interaction between pixels decreases exponentially with the distance.

Veksler [293] noted that for large neighborhoods the dense Potts model approaches the *cardinality potential*, which only considers the volumes of the segments, suggesting weaker regularization properties than the regular sparse (nearest neighbor) Potts model. Indeed, our experiments show in Chapter 4 that dense CRFs are weaker regularizers compared to the regular (sparse) Potts model (1.37). The advantage of the dense CRFs is a smother energy surface profile and, hence, easier optimization. We show that with a new more advanced optimization, the regular Potts model may outperform dense CRFs.

Convex Pairwise Model

The Potts model (1.37) penalizes pairs of sites with different labels. The penalty does not depend on the labels but only on the fact that they are different:

$$\sum_{p \in V} \sum_{q \in G_p} w_{pq}[x_p \neq x_q] \qquad \text{for } w_{pq} \geq 0. \tag{1.42}$$

A more general model allows accounting for a more complex interaction between labels:

$$\sum_{p \in V} \sum_{q \in G_p} w_{pq}\psi(x_p, x_q) \qquad \text{for } w_{pq} \geq 0. \tag{1.43}$$

The optimization of the general pairwise potential (1.43) is NP-hard.

One important example is the case when the set of labels $\mathcal{L}$ is ordered and the pairwise potential has the form:

$$\psi(x_p, x_q) = \hat{\psi}(\iota(x_p) - \iota(x_q)) \tag{1.44}$$

where $\hat{\psi}$ is a convex function and $\iota(x)$ is the index of label $x \in \mathcal{L}$. The pairwise potential $\psi(x_p, x_q)$ only depends on the difference of labels' indices. Ishikawa [131] showed that such potentials can be optimized globally by a graph cut.

Curvature Regularization

It is widely known [123, 266, 70, 245, 265, 246, 32, 117, 213, 206] that curvature is a high-order regularization criterion that addresses many limitations of length as a pairwise regularizer, see Figure 1.6(b). Local *curvature* at point P on a curve is defined as the reciprocal of the radius of the osculating circle at P, a circle that most closely approximates the curve at P [146]. The local curvature of a straight line is zero.

Absolute curvature of curve C is the curvilinear integral of absolute local curvature κ

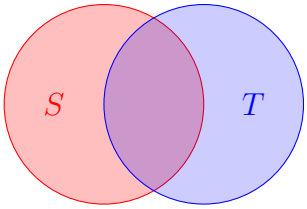
$$\mathcal{K} = \int_C |\kappa| ds.$$

Analogously, *squared curvature of curve* C is

$$\mathcal{K}^2 = \int_C \kappa^2 ds.$$

Curvature is a high-order functional. Indeed, to estimate local curvature one needs to know three infinitesimally close points on a curve required to fit an osculating circle. This suggests that cliques of the size of at least three are needed for regularization.

The curvature of the segment's boundary is not submodular. According to the definition of submodular set functions, it is enough to show that there exist segments $S, T \subseteq \Omega$ such that $E(S) + E(T) < E(S \cap T) + E(S \cup T)$. Consider the example on the right. Since both S and T are convex, the left hand side is 4π, while on the right hand side $E(S \cap T) = 2\pi$ and $E(S \cup T) > 2\pi$. Therefore, the curvature cannot be optimized with standard methods for submodular functions.



Curvature is a natural regularizer for complex objects and it has been widely explored in the past. In the context of *image segmentation* with second-order smoothness, it was studied by [245, 82, 265, 252, 246, 32, 117, 213, 206]. In these prior works, the segmentation problem with curvature regularization is formulated as minimization of energy

$$E(\boldsymbol{x}) = \sum_{p \in V} U_p(x_p) + \int_{\partial \boldsymbol{x}} |\kappa|^\alpha ds \tag{1.45}$$

where the set of pixels V is to be partitioned into two segments $(S, \bar{S} = V \setminus S)$ that are encoded via the indicator variables $\boldsymbol{x} = (x_p \,|\, p \in V)$ such that $x_p = [p \in S]$. The border (curve) between the segments is denoted $\partial \boldsymbol{x}$ and α is either 1 or 2 for absolute or squared curvature respectively. The first term in (1.45) is equivalent to the linear term in (1.37). The second term approximates the curvature integrals. Approximation of the integral curvature of the binary mask boundary is generally a hard problem. The aforementioned works study different ways to approximate the curvature integrals and to optimize them.

Curvature is also a popular prior in *stereo* or *multi-view reconstruction* [168, 212, 306]. Curvature has been used inside connectivity measures for analysis of diffusion MRI [198].

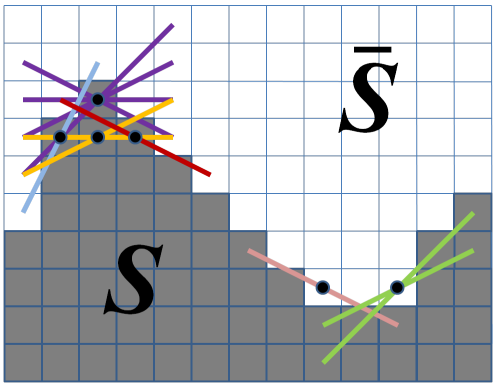


Figure 1.9: Squared curvature model of [206]. The term is designed to penalize cliques configuration 0-1-0 (segments on the left) and 1-0-1 (segments on the right). Where configuration 0-1-0 corresponds to positive curvature of the boundary of segment S and 1-0-1 corresponds to negative curvature. Figure from [206] © 2014 IEEE.

Curvature is also widely used for *inpainting* [4, 46, 206] and edge completion [113, 305, 3]. For example, *stochastic completion field* technique in [305, 198] estimates the probability that a completed/extrapolated curve passes any given point assuming it is a random walk with a bias to straight paths. Note that common edge completion methods use existing edge detectors as an input for the algorithm.

All of the mentioned works use discrete curvature except [212]. In [211, 212] Olsson *et al.* proposed a real-valued curvature model that is suitable for absolute and squared curvature estimation.

The model of [206] uses integral geometry to estimate curvature of segments boundaries using cliques of order three. The paper introduced the following energy term that approximates the squared curvature integral:

$$\sum_{(w,t,v)\in N} [x_w = x_v][x_w \neq x_t] = \sum_{(w,t,v)\in N} x_w(1-x_t)x_v + (1-x_w)x_t(1-x_v) \tag{1.46}$$

where w, t and v are image pixels, $x_p \in \{0,1\}$ is the binary segmentation variable, and $N \subset V^3$ is the special set of triple cliques (w,t,v) that satisfy $\boldsymbol{p}_t - \boldsymbol{p}_w = \boldsymbol{p}_v - \boldsymbol{p}_t$ where $\boldsymbol{p}_p$ is the 2D position vector of pixel $p \in V$. Moreover, for all $(w,t,v) \in N$ the distance $\|\boldsymbol{p}_w - \boldsymbol{q}_v\|$ is approximately the same (up to the discretization accuracy of the grid). The term is designed to penalize clique configurations 0-1-0 and 1-0-1, see Figure 1.9.

Despite being written as triple interaction, the curvature model (1.46) is equivalent to

$$\sum_{(w,t,v)\in N} x_t + x_w x_v - x_w x_t - x_v x_t, \tag{1.47}$$

which is a pairwise non-submodular term.

Optimization of curvature is an important and challenging problem. Even though there is a lot of works on curvature regularization for segmentation, there are many issues that have to be addressed. One group of issues is related to the imperfectness of the model (poor approximations of the curvature integrals). Other approaches, *e.g.* [82, 117, 246, 252], suffer from discretization artifacts. For example, [82] has very limited angular resolution while [117, 246, 252] are restricted to specific grid complexes. Proposed algorithms are often computationally expensive [117, 246, 252, 265, 206].

Due to the bias to straight lines, see Figure 1.6, the curvature is a natural regularizer for thin structures. One of the contributions of this thesis is a new thin structures detection approach employing curvature regularization, see Chapter 2.

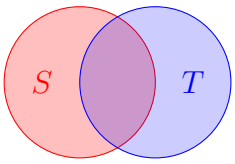
Convexity Regularization

Convexity has been identified as an important prior in human vision as well as in computer vision. Images of many natural objects have nearly convex shapes or silhouettes. Convex shapes are common in medical images. See [105, 106, 130, 175, 181]. Gorelick *et al.* [106] proposed convexity prior for discrete energy as follows

$$\sum_{l\in L} \sum_{\substack{\{w,t,v\}\subset l \\ w<t<v}} x_w(1-x_t)x_v \tag{1.48}$$

where L is the set of all lines that pass through at least three pixels on the image, and symbol $l \in L$ is the set of pixels belonging to such line, $w < t < v$ means that pixel $\boldsymbol{p}_t$ is between pixels $\boldsymbol{p}_w$ and $\boldsymbol{p}_v$. The term penalizes 1-0-1 configuration for each clique.

Similarly to the curvature, one can use the same example on the right to show that the convexity prior is not submodular. Let $E(A) = [A \text{ is convex}]$ be the indicator of the convexity of a set A. Then segments S, T yield $E(S) + E(T) < E(S \cap T) + E(S \cup T)$, which contradicts the definition of submodularity.



Convexity prior (1.48) has $O(n^2)$ terms where n is the number of pixels in the image. This makes even the evaluation of this term a computationally expensive operation,

let alone the optimization. However, it is possible to use the structure of set L and employ dynamic programming enabling efficient computation of term (1.48) and its derivatives [105, 106].

1.2.4 Combinatorial Optimization for MRF

Exact Optimization

Edmonds in [81] discovered the importance of so-called *submodular* binary energies. The notion of submodularity in combinatorial optimization is analogous to the notion of convexity in continuous optimization.

Let $(\mathbb{L}, \wedge, \vee)$ be a lattice. Function $F : \mathbb{L} \to \mathbb{R}$ is called *submodular* if for any $S, T \in \mathbb{L}$

$$F(S \wedge T) + F(S \vee T) \leq F(S) + F(T).$$

An important example of a *lattice* is a set with the intersection and union operations. Thus, set function $F : 2^V \to \mathbb{R}$ is submodular if for any $S, T \subseteq \Omega$

$$F(S \cup T) + F(S \cap T) \leq F(S) + F(T). \tag{1.49}$$

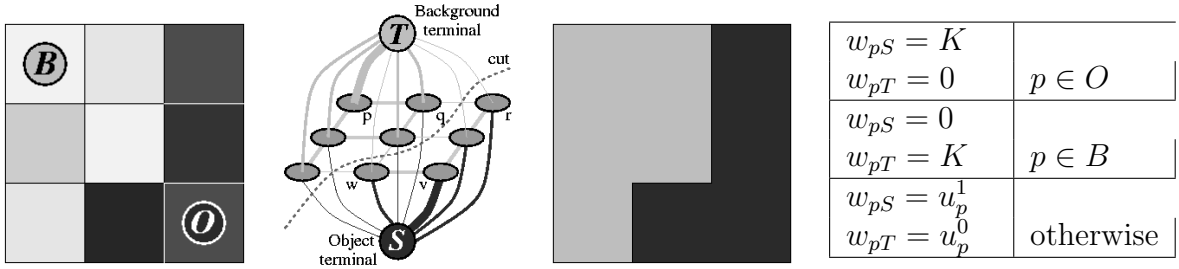
Energy (1.22) is an example of a set functions. Indeed, any labeling $\boldsymbol{x}$ defines set $S = \{p \mid p \in V \;\&\; x_p = 1\}$, and (1.22) can be expressed as a function of S

$$E(S) = E(\boldsymbol{x}).$$

Unfortunately, the proposed general polynomial algorithm for high order energies [110] have a prohibitive complexity for any real-word images.

Boykov [29, 31] popularized in the computer vision community the fact that submodular pairwise binary energies like (1.7) can be efficiently optimized via minimum graph cut, see Figure 1.10. Kolmogorov and Zabin [152] described a general class of energies that can also be optimized by graph cuts. This class includes up to third-order energies.

If the set of cliques $\mathcal{C}$ is a graph on vertices V without loops, the corresponding pairwise multi-label energy can be globally optimized by *dynamic programming* [229]. Ishikawa in [131] defined a class of pairwise multi-label energies that can be globally optimized by a minimum graph cut algorithm. In [71] it was shown that certain multi-label problems with inclusion constraints can be globally optimized via graph cuts.



$w_{pS} = K$ $w_{pT} = 0$	$p \in O$
$w_{pS} = 0$ $w_{pT} = K$	$p \in B$
$w_{pS} = u_p^1$ $w_{pT} = u_p^0$	otherwise

Figure 1.10: Graph construction for binary image segmentation of [31]. The figure shows a simple 3×3 image. The user input provides what pixel marked by "B" is part of the background segment, and pixel marked by "O" belongs to the object segment, see the image on the left. Each pixel corresponds to a graph node, see the middle left image. Neighboring pixel nodes are connected with weights as in (1.7). Two additional nodes, "T" and "S", are added to the graph and connected to each of the pixel nodes. The weights of edges connecting the terminal nodes are defined in the table on the right. K is a sufficiently large constant. The thickness of the edges on the image corresponds to their weight. One can verify, that any graph cut separating the nodes into sets $\{p \in V, x_p = 1\} \cup \{S\}$ and $\{p \in V, x_p = 0\} \cup \{T\}$ is equal to binary Potts energy (1.7). Thus, the min-cut delivers the global optimum of the energy. Figures from [31] © 2001 IEEE.

Approximate Combinatorial Optimization

Geman and Geman [97] optimized energy using *Monte-Carlo Markov Chain* (MCMC) [19] approach. In particular, they used *the Gibbs sampler* with annealing.

A large amount of work is devoted to approximate (multi-label) optimization without relying on the Monte Carlo (probabilistic) methods. Alpha-expansion [29] algorithm was proposed for a wide class of pairwise energies, which resulted in a variety of different extensions including *label costs* [72], *auxiliary cuts* [15], *local submodular approximations* [104], PBO [276]. General-purpose approximate pairwise optimization methods were proposed, namely LBP [229], TRW-S [149], QPBO [22], and others. For a detailed review and comparison please refer to [273, 137].

1.2.5 Continuous Optimization and Relaxations for MRF

Continuous relaxations of a discrete model enable continuous optimizations. A continuous function $f : [0,1]^{|V|} \to \mathbb{R}$ that coincides with the original discrete objective $F : \{0,1\}^{|V|} \to \mathbb{R}$ at all points of its discrete domain is called a *relaxation*. It holds that

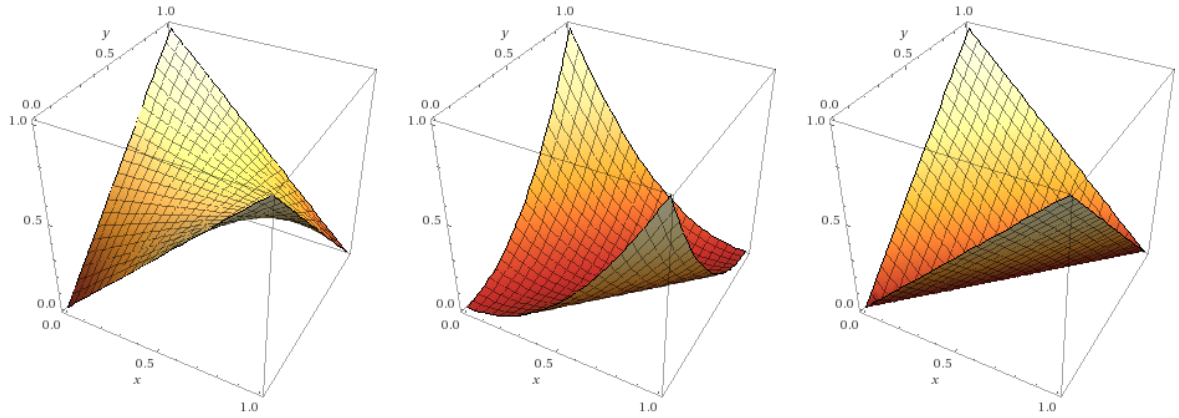
$$\min_{\boldsymbol{x} \in [0,1]^{|V|}} f(\boldsymbol{x}) \quad \leq \quad \min_{\boldsymbol{x} \in \{0,1\}^{|V|}} F(\boldsymbol{x}).$$

There is a continuum of different ways to relax a discrete model defined on finite domain $\{0,1\}^{|V|}$ to the continuous simplex $[0,1]^{|V|}$. Two properties of relaxations are important, that is *convexity* and *tightness*.

Basic Relaxations of Potts Model

Examples of *convex* relaxations of the Potts term $[x_p^l \neq x_q^l]$ in (1.37) are

- **square relaxation:**
$$\sum_l (x_p^l - x_q^l)^2, \tag{1.50}$$
- **total variation (TV) relaxation:**
$$\sum_l |x_p^l - x_q^l| \tag{1.51}$$



$x(1-y)+(1-x)y$ $(x-y)^2$ $|x-y|$

Figure 1.11: Different relaxations of Potts potential $[x \neq y]$ (binary case). LEFT: bilinear relaxation is tight and encourages discrete solutions, either $(0,0)$ or $(1,1)$. MIDDLE: square relaxation over smooths the labeling. RIGHT: total variation (TV) relaxation. Both square and TV relaxations are convex, while also equally satisfied when $x = y$ even at fractional values resulting in non-discrete solutions with *integrality gap*. The plots are obtained using Wolfram Alpha LLC. 2021. Wolfram|Alpha.

where the summation is over the set of labels $\mathcal{L}$. Convex relaxations are important in general optimization theory as they allow efficient global optimization. However, the solution of a convex relaxed problem is often significantly different from the solution of the original discrete problem. This difference is called the *integrality gap*. If the value of objective at the global solution of the relaxed problem is equal to the value of the objective (the integrality gap is zero) then the relaxation is called *tight*.

The solution to square relaxation can be interpreted as a solution to *a random walker* problem [108]. Both square and TV relaxation are convex but not tight. The exception is the TV relaxation in the case of binary labels, *i.e.* $|\mathcal{L}| = 2$.

The following relaxation is tight:

- **bilinear relaxation:**

$$\sum_l x_p^l (1 - x_q^l) = 1 - \sum_l x_p^l x_q^l \tag{1.52}$$

  where we assumed $\sum_l x_p^l = 1$ for all $p \in V$.

Ravikumar and Lafferty [235] showed that the following discrete quadratic problem

$$\sum_{p\in V}\sum_{k}\mu_{p,k}[x_p = k] \;+\; \sum_{(p,q)\in N}\sum_{k,l}\mu_{p,q,k,l}[x_p = k \,\&\, x_q = l] \;\;\rightarrow\;\; \max_{x\in\{0,1\}^{|V|}} \tag{1.53}$$

has a *tight* quadratic relaxation

$$\sum_{p\in V}\sum_{k}\mu_{p,k}x_p^k \;+\; \sum_{(p,q)\in N}\sum_{k,l}\mu_{p,q,k,l}x_p^k x_q^l \;\;\rightarrow\;\; \max_{x\in[0,1]^{|V|}} \qquad \textbf{subject to}$$

$$\sum_{k} x_p^k = 1 \qquad \forall p \in V.$$

The Potts model is a special case of the quadratic problem (1.53). Therefore, the bilinear relaxation (1.52) is tight as well. However, it is not convex.

Figure 1.11 illustrates these relaxations in the binary case.

Linear Programming

Consider general pair-wise minimization problem (1.36), which can be rewritten as

$$\sum_{p\in V}\sum_{q\in G_p}\psi_{pq}(l,k)x_{pq,lk} \;+\; \sum_{p\in V}u_p^l x_p^l \;\;\rightarrow\;\; \min_{x} \qquad \textbf{subject to} \tag{1.54}$$

$$x_p^l = \sum_{k\in\mathcal{L}} x_{pq,lk} \qquad p\in V, q\in G_p; l\in\mathcal{L}, \tag{1.55}$$

$$x_q^k = \sum_{l\in\mathcal{L}} x_{pq,lk} \qquad p\in V, q\in G_p; k\in\mathcal{L}, \tag{1.56}$$

$$x_{pq,lk}, x_p^l \in \{0,1\} \qquad p\in V, q\in G_p; l,k\in\mathcal{L}, \tag{1.57}$$

Note, that $x_{pq,lk} = x_p^l x_q^k$. This is an integer linear programming (LP) problem. An obvious linear relaxation problem is to relax discrete condition (1.57) to

$$x_{pq,lk}, x_p^l \in [0,1]. \tag{1.58}$$

One can show that the global optimum of such LP relaxation is a lower bound on the original energy (1.36). Though the LP solution can be much different from the solution of the original problem, it works well for some applications [272, 149]. This LP relaxation has been studied extensively, for an overview of methods, see [273]. In computer vision applications, the size of the problems poses the question of efficient solvers. Most iterative methods solve the dual problem, that is, they formulate a lower bound and then maximize it. For example, such problem can be formulated as a convex relaxation on trees [149, 300].

Variational Inference and Mean-field Approximation

Consider a Gibbs distribution corresponding to an MRF as in (1.19):

$$P(\boldsymbol{I}, X) = \frac{1}{Z} \exp\left(-E(\boldsymbol{I}, X)\right) \tag{1.59}$$

where Z is a normalization constant and $\boldsymbol{I}$ is the image. Here $\boldsymbol{I}$ are visible (observed) variables, variables X are hidden (unobserved) ones. The goal is to approximate the posterior distribution $P(X|\boldsymbol{I})$ of hidden variables X given the observed image $\boldsymbol{I}$. The problem of approximating the posterior distribution has been extensively studied and is known as *variational inference* [19].

Variational inference is based on the decomposition

$$\ln P(\boldsymbol{I}) = \ell(q) + \mathrm{KL}(q\|p) \tag{1.60}$$

where $\ln P(\boldsymbol{I})$ is the probability of observed variables called *evidence*, $q(X)$ is some distribution over the hidden variables, $p(X) = P(X|\boldsymbol{I})$ is the desired posterior distribution,

$$\ell(q) = \sum_{\boldsymbol{x}} q(\boldsymbol{x}) \ln\left(\frac{P(\boldsymbol{I}, \boldsymbol{x})}{q(\boldsymbol{x})}\right), \tag{1.61}$$

$$\mathrm{KL}(q\|p) = -\sum_{\boldsymbol{x}} q(\boldsymbol{x}) \ln\left(\frac{P(\boldsymbol{x}|\boldsymbol{I})}{q(\boldsymbol{x})}\right) \tag{1.62}$$

where the summations run over all possible values of variables X.

Since the KL (Kullback–Leibler) divergence is always non-negative, the functional $\ell(q)$ is a lower bound for the evidence $\ln P(\boldsymbol{I})$. One of the main properties of this decomposition is that the global maximum of lower bound ℓ coincides with the global minimum of $\mathrm{KL}(q\|p)$ and optimal $q^* = \arg\max_q \ell(q)$ is equal to the true posterior $P(X|\boldsymbol{I})$ [19].

Generally, (1.61) cannot be optimized exactly. To make optimization tractable, in variational inference, one often assumes that q belongs to a family of suitable distributions. The *mean field theory* [224] (approximation) assumes that q is a factorized distribution:

$$q(\boldsymbol{x}) = \prod_{C \in \mathcal{C}} q_C(\boldsymbol{x}_C) \tag{1.63}$$

where $\mathcal{C}$ is a subset of the power set of the hidden variables X. In particular, if each C corresponds to only one hidden variable $x_p \in X$, then

$$q(\boldsymbol{x}) = \prod_{p \in V} q_p(x_p), \tag{1.64}$$

Suppose further, that the energy function E is the pair-wise Potts model, see (1.37). Then, ℓ can be simplified as follows:

$$\ell(q) = \sum_{(p,q)\in N} \sum_{l\neq k\in\mathcal{L}} w_{pq} q_p(l) q_q(k) + \sum_p \sum_{l\in\mathcal{L}} u_p^l q_p(l) - \sum_p \sum_{l\in\mathcal{L}} q_p(l) \ln q_p(l) = \tag{1.65}$$

$$= \underbrace{\sum_{(p,q)\in N} w_{pq} \left(1 - \sum_{l\in\mathcal{L}} q_p(l) q_q(l)\right) + \sum_p \sum_{l\in\mathcal{L}} u_p^l q_p(l)}_{\text{relaxation of } E} \underbrace{- \sum_p \sum_{l\in\mathcal{L}} q_p(l) \ln q_p(l)}_{\text{entropy of } q} . \tag{1.66}$$

That is, the lower bound ℓ is the sum of two components. The first one is the relaxation of energy E where each variable x_p representing categorical random variable is replaced by a real-valued point $(q_p(l))_{l\in\mathcal{L}} \in \Delta_L$. The second one is the entropy of distribution q.

The optimization $\ell(q)$ can be done by the coordinate descent [19]. The optimal q_p^* is found from the equation

$$\ln q_p^*(l) \quad = \quad - \sum_{q:(p,q)\in N} \sum_{l'\in\mathcal{L}} w_{pq}[l \neq l'] q_q(l') - u_p^l + const. \tag{1.67}$$

The constant in expression (1.67) does not depend on x_p and thus can be determined from the normalization equation $\sum_l q_p^*(l) = 1$.

Once the optimal approximation q^* is found a simple rounding technique is typically employed to obtain a discrete solution:

$$x_p^* \quad = \quad \arg\max_{l\in\mathcal{L}} q_p^*(l).$$

Despite the attractive properties of such approximation, the found solution often not only fails to correspond to a minimum of E but often has relatively high energy. Nevertheless, in the case of strong unary potentials, the mean-filed approximation offers a simple optimization alternative. Krahenbuhl and Koltun [153] used permutohedral lattice [1] to efficiently approximate the mean-field inference for the case of potentials w_{pq} expressed as a linear combination of Gaussian kernels. That is

$$w_{pq} \quad = \quad \sum_k \alpha^{(k)} \exp\left(-\frac{1}{2}(\boldsymbol{f}_p - \boldsymbol{f}_q)^\top \boldsymbol{\Lambda}^{(k)} (\boldsymbol{f}_p - \boldsymbol{f}_q)\right) \tag{1.68}$$

where α_k is the linear combination coefficients, $\boldsymbol{f}_p \in \mathbb{R}^n$ is the feature vector of pixel p, which can incorporate the pixel's color intensities and position, and $\boldsymbol{\Lambda}^{(k)}$ is a symmetric, positive-definite precision matrix defining the shape of the corresponding kernel.

1.3 Clustering Criteria

As we note in the preamble of Section 1.2, image segmentation can be seen as pixel clustering, often regularized. The standard clustering criteria are often used as components of image segmentation objectives. This section briefly reviews the standard clustering objectives and their known biases.

1.3.1 Parametric Models

Consider the classic k-means clustering objective:

$$E(\boldsymbol{x}, \boldsymbol{\mu}_l) \;=\; \sum_{l\in\mathcal{L}} \sum_{p\in V} x_p^l \|\boldsymbol{f}_p - \boldsymbol{\mu}_l\|^2 \tag{1.69}$$

where V is the set of data points, features $\boldsymbol{f}_p \in \mathbb{R}^n$, $x_p^l \in \{0,1\}$ is the cluster assignment, the set of labels $\mathcal{L} = \{1, 2, \ldots, K\}$, and $\boldsymbol{\mu} = \{\boldsymbol{\mu}_p\}$ is the set of cluster centers (means). In the context of segmentation, V is the set of image pixels, features $\boldsymbol{f}_p$ may correspond to pixel color intensities and/or coordinates, x_p^l corresponds to segmentation indicator variables.

The clustering is obtained by joint minimization of (1.69) with respect to both $\boldsymbol{x}$ and $\boldsymbol{\mu}$, that is $E(\boldsymbol{x}, \boldsymbol{\mu}) \to \min_{\boldsymbol{x},\boldsymbol{\mu}_l}$. As a function of the cluster assignment (segmentation), the k-means objective is a high-order energy:

$$E(\boldsymbol{x}) \;=\; \min_{\boldsymbol{\mu}} E(\boldsymbol{x}, \boldsymbol{\mu}) \;=\; E(\boldsymbol{x}, \mu_l^*(\boldsymbol{x})) \quad \text{and} \quad \mu_l^*(\boldsymbol{x}) \;=\; \frac{\sum_p x_p^l \boldsymbol{f}_p}{\sum_p x_p^l}. \tag{1.70}$$

We can equivalently rewrite the energy (1.69) (up to an additive constant) in terms of *negative log-likelihoods* of the Gaussian distribution:

$$E(\boldsymbol{x}, \boldsymbol{\mu}, \boldsymbol{\Lambda}) \;=\; \sum_{l\in\mathcal{L}} \sum_{p\in V} -x_p^l \ln \mathcal{N}\big(\boldsymbol{f}_p \,|\, \boldsymbol{\mu}, \boldsymbol{\Lambda}\big) \tag{1.71}$$

where $\mathcal{N}\big(\boldsymbol{f}_p \,|\, \boldsymbol{\mu}, \boldsymbol{\Lambda}\big)$ is the Gaussian density with mean μ and positive definite covariance matrix $\boldsymbol{\Lambda}^{-1}$. The Gaussian distribution above is often inadequate to represent the complexity of color or feature variation in many applications including segmentation of natural images. The distribution of $\boldsymbol{f}_p$ can be modeled by a more complex probability estimate $P\big(\boldsymbol{f}_p \,|\, \boldsymbol{\theta}_l\big)$ with parameter vector $\boldsymbol{\theta}_l$, $l \in \mathcal{L}$ such as Gaussian mixture models (GMM), histograms, or neural networks:

$$E(\boldsymbol{x}, \boldsymbol{\theta}) \;=\; \sum_{l\in\mathcal{L}} \sum_{p\in V} -x_p^l \ln P\big(\boldsymbol{f}_p \,|\, \boldsymbol{\theta}_l\big). \tag{1.72}$$

Such general formulation is called *probabilistic k-means*. Note, that our earlier example (1.6) corresponds to the probabilistic k-means with fixed and give parameters θ.

The probabilistic k-means (1.72) has been studied by Kearns, Mansour and Ng [139]. The authors first assume that data in each cluster S_l are modeled by a probability density $P_l(\boldsymbol{f}) = P(\boldsymbol{f} \,|\, \boldsymbol{\theta}_l)$ over $\boldsymbol{f} \in \mathbb{R}^n$. Note, that P_l should be thought of as models rather than the true probability density of data. Then, the clustering is defined by a non-deterministic assigning function $F : \mathbb{R}^n \to \mathcal{L}$. Note, that F is defined[2] by $\{P_l\}$. Then, they assume the data points $\{\boldsymbol{f}_p\}$ are drawn from a fixed probability density $\mathcal{Q}(\boldsymbol{f})$ over $\mathbb{R}^n$. Then, the expectation of the probabilistic k means loss is

$$\mathbb{E}\big[-\ln P_{F(\boldsymbol{f})}(\boldsymbol{f}) \big] \tag{1.73}$$

where expectation is taken over the true distribution of data $\mathcal{Q}$. Now, for any $l \in \mathcal{L}$ and fixed F, we define the volume ratios $w_l = \Pr[F(\boldsymbol{f}) = l]$. The cluster true density is $\mathcal{Q}_l(\boldsymbol{f}) = \mathcal{Q}(\boldsymbol{f}) \Pr[F(\boldsymbol{f}) = l \,|\, \boldsymbol{f}]/w_l$ where the probability is taken over randomness in F, so

$$\mathbb{E}\big[-\ln P_{F(\boldsymbol{f})}(\boldsymbol{f}) \big] \;=\; \sum_{l \in \mathcal{L}} w_p \, \mathrm{KL}(\mathcal{Q}_l \| P_l) + H(\mathcal{Q}|F) \tag{1.74}$$

where KL is the Kullback–Leibler divergence and H is the (cross-)entropy. The first term measures how close the models P_l are to the true data densities $\mathcal{Q}_l$. The second term, cross-entropy $H(\mathcal{Q}|F)$, measures the *informativeness* of partition F defined by $\{P_l\}$:

$$H(\mathcal{Q}|F) \;=\; H(\mathcal{Q}) - \big(H(\{w_l\}) - H(F(\boldsymbol{f})|\boldsymbol{f})\big) \qquad \text{where} \tag{1.75}$$

$$H(\{w_l\}) \;=\; -\sum_{l \in \mathcal{L}} w_l \ln(w_l). \tag{1.76}$$

That is, the cross-entropy $H(\mathcal{Q}|F)$ measures how much uncertainty (entropy) is reduced by partition F. The first term $H(\mathcal{Q})$ does not depend on models $\{P_p\}$, the last term $H(F(\boldsymbol{f})|\boldsymbol{f})$ is the uncertainty of any randomness in F.

Consider the negative entropy term $-H(\{w_l\})$ in (1.75). The case of $|\mathcal{L}| = 2$ is shown in the figure on the right. Note that entropy reaches its maximum (the minimum of the negative entropy) at uniform distribution of volume ratios $\{w_l\}$, that is $w_l^* = 1/|\mathcal{L}|$. Hence, the probabilistic k means clustering objectives[3], *e.g.* used in [204, 330, 240, 72],

[2] In clustering there is typically no randomness in assigning the cluster label based on features $\boldsymbol{f}$. However, in addition to clustering criteria, there is often a regularization term in vision. This makes F depend also on regularization. This dependence may be factored out by introducing "randomness" in F.

[3] Including the standard k means (3.2)

have a *volumetric bias* to clusters of equal size. Due to prevalence of such probabilistic appearance models in the computer vision literature and applications, this property of probabilistic clustering may have significant consequences that one may need to be aware or adjust [72, 275, 30].

1.3.2 Non-parametric (kernel) Models

Probability-based methods described in the previous sections define the adherence measure of features $\boldsymbol{f}_p$ of a data point $p \in V$ as a likelihood $P(\boldsymbol{f}_p \,|\, \boldsymbol{\theta}_l)$ where $P(\cdot \,|\, \boldsymbol{\theta}_l)$ is the probability model of features generated by label $l \in \mathcal{L}$. Instead of measuring the fitness of pixel features to probabilistic parameterized models, which are usually unknown and require estimation, one may measure the "likelihood" of two pixels belonging to the same segment. Such a measure between any two pixels $\{p, q\} \subset V$ is called an *association* $a_{pq} \in \mathbb{R}$.

A simple graph-based clustering criterion, called *cut*, requires minimization of the total inter-clusters association (or equivalently maximization the total intra-cluster association):

$$\sum_{\{p,q\}\subset V} a_{pq}[x_p \neq x_q] \;=\; \sum_{l\in\mathcal{L}} (1 - \boldsymbol{x}^l)^\top \boldsymbol{A}\boldsymbol{x}^l \;=\; \sum_{l\in\mathcal{L}} -(\boldsymbol{x}^l)^\top \boldsymbol{A}\boldsymbol{x}^l + const \tag{1.77}$$

where vector $\boldsymbol{x}^l = (x^l_p)_{p\in V}$ and $x^l_p = [x_p = l]$, matrix $\boldsymbol{A} = [a_{pq}]_{p,q\in V}$. The matrix $\boldsymbol{A}$ is called the *association matrix*. Shi and Malik [253] noted in the context of two clusters that the cut criterion tends to produce one of the clusters consisting only of the most "disconnected" data point. "Disconnected" means low association of the point with the rest of the dataset[4]. Therefore, they argue that (1.77) is a poor clustering criterion. To address this bias of the cut, Shi and Malik proposed volume normalization.

[4]Note, that the bias of the cut objective observed by Shi and Malik [253] is a case of *shrinking* bias of the Potts model, see Figure 1.6, Section 1.2.3 and [26].

In total, they considered a few clustering criteria:

$$\sum_{l\in\mathcal{L}}(1-\boldsymbol{x}^l)^\top \boldsymbol{A}\boldsymbol{x}^l \qquad \text{cut (Potts equivalent)}, \tag{1.78}$$

$$\sum_{l\in\mathcal{L}}-\frac{(\boldsymbol{x}^l)^\top \boldsymbol{A}\boldsymbol{x}^l}{\boldsymbol{x}^l\cdot\mathbf{1}} \qquad \text{average association (kernel k-means [76])}, \tag{1.79}$$

$$\sum_{l\in\mathcal{L}}\frac{(1-\boldsymbol{x}^l)^\top \boldsymbol{A}\boldsymbol{x}^l}{\boldsymbol{x}^l\cdot\mathbf{1}} \qquad \text{ratio cut [114], equivalent to (1.79)}, \tag{1.80}$$

$$\sum_{l\in\mathcal{L}}-\frac{(\boldsymbol{x}^l)^\top \boldsymbol{A}\boldsymbol{x}^l}{(\boldsymbol{x}^l)^\top \boldsymbol{A}\mathbf{1}} \qquad \text{normalized cut} \tag{1.81}$$

where $\boldsymbol{x}^l\cdot\mathbf{1}=|S_l|$ is the size of segment S_l, and $\mathbf{1}$ is the vector of all ones. Shi and Malik [253] used spectral methods to solve the relaxed version of the normalized cut (1.81) and average association (1.79).

In addition, one may consider the following clustering objective:

$$\sum_{l\in\mathcal{L}}\frac{(1-\boldsymbol{x}^l)^\top \boldsymbol{A}\boldsymbol{x}^l}{\min(\boldsymbol{x}^l\cdot\mathbf{1},|V|-\boldsymbol{x}^l\cdot\mathbf{1})} \qquad \text{Cheeger cut [37]}. \tag{1.82}$$

Although an extreme case, one can express both the average association (1.79) and normalized cut (1.81) criteria as energy function of an MRF with one clique of the size of the whole graph, thus, satisfying Theorem 1.1.

Both the normalized cut and other MRF regularization have been popular in computer vision as well as in other areas of research. Recently, Tang et al. [281] combined the graph-based clustering objectives with the classic MRF regularization, benefiting from the normalization properties and long connections of the graph clustering together with local properties of MRFs.

Graph-based clustering criteria have different biases. As noted above the cut tends to isolate points. Shi and Malik noted that the average association criterion tries to find "tight" clusters while the normalized cut produces balanced partitioning. The "tightness" of the average association results has been noted empirically but never understood theoretically. In this thesis we analyze theoretically these biases and propose a method of kernel or association matrix construction that addresses the biases, see Chapter 3.

1.4 Unified View on CRF/MRF and Clustering Objectives

Section 1.2 and Section 1.3 provide an overview of MRF/CRF and clustering-based low-level models for image segmentation. The classification of these models into MRF/CRF and clustering is due to the motivation and origin of the corresponding models. From the point of view of the segmentation problem, this distinction is superficial as they all aim to solve the same problem. In the rest of the thesis, we do not distinguish those groups of methods but treat them as different instances of segmentation regularization.

1.4.1 Regularized Parametric Models

In addition to MRF priors, introduced in Section 1.2, the optimization objectives in computer vision include data fidelity or appearance models. Mumford and Shah [204] proposed the following functional to attain image segmentation:

$$E(\{S_l\}, f \,|\, g) \;=\; \sum_{l\in\mathcal{L}} \left(\sigma^{-2} \iint_{S_l} (f-g)^2 \,\mathrm{d}A \;+\; \iint_{S_l} \|\nabla f\|^2 \,\mathrm{d}x\,\mathrm{d}y \;+\; \nu|\partial S_l| \right) \tag{1.83}$$

where $\sigma, \nu > 0$ are constants, $\mathrm{d}A$ is the infinitesimal area element, $|\partial S_l|$ is the length of the boundary of continuous segment $S_l \subset \mathbb{R}^2$ for $l \in \mathcal{L} = \{0, 1, \ldots, L-1\}$, $g(x, y)$ is a continuous representation of the intensity of a 2D image, and $f(x, y) \in \mathbb{R}$ is an unknown data model of the image. Note that f is allowed to be discontinues on the segment boundaries ∂S_l. The first term ensures that f is a good approximation of g. The second term requires that f does not vary much on segment S_l, $l \in \mathcal{L}$. Note, that this implies that g is not varying much within each S_l either. The third term is a continuous version of Potts regularization [26] ensuring as short segment boundary as possible.

Importantly, the color models f in (1.83) are treated as argument of the energy and optimized simultaneously with segmentation $\{S_l \,|\, l \in \mathcal{L}\}$.

Suppose f is a constant within each of the segments, *i.e.* $f(\boldsymbol{p}) = \mu_l$ for $\boldsymbol{p} \in S_l$, then[5]

$$E(\{S_l\}, \{\mu_l\} \,|\, g) \;=\; \sum_{l\in\mathcal{L}} \left(\sigma^{-2} \iint_{S_l} (\mu_l - g)^2 \,\mathrm{d}A \;+\; \nu|\partial S_l| \right) \tag{1.84}$$

[5]This models has also been studied in [46].

where we can find a closed-form solution for optimal $\mu_l^*(S_l) = \iint_{S_l} g \,\mathrm{d}x\,\mathrm{d}y \Big/ \text{area}(S_l)$. This case has been studied by Chan and Vese in [46].

Consider a discrete version of (1.84):

$$E(\{S_l\}, \{\mu_l\} \,|\, g) \;=\; \sigma^{-2} \underbrace{\sum_{l\in\mathcal{L}} \sum_{p\in V} x_p^k \|g_p - \mu_l\|^2}_{\text{k-means}} \;+\; \underbrace{\nu \sum_{p\in V} \sum_{q\in G_p} [x_p \neq x_q]}_{\text{Potts model (1.37)}} \tag{1.85}$$

Note, the data term in (1.85) is exactly the 1D k-means objective (1.69), and the second term is the Potts model (1.37). Hence, the approach of Mumford and Shah [204] can be seen as regularized clustering where segments S_l are clusters and segmentation variables x_p^l are cluster assignments.

