Markov Random Fields (MRF) and Conditional Random Fields (CRF)

Abhinav Moudgil
Center for Visual Information Technology (CVIT)
IIIT Hyderabad, India
Email: abhinav.moudgil@research.iiit.ac.in

Abhineet Jain
Robotics Research Center (RRC)
IIIT Hyderabad, India
Email: abhineet.jain@students.iiit.ac.in

Abstract

Probabilistic models provide a strong framework to model a sequence of data, predicting next outcomes and generating new data. We explore a set of probabilistic models which follow Markov property namely, Markov Random Fields (MRF) and Conditional Random Fields (CRF). They act as powerful tools to solve problems where regions must be recognized using a set of segregating features, to predict the class of the region. Theoretical analysis of both the models and their variants are presented. In the last section, we discuss their applications in the field of computer vision and text processing.

1. Introduction

Issues in computer vision usually involve noise, and hence exact solutions are most often impossible. But in most of the cases, the pixel values in an image have feeble relationships with those further away and usually depend more strongly on those in the immediate vicinity. Thus, the latent variables of interest (i.e. pixels here) have the Markov property. Vision problems like image restoration, segmentation, image reconstruction and edge detection are well suited to the Markov Random Field optimization technique which is introduced in Section 1 along with Conditional Random Field. In Section 2, we discuss Bayesian Theory and Markov models which lead to the formulation of MRFs and CRFs. However, these probabilistic Markov models are moulded according to the problem and hence their variants are discussed in Sections 3, 4 and 5. In Section 6, we explore a way to learn the latent structure using tree-based Conditional Random Fields. In Section 7 and 8 we see that scope of MRF-CRF framework is not just limited to probabilistic models but they are often used in non-probabilistic setting along with SVM and RNN. Finally in Section 9, applications of Markov models in the field of computer vision and text processing are discussed.

1.1. Markov Random Fields

Markov Random Field is an undirected graphical model 1 which captures the joint probability distribution. It consists of an undirected graph $G = (N, E)$ in which N represent random variables. The edges E encode the conditional independence relationships via the following rule: given disjoint subsets of nodes A, B, and C, is conditionally independent of given if there is no path from any node in A to any node in B that doesn't pass through a node of C. If such a path does exist, the subsets are dependent.

The neighbour set N_n of a node n is defined to be the set of nodes that are connected to n via edges in the graph:

$$N_n = m \in N | (n, m) \in E$$

Given its neighbour set, a node n is independent of all other nodes in the graph. Therefore, following Markov property is satisfied for the conditional probability of X_n:

$$P(X_n | X_N - X_n) = P(X_n | X_{N_n}) \tag{1}$$

The Markov property tells us that the joint distribution of X is determined entirely by the local conditional distributions . From Gibbs distributions, we construct the global joint distribution from these local functions. A Gibbs distribution on the graph G takes the form:

$$P(X = x) = \prod_{C \in G} \phi_C(x_C)$$

where the product is over all maximal cliques $\phi_C(x_C)$ in the graph. A clique is a subset of nodes in which every node is connected to every other node. A maximal clique is a clique which cannot be extended by the addition of another node. Z is called the partition function, and takes the form:

$$Z = \sum_x \prod_{C \in G} \phi_C(x_C)$$

$\phi_C(x_C)$ is expressed by:

$$\phi_C(x_C) = e^{\frac{1}{T} V_c(x_c)}$$

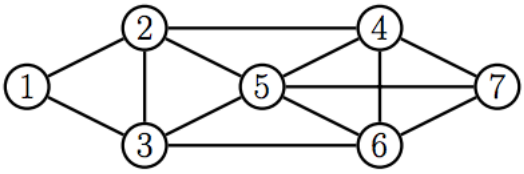

Figure 1. Markov Random Field represented as undirected graphical model. Each node represents random variable with edge denoting it's dependency. For example, node 5 depends on node 2, 3, 4, 6 and 7.

where T is temperature and often taken as 1. So $P(x)$ has the alternate form:

$$P(x) = \frac{1}{Z} e^{-\frac{1}{T}U(x)}$$

where $U(x) = \sum_{c \in C} V_c(x)$ is the called energy.

Optimization in an MRF problem involves finding the maximum of the joint probability over the graph, usually with some of the variables given by some observed data. Equivalently, as can be seen from the equations above, this can be done by minimizing the total energy, which in turn requires the simultaneous minimization of all the clique potentials.

There are ample ways for the minimization of the MRF potentials. Many of them are also relevant to optimization problems other than MRF. For example, gradient descent methods [25] are well-known techniques for finding local minima, while the closely-related method of simulated annealing attempts to find a global minimum.

Besag proposed an MRF optimization technique called Iterated Conditional Modes (ICM) [3]. Algorithm proceeds first by choosing an initial configuration for the variables. It iterates over each node in the graph and calculates the value that minimizes the energy given the current values for all the variables in its neighbourhood. At the end of an iteration, the new values for each variable become the current values, and the next iteration begins. The algorithm is guaranteed to converge, and may be terminated according to a chosen criterion of convergence.

1.2. Conditional Random Fields

A conditional random field or CRF [15] is just a version of an MRF where all the clique potentials are conditioned on input features:

$$p(\mathbf{y}|\mathbf{x}, \mathbf{w}) = \frac{1}{Z(\mathbf{x}, \mathbf{w})} \prod_{\mathbf{c}} \psi_c(\mathbf{y_c}|\mathbf{x}, \mathbf{w})$$

A CRF can be thought of as a structured output extension of logistic regression, where correlation is modelled among the output labels conditioned on the input features. We usually assume a log-linear representation of the potentials:

$$\psi_c(\mathbf{y_c}|\mathbf{x}, \mathbf{w}) = \exp(\mathbf{w_c^T} \phi(\mathbf{x}, \mathbf{y_c}))$$

where $\phi(\mathbf{x}, \mathbf{y_c})$ is a feature vector derived from the global inputs $\mathbf{x}$ and the local set of labels $\mathbf{y_c}$.

The upside of a CRF over a MRF is similar to the advantage of a discriminative classifier over a generative classifier: we can center our consideration on displaying what we think about and not all perceptions, similar to the dissemination of labels given the data. Another preferred standpoint is that we can make the potentials (or factors) of the model be data-dependent. For instance, in natural language processing issues, we can make the latent labels rely on upon global properties of the sentence, for example, which language it is composed in. It is difficult to fuse global elements into generative models.

