TTIC 31230, Fundamentals of Deep Learning

David McAllester, Winter 2018

Loopy Graphical Models

The Big Picture I

Conditional vs. Unconditional

$$\Phi^* = \operatorname*{argmin}_{\Phi} E_{(x,y)\sim \mathrm{Pop}} \; -\ln \; P(y|x)$$

$$\Phi^* = \operatorname*{argmin}_{\Phi} E_{y\sim \mathrm{Pop}} \; -\ln \; P(y)$$

This is a non-distinction: the issues in the to the conditional case are exactly the same as in the unconditional case.

The Big Picture II

The structured case: $y \in \mathcal{Y}$ where $\mathcal{Y}$ is discrete but iteration over $\hat{y} \in \mathcal{Y}$ is infeasible.

Graphical models rely on assumptions about the structure of $P_\Phi(y)$.

Loopy Graphical Models can use weaker assumptions than those supporting exact computation of $P_\Phi(y)$.

For loopy graphical Models there are various methods of performing approximate gradient descent on cross-entropy loss.

Colorization — a Self-Supervised Problem

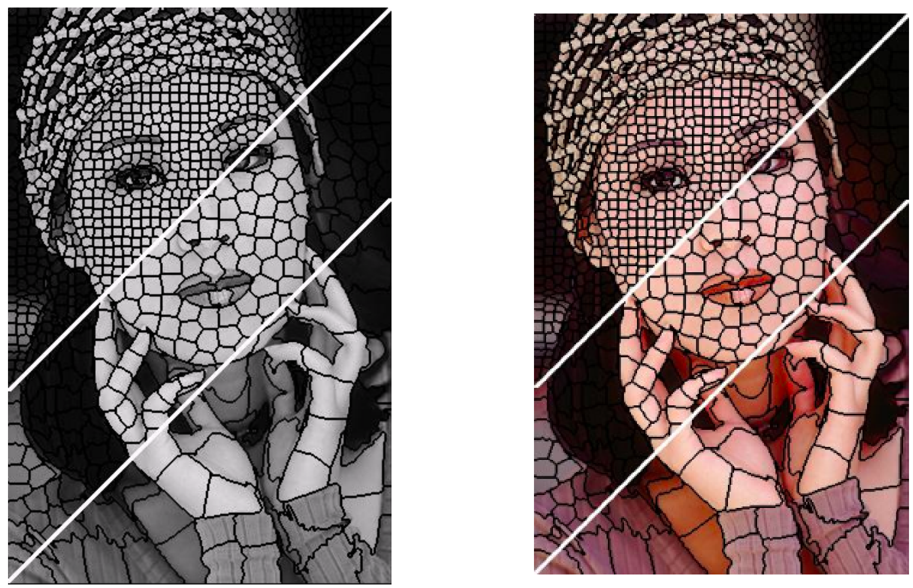
x is a black and white image.

y is a "color" image drawn from $\mathrm{Pop}(y|x)$ where the color has been rounded to one of, say, 100 color words.

$\hat{y}$ is an arbitrary color image.

$P_\Phi(\hat{y}|x)$ is the probability that model Φ assigns to the color image $\hat{y}$ given black and white image x.

Colorizing Superpixels



SLIC superpixels, Achanta et al.

$\hat{y}[i]$ is the color value of superpixel i in color image $\hat{y}$.

$\hat{y}[(i,j)]$ is the pair $(\hat{y}[i], \hat{y}[j])$ for neighbors i and j.

Exponential Softmax

$$P_\Phi(\hat{y}|x) \;=\; \operatorname*{softmax}_{\hat{y}}\; s_\Phi(\hat{y}|x)$$

Let $\mathcal{C}$ be colors, $\mathcal{I}$ be superpixels, and $\mathcal{E}$ be edges.

We will compute

a unary potential tensor $s_i[c] = s_\Phi(c|x,i)$

a binary potential tensor $s_e[c,c'] = s_\Phi(c,c'|x,e)$

$$s_\Phi(\hat{y}|x) = \sum_{i\in\mathcal{I}} s_i[\hat{y}[i]] + \sum_{e\in\mathcal{E}}\; s_e[\hat{y}[e.i],\hat{y}[e.j]]$$

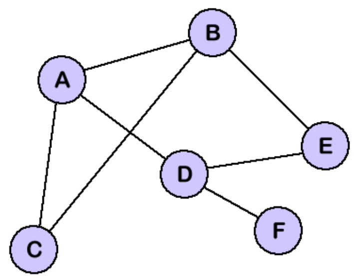
Backpropagation

The input is the image x and the parameter package Φ

$$\begin{aligned}
&\vdots \\
s_i[c] &= \ldots \\
s_e[c, c'] &= \ldots \\
\mathcal{L} &= -\ln\ P(y \mid s_{\mathcal{I}}[\mathcal{C}],\ s_{\mathcal{E}}[\mathcal{C}, \mathcal{C}])
\end{aligned}$$

We need to compute $s_i.\mathrm{grad}[c]$ and $s_e.\mathrm{grad}[c, c']$.

