Learning Human Cues in Image Segmentation

by

Jason Chang

Submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Electrical Engineering and Computer Science
at the Massachusetts Institute of Technology

June 2009

Signature of Author: _____

Department of Electrical Engineering and Computer Science
May 12, 2009

Certified by: _____

John W. Fisher III, Principal Research Scientist of EECS
Thesis Supervisor

Accepted by: _____

Terry P. Orlando, Professor of Electrical Engineering
Chair, Department Committee on Graduate Students

Learning Human Cues in Image Segmentation

by Jason Chang

Submitted to the Department of Electrical Engineering
and Computer Science on May 12, 2009
in Partial Fulfillment of the Requirements for the Degree
of Doctor of Philosophy in Electrical Engineering and Computer Science

Abstract

In this dissertation we investigate the problem of reasoning over evolving structures which describe the dependence among multiple, possibly vector-valued, time-series. Such problems arise naturally in variety of settings. Consider the problem of object interaction analysis. Given tracks of multiple moving objects one may wish to describe if and how these objects are interacting over time. Alternatively, consider a scenario in which one observes multiple video streams representing participants in a conversation. Given a single audio stream, one may wish to determine with which video stream the audio stream is associated as a means of indicating who is speaking at any point in time. Both of these problems can be cast as inference over dependence structures.

In the absence of training data, such reasoning is challenging for several reasons. If one is solely interested in the structure of dependence as described by a graphical model, there is the question of how to account for unknown parameters. Additionally, the set of possible structures is generally super-exponential in the number of time series. Furthermore, if one wishes to reason about structure which varies over time, the number of structural sequences grows exponentially with the length of time being analyzed.

We present tractable methods for reasoning in such scenarios. We consider two approaches for reasoning over structure while treating the unknown parameters as nuisance variables. First, we develop a generalized likelihood approach in which point estimates of parameters are used in place of the unknown quantities. We explore this approach in scenarios in which one considers a small enumerated set of specified structures. Second, we develop a Bayesian approach and present a conjugate prior on the parameters and structure of a model describing the dependence among time-series. This allows for Bayesian reasoning over structure while integrating over parameters. The modular nature of the prior we define allows one to reason over a super-exponential number of structures in exponential-time in general. Furthermore, by imposing simple local or global structural constraints we show that one can reduce the exponential-time com-

plexity to polynomial-time complexity while still reasoning over a super-exponential number of candidate structures.

We cast the problem of reasoning over temporally evolving structures as inference over a latent state sequence which indexes structure over time in a dynamic Bayesian network. This model allows one to utilize standard algorithms such as Expectation Maximization, Viterbi decoding, forward-backward messaging and Gibbs sampling in order to efficiently reasoning over an exponential number of structural sequences. We demonstrate the utility of our methodology on two tasks: audio-visual association and moving object interaction analysis. We achieve state-of-the-art performance on a standard audio-visual dataset and show how our model allows one to tractably make exact probabilistic statements about interactions among multiple moving objects.

Thesis Supervisor: John W. Fisher III

Title: Principal Research Scientist of Electrical Engineering and Computer Science

Acknowledgments

I would like to thank my advisor, John Fisher, for many years of guidance, encouragement and generous support. I met John during my first visit to MIT and find it difficult to imagine getting through graduate school without him. His dedication to research, his willingness to dive into the details and his humor not only made him a perfect advisor, but also a great friend.

I would also like to thank Alan Willsky and Bill Freeman for their insight and advice while serving on my thesis committee. I was welcomed into Alan's research group shortly after my Masters and have greatly appreciated our weekly grouplet discussions. It always amazes me how all those ideas and information is stored in one man's brain. Talking with Bill or even passing him in a hallway always brings a smile to my face and reminds me how fun and exciting good research is.

I would also like to thank Alex Ihler, Kinh Tieu, Kevin Wilson, Biswajit Bose, Wanmei Ou and Emily Fox for always being willing to help, listen, and discuss research ideas. Archana Venkataraman and Kevin Wilson get a special thank you for getting through early drafts of this dissertation and coming back with great comments and improvements.

I have been fortunate to have had the opportunity to interact with some the most intelligent and unique people I know while at the Computer Science and Artificial Intelligence Laboratory and in the Stochastic Systems Group. I have learned so much from them and have come away with so many great friendships. I could not adequately thank them all here.

Most importantly, I would like to thank my family. All my successes would have been empty and road blocks insurmountable without their patience, love and encouragement. I especially would like to thank my nephews Simon, Ethan, Alex and Adam and my niece Cate for reminding me of what is important. Finally, I'd like to thank my sweet Ana for giving me a reason to finish and look towards the future.

Different aspects of this thesis was supported by the Army Research office under the Heterogenous Sensor Networks MURI, the Air Force Office of Scientific Research under the Integrated Fusion, Performance Prediction, and Sensor Management for Automatic Target Exploitation MURI, the Air Force Research Laboratory under the ATR Center, and the Intelligence Advanced Research Projects Activity (formerly ARDA) under the Video Analysis and Content Extraction project

Contents

List of Figures

List of Tables

List of Algorithms

Image segmentation is a long studied problem in computer vision. While humans are able to segment images easily, the success of computational segmentation methods after decades of research is still disappointing.

Most image segmentation algorithms can be divided into a few tasks. First, some sort of distinguishing data which typically tries to capture low-level features in a local neighborhood is extracted from the pixel locations. These features are then used to optimize some surrogate energy functional or spectral graph over different segmentation configurations. Once completed, the performance of the segmentation is evaluated by either qualitatively or quantitatively comparing to other algorithms.

It is important to identify the differences between image segmentation and boundary detection because they typically have different algorithmic approaches. At a high level, image segmentation is the task of partitioning an image into disjoint, non-overlapping regions. Equivalently, one can assign discrete labels to each pixel identifying which region the pixel belongs to. Boundary detection is the task of identifying which pixels are boundaries. Marked boundaries may be open and need not partition an image into regions. Boundary detection algorithms also typically report the probability that a pixel is on the boundary (a soft decision), instead of a hard declaration of which pixels are on the boundary. Thresholding this probabiliy of boundary image can produce a true boundary detector. While these two tasks are related, the difference between the two are that the boundaries of a segmentation are implicitly closed regions that partition the image.

In previous work, the extracted image features have either been specifically chosen by the researchers or learned from data. However, the energy functionals or spectral graph edge weights are, in general, chosen by the researcher as a surrogate for the human visual system. People conjecture that minimizing energy configurations will correspond to good segmentations. However, these assumptions are only validated empirically after the segmentations have been completed. No one knows which energies or edge weights suit an image the best.

Additionally, there has been limited work on quantitatively evaluating a segmentation. While there are certain methods of doing this (e.g. precision-recall curves [6], probabilisty Rand index [11], and variation of information [**?**]), we will see that each of these methods does not properly score performance across all images. Without a proper metric on performance, trying to learn a "good" set of features for segmentation is meaningless.

In this thesis, we propose to address these issues. We first present an alternative, Bayesian approach to the typical optimization framework used in segmentation that provides a richer understanding of algorithms. We then propose to learn a good evaluation metric for rating segmentation results. Once a good metric has been established, we propose to learn both image features and the energy functionals used in optimization.