The drawback of CRFs over MRFs is that they require labeled training data, and they take more time to train. This is practically equivalent to the qualities and shortcomings of logistic regression vs naive Bayes.

2. Bayesian Theory and MRF

Bayesian Theory forms the fundamental probabilistic structure of MRFs and CRFs and it is widely exploited in solving various computer vision problems. Gerda Kamberova [12] asserted that the Bayesian method has four major components: (i) prior model which in low-level vision tasks describes prior information about input; (ii) degradation (sensor) model which models an observed degraded input (which can be image) as a result of applying some transformation to the original input; (iii) posterior model which relates the prior and degradation models, and given the observed data, for any allowable image, it represents the probability that this input has generated the observed data; (iv) loss function which is used for expressing costs of errors and preferences to particular estimators. The objective is to find an optimal estimate with respect to the prior model, the degradation model, and the loss function, given the observed data. Finding the "best" estimate is an optimization problem which is usually intractable by traditional methods and this where the MRFs have important applications in image modeling and estimation problems.

The motivation for constructing this undirected graphical model is to connect the hidden variables associated with the nodes. For the task of segmenting an image into foreground and background, each node i (i.e pixel) has an associated random variable X_i that may take the value 0 or 1, corresponding to foreground or background, respectively. In order to represent the tendency of matter to be coherent, neighboring sites are likely to have the same label. So where $(i, j) \in E$, some kind of probabilistic bias needs to

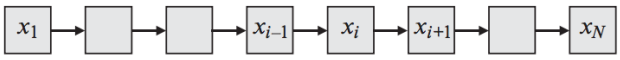

Figure 2. A simple first order Markov Chain.

be associated with the edge (i, j) such that X_i and X_j tend to have the same label both 0 or both 1. Any pixels that are nearby, not merely adjacent, are likely to have the same label. Markov models explicitly represent only the associations between relatively few pairs of pixels those pixels that are defined as neighbors because of sharing an edge in E. The great attraction of Markov Models is that they leverage a *knock-on* effect [4] that explicit short-range linkages give rise to implied long-range correlations. Thus correlations over long ranges, on the order of the diameters of typical objects, can be obtained without undue computational cost. We'll briefly introduce some fundamental probabilistic models which exploits this Markov property and lead to the formulation of MRF and CRF.

2.1. Markov Chains

It is the simplest Markov model that operates sequentially transitioning from one state to another within an allowed set of states as shown in Fig.. It consists of three elements: a state space x which is a set of values that chain is allowed to take, a transition operator $P(x_{i+1}|x_i)$ which defines probability of moving from x_i to x_{i+1} and a initial condition distribution $\pi^{(0)}$ which defines the probability of being in any one of the possible states at the initial iteration $t = 0$. The joint density can be decomposed as a product of conditional densities:

$$P(\boldsymbol{x}) = P(x_N|x_{N-1})...P(x_2|x_1)P(x_1)$$

It is *memoryless* in a sense that next state only depends on the current state and not on any of the states previous to the current and also formally known as the Markov property [1]. An alternative formalism that is commonly used is the undirected graphical model. Markov chains can also be represented in this way (figure 1.2c), corresponding to factorized decomposition:

$$P(\boldsymbol{x}) = \Phi_{N,N-1}(x_N, x_{N-1})...\Phi_{2,1}(x_2, x_1) \qquad (2)$$

Thus the Markov chain shares the elegance of Markov models generally, that long-range dependencies can be captured for the price of explicitly representing just the immediate dependencies between neighbors. Second, higher order Markov chains, where the explicit dependencies go back farther than immediate neighbors, can also be useful.

2.2. Hidden Markov Models (HMM)

Markov models are particularly useful as prior models for state variables X_i that are to be inferred from a

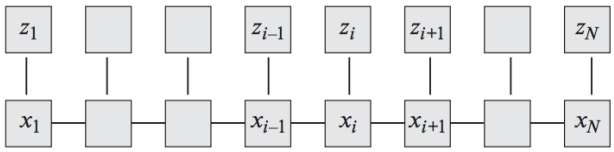

Figure 3. HMM represented as an undirected graphical model.

corresponding set of measurements or observations $z = (z_1, z_2, ..., z_i, ..., z_N)$. The observations z are themselves considered to be instantiations of a random variable $\boldsymbol{Z}$ representing the full space of observations that can arise. It leads to an inference problem in which the posterior distribution for the possible states $\boldsymbol{X}$, given the observations z, is computed via Bayess formula as:

$$P(\boldsymbol{X} = \boldsymbol{x}|\boldsymbol{Z} = \boldsymbol{z}) \propto P(\boldsymbol{Z} = \boldsymbol{z}|\boldsymbol{X} = \boldsymbol{x})P(\boldsymbol{X} = \boldsymbol{x})$$

Here $P(\boldsymbol{X} = \boldsymbol{x})$ is the prior distribution over states which is known about states $\boldsymbol{X}$ in the absence of any observations. The prior of an HMM is itself represented as a Markov chain, in which the first-order case was decomposed as a product of conditional distributions. The term $P(\boldsymbol{Z} = \boldsymbol{z}|\boldsymbol{X} = \boldsymbol{x})$ is the likelihood of the observations, which is essentially a measure of the quality of the measurements. This captures the fact that a more precise instrument produces more consistent responses z, under a given condition represented by the state $X = x$. It is often assumed that observations are independent across sites i.e. observation at site i depends only on the corresponding state. In other words:

$$P(\boldsymbol{z}|\boldsymbol{x}) = P(z_N|x_N)P(z_{N-1}|x_{N-1})...P(z_1|x_1)$$

A HMM can be expressed as an undirected graphical model, as depicted in Fig.3, in which the prior is decomposed as depicted in 2, and the likelihood is

$$P(\boldsymbol{z}|\boldsymbol{x}) = \Phi_N(x_N)\Phi_{N1}(x_{N1})...\Phi_1(x_1)$$

where $\Phi_i(x_i) = P(z_i|x_i)$.

3. Hinge Loss Markov Random Fields

Hinge Loss Markov Random Fields (HL-MRF) [2] is a new class of probabilistic models which is analogues to discrete Markov Random Fields except that random variables are continuous valued in the unit interval [0, 1] and potentials are linear or squared hinge-loss functions. They are log-linear models whose features are hinge-loss functions of the variable states. HL-MRFs are built on the theory of probabilistic soft logic models (PSL) [13], making them more expressive and interpretative.