In a more general case studied by Zhu and Yuille [330], each pixel $p \in V$ of an image can be represented by n-dimensional features $\boldsymbol{f}_p \in \mathbb{R}^n$ such as combinations of RGB values, texture features, pixel coordinates, *etc.* The Gaussian distribution is not adequate to represent the complexity of color or feature variation in many applications including segmentation of natural images. The distribution of $\boldsymbol{f}_p$ can be modeled by a more complex probability estimate $P\big(\boldsymbol{f}_p \,|\, \boldsymbol{\theta}_l\big)$ with parameter vector $\boldsymbol{\theta}_l$, $l \in \mathcal{L}$ such as Gaussian mixture models (GMM), histograms, or neural networks:

$$E(\{x_p^l\}, \{\boldsymbol{\theta}_l\} \,|\, \{\boldsymbol{f}_p\}) \;=\; \underbrace{\sum_{l\in\mathcal{L}} \sum_{p\in V} -\ln P\big(\boldsymbol{f}_p \,|\, \boldsymbol{\theta}_l\big) x_p^l}_{\text{probabilistic } k \text{ means}} \;+\; \nu \sum_{p\in V} \sum_{q\in G_p} [x_p \neq x_q]. \tag{1.86}$$

GMMs as an appearance (color) data model were used in GrabCut [240] for interactive image foreground/background segmentation where RGB color $\boldsymbol{c}_p$ was used as features of pixel $p \in V$, that is $\boldsymbol{f}_p = \boldsymbol{c}_p$. In addition, they introduced a new interaction interface, *i.e.* bounding boxes, to set the initial values of segments $\{S_l\}$. Their optimization iterates GMM fitting and graph cut [31] to re-estimate the segments and GMM models.

Consider the case of histograms as in [31, 295]. For discrete[6] $\boldsymbol{f}_p$, the data term of (1.86)

$$\sum_{l\in\mathcal{L}} \sum_{p\in V} -\ln P\big(\boldsymbol{f}_p \,|\, \boldsymbol{\theta}_l\big) x_p^l \;=\; \sum_{l\in\mathcal{L}} \sum_{p\in V} -\ln \frac{n(\boldsymbol{f}_p \,|\, S_l)}{|S_l|} x_p^l \tag{1.87}$$

[6] This assumption is not critical as any features histogram defines the bin index as a new discrete feature.

where $n(\boldsymbol{f} \,|\, S_l) = \sum_{p \in S_l} [\boldsymbol{f}_p = \boldsymbol{f}]$ is the number of pixels with features $\boldsymbol{f}$ in segment S_l, and $|S_l| = \sum_{\boldsymbol{f}} n(\boldsymbol{f} \,|\, S_l)$ is the number of pixels in segment S_l. Then,

$$\sum_{l \in \mathcal{L}} \sum_{p \in V} - \ln P(\boldsymbol{f}_p \,|\, \boldsymbol{\theta}_l) x_p^l \;=\; \underbrace{\sum_{l \in \mathcal{L}} \sum_{\boldsymbol{f}_p} -n(\boldsymbol{f}_p \,|\, S_l) \, \ln n(\boldsymbol{f}_p \,|\, S_l)}_{\text{concave cardinality potential}} \;+\; \underbrace{\sum_{l \in \mathcal{L}} |S_l| \, \ln |S_l|}_{\text{volumetric term}} \tag{1.88}$$

The volumetric term above is proportional to the negative entropy of partitioning. Therefore, it corresponds to the volumetric bias, as its optimal (lowest) value corresponds to segments of equal size $|S_l| = |V| / |\mathcal{L}|$. Tang *et al.* [275] argued that the appropriateness of the volumetric bias is highly application specific. One may need to use the bias to specific volumes or remove the bias at all. The cardinality potential is a concave function of segments cardinality and hence could be minimized exactly by a graph cut [275, 147].

1.4.2 Regularized Non-parametric (kernel) Models

In our previous work of Tang *et al.* [277, 278, 281], we showed that a combination of MRF/CRF and clustering objectives give better segmentation. Specifically, [277] combines normalized cut (1.81) and MRF/CRF regularization:

$$E(\boldsymbol{s}) \;=\; \underbrace{-\sum_{l \in \mathcal{L}} e(\boldsymbol{s}^l)}_{\text{normalized cut}} \;+\; \gamma \underbrace{\sum_{C \in \mathcal{C}} E_C(\boldsymbol{s}_C)}_{\text{MRF/CRF}} \quad \text{and} \quad e(\boldsymbol{x}) \;=\; \frac{\boldsymbol{x}^\top \boldsymbol{A} \boldsymbol{x}}{\boldsymbol{x}^\top \boldsymbol{A} \vec{\boldsymbol{1}}} \tag{1.89}$$

where E_C is a general CRF/MRF potential (1.22). We introduced auxiliary functions by linearizing the normalized cut term:

$$a_{(t)}(\boldsymbol{s}) \;=\; \underbrace{\sum_{l \in \mathcal{L}} \nabla e(\boldsymbol{s}^l_{(t)})^\top (\boldsymbol{s}^l - \boldsymbol{s}^l_{(t)})}_{\text{unary term}} \;+\; \gamma \sum_{C \in \mathcal{C}} E_C(\boldsymbol{s}_C) \tag{1.90}$$

where t is the iteration index and the normalized cut gradient[7]

$$\nabla e(\boldsymbol{x}) \;=\; \boldsymbol{A} \vec{\boldsymbol{1}} \frac{\boldsymbol{x}^\top \boldsymbol{A} \boldsymbol{x}}{(\boldsymbol{x}^\top \boldsymbol{A} \vec{\boldsymbol{1}})^2} \;-\; \frac{2 \boldsymbol{A} \boldsymbol{x}}{\boldsymbol{x}^\top \boldsymbol{A} \vec{\boldsymbol{1}}}. \tag{1.91}$$

[7] Note, we reinterpreted $e(\boldsymbol{s})$ as a function of continuous $\boldsymbol{s}$.

Iterations $\boldsymbol{s}_{(t+1)} \leftarrow \min_{\boldsymbol{s}} a_{(t)}(\boldsymbol{s})$ guarantee non-decreasing series of energy values $E(\boldsymbol{s}_{(t)})$ in the case of positive-definite[8] $\boldsymbol{A}$.

Note, the auxiliary functions $a_t(\boldsymbol{s})$ only add unary terms to the MRF/CRF energy. Therefore, the complexity of the inference in the MRF/CRF model does not generally increase as most low-level solvers are capable of incorporating additional unary terms. So, one may use standard low-level MRF/CRF solvers to optimize joint objective (1.89) combining the normalized cut and MRF/CRF regularization. The follow-up work [281] extended (1.89) to other spectral and kernel clustering objectives reviewed in Section 1.3.2.

See a summary of reviewed segmentation objectives in Table 1.1.

[8] If $\boldsymbol{A}$ is not positive-definite, the diagonal shift $\boldsymbol{A} \leftarrow \boldsymbol{A} + \nu \operatorname{diag}(\boldsymbol{A}\vec{\boldsymbol{1}})$ for some large ν guarantees positive-definiteness without affecting the optima of $\min_{\boldsymbol{s}} E(\boldsymbol{s})$ [281, 239].

Class	Model	Objective	Assumptions/Bias
CRF/MRF	sparse Potts (1.37) [97, 31]	$\sum_{(p,q)\in N} w_{pq}[s_p \neq s_q]$	shrinking bias [26]
	dense Potts (1.40) [153]	$\sum_{(p,q)\in V\times V} w_{pq}[s_p \neq s_q]$	equal volume bias [293]
	curvature (1.46) [206], Chapter 2	$\sum_{(w,t,v)\in N_l} [s_w = s_v][s_w \neq s_t]$	bias to straight-line boundaries [206, 211]
	convexity (1.48) [105]	$\sum_{\text{line } \ell} \sum_{\substack{\{w,t,v\}\subset\ell \\ w<t<v}} s_w(1-s_t)s_v$	bias to convex boundaries [105]
Clustering	k-means (1.69)	$\min_{\boldsymbol{\mu}} \sum_{l\in L}\sum_{p\in V} s_p^l \|\boldsymbol{f}_p - \boldsymbol{\mu}_l\|^2$	equal volume bias [140, 30]
	average association (kernel k-means) (1.79) [253]	$\sum_{l\in L} -\frac{(\boldsymbol{s}^l)^\top \boldsymbol{A}\boldsymbol{s}}{\boldsymbol{s}^l \cdot \vec{\mathbf{1}}}$	density mode bias Chapter 3
	normalized cut (1.81) [253]	$E_{NC}(\boldsymbol{s}) = \sum_{l\in L} -\frac{(\boldsymbol{s}^l)^\top \boldsymbol{A}\boldsymbol{s}}{(\boldsymbol{s}^l)^\top \boldsymbol{A}\vec{\mathbf{1}}}$	sparse subset bias Chapter 3
mixed	kernel-cut (1.89) [281]	$E_{NC}(\boldsymbol{s}) + \gamma \sum_{C\in\mathcal{C}} E_C(\boldsymbol{s}_C)$	inherited from normalized cut and MRF E_C (1.22)

Table 1.1: Summary of some low-level objectives for segmentation.

1.5 From Low-level to Semantic Segmentation

In the previous sections, we have only considered low-level segmentation problems, such as interactive segmentation. These problems can be formulated as different forms of (regularized) clustering which is based on low-level pixel features, such as intensity, colors, xy coordinates, and others, see Section 1.4. *Semantic segmentation* is the problem of partitioning the image into *semantic* segments such that each segment corresponds to a semantic class. The set of classes is typically given in advance together with a dataset of annotated images. For example, VOC dataset [84] defines the following classes: *background*, *aeroplane*, *bicycle*, *bird*, *boat bottle*, *bus*, *car*, *cat*, *etc.* The task of semantic segmentation is solved by employing supervised machine learning techniques.

Machine learning generally refers to the set of general-purpose models and algorithms allowing automatic (or semi-automatic) recognition of patterns in large datasets. Compared to traditional learning/analysis, machine learning aims to efficiently solve a particular task without relying on specific human-designed instructions, but rather extracting patterns from just looking into data. The classical examples of tasks studied in machine learning include classification, tracking, detection, clustering, *etc.*

The simplistic low-level features are insufficient to enable the basic ML methods to solve semantic segmentation (or classification). The literature suggests many handcrafted features and representations that can improve the recognition task, including SIFT [177], HoG [69], textons [166, 180] and ways of features encoding [48].

Another, and more successful, approach is to design a learning system that can "learn" the best features for a particular task. Ultimately, such an approach prevailed in many ML applications using deep learning. Due to the abundance of (annotated) data and computational resources and new optimization heuristics [121, 154], the machine learning community has developed a family of *deep neural networks* (DNN), see overview in [163], that revolutionized almost all traditional artificial intelligence applications. Starting with [154] deep learning systems have surpassed previous state-of-the-art methods in computer vision [154, 176, 100, 115], speech recognition [120], classic games such as chess and go [256], automatic text processing and translation [17, 13], *etc.* In some cases, deep learning systems outperform human annotators [115].

The next sections review deep learning models for classification and semantic segmentation problems.

1.5.1 Classification Neural Networks and Deep Features

One of the earliest artificial neural networks is *perceptron* [237]. Perceptron is a linear binary classifier. Perceptron is a function

$$\phi : \mathbb{R}^n \to \{-1, 1\} \quad \textbf{s.t.} \quad \phi(\boldsymbol{x}) \;=\; \text{sign}(\boldsymbol{w} \cdot \boldsymbol{x}) \tag{1.92}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ is the input vector, $\boldsymbol{w} \in \mathbb{R}^n$ is a row-vector of weights. Perceptron is a linear classifier. *Multilayer perceptron* (MLP) is a generalization:

$$\phi : \mathbb{R}^n \to \mathbb{R}^L \quad \textbf{s.t.} \quad \phi(\boldsymbol{x}) \;=\; \phi_K(\sigma(\cdots \phi_2(\sigma(\phi_1(\boldsymbol{x}))) \cdots)) \tag{1.93}$$

where K is the number of layers, L is the predefined number of classes, $\forall k \in \{1, \ldots, K\}$ $\phi_k(\boldsymbol{x}) = \boldsymbol{W}_k \boldsymbol{x}$, weight matrix $\boldsymbol{W}_k : \mathbb{R}^{n_l} \to \mathbb{R}^{n_{l+1}}$ such that $n_1 = n$ and $n_{K+1} = L$, and σ is an element-wise non-linear *activation* function. Functions ϕ_k are called layers and the intermediate values $\boldsymbol{f}^k = \phi_k(\sigma(\cdots \phi_2(\sigma(\phi_1(\boldsymbol{x}))) \cdots))$ are called *(deep) features* at layer k.

The exploding literature on deep learning has proposed a multitude of various designs of layer functions ϕ_k, to the point that it becomes difficult to adequately define the class of deep neural network functions. It may seem that the defining property is that DNN is a composition of differentiable functions[9]. However, it is obvious that differentiability is not required due to the existence of a wide class of binary or quantized networks, *e.g.* [64].

In the following, we use a K-layers *feed-forward neural networks* of the form:

$$\phi : \mathbb{R}^n \to \mathbb{R}^L \quad \textbf{s.t.} \quad \phi(\boldsymbol{x}) \;=\; \phi_K(\cdots \phi_2(\phi_1(\boldsymbol{x}))) \cdots)) \tag{1.94}$$

where ϕ_k is some (differentiable) function $\mathbb{R}^{n_k} \to \mathbb{R}^{n_{k+1}}$. Note, we removed explicit activation functions σ, which could be represented by functions ϕ_k. Each layer function ϕ_k is assumed to be parameterized by a vector $\boldsymbol{\theta}_l$. To explicitly denote the dependence on the parameters we write $\phi_k(\boldsymbol{f}|\boldsymbol{\theta}_l)$ and $\phi(\boldsymbol{x}|\boldsymbol{\theta})$ where $\boldsymbol{\theta}$ is the concatenation of all parameter vectors $\boldsymbol{\theta}_l$.

A DNN can include the following layers:

- Linear (fully-connected) layer $\phi_k(\boldsymbol{f}|\boldsymbol{W}) = \boldsymbol{W}\boldsymbol{f}$ where $\boldsymbol{W}$ is a weight matrix.
- Element-wise activation layer $\phi_k(\boldsymbol{f}) = \sigma(\boldsymbol{f})$ where $o_i = \sigma(f_i)$, $o_i \in \mathbb{R}$ and $f_i \in \mathbb{R}$ are elements of vectors $\boldsymbol{o} = \sigma(\boldsymbol{f})$ and $\boldsymbol{f}$ correspondingly for $i \in \{1, \ldots, \dim(\boldsymbol{f})\}$.

[9] Leading to even new terms such as *Differentiable Programming*. See *"Deep Learning est mort. Vive Differentiable Programming!"* https://www.facebook.com/yann.lecun/posts/10155003011462143. Facebook post by Yann LeCun, Jan 2018.

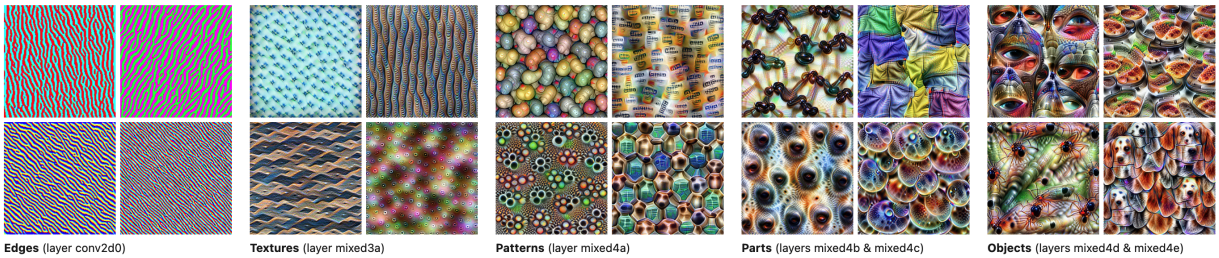


Figure 1.12: Deep features of GoogLeNet [271]. As the layer depth increases (left to right) the corresponding features are learned to recognize more abstract and complex patterns in the image, from edges and textures to objects and their parts. Figure from [209].

- Convolutions $\phi_k(\boldsymbol{f}|\boldsymbol{W}) = \boldsymbol{f} \circ \boldsymbol{W}$ where $\boldsymbol{W}$ is the convolution kernel. Note, that in the case of grid data (such as images) vectors $\boldsymbol{f}$ inherit a multi-dimensional grid such that the operator $\circ$ can be defined as multi-dimensional convolution on the corresponding multi-dimensional arrays. These types of layers could be extended to include stride, padding, dilation [315], deformation fields [68], *etc.* Note, convolution is a special case of a linear layer.

- Pooling layers aggregate adjacent (with respect to the inherited grid) features using an aggregation function, which is typically max or average.

The DNNs that are composed of the above-mentioned layers are called *convolutional neural networks* (CNN). The advantages of CNN *vs.* the fully-connected networks include a lower number of parameters and translational equivariance. The former improves the quality of training by reducing overfitting, while the former is natural in the context of vision where the translation of images should not result in the change of the classification result.

Above we have not prescribed explicitly a grid to the intermediate features $\boldsymbol{f}^k$. The output of each layer function ϕ_k above can be arranged into a grid naturally, so that features $\boldsymbol{f}^k$ at layer k can be partitioned with respect to the grid, that is $\forall p \in V_k$: $\boldsymbol{f}^k_p \in \mathbb{R}^{C_k}$ where V_k is the set of "pixels" (spatial locations), and C_k is the number of channels at layer k.

Feature visualization is a powerful tool, providing insight into the deep features learned automatically from data [83, 209, 259, 179, 270]. The basic idea behind the feature visualization is to find or generate an image that maximizes activation (output) of the specific feature $\boldsymbol{f}^k_p \to \max_{\boldsymbol{I}}$ or entire channel $\sum_p \boldsymbol{f}^k_p \to \max_{\boldsymbol{I}}$. The examples of images that respond best w.r.t. channel activations are in Figure 1.12.

1.5.2 Towards Fully-supervised Semantic Segmentation

Semantic segmentation is an example of so-called "*image-to-image*" prediction or training where the problem is to transform an input image into an output image. In semantic segmentation the output image is a label map, see Figure 1.4 on the right. Other image-to-image tasks include stereo prediction [141, 220, 47] where the input is a stereo-pair of images and the output image is the disparity map. In edge prediction [311], the output is the edge map. In optical flow [314], the output image is the motion map. In monocular depth prediction [101, 328, 314, 319, 102], the output is a depth map. There are also image restoration [329], super-resolution [164, 171, 302, 321] networks, *etc.*

In the following, we review a basic learning loss for the fully supervised image segmentation problem. In a *fully supervised* setting it is assumed that all pixels (or almost all) of each image in a training dataset are assigned a semantic label, see example in Figure 1.4.

Segmentation Architectures

This section reviews methods exploiting the fact that segmentation can be cast as a classification of image patches. In the simplest form, splitting the image into a set of overlapping patches (in a sliding window fashion) is a valid but inefficient approach. The methods discussed below use different techniques to efficiently perform the segmentation.

The advantage of adapting a classification network, *e.g.* AlexNet [154] or VGG [258], is the ability to use models pre-trained on a large classification dataset. For example, the ImageNet classification dataset [74] contains millions of images while a common Pascal-VOC [84] segmentation dataset contains tens of thousands of images. Pre-training on classification datasets significantly boosts the performance and is *de facto* the standard [116].

Adapting classification networks for segmentation A typical convolutional neural network, *e.g.* AlexNet or VGG, consists of three types of layers: convolution with non-linear activation, (max-)pooling, and fully connected layers. Fully connected layers perform a linear operation and some non-linear activation. Thus, they can be treated as a special case of a convolutional layer. Matan *et al.* in [192] noted that both convolutional and pooling layers can be used for arbitrary sized images to output dense predictions. They used it in the context of digit strings recognition. In the context of deep semantic image segmentation, it was used by Long *et al.* [176]. Such an approach is often referred to as fully convolutional networks (FCN). See illustration in Figure 1.13 for details.

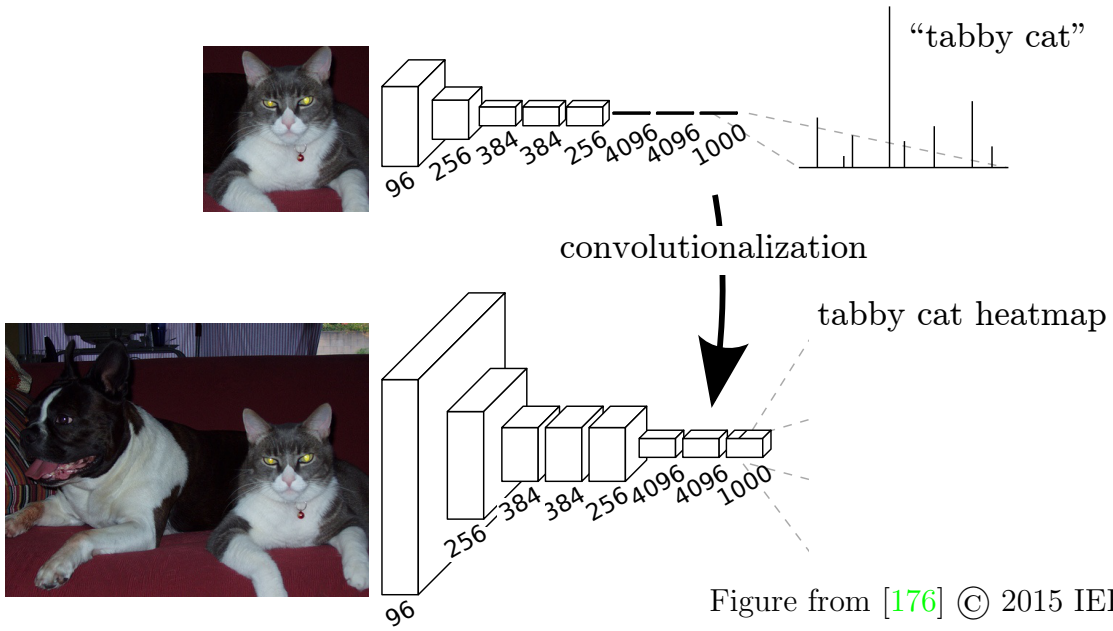


Figure from [176] © 2015 IEEE.

Figure 1.13: Fully convolutional network reinterprets the fully connected layers in a classification network (on the top) as convolutional layers. The resulting network (on the bottom) does not assume any particular size of the images and can be used for coarse semantic segmentation. **Notation:** the rectangular blocks represent intermediate outputs (feature maps) of the net's layers. Each block has two spatial dimensions and one feature dimension. The spatial dimensions are determined by the input and convolution kernel size. The numbers below blocks denote the feature dimensionality. It is implicitly assumed that between each consecutive pair of the blocks there are convolution layer, activation layer, and optional pooling. The relative size of blocks reflects down-sampling along the spatial dimensions (due to convolution strides and/or max-pooling) and an increase in feature dimension (due to an increased number of convolutions within a single convolution layer). **Convolutionalization:** the fully connected layers (last three feature maps) are treated as convolutions. As a result, the application of the network to images of large size yields dense coarse predictions.

To adapt to a different number (and meaning) of semantic labels between specific classification and segmentation datasets, the last layer of the classification network is usually replaced by a randomly initialized layer of a size appropriate for segmentation.

The resulting resolution of prediction is coarse, for example in FCN the resolution drops 32 times in each of spatial dimensions. To improve the resolution additional layers combining up-sampling with early layers' feature maps (skip connections) are typically employed, see [176, 236, 12].

Another common technique explored for segmentation in [53] increasing spatial resolution of FCNs (without retraining of the corresponding classification network) is the removal of the max-pooling operation. To adjust the subsequent convolutional layers to the change of scale, the subsequent convolutions are "dilated", a technique also known as algorithm *á trous* in signal processing.

To achieve the state-of-the-art results, additional improvements are required. In particular, [236, 53, 324, 55] add spatial pyramid pooling, *i.e.* combining features computed at different scales via varying dilation factor, global pooling for capturing the context, additional decoders, hour-glass architectures, and others. For details refer to [55]. Interestingly, early methods [53] post-process (in testing time) the output of the network by a dense Potts CRF [153] to ensure better alignment with image edges. Other approaches [5, 267] use *pixel-adaptive convolutions*, which is related to bilateral filtering [284], to ensure edge alignment. Recent deep methods do not use CRF inference as post-processing [54]. Zheng *et al.* [325] cast the dense Potts CRF [153] mean-field inference iterations as a recurrent neural network that allows backpropagation through the recurrent layers.

Alternative architectures Instead of adapting a classification model, U-Net architecture [236] was directly designed for image segmentation. U-Net is designed to segment high-resolution medical images by splitting the image into a set of tiles. Each tile is independently processed by U-Net. Due to overlapping tiles, the prediction is smooth across the whole image. Similarly, to the approaches adopting classification networks, U-Net begins with the contracting part (encoder), which reduces the spatial dimension while increasing the feature dimensionality. This helps "to capture context". The distinctive second part of the network (decoder) upsamples the features in the same symmetric manner as the first part, see Figure 1.14. This part "enables precise localization".

Non-convolutional networks have also demonstrated recently state-of-the-art results, *e.g.* based on self-attention or hybrid architectures [291, 326, 96].

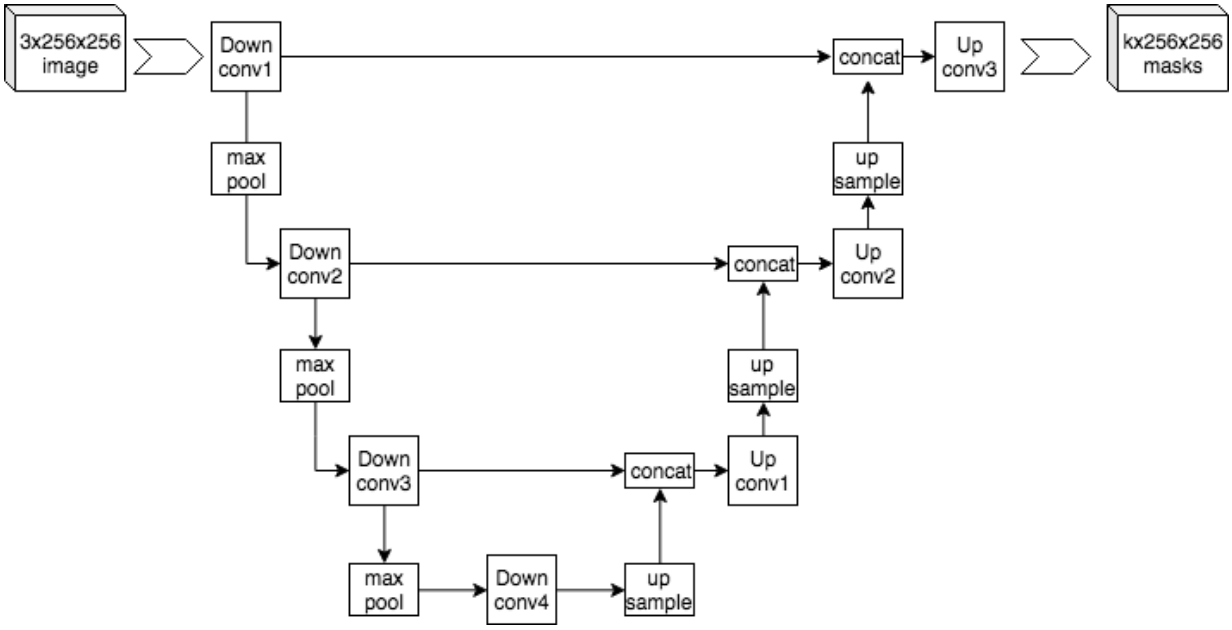


Figure 1.14: U-Net architecture [236]. Each blue box corresponds to a feature map. Figure due to Mehrdad Yazdani, CC BY-SA 4.0, via Wikimedia Commons.

Losses for Fully Supervised Segmentation

We denote the number of different pixel labels L, and the output of the network $\phi(\boldsymbol{I}, \boldsymbol{\theta})$ where $\boldsymbol{\theta}$ is a vector of network parameters, $\boldsymbol{I} \in \mathbb{R}^{3\times n\times m}$ is an input RGB image of size $n \times m$, the networks output $\phi(\boldsymbol{I} \,|\, \boldsymbol{\theta}) \in \Delta_L^{n\times m} \subset \mathbb{R}^{L\times n\times m}$ such that the network predicts the multinomial distribution of L class labels $\Delta_L = \{(x^1, \ldots, x^L) \in \mathbb{R}^L \mid \sum_l x^l = 1 \;\&\; x^l \geq 0\}$ at each pixel of the input. Note, the output of the network is of same dimensionality $(n \times m)$ as the input. The subscript index $\boldsymbol{I}_p \in \mathbb{R}^3$ denotes the color of pixel p. Let V be the set of all pixels on an image. We will assume an image $\boldsymbol{I}$ and the corresponding correct labeling $\boldsymbol{y} = (\boldsymbol{y}_p)_{p\in V}$ are stochastically generated.

The goal of the training is to minimize the following loss function:

$$\mathbb{E}_{(\boldsymbol{I},\boldsymbol{y})}\, H(\boldsymbol{y} \,|\, \phi(\boldsymbol{I} \,|\, \boldsymbol{\theta})) \quad = \quad \mathbb{E}_{(\boldsymbol{I},\boldsymbol{y})} \sum_{p\in V} H\left(\boldsymbol{y}_p \,\middle|\, [\phi(\boldsymbol{I} \,|\, \boldsymbol{\theta})]_p\right), \tag{1.95}$$

which is the expected cross entropy between one-hot distribution of true label (denoted by vector $\boldsymbol{y}_p$) and the distribution predicted by network $[\phi(\boldsymbol{I} \,|\, \boldsymbol{\theta})]_p \in \Delta_L$ at each pixel $p \in V$.

1.5.3 Network Optimization Basics

Let the expected loss be

$$\ell(\boldsymbol{\theta}) \;\; := \;\; \mathbb{E}_{(\boldsymbol{I},\boldsymbol{y})}\; g(\phi(\boldsymbol{I}\,|\,\boldsymbol{\theta}), \boldsymbol{y}) \tag{1.96}$$

where g is the loss assigned to prediction $\phi(\boldsymbol{I}\,|\,\boldsymbol{\theta})$ for image $\boldsymbol{I}$ with ground truth label $\boldsymbol{y}$. The goal is to find optimal parameters of the model

$$\boldsymbol{\theta}^* \;\; = \;\; \arg\min_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}). \tag{1.97}$$

The standard gradient descent procedure iteratively updates current parameters

$$\boldsymbol{\theta}^{(t+1)} \;\; \leftarrow \;\; \boldsymbol{\theta}^{(t)} - \alpha\, \nabla \ell(\boldsymbol{\theta}^{(t)}) \tag{1.98}$$

where $\nabla\ell$ is the gradient of ℓ, and $\alpha > 0$ is called *learning rate*.

In practice, there is no access to the probability distribution of images and labels. Instead, a large collections of labeled images is assumed to be available. Then, the expectation loss ℓ is approximated by sample mean:

$$\hat{\ell}(\boldsymbol{\theta}) \;\; := \;\; \frac{1}{T} \sum_{j=1}^{T} g(\phi(\boldsymbol{I}\,|\,\boldsymbol{\theta}), \boldsymbol{y}) \;\; \approx \;\; \ell(\boldsymbol{\theta}) \tag{1.99}$$

where T is the size of the sample. The *stochastic gradient descent* (SGD) updates are

$$\boldsymbol{\theta}^{(t+1)} \;\; \leftarrow \;\; \boldsymbol{\theta}^{(t)} - \alpha\, \nabla \hat{\ell}(\boldsymbol{\theta}^{(t)}). \tag{1.100}$$

To speed up the convergence and prevent oscillations the learning rate is adjusted during training in accord with a given in advance learning rate schedule or adaptively [227].

For example, the segmentation loss (1.95) is approximated by:

$$\frac{1}{T} \sum_{j=1}^{T} \sum_{p \in V} H\left(\boldsymbol{y}_p^j \,\middle|\, [\phi(\boldsymbol{I}^j|\boldsymbol{\theta})]_p\right) \;\; = \;\; -\frac{1}{T} \sum_{j=1}^{T} \sum_{p \in V} \boldsymbol{y}_p^j \cdot \log[\phi(\boldsymbol{I}^j|\boldsymbol{\theta})]_p. \tag{1.101}$$

where log is taken over individual components of vector $[\phi(\boldsymbol{\theta}, \boldsymbol{I}^j)]_p$. Sample $B = \{(\boldsymbol{I}^j, \boldsymbol{y}^j) \mid j = 1, \ldots, T\}$ is called *mini-batch* and is drawn from a dataset of labeled examples such as Pascal VOC [84], COCO [173], Cityscapes [61] and others.

The initial parameters $\boldsymbol{\theta}^{(0)}$ are usually initialized with random noise. It is important to note the performance of deep models depends significantly on the initialization [269].

The accuracy of the approximate gradients $\nabla\hat{\ell}$ in (1.100) increases with the sample size T. It quickly becomes impractical choosing large values of T due to increased demands on memory and computation time. One needs to find a balance between the training speed, available resources and training quality [24].

There are many alternative techniques that aim to increase the accuracy (or equivalently reduce variance) of the gradient updates. Training with *momentum* [241] uses a linear combination of gradients from the previous iterations instead of the sample gradient:

$$\boldsymbol{\theta}^{(t+1)} \quad \leftarrow \quad \boldsymbol{\theta}^{(t)} - \alpha \sum_{k=0}^{t} \beta^{(k,t)} \nabla\hat{\ell}(\boldsymbol{\theta}^{(k)}). \tag{1.102}$$

Polyak-Ruppert method [243, 232] proposes averaging of the model parameters $\sum_k \beta^{(k)}\boldsymbol{\theta}^{(k)}$ over stochastic trajectories. Averaging methods based on this idea proved to be beneficial in deep model training for various vision problems [134, 8, 207, 312].

Modifications of the SGD creating per-parameter adaptive learning rates include AdaGrad [78], RMSProp [119], Adam [145], *etc.*

The second-order methods compute parameters update in the form $\Delta\boldsymbol{\theta} = \boldsymbol{H}^{-1}\nabla\ell(\boldsymbol{\theta})$, *cf.* (1.100), where $\boldsymbol{H}$ is the Hessian or its approximation [263, 38, 242]. In neural networks, computing the Hessian is infeasible, so various approximations are used, *e.g.* diagonal or low-rank [19]. The efficient computation of Hessian-vector products is possible [230, 248]; while solving linear systems with Hessian is still challenging [269]. Another group of methods is based on employing Gaussian-Newton matrix and K-FAC approximations [188, 10, 23, 218].

Regularization in DNN training The optimization of neural network losses exploits a number of regularization techniques—such as weight decay, dropout, batch normalization [155, 264, 128]. This type of regularization operates on the level of the intermediate layers and parameters of the network. This is different from MRF/CRF discussed in Section 1.2 where regularization is applied to the shape of the segments predicted by a network.

1.5.4 Topological Losses for Fully-supervised Segmentation

Loss (1.95) treats each pixel independently. In the context of segmenting many types of objects with complex typologies such as vessels or neurons, it has been noted in the literature that such *pixel-wise* losses produce a pure-quality segmentation network. A network

trained only on a pixel-wise loss often produces segmentation mistakes, *i.e.* gaps and spurious structures, see [200, 125, 58, 126, 216]. These works use various techniques detecting topologically important structures and develop corresponding losses for supervised deep neural network training.

For example, Mosinska et al. [200] used deep features of an auxiliary pre-trained on ImageNet network (VGG) to construct a higher-order loss. To compute the loss, they run both the ground-truth and predicted segmentation masks through the auxiliary network collecting intermediate deep features. Their topological loss penalizes the squared ℓ^2 distance between these deep features. They found that, in practice, such a loss effectively penalizes topological mistakes. Since such an approach does not explicitly rely on topologically-motivated methods, in general, it cannot guarantee the training of topologically plausible structures.

Hu et al. [125], Clough et al. [58], Hu et al. [126] identify topological features in the image, *e.g.* connected components, loops, holes, or extrema, saddle-points and ridges of the likelihood maps. Hu et al. [125] use the theory of persistent homology. In particular, they vary the threshold value of a real-valued likelihood map. They process each of the resulting binary maps detecting various topological features. They create a so-called persistence diagram that records the times of appearance and disappearance of these features. Then the optimal matching is computed between the diagrams for the prediction and ground truth. Finally, the loss is defined as the squared ℓ^2 between such diagrams. Instead of computing the persistence diagrams, Clough et al. [58] count the number of different topological features and, then, penalize the difference of these counts. The approach of [126] uses reweighting of the pixels that lie on the critical structures.

Oner et al. [216] construct a *connectivity-oriented* loss by expressing the connectivity of thin structures, such as roads and canals, as disconnections in the background segment. Their approach considers the paths between background regions disconnected in the ground truth. Intuitively, in a good prediction, any such path must visit a pixel with low prediction likelihood (low values correspond to thin structures). First, they consider the paths between pairs of pixels that violate such intuition the most, *i.e.* the paths that maximize the lowest likelihood of visited pixels. Then, the loss further penalizes this lowest likelihood.

1.6 Low-level Regularization for Weakly-supervised Semantic Segmentation

Even though deep networks can solve some computer vision problems very well (as benchmarked on specific datasets), they have notoriously limited generalization capability to new datasets. Slight changes in the lighting, view angle, texture, and occlusion patterns, changing the context result in significant performance degradation[10] [317]. In addition, to perform well these methods require lots of labeled data during training.

One way of addressing these issues is collecting and labeling ever-larger datasets in the hope of capturing more diversely the set of real-world images. This approach has many challenges, one of which is the cost of labeling. While some computer vision problems—*e.g.* image classification—allow cheap annotations, others—in particular image segmentation—require extensive manual work and training of the annotators [173]. Thus the segmentation ground truth is expensive and typical segmentation datasets are much smaller than classification datasets. For example, the ImageNet classification dataset [74] contains over 10^6 labeled images, while Pascal VOC 2012 dataset [84] has less than 10^4 fully labeled segmentations.

One potent approach of reducing the cost of segmentation labeling is called weak supervision where the annotations are required only for a subset of pixels, *e.g.* in Figure 1.5. In extreme cases, the annotations can be given in the form of clicks or even image tags that identify the presence of semantic categories without any location information. While weak supervision provides less information about the scene compared to full supervision, it has the potential to benefit from additional prior information about segment shapes.

Interestingly, many weakly supervised settings in segmentation, namely scribble annotations or bounding boxes, closely resemble low-level interactive segmentation problems, *i.e.* [31, 29, 240, and others]. It is well established in low-level vision that shape priors drastically improve the quality of segmentation.

This section reviews recent approaches to weakly supervised learning for semantic segmentation. That is either the pixel labels are only provided for a few pixels, or bounding boxes for the segments are given, or only image-level labels are given without any information about segment localization. As pointed earlier such an approach allows much faster and cheaper labeling. The downside is that resulting trained systems under-perform fully supervised methods.

[10] This property of changing the patterns in data from one dataset to another is often referred to as *domain shift*. The problem of bridging the performance gap is called *domain adaptation* [36, 16, 95].

Approaches as simple as *partial cross-entropy* (PCE) often work better than complex heuristics [279]. PCE extends the loss (1.95) such that it ignores all unlabeled pixels:

$$L_{pce}(\boldsymbol{\theta}) \quad = \quad \mathbb{E}_{(\boldsymbol{I},\boldsymbol{y})} \; -\frac{1}{|V_l|}\sum_{p\in V_l} \boldsymbol{y}_p \cdot \log[\phi(\boldsymbol{I}\,|\,\boldsymbol{\theta})]_p \tag{1.103}$$

where V_l denotes the set of all labeled pixels on image $\boldsymbol{I}$.

1.6.1 Regularized Losses

To motivate a specific pair-wise loss for semantic segmentation training, we will review a work of Weston et al. [303] on the digit classification. In their setup, the training data comes in two flavors. First, there is a set of digit images where for each image the true digit label is known. Second, there is an unlabeled set of digit images. The aim is to utilize the unlabeled set for better network training. This is an example of semi-supervised learning, see Section 1.2.2. The success of semi-supervised and unsupervised learning relies on several assumptions about data. Weston *et al.* rely on a smoothness assumption stating that the output computed on close examples should be close as well. Their loss[11] combines standard cross-entropy and regularization:

$$\mathbb{E}_{(\boldsymbol{I},y)}\; H(y|\phi(\boldsymbol{I}\,|\,\boldsymbol{\theta})) \quad + \quad \mathbb{E}_{(\boldsymbol{I},\boldsymbol{I}')}\; w(\boldsymbol{I},\boldsymbol{I}')\left\|\phi(\boldsymbol{I}\,|\,\boldsymbol{\theta}) - \phi(\boldsymbol{\theta},\boldsymbol{I}')\right\|^2 \tag{1.104}$$

where w is a weight assigned to the image pair $(\boldsymbol{I},\boldsymbol{I}')$ that is designed to be large for "close" examples and is small for "distant" examples, see [303] for details. Note, the first expectation is approximated using samples from the labeled set, while the second expectation is approximated using both labeled and unlabeled training datasets.

The ideas of Weston et al. [303] have been developed and extended to semantic segmentation deep networks in the works of Tang et al. [279, 280]. They propose *regularized losses* for semantic segmentation, which provides the state-of-the-art results. Tang et al. [279, 280] combine partial cross entropy loss (1.103) with interactive segmentation objectives (energies). We reviewed these interactive segmentation methods in detail in Section 1.2.2. In short, the network training loss becomes:

$$L_{reg}(\boldsymbol{\theta}) \quad = \quad L_{pce}(\boldsymbol{\theta}) \quad + \quad \mathbb{E}_{\boldsymbol{I}}\; R(\phi(\boldsymbol{I}\,|\,\boldsymbol{\theta})) \tag{1.105}$$

[11] Note that unlike (1.95), in image classification (1.104) a single label y is assigned to image $\boldsymbol{I}$ and the network output is defined accordingly as $\phi(\boldsymbol{I}\,|\,\boldsymbol{\theta}) \in \Delta_L$.

where regularizer $R(\boldsymbol{y})$ enforces a regularity constraint on prediction $\boldsymbol{y}$, for example, alignment with edges or smoothness. One important problem with regularized losses is that many interesting regularizers, *e.g.* [31, 147, 72, 213], are defined only for discrete labelings while neural networks produce real-valued predictions in form of distribution. In addition, the loss function should be differentiable. As a result, loss (1.105) requires using either relaxed regularizers, such as ones reviewed in Section 1.2.5, or the ones defined on real values.