Chapter 1

Background

1.0.1 Segmentation Scoring

Because of the ill-posed nature of segmentation [5], multiple plausible segmentations can exist for a single image. For example, consider the human segmentations shown in Figure 3.0.5. Though there are many similarities across the solutions, discrepancies occur near the boundaries of objects, from the illusory edges, and are a result of the granularity of the segmentation. These varying solutions pose a very important question in segmentation: how does one evaluate the performance of a segmentation algorithm across the *set* of ground truth images?

The Berkeley Segmentation Dataset (BSD) [6] attempts to address this question. The dataset consists of a set of 300 images (100 test images, and 200 training images), each of which have been segmented by multiple experts. Additionally, an evaluation criterion for boundary detection using precision-recall curves was developed to score algorithms [7]. While the precision-recall curves may accurately assess the performance of a boundary detector, it is usually not a good measure of region-based segmentation algorithms. For example, consider the segmentations in Figure 1.1. While the bound-

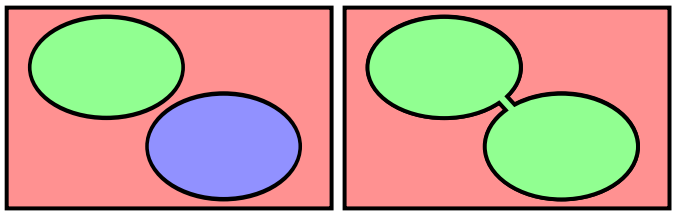


Figure 1.1: An example of how segmentations can differ greatly even when boundary detection is similar. Colors indicate segmentation labels.

aries in these two segmentations differ only by a small amount (the strait connecting the two regions), the segmentations are quite different. Boundary detection evaluation methods are clearly not suited to evaluate segmentations. We discuss a few previously presented methods on scoring segmentations.

■ 1.0.2 Probabilistic Rand Index

One popular way to compare the agreement between two segmentations is by using the Rand Index (RI) [9]. If a pair of pixels, $\{i, j\}$, was drawn at random from the image domain, the RI finds the empirical expected value of this pair agreeing across both segmentations, L^t and L^k. Mathematically, it can be expressed as

$$\mathrm{RI}\left(\ell^t, \ell^k\right) = \hat{\mathbf{E}}_{i,j}\left[g_{i.j}\left(\ell^t, \ell^k\right)\right] = \frac{1}{\binom{N}{2}} \sum_{\{i,j\}} g_{i,j}\left(\ell^t, \ell^k\right), \tag{1.1}$$

where N is the number of pixels in the image, the notation $\sum_{\{i,j\}}$ means the sum indexes over all possible pairs of pixels, $\{i, j\}$, and the agreement is defined as the following

$$g_{i,j}\left(\ell^t, \ell^k\right) = \mathbb{1}\left[\ell_i^t = \ell_j^t\right] \mathbb{1}\left[\ell_i^k = \ell_j^k\right] + \mathbb{1}\left[\ell_i^t \neq \ell_j^t\right] \mathbb{1}\left[\ell_i^k \neq \ell_j^k\right]. \tag{1.2}$$

Given an algorithm's segmentation result, ℓ^t, the performance of the algorithm could be scored by considering the Rand Index between ℓ^t and one ground truth segmentation, ℓ^k. However, as stated previously, image segmentation is an ill-posed problem, and results should be evaluated across a *set* of ground truth segmentations. This has recently led to the development of the Probabilistic Rand Index (PRI) [10] and the Normalized Probabilistic Rand Index (NPRI) [11], which is an unbiased and normalized version of the PRI. The PRI is essentially the average Rand Index over a set ground truth segmentations, $\left\{L^1, L^2, ..., L^K\right\}$, which we denote as $L^{1...K}$:

$$\mathrm{PRI}\left(\ell^t, \ell^{1...K}\right) = \frac{1}{K} \sum_k \mathrm{RI}\left(\ell^t, \ell^k\right) \tag{1.3}$$

Variation of Information

The variation of information [8] is an information theoretic evaluation criterion that tries to capture the amount of information that is different between two clusters (or equivalently, segmentations). The variation of information can be written in the follow-

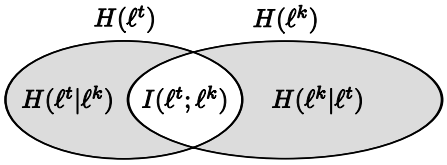


Figure 1.2: A graphical representation of the quantities in the variation of information.

ing ways:

$$VI(\ell^t, \ell^k) = H(\ell^t) + H(\ell^k) - 2I(\ell^t; \ell^k) \tag{1.4}$$

$$= H(\ell^t|\ell^k) + H(\ell^k|\ell^t), \tag{1.5}$$

where $H(\cdot)$ is the entropy, $I(\cdot)$ is the mutual information, and $H(\cdot|\cdot)$ is the conditional entropy. Figure 1.2 shows the quantities of the variation of information graphically. The first conditional entropy term captures the amount of information in the test segmentation, ℓ^t, not contained in the ground truth segmentation, ℓ^k. The second conditional entropy term captures the amount of information in the ground truth segmentation not contained in the test segmentation.

Global/Local Consistency Error

While creating the BSDS, the authors also introduced two segmentation measures called the Global Consistency Error (GCE) and the Local Consistency Error (LCE) [6]. By defining the region of a segmentation to be the set of pixels containing one label

$$R_l^t = \left\{ j; \ell_j^t = l \right\}, \tag{1.6}$$

the local refinement error can be defined as

$$E(\ell^t, \ell^k, i) = \frac{\left| R_{\ell_i^t}^t \setminus R_{\ell_i^k}^k \right|}{\left| R_{\ell_i^t}^t \right|}. \tag{1.7}$$

The GCE and LCE are then just:

$$GCE(\ell^t, \ell^k) = \frac{1}{|\Omega|} \min \left\{ \sum_{i \in \Omega} E(\ell^t, \ell^k, i), \ \sum_{i \in \Omega} E(\ell^k, \ell^t, i) \right\} \tag{1.8}$$

$$LCE(\ell^t, \ell^k) = \frac{1}{|\Omega|} \sum_{i \in \Omega} \min \left\{ E(\ell^t, \ell^k, i), \ E(\ell^k, \ell^t, i) \right\} \tag{1.9}$$

Both consistency error measures were formulated under the assumption that segmentations are hierarchical, and therefore, do not penalize segmentations for having a particular level of granularity. As the authors point out, a segmentation containing a single label (i.e. all pixels contained in one region), or a segmentation containing $|\Omega|$ labels (i.e. all pixels having different labels) both zero error.

■ 1.0.3 Learning Features

Two important concepts when trying to learn feature extractors from images are the recent deveopments in learning deep Restricted Boltzmann Machines and convolutional networks. We briefly cover these here.

Convolutional Neural Networks

The first substantial step in learning image descriptors was a result of convolutional neural networks [**?**]. CNNs are a very specific form of neural networks with many connections and only a few weights. They are set up in such a way that the learned weights equivalently act as a convolutional filter on an image. In typical CNNs, the weights of the filter (i.e. the filter coefficients) are learned in a supervised fashion using backpropagation to minimize some surrogate energy.