General Markov Random Fields (MRFs)



$$s(\hat{y}) = \sum_{i \in \text{Nodes}} s_i[\hat{y}[i]] + \sum_{e \in \text{Edges}} s_e[\hat{y}[e.i], \hat{y}[e.j]]$$

Node Potentials

Edge Potentials

An Example

Consider an image with three superpixels A, B and C where each superpixel is to labeled as either "foreground" or background.

Suppose the unary potentials are all zero.

$$
\begin{aligned}
s_A(\text{Foreground}) = s_A(\text{Background}) = 0 \\
s_B(\text{Foreground}) = s_B(\text{Background}) = 0 \\
s_C(\text{Foreground}) = s_C(\text{Background}) = 0
\end{aligned}
$$

The Binary Potentials

Let F_A be the proposition that A is forground and similarly for F_B and F_C.

We can express $P_A \Rightarrow P_B$ with

$$s_{A,B}(\text{Foreground}, \text{Background}) = -1$$
$$s_{A,B}(\text{Foreground}, \text{Foreground}) = 1$$
$$s_{A,B}(\text{Background}, \text{Background}) = 1$$
$$s_{A,B}(\text{Background}, \text{Foreground}) = 1$$

The binary potentials are then given by $F_A \Rightarrow F_B$, $F_B \Rightarrow F_C$, $F_C \Rightarrow F_A$.

The Full Configuration Potential

For any configuration $\hat{y}$ we have that $s(\hat{y})$ is the sum of the unary and binary potentials.

If none are foreground we have $s(\hat{y}) = 3$

If one is foreground we have $s(\hat{y}) = -1 + 1 + 1 = 1$

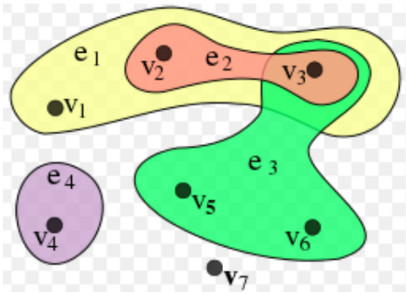
If two are foreground we also have $s(\hat{y}) = -1 + 1 + 1 = 1$

If all are foreground we have $s(\hat{y}) = 3$.

$$Z = 6 * 1 + 2 * 3 = 12 \quad P_A(\text{Foregound}) = \frac{3 * 1 + 3}{12} = \frac{1}{2}$$

Hyper-Graphs: More General and More Concise

A hyper-edge is a subset of nodes.



$$s(\hat{y}) = \sum_{i \in \text{Nodes}} s_i[\hat{y}[i]] + \sum_{e \in \text{Edges}} s_e[\hat{y}[e.i], \hat{y}[e.j]]$$

$$s(\hat{y}) = \sum_{e \in \text{HyperEdges}} s_e[\hat{y}[e]]$$

Backpropagation

The input is the image x and the parameter package Φ

$$\begin{aligned} &\vdots \\ s_e[\hat{y}] &= \ldots \\ \mathcal{L} &= -\ln\ P(y \mid s_{\mathcal{E}}[\mathcal{Y}]) \end{aligned}$$

We abbreviate $P(\hat{y} \mid s_{\mathcal{E}}[\mathcal{Y}])$ as $P_s(\hat{y})$ — the distribution on $\hat{y}$ defined by the tensor s.

We need to compute $\nabla_s - \ln P_s(y)$, or equivalently, $s_e.\mathrm{grad}[\tilde{y}]$.

