CS 533: Natural Language Processing

Marginal Decoding, Conditional Random Fields

Karl Stratos

Rutgers University

Review: Tagging by Generative Probabilistic Tagger

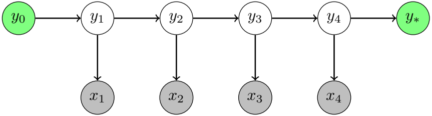
- Tagging: Map sentence $x_{1:T} = (x_1 \ldots x_T) \in \mathcal{V}^T$ to label sequence $y_{1:T} = (y_1 \ldots y_T) \in \mathcal{Y}^T$
- Generative model: *joint* distribution, chain rule

$$p_\theta(x_{1:T},\ y_{1:T}) = \prod_{t=1}^{T} p_\theta(y_t|x_{<t}, y_{<t}) \times p_\theta(x_t|x_{<t}, y_{\leq t}) \times p_\theta(y_*|x_{\leq T}, y_{\leq T})$$

- (First-order) Hidden Markov models (HMMs)

$$p_\theta(x_{1:T},\ y_{1:T}) = \prod_{t=1}^{T} \underbrace{t_\theta(y_t|y_{t-1})}_{\text{transition prob}} \times \underbrace{o_\theta(x_t|y_t)}_{\text{emission prob}} \times t_\theta(y_*|y_T)$$

- Simplest form of labeled sequence generation, marginalization and inference tractable



Review: Exact Marginalization by Forward Algorithm

- **Marginalization.** What is the *marginal* probability of $x_{1:T}$ under the model?

$$p_\theta(x_{1:T}) = \sum_{y_{1:T} \in \mathcal{Y}^T} p_\theta(x_{1:T},\ y_{1:T})$$

- **Forward algorithm.** Fills out table $\pi \in \mathbb{R}^{T \times |\mathcal{Y}|}$ defined as

$$\pi(t, y) = \sum_{y_1 \ldots y_t \in \mathcal{Y}^t:\ y_t = y} p_\theta(x_1 \ldots x_t, y_1 \ldots y_t)$$

by computing for all $y, y' \in \mathcal{Y}$ and $t > 1$ left-to-right

$$\pi(1, y) = t_\theta(y|y_0) \times o_\theta(x_1|y)$$
$$\pi(t, y') = \sum_{y \in \mathcal{Y}} \pi(t-1, y) \times t_\theta(y'|y) \times o_\theta(x_t|y')$$

- Return $p_\theta(x_{1:T}) = \sum_{y \in \mathcal{Y}} \pi(T, y) \times t_\theta(y_*|y)$

Review: Exact Inference by Viterbi Algorithm

- **Inference.** What is the most probable $y_{1:T} \in \mathcal{Y}^T$ of $x_{1:T}$ under the model?

$$y^*_{1:T} = \operatorname*{arg\,max}_{y_{1:T} \in \mathcal{Y}^T} \; p_\theta(x_{1:T},\; y_{1:T})$$

- **Viterbi algorithm.** Fills out table $\pi \in \mathbb{R}^{T \times |\mathcal{Y}|}$ defined as

$$\pi(t, y) = \max_{y_1 \ldots y_t \in \mathcal{Y}^t :\; y_t = y} \; p_\theta(x_1 \ldots x_t, y_1 \ldots y_t)$$

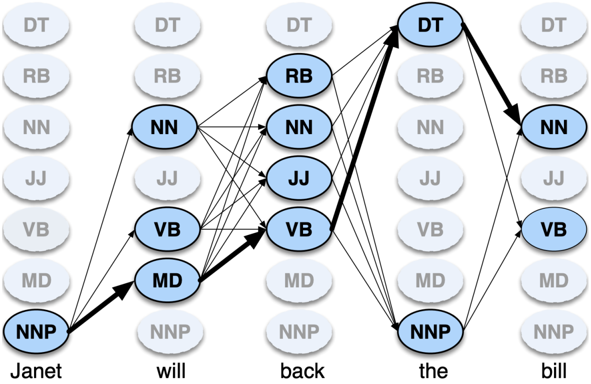
  Same as forward, only switch sum to max. Then $p_\theta(y^*_{1:T}|x_{1:T}) = \max_{y \in \mathcal{Y}} \pi(T, y) \times t_\theta(y_*|y)$