Commonly in CNNs, each layer of the neural network is composed of a convolutional layer and a sub-sampling or pooling layer. In classification problems, it has been shown [**?**] that the learning of pooling layers is much more important. With the use of random filters (as compared to learned ones), classification performance only slightly decreases. However, typical CNNs must be trained in a supervised fashion. While recent work (e.g. Ranzato et al. [**?**]) has developed an unsupervised learning algorithm for CNNs, they are still subject to the particular surrogate function to minimize over. Additionally, as we will see, CNNs are analytical (as opposed to generative) models, which limits its utility.

Restricted Boltzmann Machines

A Restricted Boltzmann Machine (RBM) is a single layer within a deep RBM. An RBM consists of a set of "visible" variables (denoted $\overline{v}$) and a set of "hidden" variables (denoted $\overline{h}$). For computational reasons, a typical RBM models the visible variables as being conditionally independent given the hidden variables and the hidden variables being conditionally independent given the visible variables. An example of this RBM

structure is shown in Figure 1.3. The weight between each node is denoted w_{ij} where i indexes an observed node and j indexes a hidden node.

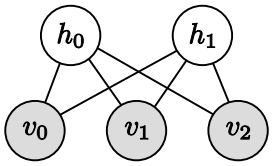


Figure 1.3: An example of a typical RBM.

The RBM is typically governed by an energy functional that depends on the weights of the edges (which can be viewed as parameters of the model) and both the visible and hidden variables. In Bayesian methods, it is often convenient to view the negative exponentiated energy functional as being proportional to the likelihood:

$$P(\overline{v}, \overline{h}; W) \propto \exp\left[-E(\overline{v}, \overline{h}; W)\right] \tag{1.10}$$

Particular choices of the energy functionals can lead to simpler inference and sampling. For example, one choice of the energy functional is

$$E(\overline{v}, \overline{h}; W) = \frac{1}{2}\sum_i v_i^2 - \sum_{i,j} v_i w_{ij} h_j - \sum_j b_j h_j - \sum_i c_i v_i, \tag{1.11}$$

where **b** and **c** are associated biases for the hidden and visible variables respectively. In this particular case, conditioned on the hidden variables, the visible variables are independent and Gaussian. Additionally, conditioned on the visible variables, the hidden variables are independent Bernoulli random variables.

Most energy functionals embed an independence structure; in these cases, we can write the posterior of the visible variables as follows:

$$P(\overline{v}|\overline{h}; W) = \frac{\prod_j P(\overline{v}|h_j; \overline{w_j})}{\sum_{\overline{v}' \in \mathbb{V}} \prod_j P(\overline{v}'|h_j; \overline{w_j})}, \tag{1.12}$$

where $\mathbb{V}$ is the set of all possible values of $\overline{v}$. The log likelihood is

$$\log P(\overline{v}|\overline{h}; W) = \sum_j \log P(\overline{v}|h_j; \overline{w_j}) - \log \sum_{\overline{v}' \in \mathbb{V}} \prod_j P(\overline{v}'|h_j; \overline{w_j}). \tag{1.13}$$

One would like to learn the weights, W that maximize the likelihood of the *observed*, visible variables. This maximum likelihood (ML) parameter estimate can be expressed

as:

$$W^* = \arg\max_W \left[\prod_{\overline{\mathbf{v}} \in \mathbf{V}} P(\overline{\mathbf{v}}|\overline{h}; W) \right] = \arg\max_W \left[\sum_{\overline{\mathbf{v}} \in \mathbf{V}} \log P(\overline{\mathbf{v}}|\overline{h}; W) \right], \tag{1.14}$$

where $\overline{\mathbf{v}}$ is the vector of one observed, visible variables, and $\mathbf{V}$ is the set of all observed, visible variables. This estimate can equivalently be expressed as the empirical expected value of the log likelihood taken over the observed, visible variables

$$W^* = \arg\max_W \mathbb{E}_{\mathbf{V}} \left[\log P(\overline{\mathbf{v}}|\overline{h}; W) \right]. \tag{1.15}$$

This optimization can be approximately found using gradient ascent. The gradient of the log likelihood for one observation is

$$\frac{\partial \log P(\overline{\mathbf{v}}|\overline{h}; W)}{\partial w_{ij}} = \frac{\partial \log P(\overline{\mathbf{v}}|h_j; \overline{w_j})}{\partial w_{ij}} - \sum_{\overline{v} \in \mathbb{V}} P(\overline{v}|\overline{h}; W) \frac{\partial \log P(\overline{v}|h_j; \overline{w_j})}{\partial w_{ij}} \tag{1.16}$$

$$= \frac{\partial \log P(\overline{\mathbf{v}}|h_j; \overline{w_j})}{\partial w_{ij}} - \mathbb{E}_{\mathbb{V}} \left[\frac{\partial \log P(\overline{v}|h_j; \overline{w_j})}{\partial w_{ij}} \right] \tag{1.17}$$

Thus, the entire gradient over all observations (combined with Equations 1.10 and 1.11) is

$$\frac{\partial \mathbb{E}_{\mathbf{V}} \left[\log P(\overline{\mathbf{v}}|\overline{h}; W) \right]}{\partial w_{ij}} = \mathbb{E}_{\mathbf{V}} \left[\frac{\partial \log P(\overline{\mathbf{v}}|h_j; \overline{w_j})}{\partial w_{ij}} \right] - \mathbb{E}_{\mathbb{V}} \left[\frac{\partial \log P(\overline{v}|h_j; \overline{w_j})}{\partial w_{ij}} \right] \tag{1.18}$$

$$= \mathbb{E}_{\mathbf{V}} \left[\mathbf{v}_i h_j \right] - \mathbb{E}_{\mathbb{V}} \left[v_i h_j \right]. \tag{1.19}$$

The expectations in the above equation can be estimated using samples from the distributions. Because the observed variables are conditionally independent given the hidden variables (and vice versa), we can perform block Gibbs sampling to draw samples from the joint:

1. Iteration 0:
   (a) Set the observed variables to be the actual observation: $\overline{v}^{(0)} = \overline{\mathbf{v}}$
   (b) Draw a set of hidden variables from the posterior: $\left\{ h_j^{(0)} \sim P(h_j|\overline{v}^{(0)}; W^{(0)}) \right\}$
2. Iteration $n > 0$:
   (a) Draw a set of observed variables from the posterior: $\left\{ v_i^{(n)} \sim P(v_i|\overline{h}^{(n)}; W^{(n)}) \right\}$
   (b) Draw a set of hidden variables from the posterior: $\left\{ h_j^{(n)} \sim P(h_j|\overline{v}^{(n)}; W^{(n)}) \right\}$

The first term in Equation 1.19, $\mathbb{E}_{\mathbf{V}}\left[\mathbf{v}_i h_j\right]$ can be estimated from samples from iteration 0, and the esecond term, $\mathbb{E}_{\mathbb{V}}\left[v_i h_j\right]$ can be estimated from samples from iteration ∞ as follows:

$$\frac{\partial \mathbb{E}_{\mathbf{V}}\left[\log P(\overline{\mathbf{v}}|\overline{h}; W)\right]}{\partial w_{ij}} = \left\langle v_i^{(0)}\, h_j^{(0)} \right\rangle - \left\langle v_i^{(\infty)}\, h_j^{(\infty)} \right\rangle, \tag{1.20}$$

where $\langle \cdot \rangle$ denotes the sample average. It can be shown that the gradient of the log likelihood is equivalent to the gradient of the following negative KL divergence:

$$\frac{\partial \mathbb{E}_{\mathbf{V}}\left[\log P(\overline{\mathbf{v}}|\overline{h}; W)\right]}{\partial w_{ij}} = \frac{\partial D\left(\overline{\mathbf{v}} \| \overline{v}\right)}{\partial w_{ij}} = \frac{\partial D\left(\overline{v}^{(0)} \| \overline{v}^{(\infty)}\right)}{\partial w_{ij}} \tag{1.21}$$

