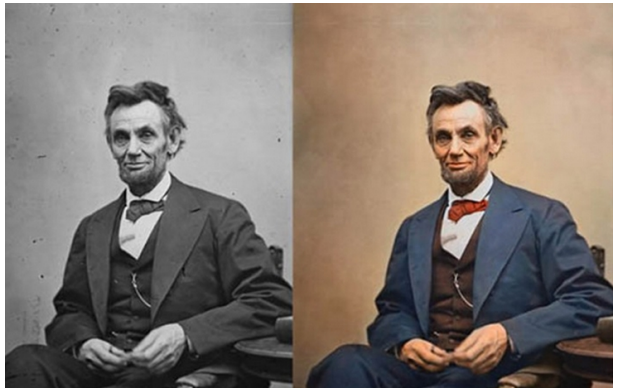
TTIC 31230, Fundamentals of Deep Learning

David McAllester, Winter 2018

Deep Graphical Models, Review and Continuation

Connectionist Temporal Classification CTC

Labeling Pixels



x is a black and white image.

y is a color image drawn from Pop$|x$.

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

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

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

Graphical Model Classification

Directed (Local Softmax)

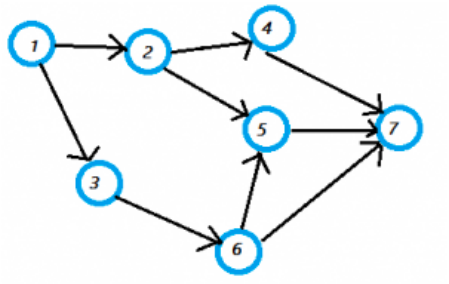
vs.

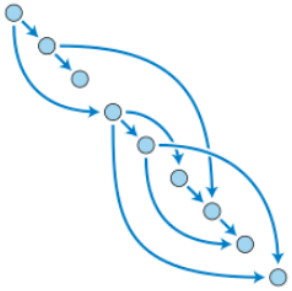
Markov Random Field (MRF) (Global Softmax)

Tree Structured vs. Loopy

Latent Variables vs. Fully Observed

Fully Observed Directed (Local Softmax) Models



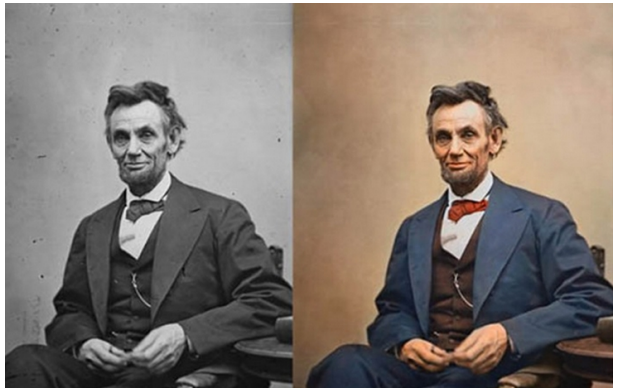


A language model has this structure.

Each node is assigned a probability distribution based on the value of its parents.

In this case we can backpropagate through the softmax operations in the normal way.

Labeling Pixels with a Fully Observed MRF



x is a black and white image.

y is a color image drawn from Pop$|x$.

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

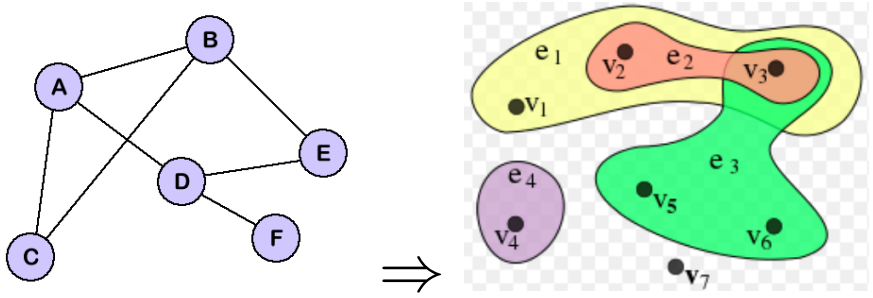
$\hat{y}[i]$ is the color value of pixel i in image $\hat{y}$.