Tang *et al.* [280] incorporated the bilinear-relaxation (1.52) of the dense Potts model (1.40) into network loss:

$$R(\boldsymbol{s}) \quad = \quad (1-\boldsymbol{s})^{\top}\boldsymbol{W}\boldsymbol{s} \tag{1.106}$$

where vector $\boldsymbol{s} \in \Delta_L^{n\times m}$ and $\boldsymbol{W}$ is the matrix of Gaussian weights, see [280] for details. See Section 1.2.5 for discussion on various relaxations and their optimization in the context of low-level segmentation. In the deep learning context, the regularized segmentation loss (1.105) is optimized based on the SGD as any other neural network.

While SGD optimization works well for certain regularizers, *e.g.* dense random walker (1.50) [303] and bilinear relaxation (1.52) of dense Potts model (1.40) [279, 280], we show in Chapter 4 that it fails [187] to optimize some interesting and strong regularizers, such as sparse Potts model (1.37). We propose a new *alternating direction methods* (ADM) approach for weakly supervised semantic segmentation. Based on splitting, it allows using many previously developed low-level continuous and combinatorial optimizers including those reviewed in Sections 1.2.4 and 1.2.5.

On Regularization Approaches in Full Supervision Loss (1.101) is employed by a vast majority of DNN segmentation works. Regularization of the segmentation produced by a network may not be beneficial as the accurately and fully labeled ground truth contains all information about shapes. So, imposing additional assumptions in the form of regularization may be suboptimal. An exception, however, is the case of noisy labels that can be addressed by regularization.

Note that using regularized loss during training is different from incorporating low-level segmentation methods as layers which are used both in training and inference. The former aims to make a network to predict solutions that conform with the regularization. The latter achieves regularity by explicitly incorporating additional (low-level) optimization into the network pipeline. For example, Zheng *et al.* [325] incorporated mean-field solver [153] for the dense (fully-connected) CRF Potts model (1.40) as a network layer.

1.6.2 Other Regularization Approaches

One simple idea on weakly supervised learning is using classic interactive segmentation methods to compute full labeling by completing the partial labeling [172, 143]. Such full labeling ("fake" ground truth) is then used for training. We refer to such methods as *proposal generation*. This approach is simple but requires highly accurate interactive segmentation performance, often at a cost of many specialized heuristics and tricks, to achieve good training results [143]. The reason can be the early commitment to mistakes: once a mistake is allowed in the generated labels it adversely affects the network training.

Expectation maximization approaches [221] use the EM algorithm to incorporate prior knowledge of the joint distribution of level tags and pixel labels. This in particular allows imposing certain volume biases on segments. Note, that many classic interactive image segmentation methods also have volumetric bias [30] and therefore many proposal generation methods implicitly rely on the volume bias.

Most of the image-level supervision methods rely on *class activation maps* (CAM) of classification networks [327, 259, 14, 217, 148, 5]. CAM highlights the most discriminative regions on the image and is based on gradient back-propagation [327, 259]. Similar to MIL methods CAMs are biased to segment small discriminative parts of the objects rather than whole objects. So methods rely on volume expansion loss terms [148, 5] to expand the segmented region.

1.7 Motivation and Contributions

At the end of the introduction, we summarise the research problems addressed in this thesis based on discussion of the literature in the previous sections. The contributions presented in the thesis are split into three chapters addressing different problems. Each chapter is a fairly independent module with its own introduction and related work review, which can be read separately. The chapters are based on the following publications: [184, 323, 186, 187, 183].

In Chapter 2, we focus on the unsupervised vascular tree extraction problem in large 3D volumes containing complex "entanglements" of near-capillary vessels, see Figure 1.15. We develop a new general high-order model directly regularizing the centerline's curvature. Our model alleviates significant limitations of the standard surface regularization methods due to their severe shrinking bias in the context of thin structures. In general, the shrinking bias is a well-known example of a bias in standard regularization methods [26]. Part of this thesis was devoted to characterizing new forms of biases in classical segmentation models

that were not previously understood. In Chapter 3, we develop new theories establishing data density biases in standard kernel clustering. This theoretical understanding inspires our new segmentation algorithms avoiding such biases. Chapters 2 and 3 discuss regularized objectives in the context of unsupervised segmentation. Such objectives are also known to be useful for DNN semantic segmentation [325, 53] and particularly in weakly-supervised settings [148, 172, 143, 280]. In the last Chapter 4, we demonstrate that optimization of such losses is problematic for the SGD dominating deep learning. This is expected as the gradient descent gives much weaker results than more advanced optimization in low-level segmentation, see Section 1.2.4. We develop a new algorithm based on trust-region that can be seen as a high-order chain rule or backpropagation. Our approach enables the use of practically any low-level (high-order) regularization and their known global solvers in deep learning. We achieve the state-of-the-art results in weakly-supervised semantic segmentation using a well-motivated Potts model (1.37) and alpha-expansion solver [29].

Subsections 1.7.1, 1.7.2 and 1.7.3 below provide more detailed overview of the contributions in Chapters 2, 3 and 4, correspondingly.

1.7.1 Curvature Regularization for Vessel Tree Extraction

Section 1.1.1 identified the shrinking bias as a major limitation of the basic pairwise MRF (1.37) and other surface regularization discussed in Section 1.2. This is due to the fact that the model approximates the length of the segments' boundaries in 2D and surface area in 3D [26]. This manifests itself in the severe inadequacy of such MRF models for the task of segmenting thin objects such as edges, surfaces, roads, blood vessels, and neurons [206, 184, 211, 213]. See examples in Figure 1.2. We noted that prior works on regularization, *e.g.* based on curvature [206, 213, 123, 266, 70], do not have this limitation. However, these approaches are either computationally expensive or suffer from discretization artefacts. The complexity increases dramatically due to new high resolution imaging. In particular, we focus on challenging noisy micro-CT 3D images of blood vessels containing large scale trees with thousands of bifurcations and where the vast majority of vessels are near-capillary, see Figure 1.15. The sheer volume of data makes it practically impossible to employ supervised methods due to the prohibitive cost of annotations.

To address this problem, we propose and analyze a new unsupervised vasculature extraction model based on curvature regularization and show how to efficiently optimize it in Chapter 2. The contributions in the chapter were published in [184]. Unlike most previous approaches, we simultaneously detect and delineate large-scale thin structures with sub-pixel localization and real-valued orientation estimation. Unlike prior work,

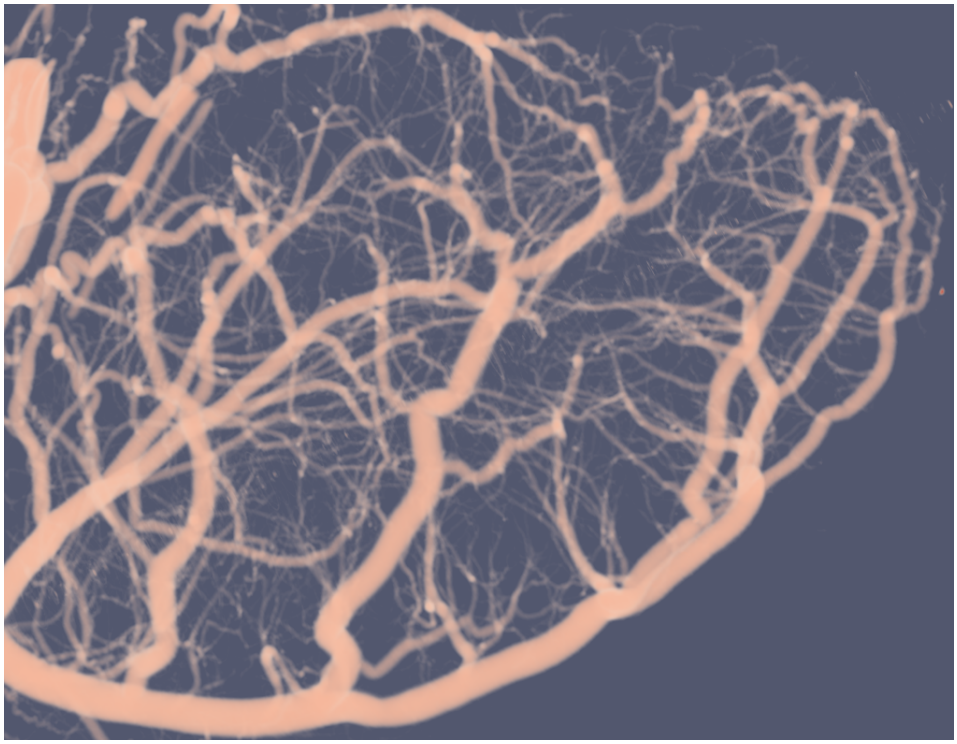


Figure 1.15: Example of thin structures: vessels in 3D volume. Due to the shrinking bias (see Section 1.2.3) there is no hope to successfully employ the basic pair-wise regularization (1.37) for vessel segmentation.

which either penalizes the segment shape [206, 213] or employs fixed topology (*e.g.* snakes) [123, 266, 70, 9], we regularize the vessels centerlines, which can form a tree of an arbitrary topology. Our objective function combines detection likelihoods with a prior minimizing curvature of the centerlines. Our optimization algorithm applies to quadratic or absolute curvature. We show that a detection system built on it achieves the state-of-the-art results in blood vascular tree reconstruction in synthetic and real data. Our work extends to other thin structure detection problems such as low-level edge detection.

1.7.2 Density Biases: New Theories and Algorithms

The shrinking bias discussed in the previous section is a manifestation of a larger phenomenon. As demonstrated by Geman and Geman [97], any MRF model bears some probabilistic assumptions about the shape of the segments, see Section 1.2. The same is

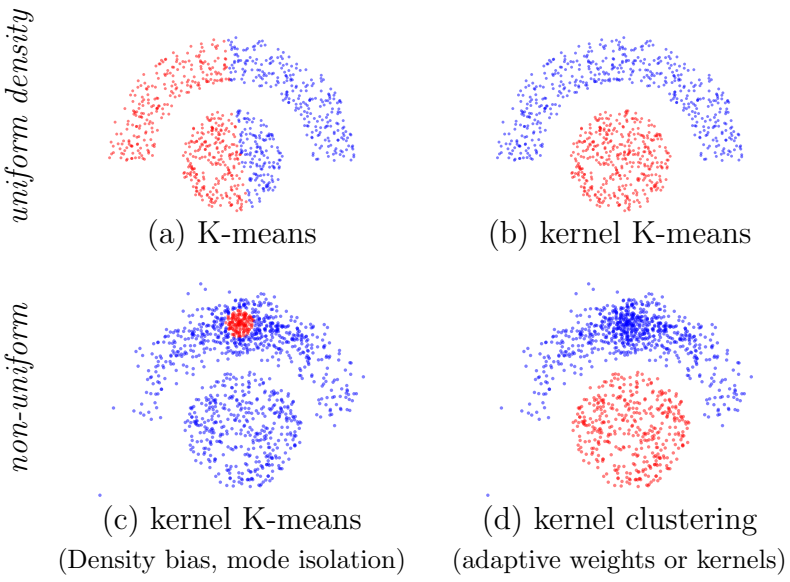
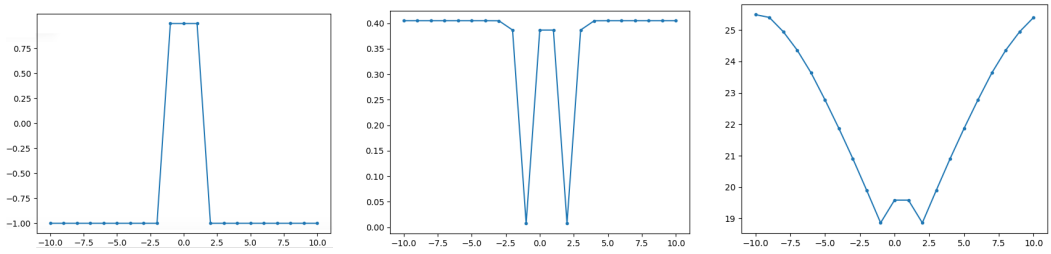


Figure 1.16: Kernel K-means with Gaussian kernel (3.1) gives desirable nonlinear separation for *uniform* density clusters (a,b). But, for *non-uniform* clusters in (c) it either isolates a small dense "clump" for smaller bandwidth σ due to the density bias or gives results like (a) for larger σ. No fixed σ yields solution (d) given by our locally adaptive kernels or weights eliminating the bias, see Chapter 3.

true for other objectives. In particular, widely-used pairwise (kernel) clustering objectives, reviewed in Section 1.3.2, also have biases. Shi and Malik [253] and others empirically observed that kernel clustering has a practically significant bias to small dense clusters, see Figure 1.16(c). However, its causes have not been understood theoretically, even though many attempts were made to improve the results [253, 318, 281].

In Chapter 3, we analyze these kernel clustering objectives and theoretically establish *data density biases*. Our Theorem 3.2 for kernel k-means (1.79) links this bias to the data density stating that the kernel k-means solution isolates the data density modes under mild conditions. Our analysis extends to other kernel clustering objectives relating their biases to the density mode isolation. The contributions in the chapter were published in [186]. These findings suggest that a principled solution for these biases should directly address data density inhomogeneity. In particular, we show that density equalization can be implicitly achieved using either our locally adaptive weights or a general class of Riemannian (geodesic) kernels, see Figure 1.16(d). Our density equalization principle unifies many popular kernel clustering criteria including *normalized cut* (1.81), which we show has a bias to sparse subsets inversely related to the kernel k-means bias. Our synthetic



(a) 1D image (b) grid CRF (1.37) (c) dense CRF (1.40)

Figure 1.17: Synthetic segmentation example for grid and dense CRF (Potts) models: (a) intensities $I(x)$ on 1D image. The cost of segments $S^t = \{x \mid x < t\}$ with different discontinuity points t according to (b) nearest-neighbor (grid) Potts and (c) larger-neighborhood (dense) Potts. The latter gives smoother cost function, but its flatter minimum may complicate discontinuity localization as shown in Figure 1.18.

(a) image + seeds (b) grid CRF (1.37) (c) dense CRF (1.40)

Figure 1.18: Low-level segmentation example for sparse (b) and dense (c) Potts models for image with seeds (a). Sparse Potts gives smoother segment boundary with better edge alignment, while dense CRF inference often gives noisy boundary.

and real data experiments illustrate these density biases and proposed solutions.

1.7.3 Higher-order Optimization for Regularized DNNs Losses

In the context of semantic segmentation, regularization of the segments' shape is commonly integrated into DNN architectures [325, 53, 148]. Since the MRF/CRF regularization is designed to address the ill-posedness and lack of supervision in low-level vision [97, 138, 41, 31, 66], it is particularly well suited for the weakly-supervised DNN segmentation [148, 172, 143, 280], see Section 1.6.

From optimization point of view, there are significant limitations for shape regulariza-

tion in DNN segmentation. In general, deep learning is dominated by the first-order optimization algorithms based on the gradient descent. Both loss functions and architectures are often explicitly tuned to be better amenable to these standard local optimization methods. Notably, common regularization approaches in DNN segmentation [148, 325, 53, 280] are limited to regularizers that are easy for the gradient descent, in particular the "smooth" *dense* Potts [153] model in Figure 1.17(c).

In Chapter 4, we consider a well-motivated regularized loss function, *i.e.* *grid* Potts [97, 31] in Figure 1.17(b), which has stronger regularization properties [293] than the *dense* Potts model, see an example in Figure 1.18. However, it cannot be optimized by the gradient descent effectively due to its highly non-convex nature, see Figure 1.17(b). We propose a new alternative higher-order optimizer for deep network training. Our principled algorithm combines standard neural network training, reviewed in Section 1.5.3, with the alpha-expansion algorithm [29], a standard solver for the grid Potts model. Our optimization approach improves the state-of-the-art in weakly-supervised segmentation using the grid Potts model, where SGD performs poorly. As we show, SGD's best result requires "smoother" dense Potts model, but it is still significantly weaker than our state-of-the-art based on the stronger grid Potts regularizer. Our general approach can train segmentation networks using practically any low-level MRF/CRF regularizers and their solvers, see Sections 1.2.4 & 1.2.5. The contributions in the chapter were published in [187, 183].

Chapter 2

Curvature for Thin Structures

Many applications in vision require estimation of thin structures such as boundary edges, surfaces, roads, blood vessels, neurons, etc. Unlike most previous approaches, we simultaneously detect and delineate thin structures with sub-pixel localization and real-valued orientation estimation. This is an ill-posed problem that requires regularization. The standard length or area-based regularizers are severally inadequate due to their strong shrinking bias, see Section 1.2.3. We propose an objective function combining detection likelihoods with a prior minimizing curvature of the center-lines. Unlike simple block-coordinate descent, we develop a novel algorithm that is able to perform joint optimization of location and detection variables more effectively. Our lower bound optimization algorithm applies to quadratic or absolute curvature. The proposed early vision framework is sufficiently general and it can be used in many higher-level applications. We illustrate the advantage of our approach on a range of 2D and 3D examples.

2.1 Background and related work

This chapter is focused on the general concept of a *center-line*, which could be defined in different ways. For example, the Canny approach to edge detection implicitly defines a center-line as a "ridge" of intensity gradients [39]. Standard methods for shape skeletons define medial axis as singularities of a distance map from a given object boundary [254, 255]. In the context of thin objects like edges, vessels, etc, we consider a center-line to be a smooth curve minimizing orthogonal projection errors for the points of the thin structure.

We study the curvature of the center-line as a regularization criterion for its inference. In general, the curvature is actively discussed in the context of thin structures. For example,

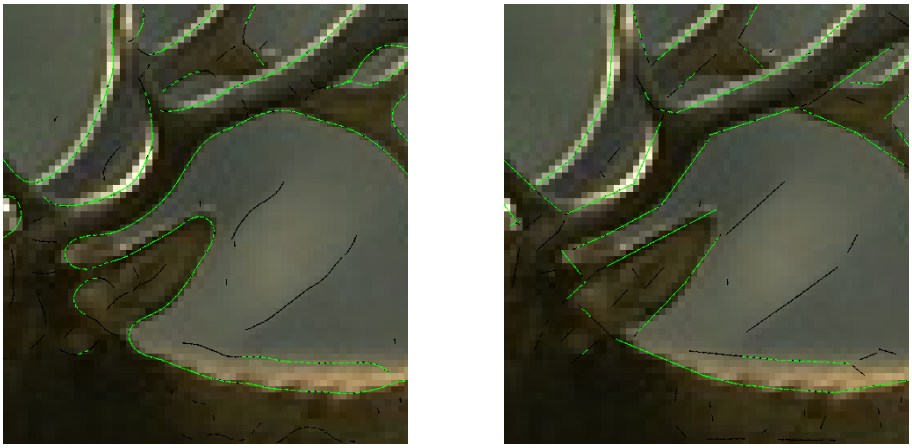
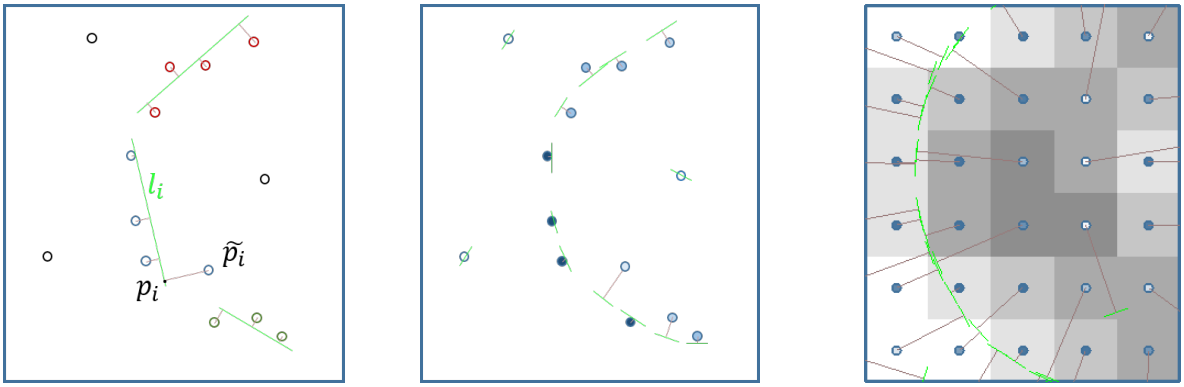


Figure 2.1: Curvature regularization based edge detection. The result of our algorithm for squared (on the left) and absolute (on the right) curvature approximations. Green and black lines correspond to edges with high and medium confidence measure correspondingly. Note the strong bias to straight lines on the right: the energy prefers a small number of sharp corners rather than many smooth corners like on the left.

it is well known that curvature of the object boundary has a significant effect on the medial axis [144, 254]. In contrast, we are directly concerned with the curvature of the center-line, not the curvature of the object boundary. Moreover, we do not assume that the boundary of a thin structure (*e.g.* vessel or road) is given. Detection variables are estimated simultaneously with the center-line. This paper proposes a general energy formulation and an optimization algorithm for the detection and subpixel delineation of thin structures based on curvature regularization.

Curvature is a natural regularizer for thin structures and it has been widely explored in the past. In the context of image segmentation with second-order smoothness it was studied by [245, 265, 246, 32, 117, 213, 206]. It is also a popular second-order prior in stereo or multi-view-reconstruction [168, 212, 306]. Curvature has been used inside connectivity measures for analysis of diffusion MRI [198]. Curvature is also widely used for *inpainting* [4, 45] and edge completion [113, 305, 3]. For example, *stochastic completion field* technique in [305, 198] estimates probability that a completed/extrapolated curve passes any given point assuming it is a random walk with bias to straight paths. Note that common edge completion methods use existing edge detectors as an input for the algorithm.

In contrast to these prior works, this paper proposes a general low-level regularization framework for detecting thin structures with accurate estimation of location and orienta-



(a) Olsson's model [211] (b) Our model for cloud of points (c) Our model for grid points

Figure 2.2: Comparison with [211]. An empty circle in (b) and (c) denotes low confidence and a dark blue circle means high confidence.

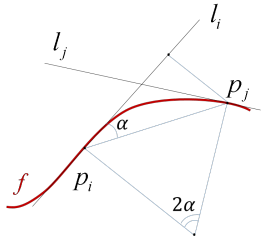
tion. In contrast to [305, 113, 198] we explicitly minimize the integral of curvature along the estimated thin structure. Unlike [112] we do not use curvature for grouping pre-detected thin structures, we use curvature as a regularizer during the detection stage.

2.1.1 Curvature for thin structures

Our regularization framework is based on the curvature estimation formula proposed by Olsson et al. [211, 212] in the context of surface fitting to point clouds for multi-view reconstruction, see Figure 2.2(a). One assumption in [211, 212] is that the data points are noisy readings of the surface. While the method allows outliers, their formulation is focused on estimation of local surface patches. Our work can be seen as a generalization to detection problems where majority of the data points, *e.g.* image pixels in Figure 2.2(c), are not within a thin structure. In addition to local tangents, our method estimates probability that the point is a part of the thin structure. Section 2.2 discusses in details this and other significant differences from the formulation in [211, 212].

Assuming $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$ are neighboring points on a thin structure, *e.g.* a curve, Olsson and Boykov [211] evaluate the local curvature as follows. Let l_i and l_j be the tangent lines to the curve at points $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$. Then the authors propose the following approximation for the absolute curvature

$$|\kappa(l_i, l_j)| = \frac{\|l_i - \boldsymbol{p}_j\| + \|l_j - \boldsymbol{p}_i\|}{\|\boldsymbol{p}_i - \boldsymbol{p}_j\|} \tag{2.1}$$



and for the squared curvature

$$\kappa^2(l_i, l_j) = \frac{\|l_i - \boldsymbol{p}_j\|^2 + \|l_j - \boldsymbol{p}_i\|^2}{\|\boldsymbol{p}_i - \boldsymbol{p}_j\|^2} \tag{2.2}$$

where $\|l_i - \boldsymbol{p}_j\|$ is the distance between point $\boldsymbol{p}_j$ and line l_i.

Assume that the curve $\boldsymbol{r} = f(\tau)$ is parameterized by arc-length τ such that $\tau_1 \leq \tau \leq \tau_M$. If $(\tau_1, \tau_2, \ldots, \tau_M)$ is an increasing parameter sequence then the curvature of f can be approximated by

$$\int |\kappa|^\alpha d\tau \quad \approx \sum_{(i,j)\in N} |\kappa(l_i, l_j)|^\alpha \tag{2.3}$$

where $N = \{(i, i+1) \mid i = 1, 2, \ldots M-1\}$ is a neighborhood system for curve points $\boldsymbol{p}_i = f(\tau_i)$ and l_i is its tangent line at point $\boldsymbol{p}_i$.

Olsson and Boykov [211] use regularization for fitting a surface (or curve) to a cloud of points in 3D (or 2D) space. Every observed point $\tilde{\boldsymbol{p}}_i$ is treated as a noisy measurement of some unknown point $\boldsymbol{p}_i$ that is the closest point on the estimated surface, see Figure 2.2(a). Each $\tilde{\boldsymbol{p}}_i$ is associated with unknown local surface patch l_i that is a tangent plane for the surface at $\boldsymbol{p}_i$. The proposed surface fitting energy combines curvature-based regularization with the first order data fidelity term

$$E(L) = \sum_{(i,j)\in N} |\kappa(l_i, l_j)|^\alpha w_{ij} + \sum_i \frac{1}{\sigma^2} \|l_i - \tilde{\boldsymbol{p}}_i\|^2 \tag{2.4}$$

where $L = \{l_i\}$ is the set of tangents, N is a neighborhood system, σ is non-negative constant, w_{ij} is a positive constant such that $\sum_{j\in N_i} w_{ij} = 1$. To minimize (2.4), the algorithm in [211] iteratively optimizes the assignment variables for a limited number of tangent proposals, and then re-estimates tangent plane parameters, see Figure 2.2(a).

In contrast to [211], our method estimates thin structures in the image grid where, *a priori*, it is unknown which pixels belong to the thin structure, see Figure 2.2(c). We introduce set $X = \{x_i\}$ of indicator variables $x_i \in \{0, 1\}$ where $x_i = 1$ iff pixel $\tilde{p}_i$ belongs to the structure. The coordinates of pixel $\tilde{p}_i$ are denoted $\tilde{\boldsymbol{p}}_i \in \mathbb{R}^2$. Our basic energy (2.5) and its extensions combine unary detection potentials with curvature regularization. Due to the regularity of our grid neighborhood, we use constant weights w_{ij}, which are omitted from now on. We propose a different optimization technique estimating a posteriori distribution of x_i and separate tangents l_i at each point. As illustrated in Figure 2.2(b), our framework is also applicable to energy (2.4) and multi-view reconstruction problem as in [211, 212].

Parent and Zucker [223] formulate a closely related *trace inference* problem for detecting curves in 2D grid. Similarly to us, they estimate indicator variables x_i and tangents l_i. However, they estimate x_i and l_i by enforcing a *co-circularity* constraint assuming given local *curvature information*, which they estimate in advance. In contrast, we simultaneously estimate x_i and l_i by optimizing objective (2.5) that directly regularizes curvature of the underlying thin structure. Moreover, [223] quantizes curvature information and tangents while our model uses real valued curvature and tangents. The extension of [223] to 3D is not trivial.

Similarly to [211, 223] we estimate tangents only at a *finite* set of points. Additional regularization is required if continuous center-line between these points is needed [136].

2.1.2 Unsupervised vasculature estimation methods

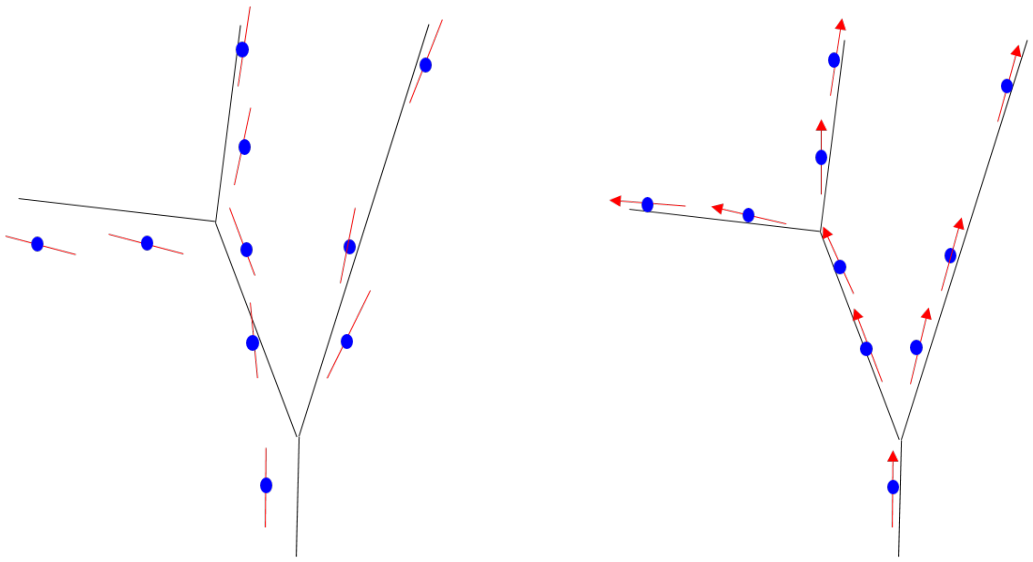
Unsupervised vessel tree estimation methods for complex high-resolution volumetric vasculature data, such as in Figure 1.15, combine low-level vessel filtering and algorithms for computing global tree structures based on constraints from anatomy, geometry, physics, *etc.* Below we review the most relevant standard methodologies.

Low-level vessel estimation: Anisotropy of tubular structures is exploited by standard vessel filtering techniques, *e.g.* Frangi et al. [90]. Combined with non-maximum suppression, local tubularity filters provide estimates for vessel centerline points and tangents, see Figure 2.3(a). Technically, elongated structures can be detected using intensity Hessian spectrum [90], *optimally oriented flux* models [162, 287], steerable filters [91], path operators [193] or other anisotropic models. Dense local vessel detections can be denoised using curvature regularization [223].

This work proposes a method that can be seen as a low-level vessel detection based on curvature regularization. Our state-of-the-art follow-up works [322, 323] use prior knowledge about divergence or convergence of vessel trees (arteries vs veins) to estimate an *oriented* flow pattern, see Figure 2.3(b), which significantly improves the quality of bifurcation reconstruction.

Thinning: One standard approach to vessel topology estimation is via *medial axis* [254]. This assumes known vessel segmentation (volumetric mask) [194], which can be computed only for relatively thick vessels. Well-formulated segmentation of thin structures requires Gaussian- or min-curvature surface regularization that has no known practical algorithms. Segmentation is particularly unrealistic for sub-voxel vessels.

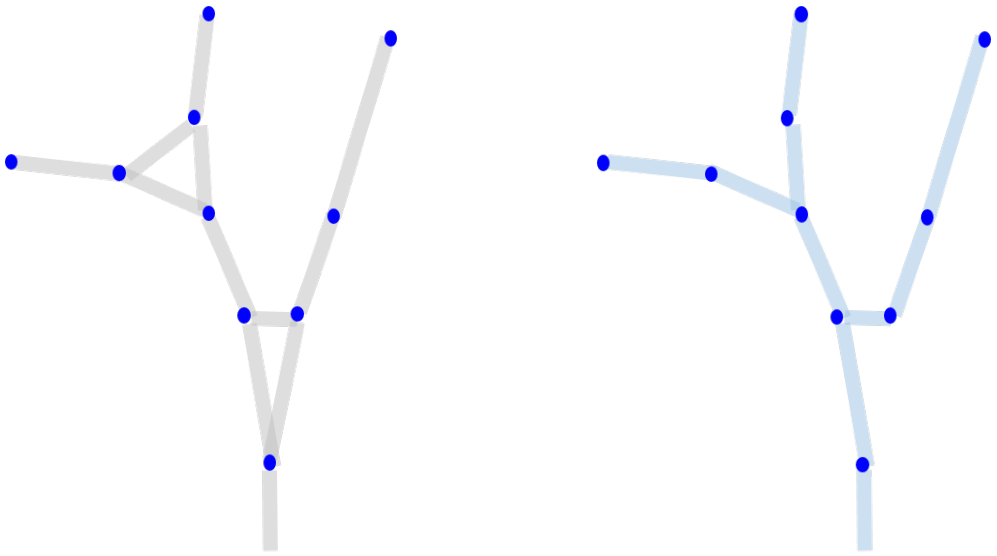
Geodesics and shortest paths: Geodesics [75, 50] and shortest paths [87] are often used for AB-interactive reconstruction of vessels between two specified points. A vessel is



(a) Frangi filtering [90]

(b) *oriented* flow pattern [322]

Figure 2.3: *Low-level vessel estimation*: True centerline is black. Blue voxels in (a) are local maxima of some tubularity measure [90, 162, 287, 91] in the direction orthogonal to the estimated centerline tangents (red). Our regularization can estimate subpixel centerline points (b) and oriented tangents in the follow-up [322] (red flow field).



(a) geodesic tubular graph (b) MST

Figure 2.4: *Global vessel tree reconstruction*: (a) geodesic tubular graph is based on low-level estimates in Figure 2.3. Graph edges represent distances, geodesics, or other symmetric (undirected) properties. MST reconstruction quality (b) depends on the graph construction (nodes, neighborhoods, edge weights).

represented by the shortest path with respect to some anisotropic continuous (Riemannian) or discrete (graph) metric based on a local tubularity measure. Interestingly, the minimum path in an "elevated" search space combining spatial locations and radii can simultaneously estimate the vessel's centerline and diameter, implicitly representing vessel segmentation [169, 18]. Unsupervised methods widely use geodesics as their building blocks.

Spanning trees: The standard graph concept of a *minimum spanning tree* (MST) is well suited for unsupervised reconstruction of large trees with unknown complex topology [103, 310, 287, 199]. MST is closely related to the *shortest paths* and *geodesics* since its optimality is defined with respect to its length. Like shortest paths, globally optimal MST can be computed very efficiently. In contrast to the shortest paths, MST can reconstruct arbitrarily complex trees without user interaction.

The quality of MST vessel tree reconstruction depends on the underlying graph construction, see Figure 2.4. Graphs designed for reconstructing thin tubular structures as their spanning tree (or sub-tree) are often called *tubular graphs*. Typically, the nodes are "anchor" points generated by low-level vessel estimators, *e.g.* see Figure 2.3. Such anchors

represent sparse [288] or semi-dense (this work) samples from the estimated tree structure that may be corrupted by noise and outliers. Pairwise edges on a tubular graph typically represent distances or geodesics between the nodes, as in AB-interactive methods discussed earlier. Such graphs are called *geodesic tubular graphs*.

There are numerous variants of tubular graph constructions designed to represent various thin structures as MST [103, 310, 287, 199] or shortest path trees [231]. There are also interesting and useful extensions of MST addressing tubular graph outliers, *e.g.* k-MST [286] and integer programming technique in [288]. Such approaches are more powerful as they seek minimum sub-trees that can automatically exclude outliers. However, the corresponding optimization problems are NP-hard and require approximations. Such methods are expensive compared to the low-order polynomial complexity of MST. They are not practical for dense reconstruction problems in high-resolution vasculature volumes.

2.1.3 Contributions

It is known that curvature of an object boundary is an important shape descriptor [250] with a significant effect on *medial axis* [144, 254], which is not robust even to minor perturbations of the boundary. In the context of thin objects (*e.g.* edges, vessels) we study a concept of a center-line (a smooth 1D curve minimizing the sum of projection errors), which is different from *medial axis*. We regularize the curvature of the center-line. Unlike many standard methods for center-lines, we do not assume that the shape of the object is given and propose a general low-level vision framework for thin structure detection combined with sub-pixel localization and real-valued orientation of its center-line. Therefore, we propose an approach that takes into account all possible configurations of the indicator variables while estimating the tangents. This significantly improves stability with respect to local minima. Our optimization method uses variational inference and trust region frameworks adapted to absolute and quadratic curvature regularization.

Our proof-of-the-concept experiments demonstrate encouraging results in the context of edge and vessel detection in 2D and 3D images. In particular, we obtain promising results for estimating highly detailed vessels structure on high-resolution microscopy CT volumes. We also show examples of sub-pixel edge detection regularizing curvature. While there are no databases for comparing edge detectors with real-valued location and orientation estimation, we obtained competitive results on a pixel-level edge detection benchmark [111]. Our general early vision methodology can be integrated into higher semantic level boundary detection techniques, *e.g.* [189], but this is outside the scope of this work. Our current sequential implementation is not tuned to optimize performance. Its running time for edges

in 2D image of Figure 2.1 is 20 seconds and for vessels in 3D volume of Figure 2.12 is one day. However, our method is highly-parallelizable on GPU and fast real-time performance on 2D images can be achieved.

In Section 2.2 we describe the proposed model and discuss a simple block-coordinate descent optimization algorithm and its drawbacks. In Section 2.3.2 we propose a new optimization method for our energy based on variational inference framework. In Section 2.3.3 we describe the details of the proposed method and discuss the difference between squared and absolute curvatures (Section 2.3.4). We describe several applications of the proposed framework in Section 2.4 and conclude in Chapter 4.5.2.

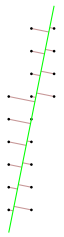
2.2 Energy Formulation

In the introduction we informally defined the center-line of a thin structure as a smooth curve minimizing orthogonal projection errors. Here we present the energy formalizing this criterion. First we note that in our model the curve is not defined explicitly but through points $\boldsymbol{p}_i$ it passes and tangent lines l_i at these points. The energy is given by

$$E\ (L, X) = \sum_{(i,j)\in N} \kappa^2(l_i, l_j) x_i x_j + \\ + \sum_i \frac{1}{\sigma^2} \|l_i - \tilde{\boldsymbol{p}}_i\|_+^2 x_i + \sum_i \lambda_i x_i \tag{2.5}$$

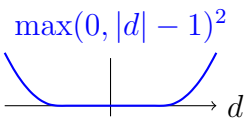
where N is a neighborhood system, $X = \{x_i\}$ is a set of indicator variables $x_i \in \{0, 1\}$ where $x_i = 1$ iff pixel $\tilde{p}_i$ belongs to the thin structure, λ_i define unary potentials penalizing/rewarding presence of the structure at $\tilde{p}_i$. In contrast to (2.4), potentials λ_i define the data term while $\frac{1}{\sigma^2}\|l_i - \tilde{\boldsymbol{p}}_i\|_+^2$ is a soft constraint.

We explore two choices of the soft constraint $\|l_i - \tilde{\boldsymbol{p}}_i\|_+$. The first one uses Euclidean distance. In that case it models normally distributed errors. Although it is appropriate for many applications, *e.g.* surface estimation in multi-view reconstruction [211, 212], the normal errors assumption is no longer valid for the image grid because the discretization errors are not Gaussian. In fact, using Euclidean distance may make the soft constraint term proportional to the length of the center-line, see illustration on the right.



Thus, we also propose a truncated form of Euclidean distance:

$$\|l_i - \tilde{\boldsymbol{p}}_i\|_+ = \max(0, \|l_i - \tilde{\boldsymbol{p}}_i\| - 1). \tag{2.6}$$



This does not penalize tangent lines l_i that are within one pixel from points $\tilde{p}_i$. Different applications may require a different choice of no-penalty threshold.

Extensions. We can extend the energy (2.5) by adding other terms that encourage various other useful properties. For example, energy

$$E'(L, X) = E(L, X) - \gamma \sum_{(i,j)\in N} x_i x_j \tag{2.7}$$

for $\gamma > 0$ will reward well aligned tangents. The effect of this term is shown in Figure 2.8. This term is similar to edge "repulsion" in MRF-based segmentation. The overall pairwise potential $(\kappa(l_i, l_j) - \gamma)x_i x_j$ encourages edge continuity.

Another extension is incorporating a prior on the center-line direction g_i at pixel $\tilde{p}_i$:

$$E'(L, X) = E(L, X) + \beta \sum_i m(l_i, g_i)^2 x_i. \tag{2.8}$$

The term $m(l_i, g_i)$ measures how well tangent line l_i is aligned with prior g_i:

$$m(l_i, g_i) = \|g_i\| \sin \angle(l_i, g_i). \tag{2.9}$$

The magnitude of g_i constitutes the confidence measure. For example, vectors g_i could be obtained from the image gradients or the eigenvectors in the vesselness measure [90].

2.3 Optimization

To motivate our optimization approach for energy (2.5) described in Section 2.3.2, first we describe a simpler optimization algorithm and discuss its drawbacks.

2.3.1 Block-coordinate Descent Optimization

The most obvious way to optimize energy (2.5) is a block-coordinate descent. The optimization alternates two steps described in Algorithm 1. The auxiliary energy optimized on line 4 is a non-linear least square problem and can be optimized by a trust-region approach, see Section 2.3.3. The auxiliary function on line 5 is a non-submodular binary pairwise energy that can be optimized with TRWS [149].

Algorithm 1: Block-coordinate descent

1 Initialize L^0 and X^0 ;
2 $k \leftarrow 0$;
3 **while** *not converged* **do**
4 | Optimize $L^{k+1} \leftarrow \arg\min_L E(L, X^k)$;
5 | Optimize $X^{k+1} \leftarrow \arg\min_X E(L^{k+1}, X)$;
6 | $k \leftarrow k + 1$;
7 **end**

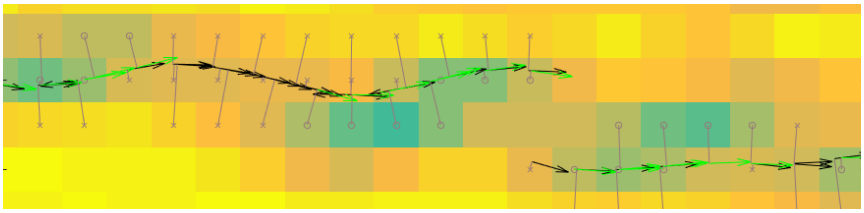
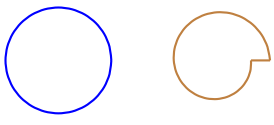


Figure 2.5: An example of local minima for block-coordinate descent Algorithm 1. The more "blue" is a pixel, the more likely it is to lie on an edge. Green arrows correspond to pixels that were initialized as edges. Black arrows correspond to the edges detected by Algorithm 1. This local minimum consists of two disconnected center-lines. The globally minimum solution smoothly connects the two pieces into a single center-line.