In practice, waiting for the Markov chain to converge can take a long time. Thus, minimizing contrastive divergence [2] is typically used as an approximation:

$$\frac{\partial \left(D\left(\overline{v}^{(0)} \| \overline{v}^{(1)}\right) - D\left(\overline{v}^{(1)} \| \overline{v}^{(\infty)}\right)\right)}{\partial w_{ij}} = \left\langle v_i^{(0)}\, h_j^{(0)} \right\rangle - \left\langle v_i^{(1)}\, h_j^{(1)} \right\rangle. \tag{1.22}$$

Learning in Deep Networks

One typically stacks multiple layers of RBMs on top of each other to learn a hierarchical set of features. The lower level RBMs can then typically learn low level features, while higher level RBMs can combine these features to identify higher level features. While training a single layerered RBM is straightforward, reliably training these deep belief networks can be difficult. Hinton et al. [3] proposed a greedy layer-wise training algorithm that performs well in practice.

Convoluational RBMs

While deep belief networks (DBNs) consisting of mulitple layers of RBMs have shown considerable success in small image tasks (e.g. [3]), they do not scale well with image size. There has been some more recent work [4, 1] on convolutional deep belief networks, where each layer consists of a small RBM that is applied at every location to the image. These convolutional RBM (CRMB) layers can be thought of as sharing weights between visible and hidden variables across the entire image. The weights are constrained in such a way they equivalently produce a convolution on a layer of nodes. Figure 1.0.3 shows an example of tied weights that perform a one-dimensional convolution of support three. These generative models have shown encouraging results on recognition and other computer vision tasks.

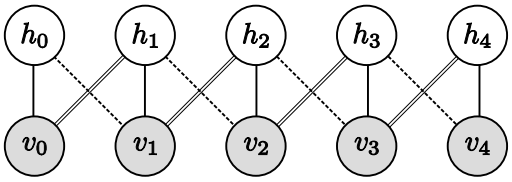


Figure 1.4: An example of tied weights that perform a finite one-dimensional convolution of support three. Each unique weight is represented with a different line (double, single, or dotted).

Chapter 2

Proposed Work

Chapter 3

Preliminary Work

■ 3.0.4 Sampling from Energy Functionals

Level-set based segmentation is often formulated as an energy minimization problem, where some energy functional is chosen such that a "good" segmentation occurs at low energies. It is often the case that, either due to the ill-posedness of unsupervised segmentation or the stochastic nature of a well posed formulation, multiple plausible explanations exist. In these cases, characterization of the posterior distribution of segmentations may offer a more informative solution to the problem. Consequently, a common alternative to the minimization formulation is to recast the problem as one of Bayesian inference, where the energy functional is viewed as the negative log of the posterior:

$$p(\ell|X) \propto \exp(-E(\ell; X)), \tag{3.1}$$

where ℓ is the matrix of labels assigned to each pixel (i.e. the segmentation) and X is the data at each pixel. We developed a Metropolis Hastings MCMC sampler to efficiently sample from these distributions in [**?**]. The algorithm represents the segmentation implicitly using a level-set function, φ.

Metropolis-Hastings MCMC Sampling

One can construct a Metropolis-Hastings sampler [**?**] as follows. Let $\hat{\varphi}^{(t+1)}$ be a proposed sample of the implicit representation (i.e. the level-set function) generated from a distribution $q(\hat{\varphi}^{(t+1)}|\varphi^{(t)})$ conditioned on the current sample, $\varphi^{(t)}$. The superscript values (t) and $(t+1)$ index the sampling iteration and the hat indicates a proposed

sample. This new sample is then accepted with probability

$$\Pr\left[\varphi^{(t+1)} = \hat{\varphi}^{(t+1)} \,\middle|\, \varphi^{(t)}\right] = \min\left[\overbrace{\underbrace{\frac{\pi\left(\hat{\varphi}^{(t+1)}\right)}{\pi\left(\varphi^{(t)}\right)}}_{\text{Posterior Sample Ratio}} \cdot \underbrace{\frac{q\left(\varphi^{(t)}\middle|\hat{\varphi}^{(t+1)}\right)}{q\left(\hat{\varphi}^{(t+1)}\middle|\varphi^{(t)}\right)}}_{\text{Forward-Backward Ratio}}}^{\text{Hastings Ratio}}, 1\right]. \quad (3.2)$$

Otherwise, $\varphi^{(t+1)} = \varphi^{(t)}$. Convergence to the stationary distribution occurs after a suitable number of iterations (i.e. the mixing time) which produces a *single* sample from the posterior. Evaluating the Hastings ratio, the product of the two ratios in the acceptance probability, has been the primary barrier for implementing MCMC methods over implicit representations. In particular, one needs to solve a correspondence problem to compute the probability of generating the forward and reverse transition (in the forward-backward ratio). Doing so satisfies the condition of detailed balance which, in addition to ergodicity, is sufficient for convergence to the desired posterior distribution.

As with any level-set representation, one needs to choose the magnitude of the level-set, φ, away from the curve. Previous sampling methods have constrained the level-set function to be a signed distance function (SDF). Chen [**?**] solves the correspondence problem by generating perturbations that are SDF-preserving, thus having a one-to-one mapping from forward and reverse transitions. An alternative is to produce a non-SDF-preserving perturbation and reinitialize the level set function to an SDF at each iteration. However, this creates a many-to-many correspondence problem which significantly increases the computational complexity of the forward-backward ratio.

Our idea is straightforward: do not constrain the level-set function to be an SDF. SDFs provide advantages in terms of numerical stability and the computation of the curvature (see [**?**] for details) for optimization based methods. As the method here is not PDE-based and optimization is not the specific goal, there is essentially no penalty for using an alternative. While our level-set function no longer satisfies the SDF property, we still benefit from the way implicit representations handle topological changes and re-parameterization. Furthermore, this greatly simplifies the design and evaluation of a proposal distribution by allowing for straightforward evaluation of the Hastings ratio.

Strategic Bias in the Proposal Distribution

We note that the closer $q(\circ|\triangle)$ is to $\pi(\circ)$, the closer the Hastings ratio is to unity and the higher the acceptance rate. Consequently, designing proposal distributions which

capture essential, application-specific characteristics of the posterior distribution can improve convergence speeds by reducing the number of rejected samples. By relaxing the SDF constraint on the level-set function, many potential proposal distributions will result in a tractable evaluation of the Hastings ratio. Without care, however, the majority of these proposal distributions will have very poor mixing times. Thus, our aim is to design a proposal distribution that is easily evaluated, has a high acceptance rate, and explores the configuration space via large perturbations.

In Equation 3.2, the Hastings ratio consists of the posterior sample ratio (PSR) and the forward-backward ratio (FBR). The PSR represents the ratio of the posterior probability of the new sample over that of the old. Generating samples that have higher posteriors will produce high values of this ratio. The FBR represents the probability of generating the previous sample conditioned on the new one (the backward transition) over the probability of generating the new sample conditioned on the previous one (the forward transition).