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

The Tensor $f_\Phi(x)$

We can run a CNN, perhaps a version of Resnet152 that maintains image dimensions, to compute a top layer tensor $f_\Phi(x)$.

The Tensor $f_\Phi(x)$



$$f_\Phi(x)(\hat{y}) = \sum_{i \in \text{Nodes}} f_\Phi(x)[i, \hat{y}[i]] + \sum_{(i,j) \in \text{Edges}} f_\Phi(x)[(i,j), \hat{y}[(i,j)]]$$

Node Potentials Edge Potentials

$$f_\Phi(x)(\hat{y}) = \sum_{\alpha \in \substack{\text{HyperEdges} \\ (\text{Nodes} \cup \text{Edges})}} f_\Phi(x)[\alpha, \hat{y}[\alpha]]$$

The Shape of $f_\Phi(x)$

$$f_\Phi(x)(\hat{y}) = \sum_{\alpha \in \substack{\text{HyperEdges} \\ (\text{Nodes} \cup \text{Edges})}} f_\Phi(x)[\alpha, \hat{y}[\alpha]]$$

The tensor $f_\Phi(x)$ has shape $[\alpha, \tilde{y}]$

$\tilde{y}$ ranges over assignments of values to α.

Global Softmax

$$Q_{f_\Phi(x)}(\hat{y}) = \underset{\hat{y}}{\text{softmax}}\ f_\Phi(x)(\hat{y})$$

$$Q_f(\hat{y}) = \frac{1}{Z}\, e^{f(\hat{y})} \qquad Z = \sum_{\hat{y}} e^{f(\hat{y})}$$

$$f_\Phi(x)(\hat{y}) = \sum_{\alpha} f_\Phi(x)[\alpha, \hat{y}[\alpha]]$$

Fundamental Equation of Deep Learning

$$\Phi^* = \underset{\Phi}{\operatorname{argmin}} \quad E_{(x,y)\sim \mathrm{Pop}} - \log Q_\Phi(y)$$

Fundamental Equations of MRFs (Global Softmax)

$$\Phi^* = \underset{\Phi}{\operatorname{argmin}} \quad E_{(x,y)\sim \mathrm{Pop}} - \log Q_{f_\Phi(x)}(y)$$

$$Q_f(\hat{y}) = \underset{\hat{y}}{\operatorname{softmax}} \; f(\hat{y})$$

$$f(\hat{y}) = \sum_{\alpha} f[\alpha, \hat{y}[\alpha]]$$

Back-Propagation Through a Global Softmax

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

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

$$= E_{\hat{y}\sim Q_f}\mathbb{1}[\hat{y}[\alpha] = \tilde{y}] - \mathbb{1}[y[\alpha] = \tilde{y}]$$

Training Methods for MRFs

- MCMC Sampling — Metropolis-Hastings or Gibbs Sampling.
- Message Passing, a.k.a. Belief Propagation or Loopy BP.
- Pseudolikelihood
- Contrastive Divergence

MCMC

We estimate $P_{\hat{y} \sim Q_f}(\hat{y}[\alpha] = \tilde{y})$ by sampling.

We define a Markov chain whose stationary distribution is Q_f.
— Metropolis-Hastings, Gibbs or Hamiltonian MCMC.

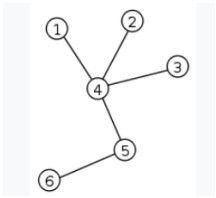
We sample $\hat{y}$ by running the chain and use

$$f.\text{grad}[\alpha, \tilde{y}] = \mathbb{1}[\hat{y}[\alpha] = \tilde{y}] - \mathbb{1}[y[\alpha] = \tilde{y}]$$

The expectation of f.grad under the sampling is correct if we run the chain long enough.