Mathematically, let Let $\mathbf{Y} = (Y_1, ..., Y_n)$ be a vector of n variables and $\mathbf{X} = (X_1, ..., X_{\acute{n}})$ a vector of $\acute{n}$ variables with joint domain $\mathbf{D} = [0, 1]^{n+\acute{n}}$. Let $\phi = (\phi_1, ..., \phi_m)$ be m continuous potentials of the form

$$\phi_j(\mathbf{Y}, \mathbf{X}) = [\max\{l_j(\mathbf{Y}, \mathbf{X}), 0\}]^{pj}$$

where l_j is a linear function of $\mathbf{X}$ and $\mathbf{Y}$ and $p_j \in \{1, 2\}$. Let $C = (C_1, ..., C_r)$ be linear constraint functions associated with index sets denoting equality constraints ϵ and inequality constraints I, which define the feasible set

$$\tilde{\mathbf{D}} = \left\{ \mathbf{Y}, \mathbf{X} \in D \middle| \begin{array}{l} C_k(\mathbf{Y}, \mathbf{X}) = 0, \forall k \in \epsilon \\ C_k(\mathbf{Y}, \mathbf{X}) \geq 0, \forall k \in I \end{array} \right\}$$

For $\mathbf{Y}, \mathbf{X}\tilde{D}$, given a vector of non-negative free parameters, i.e., weights, $\lambda = (\lambda_1, ..., \lambda_m)$, a constrained hinge-loss energy function f_λ is defined as

$$f_\lambda(\mathbf{Y}, \mathbf{X}) = \sum_{j=1}^{m} \lambda_j \phi_j(\mathbf{Y}, \mathbf{X})$$

A hinge-loss Markov random field P over random variables $\mathbf{Y}$ and conditioned on random variables $\mathbf{X}$ is a probability density defined as follows: if $\mathbf{Y}, \mathbf{X} \notin \tilde{\mathbf{D}}$, then $P(\mathbf{Y}|\mathbf{X}) = 0$; if $\mathbf{Y}, \mathbf{X} \in \tilde{\mathbf{D}}$, then

$$P(\mathbf{Y}|\mathbf{X}) = \frac{1}{Z(\lambda)} \exp[-f_\lambda(\mathbf{Y}, \mathbf{X})],$$

where $Z(\lambda) = \int_{\mathbf{Y}} \exp[-f_\lambda(\mathbf{Y}, \mathbf{X})]$.

The potentials and weights can be grouped together into templates, which can be used to define general classes of HL-MRFs that are parameterized by the input data. Let $\Gamma = (t_1, ..., t_s)$ denote a vector of templates with associated weights $\lambda = (\lambda_1, ..., \lambda_s)$. Potentials are partitioned by their associated templates and let $\Phi_q(\mathbf{Y}, \mathbf{X}) = \sum_{j \in t_q} \phi_j(\mathbf{Y}, \mathbf{X})$ for all $tq \in \Gamma$. In the HL-MRF, the weight of the jth hinge-loss potential is set to the weight of the template from which it was derived, i.e., $\lambda_j = \lambda_q$, for each $j \in t_q$.

3.1. MPE Inference for HL-MRFs

MPE inference for HL-MRFs requires finding a feasible assignment that minimizes f_λ. Performing MPE inference quickly is crucial, especially because weight learning often requires performing inference many times with different weights. Here, HL-MRFs have a distinct advantage over general discrete models, since minimizing f_λ is a convex optimization rather than a combinatorial one. In this section, an efficient MPE inference algorithm for HL-MRFs is discussed.

The algorithm starts by y initializing local copies of the variables that appear in each potential and constraint, along with a corresponding Lagrange multiplier for each copy. Then, until convergence, it iteratively updates Lagrange multipliers and solves suproblems induced by the HL-MRFs potentials and constraints. If the subproblem is induced by a potential, it sets the local variable copies to a balance between the minimizer of the potential and the emerging consensus. Eventually the Lagrange multipliers will enforce agreement between the local copies and the consensus. If instead the subproblem is induced by a constraint in the HL-MRF, the algorithm projects the consensus variables current values to that constraints feasible region. The consensus variables are updated at each iteration based on the values of their local copies and corresponding Lagrange multipliers, and clipped to [0,1].

Algorithm 1 MPE Inference for HL-MRFs

Input: HL-MRF$(\mathbf{Y}, \mathbf{X}, \phi, \lambda, C, \epsilon, I), \rho > 0$
Initialize $\mathbf{y}_j$ as copies of the variables $\mathbf{Y}_j$ that appear in ϕ_j, j = 1,...,m.
Initialize $\mathbf{y}_{k+m}$ as copies of the variables $\mathbf{Y}_{k+m}$ that appear in C_k, k = 1,...,r.
Initialize Lagrange multipliers α_i corresponding to variable copies $\mathbf{y}_i$, i = 1, . . . , m + r
while not converged **do**
  for $j = 1, ..., m$ **do**
    $\alpha_j \leftarrow \alpha_j + \rho(\mathbf{y_j} - \mathbf{Y_j})$
    $\mathbf{y}_j \leftarrow \mathbf{Y}_j - \alpha_j/\rho$
    if $\ell(\mathbf{y}_j, \mathbf{X}) > 0$ **then**

$$\mathbf{y}_j \leftarrow \arg\min_{\mathbf{y}_j} \begin{bmatrix} \lambda_j[\ell_j(\mathbf{y_j}, \mathbf{X})]^{p_j} \\ \frac{\rho}{2}\|\mathbf{y_j} - \mathbf{Y_j} + \frac{1}{\rho}\alpha_j\|_2^2 \end{bmatrix}$$

      if $\ell(\mathbf{y}_j, \mathbf{X}) < 0$ **then**
        $\mathbf{y}_j \leftarrow \text{Proj}_{\ell_j=0}(\mathbf{Y_j})$
      end if
    end if
  end for
  for $k = 1, ..., r$ **do**
    $\alpha_{k+m} \leftarrow \alpha_{k+m} + \rho(\mathbf{y}_{k+m} - \mathbf{Y}_{k+m})$
    $\mathbf{y}_{k+m} \leftarrow \text{Proj}_{C_k}(\mathbf{Y}_{k+m})$
  end for
  for $i = 1, ..., n$ **do**
    $Y_i \leftarrow \frac{1}{|\text{copies}(Y_i)|} \sum_{y_c \in \text{copies}(Y_i)} \left(y_c + \frac{\alpha_c}{\rho}\right)$
    Clip Y_i to [0, 1]
  end for
end while

3.2. Weight Learning

The canonical approach for learning parameters λ is to maximize the log-likelihood of training data. The partial derivative of the log-likelihood with respect to a parameter

λ_q is

$$\frac{\partial \log p(\mathbf{Y}|\mathbf{X})}{\partial \Lambda_q} = \mathbb{E}_\Lambda[\Phi_q(\mathbf{Y},\mathbf{X})] - \Phi_q(\mathbf{Y},\mathbf{X}),$$

where $\mathbb{E}_\lambda$ is the expectation under the distribution de- fined by λ. The voted perceptron algorithm [5] optimizes λ by taking steps of fixed length in the direction of the gradient, then averaging the points after all steps. Any step that is outside the feasible region is projected back before continuing. For a smoother ascent, it is often helpful to divide the q-th component of the gradient by the number of groundings $|t_q|$ of the qth template [18].