Fan et al. [**?**] suggest using a proposal distribution biased by the curvature to favor samples that fit the prior model. Here, we develop a proposal which favors both the likelihood and prior model. This generally produces higher PSR values, but biases the FBR toward smaller values (see the supplemental materials for an illustrative example). Thus, our goal is to develop a proposal distribution with a higher overall Hastings ratio (the product of the PSR and the FBR), where deleterious effects on the FBR are compensated with increases in the PSR. Exploiting the simple observation that neighboring pixels tend to have the same label, we can develop a proposal that has this property.

We construct an additive perturbation, $\mathbf{f}$, to $\varphi^{(t)}$,

$$\hat{\varphi}^{(t+1)} = \varphi^{(t)} + \mathbf{f}^{(t)}, \tag{3.3}$$

by first sampling from a point process, attributing the points with values sampled from a biased Gaussian distribution and then smoothing with a lowpass filter. We refer to this process as Biased and Filtered Point Sampling (BFPS). The lowpass filter captures the property that pixels in close proximity have higher probability of being in the same region while the *choice* of bias favors points with high likelihood under the energy functional. The result is dramatically increased PSRs using large biased moves while only slightly decreasing the FBR. Mathematically this is expressed as

$$\mathbf{f}^{(t)} = \mathbf{h}^{(t)} * \left(\mathbf{c}^{(t)} \circ \mathbf{n}^{(t)} \right), \tag{3.4}$$

$$n_i^{(t)} \sim \mathcal{N}\left(\mu_i^{(t)}, \sigma^2\right), \quad c_i^{(t)} \sim \text{Bernoulli}\left(p_{c_i}^{(t)}\right), \tag{3.5}$$

where '$*$' denotes convolution and '$\circ$' denotes the element-wise product. We bias the Gaussian RVs with the gradient velocity, $\mathbf{v}^{(t)}$, (the negative gradient of the energy functional) to prefer moving to more probable configurations:

$$\mu_i^{(t)} = \alpha_n \left[-\frac{\partial E\left(\varphi^{(t)}\right)}{\partial \varphi^{(t)}} \right]_i = \alpha_n v_i^{(t)}, \tag{3.6}$$

where α_n is a weighting parameter. The probability associated with each point, c_i, is also carefully selected to favor selecting points which are better explained in another region. Specifically, it is chosen to be higher for points that have a gradient velocity that is large in magnitude *and* has the opposite sign of the current level-set value:

$$p_{c_i}^{(t)}(1) \propto \alpha_c \exp\left[-v_i^{(t)} \cdot \text{sign}\left(\varphi_i^{(t)}\right) \right] + (1 - \alpha_c), \tag{3.7}$$

where α_c is a parameter that trades off the bias with a uniform distribution. Additionally, we define the variable γ as $\frac{1}{|\Omega|} \sum_{i \in \Omega} p_{c_i}^{(t)}(1) = \gamma$, which approximates the average probability that a random point will be selected, where Ω is the set of all pixels. Because $p_{c_i}^{(t)}(1)$ is only defined up to a scale factor, we can renormalize its value to achieve any γ. In practice, α_n, α_c, and γ are dynamically adapted to maintain a minimum acceptance rate, and $\mathbf{h}^{(t)}$ is chosen to be a circularly symmetric (truncated) Gaussian kernel with a scale parameter randomly chosen from a finite set of values. Randomly chosen scale parameters introduce a minor complication (which we address), but empirically result in faster mixing times.

This algorithm can additionally be extended to M-ary segmentation. Details about the formulation and results obtained using this sampling algorithm can be found in [**?**]

■ 3.0.5 Segmentation Scoring

We have also done some preliminary work in developing a suitable way to score segmentation results. This is a difficult task to evaluate from the BSDS for a few reasons. The segmentations in the BSDS are mostly object-based segmentation, where pixels are grouped by the semantic object to which they belong. This type of segmentation leads to the marking of illusory edges which can be attributed to the prior knowledge of the appearance of an object. We again refer to the segmentations shown in Figure 3.0.5.

Figure 3.1: An example from the Berkeley Segmentation Dataset [6]. The original image is shown on the left, and the five human segmentations are shown to the right. Notice the illusory edges of the hair and the black vest against the black background.

Without incorporating a more complicated object prior, simpler appearance-based segmentation algorithm are typically unable to capture the illusory edges depcited in 3.0.5.

The human segmentations of the BSDS only contain boundary maps outlined by the experts. Another reason that the BSDS is currently not well-suited for segmentation evaluation is because of the disconnect between boundary detection and segmentation; a segmentation uniquely determines a binary boundary map, but a binary boundary map does *not* uniquely determine a segmentation. This one-to-many mapping is caused by the more complete representation of segmentation labels, which allow separate connected components to have the same label. For example, in Figure 3.0.5, the textured shirt has two connected components (the chest, and the sleeve). These two connected components, in an accurate appearance-based segmentation, should have the same label. However, the human boundary maps of the BSDS do not contain information about which connected components have the same label.

One straightforward approach is to segment an image, and then label each connected component in the segmentation with a unique label. This, however, can lead to inaccurate performance evaluation when there are two regions that are connected by a small strait. For example, the background in Figure contains a large region of black and a black stripe. In the first human segmentation, the stripe is connected to the background and thus contains the same label. In subsequent segmentations, however, the stripe is labeled a different region because the expert continued the stripe a few pixels farther to the edge of the image. This ambiguity in segmentation labels can not easily be corrected in a performance measure.

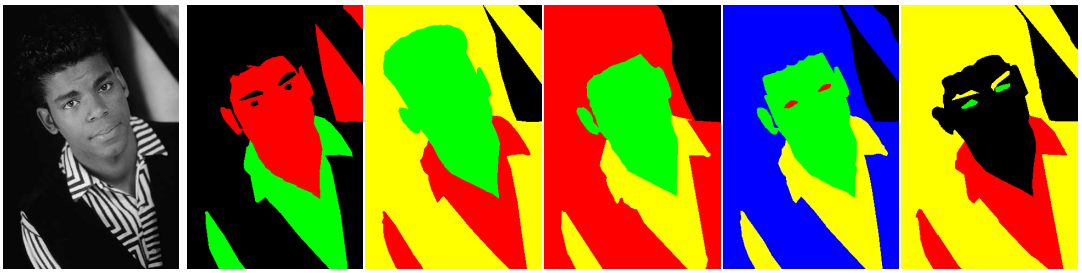


Figure 3.2: An example of the altered ground truth from the Berkeley Segmentation Dataset [6]. The original image is shown on the left, and the five altered human segmentations are shown to the right. Ground truth has been changed by grouping regions by similar appearances.

We have therefore altered the ground truth dataset of the BSDS by grouping segmented connected components by similar appearances. The grouping is performed while trying to respect the original intent of the experts by only merging regions and not creating or re-segmenting an image. An example of the altered ground truth is shown in Figure 3.2.

Although the new ground truth now represents what we would like to quantify, we still need to choose a particular criterion to score the performance of an algorithm.