- But this only gives us max probability, must keep a **backtracking** table to record the label path during Viterbi

$$\beta(t, y') = \operatorname*{arg\,max}_{y \in \mathcal{Y}} \;\; \pi(t-1, y) \times t_\theta(y'|y) \times o_\theta(x_t|y')$$

Constrained Inference

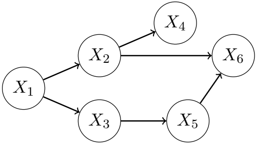
- Easy to modify Viterbi to only consider certain paths, e.g.,
  - **NER.** If $y_t = \texttt{B-PER}$, then we must have $y_{t+1} \in \{\texttt{I-PER}, \texttt{O}\}$.
  - **POS.** For efficiency, only allow $y_{t+1} \in \mathcal{Y}(y_t)$ where $\mathcal{Y}(y_t)$ is the set of tags following y_t in training data



(Image credit: Jurafsky and Martin)

Directed Graphical Models (DGMs)

- HMM is a special case of a **directed graphical model** (DGM), aka. **Bayesian network** (Bayes net)
- Directed acyclic graph (DAG) representing a joint distribution, (lack of) directed edges encode conditional independence assumptions
- An example DGM (example credit: David Blei)

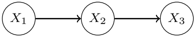


$$\Pr(X) = \Pr(X_1)\Pr(X_2|X_1)\Pr(X_3|X_1)$$
$$\Pr(X_4|X_2)\Pr(X_5|X_3)\Pr(X_6|X_2, X_5)$$

- Represents a joint distribution over $X = (X_1 \ldots X_6)$
  - Each $X_i \in \mathcal{X}_i$ has its own possible values
  - What independence assumptions are we making here?
- Again, two central calculations
  - Marginalization: e.g., $\Pr(X_2 = c) = \sum_{x:x_2=c} \Pr(X = x)$
  - Inference: $x^* = \arg\max_x \Pr(X = x)$

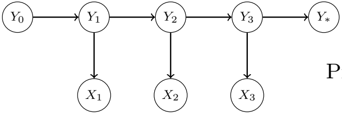
Examples of DGM

- n-gram language models with Markov order 1



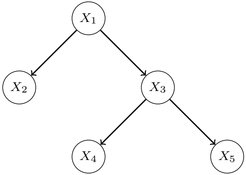
$$\Pr(X) = \Pr(X_1)\Pr(X_2|X_1)\Pr(X_3|X_2)$$

- HMMs



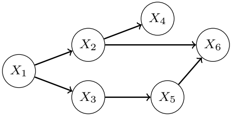
$$\Pr(X, Y) = \prod_{t=1}^{3} \Pr(Y_t|Y_{t-1})\Pr(X_t|Y_t)\Pr(Y_*|Y_3)$$

- Trees



$$\Pr(X) = \Pr(X_1)\Pr(X_2|X_1)\Pr(X_3|X_1)\Pr(X_4|X_3)\Pr(X_5|X_3)$$

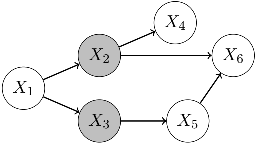
- General DAGs

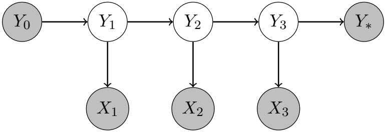


$$\begin{aligned}\Pr(X) = &\Pr(X_1)\Pr(X_2|X_1)\Pr(X_3|X_1)\\ &\Pr(X_4|X_2)\Pr(X_5|X_3)\Pr(X_6|X_2, X_5)\end{aligned}$$