We found that Algorithm 1 is extremely sensitive to local minima, see Figure 2.5. The reason is that tangents l_i for points with indicator variables $x_i^k = 0$ do not participate in optimization on line 4. To improve performance of block-coordinate descent, we tried heuristics to extrapolate tangents into such regions. We found that good heuristics should have the following two properties.

First, since integral of curvature is sensitive to small local errors (see the figure on the right), the extrapolating procedure should yield close tangents for neighbors. Otherwise step 5 of the algorithm is ineffective. This issue could be partially solved by using energy (2.7). In this case it can be beneficial to connect two tangents even if there is some misalignment error.



Second, the heuristic should envision that some currently disconnected curves may lie on the same center-line, see Figure 2.5.

The first property was easy to incorporate, while the second would require sophisti-

cated edge continuation methods, *e.g.* a stochastic completion field [305, 198]. Instead we develop a new optimization procedure in Section 2.3.2 based on variational inference. The advantage of our new procedure is that it is closer to joint optimization of L and X.

2.3.2 Variational Inference

Ideally, we wish to jointly optimize (2.5) with respect to all variables. This is a mixed integer non-linear problem with an enormous number of variables. Thus, it is intractable. However, we can introduce elements of joint optimization based on stochastic *variational inference* framework. The proposed approach takes into account all possible configurations of indicator variables x_i while estimating tangents l_i. This significantly improves stability w.r.t. local minima.

Energy (2.5) corresponds to a Gibbs distribution:

$$P(I, X, L') = \frac{1}{Z} \exp\left(-E(L', X)\right) \tag{2.10}$$

where Z is a normalization constant and the image is given by data fidelity terms $I = \{\lambda_i\}$. Here I are visible variables, indicator variables X and tangents $L' = \{l'_i\}$ are hidden ones. We add a prime sign for tangent notation to distinguish values of random variables and parameters of the distribution. Our goal is to approximate the posterior distribution $P(X, L'|I)$ of unobserved (hidden) indicators X and tangents L' given image I. The problem of approximating the posterior distribution has been extensively studied and is known as *variational inference* [19].

Variational inference is based on the decomposition

$$\ln P(I) = \mathcal{L}(q) + \mathrm{KL}(q\|p) \tag{2.11}$$

where $\ln P(I)$ is the *evidence*, $q(X, L')$ is a distribution over the hidden variables, $p(X, L') = P(X, L'|I)$ is the posterior distribution, and

$$\mathcal{L}(q) = \sum_X \int q(X, L') \ln\left(\frac{P(I, X, L')}{q(X, L')}\right) dL', \tag{2.12}$$

$$\mathrm{KL}(q\|p) = -\sum_X \int q(X, L') \ln\left(\frac{P(X, L'|I)}{q(X, L')}\right) dL'. \tag{2.13}$$

Since KL (Kullback–Leibler divergence) is always non-negative, the functional $\mathcal{L}(q)$ is a lower bound for the evidence $\ln P(I)$. One of the nice properties of this decomposition is

Algorithm 2: Block-Coordinate Descend for Variational Inference

```
Initialize L^0 and Q^0 ;
k ← 0 ;
while not converged do
    Optimize L^{k+1} ← arg max_L L(Q^k, L) ;
    Optimize Q^{k+1} ← arg max_Q L(Q, L^{k+1}) ;
    k ← k + 1 ;
return L^k, Q^k
```

that the global maximum of lower bound $\mathcal{L}$ coincides with the global minimum of $\mathrm{KL}(q\|p)$ and optimal $q^*(X, L') = \arg\max_q \mathcal{L}(q)$ is equal to the true posterior $P(X, L'|I)$ [19].

Unfortunately (2.12) cannot be optimized exactly. To make optimization tractable, in variational inference framework one assumes that q belongs to a family of suitable distributions. In this work we will assume that q is a factorized distribution (*mean field theory* [224]):

$$q(X, L') = q(X)q(L'), \tag{2.14}$$

$$q(X) = \prod_i q_i(x_i) = \prod_i q_i^{x_i}(1 - q_i)^{1-x_i}, \tag{2.15}$$

$$q(L') = \prod_i \delta(l'_i - l_i) \tag{2.16}$$

where $\delta(l'_i - l_i)$ is a deterministic (degenerate) distribution with parameter l_i. Under this assumption lower bound functional $\mathcal{L}$ becomes a function of parameters q_i and l_i. We denote this function $\mathcal{L}(Q, L)$ where $Q = \{q_i\}$ and $L = \{l_i\}$.

The proposed algorithm is defined by Algorithm 2. It optimizes lower bound $\mathcal{L}(Q, L)$ in block-coordinate fashion. The algorithm returns optimal tangents l_i^*, see Figure 2.7(b), and optimal probabilities q_i^*, see Figure 2.7(c).

Now we consider optimization of $\mathcal{L}$ over L. Taking into account (2.16), (2.5) and (2.12) we can derive

$$\arg\max_L \mathcal{L}(Q^k, L) = \arg\min_L \sum_X q^k(X)E(X, L) =$$

$$= \arg\min_L \sum_{(i,j)\in N} \psi_{ij} q_i^k q_j^k + \sum_i \psi_i q_i^k. \tag{2.17}$$

where

$$\psi_{ij} \equiv \kappa^2(l_i, l_j),$$
$$\psi_i \equiv \frac{1}{\sigma^2}\|l_i - \tilde{\boldsymbol{p}}_i\|_+^2 + \lambda_i.$$

In case of (2.7) we redefine $\psi_{ij} \equiv \kappa^2(l_i, l_j) - \gamma$, and in case of (2.8) we redefine $\psi_i \equiv \frac{1}{\sigma^2}\|l_i - \tilde{\boldsymbol{p}}_i\|_+^2 + \lambda_i + \beta\, m(l_i, g_i)$.

We see that optimization of $\mathcal{L}(Q^k, L)$ with respect to L is a non-linear least square problem. For optimization details please refer to Section 2.3.3.

The optimization w.r.t. Q can be done by coordinate descent as in [19]:

$$\ln q_i^*(x_i) = \mathrm{E}_{j\neq i}[\ln P(I, X|L)] + const = \\ = -x_i \left(\sum_{j:(i,j)\in N} \psi_{ij} q_j + \psi_i \right) + const. \tag{2.18}$$

The constant in expression (2.18) does not depend on x_i and thus can be determined from the normalization equation $q_i(1) + q_i(0) = 1$. We initialize $q^0(x_i) = \exp(-x_i\psi_i)/(1 + \exp(-\psi_i))$ on line 1 of Algorithm 2. We iterate over all pixels update step (2.18) on line 5 until convergence, which is guaranteed by convexity of $\mathcal{L}$ with respect to each q_i [19].

Note that if we further restrict q to be a degenerate distribution (meaning $q(x_i) \in \{0, 1\}$) we will get the block-coordinate descend Algorithm 1.

The initialization of L^0 is application dependent. In many cases some information about direction of a thin structure is available. Concrete initialization examples are described in Section 2.4.

Alternative interpretations. The goal of Algorithm 1 is to find $\min_{L,X} E(L, X)$, which is equivalent to

$$\max_L \max_X (-E(L, X)). \tag{2.19}$$

As shown in Section 2.2 optimization of 2.19 in a block-coordinate fashion requires optimization of tangents L with fixed indicator variables X. This necessitates extrapolation of tangents. Instead we propose to optimize L taking into account all possible configurations of X. That is we propose to replace maximum with smooth maximum:

$$\max_L \sum_X \exp(-E(L, X)). \tag{2.20}$$

Then we can write down a decomposition similar to (2.11), which provides a lower bound yielding the same optimization procedure.

The proposed procedure is closely related to the EM algorithm[73] where we treat tangents L as the parameters of the distribution. However, in this case the normalization constant of the distribution depends on L and optimization problem is intractable. One possible way to fix this issue is to use a *pseudo likelihood* [170].

2.3.3 Trust Region for Tangent Estimation

Optimization of the auxiliary functions on line 4 of Algorithms 1 and 2 as well as energy (2.4) is a non-linear least square problem. In [211, 212] energy (2.4) is optimized using discrete multi-label approach in the context of surface approximation. In our work we adopt the inexact Levenberg-Marquardt method in [307], which is a trust region second order continuous iterative optimization method.

Each iteration consists of several steps. First, the method linearizes:

$$\mathcal{L}(q^k, L + \delta L) \approx \mathcal{L}(\delta L) \equiv \tag{2.21}$$

$$\equiv \sum_{(i,j)\in N} \left(|\kappa(l_i, l_j)| + \frac{\partial \kappa}{\partial l_i}\delta l_i + \frac{\partial \kappa}{\partial l_j}\delta l_j \right)^2 q_i^k q_j^k + \sum_i \frac{1}{\sigma^2} \left(\|l_i - \tilde{\boldsymbol{p}}_i\|_+ + \frac{\partial d}{\partial l_i}\delta l_i \right)^2 q_i^k \tag{2.22}$$

where for compact notation we define $\kappa \equiv |\kappa(l_i, l_i)|$ and $d \equiv \|l_i - \tilde{\boldsymbol{p}}_i\|_+$. We use [2] for automatic calculation of derivatives.

Second, the algorithm solves the minimization problem

$$\delta L^* = \arg\min_{\delta L} \mathcal{L}(\delta L) + \lambda \|\delta L\|^2 \tag{2.23}$$

where λ is a positive damping factor, which determines the trust region. The method uses an inexact iterative algorithm for this task.

The last stage of iteration is to compare the predicted energy change $\mathcal{L}(\delta L^*) - \mathcal{L}(\vec{0})$ with the actual energy change $\mathcal{L}(q^k, L + \delta L^*) - \mathcal{L}(\vec{0})$. Depending on the result of comparison the method updates variables L and damping factor λ. For more details please refer to [307].

The most computationally expensive part of Algorithm 2 is trust region optimization described in this subsection. From the technical point of view it consists of derivatives computation and basic linear algebra operations. Fortunately, these operations could be easily parallelized on GPU. We leave the GPU implementation for a future work.

2.3.4 Quadratic vs Absolute Curvature

Previous sections assume squared curvature, but everything can be adapted to the absolute curvature too. We only need to discuss how to optimize (2.17) for the absolute curvature. We use the following approximation:

$$\frac{\|l_i - \boldsymbol{p}_j\|}{\|\boldsymbol{p}_i - \boldsymbol{p}_j\|} \approx \frac{\|l_i - \boldsymbol{p}_j\|^2}{\|\boldsymbol{p}_i - \boldsymbol{p}_j\|^2} \cdot w_{ij} \tag{2.24}$$

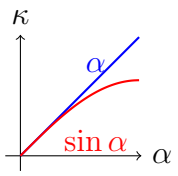
where

$$w_{ij} = \frac{\|\boldsymbol{p}_i - \boldsymbol{p}_j\| + \epsilon}{\|l_i - \boldsymbol{p}_j\| + \epsilon} \tag{2.25}$$

and ϵ is some non-negative constant. If $\epsilon = 0$ we have an approximation of the absolute curvature, if $\epsilon \to \infty$ we have an approximation of the squared curvature.

The trust region approach (see Section 2.3.3) works with approximations of functions. It does not require any particular approximation like in the Levenberg-Marquardt method [167, 307]. Thus we can approximate the absolute curvature by treating w_{ij} as constants in (2.24) and linearizing $\kappa(l_i, l_j)$ analogously to the squared curvature case.

The approximation of curvature given by [211] is derived under the assumption that the angles between neighbor tangents are small. Under this assumption the sine of an angle is approximately equal to the angle. And the approximation essentially computes the sines of the angles rather than the angles themselves. As a result it significantly underestimates the curvature of sharp corners.



For example, let us consider the integral of absolute curvature over a circle and a square. The integral of the approximation is 2π and 4 correspondingly, while the integral of the true absolute curvature is 2π in both cases. So the energy using this approximation of absolute curvature tends to distribute curvature into a small number of sharp corners showing strong bias to straight lines. Although approximation of squared curvature also underestimates curvature of sharp corners, it does not have a strong bias to straight lines. See figures 2.1 and 2.6 for comparison of the approximations.

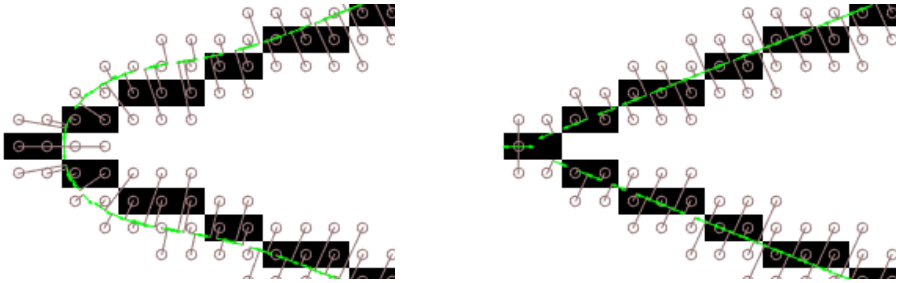
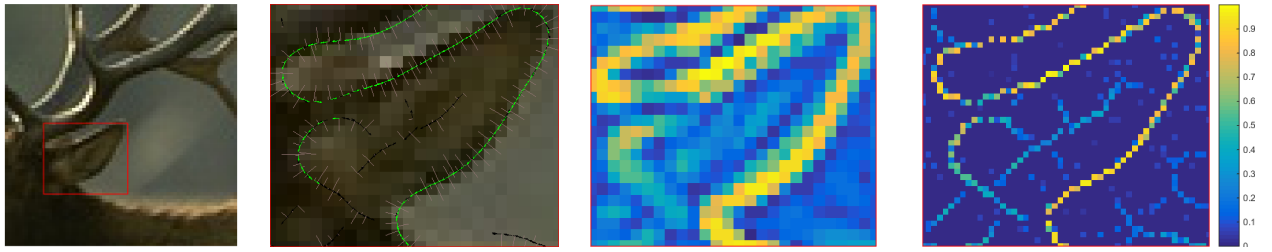


Figure 2.6: The difference between squared (left) and absolute (right) curvature approximations on an artificial example. Note the ballooning bias of squared curvature.



(a) Original image (b) Estimated edges (c) Probabilities q_i (d) Subpixel probabilities $\tilde{q}_p$

Figure 2.7: The result of the proposed algorithm. The original image is shown on (a). The zoomed in region is shown with a red box. Estimated tangents are shown in (b). Green color denotes tangents corresponding to pixels $\tilde{p}_i$ such that $q_i \geq \frac{1}{2}$, and tangents corresponding to pixels with $q_i \geq \frac{1}{4}$ are shown in black. (c) shows probabilities q_i. (d) shows the probabilities at doubled resolution produced by projecting points to their tangents: $\tilde{q}_p = q_i$.

2.4 Applications

2.4.1 Contrast Edges

Here we consider an application of our method to edge detection and real-valued edge localization.

Sobel gradient operator [262] returns the gradient magnitude and direction for every image pixel. The high gradient magnitude is an evidence of a contrast edge. The direction of the gradient is a probable direction of the edge. We use the output of the gradient operator to define data fidelity terms of energy (2.7). For every pixel $\tilde{p}_i$ let g_i be the gradient vector returned by the operator. We normalize vectors g_i by the sample variance of their magnitudes over the whole image. We define likelihood λ_i using hand picked

(a) $\gamma = 0$ (b) $\gamma = 0.25$

Figure 2.8: The effect of γ in energy (2.7). Tangents l_i whose $q_i \geq \frac{1}{2}$ are shown in green, tangents such that $\frac{1}{4} < q_i < \frac{1}{2}$ are shown in black. Increasing γ results in increasing probabilities q_i of well aligned tangents.

Figure 2.9: Examples of the output. The first row shows original images from CFGD database [111]. The second row shows edge masks at the original resolution produced by our algorithm.

linear transformation of the gradient magnitude: $\lambda_i = 1.8 - 1.4 \cdot \|g_i\|$. These parameters were optimized on a single picture shown in Figure 2.7(a). The initial tangents (line 1 of Algorithm 2) l_i are collinear with gradients g_i and pass through pixels $\tilde{p}_i$.

The results in figures 2.1, 2.7-2.11 were obtained by optimizing energy (2.7) using (2.6) as a soft constraint, with parameters $\sigma = 1$, $\gamma = 0.25$ and 8-grid N.

According to our model pixel $\tilde{\boldsymbol{p}}_i$ is a noisy measurement of point $\boldsymbol{p}$ on a contrast edge. Denoised point $\boldsymbol{p}_i$ is the projection of $\tilde{\boldsymbol{p}}_i$ onto l_i. To generate an edge mask (possibly at a higher resolution) we can quantize $\boldsymbol{p}_i$ and use q_i as values at quantized $\boldsymbol{p}_i$. If during this process we have a conflict such that several points are quantized into same pixel we choose the one with maximum probability. Figure 2.7(d) shows an edge mask whose resolution

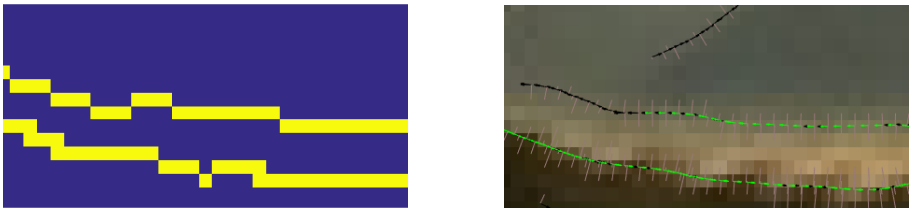


Figure 2.10: Comparison with Canny edge detector [39]. Note that Canny only produces the labeling of the pixels.

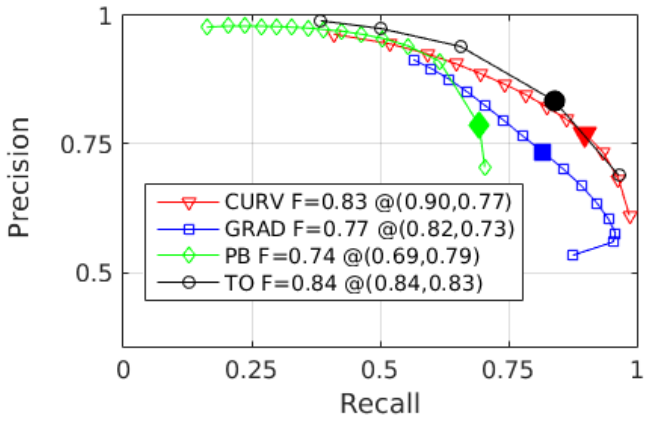


Figure 2.11: Comparison of out method (CURV) with the baseline gradients (GRAD), Pb [190] and the third order filter (TO) [274] on the database of [111]. Evaluation of Pb & TO is given by [111].

was doubled. Figure 2.9 shows examples of the edge mask at the original resolution.

We also compared our results with a few edge detection algorithms whose result is an edge mask, see Figure 2.11. This shows that our general method achieves F-measure of 0.83, which is very close to F-measure of 0.84, given by the best evaluated algorithm in [274]. Please note that [274] was designed specifically for edge detection in images, while our approach is a generic method for thin structure delineation.

2.4.2 Vessels in 3D

For the experiments in this section we used a *microscopic computer tomography* [109, 122] scan of the mouse's heart. The scan is a 3D volume of size 585x525x892. For the both experiments the volume was preprocessed with a popular vessel detection filter of [90]. For

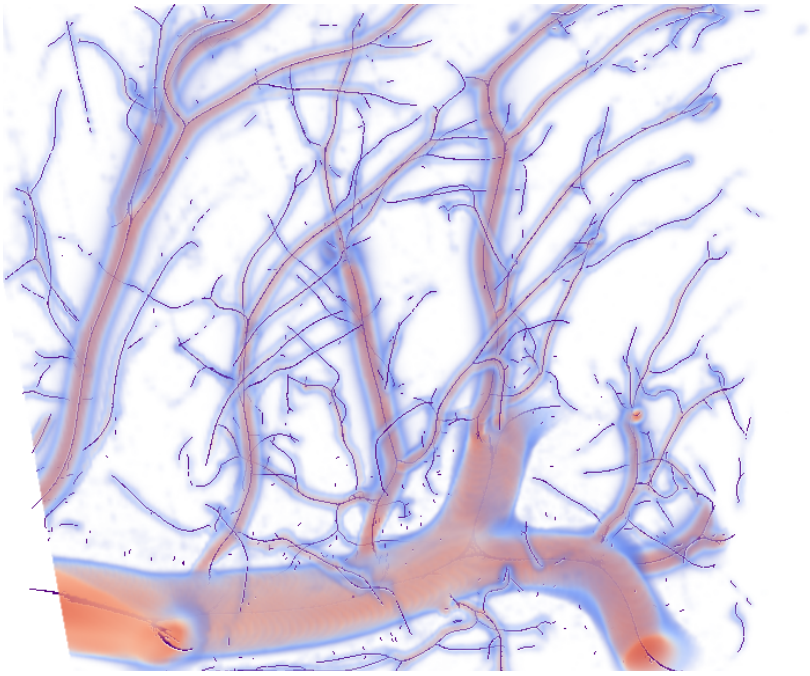


Figure 2.12: Example output of vessel center-line detection in 3D. Only tangents l_i with probabilities $q_i \geq \frac{1}{2}$ are shown (in purple). See the full-volume raw data in Figure 1.15.

every voxel $\tilde{p}_i$ the filter returns *vesselness measure* v_i such that higher values of v_i indicate higher likelihood of vessel presence at voxel $\tilde{p}_i$ with coordinates $\tilde{\boldsymbol{p}}_i \in \mathbb{R}^3$. The filter also estimates direction g_i and scale σ_i of a vessel.

For this application we use extension (2.8) of energy (2.5). Coefficient $\frac{1}{\sigma}$ in front of the soft constraint in the energy determines how far tangents l_i can move from voxels $\tilde{p}$. Since this data has high variability in vessel thickness, we cannot use the same σ for every voxel. We substitute σ_i produced by the vesselness filter for σ in energy (2.8):

$$E(L, X) = \sum_{(i,j)\in N} \kappa^2(l_i, l_j) x_i x_j + \tag{2.26}$$

$$+ \sum_i \left(\frac{1}{k^2 \sigma_i^2} \|l_i - \tilde{\boldsymbol{p}}_i\|^2 + \beta m(g_i, l_i) + \lambda_i \right) x_i \tag{2.27}$$

where k is a positive constant and λ_i is obtained from vesselness measure v_i by the same linear transformation that we use in Section 2.4.1. We set $\beta = 0.5$ and $k = 20$ and use 26-connected neighborhood system N.

For the first experiment we cropped the volume forming a subvolume of size 81x187x173. We also removed 85% of voxels with the lowest values of v_i. That yields about $3 \cdot 10^6$ variables to be optimized. Figure 2.12 shows the result.

The goal of the second experiment is to extract a few trees describing the cardiovascular system of the whole heart. To decrease the running time we perform Canny's [39] hysteresis thresholding to detect one-dimensional ridges in the volume. We substitute vesselness measure for intensity gradients in Canny's procedure. Then we set $q_i = 1$ for voxels detected as ridges and $q_i = 0$ for other voxels. This yields approximately the same number of optimization variables. Then we optimize tangents by the algorithm described in Section 2.3.3. Then the estimated center-line points are grouped based on the tangent and proximity information into a graph and a minimum spanning tree algorithm extracts the trees. The result is shown in Figure 2.13.

2.5 Discussion

We present a novel general early-vision framework for *simultaneous* detection and delineation of thin structures with sub-pixel localization and real-valued orientation estimation. The proposed energy combines likelihoods, indicator (detection) variables and squared or absolute curvature regularization. We present an algorithm that optimizes localization and orientation variables considering all possible configuration of indicator variables. We discuss the properties of the proposed energy and demonstrate a wide applicability of the framework on 2D and 3D examples.

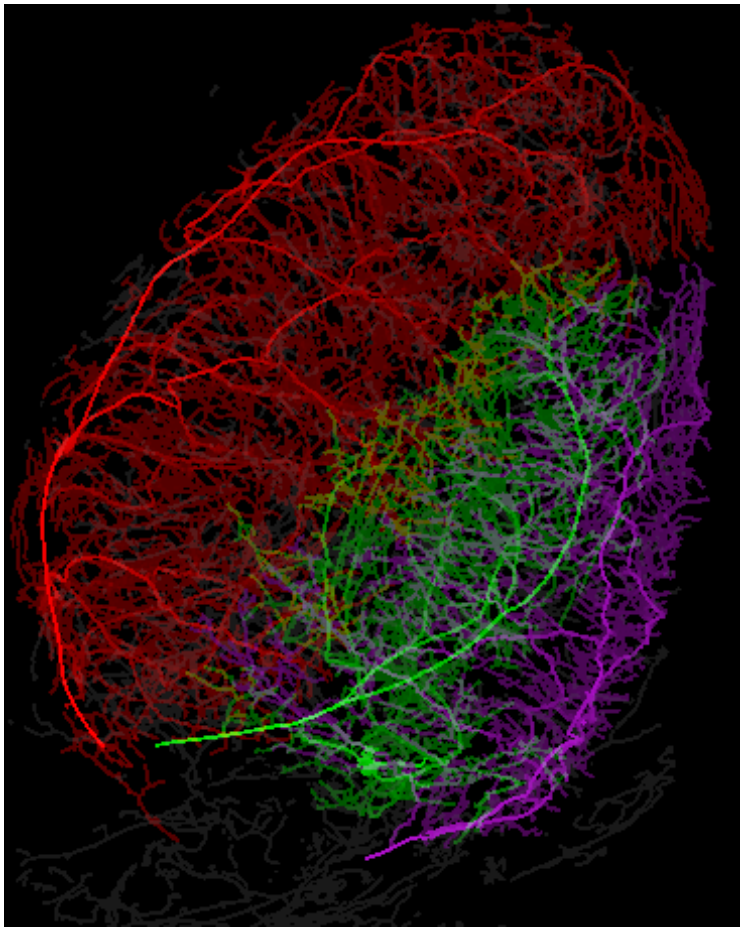


Figure 2.13: Center-line fitting for mouse heart vessels. The raw data is show in Figure 1.15. Three main branches are show in color. Other tangents are shown in dark gray.

Chapter 3

Kernel Clustering: Density Biases and Solutions

Clustering is widely used in data analysis where kernel methods are particularly popular due to their generality and discriminating power. In machine learning, *kernel clustering* is a well-established data analysis technique [247, 290, 253, 203, 320, 99, 76, 228, 56, 135] that can identify non-linearly separable structures, see Figure 3.1(a-b). Kernel methods are also popular in image segmentation [253, 316, 76, 318, 281].

However, kernel clustering has a practically significant bias to small dense clusters, *e.g.* empirically observed by Shi and Malik [253]. Its causes have never been analyzed and understood theoretically, even though many attempts were made to improve the results. We provide conditions and formally prove this bias in kernel clustering. Previously, Breiman [33] proved a bias to histogram mode isolation in discrete Gini criterion for decision tree learning. We found that kernel clustering reduces to a continuous generalization of Gini criterion for a common class of kernels where we prove a bias to density mode isolation and call it Breiman's bias. These theoretical findings suggest that a principled solution for the bias should directly address data density inhomogeneity. In particular, we show that density equalization can be implicitly achieved using either locally adaptive weights or a general class of Riemannian (geodesic) kernels. Our density equalization principle unifies many popular kernel clustering criteria including normalized cut, which we show has a bias to sparse subsets inversely related to Breiman's bias. Our synthetic and real data experiments illustrate these density biases and proposed solutions.

Section 3.1.1 reviews the kernel K-means and related clustering objectives, some of which have theoretically explained biases, see Section 1.3.2 and Section 3.1.2. In particu-

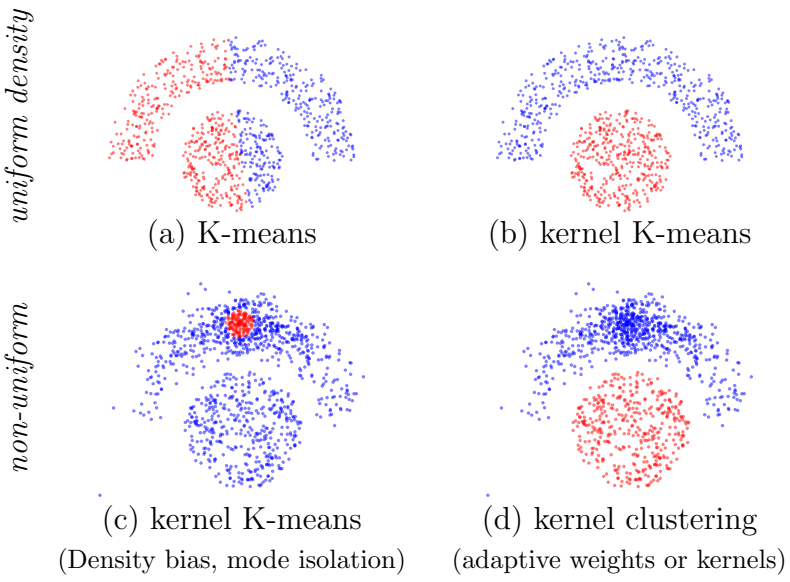


Figure 3.1: Kernel K-means with Gaussian kernel (3.1) gives desirable nonlinear separation for *uniform* density clusters (a,b). But, for *non-uniform* clusters in (c) it either isolates a small dense "clump" for smaller bandwidth σ due to the density bias or gives results like (a) for larger σ. No fixed σ yields solution (d) given by our locally adaptive kernels or weights eliminating the bias, see Chapter 3.

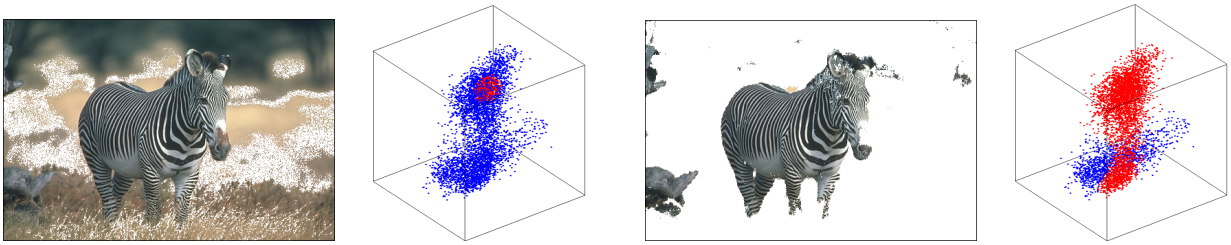
lar, Section 3.1.2 describes the discrete *Gini clustering criterion* standard in decision tree learning where Breiman [33] proved a bias to histogram mode isolation.

Empirically, it is well known that kernel K-means or *average association* (see Section 3.1.1) has a bias to so-called "tight" clusters for small bandwidths [253]. Figure 3.1(c) demonstrates this bias on a non-uniform modification of a typical toy example for kernel K-means with common Gaussian kernel

$$k(\boldsymbol{x},\boldsymbol{y}) \quad \propto \quad \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{y}\|^2}{2\sigma^2}\right). \tag{3.1}$$

This work shows in Section 3.2 that under certain conditions kernel K-means approximates the *continuous* generalization of the Gini criterion where we formally prove a mode isolation bias. A similar bias in the discrete case was analyzed by Breiman. Thus, we refer to the "tight" clusters in kernel K-means as *Breiman's bias*.

We propose a *density equalization* principle directly addressing the cause of Breiman's bias. First, Section 3.3 discusses modification of the density with adaptive point weights.



(a) Breiman's bias (b) good clustering

Figure 3.2: Example of Breiman's bias on real data. Feature vectors are 3-dimensional LAB colours corresponding to image pixels. Clustering results are shown in two ways. First, *red* and *blue* show different clusters inside LAB space. Second, pixels with colours in the "background" (red) cluster are removed from the original image. (a) shows the result for kernel K-means with a fixed-width Gaussian kernel isolating a small dense group of pixels from the rest. (b) shows the result for an adaptive kernel, see Section 3.4.

Then, Section 3.4 shows that a general class of locally adaptive *geodesic kernels* [135] implicitly transforms data and modifies its density. We derive "density laws" relating adaptive weights and kernels to density transformations. They allow to implement *density equalization* resolving Breiman's bias, see Figure 3.1(d). One popular heuristic [318] approximates a special case of our Riemannian kernels.

Besides mode isolation, kernel clustering may have the opposite density bias, *e.g. sparse subsets* in Normalized Cut [253], see Figure 3.9(a). Section 3.5 presents "normalization" as implicit *density inversion* establishing a formal relation between sparse subsets and Breiman's bias. Equalization addresses any density biases. Interestingly, density equalization makes many standard kernel clustering criteria conceptually equivalent, see Section 3.6.

3.1 Background and related work

3.1.1 Kernel K-means

A popular data clustering technique, *kernel K-means* [247] is a generalization of the basic *K-means* method. Assuming Ω denotes a finite set of points and $\boldsymbol{f}_p \in \mathbb{R}^N$ is a feature (vector) for point p, the basic K-means minimizes the sum of squared errors within clusters,

that is, distances from points $\boldsymbol{f}_p$ in each cluster $S_k \subset \Omega$ to the cluster centers $\boldsymbol{m}_k \in \mathbb{R}^N$

$$\left(\begin{array}{c} \textbf{k-means} \\ \textbf{criterion} \end{array} \right) \qquad \sum_k \sum_{p \in S^k} \|\boldsymbol{f}_p - \boldsymbol{m}_k\|^2. \tag{3.2}$$

Instead of clustering data points $\{\boldsymbol{f}_p \mid p \in \Omega\} \subset \mathbb{R}^N$ in their original space, kernel K-means uses mapping $\phi \;:\; \mathbb{R}^N \to \mathcal{H}$ embedding input data $\boldsymbol{f}_p \in \mathbb{R}^N$ as points $\boldsymbol{\phi}_p \equiv \phi(f_p)$ in a higher-dimensional Hilbert space $\mathcal{H}$. Kernel K-means minimizes the sum of squared errors in the embedding space corresponding to the following (mixed) objective function

$$F(S, m) \;=\; \sum_k \sum_{p \in S^k} \|\boldsymbol{\phi}_p - \boldsymbol{m}_k\|^2 \tag{3.3}$$

where $S = \{S^1, S^2, \ldots, S^K\}$ is a partitioning (clustering) of Ω into K clusters, $m = (\boldsymbol{m}_1, \boldsymbol{m}_2, \ldots \boldsymbol{m}_K)$ is a set of parameters for the clusters, and $\|.\|$ denotes the Hilbertian norm[1]. Kernel K-means finds clusters separated by hyperplanes in $\mathcal{H}$. In general, these hyperplanes correspond to non-linear surfaces in the original input space $\mathbb{R}^N$. Unlike (3.3), standard K-means objective (3.2) is able to identify only linearly separable clusters in $\mathbb{R}^N$.

Optimizing F with respect to the parameters yields closed-form solutions corresponding to the cluster means in the embedding space:

$$\hat{\boldsymbol{m}}_k = \frac{\sum_{q \in S^k} \boldsymbol{\phi}_q}{|S^k|} \tag{3.4}$$

where $|.|$ denotes the cardinality (number of points) in a cluster. Plugging optimal means (3.4) into objective (3.3) yields a high-order function, which depends solely on the partition variable S:

$$F(S) \;=\; \sum_k \sum_{p \in S^k} \left\| \boldsymbol{\phi}_p - \frac{\sum_{q \in S^k} \boldsymbol{\phi}_q}{|S^k|} \right\|^2. \tag{3.5}$$

Expanding the distances in (3.5), we obtain an equivalent pairwise clustering criterion expressed solely in terms of inner products $\langle \phi(\boldsymbol{f}_p), \phi(\boldsymbol{f}_q) \rangle$ in the embedding space $\mathcal{H}$:

$$F(S) \;\stackrel{c}{=}\; -\sum_k \frac{\sum_{pq \in S^k} \langle \phi(\boldsymbol{f}_p), \phi(\boldsymbol{f}_q) \rangle}{|S^k|} \tag{3.6}$$

[1]Our later examples use finite-dimensional embeddings ϕ where $\mathcal{H} = \mathbb{R}^M$ is an Euclidean space ($M \gg N$) and $\|.\|$ is the Euclidean norm.

where $\stackrel{c}{=}$ means equality up to an additive constant. The inner product is often replaced with kernel k, a symmetric function:

$$k(\boldsymbol{x}, \boldsymbol{y}) := \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{y}) \rangle. \tag{3.7}$$

Then, kernel K-means objective (3.5) can be presented as

$$\left(\begin{array}{c} \textbf{kernel} \\ \textbf{k-means} \\ \textbf{criterion} \end{array} \right) \qquad F(S) \stackrel{c}{=} -\sum_k \frac{\sum_{pq \in S^k} k(\boldsymbol{f}_p, \boldsymbol{f}_q)}{|S^k|}. \tag{3.8}$$

Formulation (3.8) enables optimization in high-dimensional space $\mathcal{H}$ that only uses kernel computation and does not require computing the embedding $\phi(\boldsymbol{x})$. Given a kernel function, one can use the kernel K-means without knowing the corresponding embedding. However, not any symmetric function corresponds to the inner product in some space. Mercer's theorem [290] states that any *positive semidefinite* (p.s.d.) kernel function $k(\boldsymbol{x}, \boldsymbol{y})$ can be expressed as an inner product in a higher-dimensional space. While p.s.d. is a common assumption for kernels, pairwise clustering objective (3.8) is often extended beyond p.s.d. affinities. There are many other extension of kernel K-means criterion (3.8). Despite the connection to density modes made in our work, kernel clustering has only a weak relation to *mean-shift* [60], *e.g.* see [281].

Related graph clustering criteria

Positive semidefinite kernel $k(\boldsymbol{f}_p, \boldsymbol{f}_q)$ in (3.8) can be replaced by an arbitrary pairwise similarity or affinity matrix $\boldsymbol{A} = [A_{pq}]$. This yields the *average association* criterion, which is known in the context of graph clustering [253, 239, 76]:

$$-\sum_k \frac{\sum_{pq \in S^k} A_{pq}}{|S^k|}. \tag{3.9}$$

The standard kernel K-means algorithm [76, 56] is not guaranteed to decrease (3.9) for improper (non p.s.d.) kernel $k(\boldsymbol{f}_p, \boldsymbol{f}_q) := A_{pq}$. However, [239] showed that dropping p.s.d. assumption is not essential: for arbitrary association A there is a p.s.d. kernel k such that objective (3.8) is equivalent to (3.9) up to a constant.

In [253] authors experimentally observed that the average association (3.9) or kernel K-means (3.8) objectives have a bias to separate small dense group of data points from the rest, *e.g.* see Figure 3.2.

Besides average association, there are other pairwise graph clustering criteria related to kernel K-means. *Normalized cut* is a common objective in the context of spectral clustering [253, 298]. It optimizes the following objective

$$-\sum_k \frac{\sum_{pq\in S^k} A_{pq}}{\sum_{p\in S^k} d_p}. \tag{3.10}$$

where $d_p = \sum_{q\in\Omega} A_{pq}$. Note that for $d_p = 1$ equation (3.10) reduces to (3.9). It is known that Normalized cut objective is equivalent to a weighted version of kernel K-means criterion [11, 76].

Probabilistic interpretation via kernel densities

Besides *kernel clustering*, kernels are also commonly used for *probability density estimation*. This section relates these two independent problems. Standard *multivariate kernel density estimate* or *Parzen density estimate* for the distribution of data points within cluster S^k can be expressed as follows [19]:

$$\mathcal{P}_{\boldsymbol{\Sigma}}(\boldsymbol{x}|S^k) \;:=\; \frac{\sum_{q\in S^k} k(\boldsymbol{x},\boldsymbol{f}_q)}{|S^k|}, \tag{3.11}$$

with kernel k having the form:

$$k(\boldsymbol{x},\boldsymbol{y}) \;=\; |\boldsymbol{\Sigma}|^{-\frac{1}{2}}\,\psi\left(\boldsymbol{\Sigma}^{-\frac{1}{2}}(\boldsymbol{x}-\boldsymbol{y})\right) \tag{3.12}$$

where ψ is a symmetric multivariate density and Σ is a symmetric positive definite *bandwidth* matrix controlling the density estimator's smoothness. One standard example is the Gaussian (normal) kernel (3.1) corresponding to

$$\psi(\boldsymbol{t}) \;\propto\; \exp\left(-\frac{\|\boldsymbol{t}\|^2}{2}\right), \tag{3.13}$$

which is common in kernel density estimation [19] and kernel clustering [99, 253].

The choice of bandwidth Σ is crucial for accurate density estimation, while the choice of ψ plays only a minor role [249]. There are numerous works regarding kernel selection for accurate density estimation using either fixed [257, 249, 133] or variable bandwidth [283]. For example, Scott's *rule of thumb* is

$$\sqrt{\boldsymbol{\Sigma}_{ii}} = \frac{r_i}{\sqrt[N+4]{n}}, \qquad \boldsymbol{\Sigma}_{ij} = 0 \text{ for } i \neq j \tag{3.14}$$

where n is the number of points, and r_i^2 is the variance of the i-th feature that could be interpreted as the range or scale of the data. Scott's rule gives optimal *mean integrated squared error* for normal data distribution, but in practice it works well in more general settings. In all cases the optimal bandwidth for sufficiently large datasets is a small fraction of the data range [80, 19]. For shortness, we use adjective *r-small* to describe bandwidths providing accurate density estimation.