Parameterized PRI Form

Using the PRI to evaluate segmentation results against a set of ground truth images seems intuitively plausible. In fact, oftentimes the PRI is a fairly good criterion for scoring the success of an algorithm. However, there are times when it does not perform well. Figure 3.3 shows a few examples of this. Clearly, PRI does not seem to rank segmentations well for these images. To understand why this occurs, let us first rewrite the expression for PRI. We begin by defining the following functional:

$$\text{fPRI}\left(\ell^t, \ell^{1\ldots K}, p^k(l_1, l_2)\right) = \frac{1}{K}\sum_k \sum_{\{l_1,l_2\}} p^k(l_1, l_2)\mathbf{E}_{i.j}\left[g_{i,j}\left(\ell^t, \ell^k\right)|\ell_i^k = l_1, \ell_j^k = l_2\right]. \tag{3.8}$$

Notice that this is an iterated expectation over label pairs, with a prior on label pairs defined by $p(l_1, l_2)$. With some manipulation, we can express the PRI in this functional

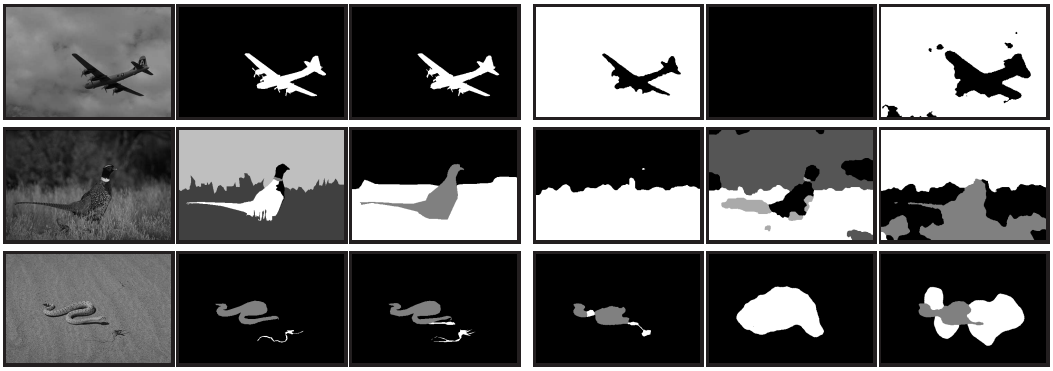


Figure 3.3: Segmentation ranking according to PRI. The first column is the original image and the second and third column are two (of many) ground truth segmentation. The third, fourth, and fifth column show segmentations obtained using various algorithms, ranked from best to worst according to PRI.

form:

$$\mathrm{PRI}\left(\ell^t, \ell^{1\ldots K}\right) = \mathrm{fPRI}\left(\ell^t, \ell^{1\ldots K}, p^k_{\mathrm{PRI}}(l_1, l_2)\right) \tag{3.9}$$

with a prior on label pairs proportional to the number of pixel pairs with labels $\{l_1, l_2\}$. This prior can be expressed as

$$p^k_{\mathrm{PRI}}(l_1, l_2) = \frac{\left(\mathbb{1}\left[l_1 = l_2\right] \cdot \binom{N_{l_1}}{2} + \mathbb{1}\left[l_1 \neq l_2\right] \cdot N_{l_1} N_{l_2}\right)}{\binom{N}{2}}, \tag{3.10}$$

where N_l is the number of pixels with label l.

We propose that one discrepancy between how humans evaluate segmentations and how they are scored according to PRI is caused by uneven region sizes in the ground truth. When two regions have very different sizes, the prior on label pairs is unevenly weighted across the regions. For example, the image of the plane in Figure 3.3 contains one very large region (the background), and a much smaller region (the plane). Assigning the background with l_1 and the foreground with l_2, we see that the prior on label pairs has the following relationship:

$$p^k_{\mathrm{PRI}}(l_1, l_1) \gg p^k_{\mathrm{PRI}}(l_1, l_2) \gg p^k_{\mathrm{PRI}}(l_2, l_2).$$

We can think of $p(l_1, l_2)$ as the data-dependent weight given to the label pair (l_1, l_2). In other words, it is the weight assigned to the success rate of an algorithm when one

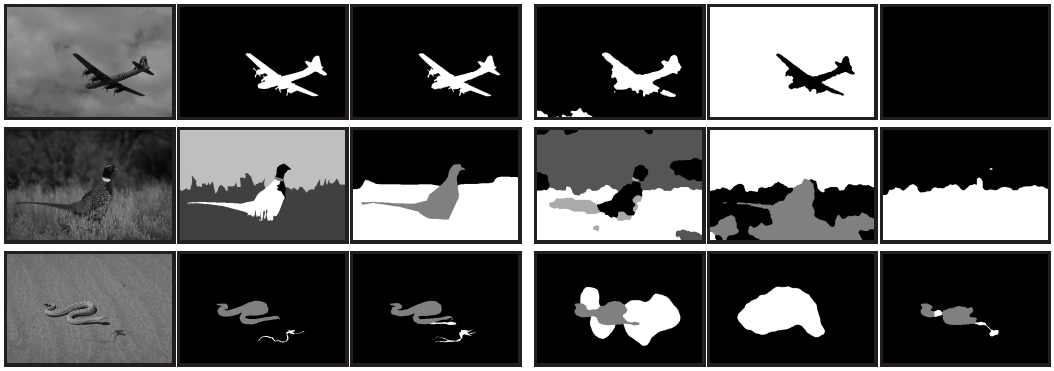


Figure 3.4: Segmentation ranking according to UPRI. The first column is the original image and the second and third column are two (of many) ground truth segmentation. The third, fourth, and fifth column show segmentations obtained using various algorithms, ranked from best to worst according to UPRI.

pixel is drawn from the labeled region l_1, and another one is drawn from the labeled region l_2. This relationship of weights for PRI indicates that more weight is assigned to correctly assigning a pair of pixels that both come from the background than when both pixels come from the foreground. This is one reason that the constant segmentation has a better PRI than the last column in Figure 3.3.

With this observation in mind, we propose a new evaluation criterion called the Uniform Probabilistic Rand Index (UPRI), which takes on the same functional form as the PRI (Equation 3.8), but has a uniform prior on label pairs:

$$\text{UPRI}\left(\ell^t, \ell^{1\ldots K}\right) = \text{fPRI}\left(\ell^t, \ell^{1\ldots K}, \frac{1}{\binom{L_k}{2}}\right) \tag{3.11}$$

Figure 3.4 shows the updated rankings using the UPRI. Notice that using UPRI to evaluate segmentation performance has fixed many of the problems in ranking that PRI failed at. However, an undesirable outcome with using UPRI is that the best airplane and snake segmentations are no longer truly the best. We attribute this result to the fact that the Rand Index does not penalize mislabeling pixel pairs as much as it should.

