Machine Learning in Computer Vision
Markov Random Fields – Part IV

Oren Freifeld
Computer Science, Ben-Gurion University

April 11, 2018
1 Few More Inference Methods

2 Select Applications of MRFs
   - Interactive Segmentation
   - Superresolution and Texture Synthesis
   - Learned High-order Fields and their Applications
   - MRFs on Non-lattice Graphs
Belief Propagation (BP) on Markov Random Trees

- Similar to Dynamic Programming
- Two variants: max-product (for computing MAP solutions) and sum-product (for computing marginals, expectations, and normalizers)
- Exact
Belief Propagation (BP) on Markov Random Trees

Here are details for the max-product, assuming a pairwise model,

\[ p(x) \propto \prod_{i \sim j} F(x_i, x_j) \]

- **First stage:** “messages” are passed from leaf nodes to their parents which in turn pass messages to their parents and so on till the root. The message \( m_{i \rightarrow j}(x_j) \) from a node \( i \) to its parent \( j \) is
  \[ m_{i \rightarrow j}(x_j) = \max_{x_i} F(x_i, x_j) \prod_{k \in \text{children}(i)} m_{k \rightarrow i}(x_i) \]
  The MAP label at the root \( r \) is:
  \[ \hat{x}_r = \arg \max_{x_r} \prod_{k \in \text{children}(i)} m_{k \rightarrow i}(x_r) \]

- **Second stage:** Given the MAP label \( \hat{x}_p \) of variable \( p \), the label of any of its children \( i \), is:
  \[ \hat{x}_i = \arg \max_{x_i} F(\hat{x}_p, x_i) \prod_{k \in \text{children}(i)} \]
  Sum-product is simpler: replace max with sum; there is no second stage.
Loopy Belief Propagation (BP) on General MRFs

- Might converge to a non-global maximum.
- Might not converge at all.
- Empirically works well in many cases (and fails in many others).
Loopy Belief Propagation (BP) on General MRFs

Consider an MRF with pairwise potentials:

\[
p(x) \propto \exp \left( - \sum_{i \sim j} \psi(x_i, x_j) \right)
\]

As an example, here is the sum-product case:

- initialize \( m^{t=0}_{i \rightarrow j} = 1 \) \( \forall \) neighbor pairs \((i, j)\)

- Update rule at iteration \( t \):

\[
m^{t}_{i \rightarrow j}(x_j) = \sum_{x_i} \exp \left( -\psi_{i,j}(x_i, x_j) \right) \prod_{k: k \in \eta_i, k \neq j} m^{t-1}_{k \rightarrow i}(x_i)
\]

\[
b^t_j(x_j) \propto \prod_{i \in \eta_j} m^t_{i \rightarrow j}(x_i)
\]

- Hope #1: This will converge.

- Hope #2: \( b^t_i(x_i) \rightarrow \frac{1}{\text{const}} p_i(x_i) \)
Iterated Conditional Modes (ICM)

- Another inference method. Similar to Gibbs sampling, except that instead of updating $x_s$ with a sample from $p(x_s | x_{\eta_s})$, we update it using $\text{arg max}_{x_s} p(x_s | x_{\eta_s})$.

- Very greedy... Typically converges in few sweeps. Usually to a local maximum.
Other Inference Methods

- **Graph Cut**: A popular argmax method in MRFs in computer vision; especially effective for discrete pairwise models.
- **Variational approximation**: approximate $p$ with $q \propto \prod_s F_s(x_s)$
Interactive Foreground Extraction using GraphCut and GrabCut

- Relies on discrete optimization and a regional selection interface.
- The two main algorithms
  - GraphCut
  - GrabCut (which expands on GrabCut) – the underlying algorithm for the Background Removal tool in MS Office 2010
Interactive Foreground Extraction using GraphCut and GrabCut

Left: GraphCut example. Right: GrabCut example

Figures from “Markov random fields for vision and image processing” (editors: Blake and Rother, 2011)
Interactive Foreground Extraction using GraphCut