If kernel k has form (3.12) up to a positive multiplicative constant then kernel K-means objective (3.8) can be expressed in terms of kernel densities (3.11) for points in each cluster [99]:

$$F(S) \stackrel{c}{=} -\sum_k \sum_{p \in S^k} \mathcal{P}_{\mathbf{\Sigma}}(\boldsymbol{f}_p | S^k). \tag{3.15}$$

3.1.2 Other clustering criteria and their known biases

One of the goals of this work is a theoretical explanation for the bias of kernel K-means with small bandwidths toward tight dense clusters, which we call *Breiman's bias*, see Figures 3.1-3.2. This bias was observed in the past only empirically. As discussed in Section 3.4.1, large bandwidth reduces kernel K-means to basic K-means where bias to equal cardinality clusters is known [139]. This section reviews other standard clustering objectives, entropy and Gini criteria, that have biases already well-understood theoretically. In Section 3.2 we establish a connection between Gini clustering and kernel K-means in case of *r-small* kernels. This connection allows theoretical analysis of Breiman's bias in kernel K-means.

Probabilistic K-means and entropy criterion

Besides non-parametric kernel K-means clustering there are well-known parametric extensions of basic K-means (3.2) based on probability models. *Probabilistic K-means* [139] or *model based clustering* [89] use some given likelihood functions $P(\boldsymbol{f}_p | \boldsymbol{m}_k)$ instead of distances $\|\boldsymbol{f}_p - \boldsymbol{m}_k\|^2$ in (3.2) as in clustering objective

$$-\sum_k \sum_{p \in S^k} \log P(\boldsymbol{f}_p | \theta_k). \tag{3.16}$$

Note that objective (3.16) reduces to basic K-means (3.2) for Gaussian probability model $P(.|\theta_k)$ with a fixed scalar covariance matrix and set θ_k containing a single element $\boldsymbol{m}_k$.

In probabilistic K-means (3.16) models can differ from Gaussians depending on *a priori* assumptions about the data in each cluster, *e.g.* gamma, Gibbs, or other distributions can

be used. For more complex data, each cluster can be described by highly-descriptive parametric models such as Gaussian mixtures (GMM). Instead of kernel density estimates in kernel K-means (3.15), probabilistic K-means (3.16) uses parametric distribution models. Another difference is the absence of the log in (3.15) compared to (3.16).

The analysis in [139] shows that in case of highly descriptive model P, *e.g.* GMM or histograms, (3.16) can be approximated by the standard *entropy criterion* for clustering:

$$\begin{pmatrix}\textbf{entropy}\\ \textbf{criterion}\end{pmatrix} \qquad \sum_k |S^k| \cdot H(S^k) \tag{3.17}$$

where $H(S^k)$ is the entropy of the distribution of the data in S^k:

$$H(S^k) \;:=\; -\int P(\boldsymbol{x}|\theta_k) \log P(\boldsymbol{x}|\theta_k)\, \mathrm{d}\boldsymbol{x}.$$

The discrete version of the entropy criterion is widely used for learning binary decision trees in classification [34, 19, 67]. It is known that the entropy criterion above is biased toward equal size clusters [33, 139, 30].

Discrete Gini impurity and criterion

Both Gini and entropy clustering criteria are widely used in the context of decision trees [19, 67]. These criteria are used to decide the best split at a given node of a binary classification tree [34]. The Gini criterion can be written for clustering $\{S^k\}$ as

$$\begin{pmatrix}\textbf{discrete}\\ \textbf{Gini}\\ \textbf{criterion}\end{pmatrix} \qquad \sum_k |S^k| \cdot G(S^k) \tag{3.18}$$

where $G(S^k)$ is the *Gini impurity* for the points in S^k. Assuming discrete feature space $\mathcal{L}$ instead of $\mathbb{R}^N$, the Gini impurity is

$$G(S^k) \;:=\; 1 - \sum_{l\in\mathcal{L}} \mathcal{P}(l\,|S^k)^2 \tag{3.19}$$

where $\mathcal{P}(l|S^k)$ is the empirical probability (histogram) of discrete-valued features $l \in \mathcal{L}$ in cluster S^k.

Similarly to the entropy, Gini impurity $G(S^k)$ can be viewed as a measure of sparsity or "peakedness" of the distribution for points in S^k. Note that (3.18) has a form similar to the entropy criterion in (3.17), except that entropy H is replaced by the Gini impurity. Breiman [33] analyzed the theoretical properties of the discrete Gini criterion (3.18) when $\mathcal{P}(\cdot\,|S^k)$ are *discrete histograms*. He proved

Theorem 3.1 (Breiman)**.** *For $K = 2$ the minimum of the Gini criterion (3.18) for discrete Gini impurity (3.19) is achieved by assigning all data points with the highest-probability feature value in $\mathcal{L}$ to one cluster and the remaining data points to the other cluster.* □

3.2 Breiman's bias (numerical features)

In this section we show that the kernel K-means objective reduces to a novel *continuous* Gini criterion under some general conditions on the kernel function, see Section 3.2.1. We formally prove in Section 3.2.2 that the optimum of the continuous Gini criterion isolates the data density mode. That is, we show that the discussed earlier biases observed in the context of clustering [253] and decision tree learning [33] are the same phenomena. Section 3.2.3 establishes connection to maximum cliques [202] and *dominant sets* [228].

For further analysis we reformulate the problem of clustering a discrete set of points $\{\boldsymbol{f}_p \,|\, p \in \Omega\} \subset \mathbb{R}^N$, see Section 3.1.1, as a continuous domain clustering problem. Let P be a probability measure over domain $\mathbb{R}^N$ and ρ be the corresponding continuous probability density function such that the discrete points $\boldsymbol{f}_p$ could be treated as samples from this distribution. The clustering of the continuous domain will be described by an *assignment function* $s : \mathbb{R}^N \to \{1, 2, \ldots, K\}$. Density ρ implies conditional probability densities $\rho_k^s(\boldsymbol{x}) := \rho(\boldsymbol{x} \,|\, s(\boldsymbol{x}) = k)$. Feature points $\boldsymbol{f}_p$ in cluster S^k could be interpreted as a sample from conditional density ρ_k^s.

Then, the continuous clustering problem is to find an assignment function optimizing a clustering criteria. For example, we can analogously to (3.18) define continuous Gini clustering criterion

$$\left(\begin{array}{c}\textbf{continuous}\\ \textbf{Gini}\\ \textbf{criterion}\end{array}\right) \qquad \sum_k w_k \cdot G(s,k), \tag{3.20}$$

where w_k is the probability to draw a point from k-th cluster and

$$G(s,k) \;:=\; 1 - \int \rho_k^s(\boldsymbol{x})^2 \,\mathrm{d}\boldsymbol{x}. \tag{3.21}$$

In the next section we show that kernel K-means energy (3.15) can be approximated by continuous Gini-clustering criterion (3.20) for *r-small* kernels.

3.2.1 Kernel K-means and continuous Gini criterion

To establish the connection between kernel clustering and the Gini criterion, let us first recall Monte-Carlo estimation [139], which yields the following expectation-based approximation for a continuous function $g(\boldsymbol{x})$ and cluster $C \subset \Omega$:

$$\sum_{p \in C} g(\boldsymbol{f}_p) \approx |C| \int g(\boldsymbol{x})\, \rho_C(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x} \tag{3.22}$$

where ρ_C is the "true" continuous density of features in cluster C. Using (3.22) for $C = S^k$ and $g(x) = \mathcal{P}_{\boldsymbol{\Sigma}}(x|S^k)$, we approximate the kernel density (3.15) by its expectation

$$F(S) \overset{c}{\approx} -\sum_k |S^k| \int \mathcal{P}_{\boldsymbol{\Sigma}}(\boldsymbol{x}|S^k)\, \rho_k^s(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x}. \tag{3.23}$$

Note that partition $S = (S^1, \ldots, S^K)$ is determined by dataset Ω and assignment function s. We also assume

$$\mathcal{P}_{\boldsymbol{\Sigma}}(\cdot\,|S^k) \approx \rho_k^s(\cdot). \tag{3.24}$$

This is essentially an assumption on kernel bandwidth. That is, we assume that kernel bandwidth gives accurate density estimation. For shortness, we call such bandwidths *r-small*, see Section 3.1.1. Then (3.23) reduces to approximation

$$F(S) \overset{c}{\approx} -\sum_k |S^k| \cdot \int \rho_k^s(\boldsymbol{x})^2\, \mathrm{d}x \overset{c}{\equiv} \sum_k |S^k| \cdot G(s,k). \tag{3.25}$$

Additional application of Monte-Carlo estimation $|S^k|/|\Omega| \approx w_k$ allows replacing set cardinality $|S^k|$ by probability w_k of drawing a point from S^k. This results in continuous Gini clustering criterion (3.20), which approximates (3.15) or (3.8) up to an additive and positive multiplicative constants.

Next section proves that the continuous Gini criterion (3.20) has a similar bias observed by Breiman in the discrete case.

3.2.2 Breiman's bias in continuous Gini criterion

This section extends Theorem 3.1 to continuous Gini criterion (3.20). Since Section 3.2.1 has already established a close relation between continuous Gini criterion and kernel K-means for *r-small* bandwidth kernels, then Breiman's bias also applies to the latter. For simplicity, we focus on $K = 2$ as in Breiman's Theorem 3.1.

Theorem 3.2 (Breiman's bias in continuous case)**.** *For $K = 2$ the continuous Gini clustering criterion* (3.20) *achieves its optimal value at the partitioning of $\mathbb{R}^N$ into regions*

$$s_1 = \arg\max_x \rho(x) \quad \textit{and} \quad s_2 = \mathbb{R}^N \setminus s_1.$$

Proof. The statement follows from Lemma 3.2 below. □

We denote mathematical expectation of function $z : \Omega \to \mathbb{R}^1$

$$\mathbb{E}z := \int z(\boldsymbol{x})\rho(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}.$$

Minimization of (3.20) corresponds to maximization of the following objective function

$$L(s) := w \int \rho_1^s(\boldsymbol{x})^2\,\mathrm{d}\boldsymbol{x} + (1-w)\int \rho_2^s(\boldsymbol{x})^2\,\mathrm{d}\boldsymbol{x} \tag{3.26}$$

where the probability to draw a point from cluster 1 is

$$w := w_1 = \int_{s(\boldsymbol{x})=1} \rho(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \mathbb{E}[s(\boldsymbol{x}) = 1]$$

where $[\cdot]$ is the indicator function. Note that *mixed joint density*

$$\rho(\boldsymbol{x}, k) = \rho(\boldsymbol{x}) \cdot [s(\boldsymbol{x}) = k]$$

allows to write conditional density ρ_1^s in (3.26) as

$$\rho_1^s(\boldsymbol{x}) = \frac{\rho(\boldsymbol{x}, 1)}{P(s(\boldsymbol{x}) = 1)} = \rho(\boldsymbol{x}) \cdot \frac{[s(\boldsymbol{x}) = 1]}{w}. \tag{3.27}$$

Equations (3.26) and (3.27) give

$$\begin{aligned} L(s) &= \frac{1}{w}\int \rho(\boldsymbol{x})^2[s(\boldsymbol{x}) = 1]\,\mathrm{d}\boldsymbol{x} \\ &\quad + \frac{1}{1-w}\int \rho(\boldsymbol{x})^2[s(\boldsymbol{x}) = 2]\,\mathrm{d}\boldsymbol{x}. \end{aligned} \tag{3.28}$$

Introducing notation

$$I := [s(\boldsymbol{x}) = 1] \quad \text{and} \quad \rho := \rho(\boldsymbol{x})$$

allows to further simplify the objective function as

$$L(s) \;=\; \frac{\mathbb{E}I\rho}{\mathbb{E}I} \;+\; \frac{\mathbb{E}(1-I)\rho}{1-\mathbb{E}I}. \tag{3.29}$$

Without loss of generality assume that $\frac{\mathbb{E}(1-I)\rho}{1-\mathbb{E}I} \le \frac{\mathbb{E}I\rho}{\mathbb{E}I}$ (the opposite case would yield a similar result). We now need following

Lemma 3.1. *Let a, b, c, d be some positive numbers, then*

$$\frac{a}{b} \le \frac{c}{d} \implies \frac{a}{b} \le \frac{a+c}{b+d} \le \frac{c}{d}.$$

Proof. Use reduction to a common denominator. □

Lemma 3.1 implies inequality

$$\frac{\mathbb{E}(1-I)\rho}{1-\mathbb{E}I} \;\le\; \mathbb{E}\rho \;\le\; \frac{\mathbb{E}I\rho}{\mathbb{E}I}, \tag{3.30}$$

which is needed to prove the lemma below.

Lemma 3.2. *Assume that function s_ε is*

$$s_\varepsilon(\boldsymbol{x}) \;:=\; \begin{cases} 1, & \rho(\boldsymbol{x}) \ge \sup_{\boldsymbol{x}} \rho(\boldsymbol{x}) - \varepsilon, \\ 2, & \textit{otherwise.} \end{cases} \tag{3.31}$$

Then

$$\sup_s L(s) = \lim_{\varepsilon \to 0} L(s_\varepsilon) = \mathbb{E}\rho + \sup_{\boldsymbol{x}} \rho(\boldsymbol{x}). \tag{3.32}$$

Proof. Due to monotonicity of expectation we have

$$\frac{\mathbb{E}I\rho}{\mathbb{E}I} \le \frac{\mathbb{E}\left(I \sup_x \rho(x)\right)}{\mathbb{E}I} = \sup_x \rho(x). \tag{3.33}$$

Then (3.30) and (3.33) imply

$$L(s) = \frac{\mathbb{E}I\rho}{\mathbb{E}I} + \frac{\mathbb{E}(1-I)\rho}{1-\mathbb{E}I} \le \sup_{\boldsymbol{x}} \rho(\boldsymbol{x}) + \mathbb{E}\rho. \tag{3.34}$$

That is, the right part of (3.32) is an upper bound for $L(s)$.

Let $I_\varepsilon \equiv [s_\varepsilon(x) = 1]$. It is easy to check that

$$\lim_{\varepsilon\to 0} \frac{\mathbb{E}(1 - I_\varepsilon)\rho}{1 - \mathbb{E}I_\varepsilon} = \mathbb{E}\rho. \tag{3.35}$$

Definition (3.31) also implies

$$\lim_{\varepsilon\to 0} \frac{\mathbb{E}I_\varepsilon\rho}{\mathbb{E}I_\varepsilon} \geq \lim_{\varepsilon\to 0} \frac{\mathbb{E}(\sup_{\boldsymbol{x}} \rho(\boldsymbol{x}) - \varepsilon)I_\varepsilon}{\mathbb{E}I_\varepsilon} = \sup_{\boldsymbol{x}} \rho(\boldsymbol{x}). \tag{3.36}$$

This result and (3.33) conclude that

$$\lim_{\varepsilon\to 0} \frac{\mathbb{E}I_\varepsilon\rho}{\mathbb{E}I_\varepsilon} = \sup_{\boldsymbol{x}} \rho(\boldsymbol{x}). \tag{3.37}$$

Finally, the limits in (3.35) and (3.37) imply

$$\begin{aligned} \lim_{\varepsilon\to 0} L(s_\varepsilon) &= \lim_{\varepsilon\to 0} \frac{\mathbb{E}(1 - I_\varepsilon)\rho}{1 - \mathbb{E}I_\varepsilon} + \lim_{\varepsilon\to 0} \frac{\mathbb{E}I_\varepsilon\rho}{\mathbb{E}I_\varepsilon} \\ &= \mathbb{E}\rho + \sup_{\boldsymbol{x}} \rho(\boldsymbol{x}). \end{aligned} \tag{3.38}$$

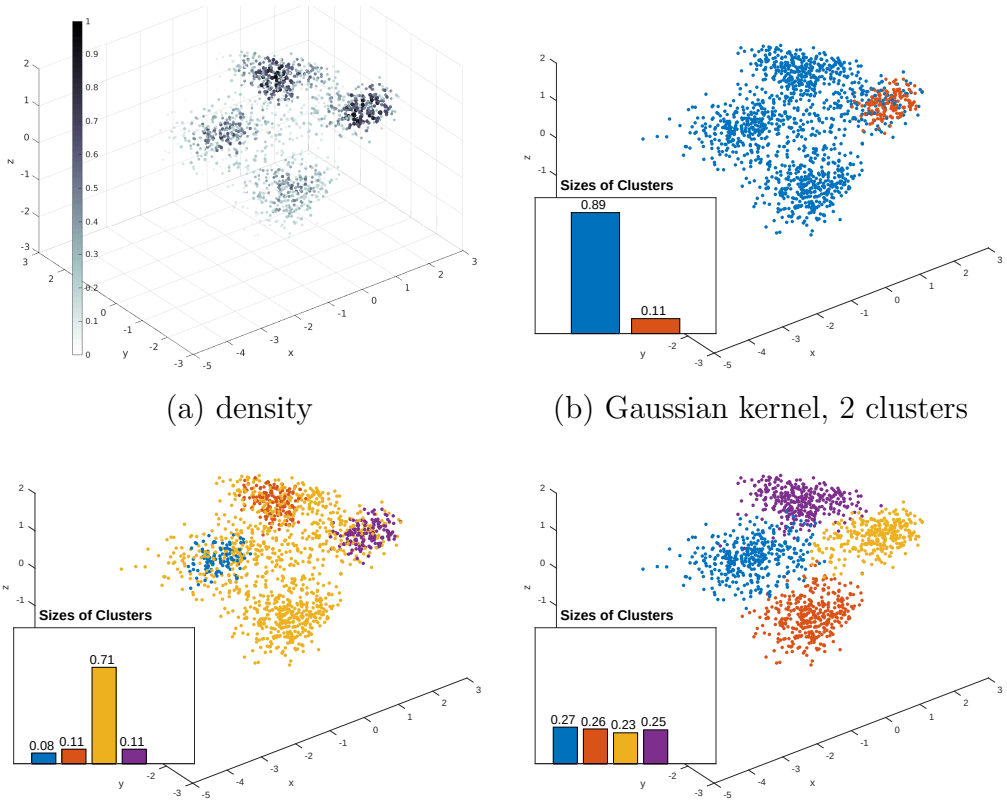
This equality and bound (3.34) prove (3.32). □

This result states that the optimal assignment function separates the mode of the density function from the rest of the data. The proof considers case $K = 2$ for continuous Gini criterion approximating kernel K-means for *r-small* kernels. The multi-cluster version for $K > 2$ also has Breiman's bias. Indeed, it is easy to show that any two clusters in the optimal solution shall give optimum of objective (3.20). Then, these two clusters are also subject to Breiman's bias. See a multi-cluster example in Figure 3.3.

Practical considerations: While Theorem 3.2 suggests that the isolated density mode should be a single point, in practice Breiman's bias in kernel k-means isolates a slightly wider cluster around the mode, see Figures 3.2, 3.3, 3.7(a-d), 3.8. Indeed, Breiman's bias holds for kernel k-means when the assumptions in Section 3.2.1 are valid. In practice, shrinking of the clusters invalidates approximations (3.23) and (3.24) preventing the collapse of the clusters.

3.2.3 Connection to maximal cliques and dominant sets

Interestingly, there is also a relation between *maximum cliques* and *density modes*. Assume 0-1 kernel $[\|x - y\| \leq \sigma]$ with bandwidth σ. Then, kernel matrix A is a connectivity matrix



(a) density (b) Gaussian kernel, 2 clusters

(c) Gaussian kernel, 4 clusters (d) *KNN* kernel, 4 clusters

Figure 3.3: Breiman's bias in clustering of images. We select 4 categories from the LabelMe dataset [210]. The last fully connected layer of the neural network in [154] gives 4096-dimensional feature vector for each image. We reduce the dimension to 5 via PCA. For visualization purposes, we obtain 3D embeddings via MDS [65]. (a) Kernel densities estimates for data points are color-coded: darker points correspond to higher density. (b,c) The result of the kernel K-means with the Gaussian kernel (3.1). Scott's rule of thumb defines the bandwidth. Breiman's bias causes poor clustering, *i.e.* small cluster is formed in the densest part of the data in (b), three clusters occupy few points within densest regions while the fourth cluster contains 71% of the data in (c). The *normalized mutual information* (NMI) in (c) is 0.38. (d) Good clustering produced by *KNN* kernel u_p (Example 3.3) gives NMI of 0.90, which is slightly better than the basic K-means (0.89).

corresponding to a *σ-disk graph*. Intuitively, the maximum clique on this graph should be inside a disk with the largest number of points in it, which corresponds to the density mode.

Formally, mode isolation bias can be linked to both maximum clique and its weighted-graph generalization, *dominant set* [228]. It is known that maximum clique [202] and *dominant set* [228] solve a two-region clustering problem with energy

$$-\frac{\sum_{pq\in S^1} A_{pq}}{|S^1|} \tag{3.39}$$

corresponding to average association (3.9) for $K = 1$ and $S^1 \subseteq \Omega$. Under the same assumptions as above, Gini impurity (3.21) can be used as an approximation reducing objective (3.39) to

$$\frac{\mathbb{E}I\rho}{\mathbb{E}I}. \tag{3.40}$$

Using (3.33) and (3.37) we can conclude that the optimum of (3.40) isolates the mode of density function ρ. Thus, clustering minimizing (3.39) for *r-small* bandwidths also has Breiman's bias. That is, for such bandwidths the concepts of maximum clique and dominant set for graphs correspond to the concept of *mode isolation* for data densities. Dominant sets for the examples in Figures 3.1(c), 3.2(a), and 3.7(d) would be similar to the shown mode-isolating solutions.

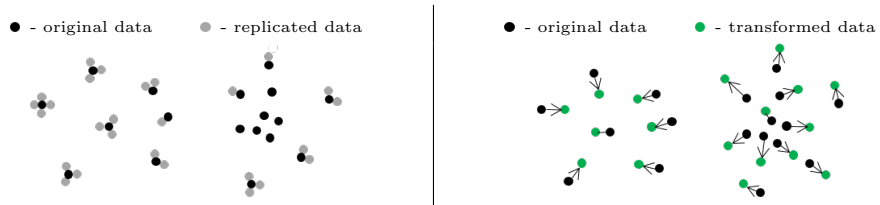
3.3 Adaptive weights solving Breiman's bias

We can use a simple modification of average association by introducing weights $w_p \geq 0$ for each point "error" within the equivalent kernel K-means objective (3.3)

$$F_w(S, \boldsymbol{m}) \;=\; \sum_k \sum_{p\in S^k} w_p \|\boldsymbol{\phi}_p - \boldsymbol{m}_k\|^2. \tag{3.41}$$

Such weighting is common for K-means [80]. Similarly to Section 3.1.1 we can expand the Euclidean distances in (3.41) to obtain an equivalent *weighted average association* criterion generalizing (3.9)

$$-\sum_k \frac{\sum_{pq\in S_k} w_p w_q A_{pq}}{\sum_{p\in S_k} w_p}. \tag{3.42}$$



(a) adaptive weights (Section 3.3) | (b) adaptive kernels (Section 3.4.3)

Figure 3.4: *Density equalization* via (a) adaptive weights and (b) adaptive kernels. In (a) the density is modified as in (3.43) via "replicating" each data point inverse-proportionately to the observed density using $w_p \propto 1/\rho_p$. For simplicity (a) assumes positive integer weights w_p. In (b) the density is modified according to (3.58) for bandwidth (3.61) via implicit embedding of data points in a higher dimensional space that changes their relative positions.

Weights w_p have an obvious interpretation based on (3.41); they change the data by replicating each point p by a number of points in the same location (Figure 3.4a) in proportion to w_p. Therefore, this weighted formulation directly modifies the data density as

$$\rho'_p \propto w_p \rho_p \tag{3.43}$$

where ρ_p and ρ'_p are respectively the densities of the original and the new (replicated) points. The choice of $w_p = 1/\rho_p$ is a simple way for equalizing data density to solve Breiman's bias. As shown in Figure 3.4(a), such a choice enables low-density points to be replicated more frequently than high-density ones. This is one of density equalization approaches giving the solution in Figure 3.1(d).

3.4 Adaptive kernels solving Breiman's bias

Breiman's bias in kernel K-means is specific to *r-small* bandwidths. Thus, it has direct implications for the bandwidth selection problem discussed in this section. Note that kernel bandwidth selection for *clustering* should not be confused with kernel bandwidth selection for *density estimation*, an entirely different problem outlined in Section 3.1.1. In fact, *r-small* bandwidths give accurate density estimation, but yield poor clustering due to Breiman's bias. Larger bandwidths can avoid this bias in clustering. However, Section 3.4.1 shows that for extremely large bandwidths kernel K-means reduces to standard K-means, which loses ability of non-linear cluster separation and has a different bias to equal cardinality clusters [139, 30].

In practice, avoiding extreme bandwidths is problematic since the notions of *small* and *large* strongly depend on data properties that may significantly vary across the domain, *e.g.* in Figure 3.1(c,d) where no fixed bandwidth gives a reasonable separation. This motivates *locally* adaptive strategies. Interestingly, Section 3.4.2 shows that any locally adaptive bandwidth strategy implicitly corresponds to some data embedding $\Omega \to \mathbb{R}^{N'}$ deforming density of the points. That is, locally adaptive selection of bandwidth is equivalent to selection of density transformation. Local kernel bandwidth and transformed density are related via the *density law* established in (3.59). As we already know from Theorem 3.2, Breiman's bias is caused by high non-uniformity of the data, which can be addressed by density equalizing transformations. Section 3.4.3 proposes adaptive kernel strategies based on our *density law* and motivated by a *density equalization* principle addressing Breiman's bias. In fact, a popular locally adaptive kernel in [318] is a special case of our density equalization principle.

3.4.1 Overview of extreme bandwidth cases

Section 3.2.1 and Theorem 3.2 prove that for *r-small* bandwidths the kernel K-means is biased toward "tight" clusters, as illustrated in Figures 3.1, 3.2 and 3.7(d). As bandwidth increases, continuous kernel density (3.11) no longer approximates the true distribution ρ_k^s violating (3.24). Thus, Gini criterion (3.25) is no longer valid as an approximation for kernel K-means objective (3.15). In practice, Breiman's bias disappears gradually as bandwidth gets larger. This is also consistent with experimental comparison of smaller and larger bandwidths in [253].

The other extreme case of bandwidth for kernel K-means comes from its reduction to basic K-means for large kernels. For simplicity, assume Gaussian kernels (3.1) of large bandwidth σ approaching data diameter. Then the kernel can be approximated by its Taylor expansion $\exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{y}\|^2}{2\sigma^2}\right) \approx 1 - \frac{\|\boldsymbol{x}-\boldsymbol{y}\|^2}{2\sigma^2}$ and kernel K-means objective (3.8) for $\sigma \gg \|\boldsymbol{x}-\boldsymbol{y}\|$ becomes[2] (up to a constant)

$$\sum_k \frac{\sum_{pq\in S^k} \|\boldsymbol{f}_p - \boldsymbol{f}_q\|^2}{2\sigma^2|S^k|} \overset{c}{=} \frac{1}{\sigma^2} \sum_k \sum_{p\in S^k} \|\boldsymbol{f}_p - \boldsymbol{m}_k\|^2, \tag{3.44}$$

which is equivalent to basic K-means (3.2) for any fixed σ.

Figure 3.5 summarizes kernel K-means biases for different bandwidths. For large bandwidths the kernel K-means loses its ability to find non-linear cluster separation due to

[2]Relation (3.44) easily follows by substituting $\boldsymbol{m}_k \equiv \frac{1}{|S^k|}\sum_{p\in S^k} \boldsymbol{f}_p$.

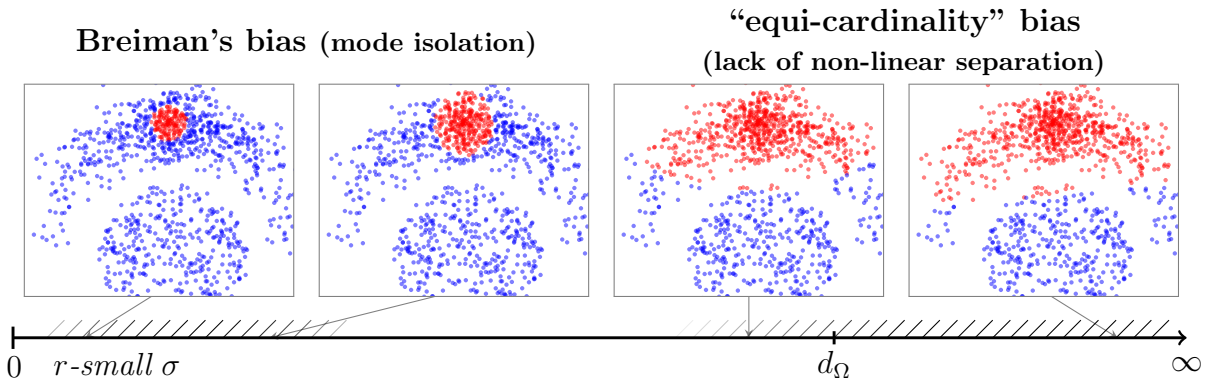


Figure 3.5: Kernel K-means biases over the range of bandwidth σ. Data diameter is denoted by $d_\Omega = \max_{pq\in\Omega} \|\boldsymbol{f}_p - \boldsymbol{f}_q\|$. Breiman's bias is established for *r-small* σ in Section 3.1.1. Points stop interacting for σ smaller than *r-small* making kernel K-means fail. Larger σ reduce kernel K-means to the basic K-means removing an ability to separate the clusters non-linearly. In practice, there could be no intermediate good σ. In the example of Figure 3.1(c), any fixed σ leads to either Breiman's bias or to the lack of non-linear separability.

reduction to the basic K-means. Moreover, it inherits the bias to equal cardinality clusters, which is well-known for the basic K-means [139, 30]. On the other hand, for small bandwidths kernel K-means has Breiman's bias proven in Section 3.2. To avoid the biases in Figure 3.5, kernel K-means should use a bandwidth neither too small nor too large. This motivates locally adaptive bandwidths.

3.4.2 Adaptive kernels as density transformation

This section shows that kernel clustering (3.8) with any *locally adaptive bandwidth* strategy satisfying some reasonable assumptions is equivalent to *fixed bandwidth* kernel clustering in a new feature space (Theorem 3.3) with a deformed point density. The adaptive bandwidths relate to density transformations via *density law* (3.59). To derive it, we interpret *adaptiveness* as non-uniform variation of distances across the feature space. In particular, we use a general concept of *geodesic kernel* defining adaptiveness via a metric tensor and illustrate it by simple practical examples.

Our analysis of Breiman's bias in Section 3.2 applies to general kernels (3.12) suitable for density estimation. Here we focus on clustering with kernels based on *radial basis*

functions ψ s.t.

$$\psi(\boldsymbol{x}-\boldsymbol{y}) = \psi(\|\boldsymbol{x}-\boldsymbol{y}\|). \tag{3.45}$$

To obtain adaptive kernels, we replace Euclidean metric with Riemannian inside (3.45). In particular, $\|\boldsymbol{x}-\boldsymbol{y}\|$ is replaced with *geodesic distances* $d_g(\boldsymbol{x},\boldsymbol{y})$ between features $\boldsymbol{x},\boldsymbol{y} \in \mathbb{R}^N$ based on any given metric tensor $g(\boldsymbol{f})$ for $\boldsymbol{f} \in \mathbb{R}^N$. This allows to define a *geodesic* or *Riemannian* kernel at any points $\boldsymbol{f}_p$ and $\boldsymbol{f}_q$ as in [135]

$$k_g(\boldsymbol{f}_p,\boldsymbol{f}_q) \;:=\; \psi(d_g(\boldsymbol{f}_p,\boldsymbol{f}_q)) \;\equiv\; \psi(d_{pq}) \tag{3.46}$$

where $d_{pq} := d_g(\boldsymbol{f}_p,\boldsymbol{f}_q)$ is introduced for shortness.

In practice, the metric tensor can be defined only at the data points $\boldsymbol{g}_p := g(\boldsymbol{f}_p)$ for $p \in \Omega$. Often, quickly decaying radial basis functions ψ allow Mahalanobis distance approximation inside (3.46)

$$d_g(\boldsymbol{f}_p,\boldsymbol{x})^2 \;\approx\; (\boldsymbol{f}_p-\boldsymbol{x})^\top \boldsymbol{g}_p\,(\boldsymbol{f}_p-\boldsymbol{x}), \tag{3.47}$$

which is normally valid only in a small neighborhood of $\boldsymbol{f}_p$. If necessary, one can use more accurate approximations for $d_g(\boldsymbol{f}_p,\boldsymbol{f}_q)$ based on Dijkstra [62] or Fast Marching [251].

Example 3.1 (***Adaptive non-normalized***[3] ***Gaussian kernel***)**.** Mahalanobis distances based on (adaptive) bandwidth matrices $\boldsymbol{\Sigma}_p$ defined at each point p can be used to define adaptive kernel

$$\kappa_p(\boldsymbol{f}_p,\boldsymbol{f}_q) \;:=\; \exp -\frac{1}{2}(\boldsymbol{f}_p-\boldsymbol{f}_q)^\top \boldsymbol{\Sigma}_p^{-1}(\boldsymbol{f}_p-\boldsymbol{f}_q), \tag{3.48}$$

which equals fixed bandwidth Gaussian kernel (3.1) for $\boldsymbol{\Sigma}_p = \sigma^2\boldsymbol{I}$. Kernel (3.48) approximates (3.46) for exponential function ψ in (3.13) and tensor g continuously extending matrices $\boldsymbol{\Sigma}_p^{-1}$ over the whole feature space so that $\boldsymbol{g}_p = \boldsymbol{\Sigma}_p^{-1}$ for $p \in \Omega$. Indeed, assuming matrices $\boldsymbol{\Sigma}_p^{-1}$ and tensor g change slowly between points within bandwidth neighbourhoods, one can use (3.47) for all points in

$$\kappa_p(\boldsymbol{f}_p,\boldsymbol{f}_q) \;\approx\; \exp\frac{-d_g(\boldsymbol{f}_p,\boldsymbol{f}_q)^2}{2} \;\equiv\; \exp\frac{-d_{pq}^2}{2} \tag{3.49}$$

due to exponential decay outside the bandwidth neighbourhoods.

[3] Lack of normalization as in (3.48) is critical for *density equalization* resolving Breiman's bias, which is our only goal for adaptive kernels. Note that without kernel normalization as in (3.12) Parzen density formulation of kernel k-means (3.15) no longer holds invalidating the relation to Gini and Breiman's bias in Section 3.2. On the contrary, *normalized* variable kernels are appropriate for *density estimation* [283] validating (3.15). They can also make approximation (3.24) more accurate strengthening connections to Gini and Breiman's bias.

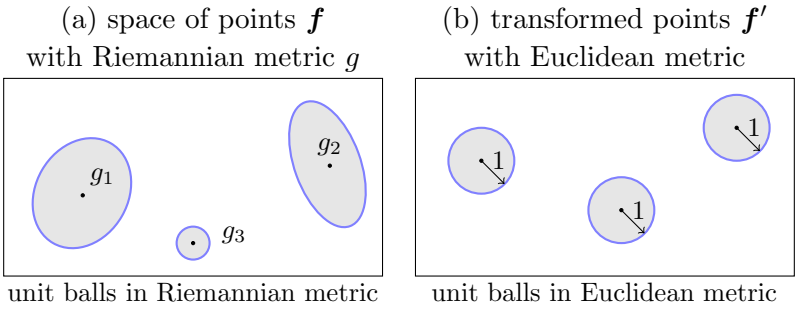


Figure 3.6: Adaptive kernel (3.46) based on Riemannian distances (a) is equivalent to fixed bandwidth kernel after some *quasi-isometric* (3.50) embedding into Euclidean space (b), see Theorem 3.3, mapping ellipsoids (3.52) to balls (3.54) and modifying data density as in (3.57).

Example 3.2 (***Zelnik-Manor & Perona kernel [318]***)**.** This popular kernel is defined as $\kappa_{pq} := \exp \frac{-\|\boldsymbol{f}_p - \boldsymbol{f}_q\|^2}{2\sigma_p\sigma_q}$. This kernel's relation to (3.46) is less intuitive due to the lack of "local" Riemannian tensor. However, under assumptions similar to those in (3.49), it can still be seen as an approximation of geodesic kernel (3.46) for some tensor g such that $\boldsymbol{g}_p = \sigma_p^{-2}\boldsymbol{I}$ for $p \in \Omega$. They use heuristic $\sigma_p = R_p^K$, which is the distance to the K-th nearest neighbour of $\boldsymbol{f}_p$.

Example 3.3 (***KNN kernel***)**.** This adaptive kernel is defined as $u_p(\boldsymbol{f}_p, \boldsymbol{f}_q) = [\boldsymbol{f}_q \in KNN(\boldsymbol{f}_p)]$ where $KNN(\boldsymbol{f}_p)$ is the set of K nearest neighbors of $\boldsymbol{f}_p$. This kernel approximates (3.46) for uniform function $\psi(t) = [t < 1]$ and tensor g such that $\boldsymbol{g}_p = \boldsymbol{I}/(R_p^K)^2$.

Theorem 3.3. *Clustering* (3.8) *with (adaptive) geodesic kernel* (3.46) *is equivalent to clustering with fixed bandwidth kernel* $k'(\boldsymbol{f}'_p, \boldsymbol{f}'_q) := \psi'(\|\boldsymbol{f}'_p - \boldsymbol{f}'_q\|)$ *in new feature space* $\mathbb{R}^{N'}$ *for some radial basis function* ψ' *using the Euclidean distance and some integer* N'*.*

Proof. A powerful general result in [174, 107, 239] states that for any symmetric matrix (d_{pq}) with zeros on the diagonal there is a constant h such that squared distances

$$\widetilde{d}_{pq}^2 \quad = \quad d_{pq}^2 + h^2[p \neq q] \tag{3.50}$$

form an *Euclidean matrix* $(\widetilde{d}_{pq})$. That is, there exists some Euclidean embedding $\Omega \to \mathbb{R}^{N'}$ where for $\forall p \in \Omega$ there corresponds a point $\boldsymbol{f}'_p \in \mathbb{R}^{N'}$ such that $\|\boldsymbol{f}'_p - \boldsymbol{f}'_q\| = \widetilde{d}_{pq}$, see Figure 3.6. Therefore,

$$\psi(d_{pq}) \quad = \quad \psi\left(\sqrt{\widetilde{d}_{pq}^2 - h^2\,[d_{pq} \geq h]}\right) \quad \equiv \quad \psi'(\widetilde{d}_{pq}) \tag{3.51}$$

for $\psi'(t) := \psi(\sqrt{t^2 - h^2[t \geq h]})$ and $k_g(\boldsymbol{f}_p, \boldsymbol{f}_q) = k'(\boldsymbol{f}'_p, \boldsymbol{f}'_q)$. □

Theorem 3.3 proves that *adaptive* kernels for $\{\boldsymbol{f}_p\} \subset \mathbb{R}^N$ can be equivalently replaced by a *fixed* bandwidth kernel for some implicit embedding[4] $\{\boldsymbol{f}'_p\} \subset \mathbb{R}^{N'}$ in a new space. Below we establish a relation between three local properties at point p: adaptive bandwidth represented by matrix $\boldsymbol{g}_p$ and two densities ρ_p and ρ'_p in the original and the new feature spaces. For $\varepsilon > 0$ consider an ellipsoid in the original space $\mathbb{R}^N$, see Figure 3.6(a),

$$B_p \;:=\; \{\boldsymbol{x} \mid (\boldsymbol{x} - \boldsymbol{f}_p)^\top \boldsymbol{g}_p \, (\boldsymbol{x} - \boldsymbol{f}_p) \leq \varepsilon^2\}. \tag{3.52}$$

Assuming ε is small enough so that approximation (3.47) holds, ellipsoid (3.52) covers features $\{\boldsymbol{f}_q \,|\, q \in \Omega_p\}$ for subset of points

$$\Omega_p \;:=\; \{q \in \Omega \mid d_{pq} \leq \varepsilon\}. \tag{3.53}$$

Similarly, consider a ball in the new space $\mathbb{R}^{N'}$, see Figure 3.6(b),

$$B'_p \;:=\; \{\boldsymbol{x} \mid \|\boldsymbol{x} - \boldsymbol{f}'_p\|^2 \leq \varepsilon^2 + h^2\} \tag{3.54}$$

covering features $\{\boldsymbol{f}'_q \,|\, q \in \Omega'_p\}$ for points

$$\Omega'_p \;:=\; \{q \in \Omega \mid \widetilde{d}^2_{pq} \leq \varepsilon^2 + h^2\}. \tag{3.55}$$

It is easy to see that (3.50) implies $\Omega_p = \Omega'_p$. Let ρ_p and ρ'_p be the densities[5] of points within B_p and B'_p correspondingly. Assuming $|\cdot|$ denotes volumes or cardinalities of sets, we have

$$\rho_p \cdot |B_p| \;=\; |\Omega_p| \;=\; |\Omega'_p| \;=\; \rho'_p \cdot |B'_p|. \tag{3.56}$$

Omitting a constant factor depending on ε, h, N and N' we get

$$\rho'_p \;=\; \rho_p \, \frac{|B_p|}{|B'_p|} \;\propto\; \rho_p \, |\det \boldsymbol{g}_p|^{-\frac{1}{2}} \tag{3.57}$$

representing the general form of the *density law*. For the basic isotropic metric tensor such that $g_p = I/\sigma_p^2$ it simplifies to

$$\rho'_p \;\propto\; \rho_p \, \sigma_p^N. \tag{3.58}$$

[4]The implicit embedding implied by Euclidean matrix (3.50) should not be confused with embedding in the Mercer's theorem for kernel methods.

[5]We use the physical rather than probability density. They differ by a factor.