Message Passing (Loopy BP)

We compute $P_{\hat{y}\sim Q_f}(\hat{y}[\alpha] = \tilde{y})$ by message passing.



$$P_{\hat{y}\sim Q_f}(\hat{y}[i] = \tilde{y}) = \frac{1}{Z} e^{f[i,\tilde{y}]} \prod_{j \in N(i)} Z_{j \to i}(\tilde{y})$$

$$P_{\hat{y}\sim Q_f}(\hat{y}[(i,j)] = (\tilde{y}_1, \tilde{y}_2)) = \frac{1}{Z} e^{f[i,\tilde{y}_1]} e^{f[(i,j),(\tilde{y}_1,\tilde{y}_2)]} e^{f[j,\tilde{y}_2]} (\prod \dots)$$

Pseudolikelihood

In Pseudolikelihood we replace the objective $-\log Q_f(\hat{y})$ with the objective $-\log \tilde{Q}_f(\hat{y})$ where

$$\tilde{Q}_f(\hat{y}) \doteq \prod_i Q_f(\hat{y}[i] \mid \hat{y}[\text{Neighbors}(i)])$$

$$\text{loss}(f) \doteq -\log \tilde{Q}(y)$$

$$f.\text{grad}[\alpha, \tilde{y}] = \sum_i -\partial \log Q_f[\hat{y}[i] \mid \hat{y}[\text{Neighbors}(i)]) \;/\; \partial f[\alpha, \tilde{y}]$$

Pseudolikelihood Theorem

If $\text{Pop} = Q_f$ for some tensor f then for

$$f^* = \underset{f}{\text{argmin}} \; E_{y \sim \text{Pop}} \; -\log \tilde{Q}_f(y)$$

we have $Q_{f^*} = \text{Pop}$

Contrastive Divergence

Algorithm (CDk): Run k steps of MCMC for $Q_f(\hat{y})$ **starting from** y to get $\hat{y}$.

Then set

$$f.\text{grad}[\alpha, \tilde{y}] = \mathbb{1}[\hat{y}[\alpha] = \tilde{y}] - \mathbb{1}[y[\alpha] = \tilde{y}]$$

This is the same update as MCMC.

Theorem: If $Q_f(\hat{y}) = \text{Pop}$ then the expected update is zero.

Here we can take $k = 1$ — no mixing time required.

Label Adjustment

Label adjustment operates on **error** rather than log loss.

We can train translation directly on BLEU score.

We can train segmentation directly on intersection over union (IOU) score.

We can train cancer screening for different costs of false positives and false negative.

Minimizing Error

$$\Phi^* = \operatorname*{argmin}_{\Phi} E_{(x,y)\sim \mathrm{Pop}}\, \mathrm{Err}(y, \hat{y}^*(f_\Phi(x)))$$

$$\hat{y}^*(f) = \operatorname*{argmax}_{\hat{y}}\ f(\hat{y})$$

$$f(\hat{y}) = \sum_{\alpha} f[\alpha, \hat{y}[\alpha]]$$

Note that $y^*(f)$ is not differentiable in f.

Label Adjustment

$$f(\hat{y}) = \sum_{\alpha} f[\alpha, \hat{y}[\alpha]]$$

$$\hat{y}^*(f) = \underset{\hat{y}}{\text{argmax}} \; f(\hat{y})$$

$$\breve{y}^*(f, \epsilon) = \underset{\breve{y}}{\text{argmax}} \; f(\breve{y}) - \epsilon \text{Err}(y, \breve{y}) \quad \textbf{(adjusted label)}$$

$$f.\text{grad}[\alpha, \tilde{y}] = \mathbb{1}[\hat{y}^*(f_\Phi(x))[\alpha] = \tilde{y}] - \mathbb{1}[\breve{y}^*(f_\Phi(x), \epsilon)[\alpha] = \tilde{y}]$$