- Trimap: $T = \{T_F, T_B, T_U\}$ (foreground, background, unknown)
- Pixel intensities: $y_1, \ldots, y_N$ ($N$ is the number of labels)
- $h_B, h_F = (h_B(\cdot), h_F(\cdot))$: color histograms (known or based on the user’s choice)
- Observation models:

  \[ P(y_s | x_s = 1) = h_F(y_s) \]

  and

  \[ P(y_s | x_s = 0) = h_B(y_s) \]
Interactive Foreground Extraction using GraphCut

- $x$: binary segmentation
- $y$: observed image
- $T_B$ and $T_F$ are sets of pixels selected by the user as foreground and background
- $T_U$: the remaining pixels (whose segmentation is unknown)
- $h_B$ & $h_F$ color histograms in $T_B$ and $T_F$
Interactive Foreground Extraction using GraphCut

\[ p(x|y, h_B, h_F) \propto \exp \left( -E(x, h_B, h_F, y) \right) \]

\[ E(x, h_B, h_F, y) = U(x, h_B, h_F, y) + V(x, y) \]

\[ U(x, h_B, h_F, y) = \]

\[ - \left( \sum_{i \in T_U, x_s = 0} \log h_B(y_s) \right) - \left( \sum_{i \in T_U, x_s = 1} \log h_F(y_s) \right) + \left( \sum_{i \in T_F \cup T_B} H(x_s, i) \right) \]

\[ H(x_s, i) = \gamma \text{ if } (x_s = 0 \text{ and } i \in T_F) \text{ or } (\text{if } x_s = 0 \text{ and } i \in T_B) \]

\[ V(x, y) = \sum_{s \sim t: x_s \neq x_t} \frac{1}{\text{distance}(s, t)}(\lambda_1 + \lambda_2 \exp(-\beta \|y_s - y_t\|^2)) \]

\[ \lambda_2 = 0 \text{ is an Ising-like model; } \lambda_2 > 0 \text{ relaxes the tendency to smoothness in high-contrast images.} \]

\[ \hat{x} = \arg \min_x E(x, h_B, h_F, y) \text{ (found using graph cut)} \]
GrabCut is conceptually similar, except that the color distributions are estimated as well. Typically the color distributions are represented by Gaussian Mixture Model (GMMs). We will learn about GMM later on.

In comparison with the approach using graphcut, this requires less interaction from the user.
The approach described here combines:

- An example-based approach;
- An MRF-based approach

(there are also other MRF-based approaches for these tasks)
Superresolution

- In the superresolution case, assume the degradation process was blurring followed by downsampling.
- Apply this process to a training set, creating high/low res pairs.
- From the generated low-res examples, create “upsampled low-res images” and use it to create several “bands” (I omit the details).
- Given an input patch, consider the k-nearest neighbors at the training data.
Superresolution

Select Applications of MRFs

Superresolution and Texture Synthesis

Figures from “Markov random fields for vision and image processing”, 2011)
**Superresolution**

Left to right: Low-res input, Bi-cubic interpolation, argmax MRF, ground truth

---

Figures from “Markov random fields for vision and image processing”, 2011
Superresolution

Left to right: Low-res input, Bi-cubic interpolation, argmax MRF, ground truth

Figures from “Markov random fields for vision and image processing”, 2011
Superresolution

Left to right:
Low-res input, Bi-cubic interpolation, nearest-neighbor, argmax MRF, ground truth

Figures from “Markov random fields for vision and image processing”, 2011
Select Applications of MRFs

Superresolution and Texture Synthesis

Texture Synthesis

This optimal path problem has a well-known solution through dynamic programming [4], which has been exploited in various vision and graphics applications [24, 6]. This is equivalent to finding the maximum posterior probability through max-product belief propagation. We summarize the algorithm:

Initialization:
\[ p(i,1) = d(i,1) \]

for \( j = 2:N \)
\[ p(i,j) = p(i, j-1) + \min_k d(k,j) \]

end

, where the values considered for the minimization over \( k \) are \( i \), and \( i \pm 1 \). Using an auxiliary set of pointers indicating the optimal value of the \( \min_k \) operation at each iteration, the path \( q(i) \) can be found from the values of \( p(i,j) \). This method has also been used to hide image seams in "seam carving", [1].