Observed vs Unobserved Variables in DGM

- Typically some part of a DGM is observed





$X_2 = x_2, \; X_3 = x_3$

$X_1 = x_1, \; X_2 = x_2, \; X_3 = x_3$

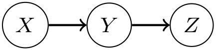
- We want to calculate various probabilities in the presence of observed variables, such as
  - Left: Probability of the observed event $\Pr(X_2 = x_2, X_3 = x_3)$
  - Right: Highest probability of label sequence $\max_{y_1,y_2,y_3} \Pr(X_1 = x_1, X_2 = x_2, X_3 = x_3, Y_1 = y_1, Y_2 = y_2, Y_3 = y_3)$. This is what Viterbi computes.
- Conditional independence assumptions in DGMs make efficient marginalization/inference feasible
  - Recall: X, Z independent ($X \perp\!\!\!\perp Z$) conditioned on Z iff

$$\Pr(X = x | Y = y, Z = z) = \Pr(X = x | Y = y)$$

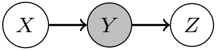
for all values of x, y, z (equiv. $p(x, y|z) = p(x|z)p(y|z)$)

Rules of Conditional Independence in DGMs

- The future is independent of the past given the present (Markov assumption)

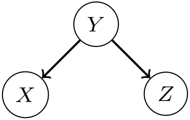


$X \not\perp\!\!\!\perp Z$

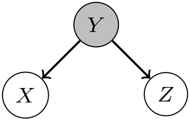


$X \perp\!\!\!\perp Z \mid Y$

- Children are independent of each other given their parent

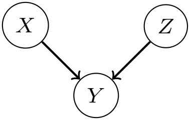


$X \not\perp\!\!\!\perp Z$

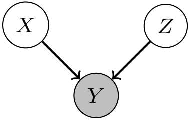


$X \perp\!\!\!\perp Z \mid Y$

- Causes are independent, but become dependent if effect is observed



$X \perp\!\!\!\perp Z$



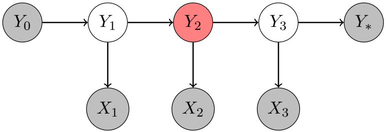
$X \not\perp\!\!\!\perp Z \mid Y$

- Exercise: Verify independence claims mathematically, and think of examples for non-independence claims

Marginal Decoding

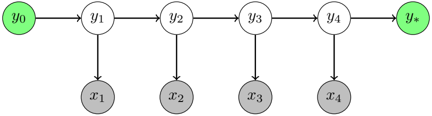
- Back to HMM: Given $x_{1:T}$ predict for *each* position $t = 1 \ldots T$

$$y_t^+ = \arg\max_{y \in \mathcal{Y}} \underbrace{\sum_{y_{1:T} \in \mathcal{Y}^T : y_t = y} p_\theta(x_{1:T},\ y_{1:T})}_{\text{"marginal" } \mu(t,y)}$$



- This is known as **marginal decoding**. This is in general *not* the same as Viterbi decoding
  - Better for per-position performance metric like POS tagging accuracy (can yield 1-2% improvement)
  - Worse for structure modeling like F1 in NER (why?)
- Central calculation: How to compute $\mu(t, y)$ for all $t = 1 \ldots T$ and $y \in \mathcal{Y}$?
- Answer: Application of forward and **backward** probabilities

Decomposition of Marginal Under HMMs



Future independent of past given y_t by Markov assumption

$$p_\theta(x_{1:T},\ y_{1:T}) \stackrel{*}{=} p_\theta(x_{\leq t},\ y_{\leq t}) \times p_\theta(x_{>t},\ y_{>t}|y_t)$$

Therefore marginal given by

$$\begin{aligned}\mu(t, y) &= \sum_{y_{1:T}:\ y_t = y} p_\theta(x_{\leq t},\ y_{\leq t}) \times p_\theta(x_{>t},\ y_{>t}|y_t) \\ &= \underbrace{\left(\sum_{y_{1:t}:\ y_t = y} p_\theta(x_{\leq t},\ y_{\leq t})\right)}_{\text{Forward prob!}} \underbrace{\left(\sum_{y_{>t}} p_\theta(x_{>t},\ y_{>t}|y_t = y)\right)}_{\text{How to compute this?}}\end{aligned}$$

Backward Algorithm

- DP similar to forward, but instead fills out *right-to-left*

$$\overleftarrow{\pi}(t, y) = \sum_{y_{t+1} \ldots y_T \in \mathcal{Y}^{T-t}} p_\theta(x_{t+1} \ldots x_T, y_{t+1} \ldots y_T \mid y_t = y)$$