4. Variants of CRFs

4.1. High-order Semi-CRFs

A conditional version which uses properties of labels before the suffix labels being considered, making extension to the high-order semi-Markov features simpler. [6] A semi-CRF defines a conditional distribution over all possible segmentations $\mathbf{s}$ of an input sequence $\mathbf{x}$ such that:

$$P(\mathbf{s}|\mathbf{x}) = \frac{1}{Z_\mathbf{x}} \exp\big(\sum_{i=1}^{m}\sum_{t=1}^{|\mathbf{s}|}\lambda_i f_i(\mathbf{x},\mathbf{s},t)\big)$$

where $Z_\mathbf{x} = \sum_\mathbf{s} \exp(\sum_i \sum_t \lambda_i f_i(\mathbf{x},\mathbf{s},t))$ is the partition function with the summation over all segmentations of $\mathbf{x}$, and $\{f_i(\mathbf{x},\mathbf{s},t)\}_{1\le i\le m}$ is the set of semi-Markov features, each of which has a corresponding weight. Features can be defined as

$$f_i(\mathbf{x},\mathbf{s},t) = \left\{ \begin{array}{ll} g_i(\mathbf{x},u_t,v_t) & \text{if} y_{t-|\mathbf{z}^i|+1}...y_t = \mathbf{z}^i \\ 0 & \text{otherwise} \end{array} \right\}$$

where $\mathbf{z}_i \in \mathcal{Y}^{|\mathbf{z}_i|}$ is a segment label pattern associated with f_i, and $\mathbf{x}$ is a segmentation or a partial segmentation for $\mathbf{x}$. The function $f_i(\mathbf{x},\mathbf{s},t)$ depends on the t-th segment as well as the label pattern $\mathbf{z}^i$ and is said to be of order $|\mathbf{z}^i|-1$. The order of the resulting semi-CRF is the maximal order of the features.

4.1.1 Training

Given a training set $\mathcal{T}$, the model parameters $\vec{\lambda} = (\lambda_1,...,\lambda_m)$ are estimated by maximizing the regularized log-likelihood function

$$\mathcal{L}_\mathcal{T}(\vec{\lambda}) = \sum_{(\mathbf{x},\mathbf{s})\in\mathcal{T}} \log P(\mathbf{s}|\mathbf{x}) - \sum_{i=1}^{m}\frac{\lambda_i^2}{2\sigma_{reg}^2}$$

where σ_{reg} is a regularization parameter. This function is convex, and thus can be maximized using any convex optimization algorithm. L-BFGS method [17] is used, which requires computation of the value of $\mathcal{L}_\mathcal{T}(\vec{\lambda})$ and its partial derivatives.

4.1.2 Decoding

The most likely segmentation for high-order semi-CRF is computed by a Viterbi-like decoding algorithm. It is the same as the forward algorithm with the sum operator replaced by the max operator. Define

$$\delta_\mathbf{x}(j,\mathbf{p}^\mathbf{i}) = \max_{\mathbf{s}\in\mathbf{P}_{j,\mathbf{P}^i}} \exp(\sum_k\sum_t \lambda_k f_k(\mathbf{x},\mathbf{s},t))$$

The most likely segmentation can be obtained using backtracking from $\max_i \delta_\mathbf{x}(|\mathbf{x}|,\mathbf{p}^i)$.

4.2. Latent Dynamic Conditional Random Fields

The task is to learn a mapping between a sequence of observations $\mathbf{x} = x_1, x_2, ..., x_m$ and a sequence of labels $\mathbf{y} = y_1, y_2, ..., y_m$. Each y_j is a class label for the jth token of a observation sequence and is a member of a label set, Y. For each sequence, we also assume a vector of latent variables $\mathbf{h} = h_1, h_2, ..., h_m$, which is not observable. To keep the training efficient, we restrict the model to have a disjointed set of latent variables associated with each class label; Each h_j is a member of a set, $\mathbf{H}_{y_j}$, which represents the possible latent variables for the class label y_j. Following the problem definition described above, LDCRFs are defined as [19]:

$$P(\mathbf{y}|\mathbf{x},\Theta) = \sum_\mathbf{h} P(\mathbf{y}|\mathbf{h},\mathbf{x},\Theta)P(\mathbf{h}|\mathbf{x},\Theta)$$

where Θ represents the parameters of the model. Since sequences with $\mathbf{h}_j \notin \mathbf{H}_{y_j}$ will have $P(\mathbf{y}|\mathbf{x},\Theta) = 0$ by the restriction of disjointed association, the model can be further defined as [19]:

$$P(\mathbf{y}|\mathbf{x},\Theta) = \sum_{\mathbf{h}\in\mathbf{H}_{y_1}\times...\times\mathbf{H}_{y_m}} P(\mathbf{h}|\mathbf{x},\Theta)$$

where $P(\mathbf{h}|\mathbf{x},\Theta)$ is defined using the usual conditional random field formulation:

$$P(\mathbf{h}|\mathbf{x},\Theta) = \frac{\exp\Theta\cdot\mathbf{f}(\mathbf{h},\mathbf{x})}{\sum_{\forall\mathbf{h}}\exp\Theta\cdot\mathbf{f}(\mathbf{h},\mathbf{x})}$$

in which $\mathbf{f}(\mathbf{h},\mathbf{x})$ is the global feature vector. Given training sequences $(\mathbf{x_i},\mathbf{y_i})$ for $i = 1...d$, training is performed by optimizing the likelihood of the training data, usually with a regularization of $R(\Theta)$:

$$L(\Theta) = \sum_{i=1}^{d}\log P(\mathbf{y_i^*}|\mathbf{x_i},\Theta) - R(\Theta)$$

5. Sequential Deep Belief Networks

A few techniques have been proposed to use hidden variables with CRFs that may be equipped for displaying regularities in the data that are not expressed in the features but rather guide in classification.