Back-Propagation Through An Exponential Softmax

$$\begin{aligned} \text{loss}(s, y) &= -\ln\left(\frac{1}{Z(s)}\, e^{s(y)}\right) \\ &= \ln Z(s) - s(y) \end{aligned}$$

$$s_e.\text{grad}[\tilde{y}] = \left(\frac{1}{Z}\sum_{\hat{y}} e^{s(\hat{y})}\left(\partial s(\hat{y})/\partial s_e[\tilde{y}]\right)\right) - \left(\partial s(y)/\partial s_e[\tilde{y}]\right)$$

Back-Propagation Through An Exponential Softmax

$$
\begin{aligned}
s_e.\mathrm{grad}[\tilde{y}] &= \left(\frac{1}{Z}\sum_{\hat{y}} e^{s(\hat{y})}\,(\partial s(\hat{y})/\partial s_e[\tilde{y}])\right) - (\partial s(y)/\partial s_e[\tilde{y}]) \\
&= \left(\sum_{\hat{y}} P_s(\hat{y})\,(\partial s(\hat{y})/\partial s_e[\tilde{y}])\right) - (\partial s(y)/\partial s_e[\tilde{y}]) \\
&= E_{\hat{y}\sim P_s}\mathbb{1}[\hat{y}[e]=\tilde{y}] - \mathbb{1}[y[e]=\tilde{y}] \\
&= P_{\hat{y}\sim P_s}(\hat{y}[e]=\tilde{y}) - \mathbb{1}[y[e]=\tilde{y}]
\end{aligned}
$$

Hyperedge Marginals

$$s.\text{grad}[e, \tilde{y}] = P_{\hat{y} \sim P_s}(\hat{y}[e] = \tilde{y}) - \mathbb{1}[y[e] = \tilde{y}]$$

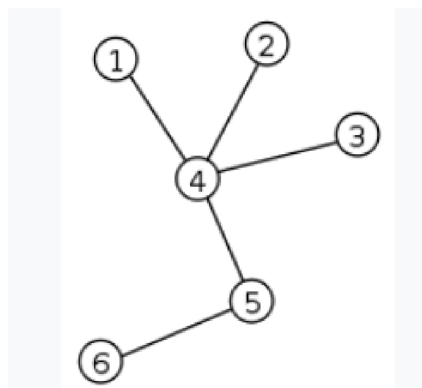
We will write $P_e(\tilde{y})$ for $P_{\hat{y} \sim P_s}(\hat{y}(e) = \tilde{y})$.

To compute s.grad it suffices to compute $P_e(\tilde{y})$.

We now focus on computing the hyperedge marginals for a given hyperedge score function (MRF) s.

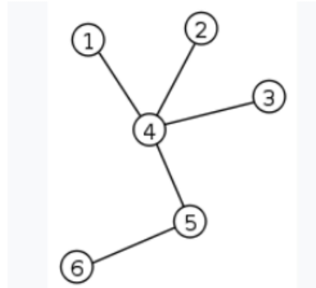
For Tree-Strutured Models
The Hyperedge Marginals Can be Computed Exactly

$$s.\mathrm{grad}[e, \tilde{y}] = {\color{red} P_e(\tilde{y})} - \mathbb{1}[y[e] = \tilde{y}]$$



For trees we can compute $P_e(\tilde{y})$ exactly by message passing, a.k.a., belief propagation.

Defining the Messages

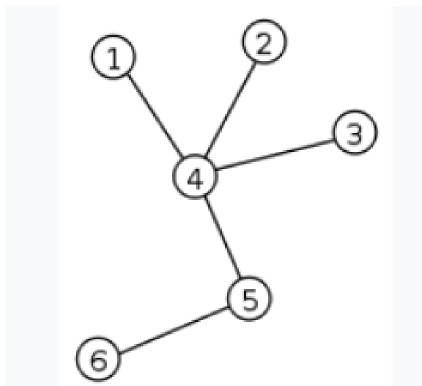


For each edge $\{i, j\}$ and possible value $\tilde{y}$ for node i we define $Z_{j \rightarrow i}[\tilde{y}]$ to be the partition function for the subtree attached to i through j and with $\hat{y}[i]$ restricted to $\tilde{y}$.

The function $Z_{j \rightarrow i}$ on the possible values of node i is called the **message** from j to i.

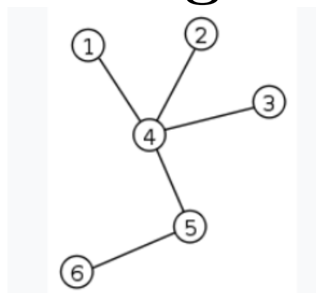
The reverse direction message $Z_{i \rightarrow j}$ is defined similarly.

