NLP 201
Undirected Graphical Models

Jeffrey Flanigan

November 16, 2021

University of California Santa Cruz
jmflanig@ucsc.edu

Many slides and figures from David Sontag, Eric Xing, Matt Gormley and Noah Smith

- A2 is due today
- Please remember to cc me and Changmao when giving feedback
- There will be a 10% **penalty** on the final assignment for those who neglect to give feedback

Plan for Today

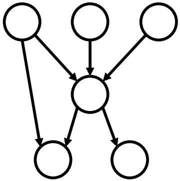
- Undirected graphical models (Markov Random Fields, MRFs)
- Conditional Random Fields (CRFs)
- Factor graphs

Graphical Models

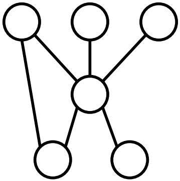
- Nodes represent random variables
- Edges represent dependence between random variables
- Trained to maximize joint or conditional probability

Three Types of Graphical Models

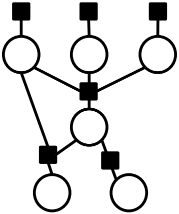
Directed Graphical Model



Undirected Graphical Model



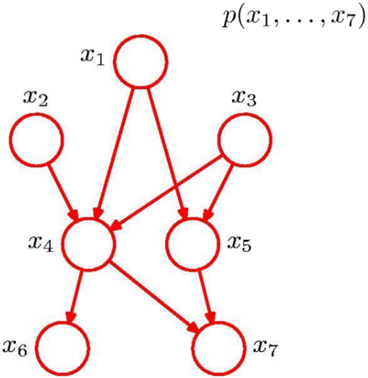
Factor Graph



Today: Undirected Graphical Models

But first: Independence in Bayesian Networks

Directed Graphical Models (Bayesian Networks)



General Factorization

$$p(\mathbf{x}) = \prod_{k=1}^{K} p(x_k | \mathrm{pa}_k)$$

Conditional Independence

a is independent of b given c

$$p(a|b,c) = p(a|c)$$

Equivalently

$$\begin{aligned} p(a,b|c) &= p(a|b,c)p(b|c) \\ &= p(a|c)p(b|c) \end{aligned}$$

Notation

$$a \perp\!\!\!\perp b \mid c$$

Definition: Markov Blanket

Given a node X, the **Markov blanket** for X is the minimal set of nodes that makes X conditionally independent of all the other nodes given the Markov blanket.

Let G be a graph, and let B be Markov blanket for X.

$$X \perp\!\!\!\perp (G - \{X\} - B) | B$$

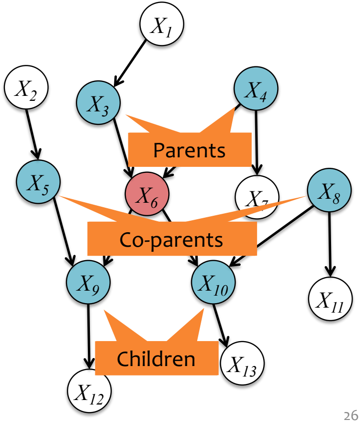
(If B is the Markov blanket of node X, then X is conditionally indep. of everything else in G given B)

Markov Blanket (Directed)

Def: the **co-parents** of a node are the parents of its children

Def: the **Markov Blanket** of a node in a directed graphical model is the set containing the node's parents, children, and co-parents.

Example: The Markov Blanket of X_6 is $\{X_3, X_4, X_5, X_8, X_9, X_{10}\}$



Undirected Graphical Models: Markov Random Fields

- An alternative representation for joint distributions is as an **undirected graphical model**
- As in BNs, we have one node for each random variable
- Rather than CPDs, we specify (non-negative) **potential functions** over sets of variables associated with cliques C of the graph,

$$p(x_1, \dots, x_n) = \frac{1}{Z} \prod_{c \in C} \phi_c(\mathbf{x}_c)$$

  Z is the **partition function** and normalizes the distribution:

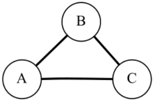
$$Z = \sum_{\hat{x}_1, \dots, \hat{x}_n} \prod_{c \in C} \phi_c(\hat{\mathbf{x}}_c)$$

- Like CPD's, $\phi_c(\mathbf{x}_c)$ can be represented as a table, but it is *not normalized*
- Also known as **Markov random fields** (MRFs) or Markov networks