The hidden CRF (HCRF) affixs a multinomial hidden state to every phone class and upgrades the marginal likelihood [9], with the goal that subclasses might be induced that are simpler to perceive than the unique classes. Another effective approach models every phone as a succession of three sub-phones, the limits of which are latent [24]. Other work utilizes a multi-layer CRF as a part of which the data is mapped through different layers of multinomial arrangements that might be either order-1, or order-0 Markov (that is, conditionally independent given the input) [27]. These methodologies are more powerful than a CRF with no latent structure, however better results have been acquired with a stronger latent representation for static classification problems with Deep Belief Networks (DBNs).

Deep Belief Networks [10] have risen as an exactly viable model for inciting rich element representations of static (nonsequential) information. Every layer of latent representation is found out by training a Restricted Boltzmann Machine (RBM) to show the data distribution at the following lower layer, utilizing e.g., Contrastive Divergence. In this paper [1], a novel structure called a successive RBM (SRBM) is presented that permits dependencies between comparing latent units at adjoining time spans in the hidden layer. Correct sampling of hidden structures given the input and calculation of conditional expectations stays tractable in the SRBM, it includes just matrix multiplication and conventional forward-backward calculations, so CD training is conceivable. SRBMs are stacked and added to a sequence classifier (i.e. CRF). The expectation is to give the model a chance to implement smoothness in the hidden layers crosswise over time periods, and to permit the hidden variables to conceivably model longerrange wonders. At the end, the parameters of all the layers are fine-tuned utilizing a back-propagation-like algorithm. The subsequent model is known as a Sequential DBN (SDBN).

6. Learning Tree CRFs

In this section, we discuss maximum spanning tree-based strategies for learnin the structure of tree Conditional Random Fields (CRFs). citebradley2010learning. Edge weights are utilized which exploit local inputs, and along these lines scale to extensive issues.

Low-treewidth CRFs: CRFs for which the yields $\mathcal{Y}$ shape a low-treewidth graphare a key exemption, allowing tractable inference and parameter learning [20]. Let $\mathcal{X}$ be a set of input variables and $\mathcal{Y}$ a set of output variables.

If the CRF's graph (over $\mathcal{Y}$, not $\mathcal{X}$) is low-treewidth, inference and parameter learning are tractable. There are three learning tasks: feature selection, structure learning, and parameter learning.

Feature selection means to select inputs X_{C_j} relevant to each set Y_{C_j}; i.e., a factor ϕ_j involving Y_{C_j} can only use the selected inputs: $\phi_j = \phi_j(Y_{C_j}, X_{C_j})$.

Structure learning means to select outputs Y_{C_j} for each factor, thus choosing a graph over $\mathcal{Y}$ representing conditional independence in the distribution.

Parameter learning means choosing the values of the factors $\phi_j(Y_{C_j}, X_{C_j})$, where the factor domain $Y_{C_j} \cup X_{C_j}$ is fixed. This has been well-studied for both tractable structures and intractable.

6.1. The Standard Approach

Let $Y_{ij} \equiv \{Y_i, Y_j\}$. The inputs for Y_i specified by the input mapping are X_i; likewise, X_{ij} corresponds to Y_{ij}. The conditional log likelihood of a tree CRF decomposes over the edges (i, j) and vertices i in the tree T. Let Q be our model and P the true distribution. Using the (optimal) parameters $Q(A|B) = P(A|B)$,

$$\begin{aligned} E_P[\log Q(\mathcal{Y}|\mathcal{X})] &= \sum_{ij \in T} E_P[\log Q(Y_{ij}|\mathcal{X})] \\ &\quad - \sum_i (deg_i - 1) E_P[\log Q(Y_i|\mathcal{X})] \\ &= \sum_{ij \in T} I_P(Y_i, Y_j|\mathcal{X}) + C \end{aligned}$$

where subscript P means w.r.t. P, deg_i is the degree of vertex i, and C is a constant w.r.t. the structure.

Assuming the global Conditional Mutual Information (CMI) $I(Y_i; Y_j|\mathcal{X})$ is easy to compute, we can recover the maximum likelihood model: $\forall(i, j)$, compute $I(Y_i; y_j|\mathcal{X})$, and choose the maximum spanning tree. This method was proposed by Friedman et al. (1997) [8] for learning Tree-Augmented Naive Bayes classifiers.

Unfortunately, as the dimensionality of $\mathcal{X}$ increases, this strategy rapidly gets to be intractable. This perception underlines the need to parametrize our model with local inputs $\mathcal{X}_i \subset \mathcal{X}$, as opposed to global inputs $\mathcal{X}_i = \mathcal{X}$, to guarantee scalability w.r.t. $\mathcal{X}$, ie. just ascertaining probabilities of the form $P(Y_{ij}|X_{ij})$ conditioned on small sets X_{ij}. With local inputs, however, the partition function keeps the conditional log likelihood from breaking down over edges and vertices:

$$\log P(\mathcal{Y}|\mathcal{X}) = -\log Z(\mathcal{X}) + \sum_{(i,j) \in T} \log \phi_{ij}(Y_{ij}, X_{ij})$$

Our essential objective is to conquer the intractability of the partition function by inferring an important local edge score $Score(i, j)$ usable in Algorithm 2.

6.2. Score Decay Assumption

Definition If the true model $P(\mathcal{Y}|\mathcal{X})$ is a tree T, the Score Decay Assumption (SDA) states that, for any edge $(i, j) \in T$, if a path in T of $length > 1$ between k, l includes (i, j), then $Score(i, j) > Score(k, l)$.

Algorithm 2 Tree CRF Structure Learner

input Dataset D over $\mathbf{y}, \mathbf{x}$; inputs $\mathbf{x}_i$ for each $\mathbf{y}_i$
Initialize G to be the complete graph over y.
for all $(\mathbf{y}_i, \mathbf{y}_j)$ **do**
  In G, set Weight$(\mathbf{y}_i, \mathbf{y}_j) \longleftarrow$ Score(i, j)
end for
return $MaxSpanningTree(G)$

Intuitively, the SDA says the score between vertex pairs decays with distance. However this is less strict than requiring, e.g., that the presumption hold regardless of whether (i, j) is an edge in T; hence, the SDA does not depend on comparing pairs of edges not in T. This condition is necessary and sufficient for recovering trees. The following theorem follows.