- Base case: $\overleftarrow{\pi}(T, y) = t_\theta(y_* | y)$
- Main body: For $t = T - 1 \ldots 1$, for $y \in \mathcal{Y}$,

$$\begin{aligned}
\overleftarrow{\pi}(t, y) &= \sum_{y_{>t}} p_\theta(x_{>t}, y_{>t} \mid y_t = y) \\
&\overset{*}{=} \sum_{y_{>t+1}} \sum_{y'} p_\theta(x_{>t+1}, y_{>t+1} \mid y_{t+1} = y') \times t_\theta(y|y') \times o_\theta(x_t|y') \\
&= \sum_{y'} \underbrace{\overleftarrow{\pi}(t+1, y')}_{\text{already computed}} \times t_\theta(y|y') \times o_\theta(x_t|y')
\end{aligned}$$

- Runtime same as forward: $O(T\,|\mathcal{Y}|^2)$

Summary of Marginal Decoding

Assuming HMM parameters defining transition $t_\theta(y'|y)$ and emission $o_\theta(x|y)$ probabilities, given sentence $x_{1:T} \in \mathcal{V}^T$,

1. Run forward algorithm to compute for all t, y

$$\pi(t, y) = \sum_{y_1 \ldots y_t \in \mathcal{Y}^t:\ y_t = y} p_\theta(x_1 \ldots x_t, y_1 \ldots y_t)$$

2. Run backward algorithm to compute for all t, y

$$\overleftarrow{\pi}(t, y) = \sum_{y_{t+1} \ldots y_T \in \mathcal{Y}^{T-t}} p_\theta(x_{t+1} \ldots x_T, y_{t+1} \ldots y_T \mid y_t = y)$$

3. For all t, y calculate the marginal probability by

$$\mu(t, y) = \pi(t, y) \times \overleftarrow{\pi}(t, y)$$

4. For each position $t = 1 \ldots T$, predict as the label of x_t

$$y_t^+ = \arg\max_{y \in \mathcal{Y}} \ \mu(t, y)$$

Backpropagation as Backward Algorithm

- Recall: In computation graph DAG with output scalar variable x^ω, backpropagation computes $z^i := \nabla_{x^i} x^\omega$ by

$$z^i = \sum_{j \in \textbf{ch}(i)} z^j \times \nabla_{x^i} x^j \tag{1}$$

- Uses the fact that i affects ω only through its children nodes
- Equivalent/alternative view: (1) is "backward algorithm" for

$$z^i = \sum_{(i_1 \ldots i_n) \in P(i,\omega)} \nabla_{x^{i_{n-1}}} x^{i_n} \times \cdots \times \nabla_{x^{i_1}} x^{i_2} \tag{2}$$

  where $P(i, \omega)$ is an exponentially large set of all possible paths from i to ω, applies chain rule on each entire path.
- Why: Just rewrite (2) using DAG structure

$$\sum_{j \in \textbf{ch}(i)} \left(\sum_{(i_2 \ldots i_n) \in P(j,\omega)} \nabla_{x^{i_{n-1}}} x^{i_n} \times \cdots \times \nabla_{x^{i_2}} x^{i_3} \right) \times \nabla_{x^i} x^j$$

Discriminative Tagger

- Model defines a *conditional* distribution $p_\theta(y_1 \dots y_T | x_1 \dots x_T)$ over label sequences, given a sentence
  - *Cannot* generate $x_1 \dots x_T$, only predict label sequences
  - But if we only care about tagging, discriminative is sufficient
  - Discriminative possibly more effective than generative (esp with small labeled data), no need to learn input distribution
- Model: $\textbf{score}_\theta : \mathcal{V}^T \times \mathcal{Y}^T \to \mathbb{R}$ assigning score to any sentence paired with a tag sequence
  - Training: Minimize cross-entropy loss $H(\mathbf{pop}, p_\theta)$ where

$$p_\theta(y_{1:T}|x_{1:T}) = \frac{\exp(\textbf{score}_\theta(x_{1:T}, y_{1:T}))}{\sum_{y'_{1:T} \in \mathcal{Y}^T} \exp(\textbf{score}_\theta(x_{1:T}, y'_{1:T}))}$$