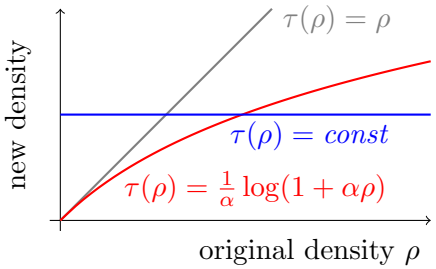
Thus, bandwidth σ_p can be selected adaptively based on any desired transformation of density $\rho'_p \equiv \tau(\rho_p)$ using

$$\sigma_p \quad \propto \quad \sqrt[N]{\tau(\rho_p)/\rho_p}. \tag{3.59}$$

where observed density ρ_p in the original feature space can be evaluated at any point p using any standard estimators, *e.g.* (3.11).

3.4.3 Density equalizing locally adaptive kernels

Bandwidth formula (3.59) works for any density transform τ. To address Breiman's bias, one can use density equalizing transforms $\tau(\rho) = const$ or $\tau(\rho) = \frac{1}{\alpha}\log(1+\alpha\rho)$, which even up the highly dense parts of the feature space as illustrated on the right. Some empirical results using density equalization $\tau(\rho) = const$ for synthetic and real data are shown in Figures 3.1(d) and 3.7(e,f).



One way to estimate the density in (3.59) is *KNN* approach [19]

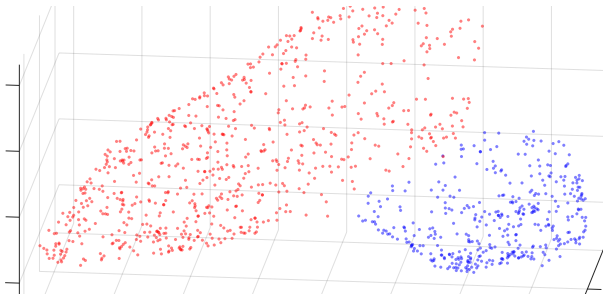
$$\rho_p \quad \approx \quad \frac{K}{nV_K} \quad \propto \quad \frac{K}{n(R_p^K)^N} \tag{3.60}$$

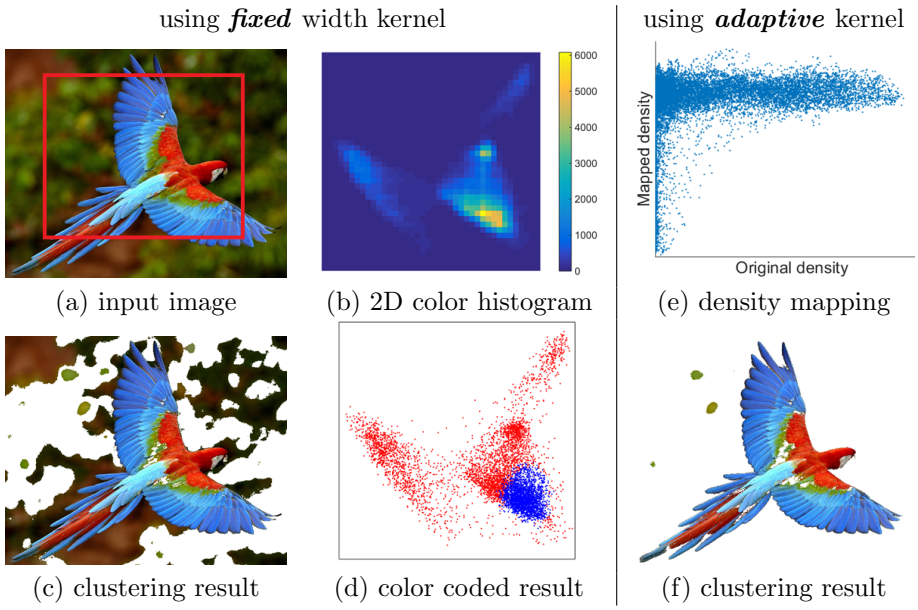
where $n \equiv |\Omega|$ is the size of the dataset, R_p^K is the distance to the K-th nearest neighbor of $\boldsymbol{f}_p$, V_K is the volume of a ball of radius R_p^K centered at $\boldsymbol{f}_p$. Then, density law (3.59) for $\tau(\rho) = const$ gives

$$\sigma_p \quad \propto \quad R_p^K \tag{3.61}$$

consistent with heuristic bandwidth in [318], see Example 3.2.

The result in Figure 3.1(d) uses adaptive Gaussian kernel (3.48) for $\Sigma_p = \sigma_p \boldsymbol{I}$ with σ_p derived in (3.61). Theorem 3.3 claims equivalence to a fixed bandwidth kernel in some transformed higher-dimensional space $\mathbb{R}^{N'}$. Bandwidths (3.61) are chosen specifically to equalize the data density in this space so that $\tau(\rho) = const$. The picture on the right illustrates such density equalization for the





(a) input image (b) 2D color histogram (e) density mapping

(c) clustering result (d) color coded result (f) clustering result

Figure 3.7: (a)-(d): *Breiman's bias* for fixed bandwidth kernel (3.1). (f): result for (3.48) with adaptive bandwidth (3.61) s.t. $\tau(\rho) = const$. (e) *density equalization*: scatter plot of empirical densities in the original/new feature spaces obtained via (3.11) and (3.50).

regularization (boundary smoothness)	**average error, %** Gaussian AA	Gaussian NC	*KNN* AA	*KNN* NC
none†	20.4	17.6	**12.2**	12.4
Euclidean length*	15.1	16.0	**10.2**	11.0
contrast-sensitive*	9.7	13.8	**7.1**	7.8

Table 3.1: Interactive segmentation results. AA stands for the average association, NC stands for the normalized cut. Errors are averaged over the GrabCut dataset [240], see samples in Figure 3.8. *We use [281, 277] for a combination of Kernel K-means objective (3.8) with *Markov Random Field* (MRF) regularization terms. The relative weight of the MRF terms is chosen to minimize the average error on the dataset. †Without the MRF term, [281] and [277] correspond to the standard kernel K-means [76, 56].

data in Figure 3.1(d). It shows a 3D projection of the transformed data obtained by *multi-dimensional scaling* [65] for matrix $(\widetilde{d}_{pq})$ in (3.50). The observed density equalization removes Breiman's bias from the clustering in Figure 3.1(d).

Real data experiments for kernels with adaptive bandwidth (3.61) are reported in Figures 3.2, 3.3, 3.7, 3.8 and Table 3.1. Figure 3.7(e) illustrates the empirical *density equalization* effect for this bandwidth. Such data homogenization removes the conditions leading to Breiman's bias, see Theorem 3.2. Also, we observe empirically that *KNN* kernel is competitive with adaptive Gaussian kernels, but its sparsity gives efficiency and simplicity of implementation.

3.5 Normalized Cut and Breiman's bias

Breiman's bias for kernel K-means criterion (3.8), a.k.a. *average association* (AA) (3.9), was empirically identified in [253], but our Theorem 3.2 is its first theoretical explanation. This bias was the main critique against AA in [253]. They also criticize *graph cut* [308] that "favors cutting small sets of isolated nodes". These critiques are used to motivate *normalized cut* (NC) criterion (3.10) aiming at balanced clustering without "clumping" or "splitting".

We do not obeserve any evidence of the *mode isolation bias* in NC. However, Section 3.5.1 demonstrates that NC still has a bias to isolating sparse subsets. Moreover, using the general density analysis approach introduced in Section 3.4.2 we also show in Section 3.5.2 that *normalization* implicitly corresponds to some density-inverting embedding

Figure 3.8: Representative interactive segmentation results. Regularized average association (AA) with fixed bandwidth kernel (3.1) or adaptive *KNN* kernels (Example 3.3) is optimized as in [281]. Red boxes define initial clustering, green contours define ground-truth clustering. Table 3.1 provides the error statistics. Breiman's bias manifests itself by isolating the most frequent color from the rest.

of the data. Thus, *mode isolation* (Breiman's bias) in this implicit embedding corresponds to the *sparse subset bias* of NC in the original data.

3.5.1 Sparse subset bias in Normalized Cut

The normalization in NC does not fully remove the bias to small isolated subsets and it is easy to find examples of "splitting" for weakly connected nodes, see Figure 3.9(a). The motivation argument for the NC objective below Fig.1 in [253] implicitly assumes similarity matrices with zero diagonal, which excludes many common similarities like Gaussian kernel (3.1). Moreover, their argument is built specifically for an example with a single isolated point, while an isolated pair of points will have a near-zero NC cost even for zero diagonal similarities.

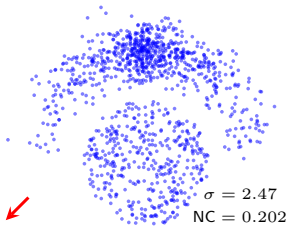
Intuitively, this NC issue can be interpreted as a bias to the "sparsest" subset (Figure 3.9a), the opposite of AA's bias to the "densest" subset, *i.e.* Breiman's bias (Figure 3.1c). The next subsection discusses the relation between these opposite biases in detail. In any case, both of these density inhomogeneity problems in NC and AA are directly addressed by our *density equalization* principle embodied in adaptive weights $w_p \propto 1/\rho_p$ in Section 3.3 or in the locally adaptive kernels derived in Section 3.4.3. Indeed, the result in Figure 3.1(d) can be replicated with NC using such adaptive kernel. Interestingly, [318] observed another data non-homogeneity problem in NC different from the sparse subset bias in Figure 3.9(a), but suggested a similar adaptive kernel as a heuristic solving it.

3.5.2 Normalization as density inversion

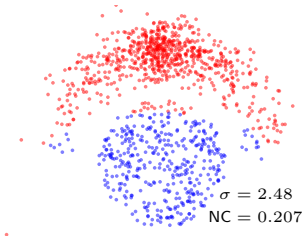
The bias to sparse clusters in NC with small bandwidths (Figure 3.9a) seems the opposite of mode isolation in AA (Figure 3.1c). Here we show that this observation is not a coincidence since NC can be reduced to AA after some density-inverting data transformation. While it is known [11, 76] that NC is equivalent to *weighted* kernel K-means (*i.e. weighted* AA) with some modified affinity, this section relates such kernel modification to an implicit density-inverting embedding where *mode isolation* (Breiman's bias) corresponds to *sparse clusters* in the original data.

First, consider standard weighted AA objective for any given affinity/kernel matrix $\hat{A}_{pq} = k(\boldsymbol{f}_p, \boldsymbol{f}_q)$ as in (3.42)

$$-\sum_k \frac{\sum_{pq\in S_k} w_p w_q \hat{A}_{pq}}{\sum_{p\in S_k} w_p}.$$



(a) NC for smaller bandwidth
(bias to "sparsest" subsets)



(b) NC for larger bandwidth
(loss of non-linear separation)

Figure 3.9: Normalized Cut with kernel (3.1) on the same data as in Figure 3.1(c,d). For small bandwidths NC shows bias to small isolated subsets (a). As bandwidth increases, the first non-trivial solution overcoming this bias (b) requires bandwidth large enough so that problems with non-linear separation become visible. Indeed, for larger bandwidths the node degrees become more uniform $d_p \approx const$ reducing NC to average association, which is known to degenerate into basic K-means (see Section 3.4.1). Thus, any further increase of σ leads to solutions even worse than (b). In this simple example no fixed σ leads NC to a good solution as in Figure 3.1(d). That good solution uses adaptive kernel from Section 3.4.3 making specific clustering criterion (AA, NC, or AC) irrelevant, see (3.71).

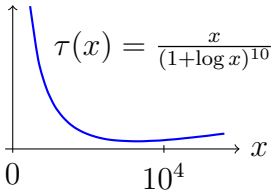
Clearly, weights based on node degrees $w = d$ and "normalized" affinities $\hat{A}_{pq} = \frac{A_{pq}}{d_p d_q}$ turn this into NC objective (3.10). Thus, average association (3.9) becomes NC (3.10) after two modifications:

- replacing A_{pq} by normalized affinities $\hat{A}_{pq} = \frac{A_{pq}}{d_p d_q}$ and
- introducing point weights $w_p = d_p$.

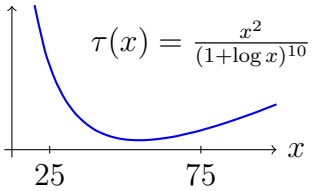
Both of these modifications of AA can be presented as implicit data transformations modifying denisty. In particular, we show that the first one "inverses" density turning sparser regions into denser ones, see Figure 3.10(a). The second data modification is generally discussed as a density transform in (3.43). We show that node degree weights $w_p = d_p$ do not remove the "density inversion".

For simplicity, assume standard Gaussian kernel (3.1) based on Euclidean distances $d_{pq} = \|\boldsymbol{f}_p - \boldsymbol{f}_q\|$ in $\mathbb{R}^N$

$$A_{pq} = \exp \frac{-d_{pq}^2}{2\sigma^2}. \tag{3.62}$$



(a) density transform (3.68)
(kernel normalization only)



(b) density transform (3.69)
(with additional point weighting)

Figure 3.10: "Density inversion" in sparse regions. Using node degree approximation $d_p \propto \rho_p$ (3.70) we show representative density transformation plots (a) $\bar{\rho}_p = \tau(\rho_p)$ and (b) $\rho'_p = \tau(\rho_p)$ corresponding to AA with kernel modification $\hat{A}_{pq} = \frac{A_{pq}}{d_p d_q}$ (3.68) and additional point weighting $w_p = d_p$ (3.69) exactly corresponding to NC. This additional weighting weakens the density inversion in (b) compared to (a), see the x-axis scale difference. However, it is easy to check that the minima in (3.68) and (3.69) are achieved at some x^* exponentially growing with $\bar{N}$. This makes the density inversion significant for NC since $\bar{N}$ may equal the data size.

To convert AA into NC we first need an affinity "normalization"

$$\hat{A}_{pq} = \frac{A_{pq}}{d_p d_q} = \exp \frac{-d_{pq}^2 - 2\sigma^2 \log(d_p d_q)}{2\sigma^2} = \exp \frac{-\hat{d}_{pq}^2}{2\sigma^2} \tag{3.63}$$

equivalently formulated as a modification of distances

$$\hat{d}_{pq}^2 \;:=\; d_{pq}^2 + 2\sigma^2 \log(d_p d_q). \tag{3.64}$$

Using a general approach in the proof of Theorem 3.3, there exists some Euclidean embedding $\bar{\boldsymbol{f}}_p \in \mathbb{R}^{\bar{N}}$ and constant $h \geq 0$ such that

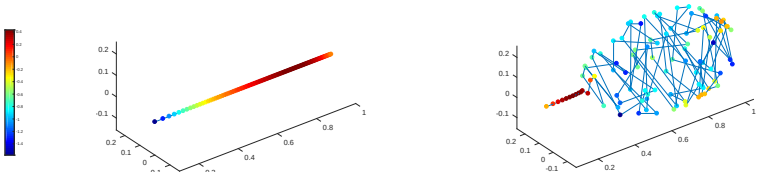
$$\bar{d}_{pq}^2 \;:=\; \|\bar{\boldsymbol{f}}_p - \bar{\boldsymbol{f}}_q\|^2 \;=\; \hat{d}_{pq}^2 + h^2[p \neq q]. \tag{3.65}$$

Thus, modified affinities $\hat{A}_{pq}$ in (3.63) correspond to the Gaussian kernel for the new embedding $\{\bar{\boldsymbol{f}}_p\}$ in $\mathbb{R}^{\bar{N}}$

$$\hat{A}_{pq} \;\propto\; \exp \frac{-\bar{d}_{pq}^2}{2\sigma^2} \;\equiv\; \exp \frac{-\|\bar{\boldsymbol{f}}_p - \bar{\boldsymbol{f}}_q\|^2}{2\sigma^2}. \tag{3.66}$$

Assuming $d_q \approx d_p$ for features $\boldsymbol{f}_q$ near $\boldsymbol{f}_p$, equations (3.64) and (3.65) imply the following relation for such neighbors of $\boldsymbol{f}_p$

$$\bar{d}_{pq}^2 \;\approx\; d_{pq}^2 + h^2 + 4\sigma^2 \log(d_p). \tag{3.67}$$



(a) original data $\{\boldsymbol{f}_p\} \subset \mathbb{R}^1$ (b) embedding $\{\bar{\boldsymbol{f}}_p\} \subset \mathbb{R}^{\bar{N}}$

Figure 3.11: Illustration of "density inversion" for 1D data. The original data points (a) are getting progressively denser along the line. The points are color-coded according to the log of their density. Plot (b) shows 3D approximation $\{\boldsymbol{y}_p\} \subset \mathbb{R}^3$ of high-dimensional Euclidean embedding $\{\bar{\boldsymbol{f}}_p\} \subset \mathbb{R}^{\bar{N}}$ minimizing metric errors $\sum_{pq} \left(\hat{d}_{pq}^2 - \|\boldsymbol{y}_p - \boldsymbol{y}_q\|^2\right)^2$ where $\hat{d}_{pq}$ are distances (3.64).

Then, similarly to the arguments in (3.56), a small ball of radius ε centered at $\boldsymbol{f}_p$ in $\mathbb{R}^N$ and a ball of radius $\sqrt{\varepsilon^2 + h^2 + 4\sigma^2 \log(d_p)}$ at $\bar{\boldsymbol{f}}_p$ in $\mathbb{R}^{\bar{N}}$ contain the same number of points. Thus, similarly to (3.57) we get a relation between densities at points $\boldsymbol{f}_p$ and $\bar{\boldsymbol{f}}_p$

$$\bar{\rho}_p \;\approx\; \frac{\rho_p\, \varepsilon^N}{(\varepsilon^2 + h^2 + 4\sigma^2 \log(d_p))^{\bar{N}/2}}. \tag{3.68}$$

This implicit density transformation is shown in Figure 3.10(a). Sub-linearity in dense regions addresses mode isolation (Breiman's bias). However, sparser regions become relatively dense and kernel-modified AA may split them. Indeed, the result in Figure 3.9(a) can be obtained by AA with normalized affinity $\frac{A_{pq}}{d_p d_q}$.

The second required modification of AA introduces point weights $w_p = d_p$. It has an obvious equivalent formulation via data points replication discussed in Section 3.3, see Figure 3.4(a). Following (3.43), we obtain its implicit density modification effect $\rho'_p = d_p \bar{\rho}_p$. Combining this with density transformation (3.68) implied by affinity normalization $\frac{A_{pq}}{d_p d_q}$, we obtain the following density transformation effect corresponding to NC, see Figure 3.10(b),

$$\rho'_p \;\approx\; \frac{d_p\, \rho_p\, \varepsilon^N}{(\varepsilon^2 + h^2 + 4\sigma^2 \log(d_p))^{\bar{N}/2}}. \tag{3.69}$$

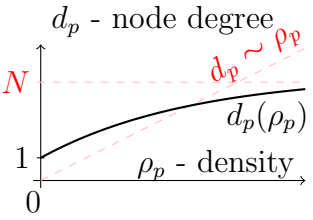
The density inversion in sparse regions relates NC's result in Figure 3.9(a) to Breiman's bias for embedding $\{\bar{f}_p\}$ in $\mathbb{R}^{\bar{N}}$.

Figure 3.10 shows representative plots for density transformations (3.68), (3.69) using the following node degree approximation based on Parzen approach (3.11) for Gaussian

affinity (kernel) A

$$d_p = \sum_q A_{pq} \propto \rho_p. \tag{3.70}$$

Empirical relation between d_p and ρ_p is illistrated below: some overestimation occurs for sparcer regions and underestimation happens for denser regions. The node degree for Gaussian kernels has to be at least 1 (for an isolated node) and at most N (for a dense graph).



3.6 Discussion

Density equalization with adaptive weights in Section 3.3 or adaptive kernels in Section 3.4 are useful for either AA or NC due to their density biases (mode isolation or sparse subset). Interestingly, kernel clustering criteria discussed in [253] such as normalized cut (NC), *average cut* (AC), average association (AA) or kernel K-means are practically equivalent for such adaptive methods. This can be seen both empirically (Table 3.1) and conceptually. Note, weights $w_p \propto 1/\rho_p$ in Section 3.3 produce modified data with near constant node degrees $d'_p \propto \rho'_p \propto 1$, see (3.70) and (3.43). Alternatively, KNN kernel (Example 3.3) with density equalizing bandwidth (3.61) also produces nearly constant node degrees $d_p \approx K$ where K is the neighborhood size. Therefore, both cases give

$$-\frac{\sum_{pq\in S^k} A_{pq}}{\sum_{p\in S^k} d_p} \;\propto\; -\frac{\sum_{pq\in S^k} A_{pq}}{K\,|S^k|} \;\overset{c}{\approx}\; \frac{\sum_{p\in S^k, q\in \bar{S}^k} A_{pq}}{K\,|S^k|}, \tag{3.71}$$

which correspond to NC (3.10), AA (3.9), and AC criteria. As discussed in [253], the last objective also has very close relations with standard partitioning concepts in spectral graph theory: *isoperimetric* or *Cheeger number*, *Cheeger set*, *ratio cut*.

This equivalence argument applies to the corresponding clustering objectives and is independent of specific optimization algorithms developed for them. Interestingly, the relation between (3.9) and basic K-means objective (3.3) suggests that standard Lloyd's algorithm can be used as a basic iterative approach for approximate optimization of all clustering criteria in (3.71). In practice, however, kernel K-means algorithm corresponding to the exact high-dimensional embedding $\{\boldsymbol{\phi}_p\}$ in (3.3) is more sensitive to local minima compared to iterative K-means over approximate lower-dimensional embeddings based on

the principal component analysis (PCA) [281][6].

This chapter identifies and proves density biases, *i.e.* isolation of modes or sparsest subsets, in many well-known kernel clustering criteria such as kernel K-means (average association), ratio cut, normalized cut, dominant sets. In particular, we show conditions when such biases happen. Moreover, we propose density equalization as a general principle for resolving such biases. We suggest two types of density equalization techniques using adaptive weights or adaptive kernels. We also show that density equalization unifies many popular kernel clustering objectives by making them equivalent.

[6]K-means is also commonly used as a discretization heuristic for *spectral relaxation* [253] where a similar eigen analysis is motivated by spectral graph theory [49, 77, 86] defferently from PCA dimensionalty reduction in [281].

Chapter 4

MRF/CRF Optimization in Deep Learning

Acquisition of training data for the standard semantic segmentation is expensive if requiring each pixel to be labeled. Yet, current methods significantly deteriorate in weakly supervised settings, *e.g.* where a fraction of pixels is labeled or when only image-level tags are available, see Section 1.6. It has been shown that regularized losses—originally developed for unsupervised low-level segmentation and representing geometric priors on pixel labels—can considerably improve the quality of weakly supervised training. However, many common priors require optimization stronger than gradient descent, see Section 1.2.4 and 1.2.5. Thus, such regularizers have limited applicability in deep learning. We propose a new robust trust region approach for regularized losses improving the state-of-the-art results. Our approach can be seen as a higher-order generalization of the classic backpropagation/chain rule. It allows neural network optimization to use strong low-level solvers for the corresponding regularizers, including discrete ones.

4.1 Preliminaries

We proposes a higher-order optimization technique for neural network training. While focused on semantic image segmentation, our main algorithmic idea is simple and general - integrate the standard *trust region* principle into the context of *backpropagation*, *i.e.* the chain rule. We reinterpret the classic chain rule: instead of the chain of gradients/derivatives for a composition of functions, we formulate the corresponding chain of

hidden optimization sub-problems. Then, inspired by the *trust region* principle, we can substitute a standard linear approximation solver (gradient descent) at any chain with a better higher-order solver. In short, we replace the classic differentiation chain rule by the trust region chain rule in the context of backpropagation.

Our work is motivated by the well-known challenges presented to the gradient descent by typical regularization losses or geometric priors/energies ubiquitous in the context of weakly-supervised or unsupervised segmentation. To validate our approach, we present semantic segmentation results improving the state-of-the-art in the challenging setting where the training data has only a fraction of pixels labeled. The generality of our main principle (trust region chain rule) and our promising results for a difficult problem encourage further research. In fact, this work applies trust region principle only to the last "chain" in the network. We discuss several promising extensions for future work.

The rest of the introduction is organized as follows. To create a specific context for our general approach to network training, we review loss functions relevant for weakly-supervised or unsupervised segmentation. First, Section 4.1.1 discusses several standard geometric priors, regularization energies, clustering criteria, and their powerful solvers originally developed for low-level segmentation or general machine learning. Then, Section 4.1.2 outlines the use of such regularization objectives as losses for network training in the context of weakly supervised semantic (high-level) segmentation. We also review the standard *trust region* principle (Section 4.1.4) and highlight our main contributions (Section 4.1.5) based on the general idea of applying trust region (with powerful solvers) to network training.

4.1.1 Regularized energies in low-level segmentation

Assuming discrete segmentation $s \in \{1, 2, \dots, K\}^N$ where K is the number of categories and N is the number of image pixels, one common low-level segmentation energy can be represented as

$$E(s) \;=\; -\sum_i \log P(\boldsymbol{I}_i|s_i) \;+\sum_{\{i,j\}\in N} w_{ij}\,[s_i \neq s_j] \tag{4.1}$$

where $\boldsymbol{I}_i$ is a low-level feature (*e.g.* intensity, color, texture) at pixel i with distribution functions $P(\cdot|k)$ for each category k, neighborhood system N describes any pairwise connectivity (typically 4-, 8-grid [31] or denser [153]), weights w_{ij} represent given pairwise affinities (typically Gaussian kernel for low-level features $\boldsymbol{I}_i$ and $\boldsymbol{I}_j$ [29, 31, 240, 153]), and $[\cdot]$ is the Iverson bracket operator returning 1 if the argument is true and 0 otherwise. The energy above combines the log-likelihoods term enforcing consistency with given

(low-level) feature distributions and a pairwise regularizer (Potts model) term enforcing geometric prior on shape smoothness with alignment to image intensity edges.

The Potts model has several efficient combinatorial [29] and LP-relaxation solvers [149, 158]. Besides, there are many regularization objectives that are closely related to the first-order shape regularization in (4.1), but derived from a different discrete or continuous formulation of the low-level segmentation and equipped with their own efficient solvers, *e.g.* geodesic active contours [41], snakes [138], power watersheds [63], to name a few. Moreover, there are many other regularization terms going beyond the basic first-order smoothness (boundary length) enforced by the Potts term in (4.1). The extensions include curvature [252, 213, 206], Pn-Potts [147], convexity [106, 105, 130], etc.

Common continuous formulations of the low-level segmentation use *relaxed* variable $\boldsymbol{s} \in \Delta_K^N$ combining pixel-specific distributions $\boldsymbol{s}_i = (s_i^1, \ldots, s_i^K) \in \Delta_K$ over K categories, where Δ_K is the *probability simplex*. In this case the segmentation objective/energy should also be relaxed, *i.e.*, defined over real-values arguments. For example, one basic relaxation of the Potts segmentation energy in (4.1) is

$$-\sum_i \sum_k s_i^k \log P(\boldsymbol{I}_i|k) \; + \sum_{\{i,j\} \in N} w_{ij} \, \|\boldsymbol{s}_i - \boldsymbol{s}_j\|^2 \tag{4.2}$$

using a linear relaxation of the likelihood term and a quadratic relaxation of the Potts model. Note that there could be infinitely many alternative relaxations. Any specific choice affects the properties of the relaxed solution, as well as the design of the corresponding optimization algorithm. For example, simple *quadratic* relaxation in (4.2) is convex suggesting simpler optimization, but its known to be a non-tight relaxation of the Potts model [235] leading to weaker regularization properties unrelated to geometry or shape. There are many better alternatives, *e.g.* using different norms [63] or other convex formulations [44, 42, 43]. The *bilinear* relaxation of the Potts term below

$$-\sum_i \sum_k s_i^k \log P(\boldsymbol{I}_i|k) \; + \sum_k (1 - s^k)^\top W \, s^k \tag{4.3}$$

is tight [235], but it is non-convex and, therefore, more difficult to optimize. In the formula above, vector $\boldsymbol{s}^k := (s_i^k)$ combines segmentation variables for soft-segment k, and $N \times N$ affinity matrix $W_{ij} = w_{ij} \, [\{i,j\} \in N]$ represents the neighborhood system N and all pairwise (*e.g.* Gaussian) affinities w_{ij} between image pixels. Note that Potts regularization is closely related to the *Normalized cut* objective $\sum_k \frac{(1-\boldsymbol{s}^k)^\top \boldsymbol{W} \, s^k}{1^\top \boldsymbol{W} \, \boldsymbol{s}^k}$ for unsupervised segmentation [253].

It is common to combine energies like (4.1),(4.2),(4.3) with constraints based on user interactions (weak supervision). While there are different forms of such supervision, the

most basic one is based on adding the seed loss [31] defined over pixels in subset Ω_{seeds} with user-specified category labels y_i. Assuming $s_i \in \Delta_K$, it can be written as a partial *cross entropy* (PCE) for pixels $i \in \Omega_{\text{seeds}}$

$$E_{\text{seeds}}(\boldsymbol{s}) \;=\; -\sum_{i \in \Omega_{\text{seeds}}} \log s_i^{y_i} \tag{4.4}$$

and, when restricted to *one-hot* s_i representing hard segmentation, it reduces to the hard constraints over seeds [31]. That is, for integer-valued $s_i \in \{1, \ldots, K\}$ the seed loss is equivalent to $\sum_{i \in \Omega_{\text{seeds}}} \lambda \, [s_i = y_i]$ for infinitely large λ.

The log-likelihood loss, *e.g.* the first term in (4.1) or (4.3), is common in low-level segmentation and its importance cannot be underestimated. In basic formulations, the distributions of (low-level) features $P(\cdot|k)$ can be assumed given for each category k. However, if such distributions are not known *a priori*, their representation $P(\cdot|\boldsymbol{\theta}_k)$ can explicitly include unknown distribution parameters $\boldsymbol{\theta}_k$ for each category k. Then, the overall loss $E(\boldsymbol{s}, \boldsymbol{\theta})$ adds $\theta = \{\boldsymbol{\theta}_k\}$ as an extra variable. Optimization of $E(\boldsymbol{s}, \boldsymbol{\theta})$ over both s and $\boldsymbol{\theta}$ corresponds to joint estimation of segmentation and *maximum likelihood* (ML) estimation of distribution parameters, as in well-known unsupervised low-level segmentation formulations by Zhu & Yuille [330] and Chan & Vese [46]. Similar ideas are also used in box-interaction methods [240].

4.1.2 Regularized losses in DNN segmentation

Unlike low-level segmentation methods based on readily available low-dimensional features (like color, texture, contrast edges), deep neural network (DNN) approaches to segmentation learn complex high-dimensional "deep" features that can discriminate semantic categories. Thus, one can refer to such methods as *high-level* segmentation, and to such learned features as *high-level* features.

The most standard way to train segmentation networks is based on *full supervision* requiring a large collection of images where all pixels are accurately labeled. Such training data is expensive to get. The training is based on minimizing the cross-entropy (CE) loss similar to the seed loss in low-level segmentation. For simplicity focusing on a single training image, CE loss is

$$E_{\text{CE}}(s(\boldsymbol{\theta})) \;=\; -\sum_{i} \log s_i^{y_i}(\boldsymbol{\theta}) \tag{4.5}$$

where $s(\boldsymbol{\theta}) = \phi(\boldsymbol{\theta}) \in \Delta_K^N$ is the (relaxed) segmentation output of the network $\phi(\boldsymbol{\theta})$ with parameters $\boldsymbol{\theta}$. For brevity, here and later in this chapter we omit the actual test image from the arguments of the network function ϕ.

The fundamental difference with low-level segmentation reviewed above is that instead of minimizing losses E directly over segmentation variable $\boldsymbol{s}$, now the optimization arguments are parameters $\boldsymbol{\theta}$ of the network producing such segmentation. Estimating parameters $\boldsymbol{\theta}$ can be interpreted as learning deep features. Note that this task is much more complex than ML estimation of distribution parameters for $P(\boldsymbol{I}|\boldsymbol{\theta})$ in low-level segmentation with fixed low-level features $\boldsymbol{I}$, as reviewed above. This explains why network optimization requires a large set of fully labeled training images, rather then a single image (unlabeled or partially-labeled), as in low-level segmentation.

The goal of weakly supervised segmentation is to train the network with as little supervision as possible. First of all, it is possible to train using only a subset of labeled pixels (seeds) in each image [148, 279] in exact analogy with (4.4)

$$E_{\text{PCE}}(s(\boldsymbol{\theta})) \;=\; -\sum_{i \in \Omega_{\text{seeds}}} \log s_i^{y_i}(\boldsymbol{\theta}) \tag{4.6}$$

In particular, as shown in [279], this simple, but principled approach can outperform more complex heuristic-based techniques. To improve weakly-supervised training, it is also possible to use standard low-level regularizes, as in Section 4.1.1, that leverage a large number of unlabeled pixels [148, 279, 280, 187]. For example, [280] achieves the state-of-the-art using bilinear relaxation of the Potts model in (4.3)

$$E^{\text{bl}}_{\text{Potts}}(s(\boldsymbol{\theta})) \;=\; \sum_k (1 - s^k(\boldsymbol{\theta}))^\top W\, s^k(\boldsymbol{\theta}) \tag{4.7}$$

as an additional regularization loss over all (including unlabeled) pixels. For some $\nu > 0$, their continuous total loss

$$E \;=\; E_{\text{PCE}} \;+\; \nu\, E^{\text{bl}}_{\text{Potts}}. \tag{4.8}$$

More generally, standard regularization losses from low-level segmentation are commonly used in the context of segmentation networks. Such losses and their solvers are ubiquitous in weak-supervision techniques using seeds or boxes to generate fully-labeled *proposals* [143, 172]. Optimization of low-level regularizers is also common for network's output post-processing, typically improving performance during testing [53]. Also, the corresponding low-level solvers can be directly integrated as solution-improving layers [325].

4.1.3 Weakly supervised semantic segmentation

Weak supervision for deep neural network semantic segmentation comes in many different forms, *e.g.* image-level tags [225, 221, 148], scribbles/clicks [172, 279, 280, 187], and bounding boxes [221, 143, 142]. These works employ a large variety of strategies to compensate for the lack of labels. The concept of *multiple instance learning* (MIL) naturally fits the weakly supervised setting. Since generic MIL methods produce small unsatisfactory segments, more specialized methods are needed. For example, methods [225, 142] impose constraints on the output of the neural network during learning. There are several segmentation-specific constraints, such as size bias, constraints on present labels, tightness [165], *etc.* [148, 280, 187] incorporate edge alignment constraints. Proposal generation methods [143, 172] aim to generate/complete the ground truth to use fully-supervised learning. However, DNNs are vulnerable to errors in proposals. More robust approaches use EM [221] or ADMM [187] to iteratively correct errors in "proposals".

Some related prior work on weakly supervised DNN segmentation [172] uses some specific non-robust version of the joint loss related to our approach. Similar losses (studied in segmentation since 1980s) do not imply similar algorithms. In particular, they iterate explicit low-level segmentation of super-pixels [85] and pixel-level network training, where at each iteration the network is trained from scratch[1] and to convergence. They motivate such integration by improved results only. They also argue that "when network gradually learns semantic content, the high-level information can help with the graph-based scribble propagation", suggesting their main focus on improved "proposals". As shown in [279, 280], their method is outperformed by using only the partial cross entropy on seeds (4.6).

4.1.4 Classic *trust region* optimization

Trust region is a general approximate iterative local optimization method [25] allowing to use approximations with good solvers when optimizing arbitrarily complex functions. To optimize $g(\boldsymbol{x})$, it solves sub-problem $\min_{\|\boldsymbol{x}-\boldsymbol{x}_t\|\leq\epsilon} \tilde{g}(\boldsymbol{x})$ where function $\tilde{g} \approx g$ is an approximation that can be "trusted" in some region $\|\boldsymbol{x} - \boldsymbol{x}_t\| \leq \epsilon$ around the current solution. If $\tilde{g}$ is a linear expansion of g, this reduced to the gradient descent. More accurate higher-order approximations can be trusted over larger regions allowing larger steps. The sub-problem is often formulated as unconstrained Lagrangian optimization $\min_{\boldsymbol{x}} \tilde{g}(\boldsymbol{x}) + \lambda\|\boldsymbol{x} - \boldsymbol{x}_t\|$ where λ indirectly controls the step size.

[1] That is, resetting the network to the ImageNet pre-trained parameters.

4.1.5 Related optimization work and contributions

The first-order methods based on stochastic gradient descent dominate deep learning due to their simplicity, efficiency, and scalability. However, they often struggle to escape challenging features of the loss profile, *e.g.* "valleys", as the gradients lack information on the curvature of the loss surface. Adam [145] combines gradients from many iterations to gather such curvature information. On the other hand, the second-order methods compute parameters update in the form $\Delta\theta = \boldsymbol{H}^{-1}\nabla_\theta E(\phi(\boldsymbol{\theta}))$, *cf.* (4.10), where $\boldsymbol{H}$ is the Hessian or its approximation. In neural networks, computing the Hessian is infeasible, so various approximations are used, *e.g.* diagonal or low-rank [19]. The efficient computation of Hessian-vector products is possible [230, 248]; while solving linear systems with Hessian is still challenging [269]. Another group of methods is based on employing Gaussian-Newton matrix and K-FAC approximations [188, 10, 23, 218].

Our approach is related to the proximal methods [191], in particular to the *proximal backpropagation* [93] and *penalty method* [40]. In these works, the "separation" of the gradient update into implicit layer-wise optimization problems is formulated as a gradient update of a certain energy function. Taylor *et al.* [282] use *ADMM* splitting approach to separate optimization over different layers in distributed fashion. These works focus on neural network parameter optimization replacing backpropagation altogether. In contrast to [40, 282, 93], we are primarily focused on optimization for complex loss functions in the context of the weakly supervised semantic segmentation, see Section 4.1.2, while others focus on replacing the backpropagation in the intermediate layers. Also, unlike us, these methods use the squared Euclidean norm in their proximal formulations. Chen and Teboulle [51] generalize the proximal methods to *Bregman divergences*, a more general class of functions which includes both the Euclidean distance and KL-divergence. Nesterov in [205] uses the Euclidean norm with a higher power improving the convergence of the proximal method.

Our contribution are as follows:

- New trust region optimization for DNN segmentation integrating higher-order low-level solvers into training. Differentiability of the loss is not required as long as there is a good solver, discrete or continuous. The classic differentiation chain rule is replaced by the trust region chain rule in the context of backpropagation.
- The local optimization in trust region framework allows to use arbitrary metrics, instead of Euclidean distance implicit for the standard gradient descent. We discuss different metrics for the space of segmentations and motivate a robust version of KL-divergence.

- We show benefits of our optimization for regularization losses in weakly supervised DNN segmentation, compared to the gradient descent. We set new state-of-the-art results for weakly supervised segmentation with scribbles achieving consistently the best performance at all levels of supervision, *i.e.* from point-clicks to full-length scribbles.

4.2 Trust region for loss optimization

Backpropagation is the dominant method for optimizing network losses during training. It represents the gradient descent with respect to model parameters $\boldsymbol{\theta}$ where the gradient's components are gradually accumulated using the classic *chain rule* while traversing the network layers starting from the output directly evaluated by the loss function.

Motivated by the use of hard-to-optimize regularization losses (Section 4.1.1) in the context of weakly-supervised segmentation (Section 4.1.2), we propose higher-order *trust region* approach to network training. While this general optimization approach can be developed for any steps of the backpropagation (*i.e.* chain rule) between internal layers, we focus on the very first step where the loss function is composed with the network output

$$\min_{\theta \in \mathbb{R}^m} \; E(\phi(\boldsymbol{\theta})) \tag{4.9}$$

where some scalar loss function

$$E : \mathbb{R}^n \to \mathbb{R}^1$$

is defined over n-dimensional output of a network/model

$$\phi : \mathbb{R}^m \to \mathbb{R}^n.$$

Since during training the network's input is limited to fixed examples, for simplicity we restrict the arguments of network function ϕ to its training parameters $\boldsymbol{\theta} \in \mathbb{R}^m$. Also note that, as a convention, this chapter reserves the boldface font for vector functions (*e.g.* network model ϕ) and for matrix functions (*e.g.* model's Jacobian $\boldsymbol{J}_\phi$).

The main technical ideas of the trust region approach to network optimization (4.9) in this section are fairly general. However, to be specific and without any loss of generality, this and (particularly) later sections can refer to the output of the network as *segmentation* so that

$$\mathbb{R}^n = \mathbb{R}^{N \times K}$$

where N is the number of image pixels and K is the number of distinct semantic classes. This is not essential.

Our general trust region approach to (4.9) can be seen as a higher-order extension of the classic chain rule for the composition $E \circ \phi$ of the loss functions E and model ϕ. For the classic chain rule in the standard backpropagation procedure, it is critical that both E and ϕ are differentiable. In this case, the classic chain rule for the objective in (4.9) gives the following gradient descent update for parameters $\boldsymbol{\theta}$

$$\Delta\boldsymbol{\theta} \quad = \quad -\alpha \, \nabla E^\top \, \boldsymbol{J}_\phi \tag{4.10}$$

where $\Delta\boldsymbol{\theta} \equiv \boldsymbol{\theta} - \boldsymbol{\theta}_t$ is an update of the model parameters from the current solution, α is the learning rate, ∇ is the gradient operator, and $\boldsymbol{J}_\phi$ is the model's *Jacobian*

$$\boldsymbol{J}_\phi := \left[\frac{\partial \phi_i}{\partial \theta^j}\right].$$

We would like to rewrite the classic chain rule (4.10) in an equivalent form explicitly using a variable for segmentation $\boldsymbol{s} \in \mathbb{R}^n$, which is an implicit (hidden) argument of the loss function E in (4.9). Obviously, equation (4.10) is equivalent to two separate updates for the segmentation $\Delta\boldsymbol{s} \equiv \boldsymbol{s} - \boldsymbol{s}_t$ and for the model parameters $\Delta\boldsymbol{\theta} \equiv \boldsymbol{\theta} - \boldsymbol{\theta}_t$

$$\Delta\boldsymbol{s} \quad = \quad -\,\alpha \;\nabla E^\top \tag{4.11}$$

$$\Delta\boldsymbol{\theta} \quad = \quad \Delta\boldsymbol{s} \;\boldsymbol{J}_\phi \tag{4.12}$$

where the gradient ∇E is computed at the current segmentation $\boldsymbol{s}_t := \phi(\boldsymbol{\theta}_t)$. Note that $\boldsymbol{s} \in \mathbb{R}^n$ represents points (*e.g.* segmentations) in the same space as the network output $\phi(\theta) \in \mathbb{R}^n$, the two should be clearly distinguished in the discourse. We will refer to $\boldsymbol{s}$ as (explicit) segmentation *variable*, while $\phi(\boldsymbol{\theta})$ is referred to as segmentation *output*.