Theorem 1 *Suppose $P(\mathcal{Y}|\mathcal{X})$ is representable by a tree CRF with structure T. The Score Decay Assumption holds for a score S w.r.t. P iff Algorithm 2 using score S can recover T.*

Local scores $S(i, j) = (Y_{ij}, X_{ij})$ should be used for scalability. Simplifying assumption about input mapping: a factor involving Y_{ij} depends on $X_i \cap X_j$. A general class of such scores is defined as:

Definition The *Local Linear Entropy Scores* are scores $Score(i, j)$ representable as linear combinations of entropies over subsets of Y_i, Y_j, X_i, X_j.

This class of scores includes, for example, the local Conditional Mutual Information $I(Y_i; Y_j|X_{ij}) = H(Y_i|X_{ij}) + H(Y_j|X_{ij}) - H(Y_{ij}|X_{ij})$. Unfortunately, the Local Linear Entropy Scores are insufficient in general for recovering tree CRFs.

6.3. Heuristic Scores

6.3.1 Piecewise Likelihood

Piecewise likelihood approximates the log likelihood by upper-bounding $Z(\mathcal{X})$. The bound effectively divides $Z(\mathcal{X})$ into one term for each factor. For pairwise models, if we combine factors with their Z terms, piecewise likelihood becomes $\sum(i, j) \log P(Y_{ij}|X_{ij})$. This Local Linear Entropy Score is usable in Algorithm 2: $S(i, j) = E_P[\log P(Y_{ij}|X_{ij})]$.

However, piecewise likelihood is a poor edge score. If a pair (Y_i, X_i) share a strong potential, piecewise likelihood weights will likely result in a star structure over $\mathcal{Y}$ with Y_i at the center.

6.3.2 Local CMI

The first Local Linear Entropy Score can be a local version of global CMI $I(Y_i; Y_j|\mathcal{X})$. The local CMI is:

$$I(Y_i; Y_j|X_{ij}) = E_P[\log P(Y_{ij}|X_{ij})] - E_P[\log P(Y_i|X_{i,j}) + \log P(Y_j|X_{i,j})]$$

The local CMI score is interpretable as the piecewise likelihood (first line), minus correction terms for the edges endpoints (second line).

6.3.3 Decomposable Conditional Influence

To overcome the weaknesses of the above scoring criterion, Decomposable Conditional Influence is defined as:

$$DCI(i, j) \equiv E_P[\log P(Y_{ij}|X_{ij})] - E_P[\log P(Y_i|X_i) + \log P(Y_j|X_j)]$$

The first expectation is the piecewise likelihood (as for local CMI), but the second has terms equal to the edges endpoints scores in the disconnected model $P_disc(\mathbf{y}|\mathbf{x}) \equiv Q_i P(\mathbf{y}_i|\mathbf{x}_i)$, giving the following result:

Proposition *When building a spanning tree T, if we add edge (i, j) and T does not yet contain edges adjacent to i, j, then DCI is an exact measure of the likelihood gain from adding edge (i, j).*

DCI performs very well in practice.

7. CRF as RNN

In this section, we'll discuss a formulation which combines the strengths of both CNNs and CRF based graphical models in one unified framework. It's an efficient deep learning solution for the pixel-level semantic image segmentation problem. Mean-field approximate inference for the dense CRF with Gaussian pairwise potentials is formulated as a Recurrent Neural Network (RNN) which can refine coarse outputs from a traditional CNN in the forward pass, while passing error differentials back to the CNN during training. Hence, the whole deep network, which comprises of a traditional CNN and an RNN for CRF inference, can be trained utilizing the usual back-propagation algorithm. The individual steps of a mean field algorithm [14] is described as CNN layers. Mean field algorithm in dense CRF can be broken down into five components namely, Initialization, Message Passing, Weighting Filter Outputs, Compatibility Transform and Normalization. Using this deduction, repeated mean-field iterations is formulated as an RNN [28].

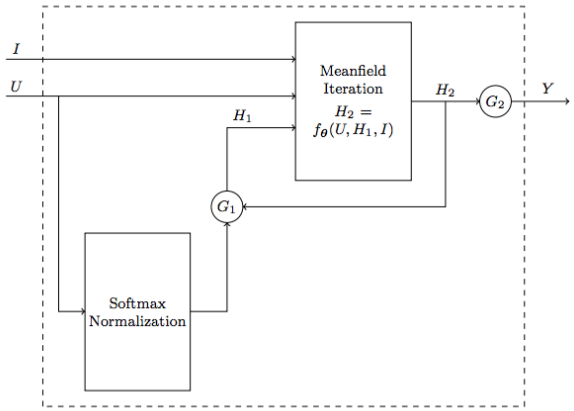

Figure 4. The CRF-RNN Network. Gating functions G_1 and G_2 are fixed.

7.1. Formulation

Let function f_θ to denote the transformation done by one mean-field iteration: given an image I, pixel-wise unary potential values U and an estimation of marginal probabilities $Q_i n$ from the previous iteration, the next estimation of marginal distributions after one mean-field iteration is given by $f_\theta(U, Q_i n, I)$. The vector $\boldsymbol{\theta} = \{w^{(m)}, \mu(l, \acute{l})\}, m \in \{1, ..., M\}, l, \acute{l} \in \{l_1, ..., l_L\}$ represents the CRF parameters. Multiple mean-field iterations can be implemented by repeating the above stack of layers in such a way that each iteration takes Q value estimates from the previous iteration and the unary values in their original form. This is equivalent to treating the iterative mean-field inference as a Recurrent neural Network (RNN) as shown in Fig.4. The behaviour of the network is given by the following equations where T is the number of mean-field iterations:

$$H_1(t) = \begin{cases} \text{softmax}(U), & t = 0 \\ H_2(t-1), & 0 < t \leq T \end{cases}$$

$$H_2(t) = f_{\boldsymbol{\theta}}(U, H_1(t), I), \ 0 \leq t \leq T,$$

$$Y(t) = \begin{cases} 0, & 0 \leq t < T \\ H_2(t), & t = T \end{cases}$$

Parameters of the CRF-RNN are the same as the mean-field parameters and denoted by θ here. Error differentials w.r.t. these parameters can be learnt in the RNN setting using the standard back-propagation through time algorithm [26]. Usually after five iterations, results doesn't improve significantly. So it does not suffer from the vanishing and exploding gradient problem inherent to deep RNN.