We therefore consider another measure in the same functional form, called the Area-Weighted Probabilistic Rand Index (APRI). The original PRI was essentially weighting the priors by the square of the number of pixels in the region; the APRI instead weights

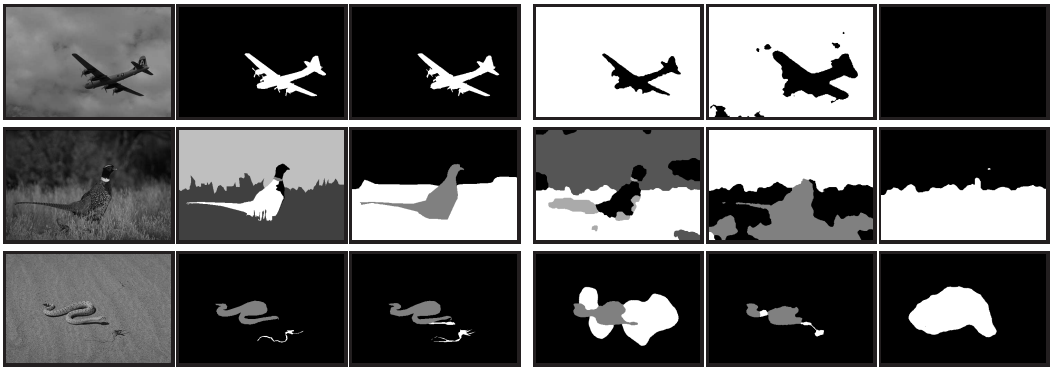


Figure 3.5: Segmentation ranking according to APRI. The first column is the original image and the second and third column are two (of many) ground truth segmentation. The third, fourth, and fifth column show segmentations obtained using various algorithms, ranked from best to worst according to APRI.

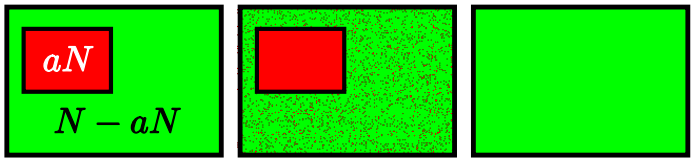
the priors proportional to the number of pixels in the region. The APRI can be written as:

$$\text{APRI}\left(\ell^t, \ell^{1\ldots K}\right) = \text{fPRI}\left(\ell^t, \ell^{1\ldots K}, \frac{(N_{l_1} + N_{l_2})}{2}\right). \tag{3.12}$$

Figure 3.5 shows the ranksing using the APRI.

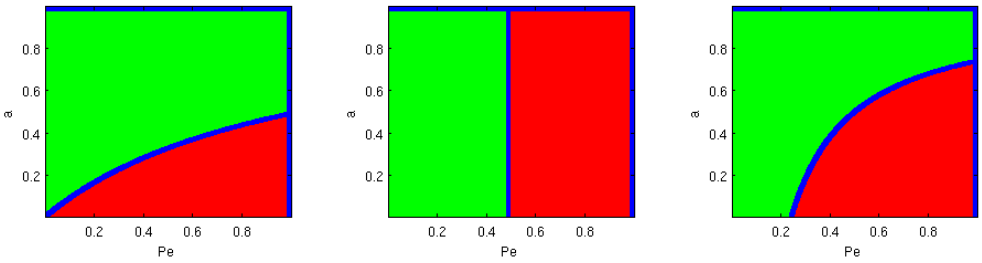
It seems like the errors with the plane have been fixed, but those of the snake have not. At this point, we believe that there is no clear way of choosing these priors to accurately evaluate segmentations across all images.

To understand the differences between PRI, UPRI, and APRI, we consider the following example. We examine the relationship of the measures with region size and error rates on segmentation. For an image with N pixels, we generate a ground truth segmentation with two regions: one of size aN, and one of size $N - aN$. We then corrupt the $N - aN$ region with some probability of error Pe. We compare the three evaluation criteria using this corrupted segmentation with a segmentation that puts all pixels in the same label. Figure 3.6 shows an example ground truth, corrupted segmentation, and single region segmentation. For a range of a and Pe values, the decision regions for the better segmentation (the corrupted vs. the single region) based on PRI, UPRI, and APRI are shown in Figure 3.7. When using PRI as the segmentation measure, the better segmentation depends on the relative area of the ground truth and the probability of error. For example, when the ground truth contains a small region



(a) Ground truth (b) Corrupted (c) Single region

Figure 3.6: The corrupted segmentation incorrectly labels a pixel in the green region with probability Pe. We evaluate PRI, UPRI, and ULI for a range of a and Pe values.



(a) PRI (b) UPRI (c) APRI

Figure 3.7: The decision regions based on different measures. The green region corresponds to the single region segmentation having a higher score than the corrupted segmentation. The red region corresponds to the corrupted segmentation having a higher score than the single region segmentation. The blue lines indicate where the two segmentations are equal.

(i.e. when a is small), only very small Pe values can be tolerated. In general, PRI favors the single region segmentation when the ground truth contains a small region. This can be directly attributed to the prior on label pairs for PRI (expressed in Equation 3.14) which puts higher priors on larger region pairs. When a is small, PRI does not strongly penalize segmentations if the small aN region is missed because most of the prior weight is on the larger region. In stark contrast, because of the uniform prior on label pairs, UPRI has no dependence on the relative areas of the regions. The Area-Weighted PRI is somewhere in between PRI and UPRI.

Parameterized Likelihood Form

We have additionally considered another parameterized form of evaluation criteria related to the PRI which we will refer to as the Likelihood Index. Measures in this functional form have depend on the number of observed segmentations (assuming some random, generative model), which could be important when considering the ill-posed nature of the segmentation problem. When developing the PRI, the authors of [10] express it as the following

$$\mathrm{PRI}\left(\ell^t, \ell^{1\ldots K}\right) = \frac{1}{\binom{N}{2}} \sum_{\{i.j\}} p_{i,j}\left(\ell^t\right) = \mathbf{E}_{i,j}\left[p_{i,j}\left(\ell^t\right)\right], \tag{3.13}$$

where the likelihood of a pair of pixels, $p_{i,j}\left(\ell^t\right)$ is defined as

$$p_{i,j}\left(\ell^t\right) = \mathbb{1}\left[\ell_i^t = \ell_j^t\right] \Pr\left(\ell_i = \ell_j\right) + \mathbb{1}\left[\ell_i^t \neq \ell_j^t\right] \Pr\left(\ell_i \neq \ell_j\right), \tag{3.14}$$

and the probabilities, $Pr\left(\ell_i = \ell_j\right)$ and $Pr\left(\ell_i \neq \ell_j\right)$, are the empirical probabilities from the set of ground truth segmentations. In other words, the PRI is finding the empirical expected value of $p_{i,j}\left(\ell^t\right)$.

We take a different approach by considering the likelihood of the entire labeling (properly normalized). We call this measure the Likelihood Index (LI):

$$\mathrm{LI}\left(\ell^t, \ell^{1\ldots K}\right) = \prod_{\{i,j\}} p_{i,j}\left(\ell^t\right)^{\frac{1}{\binom{N}{2}}}. \tag{3.15}$$

The LI can also be expressed as

$$\mathrm{LI}\left(\ell^t, \ell^{1\ldots K}\right) = \exp\left[\mathbf{E}_{i,j}\left[\log p_{i,j}\left(\ell^t\right)\right]\right]. \tag{3.16}$$

We note here that this form of the Likelihood Index is very similar to the PRI in Equation 3.13; the only difference between the two expressions is the use of a different loss function (log versus linear) which, as eluded to earlier, was the root of the problem in the UPRI measure.