The updates in (4.11) and (4.12) correspond to two distinct optimization sub-problems. Clearly, (4.11) is the gradient descent step for the loss $E(\boldsymbol{s})$ locally optimizing its linear Taylor approximation $\tilde{E}_{\text{linear}}(\boldsymbol{s}) = E(\boldsymbol{s}_t) + \nabla E^\top \Delta\boldsymbol{s}$ over (explicit) segmentation variable $\boldsymbol{s} \in B(\boldsymbol{s}_t) \subset R^n$ in a neighborhood (ball) around $\boldsymbol{s}_t$

$$\boldsymbol{s}_{t+1} \quad = \quad \arg\min_{\boldsymbol{s} \in B(\boldsymbol{s}_t)} \tilde{E}_{\text{linear}}(\boldsymbol{s}). \tag{4.13}$$

While less obvious, it is easy to verify that $\boldsymbol{\theta}$-update in (4.12) is the gradient descent step

$$\Delta\boldsymbol{\theta} \;=\; -\frac{1}{2}\nabla_{\boldsymbol{\theta}} \, \|\boldsymbol{s}_{t+1} - \phi(\boldsymbol{\theta})\|^2 \tag{4.14}$$

corresponding to optimization of the least-squares objective

$$\min_{\boldsymbol{\theta}} \ \|\boldsymbol{s}_{t+1} - \phi(\boldsymbol{\theta})\|^2 \tag{4.15}$$

based on the solution $\boldsymbol{s}_{t+1} \equiv \Delta\boldsymbol{s} + \phi(\boldsymbol{\theta}_t)$ for problem (4.13).

Our trust region approach to network training (4.9) is motivated by the principled separation of the chain rule (4.10) into two sub-problems (4.13) and (4.15). Instead of the gradient descent, low-level optimization of the loss in (4.13) can leverage powerful higher-order solvers available for many popular loss functions, see Section 4.1.1. In particular, the majority of common robust loss functions for unsupervised or weakly-supervised computer vision problems are well-known to be problematic for the gradient descent. For example, their robustness (boundedness) leads to *vanishing gradients* and sensitivity to local minima. At the same time, the gradient descent can be left responsible for the least-squares optimization in (4.15). While it is still a hard problem due to size and non-convexity of the typical models $\phi(\boldsymbol{\theta})$, at least the extra difficulties introduced by complex losses E can be removed into a different sub-problem.

Formally, our trust-region approach to training (4.9) generalizes our interpretation of the classic chain rule in sub-problems (4.13) and (4.15) as shown in iterative stages A, B:

$$\text{STAGE A} \qquad (\textit{low-level optimization})$$

$$\boldsymbol{s}_{t+1} \ = \ \arg\min_{\boldsymbol{s}} \ \tilde{E}(\boldsymbol{s}) \ + \ \lambda \, d_{\text{A}}(\boldsymbol{s}, \phi(\boldsymbol{\theta}_t)) \tag{4.16}$$

$$\text{STAGE B} \qquad (\textit{network parameters update})$$

$$\underbrace{\min_{\boldsymbol{\theta}} \ d_{\text{B}}(\boldsymbol{s}_{t+1}, \phi(\boldsymbol{\theta}))}_{\Downarrow} \tag{4.17}$$

$$\Delta\boldsymbol{\theta} \ = \ - \ \gamma \, \nabla_\theta \, d_{\text{B}}(\boldsymbol{s}_{t+1}, \phi(\boldsymbol{\theta})) \tag{4.18}$$

where $\tilde{E}$ is some loss approximation, d_{A} and d_{B} are some distance/divergence measures. Instead of α in (4.11) and fixed weight $\frac{1}{2}$ in (4.14), the overall learning speed of our training procedure is controlled by two parameters: (A) scalar λ indirectly determining the step size from the current solution $\boldsymbol{s}_t = \phi(\boldsymbol{\theta}_t)$ in (4.16), and (B) scalar γ defining the step size for the gradient descent in (4.18). While both λ and γ are important for the learning speed, we mostly refer to λ as a trust region parameter, while the term *learning rate* is reserved primarily for parameter γ in (4.18), as customary for the gradient descent step size in network optimization. Note that similarly to the gradient descent (4.10), stages

A/B are iterated until convergence. While it is sensible to make several B-steps (4.18) in a row, in general, it is not necessary to wait for convergence in sub-problem (4.17) before the next A-step.

Our formulation offers several significant generalizations of the classic chain rule. **First**, instead of the linear approximation (4.13) implied by the *gradient descent* (4.11), we target higher-order approximations of the loss $\tilde{E}$ in (4.16). In some cases, the exact loss E could be used[2]. The corresponding powerful low-level solvers for (4.16) are readily available for many types of useful robust losses, see Section 4.1.1. Note that for exact solvers when $\tilde{E} = E$, one may argue for $\lambda = 0$ allowing the network to learn from the best solutions for regularized loss E implying global optima in (4.9). However, such fixed proposals (Section 4.1.2) may result in overfitting to mistakes due to well-known biases/weaknesses in common regularizers. Constraining loss optimization (4.9) to the network output manifold in $\mathbb{R}^n$ motivates $\lambda > 0$ in (4.16). More discussion is in Section 4.5.1.

Second, besides continuous/differentiable losses required by the standard backpropagation (chain rule), our trust region approach (stages A/B) allows training based on losses defined over discrete domains. There are several reasons why this extension is significant. For example, besides continuous solvers, optimization in (4.16) now can use a significantly larger pool of solvers including many powerful discrete/combinatorial methods. Moreover, this approach enables training of models with discrete decision functions, e.g. *step function* instead of *sigmoid*, or *hard-max* instead of the *soft-max*. This is further discussed in Section 4.5.1.

Third, the standard gradient descent (4.10) is implicitly defined over Euclidean metric, that manifests itself in our equations (4.13) and (4.15) via the local neighborhood topology (Euclidean ball B) and the least-squares objective (squared Euclidean distance). In contrast, when replacing ball $B(\boldsymbol{s}_t)$ in (4.13) by the *trust region* term in (4.16), we explicitly define the trust region "shape" using function d_{A}. It could be any application-specific distance metric, quasi- or pseudo-metric, divergence, *etc.* Similarly, any appropriately motivated distance, distortion, or divergence function d_{B} in (4.17) can replace the least squares objective in (4.15).

On the negative side, our trust region formulation could be more expensive due to the computational costs of the low-level solvers in stage A. In practice, it is possible to amortize stage A over multiple iterations of stage B.

[2]Note that parameter λ in (4.16) controls two properties: the size of the trust region for approximation $\tilde{E}$, as well as the network's training speed. While using exact loss $\tilde{E} = E$ implies that the trust region for such "approximation" should be the whole domain (*i.e.* $\lambda = 0$), the competing interest of limiting the training speed in (4.17) may require $\lambda > 0$.

4.3 Robust metric for trust region

The choice of metrics d_{A} and d_{B} defining the shape of the trust region above is application dependent. In the case of segmentation, the output of a neural network is typically obtained via the soft-max function. Hence, the space, in which the trust region operates, is the space of multiple categorical distributions over K categories: Δ_K^N.

Below, we generally discuss (robust) metrics over pairs of arbitrary probability distributions p, q in Δ_K^N. The goal of this section is to motivate our choice of metrics d_{A} and d_{B} in problems (4.16), (4.17) so that distribution p can be associated with the segmentation variable $\boldsymbol{s}$, and distribution q can be associated with the network output $\phi(\boldsymbol{\theta})$. Besides this connection, the following discussion of metrics over probability distributions is independent of the context of networks.

Note, metrics d_{A} or d_{B} do not have to be proper distances for the purposes of trust region optimization. Instead, one may use any divergence measure defined on space Δ_K^N. Let us consider the Kullback–Leibler divergence:

$$\text{KL}(p\|q) = \sum_{i=1}^{N}\sum_{l=1}^{K} p_i^l \log \frac{p_i^l}{q_i^l} = -\sum_{i=1}^{N}\sum_{l=1}^{K} p_i^l \log q_i^l - H(p)$$

where $\boldsymbol{p}, \boldsymbol{q} \in \Delta_K^N$, and p_i^l is the probability of pixel i to have label l, and $H(\boldsymbol{p})$ is the entropy of distribution $\boldsymbol{p}$.

A practically important case is when the distribution $\boldsymbol{p}$ is degenerate or one-hot, *i.e.* for each pixel i there exists label y_i such that $p_i^{y_i} = 1$ and for any label $k \neq y_i$ probability $p_i^k = 0$. In that case $H(\boldsymbol{p}) = 0$ and

$$\text{KL}(\boldsymbol{p}\|\boldsymbol{q}) \;=\; \sum_i -\log q_i^{y_i}, \tag{4.19}$$

which is the cross-entropy or negative log-likelihood, a standard loss when $\boldsymbol{q}$ is the probability estimate outputted by a neural network. In the following we assume (4.19).

During the trust region procedure, intermediate solutions generated by a solver in (4.16) may have a noticeable amount of misclassified pixels. It is known that many standard losses for neural networks, including cross-entropy (4.19), can result in training sensitive to idiosyncrasies in the datasets including mistakes in the ground truth [98, 182, 92]. Therefore, a robust distance measure may be needed. Our experiments show that robustness is critical. We propose a simple error model depicted in graphical model in Fig. 4.1. Let random

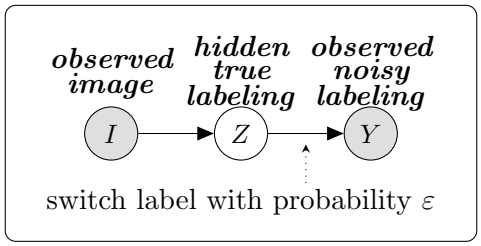


Figure 4.1: The unknown true labeling Z corresponds to observed image I. The observed labeling Y is assumed to be generated from the true Z by a simple corruption model (4.20).

variable Y_i be the observed noisy label of pixel i and Z_i be its hidden true label. We assume that the probability of observing label l given true label k is

$$\Pr(Y_i = l \,|\, Z_i = k) = \begin{cases} 1 - \varepsilon, & l = k, \\ \frac{\varepsilon}{K-1}, & l \neq k, \end{cases} \tag{4.20}$$

where ε is called the *outlier probability* [161]. The probability of pixel i having label l given image $\boldsymbol{I}$ is

$$\begin{aligned} \Pr(Y_i{=}l|\boldsymbol{I}) \;&=\; \sum_{z=1}^{K} \Pr(Y_i{=}l|Z_i{=}z)\,\Pr(Z_i{=}z|\boldsymbol{I}) \;= \\ &= a + b\,\Pr(Z_i{=}l|\boldsymbol{I}) \end{aligned} \tag{4.21}$$

where $a = \frac{\varepsilon}{K-1}$ and $b = 1 - K\,a$. The probability $\Pr(Z_i{=}z|\boldsymbol{I})$ is unknown and is replaced by probability estimate q_i^l yielding a robust version of divergence (4.19):

$$\sum_i - \log\left(a + b\, q_i^{y_i}\right). \tag{4.22}$$

Figure 4.2 compares cross-entropy (4.19) with robust loss (4.22).

Our robust cross-entropy (4.22) is related to a more general approach for classification [226, 268]. In [226], the corresponding robust cross-entropy (*forward correction*) is

$$\sum_i - \log \tilde{q}_i^{y_i} \tag{4.23}$$

where $\tilde{\boldsymbol{q}}_i = \boldsymbol{T}^\top \boldsymbol{q}_i$, and q_i is the vector of probability estimates at pixel i, and $\boldsymbol{T} = [T_{lk}]$ is the *noise transition matrix*: $T_{lk} = \Pr(Y = k \,|\, Z = l)$. The effect of different ε is shown in example in Fig. 4.3.

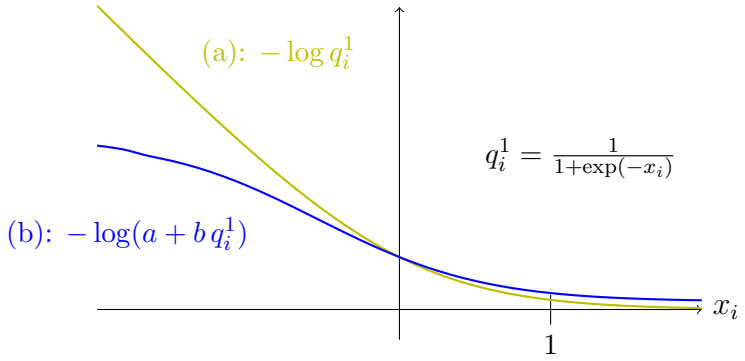


Figure 4.2: Robust loss as function of logits x_i. There are $K = 2$ classes; the ground truth label is $y_i = 1$. If the current prediction q_i^1 is confident and does not coincide with y_i, see $x_i \ll 0$ or $q_i^1 \approx 0$ on the plot, robust loss (b) becomes flatter avoiding the over-penalize in case of mistakes in the ground truth. In contrast, standard cross-entropy (a) behaves linearly, which may be detrimental to learning if the ground truth is mistaken.

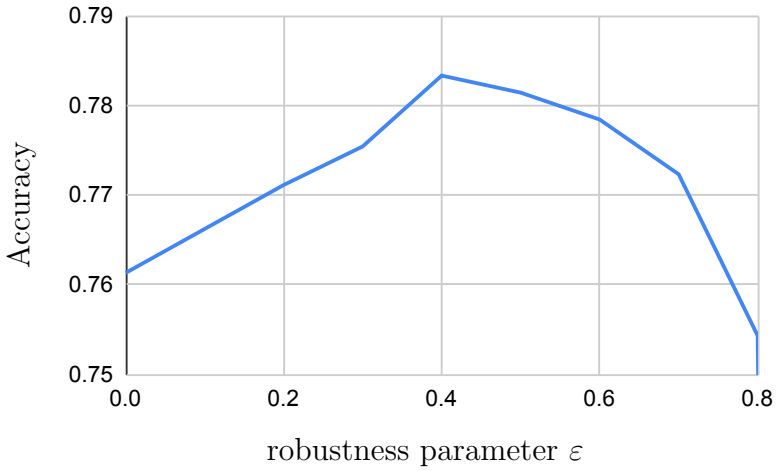


Figure 4.3: Classification accuracy on Fashion-MNIST dataset [309] with noisy labels using a network with two convolutional, two fully-connected layers and *robust loss* (4.22). The original labels were uniformly corrupted with probability $\frac{1}{2}$. The best accuracy is achieved at $\varepsilon = 0.4$, which is close to the actual noise level.

In practice, different pixels require different values of ε in (4.20). For example, in the scribble-based weakly supervised segmentation, the labels of seed pixels Ω_{seeds} are known for sure. So, $\varepsilon = 0$ for such pixels, and $\varepsilon > 0$ for all other pixels. Thus, the robust "metric"

$$\text{KL}_{\varepsilon,\Omega_{\text{seeds}}}(\boldsymbol{p}\|\boldsymbol{q}) = \sum_{i \notin \Omega_{\text{seeds}}} - \log\left(a + b\, q_i^{y_i}\right) + \sum_{i \in \Omega_{\text{seeds}}} - \log q_i^{y_i}. \tag{4.24}$$

In sum, we propose the following robust metrics for the trust region iterations (4.16) and (4.18):

$$\begin{cases} d_{\text{A}}(\boldsymbol{p}, \boldsymbol{q}) &= \text{KL}(\boldsymbol{p}\|\boldsymbol{q}), \\ d_{\text{B}}(\boldsymbol{p}, \boldsymbol{q}) &= \text{KL}_{\varepsilon,\Omega_{\text{seeds}}}(\boldsymbol{p}\|\boldsymbol{q}). \end{cases} \tag{4.25}$$

4.4 Results in weakly supervised segmentation

To validate our approach (4.16-4.18) we use standard efficient discrete solvers [29] for loss

$$\tilde{E} \;\; = \;\; E_{\text{PCE}} \; + \; E_{\text{Potts}} \tag{4.26}$$

where $E_{\text{Potts}}(s) = \sum_{\{i,j\} \in N} w_{ij}\, [s_i \neq s_j]$ is the second (regularization) term in standard low-level energy (4.1). In this case, optimization in (4.16) is limited to the corners of the simplex where E_{PCE} reduces to the hard constraints over the seeds. In (4.16-4.18) we use robust metrics (4.25). The overall method is summarized in Algorithm 3.

One natural baseline for Algorithm 3 is a standard method based on stochastic gradient descent (SGD) for regularized loss (4.8) proposed in [280], see Sec. 4.1.2. Indeed, $E^{\text{bl}}_{\text{Potts}}$ is a relaxation of E_{Potts}, as discussed in Sec. 4.1.1. Thus, (4.8) is a relaxation of (4.26). Algorithm 3 with combinatorial solver for $\tilde{E}$ in (4.26) can be seen as a *discrete* trust region approximation for (4.8). In general, our approach (4.16-4.18) allows other discrete or continuous solvers and/or other approximations $\tilde{E}$.

First, PCE-GD baseline is the standard SGD optimizing partial cross-entropy (4.6). It has been shown in [280, 279] that such approach outperforms more complex proposal (fake ground truth) generation methods such as [172]. Second, Grid-GD is the SGD over regularized loss (4.8) where the CRF neighbourhood is 8-grid. Third, Dense-GD is the approach of [280] that uses the common fully-connected (dense) Potts CRF of [153].

We use the ScribbleSup [172] annotations for Pascal VOC 2012 [84] dataset. ScribbleSup supplies scribbles, *i.e.* a small subset of image pixels ($\approx 3\%$) is labeled while the vast majority of pixels is left unlabeled.

4.4.1 Implementation details

In our experiments, we used DeeplabV3+ [55] with MobileNetV2 [244] as a backbone model.

Pretraining: We use the standard ImageNet1k [74] pretraining of the backbone models. In addition, before the optimization via Grid-GD (4.7) and Grid-TR (4.16-4.18) starts, the DeeplabV3+ models are pretrained by the PCE loss (4.6).

Meta-parameters: We train 60 epochs. We tuned the learning rates for all methods on the val set. Other meta-parameters for competitive methods were set as in the corresponding papers/code. The learning rate is polynomial with power 0.9, momentum is 0.9, batch size is 12.

Grid-TR Stage A (4.16)**:** The low-level solver[3] of the grid CRF is the α-expansion [29, 152, 27] with 8-grid neighbourhood system. The max number of α-expansion iterations is 5 achieving convergence in most cases. We restrict the set of labels to those present in the image. We amortize the STAGE A compute time by integrating it with data loading. The training is 1.3 times slower than Dense-GD.

Grid-TR Stage B (4.18)**:** To amortize the time consumed by the graph cuts, we perform $M = 5$ epochs of neural network weights updates (4.17) for each update of the segmentation variables (4.18). We use a global learning rate schedule spanning throughout iterations. See Algorithm 3.

4.4.2 Segmentation quality

The quantitative results of the weakly supervised training for semantic segmentation are presented in Figure 4.5 and Tab. 4.1. The results are presented with different levels of supervision varying from the clicks (denoted as length 0) to the full-length scribbles (denoted as length 1). Decreasing supervision results in degraded performance for all methods. We are interested to compare how different approaches perform at different levels of supervision. Our Grid-TR outperforms all the competitors at each level of supervision.

The examples of images and results shown in Fig. 4.4 demonstrate the advantages of our method, particularly w.r.t. edge alignment. Quantitatively, we evaluate the accuracy of semantic boundaries using standard trimaps [147, 153, 53, 185]. A trimap corresponds to a narrow band around the ground truth segment boundaries of varying width. An accuracy

[3] GCOv3.0: https://vision.cs.uwaterloo.ca/code/

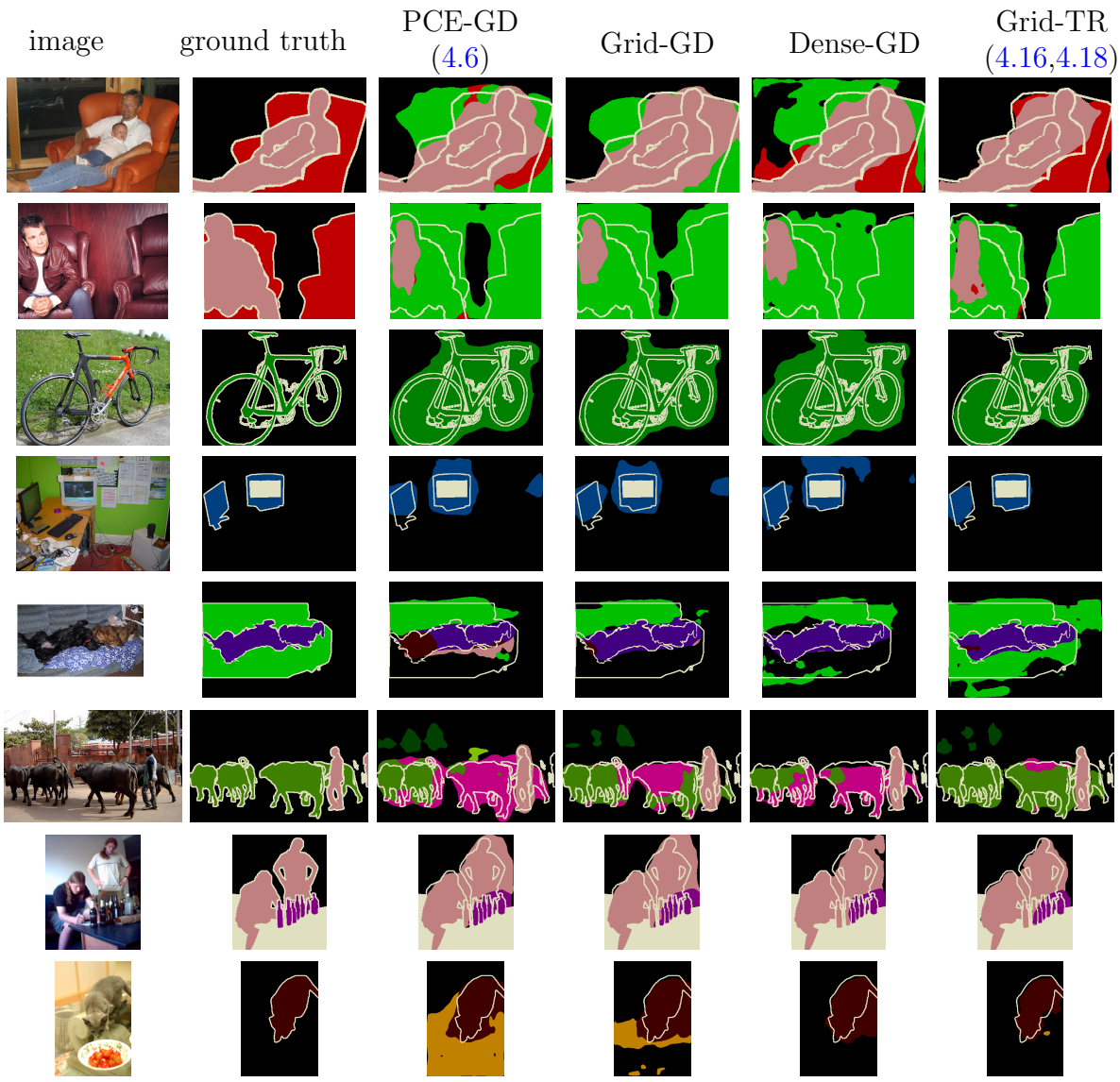


Figure 4.4: Examples of the full-scribble training results, see Table 4.1 and Figure 4.5. Note the better edge alignment of our Grid-TR.

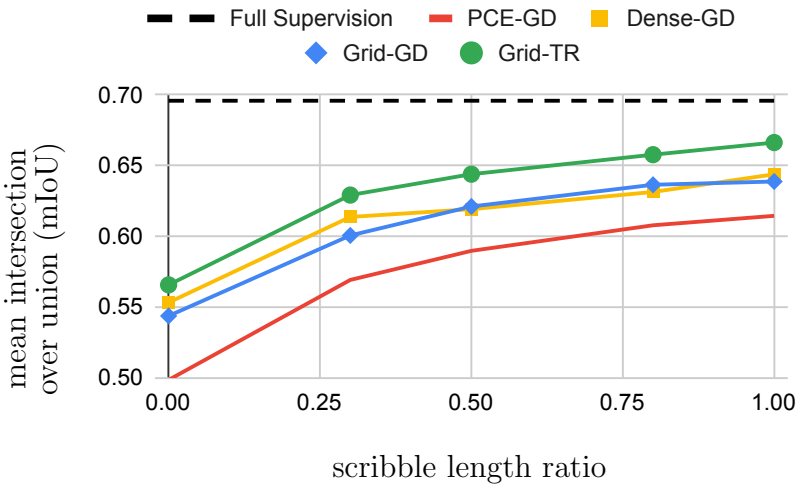


Figure 4.5: Segmentation performance on the val set of ScribbleSup [172, 84] using DeeplabV3+ [55] with MobileNetV2 [244] backbone. The supervision level varies horizontally, with 1 corresponding to the full scribbles. Our "Grid-TR" outperforms other competitors for all scribble lengths and provides a new state-of-the-art.

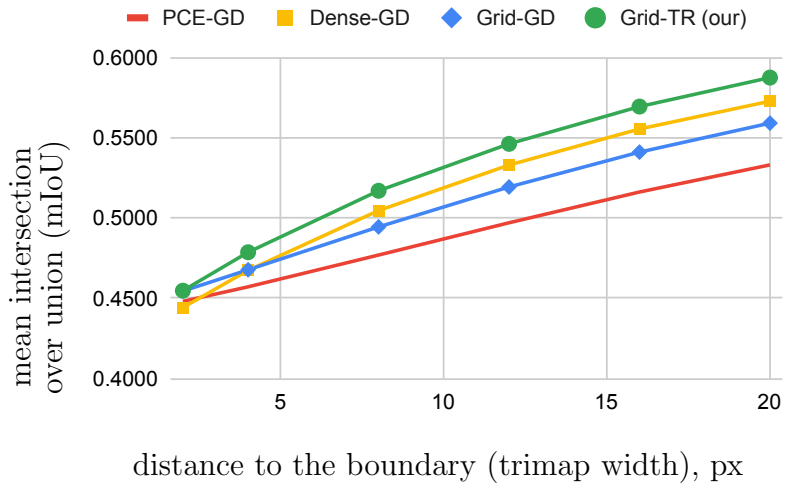


Figure 4.6: The quality of segment boundary alignment. The networks were trained on the full-length scribbles.

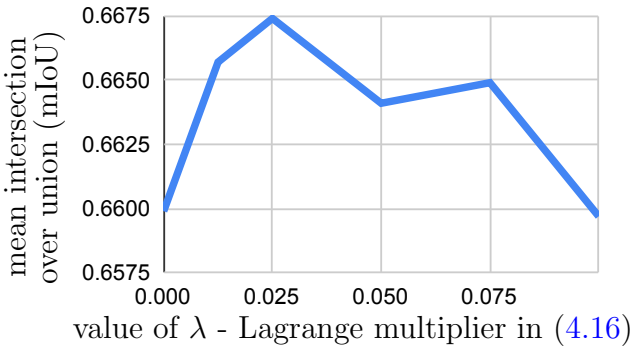


Figure 4.7: Empirical evaluation of Lagrange multiplier λ for the Trust Region term in (4.16): the plot shows how mobile-net training quality depends on λ. The context is the weakly-supervised semantic segmentation in Sec.4 with regularization loss E (using 8-grid Potts and full scribbles) based on our Trust Region chain rule with robust metric d_{B} in (4.18). For $\lambda = 0$ equation (4.16) generates fixed low-level segmentation proposals completely independent of the network. Then, the network overfits to mistakes in such proposals due to biases/weaknesses of the regularizer. As $\lambda \to \infty$, trust region becomes too small and our approach loses its advantages due to better (*e.g.* higher-order) approximation $\tilde{E}$ in (4.16). Conceptually speaking, it should get closer to the results of gradient descent, which uses basic first-order approximations.

measure, *e.g.* mIoU, is computed for pixels within each band. The results are shown in Fig. 4.6 where our approach demonstrates superior performance.

4.5 Discussion

4.5.1 On parameter λ in (4.16)

As discussed below equations (4.16) - (4.18), even for exact (global) solvers using $\tilde{E} = E$ in (4.16), the choice of $\lambda = 0$ could be sub-optimal, as demonstrated empirically here in Figure 4.7. As argued above, while $\lambda = 0$ with an exact solver may seem like a good approach to training $\min_\theta E(\phi(\boldsymbol{\theta}))$ suggesting globally optimal loss, empirically this leads to overfitting to mistakes or biases of the regularizer (e.g. the Potts model). One argument for $\lambda > 0$ discussed above is that this corresponds to the constrained optimization of (4.9) over the network manifold in $\mathbb{R}^n$. Such formulation of the training could be preferred

as constraining to neural networks can be seen as incorporation of the "deep priors", *e.g.* [289]. One can also argue that local minima of E inside the manifold of the network output in $\mathbb{R}^n$ may be preferable to the global optimum of E due to limitations of the basic (but solvable) regularizers.

Empirically, $\lambda = 0$ in (4.16) leads to a fixed set of proposals generated in a single run of stage A completely independent of the network. In contrast, $\lambda > 0$ leads to multiple distinct iterations of stage A where the network is in the feedback loop. Vice versa, instead of fixed proposals, for $\lambda > 0$ the network is exposed to a substantially larger set of solutions in stage B reduces overfitting.

Moreover, the objective in (4.16) can be motivated on its own merits independently of the objective in (4.9). It can be seen as a low-level segmentation objective that integrates class likelihoods produced by the neural network, replacing the basic likelihoods using low-level features, *e.g.* colors, as discussed in Sec.1.1. Iterations A/B can be seen as joint segmentation and model estimation, as typical for well-known low-level segmentation methods like Zhu-Yuille [330] Chan-Vese [46], or GrabCut [240]. The main difference is that our stages A/B use "deep" models. In contrast to standard methods [330, 46, 240] estimating model parameters for some standard class of probability distributions (*e.g.* GMM) over fixed low-level features like colors, we estimate deep models with millions of parameters that can be interpreted as learning high-level (semantic) features.

4.5.2 On discrete losses and decisions/activations

Our approach can train networks using discrete decisions/activations and losses defined over discrete domains. For example, (4.16)-(4.18) do not require that E is differentiable. In particular, (4.16) can be optimized over "hard" segmentations $\boldsymbol{s} \in \{0,1\}^{N\times K} \subset \Delta_K^N$ even if the network produces soft segmentations $\phi(\boldsymbol{\theta}) \in \Delta_K^N$, as long as d_{A} in (4.16) can measure a distance between discrete and continuous solutions, *e.g.* $\text{KL}(\boldsymbol{s}, \phi(\boldsymbol{\theta}))$ for one-hot and soft distributions. It is also possible to train the models with discrete decision functions $\mathbf{D}(l)$ such that $\phi(\boldsymbol{\theta}) = \mathbf{D}(l(\boldsymbol{\theta}))$ where l are logits. Then, all arguments in (4.16) are discrete. Optimization in (4.17) can be formulated over real-valued logits using d_{B} measuring a distance to subset $\{l \,|\, \mathbf{D}(l) = \boldsymbol{s}_{t+1}\} \subset \mathbb{R}^{N\times K}$.

Algorithm 3: Robust Trust Region for Potts model

Initialize model ϕ using ImageNet pretraining ;
Tune parameters $\boldsymbol{\theta}$ of model ϕ by optimizing PCE-GD loss (4.6) ;
Initialize γ with the base learning rate ;
repeat
  for *each image in dataset* **do**
    compute segmentation variable $\boldsymbol{s}$ via (4.16) using metric d_{A} in (4.25) and loss (4.26);
  end
  for M *epochs* **do**
    for *each image (batch) in dataset* **do**
      update the network parameters $\boldsymbol{\theta}$ using stochastic gradient descent for loss (4.17) with robust metric d_{B} in (4.25) ;
      update rate γ in accord with schedule;
    end
  end
until *required number of epochs is reached*;

scribble length	0	0.3	0.5	0.8	1
full supervision			0.70		
PCE-GD	0.50	0.57	0.59	0.61	0.61
Dense-GD	0.55	0.61	0.62	0.63	0.64
Grid-GD	0.54	0.60	0.62	0.64	0.64
Grid-TR (our)	**0.57**	**0.63**	**0.64**	**0.66**	**0.67**

Table 4.1: Results for ScribbleSup, see description in Figure 4.5.

Conclusions and Future Work

In conclusion, we briefly review our contributions and directions of future research.

Motivated by biases in the standard surface regularization, we presented in Chapter 2 a new low-level unsupervised thin structures extraction model, which directly regularizes the curvature of the centerline of the thin structures. Our unsupervised model can handle large multi-scale vessel extraction problems, where each volume can contain vessels with significantly varying sizes, from major veins/arteries with up to 40 voxels in diameter to nearly capillary vessels of subvoxel size with significant partial voluming artifacts, gaps, noise and outliers. The generality and flexibility of our model have been demonstrated by our follow-up work [322, 323] further extending it with useful priors, which significantly improve the quality of vessel bifurcation reconstruction.

In general, the large scale of the vessel extraction from current high-resolution 3D volumes [109, 122] necessitates minimal supervision approaches or unsupervised methods, such as ours. On the other hand, recent developments in computer vision and image analysis suggest the exploration of deep models for vessel extraction. Current deep models for thin structures mostly rely on full supervision, *e.g.* [197, 200, 201, 215]. Since full supervision can be a significant limitation for many large 3D problems, it is necessary to design practical unsupervised or weakly-supervised deep learning methods for extracting thin structures. For example, [214] proposes losses based on classical snake models to address imprecision of centerline annotations for *axons* and *dendrites* during training. It is not clear how such existing deep methods would work on complex multi-scale vessel data with a large variation of vessel sizes. It would be interesting to see such experiments. Also, [214] still uses full coverage of the centerlines, which is practically infeasible for large high-resolution vessel volumes in Chapter 2. We believe that it is necessary to develop unsupervised or weakly-supervised training techniques relying only on sparse partial annotations. For example, one can consider standard *multiple instance learning*, transductive/interactive techniques for extracting a complete structure from just a few labeled branches on a single volume, and unsupervised regularization losses. In particular, we believe that our curvature regulariza-

tion (Chapter 2), divergence [322], and confluence [323] constraints can play a significant role as priors for unsupervised and weakly-supervised deep learning methods for vessel extraction. Some ways of combining such priors and their efficient optimizers with deep learning are outlined in Chapter 4.

Chapter 3 studied popular graph (pairwise) clustering objectives, such as the normalized cut and kernel k-means, and theoretically characterized the density biases and conditions. Our theoretical insight inspired the density equalization principle as a solution to the bias. Our density equalization solutions lead to a surprising discovery of the convergence of the kernel clustering. That is, under the condition of equalized density, many kernel based criteria (such as the normalized cut, kernel k-means, and average cut) approximate each other. Future work may focus on developing theoretical characterization of the connection of kernel clustering to the Gini impurity (3.21), needed for our result. Another possible direction is the study of implications to the methods that learn kernels.

Finally, Chapter 4 presented a new optimization algorithm, higher-order back-prop, that allows using efficient low-level solvers in deep learning in the context of the weakly supervised semantic segmentation. Further research is needed to develop methods that require even less supervision. The importance of unsupervised and weakly supervised deep learning is predicated by the fact that obtaining the full annotations is prohibitively expensive for some applications, such as our large-scale vessel extraction from 3D in Chapter 2. We believe that the low-level objectives, designed to deal with the lack of supervision on a single image, and their efficient solvers will continue playing a significant role in weakly-supervised and unsupervised deep methods. Our optimization is general, and can help incorporate efficient low-level solvers inside network training. For example, one interesting research direction would explore the application of our method to discrete decisions/activations defined over discrete domains, such as binarized networks.

References

[1] Andrew Adams, Jongmin Baek, and Myers Abraham Davis. Fast high-dimensional filtering using the permutohedral lattice. *Computer Graphics Forum*, 29(2):753–762, 2010.

[2] Sameer Agarwal, Keir Mierle, and Others. Ceres solver. http://ceres-solver.org.

[3] Tao D Alter and Ronen Basri. Extracting salient curves from images: An analysis of the saliency network. *IJCV*, 27(1):51–69, 1998.

[4] Luis Alvarez, Pierre-Louis Lions, and Jean-Michel Morel. Image selective smoothing and edge detection by nonlinear diffusion. ii. *SIAM Journal on numerical analysis*, 29(3):845–866, 1992.

[5] Nikita Araslanov and Stefan Roth. Single-stage semantic segmentation from image labels. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[6] Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 33(5):898–916, 2011.

[7] Pablo Arbeláez, Bharath Hariharan, Chunhui Gu, Saurabh Gupta, Lubomir Bourdev, and Jitendra Malik. Semantic segmentation using regions and parts. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3378–3385. IEEE, 2012.

[8] Ben Athiwaratkun, Marc Finzi, Pavel Izmailov, and Andrew Gordon Wilson. There are many consistent explanations of unlabeled data: Why you should average. In *International Conference on Learning Representations (ICLR)*, 2018.

[9] Stephen R Aylward and Elizabeth Bullitt. Initialization, noise, singularities, and scale in height ridge traversal for tubular object centerline extraction. *IEEE transactions on medical imaging*, 21(2):61–75, 2002.

[10] Jimmy Ba, Roger Grosse, and James Martens. Distributed second-order optimization using kronecker-factored approximations. In *International Conference on Learning Representations (ICLR)*, 2017.

[11] Francis Bach and Michael Jordan. Learning spectral clustering. *Advances in Neural Information Processing Systems*, 16:305–312, 2003.

[12] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 39(12):2481–2495, 2017.

[13] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[14] Loris Bazzani, Alessandra Bergamo, Dragomir Anguelov, and Lorenzo Torresani. Self-taught object localization with deep networks. In *2016 IEEE winter conference on applications of computer vision (WACV)*, pages 1–9. IEEE, 2016.

[15] Ismail Ben Ayed, Lena Gorelick, and Yuri Boykov. Auxiliary cuts for general classes of higher order functionals. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1304–1311, 2013.

[16] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine Learning*, 79(1):151–175, May 2010.

[17] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

[18] Fethallah Benmansour and Laurent D Cohen. Tubular structure segmentation based on minimal path method and anisotropic enhancement. *IJCV*, 92(2):192–210, 2011.

[19] Christopher M Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[20] Michael J Black and Padmanabhan Anandan. A framework for the robust estimation of optical flow. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 231–236. IEEE, 1993.

[21] Andrew Blake and Andrew Zisserman. *Visual Reconstruction*. Cambridge, 1987.

[22] Endre Boros, PL Hammer, and X Sun. Network flows and minimization of quadratic pseudo-boolean functions. Technical report, Technical Report RRR 17-1991, RUTCOR, 1991.

[23] Aleksandar Botev, Hippolyt Ritter, and David Barber. Practical Gauss-Newton optimisation for deep learning. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 557–565, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. URL http://proceedings.mlr.press/v70/botev17a.html.

[24] Léon Bottou and Olivier Bousquet. The tradeoffs of large-scale learning. *Optimization for machine learning*, page 351, 2011.

[25] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[26] Yuri Boykov and Vladimir Kolmogorov. Computing geodesics and minimal surfaces via graph cuts. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 26–33. IEEE, 2003.

[27] 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 (PAMI)*, 26(9):1124–1137, 2004. ISSN 0162-8828.

[28] Yuri Boykov, Olga Veksler, and Ramin Zabih. Markov random fields with efficient approximations. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 648–655. IEEE, 1998.

[29] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 23(11):1222–1239, 2001.

[30] Yuri Boykov, Hossam Isack, Carl Olsson, and Ismail Ben Ayed. Volumetric bias in segmentation and reconstruction: Secrets and solutions. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1769–1777, 2015.

[31] Yuri Y Boykov and Marie-Pierre Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in nd images. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, volume 1, pages 105–112. IEEE, 2001.

[32] Kristian Bredies, Thomas Pock, and Benedikt Wirth. Convex relaxation of a class of vertex penalizing functionals. *Journal of Mathematical Imaging and Vision*, 47(3): 278–302, 2013.

[33] Leo Breiman. Technical note: Some properties of splitting criteria. *Machine Learning*, 24(1):41–47, 1996.

[34] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. CRC press, 1984.

[35] Claude R Brice and Claude L Fennema. Scene analysis using regions. *Artificial intelligence*, 1(3-4):205–226, 1970.

[36] John S. Bridle and Stephen J. Cox. Recnorm: Simultaneous normalisation and classification applied to speech recognition. In *Proceedings of the 1990 Conference on Advances in Neural Information Processing Systems 3*, NIPS-3, pages 234–240, San Francisco, CA, USA, 1990. Morgan Kaufmann Publishers Inc. ISBN 1-55860-184-8. URL http://dl.acm.org/citation.cfm?id=118850.118882.

[37] Thomas Bühler and Matthias Hein. Spectral clustering based on the graph p-laplacian. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 81–88, 2009.

[38] Richard H Byrd, Samantha L Hansen, Jorge Nocedal, and Yoram Singer. A stochastic quasi-newton method for large-scale optimization. *SIAM Journal on Optimization*, 26(2):1008–1031, 2016.