  - Inference: Given $x_{1:T}$ return $\arg\max_{y_{1:T} \in \mathcal{Y}^T} \textbf{score}_\theta(x_{1:T})$
- This is just a classifier, except that the label space is $\mathcal{Y}^T$
  - How to handle "giant softmax", find argmax label sequence?
  - Same approach: Make computation tractable by introducing **structural assumptions**, but now non-probabilistically

Markov Assumption in a Discriminative Tagger

- We *define* the score function to factorize as

$$\textbf{score}_\theta(x_{1:T}, y_{1:T}) = \sum_{t=1}^{T} \textbf{score}_\theta(x_{1:T}, y_{t-1}, y_t, t)$$

This model is called (first-order) **conditional random field** (CRF). Will discuss why later

- Only scores a label pair $y, y' \in \mathcal{Y}$ at each step t
  - But can still access the *entire* sentence (not just left/current input)! This is a major advantage of a discriminative model.
- Implications: Model distribution now

$$p_\theta(y_{1:T}|x_{1:T}) = \frac{1}{Z_\theta(x_{1:T})} \prod_{t=1}^{T} \underbrace{\exp(\textbf{score}_\theta(x_{1:T}, y_{t-1}, y_t, t))}_{t\text{-th nonnegative ``potential function''}}$$

$Z_\theta(x_{1:T}) := \sum_{y'_{1:T} \in \mathcal{Y}^T} \exp(\textbf{score}_\theta(x_{1:T}, y'_{1:T}))$ "partition function". Infer $\arg\max_{y_{1:T} \in \mathcal{Y}^T} \sum_{t=1}^{T} \textbf{score}_\theta(x_{1:T}, y_{t-1}, y_t, t)$

CRF Loss

- To optimize cross-entropy loss, given labeled sequence $x_{1:T}, y_{1:T}$ only need to compute

$$-\log p_\theta(y_{1:T}|x_{1:T}) = \underbrace{\log Z_\theta(x_{1:T})}_{\text{log partition function}} - \sum_{t=1}^{T} \textbf{score}_\theta(x_{1:T}, y_{t-1}, y_t, t)$$

- Central calculation: how to compute the log partition function? Again DP possible by Markov assumption
- **Forward algorithm**: Fill DP table for all t, y'

$$\pi(t, y') = \log\left(\sum_{y'_{1:t} \in \mathcal{Y}^t:\ y'_t = y'} \exp(\textbf{score}_\theta(x_{1:T}, y'_{1:t}))\right)$$

where $\textbf{score}_\theta(x_{1:T}, y'_{1:t}) = \sum_{l=1}^{t} \textbf{score}_\theta(x_{1:T}, y'_{l-1}, y'_l, l)$.
Then $\log Z_\theta(x_{1:T}) = \log(\sum_{y' \in \mathcal{Y}} \pi(T, y'))$.

Forward Algorithm for Computing Log Partition

- Base case: $\pi(1, y) = \textbf{score}_\theta(x_{1:T}, y_0, y, 1)$ for all $y \in \mathcal{Y}$
- Main body: For $t = 2 \ldots T$, for all $y' \in \mathcal{Y}$,

$$\pi(t, y') = \log\left(\sum_{y'_{1:t} \in \mathcal{Y}^t :\ y'_t = y'} \exp(\textbf{score}_\theta(x_{1:T}, y'_{1:t}))\right)$$

$$\stackrel{*}{=} \log\left(\sum_{y \in \mathcal{Y}} \left(\sum_{y'_{1:t-1} \in \mathcal{Y}^{t-1} :\ y'_{t-1} = y} \exp(\textbf{score}_\theta(x_{1:T}, y'_{1:t-1}))\right) \times \exp(\textbf{score}_\theta(x_{1:T}, y, y', t))\right)$$

$$= \log\left(\sum_{y \in \mathcal{Y}} \exp(\pi(t-1, y) + \textbf{score}_\theta(x_{1:T}, y, y', t))\right)$$