Figure 8: Patch samples of an input texture can be composited to form a larger texture in a number of different ways. (a) A random placement of texture samples gives strong patch boundary artifacts. (b) We can select only patches that match well with neighbors in an overlap region, but there are still some boundary artifacts in the composite image. (c) Selecting the best seam through the boundary region of neighboring patches removes most artifacts. Figure reprinted from [6].

Selected related applications by others

Markov random fields have been used extensively in image processing and computer vision. Geman and Geman brought Markov random fields to the attention of the vision community, and showed how to use MRF's as image priors in restoration applications, [13]. Poggio, Gamble and Little used MRF's in a framework unifying different computer vision modules, [21].

The example-based approach has been built on by others. This method has been used in combination with a resolution enhancement model specific to faces [2] to achieve excellent results in hallucinating details of faces [19]. Huang and Ma have proposed finding a linear combination of the candidate patches to fit the input data, then applying the same regression to the output patches, simulating a better fit to the input [25]. (A related approach was also used in [11]).

Figures from “Markov random fields for vision and image processing”, 2011
Texture Synthesis

Figure 9: A collection of source (small image) and corresponding synthesized textures made using the patch-based imagequilting method. Figure reprinted from [6].

Optimal seams for image transitions were found in a 2-d framework, using graph cuts in Kwatra et al [16]. Example-based image priors were used for image-based rendering in the work of Fitzgibbon, Wexler, and Zisserman, [9]. Fattal used edge models for image upsampling [8]. Glasner et al also used an example-based approach for super-resolution, relying on self-similarity within a single image [14].

References


Figures from “Markov random fields for vision and image processing”, 2011
Consider first the shortcomings of a pairwise-based model.

Figure 2: Typical pairwise MRF potential and results: (a) Example of a common robust potential function (negative log-probability). This truncated quadratic is often used to model piecewise smooth surfaces. (b) Image with Gaussian noise added. (c) Typical result of denoising using an ad-hoc pairwise MRF (obtained using the method of Felzenszwalb and Huttenlocher (2004)). Note the piecewise smooth nature of the restoration and how it lacks the textural detail of natural scenes.
Field of Experts (Roth and Black)

The FoE is a learned, high-order, parameterized MRF that serves as a generic image prior.

\[
p(x; \Theta) = \frac{1}{Z(\theta)} \prod_{k=1}^{K} \prod_{i=1}^{N} \phi(J_i^T x(k); \alpha_i) = \frac{1}{Z(\theta)} \exp \left( \sum_{k=1}^{K} \sum_{i=1}^{N} \psi(J_i^T x(k); \alpha_i) \right)
\]

where:

- \(x(k)\): the vectorized version of a, say, 5 \times 5 patch in \(x\) around the \(k\) pixel
- \(J_i\): a vectorized version of a, say, 5 \times 5 linear filter
- \(\{J_i\}_{i=1}^{N}\): a “filter bank”
- \(\psi\): a parametrized robust error function.
- \(\alpha_i\): a parameter for \(\psi\).
- \(\Theta = (\{J_i\}_{i=1}^{N}, \{\alpha_i\}_{i=1}^{N})\): model parameters

Learning is non-trivial (involving some smart ideas) but not so hard to implement. Inference is typically done with MCMC.
Field of Experts (Roth and Black)

Figure 4: Selection of the 5 × 5 filters obtained by training the Product-of-Experts model on a generic image database.

Figure from Roth and Black, 2009
Field of Experts (Roth and Black)

Denoising

Figure from Roth and Black, 2009
Field of Experts (Roth and Black)

Inpainting

Figure from Roth and Black, 2009
Field of Experts (Roth and Black)

Figure 7: Denoising with a Field of Experts: Full image (top) and detail (bottom). (a) Original noiseless image. (b) Image with additive Gaussian noise ($\sigma = 25$); PSNR = 20.29dB. (c) Denoised image using a Field of Experts; PSNR = 28.72dB. (d) Denoised image using the approach of Portilla et al. (2003); PSNR = 28.90dB. (e) Denoised image using non-local means (Buades et al., 2004); PSNR = 28.21dB. (f) Denoised image using standard non-linear diffusion; PSNR = 27.18dB.