Computing the Messages



$$Z_{j\to i}[\tilde{y}] = \sum_{\tilde{y}'} e^{s_j[\tilde{y}'] + s_{\{j,i\}}[\{\tilde{y}',\tilde{y}\}]} \left(\prod_{k\in N(j),\, k\neq i} Z_{k\to j}[\tilde{y}'] \right)$$

Computing Node Marginals from Messages



$$Z_i(\tilde{y}) \doteq \sum_{\hat{y}:\ \hat{y}[i]=\tilde{y}} e^{s(\hat{y})}$$

$$= e^{s_i[\tilde{y}]} \left(\prod_{j \in N(i)} Z_{j \to i}[\tilde{y}] \right)$$

$$\color{red}{P_i(\tilde{y})} = Z_i(\tilde{y})/Z, \qquad Z = \sum_{\tilde{y}} Z_i(\tilde{y})$$

Computing Edge Marginals from Messages

$$
\begin{aligned}
Z_{\{i,j\}}(\tilde{y}) &\doteq \sum_{\hat{y}:\ \hat{y}[\{i,j\}]=\tilde{y}} e^{s(\hat{y})} \\
&= e^{s[i,\tilde{y}[i]]+s[j,\tilde{y}[j]]+s[\{i,j\},\tilde{y}]} \\
&\quad \prod_{k\in N(i),\ k\neq j} Z_{k\to i}[\tilde{y}[i]] \\
&\quad \prod_{k\in N(j),\ k\neq i} Z_{k\to j}[\tilde{y}[j]] \\
P_{\{i,j\}}(\tilde{y}) &= Z_{\{i,j\}}(\tilde{y})/Z
\end{aligned}
$$

Loopy BP

Message passing is also called belief propagation (BP).

In a graph with cycles it is common to do **Loopy BP**.

This is done by initializing all message $Z_{i \to j}[\tilde{y}] = 1$ and then repeating (until convergence) the updates

$$P_{j \to i}[\tilde{y}] = \frac{1}{Z} Z_{j \to i}[\tilde{y}] \qquad Z = \sum_{\tilde{y}} Z_{j \to i}[\tilde{y}]$$

$$Z_{j \to i}[\tilde{y}] = \sum_{\tilde{y}'} e^{s[j,\tilde{y}'] + s[\{j,i\},\{\tilde{y}',\tilde{y}\}]} \left(\prod_{k \in N(j),\; k \neq i} P_{k \to j}[\tilde{y}'] \right)$$

Other Methods of Approximating Hyperedge Marginals

MCMC Sampling

Constrastive Divergence

Pseudo-Liklihood

The Big Picture I

Conditional vs. Unconditional

$$\Phi^* = \operatorname*{argmin}_{\Phi} E_{(x,y)\sim\mathrm{Pop}} \; -\ln\; P(y|x)$$

$$\Phi^* = \operatorname*{argmin}_{\Phi} E_{y\sim\mathrm{Pop}} \; -\ln\; P(y)$$

This is a non-distinction: the issues in the to the conditional case are exactly the same as in the unconditional case.

The Big Picture II

The number of possible values of y.

The binary case: $y \in \{-1, 1\}$.

The multiclass case: $y \in \mathcal{Y}$ where iteration over $\hat{y} \in \mathcal{Y}$ is feasible.

The structured case: $y \in \mathcal{Y}$ where $\mathcal{Y}$ is discrete but iteration over $\hat{y} \in \mathcal{Y}$ is infeasible.

The Big Picture I

Conditional vs. Unconditional

$$\Phi^* = \operatorname*{argmin}_{\Phi} E_{(x,y)\sim\mathrm{Pop}} \; -\ln \; P(y|x)$$

$$\Phi^* = \operatorname*{argmin}_{\Phi} E_{y\sim\mathrm{Pop}} \; -\ln \; P(y)$$

This is a non-distinction: the issues in the to the conditional case are exactly the same as in the unconditional case.

The Big Picture II

The structured case: $y \in \mathcal{Y}$ where $\mathcal{Y}$ is discrete but iteration over $\hat{y} \in \mathcal{Y}$ is infeasible.

Graphical models rely on assumptions about the structure of $P_\Phi(y)$.

Loopy Graphical Models can use weaker assumptions than those supporting exact computation of $P_\Phi(y)$.

For loopy graphical Models there are various methods of performing approximate gradient descent on cross-entropy loss.

END