Label Adjustment Theorem

$$\breve{y}^*(f,\epsilon) = \operatorname*{argmax}_{\breve{y}} f(\breve{y}) - \epsilon \mathrm{Err}(y,\breve{y})$$

$$f.\mathrm{grad}[\alpha,\tilde{y}] = \mathbb{1}[\hat{y}^*(f_\Phi(x))[\alpha] = \tilde{y}] - \mathbb{1}[\breve{y}^*(f_\Phi(x),\epsilon)[\alpha] = \tilde{y}]$$

Theorem: If $\mathrm{Pop}(x)$ is smooth then

$$\nabla_\Phi \, E_{(x,y)\sim \mathrm{Pop}} \, \mathrm{Err}(y, \hat{y}^*(f_\Phi(x)))$$

$$= \lim_{\epsilon \to 0} \frac{1}{\epsilon} E_{(x,y)\sim \mathrm{Pop}}$$

$$\sum_{\alpha,\tilde{y}} \left(\mathbb{1}[\hat{y}^*(f_\Phi(x))[\alpha] = \tilde{y}] - \mathbb{1}[\breve{y}^*(f_\Phi(x),\epsilon)[\alpha] = \tilde{y}] \right) \nabla_\Phi f_\Phi(x)[\alpha,\tilde{y}]$$

Fundamental Equations of Fully Observed MRFs

$$\Phi^* = \underset{\Phi}{\operatorname{argmin}} \quad E_{(x,y)\sim \mathrm{Pop}} - \log Q_{f_\Phi(x)}(y)$$

$$Q_f(\hat{y}) = \underset{\hat{y}}{\operatorname{softmax}} \quad f(\hat{y})$$

$$f(\hat{y}) = \sum_{\alpha} \; f[\alpha, \hat{y}[\alpha]]$$

Fundamental Equations of Latent Variable MRFs

$$\Phi^* = \underset{\Phi}{\operatorname{argmin}} \;\; E_{(x,y)\sim \mathrm{Pop}} - \log Q_{f_\Phi(x)}(y)$$

$$Q_f(\hat{z}, \hat{y}) = \underset{\hat{z},\hat{y}}{\operatorname{softmax}} \;\; f(\hat{z}, \hat{y})$$

$$f(\hat{z}, \hat{y}) = \sum_{\alpha} \; f[\alpha, \hat{z}[\alpha], \hat{y}[\alpha]]$$

$$Q_f(\hat{y}) = \sum_{\hat{z}} Q_f(\hat{z}, \hat{y})$$

Latent Variable MRF

$$
\begin{aligned}
Q_f(\hat{z}, \hat{y}) &= \frac{1}{Z}\, e^{f(\hat{z},\hat{y})} \\
Q_f(\hat{y}) &= \sum_{\hat{z}} Q_f(\hat{z}, \hat{y}) \\
&= \frac{\sum_{\hat{z}} e^{f(\hat{z},\hat{y})}}{Z} \\
Q_f(\hat{y}) &= \frac{Z(\hat{y})}{Z} \\
\text{loss}(y) &= \ln Z - \ln Z(y)
\end{aligned}
$$

Latent Variable MRF

$$\text{loss}(y, f) = \ln Z - \ln Z(y)$$

$$f.\text{grad}[\alpha, \tilde{y}, \tilde{z}] = P_{\hat{z},\hat{y} \sim Q_f}(\hat{y}[\alpha] = \tilde{y}, \hat{z}[\alpha] = \tilde{z})$$

$$- P_{\hat{z} \sim Q_f(\hat{z}|y)}(y[\alpha] = \tilde{y}, \hat{z}[\alpha] = \tilde{z})$$

Latent Variable MRF

- MCMC Sampling — Metropolis-Hastings or Gibbs Sampling.
- Loopy BP (Message Passing) (exact for trees).
- Pseudolikelihood
- Contrastive Divergence

Latent Variable Directed Models

In a directed model each softmax is locally normalized and easily computed.

Backpropagation through local softmax operations can be done in the standard way.