[39] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, (6):679–698, 1986.

[40] Miguel Carreira-Perpinan and Weiran Wang. Distributed optimization of deeply nested systems. In *Artificial Intelligence and Statistics*, pages 10–19. PMLR, 2014.

[41] Vicent Caselles, Ron Kimmel, and Guillermo Sapiro. Geodesic active contours. *International Journal of Computer Vision (IJCV)*, 22(1):61–79, 1997.

[42] Antonin Chambolle and Thomas Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145, 2011.

[43] Antonin Chambolle, Daniel Cremers, and Thomas Pock. A convex approach to minimal partitions. *SIAM Journal on Imaging Sciences*, 5(4):1113–1158, 2012.

[44] Tony Chan, S Esedoglu, and M Nikolova. Algorithms for finding global minimizers of image segmentation and denoising models. *SIAM journal on applied mathematics*, 66(5):1632–1648, 2006.

[45] Tony F Chan and Jianhong Shen. Nontexture inpainting by curvature-driven diffusions. *Journal of Visual Communication and Image Representation*, 12(4):436–449, 2001.

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

[47] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5410–5418, 2018.

[48] Ken Chatfield, Victor S Lempitsky, Andrea Vedaldi, and Andrew Zisserman. The devil is in the details: an evaluation of recent feature encoding methods. In *BMVC*, volume 2, page 8, 2011.

[49] Jeff Cheeger. A lower bound for the smallest eigenvalue of the laplacian. *Problems in Analysis, R.C. Gunning, ed.*, pages 195–199, 1970.

[50] Da Chen, Jean-Marie Mirebeau, and Laurent D Cohen. Global minimum for a finsler elastica minimal path approach. *International Journal of Computer Vision*, 122(3): 458–483, 2017.

[51] Gong Chen and Marc Teboulle. Convergence analysis of a proximal-like minimization algorithm using bregman functions. *SIAM Journal on Optimization*, 3(3):538–543, 1993.

[52] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. In *International Conference on Learning Representations (ICLR)*, 2015.

[53] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *arXiv:1606.00915*, 2016.

[54] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[55] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018.

[56] Radha Chitta, Rong Jin, Timothy Havens, and Anil Jain. Scalable kernel clustering: Approximate kernel k-means. In *KDD*, pages 895–903, 2011.

[57] Peter Clifford and John Hammersley. Markov fields on finite graphs and lattices. 1971.

[58] James Clough, Nicholas Byrne, Ilkay Oksuz, Veronika A. Zimmer, Julia A. Schnabel, and Andrew King. A topological loss function for deep-learning based image segmentation using persistent homology. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, pages 1–1, 2020. doi: 10.1109/TPAMI.2020.3013679.

[59] Guy Barrett Coleman and Harry C Andrews. Image segmentation by clustering. *Proceedings of the IEEE*, 67(5):773–785, 1979.

[60] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2002.

[61] 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/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[62] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. *Introduction to algorithms*. MIT press, 2009.

[63] Camille Couprie, Leo Grady, Laurent Najman, and Hugues Talbot. Power watershed: A unifying graph-based optimization framework. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 33(7):1384–1399, 2010.

[64] Matthieu Courbariaux, Itay Hubara, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks: Training deep neural networks with weights and activations constrained to +1 or -1. *arXiv preprint arXiv:1602.02830*, 2016.

[65] Trevor Cox and Michael Cox. *Multidimensional scaling*. CRC Press, 2000.

[66] Daniel Cremers, Mikael Rousson, and Rachid Deriche. A review of statistical approaches to level set segmentation: integrating color, texture, motion and shape. *International Journal of Computer Vision (IJCV)*, 72(2):195–215, 2007.

[67] Antonio Criminisi and Jamie Shotton. *Decision forests for computer vision and medical image analysis*. Springer Science & Business Media, 2013.

[68] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 764–773, 2017.

[69] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 886–893. Ieee, 2005.

[70] Hervé Delingette. On smoothness measures of active contours and surfaces. In *Proceedings IEEE Workshop on Variational and Level Set Methods in Computer Vision*, pages 43–50. IEEE, 2001.

[71] Andrew Delong and Yuri Boykov. Globally optimal segmentation of multi-region objects. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 285–292. IEEE, 2009.

[72] Andrew Delong, Anton Osokin, Hossam Isack, and Yuri Boykov. Fast Approximate Energy Minization with Label Costs. *International Journal of Computer Vision (IJCV)*, 96(1):1–27, January 2012.

[73] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society*, pages 1–38, 1977.

[74] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255. Ieee, 2009.

[75] Thomas Deschamps and Laurent D. Cohen. Fast extraction of minimal paths in 3d images and applications to virtual endoscopy. *Medical Image Analysis*, 5(4):281 – 299, 2001.

[76] Inderjit Dhillon, Yuqiang Guan, and Brian Kulis. Kernel k-means, spectral clustering and normalized cuts. In *KDD*, 2004.

[77] William E Donath and Alan J Hoffman. Lower bounds for the partitioning of graphs. pages 437–442, 2003.

[78] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

[79] Richard O Duda and Peter E Hart. Use of the hough transformation to detect lines and curves in pictures. *Communications of the ACM*, 15(1):11–15, 1972.

[80] Richard O Duda, Peter E Hart, and David G Stork. *Pattern classification and scene analysis*, volume 3. Wiley New York, 1973.

[81] Jack Edmonds. Submodular functions, matroids, and certain polyhedra. *Combinatorial structures and their applications*, pages 69–87, 1970.

[82] Noha Youssry El-Zehiry and Leo Grady. Fast global optimization of curvature. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3257–3264. IEEE, 2010.

[83] Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341(3):1, 2009.

[84] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective, 2015.

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

[86] Miroslav Fiedler. A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory. *Czechoslovak mathematical journal*, 25(4):619–633, 1975.

[87] M. A. T. Figueiredo and J. M. N. Leitao. A nonsmoothing approach to the estimation of vessel contours in angiograms. *IEEE Transactions on Medical Imaging*, 14(1):162–172, March 1995.

[88] David A Forsyth and Jean Ponce. *Computer vision: a modern approach*. Prentice Hall Professional Technical Reference, 2002.

[89] Chris Fraley and Adrian E Raftery. Model-Based Clustering, Discriminant Analysis, and Density Estimation. *Journal of the American Statistical Association*, 97(458): 611–631, 2002.

[90] Alejandro F Frangi, Wiro J Niessen, Koen L Vincken, and Max A Viergever. Multiscale vessel enhancement filtering. In *MICCAI'98*, pages 130–137. Springer, 1998.

[91] William T. Freeman and Edward H Adelson. The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, (9):891–906, 1991.

[92] Benoît Frénay and Michel Verleysen. Classification in the presence of label noise: a survey. *IEEE transactions on neural networks and learning systems*, 25(5):845–869, 2013.

[93] Thomas Frerix, Thomas Möllenhoff, Michael Moeller, and Daniel Cremers. Proximal backpropagation. In *International Conference on Learning Representations (ICLR)*, 2018.

[94] King-Sun Fu and JK Mui. A survey on image segmentation. *Pattern recognition*, 13 (1):3–16, 1981.

[95] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *J. Mach. Learn. Res.*, 17(1):2096–2030, January 2016. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=2946645.2946704.

[96] Yunhe Gao, Mu Zhou, and Dimitris Metaxas. Utnet: A hybrid transformer architecture for medical image segmentation. *arXiv preprint arXiv:2107.00781*, 2021.

[97] Stuart Geman and Donald Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 6(6):721–741, 1984.

[98] Stuart Geman, Elie Bienenstock, and René Doursat. Neural networks and the bias/variance dilemma. *Neural computation*, 4(1):1–58, 1992.

[99] Mark Girolami. Mercer kernel-based clustering in feature space. *IEEE Trans. Neural Networks*, 13(3):780–784, 2002.

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

[101] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 270–279, 2017.

[102] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J Brostow. Digging into self-supervised monocular depth estimation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 3828–3838, 2019.

[103] Germán González, François Fleuret, and Pascal Fua. Automated delineation of dendritic networks in noisy image stacks. In *European Conference on Computer Vision*, pages 214–227. Springer, 2008.

[104] Lena Gorelick, Yuri Boykov, Olga Veksler, Ismail Ben Ayed, and Andrew Delong. Submodularization for binary pairwise energies. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1154–1161, 2014.

[105] Lena Gorelick, Olga Veksler, Yuri Boykov, and Claudia Nieuwenhuis. Convexity shape prior for binary segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 39(2):258–271, 2016.

[106] Lena Gorelick, Olga Veksler, Yuri Boykov, and Claudia Nieuwenhuis. Convexity shape prior for binary segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 39(2):258–271, February 2017.

[107] John C Gower and Pierre Legendre. Metric and euclidean properties of dissimilarity coefficients. *Journal of classification*, 3(1):5–48, 1986.

[108] Leo Grady. Random walks for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 28(11):1768–1783, 2006.

[109] P.V. Granton, S.I. Pollmann, N.L. Ford, M. Drangova, and D.W. Holdsworth. Implementation of dual-and triple-energy cone-beam micro-ct for postreconstruction material decomposition. *Medical physics*, 35(11):5030–5042, 2008.

[110] Martin Grötschel, László Lovász, and Alexander Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1(2):169–197, 1981.

[111] Yuliang Guo and Benjamin Kimia. On evaluating methods for recovering image curve fragments. In *CVPRW*, 2012.

[112] Yuliang Guo, Naman Kumar, Maruthi Narayanan, and Benjamin Kimia. A multi-stage approach to curve extraction. In *Proceedings of the European conference on computer vision (ECCV)*, 2014.

[113] Gideon Guy and Gérard Medioni. Inferring global perceptual contours from local features. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 1993.

[114] Lars Hagen and Andrew B Kahng. New spectral methods for ratio cut partitioning and clustering. *IEEE transactions on computer-aided design of integrated circuits and systems*, 11(9):1074–1085, 1992.

[115] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034, 2015.

[116] Kaiming He, Ross Girshick, and Piotr Dollar. Rethinking imagenet pre-training. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[117] Stefan Heber, Rene Ranftl, and Thomas Pock. Approximate envelope minimization for curvature regularity. In *Proceedings of the European conference on computer vision (ECCV)*, 2012.

[118] Vu Hoang Hiep, Renaud Keriven, Patrick Labatut, and Jean-Philippe Pons. Towards high-resolution large-scale multi-view stereo. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1430–1437. IEEE, 2009.

[119] Geoffrey Hinton. Rmsprop: Divide the gradient by a running average of its recent magnitude. *Unpublished*.

[120] Geoffrey Hinton, Li Deng, Dong Yu, George Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Brian Kingsbury, et al. Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal processing magazine*, 29, 2012.

[121] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[122] David W Holdsworth and Michael M Thornton. Micro-ct in small animal and specimen imaging. *Trends in Biotechnology*, 20(8):S34–S39, 2002.

[123] Berthold KP Horn. The curve of least energy. *ACM Transactions on Mathematical Software (TOMS)*, 9(4):441–460, 1983.

[124] Berthold KP Horn and Brian G Schunck. Determining optical flow. In *Techniques and Applications of Image Understanding*, volume 281, pages 319–331. International Society for Optics and Photonics, 1981.

[125] Xiaoling Hu, Fuxin Li, Dimitris Samaras, and Chao Chen. Topology-preserving deep image segmentation. *Advances in Neural Information Processing Systems*, 32:5657–5668, 2019.

[126] Xiaoling Hu, Yusu Wang, Li Fuxin, Dimitris Samaras, and Chao Chen. Topology-aware segmentation using discrete morse theory. In *International Conference on Learning Representations*, 2020.

[127] Stephen S Intille and Aaron F Bobick. Disparity-space images and large occlusion stereo. In *Proceedings of the European conference on computer vision (ECCV)*, pages 179–186. Springer, 1994.

[128] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[129] Hossam Isack and Yuri Boykov. Energy-based geometric multi-model fitting. *International Journal of Computer Vision (IJCV)*, 97(2):123–147, April 2012.

[130] Hossam Isack, Lena Gorelick, Karin Ng, Olga Veksler, and Yuri Boykov. K-convexity shape priors for segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, Munich, Germany, September 2018.

[131] Hiroshi Ishikawa. Exact optimization for markov random fields with convex priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 25(10): 1333–1336, 2003.

[132] Ernst Ising. Contribution to the theory of ferromagnetism. *Z. Phys*, 31(1):253–258, 1925.

[133] Alan Julian Izenman. Review papers: Recent developments in nonparametric density estimation. *Journal of the American Statistical Association*, 86(413):205–224, 1991.

[134] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. In *34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018*, pages 876–885, 2018.

[135] Sadeep Jayasumana, Richard Hartley, Mathieu Salzmann, Hongdong Li, and Mehrtash Harandi. Kernel Methods on Riemannian Manifolds with Gaussian RBF Kernels. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, In press, 2015.

[136] George Kamberov and Gerda Kamberova. Ill-posed problems in surface and surface shape recovery. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2000.

[137] Jörg H Kappes, Bjoern Andres, Fred A Hamprecht, Christoph Schnörr, Sebastian Nowozin, Dhruv Batra, Sungwoong Kim, Bernhard X Kausler, Thorben Kröger, Jan Lellmann, et al. A comparative study of modern inference techniques for structured discrete energy minimization problems. *International Journal of Computer Vision (IJCV)*, 115(2):155–184, 2015.

[138] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision (IJCV)*, 1(4):321–331, 1988.

[139] Michael Kearns, Yishay Mansour, and Andrew Ng. An Information-Theoretic Analysis of Hard and Soft Assignment Methods for Clustering. In *Conf. on Uncertainty in Artificial Intelligence (UAI)*, August 1997.

[140] Michael Kearns, Yishay Mansour, and Andrew Y Ng. An information-theoretic analysis of hard and soft assignment methods for clustering. In *Learning in graphical models*, pages 495–520. Springer, 1998.

[141] Alex Kendall, Hayk Martirosyan, Saumitro Dasgupta, Peter Henry, Ryan Kennedy, Abraham Bachrach, and Adam Bry. End-to-end learning of geometry and context for deep stereo regression. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 66–75, 2017.

[142] Hoel Kervadec, Jose Dolz, Shanshan Wang, Eric Granger, and Ismail Ben Ayed. Bounding boxes for weakly supervised segmentation: Global constraints get close to full supervision. In *Proceedings of the Third Conference on Medical Imaging with Deep Learning*, volume 121 of *Proceedings of Machine Learning Research*, pages 365–381, Montreal, QC, Canada, 06–08 Jul 2020. PMLR.

[143] Anna Khoreva, Rodrigo Benenson, Jan Hosang, Matthias Hein, and Bernt Schiele. Simple does it: Weakly supervised instance and semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 876–885, 2017.

[144] Benjamin B Kimia, Allen R Tannenbaum, and Steven W Zucker. Shapes, shocks, and deformations i: the components of two-dimensional shape and the reaction-diffusion space. *IJCV*, 15(3):189–224, 1995.

[145] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[146] Morris Kline. *Calculus: an intuitive and physical approach*. Courier Corporation, 1998.

[147] Pushmeet Kohli, Philip HS Torr, et al. Robust higher order potentials for enforcing label consistency. *International Journal of Computer Vision (IJCV)*, 82(3):302–324, 2009.

[148] Alexander Kolesnikov and Christoph H Lampert. Seed, expand and constrain: Three principles for weakly-supervised image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 695–711. Springer, 2016.

[149] Vladimir Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 28(10):1568–1583, 2006.

[150] Vladimir Kolmogorov and Ramin Zabih. Computing visual correspondence with occlusions using graph cuts. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 508–515. IEEE, 2001.

[151] Vladimir Kolmogorov and Ramin Zabih. Multi-camera scene reconstruction via graph cuts. In *Proceedings of the European conference on computer vision (ECCV)*, pages 82–96. Springer, 2002.

[152] Vladimir Kolmogorov and Ramin Zabin. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 26(2):147–159, 2004.

[153] Philipp Krahenbuhl and Vladlen Koltun. Efficient inference in fully connected CRFs with Gaussian edge potentials. In *NIPS*, 2011.

[154] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[155] Anders Krogh and John A Hertz. A simple weight decay can improve generalization. In *Advances in neural information processing systems*, pages 950–957, 1992.

[156] Charles D Kuglin. The phase correlation image alignment methed. In *Proc. Int. Conference Cybernetics Society*, pages 163–165, 1975.

[157] M Pawan Kumar, PHS Ton, and Andrew Zisserman. Obj cut. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 18–25. IEEE, 2005.

[158] M Pawan Kumar, Vladimir Kolmogorov, and Philip HS Torr. An analysis of convex relaxations for map estimation of discrete mrfs. *Journal of machine learning research*, 10(Jan):71–106, 2009.

[159] Hammer P. L. and S. Rudeanu. *Boolean Methods in Operations Research and Related Areas*. Springer, 1968.

[160] John D Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289, 2001.

[161] Jan Larsen, L Nonboe, Mads Hintz-Madsen, and Lars Kai Hansen. Design of robust neural network classifiers. In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98 (Cat. No. 98CH36181)*, volume 2, pages 1205–1208. IEEE, 1998.

[162] Max W K Law and Albert C S Chung. Three dimensional curvilinear structure detection using optimally oriented flux. In *Proceedings of the European conference on computer vision (ECCV)*, pages 368–382. Springer, 2008.

[163] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553): 436, 2015.

[164] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[165] Victor Lempitsky, Pushmeet Kohli, Carsten Rother, and Toby Sharp. Image segmentation with a bounding box prior. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 277–284, 2009.

[166] Thomas Leung and Jitendra Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *International Journal of Computer Vision (IJCV)*, 43(1):29–44, 2001.

[167] Kenneth Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics 2*, pages 164–168, 1944.

[168] Gang Li and Steven W Zucker. Differential geometric inference in surface stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 32(1): 72–86, 2010.

[169] Hua Li and Anthony Yezzi. Vessels as 4-d curves: Global minimal 4-d paths to extract 3-d tubular surfaces and centerlines. *IEEE transactions on medical imaging*, 26(9):1213–1223, 2007.

[170] Stan Z Li. *Markov random field modeling in image analysis*. Springer Science & Business Media, 2009.

[171] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 136–144, 2017.

[172] Di Lin, Jifeng Dai, Jiaya Jia, Kaiming He, and Jian Sun. Scribblesup: Scribble-supervised convolutional networks for semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3159–3167, 2016.

[173] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Proceedings of the European conference on computer vision (ECCV)*, pages 740–755. Springer, 2014.

[174] James Lingoes. Some boundary conditions for a monotone analysis of symmetric matrices. *Psychometrika*, 1971.

[175] Zili Liu, David W Jacobs, and Ronen Basri. The role of convexity in perceptual completion: Beyond good continuation. *Vision research*, 39(25):4244–4257, 1999.

[176] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[177] David G Lowe. Object recognition from local scale-invariant features. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 1150–1157. Ieee, 1999.

[178] Bruce D Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th international joint conference on Artificial intelligence-Volume 2*, pages 674–679, 1981.

[179] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5188–5196, 2015.

[180] Jitendra Malik, Serge Belongie, Jianbo Shi, and Thomas Leung. Textons, contours and regions: Cue integration in image segmentation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 918–925. IEEE, 1999.

[181] Pascal Mamassian and Michael S Landy. Observer biases in the 3d interpretation of line drawings. *Vision research*, 38(18):2817–2832, 1998.

[182] Naresh Manwani and PS Sastry. Noise tolerance under risk minimization. *IEEE transactions on cybernetics*, 43(3):1146–1151, 2013.

[183] Dmitrii Marin and Yuri Boykov. Robust trust region for weakly supervised segmentation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, ©2021 IEEE.

[184] Dmitrii Marin, Yuchen Zhong, Maria Drangova, and Yuri Boykov. Thin structure estimation with curvature regularization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 397–405, December ©2015 IEEE. doi: 10.1109/ICCV.2015.53.

[185] Dmitrii Marin, Zijian He, Peter Vajda, Priyam Chatterjee, Sam Tsai, Fei Yang, and Yuri Boykov. Efficient segmentation: Learning downsampling near semantic boundaries. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2131–2141, ©2019 IEEE.

[186] Dmitrii Marin, Meng Tang, Ismail Ben Ayed, and Yuri Boykov. Kernel clustering: Density biases and solutions. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 41(01):136–147, ©2019 IEEE.

[187] Dmitrii Marin, Meng Tang, Ismail Ben Ayed, and Yuri Boykov. Beyond gradient descent for regularized segmentation losses. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, California, June ©2019 IEEE.

[188] James Martens and Roger Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In *International conference on machine learning*, pages 2408–2417. PMLR, 2015.

[189] David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 416–423, 2001.

[190] David R Martin, Charless C Fowlkes, and Jitendra Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 26(5):530–549, 2004.

[191] Bernard Martinet. Brief communication. Regularization of variational inequalities by successive approximations. *French journal of informatics and operational research. Red Series*, 4(R3):154–158, 1970.

[192] Ofer Matan, Christopher JC Burges, Yann LeCun, and John S Denker. Multi-digit recognition using a space displacement neural network. In *Advances in neural information processing systems*, pages 488–495, 1992.

[193] Odyssée Merveille, Hugues Talbot, Laurent Najman, and Nicolas Passat. Curvilinear structure analysis by ranking the orientation responses of path operators. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 40(2):304–317, 2017.

[194] Odyssée Merveille, Benoît Naegel, Hugues Talbot, and Nicolas Passat. nD variational restoration of curvilinear structures with prior-based directional regularization. *IEEE Transactions on Image Processing*, 28(8):3848–3859, 2019.

[195] Fernand Meyer and Serge Beucher. Morphological segmentation. *Journal of visual communication and image representation*, 1(1):21–46, 1990.

[196] MA Minsky and S Pappert. Project MAC Progress Report IV, 1967.

[197] Volodymyr Mnih and Geoffrey E Hinton. Learning to detect roads in high-resolution aerial images. In *European Conference on Computer Vision*, pages 210–223. Springer, 2010.

[198] Parya MomayyezSiahkal and Kaleem Siddiqi. 3d stochastic completion fields for mapping connectivity in diffusion mri. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 35(4):983–995, 2013.

[199] Stefano Moriconi, Maria A Zuluaga, H Rolf Jäger, Parashkev Nachev, Sébastien Ourselin, and M Jorge Cardoso. Inference of cerebrovascular topology with geodesic minimum spanning trees. *IEEE transactions on medical imaging*, 38(1):225–239, 2018.

[200] Agata Mosinska, Pablo Marquez-Neila, Mateusz Koziński, and Pascal Fua. Beyond the pixel-wise loss for topology-aware delineation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3136–3145, 2018.

[201] Agata Mosinska, Mateusz Koziński, and Pascal Fua. Joint segmentation and path classification of curvilinear structures. *IEEE transactions on pattern analysis and machine intelligence*, 42(6):1515–1521, 2019.

[202] Theodore S Motzkin and Ernst G Straus. Maxima for graphs and a new proof of a theorem of turán. *Canad. J. Math*, 17(4):533–540, 1965.

[203] K-R Muller, Sebastian Mika, Gunnar Ratsch, Koji Tsuda, and Bernhard Scholkopf. An introduction to kernel-based learning algorithms. *IEEE transactions on neural networks*, 12(2):181–201, 2001.

[204] David Bryant Mumford and Jayant Shah. Optimal approximations by piecewise smooth functions and associated variational problems. *Communications on pure and applied mathematics*, 1989.

[205] Yurii E Nesterov. Inexact accelerated high-order proximal-point methods. Technical report, CORE, 2020.

[206] Claudia Nieuwenhuis, Eno Toeppe, Lena Gorelick, Olga Veksler, and Yuri Boykov. Efficient squared curvature. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4098–4105, 2014.

[207] Evgenii Nikishin, Pavel Izmailov, Ben Athiwaratkun, Dmitrii Podoprikhin, Timur Garipov, Pavel Shvechikov, Dmitry Vetrov, and Andrew Gordon Wilson. Improving stability in deep reinforcement learning with weight averaging. In *Uncertainty in artificial intelligence workshop on uncertainty in Deep learning*, 2018.

[208] Ronald B Ohlander. Analysis of natural scenes. Technical report, Carnegie-Mellon univ, Pittsburgh PA, Dept of computer science, 1975.

[209] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2017. doi: 10.23915/distill.00007. https://distill.pub/2017/feature-visualization.

[210] Aude Oliva and Antonio Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision (IJCV)*, 42(3):145–175, 2001.

[211] Carl Olsson and Yuri Boykov. Curvature-based regularization for surface approximation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1576–1583. IEEE, 2012.

[212] Carl Olsson, Johannes Ulén, and Yuri Boykov. In defense of 3d-label stereo. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1730–1737. IEEE, 2013.

[213] Carl Olsson, Johannes Ulén, Yuri Boykov, and Vladimir Kolmogorov. Partial enumeration and curvature regularization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2936–2943. IEEE, 2013.

[214] Doruk Oner, Leonardo Citraro, Mateusz Koziński, and Pascal Fua. Adjusting the ground truth annotations for connectivity-based learning to delineate. *arXiv preprint arXiv:2112.02781*, 2021.

[215] Doruk Oner, Mateusz Kozinski, Leonardo Citraro, Nathan C. Dadap, Alexandra G. Konings, and Pascal Fua. Promoting connectivity of network-like structures by enforcing region separation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2021. doi: 10.1109/TPAMI.2021.3074366.

[216] Doruk Oner, Mateusz Kozinski, Leonardo Citraro, Nathan C Dadap, Alexandra G Konings, and Pascal Fua. Promoting connectivity of network-like structures by enforcing region separation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2021.

[217] Maxime Oquab, Léon Bottou, Ivan Laptev, and Josef Sivic. Is object localization for free?-weakly-supervised learning with convolutional neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 685–694, 2015.

[218] Kazuki Osawa, Yohei Tsuji, Yuichiro Ueno, Akira Naruse, Rio Yokota, and Satoshi Matsuoka. Large-scale distributed second-order optimization using kronecker-factored approximate curvature for deep convolutional neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12359–12367, 2019.

[219] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, 9(1):62–66, 1979.

[220] Jiahao Pang, Wenxiu Sun, Jimmy SJ Ren, Chengxi Yang, and Qiong Yan. Cascade residual learning: A two-stage convolutional neural network for stereo matching. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 887–895, 2017.

[221] George Papandreou, Liang-Chieh Chen, Kevin P Murphy, and Alan L Yuille. Weakly- and semi-supervised learning of a deep convolutional network for semantic image segmentation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1742–1750, 2015.

[222] Seymour A Papert. The summer vision project. 1966.

[223] Pierre Parent and Steven W Zucker. Trace inference, curvature consistency, and curve detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 11:823–839, 1989.

[224] Giorgio Parisi. *Statistical field theory*, volume 4. Addison-Wesley New York, 1988.

[225] Deepak Pathak, Philipp Krahenbuhl, and Trevor Darrell. Constrained convolutional neural networks for weakly supervised segmentation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1796–1804, 2015.

[226] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1944–1952, 2017.

[227] Josh Patterson and Adam Gibson. *Deep learning: A practitioner's approach*. " O'Reilly Media, Inc.", 2017.

[228] Massimiliano Pavan and Marcello Pelillo. Dominant sets and pairwise clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 29(1): 167–172, 2007. doi: 10.1109/TPAMI.2007.250608.

[229] Judea Pearl. *Reverend Bayes on inference engines: A distributed hierarchical approach*. Cognitive Systems Laboratory, School of Engineering and Applied Science, University of California, Los Angeles, 1982.

[230] Barak A Pearlmutter. Fast exact multiplication by the hessian. *Neural computation*, 6(1):147–160, 1994.

[231] Hanchuan Peng, Fuhui Long, and Gene Myers. Automatic 3d neuron tracing using all-path pruning. *Bioinformatics*, 27(13):i239–i247, 2011.

[232] Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization*, 30(4):838–855, 1992.

[233] Judith MS Prewitt. Object enhancement and extraction. *Picture processing and Psychopictorics*, 10(1):15–19, 1970.

[234] Lynn Herman Quam. *Computer comparison of pictures*. Number 144. Department of Computer Science, Stanford University., 1971.

[235] Pradeep Ravikumar and John Lafferty. Quadratic programming relaxations for metric labeling and markov random field map estimation. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML â€™06, page 737â€“744, New York, NY, USA, 2006. Association for Computing Machinery. ISBN 1595933832. doi: 10.1145/1143844.1143937. URL https://doi.org/10.1145/1143844.1143937.

[236] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing.

[237] Frank Rosenblatt. *The perceptron, a perceiving and recognizing automaton Project Para*. Cornell Aeronautical Laboratory, 1957.

[238] Azriel Rosenfeld. Iterative methods in image analysis. *Pattern Recognition*, 10(3):181 – 187, 1978. ISSN 0031-3203. doi: https://doi.org/10.1016/0031-3203(78)90026-2. URL http://www.sciencedirect.com/science/article/pii/0031320378900262. The Proceedings of the IEEE Computer Society Conference.

[239] Volker Roth, Julian Laub, Motoaki Kawanabe, and Joachim Buhmann. Optimal cluster preserving embedding of nonmetric proximity data. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 25(12):1540—1551, 2003.

[240] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. "GrabCut" interactive foreground extraction using iterated graph cuts. *ACM transactions on graphics (TOG)*, 23(3):309–314, 2004.

[241] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[242] David Ruppert. A newton-raphson version of the multivariate robbins-monro procedure. *The Annals of Statistics*, 13(1):236–245, 1985.

[243] David Ruppert. Efficient estimations from a slowly convergent robbins-monro process. Technical report, Cornell University Operations Research and Industrial Engineering, 1988.

[244] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4510–4520, 2018.

[245] Thomas Schoenemann, Fredrik Kahl, and Daniel Cremers. Curvature regularity for region-based image segmentation and inpainting: A linear programming relaxation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Kyoto, 2009.

[246] Thomas Schoenemann, Fredrik Kahl, Simon Masnou, and Daniel Cremers. A linear framework for region-based image segmentation and inpainting involving curvature penalization. *IJCV*, 2012.

[247] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998.

[248] Nicol N Schraudolph. Fast curvature matrix-vector products for second-order gradient descent. *Neural computation*, 14(7):1723–1738, 2002.

[249] David W Scott. *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons, 1992.

[250] Thomas B Sebastian, Philip N Klein, and Benjamin B Kimia. Recognition of shapes by editing their shock graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2004.

[251] James Albert Sethian. *Level set methods and fast marching methods*, volume 3. Cambridge university press, 1999.

[252] Alexander Shekhovtsov, Pushmeet Kohli, and Carsten Rother. Curvature prior for mrf-based segmentation and shape inpainting. In *Joint DAGM (German Association for Pattern Recognition) and OAGM Symposium*, pages 41–51. Springer, 2012.

[253] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 22:888–905, 2000.

[254] Kaleem Siddiqi and Stephen Pizer. *Medial representations: mathematics, algorithms and applications*, volume 37. Springer Science & Business Media, 2008.

[255] Kaleem Siddiqi, Sylvain Bouix, Allen Tannenbaum, and Steven W Zucker. Hamilton-jacobi skeletons. *IJCV*, 48(3):215–231, 2002.

[256] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017.

[257] Bernard W Silverman. *Density estimation for statistics and data analysis*, volume 26. CRC press, 1986.

[258] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[259] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[260] Sudipta N Sinha and Marc Pollefeys. Multi-view reconstruction using photo-consistency and exact silhouette constraints: A maximum-flow formulation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, volume 1, pages 349–356. IEEE, 2005.

[261] Irwin Sobel. History and definition of the sobel operator. *Retrieved from the World Wide Web*, 1505, 2014.

[262] Irwin Sobel and Gary Feldman. A 3x3 isotropic gradient operator for image processing. 1968.

[263] James C Spall. Adaptive stochastic approximation by the simultaneous perturbation method. *IEEE transactions on automatic control*, 45(10):1839–1853, 2000.

[264] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.

[265] Petter Strandmark and Fredrik Kahl. Curvature regularization for curves and surfaces in a global optimization framework. In *EMMCVPR*, pages 205–218. Springer, 2011.

[266] BU-Qing Su and Ding-zhe Liu. *Computational geometry: curve and surface modeling*. Academic Press Professional, Inc., 1989.

[267] Hang Su, Varun Jampani, Deqing Sun, Orazio Gallo, Erik Learned-Miller, and Jan Kautz. Pixel-adaptive convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[268] Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. Training convolutional networks with noisy labels. In *International Conference on Learning Representations (ICLR)*, 2015.

[269] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147. PMLR, 2013.

[270] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations (ICLR)*, 2014.

[271] 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/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015.

[272] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[273] Richard Szeliski, Ramin Zabih, Daniel Scharstein, Olga Veksler, Vladimir Kolmogorov, Aseem Agarwala, Marshall Tappen, and Carsten Rother. *A Comparative Study of Energy Minimization Methods for Markov Random Fields*, pages 16–29. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. ISBN 978-3-540-33835-2. doi: 10.1007/11744047_2. URL http://dx.doi.org/10.1007/11744047_2.

[274] Amir Tamrakar and Benjamin B Kimia. No grouping left behind: From edges to curve fragments. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1–8. IEEE, 2007.

[275] Meng Tang, Lena Gorelick, Olga Veksler, and Yuri Boykov. Grabcut in one cut. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Sydney, Australia, December 2013.

[276] Meng Tang, Ismail Ben Ayed, and Yuri Boykov. Pseudo-bound optimization for binary energies. In *Proceedings of the European conference on computer vision (ECCV)*, pages 691–707. Springer, 2014.

[277] Meng Tang, Dmitrii Marin, Ismail Ben Ayed, and Yuri Boykov. Normalized Cut meets MRF. In *Proceedings of the European conference on computer vision (ECCV)*, Amsterdam, Netherlands, October 2016.

[278] Meng Tang, Dmitrii Marin, Ismail Ben Ayed, and Yuri Boykov. Kernel Cuts: MRF meets kernel and spectral clustering. In *arXiv:1506.07439*, Sept. 2016.

[279] Meng Tang, Abdelaziz Djelouah, Federico Perazzi, Yuri Boykov, and Christopher Schroers. Normalized Cut Loss for Weakly-supervised CNN Segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[280] Meng Tang, Federico Perazzi, Abdelaziz Djelouah, Ismail Ben Ayed, Christopher Schroers, and Yuri Boykov. On Regularized Losses for Weakly-supervised CNN Segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, 2018.

[281] Meng Tang, Dmitrii Marin, Ismail Ben Ayed, and Yuri Boykov. Kernel cuts: Kernel and spectral clustering meet regularization. *International Journal of Computer Vision (IJCV)*, 127(5):477–511, May 2019. ISSN 1573-1405. doi: 10.1007/s11263-018-1115-1. URL https://doi.org/10.1007/s11263-018-1115-1.

[282] Gavin Taylor, Ryan Burmeister, Zheng Xu, Bharat Singh, Ankit Patel, and Tom Goldstein. Training neural networks without gradients: A scalable admm approach. In *International conference on machine learning*, pages 2722–2731, 2016.

[283] George R. Terrell and David W. Scott. Variable kernel density estimation. *The Annals of Statistics*, 20(3):1236–1265, 1992. ISSN 00905364. URL http://www.jstor.org/stable/2242011.

[284] Carlo Tomasi and Roberto Manduchi. Bilateral filtering for gray and color images. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 839–846. IEEE, 1998.

[285] Andy Tsai, Anthony Yezzi, William Wells, Clare Tempany, Dewey Tucker, Ayres Fan, W Eric Grimson, and Alan Willsky. A shape-based approach to the segmentation of medical imagery using level sets. *IEEE transactions on medical imaging*, 22(2):137–154, 2003.

[286] Engin Turetken, German Gonzalez, Christian Blum, and Pascal Fua. Automated reconstruction of dendritic and axonal trees by global optimization with geometric priors. *Neuroinformatics*, 9(2-3):279–302, 2011.

[287] Engin Turetken, Carlos Becker, Przemyslaw Glowacki, Fethallah Benmansour, and Pascal Fua. Detecting irregular curvilinear structures in gray scale and color imagery using multi-directional oriented flux. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1553–1560, 2013.

[288] Engin Turetken, Fethallah Benmansour, Bjoern Andres, Przemyslaw Glowacki, Hanspeter Pfister, and Pascal Fua. Reconstructing curvilinear networks using path classifiers and integer programming. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 38(12):2515–2530, December 2016.

[289] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9446–9454, 2018.

[290] Vladimir Vapnik. *Statistical Learning Theory*. Wiley, 1998.

[291] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[292] Olga Veksler. Stereo correspondence by dynamic programming on a tree. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 384–390. IEEE, 2005.

[293] Olga Veksler. Efficient graph cut optimization for full crfs with quantized edges. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 42(4): 1005–1012, April 2020. ISSN 0162-8828. doi: 10.1109/TPAMI.2019.2906204. URL https://doi.org/10.1109/TPAMI.2019.2906204.

[294] 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 (IJCV)*, 50(3):271–293, 2002.

[295] Sara Vicente, Vladimir Kolmogorov, and Carsten Rother. Joint optimization of segmentation and appearance models. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2009.

[296] Luc Vincent and Pierre Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE Computer Architecture Letters*, 13(06):583–598, 1991.

[297] George Vogiatzis, Carlos Hernández Esteban, Philip HS Torr, and Roberto Cipolla. Multiview stereo via volumetric graph-cuts and occlusion robust photo-consistency. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 29(12): 2241–2246, 2007.

[298] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17 (4):395–416, 2007.

[299] Arthur G Wacker. *A cluster approach to finding spatial boundaries in multispectral imagery*. Laboratory for Applications of Remote Sensing, Purdue University, 1969.

[300] Martin J Wainwright, Tommi S Jaakkola, and Alan S Willsky. Map estimation via agreement on trees: message-passing and linear programming. *IEEE transactions on information theory*, 51(11):3697–3717, 2005.

[301] Abraham Wald. Contributions to the theory of statistical estimation and testing hypotheses. *The Annals of Mathematical Statistics*, 10(4):299–326, 1939.

[302] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *Proceedings of the European conference on computer vision (ECCV)*, pages 0–0, 2018.

[303] Jason Weston, Frédéric Ratle, Hossein Mobahi, and Ronan Collobert. Deep learning via semi-supervised embedding. In *Neural networks: Tricks of the trade*, pages 639–655. Springer, 2012.

[304] Joan S Weszka and Azriel Rosenfeld. Threshold evaluation techniques. *IEEE Transactions on systems, man, and cybernetics*, 8(8):622–629, 1978.

[305] Lance R Williams and David W Jacobs. Stochastic completion fields: A neural model of illusory contour shape and salience. *Neural Computation*, 9(4):837–858, 1997.

[306] Oliver Woodford, Philip Torr, Ian Reid, and Andrew Fitzgibbon. Global stereo reconstruction under second-order smoothness priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 31(12):2115–2128, 2009.

[307] SJ Wright and John Norman Holt. An inexact levenberg-marquardt method for large sparse nonlinear least squares. *The Journal of the Australian Mathematical Society. Series B. Applied Mathematics*, 26(04):387–403, 1985.

[308] Zhenyu Wu and Richard Leahy. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 15(11):1101–1113, 1993.

[309] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

[310] Jun Xie, Ting Zhao, Tzumin Lee, Eugene Myers, and Hanchuan Peng. Automatic neuron tracing in volumetric microscopy images with anisotropic path searching. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 472–479. Springer, 2010.

[311] Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1395–1403, 2015.

[312] Guandao Yang, Tianyi Zhang, Polina Kirichenko, Junwen Bai, Andrew Gordon Wilson, and Chris De Sa. Swalp: Stochastic weight averaging in low precision training. In *International Conference on Machine Learning*, pages 7015–7024. PMLR, 2019.

[313] Anthony Yezzi, Satyanad Kichenassamy, Arun Kumar, Peter Olver, and Allen Tannenbaum. A geometric snake model for segmentation of medical imagery. *IEEE Transactions on medical imaging*, 16(2):199–209, 1997.

[314] Zhichao Yin and Jianping Shi. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1983–1992, 2018.

[315] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.

[316] Stella Yu and Jianbo Shi. Multiclass spectral clustering. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2003.

[317] Alan L Yuille and Chenxi Liu. Deep nets: What have they ever done for vision? Technical report, Center for Brains, Minds and Machines (CBMM), 2018.

[318] Lihi Zelnik-Manor and Pietro Perona. Self-tuning spectral clustering. In *Advances in NIPS*, pages 1601–1608, 2004.

[319] Huangying Zhan, Ravi Garg, Chamara Saroj Weerasekera, Kejie Li, Harsh Agarwal, and Ian Reid. Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 340–349, 2018.

[320] Rong Zhang and Alexander Rudnicky. A large scale clustering scheme for kernel k-means. In *Pattern Recognition, 2002.*, volume 4, pages 289–292, 2002.

[321] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image super-resolution. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2472–2481, 2018.

[322] Zhongwen Zhang, Dmitrii Marin, Egor Chesakov, Marc Moreno Maza, Maria Drangova, and Yuri Boykov. Divergence prior and vessel-tree reconstruction. In *IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, California, June ©2019 IEEE.

[323] Zhongwen Zhang, Dmitrii Marin, Maria Drangova, and Yuri Boykov. Confluent vessel trees with accurate bifurcations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9573–9582, ©2021 IEEE.

[324] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2881–2890, 2017.

[325] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1529–1537, 2015.

[326] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip HS Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6881–6890, 2021.

[327] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2921–2929, 2016.

[328] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe. Unsupervised learning of depth and ego-motion from video. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1851–1858, 2017.

[329] Y-T Zhou, Rama Chellappa, Aseem Vaid, and B Keith Jenkins. Image restoration using a neural network. *IEEE transactions on acoustics, speech, and signal processing*, 36(7):1141–1151, 1988.

[330] Song Chun Zhu and Alan Yuille. Region competition: Unifying snakes, region growing, and bayes/mdl for multiband image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 18(9):884–900, 1996.