With some manipulation, we can write a functional Likelihood Index (similar to Equation 3.8) as

$$\mathrm{fLI}\left(\ell^t, \ell^{1\ldots K}, p^k\left(l_1, l_2\right)\right) = \exp\left[\frac{1}{K}\sum_k \sum_{\{l_1,l_2\}} p^k(l_1,l_2)\mathbf{E}_{i.j}\left[\log p_{i,j}\left(\ell^t\right) \middle| \ell_i^k = l_1, \ell_j^k = l_2\right]\right].$$

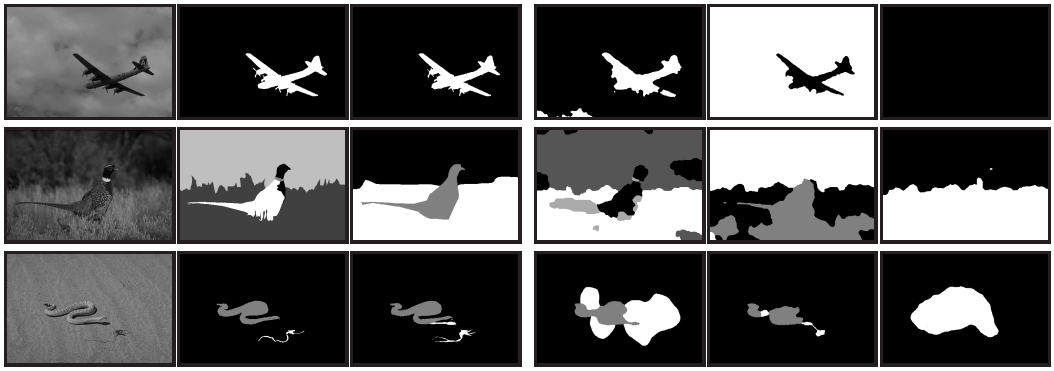


Figure 3.8: Segmentation ranking according to ULI. The first column is the original image and the second and third column are two (of many) ground truth segmentation. The third, fourth, and fifth column show segmentations obtained using various algorithms, ranked from best to worst according to ULI.

We define the Uniform Likelihood Index (ULI) which, like previously, uses this functional form with a uniform prior on label pairs:

$$\mathrm{ULI}\left(\ell^t, \ell^{1\ldots K}\right) = \mathrm{fLI}\left(\ell^t, \ell^{1\ldots K}, \frac{1}{\binom{L_k}{2}}\right) \tag{3.17}$$

We note that some regularization must be used because missing one pixel pair that all ground truth images agree on will make the likelihood index go to zero. We add uniform noise to the probability of a pixel agreeing and disagreeing in the following fashion:

$$\Pr\left(\ell_i = \ell_j\right) = \frac{1}{K + \frac{1}{5}}\left[\sum_{k=1}^{K} \mathbb{1}\left[\ell_i^k = \ell_j^k\right] + \frac{1}{10}\right], \tag{3.18}$$

$$\Pr\left(\ell_i \neq \ell_j\right) = \frac{1}{K + \frac{1}{5}}\left[\sum_{k=1}^{K} \mathbb{1}\left[\ell_i^k \neq \ell_j^k\right] + \frac{1}{10}\right]. \tag{3.19}$$

Figure 3.8 shows the updated rankings using the ULI. Notice that an algorithm's performance seems to be ranked according to how a human would rank the images.

The example in the previuos section considered how the prior on label pairs were reflected in region size and probability of error. The following example, shown in Figure 3.9, shows the difference between measures of the functional PRI form and the function LI form. We generate a set of K ground truth binary segmentations, each composed of three different sections: the first section is always labeled in one region, the second

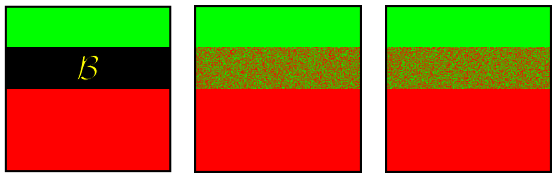


Figure 3.9: An example of generating a ground truth image. The image on the left shows the three separate sections. Pixels in the black band are randomly assigned a label for each generated ground truth image. Two example ground truth images are shown to the right.

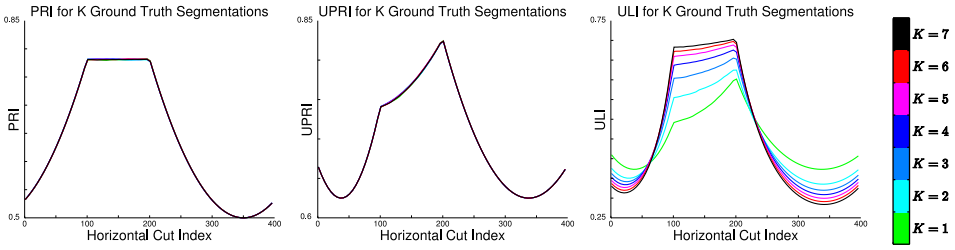


Figure 3.10: An example of generating a ground truth image. The image on the left shows the three separate sections. Pixels in the black band, $\mathcal{B}$, are randomly assigned a label for each generated ground truth image. Two example ground truth images are shown to the right.

section is always labeled in the other region, and the third region is labeled randomly. We consider segmentations containing one horizontal cut in the image for each of the measures. The different measures as a function of the horizontal cut index are shown for a different number of ground truth images in Figure 3.10. Clearly, the PRI and UPRI are not functions of the number of ground truth images whereas the ULI is. While the banded region is randomly labeled, if only one ground truth segmentation is given, one has no way of knowing that the labels in that region were assigned randomly (ignoring the spatial dependencies). Because of the statistics over the set of ground truth segmentations, it is only with multiple ground truth images where the labeling is random in each of the banded regions that one would know that the labeling is truly random. Mathematically, this can be seen by considering the probability that a pixel in one region has the same label as a pixel in the banded region, $\mathcal{B}$, as a function of the

number of ground truth images, K.

$$\Pr\left[\Pr\left[\ell_i = \ell_j | j \in \mathcal{B}\right] = \frac{k}{K}\right] = \frac{\binom{K}{k}}{2^K}, \forall k \in [0, K] \tag{3.20}$$

Chapter 4

Timeline

Bibliography

[1] Guillaume Desjardins and Yoshua Bengio. Empirical Evaluation of Convolutional RBMs for Vision. *Learning*, pages 1–13, 2008. 25

[2] Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002. 25

[3] Geoffrey E Hinton, S Osindero, and YW Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006. 25

[4] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. *Proceedings of the 26th Annual International Conference on Machine Learning ICML 09*, 26:1–8, 2009. 25

[5] J Marroquin, S Mitter, and T Poggio. Probabilistic Solution of Ill-Posed Problems in Computational Vision. *Journal of the American Statistical Association*, 82(397):76–89, 1987. 19

[6] David R 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. *Proceedings Eighth IEEE International Conference on Computer Vision ICCV 2001*, 2(July):416–423, 2001. 11, 16, 19, 21, 33, 34

[7] 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*, 26(5):530–549, 2004. 19

[8] Marina Meila. Comparing clusterings by the variation of information. *Learning Theory and Kernel Machines*, 2777(2777):173–187, 2003. 20

[9] William M Rand. Objective Criteria for the Evaluation of Clustering Methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971. 20

[10] Ranjith Unnikrishnan and Martial Hebert. Measures of Similarity. *Applications of Computer Vision and the IEEE Workshop on Motion and Video Computing, IEEE Workshop on*, 1:394, 2005. 20, 39

[11] Ranjith Unnikrishnan, C Pantofaru, and M Hebert. A Measure for Objective Evaluation of Image Segmentation Algorithms. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR05 Workshops*, 00(c):34–34, 2005. 16, 20