Fundamental Equations of Latent Variable Directed Models

$$\Phi^* = \underset{\Phi}{\operatorname{argmin}} \quad E_{(x,y)\sim \mathrm{Pop}} - \log Q_\Phi(y)$$

$$Q(\hat{z}, \hat{y}) = \prod_i Q(\hat{v}[i] \mid \hat{v}[\mathrm{Parents}(i)]) \quad \hat{v} = (\hat{z}, \hat{y})$$

We can represent Q as a tensor satisfying

$$Q[i, \hat{v}[i, \mathrm{Parents}(i)]] = Q(\hat{v}[i] \mid \hat{v}[\mathrm{Parents}(i)])$$

Q has shape $[i, \tilde{v}]$

Backpropagation to the tensor Q

Since backpropagation through local softmax operations is not a problem, it suffices to backpropagate to Q.

We want to compute $Q.\text{grad}[i, \tilde{v}]$ where $\tilde{v}$ may may include unobserved values.

There are exponentially many values of $\hat{z}$.

For a loopy (non-tree) computing $Q.\text{grad}[i, \tilde{v}]$ is #P-hard.

However, $Q.\text{grad}[i, \tilde{v}]$ can be computed efficiently for tree models or estimated in various ways for loopy models.

The Expected Gradient Identity

$$\text{loss} = -\ln\ Q(y)$$

$$\nabla_Q \text{ loss} = \frac{\nabla_Q\ Q(y)}{Q(y)}$$

$$= \sum_{\hat{z}} \frac{Q(\hat{z}, y)}{Q(y)} \nabla_Q\ \ln Q(\hat{z}, y)$$

$$= E_{\hat{z} \sim Q(\hat{z}|y)} \nabla_Q \ln Q(\hat{z}, y)$$

The Sufficient Statistics

$$\begin{aligned}
\nabla_Q \text{ loss} &= E_{\hat{z}\sim Q(\hat{z}|y)} \nabla_Q \ln Q(\hat{z}, y) \\
&= E_{\hat{z}\sim Q(\hat{z}|y)} \sum_i \nabla_Q \ln Q((\hat{z}, y)[i] \mid (\hat{z}, y)[\text{parents}(i)]) \\
Q.\text{grad}[i, \tilde{v}] &= E_{\hat{z}\sim Q(\hat{z}|y)} \frac{1}{Q[i, \tilde{v}]} \mathbb{1}[(\hat{z}, y)[i, \text{Parents}(i)] = \tilde{v}] \\
&= \frac{1}{Q[i, \tilde{v}]} E_{\hat{z}\sim Q(\hat{z}|y)} \mathbb{1}[(\hat{z}, y)[i, \text{Parents}(i)] = \tilde{v}]
\end{aligned}$$

The quantities $E_{\hat{z}\sim Q(\hat{z}|y)} \mathbb{1}[(\hat{z}, y)[i, \text{Parents}(i)] = \tilde{v}]$ are the **sufficient statistics**.

Expected Gradient (EG) and Expectation Maximization (EM)

EG: $\nabla_Q \ln Q(y) = \mathrm{E}_{\hat{z} \sim Q(\hat{z}|y)} \nabla_Q \ln Q(\hat{z}, y).$

EM: $Q^{t+1} = \mathrm{argmax}_Q \ \mathrm{E}_{\hat{z} \sim Q^t(\hat{z}|y)} \ \ln Q(\hat{z}, y).$

EG $= \nabla_Q \ln Q(y)$ equals is the gradient of the EM objective.

EG and EM have **the same sufficient statistics** (E-step).

Connectionist Temporal Classification (CTC)

Graves, Fernandez, Gomez, Schmidhuber, ICML 2006

A Latent Variable

Directed

Tree Structured

Graphical Model

Connectionist Temporal Classification (CTC)

Phonetic Transcription

A speech signal

$$x = x_1, \ \ldots, \ x_T$$

is labeled with a phone sequence

$$y = y_1, \ldots, y_N$$

with $N << T$ and with $y_n \in \mathcal{Y}$ for a set of phonemes $\mathcal{Y}$.