- Runtime: $O(T\,|\mathcal{Y}|^2)$, quadratic dependence on label set size

Viterbi Algorithm for CRFs

- Goal: Find argmax of $\sum_{t=1}^{T}$ **score**$_\theta(x_{1:T}, y_{t-1}, y_t, t)$
- DP table $\pi(t, y) = \max_{y_{1:t} \in \mathcal{Y}^t :\, y_t = y}$ **score**$_\theta(x_{1:T}, y_{1:t})$
- Same base case: $\pi(1, y) =$ **score**$_\theta(x_{1:T}, y_0, y, 1)$ for all $y \in \mathcal{Y}$
- Main body: For $t = 2 \ldots T$, for all $y' \in \mathcal{Y}$,

$$\pi(t, y') = \max_{y \in \mathcal{Y}} \ \pi(t-1, y) + \textbf{score}_\theta(x_{1:T}, y, y', t)$$

- Recover the actual argmax label sequence by backtracking:

$$\beta(t, y') = \arg\max_{y \in \mathcal{Y}} \ \pi(t-1, y) + \textbf{score}_\theta(x_{1:T}, y, y', t)$$

$$y_T^* = \arg\max_y \ \pi(T, y),\ y_{T-1}^* = \beta(T, y_T^*),\ \ldots,\ y_1^* = \beta(2, y_2^*)$$

Neural Parameterization of CRF

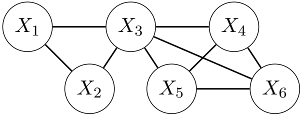
- Recall: We just need to define $\textbf{score}_\theta(x_{1:T}, y, y', t)$, from which we derive $\textbf{score}_\theta(x_{1:T}, y_{1:T})$.
- Typical parameterization (omitting biases)

$$\textbf{score}_\theta(x_{1:T}, y, y', t) = [\underbrace{\textbf{enc}_\theta(x_{1:T})}_{T \times d} \underbrace{W}_{d \times |\mathcal{Y}|}]_{t,y'} + [\underbrace{T}_{|\mathcal{Y}| \times |\mathcal{Y}|}]_{y,y'}$$

- $\textbf{enc}_\theta(x_{1:T})$: Any encoding of $x_{1:T}$, e.g., BiLSTM (Lample et al., 2016)
- Extra learnable parameters in the "CRF layer": W for computes per-position label logits, T for label transition scores
- Flexible, e.g., could define transition scores to be $v_y^\top A v_{y'}$ where $v_y \in \mathbb{R}^{d'}$ is a learnable embedding of label y

Undirected Graphical Models (UGMs/MRFs)

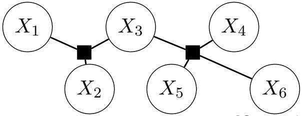
- CRF is a special case of a **undirected graphical model** (UGM), aka. **Markov random field** (MRF)
- Defines a joint distribution over variables that factorizes over *maximal* cliques C equipped with nonnegative **potential functions** ψ_C
  - Clique: A subset of nodes in MRF fully connected
  - Maximal clique: A clique that loses full connectivity if any node is added



$$\Pr(X_{1:6}) = \frac{1}{Z} \underbrace{\psi_{1:3}(X_{1:3})}_{\geq 0} \underbrace{\psi_{3:6}(X_{3:6})}_{\geq 0}$$

$$Z = \sum_{x_{1:6}} \psi_{1:3}(x_{1:3}) \psi_{3:6}(x_{3:6})$$

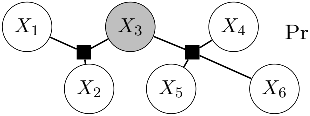
- More concisely, can write $\Pr(X) \propto \prod_C \psi_C(X_C)$, and use factor graph notation (square node fully connects neighbors)



$$\Pr(X_{1:6}) \propto \psi_{1:3}(X_{1:3}) \psi_{3:6}(X_{3:6})$$

Marginalization and Inference in MRFs

- Again, we typically observe part of a MRF. Then we work with a conditional joint distribution:



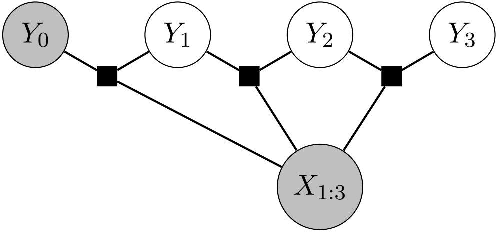
$$\Pr(X_{1:2}, X_{4:6} | X_3 = c) = \frac{1}{Z(X_3 = c)} \psi_{1:3}(X_1 X_2 c) \psi_{3:6}(X_{3:6})$$

$$Z(X_3 = c) = \sum_{x_{1:6} : x_3 = c} \psi_{1:3}(x_{1:3}) \psi_{3:6}(x_{3:6})$$

- MRF again poses general structured prediction problems, like
  - Marginalize: $\Pr(X_5 = c' | X_3 = c)$
  - Infer: $\arg\max_{x_{1:2}, x_{4:6}} \Pr(X_{1:2} = x_{1:2}, X_{4:6} = x_{4:6} | X_3 = c)$
- **Variable elimination (VE).** General "recipe" to solve these problems exactly in $O(n_{\text{infer}} m^{C_{\max}})$ time (assuming no cycles) where
  - n_{infer}: Number of variables in MRF that we're inferring
  - m: Number of possible values that variables can take
  - $C_{\max}$: Size of the *largest* maximal clique
- Too abstract to be directly useful (e.g., must specify elimination ordering), but provides a unified framework of structured prediction (e.g., forward, Viterbi are VE on chains)

CRFs as Conditional MRFs (Hence the Name)

- Given $x_{1:3}$, CRF considers the following MRF



- It has a clique at each step t consisting of at most two unobserved variables, with potential function defined as

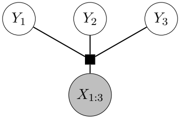
$$\psi_t(x_{1:T}, y, y') = \exp(\textbf{enc}_\theta(x_{1:T}, y, y', t)) \geq 0$$

- Distribution defined by

$$p_\theta(y_{1:3}|x_{1:3}) = \frac{\prod_{t=1}^{3} \psi_t(x_{1:3}, y_{t-1}, y_t)}{\sum_{y'_{1:3} \in \mathcal{Y}^3} \prod_{t=1}^{3} \psi_t(x_{1:3}, y'_{t-1}, y'_t)}$$

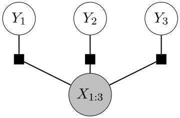
General Tagging with MRFs

- No independence assumptions: $O(T\,|Y|^T)$



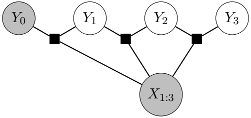
$$p_\theta(y_{1:3}|x_{1:3}) \propto \exp(\textbf{score}_\theta(x_{1:3}, y_{1:3}))$$
$$C_{\max} = 3$$

- Greedy tagging (i.e., softmax per position): $O(T\,|Y|)$



$$p_\theta(y_{1:3}|x_{1:3}) \propto \prod_{t=1}^{3} \exp(\textbf{score}_\theta(x_{1:3}, y_t, t))$$
$$C_{\max} = 1$$

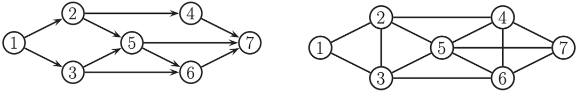
- First-order CRF: $O(T\,|Y|^2)$



$$p_\theta(y_{1:3}|x_{1:3}) \propto \prod_{t=1}^{3} \exp(\textbf{score}_\theta(x_{1:3}, y_{t-1}, y_t, t))$$
$$C_{\max} = 2$$

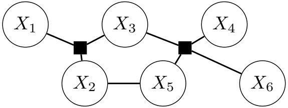
More Facts About Graphical Models

- Any (DAG-structured) DGM can be expressed by an MRF



(Image credit: Yunshu Liu)

  - Forward algorithm for HMM: VE with left-to-right elimination ordering
  - Generalizable to trees
- VE applicable only if there's no cycle (e.g., sequences, trees)
  - If cycle between unobserved variables, $O(n_{\text{infer}} m^{C_{\max}})$ runtime guarantee doesn't hold, e.g., marginalization intractable in



  - Can technically combine factors until there's no cycle and apply VE, but that's no better than brute-force
  - Efficient approximations possible: loopy belief propagation