7.2. Training

This approach comprises of a fully convolutional network stage, which predicts pixel-level labels without considering structure, followed by a CRF-RNN stage, which performs CRF-based probabilistic graphical modelling for structured prediction. The complete system is trained using the backpropagation algorithm [16] and the Stochastic Gradient Descent (SGD) procedure. During training, a whole image (or many of them) can be used as the mini-batch and the error at each pixel output of the network is computed using an appropriate loss function such as the softmax loss with respect to the ground truth segmentation of the image. FCN-8s architecture of [21] is used as the first part of our network, which provides unary potentials to the CRF. This network is based on the VGG-16 network [22] but has been restructured to perform pixel-wise prediction instead of image classification.

In the forward pass through the network, once the computation enters the CRF-RNN after passing through the CNN stage, it takes T iterations for the data to leave the loop created by the RNN. Neither the CNN that provides unary values nor the layers after the CRF-RNN (i.e., the loss layers) need to perform any computations during this time since the refinement happens only inside the RNNs loop. Once the output Y leaves the loop, next stages of the deep network after the CRF-RNN continue the forward pass. A softmax loss layer directly follows the CRF-RNN and terminates the network. During the backward pass, once the error differentials reach the CRF-RNNs output Y, they similarly spend T iterations within the loop before reaching the RNN input U in order to propagate to the CNN which provides the unary input. In each iteration inside the loop, error differentials are computed inside each component of the mean-field iteration.

8. Combining SVM and CRFs

This section talks about a two stage approach [11] to learning a sequence classifier that is exceptionally precise; scalable, and simple to use in data mining applications. The two-phase approach joins support vector machines (SVMs) and conditional random fields (CRFs). It is highly accurate profoundly precise in light of the fact that it profits by the maximum margin nature of SVMs and furthermore from the capacity of CRFs to model correlations between neighboring output tags. It is scalable in light of the fact that the input to each SVM is a small training set, and the input to the CRF has a small number of features - the SVM outputs.

In this approach, first SVMs are trained to predict the label of each input sequence element; this is a standard multiclass supervised learning task. Second, one CRF is trained to predict the output arrangement of labels utilizing as its input the outputs from the already trained SVMs. The in-

stinct is that both learning methodologies are to some degree orthogonal in their favorable circumstances, so a blend of them can yield much better result.

8.1. Multiclass SVM

For multiclass classification SVMs can be used in either one-against-all or one-against-one styles. With the one-against-all strategy, each class is trained independently against the union of all other classes. Applying the trained SVMs on a test data point (x_{ij}, y_{ij}) outpus a vector of predictions $(g_1, g_2, ..., g_c)_{ij}$, where c is the number of classes. Using the one-against-one strategy, each class is trained separately against each other class. Applying the trained SVMs to the test data point outputs a vector of predictions $(g_1, g_2, ..., g_b)_{ij}$ where $b = c(c-1)/2$.

8.2. Applying Conditional Random Fields

Given a dataset of input and output sequences (X, Y), the training objective for a CRF model is to select weights W that maximize the conditional log likelihood $\log P(Y|X; W)$, which is

$$\sum_{(\vec{x_i}, \vec{y_i}) \in (X,Y)} \log \frac{\exp \sum_{z=1}^{d} w_z F_z(\vec{x_i}, \vec{y_i})}{\sum_{\vec{y'}} \exp \sum_{z=1}^{d} w_z F_z(\vec{x_i}, \vec{y'})}$$

where F_z denote the various high level feature functions.

As is customary with CRFs, maximize a regularized version of the conditional log likelihood, that is

$$J(X, Y) = \log P(Y|X; W) + \log P(W)$$

where $\log P(W) = -\frac{\|W\|_2}{2\sigma^2}$. The objective function is maximized by gradient descent.

9. Applications of MRFs and CRFs

9.1. Range Sensing

The input to an MRF occurs at two layers, through the variables labeled x_i and the variables labeled z_i. The variables x_i correspond to the image pixels, and their values are the three-dimensional RGB value of each pixel. The variables z_i are the range measurements.

The key variables in this MRF are the ones labeled y, which model the reconstructed range at the same resolution as the image pixels. These variables are unobservable. Additional nodes labeled u and w leverage the image information into the estimated depth map y. Specifically, the MRF is defined through the following potentials:

1. The depth measurement potential is of the form

$$\Psi = \sum_{i \in L} k(y_i - z_i)^2$$

Here L is the set of indexes for which a depth measurement is available, and k is a constant weight placed on the depth measurements. This potential measures the quadratic distance between the estimated range in the high-res grid y and the measured range in the variables z, where available.

2. A depth smoothness prior is expressed by a potential of the form

$$\Phi = \sum_{i} \sum_{j \in N(i)} w_{ij}(y_i - y_j)^2$$

Here $N(i)$ is the set of nodes adjacent to i. Φ is a weighted quadratic distance between neighboring nodes.

3. The weighting factors w_{ij} are a key element, in that they provide the link to the image layer in the MRF. Each w_{ij} is a deterministic function of the corresponding two adjacent image pixels, which is calculated as follows:

$$w_{ij} = \exp(-cu_{ij})u_{ij} = \|x_i - x_j\|_2^2$$

Here c is a constant that quantifies how unwilling we are to have smoothing occur across edges in the image. The resulting MRF is now defined through the constraints Ψ and Φ. The conditional distribution over the target variables y is given by an expression of the form

$$p(y|x, z) = \frac{1}{Z} \exp(-\frac{1}{2}(\Psi + \Phi))$$

where Z is a normalizer (partition function).

9.2. Named Entity Recognition

One application of CRFs is named entity recognition [7]. This is used for information extraction. A simple approach to this is to use a chain-structured CRF, but to expand the state space from BIO to B-Per, I-Per, B-Loc, I-Loc, and Other. We can get better performance by considering long-range correlations between words. For example, we might add a link between all occurrences of the same word, and force the word to have the same tag in each occurrence. (The same technique can also be helpful for resolving the identity of pronouns.) This is known as a skip-chain CRF. See Fig. 5 for an illustration.

9.3. Stereo Vision

A classic low-level vision problem is dense stereo reconstruction, where the goal is to estimate the depth of every pixel given two images taken from slightly different angles. In this section (based on [23]), we study the the application of a simple CRF to tackle this issue.