Markov Random Field Example

$$p(x_1, \ldots, x_n) = \frac{1}{Z} \prod_{c \in C} \phi_c(\mathbf{x}_c), \qquad Z = \sum_{\hat{x}_1, \ldots, \hat{x}_n} \prod_{c \in C} \phi_c(\hat{\mathbf{x}}_c)$$

Simple example (potential function on each edge encourages the variables to take the same value):



$$\phi_{A,B}(a,b) = \begin{array}{cc|c|c|} & & \multicolumn{2}{c}{B} \\ & & 0 & 1 \\ \cline{3-4} \multirow{2}{*}{A} & 0 & 10 & 1 \\ \cline{3-4} & 1 & 1 & 10 \\ \cline{3-4} \end{array} \qquad \phi_{B,C}(b,c) = \begin{array}{cc|c|c|} & & \multicolumn{2}{c}{C} \\ & & 0 & 1 \\ \cline{3-4} \multirow{2}{*}{B} & 0 & 10 & 1 \\ \cline{3-4} & 1 & 1 & 10 \\ \cline{3-4} \end{array} \qquad \phi_{A,C}(a,c) = \begin{array}{cc|c|c|} & & \multicolumn{2}{c}{C} \\ & & 0 & 1 \\ \cline{3-4} \multirow{2}{*}{A} & 0 & 10 & 1 \\ \cline{3-4} & 1 & 1 & 10 \\ \cline{3-4} \end{array}$$

$$p(a, b, c) = \frac{1}{Z} \phi_{A,B}(a, b) \cdot \phi_{B,C}(b, c) \cdot \phi_{A,C}(a, c),$$

where

$$Z = \sum_{\hat{a}, \hat{b}, \hat{c} \in \{0,1\}^3} \phi_{A,B}(\hat{a}, \hat{b}) \cdot \phi_{B,C}(\hat{b}, \hat{c}) \cdot \phi_{A,C}(\hat{a}, \hat{c}) = 2 \cdot 1000 + 6 \cdot 10 = 2060.$$

Board work

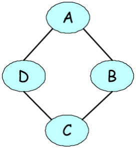
(3 min) On your own:

- In the previous example, compute $Pr(A = 1, B = 1, C = 1)$

(3 min) Discuss with a partner

Hair color example as a MRF

- We now have an **undirected** graph:



- The joint probability distribution is parameterized as

$$p(a, b, c, d) = \frac{1}{Z}\phi_{AB}(a, b)\phi_{BC}(b, c)\phi_{CD}(c, d)\phi_{AD}(a, d)\ \phi_A(a)\phi_B(b)\phi_C(c)\phi_D(d)$$

- **Pairwise potentials** enforce that no friend has the same hair color:

$$\phi_{AB}(a, b) = 0 \text{ if } a = b, \quad \text{and } 1 \text{ otherwise}$$

- **Single-node potentials** specify an affinity for a particular hair color, e.g.

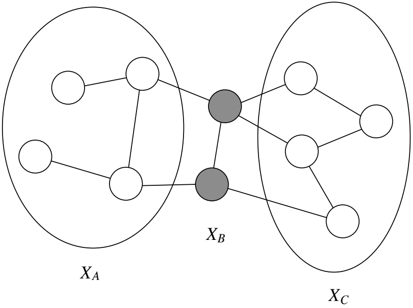
$$\phi_D(\text{“red”}) = 0.6, \quad \phi_D(\text{“blue”}) = 0.3, \quad \phi_D(\text{“green”}) = 0.1$$

The normalization Z makes the potentials **scale invariant**! Equivalent to

$$\phi_D(\text{“red”}) = 6, \quad \phi_D(\text{“blue”}) = 3, \quad \phi_D(\text{“green”}) = 1$$

Markov network structure implies conditional independencies

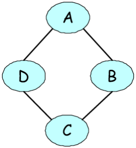
- Let G be the undirected graph where we have one edge for every pair of variables that appear together in a potential
- Conditional independence is given by **graph separation**!