The length N of y is not determined by x and the alignment between x and y is not given.

CTC

The model defines $Q_\Phi(\hat{z}|x, y)$ where $\hat{z}$ is latent.

$$\hat{z} = \hat{z}_1, \ \ldots, \ \hat{z}_T, \quad \hat{z}_t \in \mathcal{Y} \cup \{\perp\}$$

The sequence

$$y(\hat{z}) = y_1, \ \ldots, \ y_N$$

is the result of removing all the occurrences of $\perp$ from $\hat{z}$.

$$\perp, a_1, \perp, \perp, \perp, a_2, \perp, \perp, a_3, \perp \Rightarrow a_1, a_2, a_3$$

The CTC Model

$$h_1, \ldots, h_T = \mathrm{RNN}_\Phi(x_1, \ldots, x_T)$$

$$Q_\Phi(\hat{z}_t | x_1, \ldots, x_T) = \operatorname*{softmax}_{\hat{z}} e(\hat{z})^\top h_t$$

This is a locally normalized (directed) graphical model where $\hat{z}_t$ does not have any parent nodes.

The Sufficient Statistics

Since each node has no parents the sufficient statistics are

$$P_{\hat{z} \sim Q(\hat{z}|y)}(z_t = \tilde{z})$$

Dynamic Programming (Forward-Backward)

$$
\begin{aligned}
x &= x_1, \ \ldots, \ x_T \\
\hat{z} &= \hat{z}_1, \ \ldots, \ \hat{z}_T, \quad \hat{z}_t \in \mathcal{Y} \cup \{\perp\} \\
y &= y_1, \ \ldots, \ y_N, \quad y_n \in \mathcal{Y}, \quad N << T \\
y &= (\hat{z}_1, \ \ldots, \ \hat{z}_T) - \perp
\end{aligned}
$$

Forward-Backward

$$
\begin{aligned}
\vec{y}_t &= (\hat{z}_1, \ \ldots, \ \hat{z}_t) - \perp \\
F[n,t] &= Q(\vec{y}_t = y_1, \ \ldots, \ y_n) \\
B[n,t] &= Q(y_{n+1}, \ \ldots, y_N | \vec{y}_t = y_1, \ \ldots, \ y_n)
\end{aligned}
$$

Dynamic Programming (Forward-Backward)

$$
\begin{aligned}
\vec{y}_t &= (\hat{z}_1, \ \ldots, \ \hat{z}_t) - \perp \\
F[n,t] &= Q(\vec{y}_t = y_1, \ \ldots, \ y_n) \\
B[n,t] &= Q(y_{n+1}, \ \ldots, y_N | \vec{y}_t = y_1, \ \ldots, \ y_n)
\end{aligned}
$$

$$
\begin{aligned}
F[0,0] &= 1 \\
F[n,0] &= 0 \quad \text{for } n > 0 \\
F[n+1,t+1] &= Q(\hat{z}_{t+1} = \perp) F[n+1,t] + Q(\hat{z}_{t+1} = y_{n+1}) F[n,t]
\end{aligned}
$$

$$
\begin{aligned}
B[N,T] &= 1 \\
B[n,T] &= 0, \quad \text{for } n < N \\
B[n-1,t-1] &= Q(\hat{z}_{t-1} = \perp) B[n-1,t] + Q(\hat{z}_{t-1} = y_{n-1}) B[n,t]
\end{aligned}
$$

The Sufficient Statistics

$$P_{\hat{z}\sim Q(\hat{z}|y)}(\hat{z}_t = \tilde{z})$$

$$= \frac{1}{Q(y)} \sum_n \begin{cases} F[n, t-1]\; Q(\hat{z}_t)\; B[n, t] & \text{for } \hat{z}_t = \perp \\ F[n, t-1]\; Q(\hat{z}_t)\; B[n+1, t] & \text{for } \hat{z}_t = y_{n+1} \\ 0 & \text{otherwise} \end{cases}$$

END