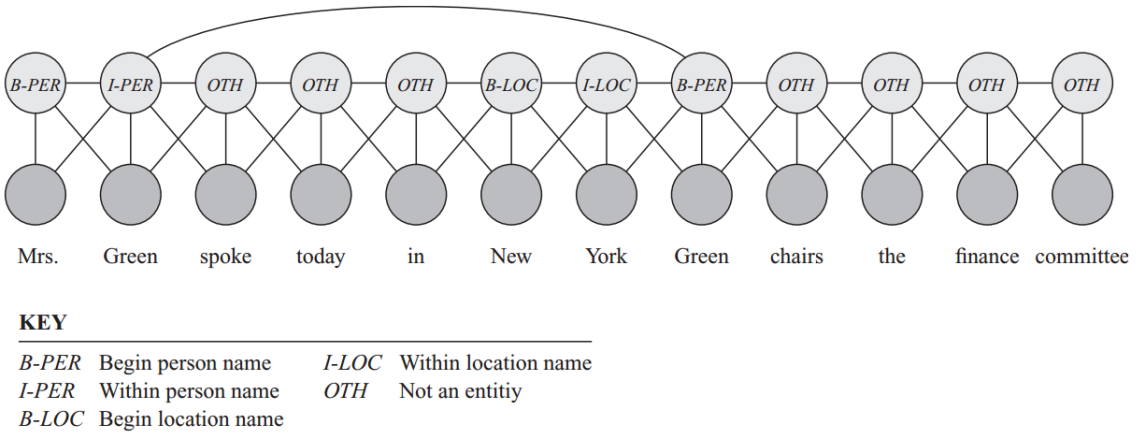

Figure 5. A skip-chain CRF for named entity recognition.

By using some standard preprocessing techniques, the depth estimation problem can be transformed to calculate the **disparity** y_s, between the pixel at location (i_s, j_s) in the left image and the corresponding pixel at location $(i_s + y_s, j_s)$ in the right image. Assuming similar intensity of corresponding pixels, define a local node potential of the form

$$\psi_s(y_s|\mathbf{x}) \propto \exp\left\{ -\frac{1}{2\sigma^2}(x_L(i_s, j_s) - x_R(i_s + y_s, j_s))^2 \right\}$$

where x_L is the left image and x_R is the right image.

We can easily add a Gaussian prior on the edges of the MRF that encodes the assumption that neighboring disparities y_s, y_t should be similar, as follows:

$$\psi_{st}(y_s, y_t) \propto \exp\left(-\frac{1}{2\gamma^2}(y_s - y_t)^2 \right)$$

The resulting model is a Gaussian CRF. We use truncated Gaussian potential of the form

$$\psi_{st}(y_s, y_t) \propto \exp\left\{ -\frac{1}{2\gamma^2} min\big((y_s - y_t)^2, \delta_0^2\big) \right\}$$

where γ encodes the expected smoothness, and δ_0 encodes the maximum penalty that will be imposed if disparities are significantly different. This is called a discontinuity preserving potential.

10. Conclusion

We began with the essential Bayesian theory and obtained the formulation of Markov Random Fields and Conditional Random Fields which are effective models to encode relationships between observations and develop consistent interpretations. We explored the concept of Markov and Conditional Random Field criteria being exploited with non probabilistic models like Support Vector Machines and Recurrent Neural Networks. Various variants of MRFs and CRFs are studied and their applications in the field of computer vision and text processing are discussed in detail.

References

[1] G. Andrew and J. Bilmes. Sequential deep belief networks. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4265–4268. IEEE, 2012.

[2] S. Bach, B. Huang, B. London, and L. Getoor. Hinge-loss markov random fields: Convex inference for structured prediction. *arXiv preprint arXiv:1309.6813*, 2013.

[3] J. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 259–302, 1986.

[4] A. Blake, P. Kohli, and C. Rother. *Markov Random Fields for Vision and Image Processing*. MIT Press, 2011.

[5] M. Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 1–8. Association for Computational Linguistics, 2002.

[6] N. V. Cuong, N. Ye, W. S. Lee, and H. L. Chieu. Conditional random field with high-order dependencies for sequence labeling and segmentation. *Journal of Machine Learning Research*, 15:981–1009, 2014.

[7] J. R. Finkel, T. Grenager, and C. Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 363–370. Association for Computational Linguistics, 2005.

[8] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine learning*, 29(2-3):131–163, 1997.

[9] A. Gunawardana, M. Mahajan, A. Acero, and J. C. Platt. Hidden conditional random fields for phone classification. In *INTERSPEECH*, pages 1117–1120, 2005.

[10] G. E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[11] G. Hoefel and C. Elkan. Learning a two-stage svm/crf sequence classifier. In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 271–278. ACM, 2008.

[12] G. Kamberova. Markov random field models: a bayesian approach to computer vision problems. *Technical Reports (CIS)*, page 491, 1992.

[13] A. Kimmig, S. Bach, M. Broecheler, B. Huang, and L. Getoor. A short introduction to probabilistic soft logic. In *Proceedings of the NIPS Workshop on Probabilistic Programming: Foundations and Applications*, pages 1–4, 2012.

[14] V. Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. *Adv. Neural Inf. Process. Syst*, 2011.

[15] J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[16] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[17] D. C. Liu and J. Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3):503–528, 1989.

[18] D. Lowd and P. Domingos. Efficient weight learning for markov logic networks. In *European Conference on Principles of Data Mining and Knowledge Discovery*, pages 200–211. Springer, 2007.

[19] L.-P. Morency, A. Quattoni, and T. Darrell. Latent-dynamic discriminative models for continuous gesture recognition. In *2007 IEEE conference on computer vision and pattern recognition*, pages 1–8. IEEE, 2007.

[20] D. Shahaf and C. Guestrin. Learning thin junction trees via graph cuts. In *AISTATS*, pages 113–120, 2009.

[21] E. Shelhamer, J. Long, and T. Darrell. Fully convolutional networks for semantic segmentation. 2016.

[22] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[23] E. B. Sudderth and W. T. Freeman. Signal and image processing with belief propagation. *IEEE Signal Processing Magazine*, 25(2):114, 2008.

[24] Y.-H. Sung and D. Jurafsky. Hidden conditional random fields for phone recognition. In *ASRU*, pages 107–112, 2009.

[25] S. Vishwanathan, N. N. Schraudolph, M. W. Schmidt, and K. P. Murphy. Accelerated training of conditional random fields with stochastic gradient methods. In *Proceedings of the 23rd international conference on Machine learning*, pages 969–976. ACM, 2006.

[26] P. J. Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[27] D. Yu, S. Wang, and L. Deng. Sequential labeling using deep-structured conditional random fields. *IEEE Journal of Selected Topics in Signal Processing*, 4(6):965–973, 2010.

[28] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr. Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1529–1537, 2015.