Figure from Roth and Black, 2009
Field of Experts (Roth and Black)

Figure 7: Denoising with a Field of Experts: Full image (top) and detail (bottom). (a) Original noiseless image. (b) Image with additive Gaussian noise ($\sigma = 25$); PSNR = 20.29dB. (c) Denoised image using a Field of Experts; PSNR = 28.72dB. (d) Denoised image using the approach of Portilla et al. (2003); PSNR = 28.90dB. (e) Denoised image using non-local means (Buades et al., 2004); PSNR = 28.21dB. (f) Denoised image using standard non-linear diffusion; PSNR = 27.18dB.

Figure 8: Other Field of Experts denoising results. Noisy input (left) and denoised image (right).

Figure from Roth and Black, 2009
Field of Experts (Roth and Black)

Figure 10: Inpainting with a Field of Experts. (a) Original image with overlaid text. (b) Inpainting result from diffusion algorithm using the FoE prior. (c) Close-up comparison between a (left), b (middle), and the results of Bertalmío et al. (2000) (right).
Field of Experts (Roth and Black)

Figure 11: **Other image inpainting results.** The top row show the masked images; the red areas are filled in by the algorithm. The bottom row show the corresponding restored images that were obtained using a $5 \times 5$ FoE model with 24 filters.
Steerable Random Fields

- Learned model, high-order cliques, spatially-adaptive (adapting to the local image structure)
- There are several differences from the FoE, but the main one is that instead of responses to linear filters, responses to steered filters are used.
Steerable Random Fields

Figure 2. Example image and steered derivatives. The derivative response orthogonal to the local structure is shown in the middle, the response aligned with the image structure on the right.

The SRF is a particular case of Conditional Random Fields (CRFs), which form a subclass of MRFs. Broadly speaking, we construct $p$ from the product of likelihood and “prior” terms as usual, except that the “prior” term in a CRF depends on the data as well (so it is not really a prior). Particularly, in the SRF model, before we evaluate (the unnormalized) $p$, we steer the filters according to the data.
Steerable Random Fields

Figure 8. **Image inpainting using a SRF.** (a) Masked image (red regions are to be filled in). (b) Inpainting with a pairwise MRF (PSNR = 37.83dB, SSIM = 0.980). (c) Inpainting with a SRF (PSNR = 41.08dB, SSIM = 0.983, without “ST update”). (d) Detail results. Top row: Original image, masked image. Bottom row: Inpainting with pairwise MRF, inpainting with SRF.

---

Figure from Roth And Black, 2007
Pictorial Structures

Figures from “Markov random fields for vision and image processing” (editors: Blake and Rother, 2011)
Deformable Structures

Figure 3. DS part deformations. (left) Deformations for three example parts. Black is the mean contour. Red and blue are ±1 standard deviations from the mean along the first 3 principal component directions. Stars mark the joint locations which deform with the contour. (right) Mean part shapes for the female and male body (14-part model). The dots represent joint points (see text).

Figure 4. Sampling from the DS model. (left) Two different torso shapes are outlined in black. Samples from the DS model are shown as dotted black lines. These are generated by starting with the torso and moving out along the tree structure. The red contour shows the most likely pose and shape for the parts. (right) Two more examples starting from different shapes of the upper arm (the model is rendered in the coordinate system of the arm).

Figure 5. Examples of the DS model in a variety of poses. Note how much the model’s left calf (magenta) varies in shape.

Figures from [Zuffi, Freifeld and Black, CVPR 2012]
Deformable Structures

Figure from Silvia Zuffi’s PhD Thesis, 2015
MRF on Superpixels

Figure from Freifeld, Li and Fisher, ICIP 2015