- $X_{\mathbf{A}} \perp X_{\mathbf{C}} \mid X_{\mathbf{B}}$ if there is no path from $a \in \mathbf{A}$ to $c \in \mathbf{C}$ after removing all variables in $\mathbf{B}$

Example

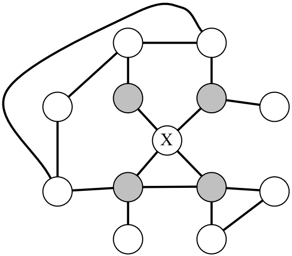
- Returning to hair color example, its undirected graphical model is:



- Since removing A and C leaves no path from D to B, we have $D \perp B \mid \{A, C\}$
- Similarly, since removing D and B leaves no path from A to C, we have $A \perp C \mid \{D, B\}$
- No other independencies implied by the graph

Markov blanket

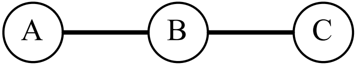
- A set **U** is a **Markov blanket** of X if $X \notin \mathbf{U}$ and if **U** is a minimal set of nodes such that $X \perp (\mathcal{X} - \{X\} - \mathbf{U}) \mid \mathbf{U}$
- In undirected graphical models, the Markov blanket of a variable is precisely its **neighbors** in the graph:



- In other words, X is independent of the rest of the nodes in the graph given its immediate neighbors

Proof of independence through separation

- We will show that $A \perp C \mid B$ for the following distribution:



$$p(a,b,c) = \frac{1}{Z}\phi_{AB}(a,b)\phi_{BC}(b,c)$$

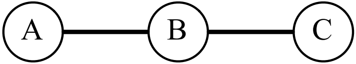
- First, we show that $p(a \mid b)$ can be computed using only $\phi_{AB}(a,b)$:

$$\begin{aligned}
p(a \mid b) &= \frac{p(a,b)}{p(b)} \\
&= \frac{\frac{1}{Z}\sum_{\hat{c}} \phi_{AB}(a,b)\phi_{BC}(b,\hat{c})}{\frac{1}{Z}\sum_{\hat{a},\hat{c}} \phi_{AB}(\hat{a},b)\phi_{BC}(b,\hat{c})} \\
&= \frac{\phi_{AB}(a,b)\sum_{\hat{c}} \phi_{BC}(b,\hat{c})}{\sum_{\hat{a}} \phi_{AB}(\hat{a},b)\sum_{\hat{c}} \phi_{BC}(b,\hat{c})} = \frac{\phi_{AB}(a,b)}{\sum_{\hat{a}} \phi_{AB}(\hat{a},b)}.
\end{aligned}$$

- More generally, the probability of a variable conditioned on its Markov blanket depends *only* on potentials involving that node

Proof of independence through separation

- We will show that $A \perp C \mid B$ for the following distribution:



$$p(a, b, c) = \frac{1}{Z}\phi_{AB}(a, b)\phi_{BC}(b, c)$$

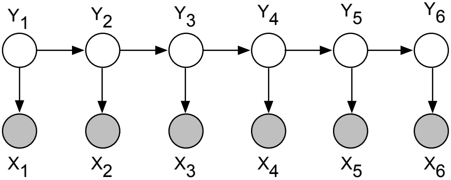
Proof.

$$\begin{aligned} p(a, c \mid b) = \frac{p(a, c, b)}{\sum_{\hat{a},\hat{c}} p(\hat{a}, b, \hat{c})} &= \frac{\phi_{AB}(a, b)\phi_{BC}(b, c)}{\sum_{\hat{a},\hat{c}} \phi_{AB}(\hat{a}, b)\phi_{BC}(b, \hat{c})} \\ &= \frac{\phi_{AB}(a, b)\phi_{BC}(b, c)}{\sum_{\hat{a}} \phi_{AB}(\hat{a}, b) \sum_{\hat{c}} \phi_{BC}(b, \hat{c})} \\ &= p(a \mid b)p(c \mid b) \end{aligned}$$

$\square$

Converting BNs to Markov networks

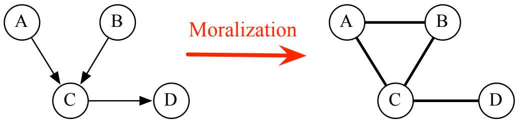
What is the equivalent Markov network for a hidden Markov model?



Many inference algorithms are more conveniently given for undirected models – this shows how they can be applied to Bayesian networks

Moralization of Bayesian networks

- Procedure for converting a Bayesian network into a Markov network
- The **moral graph** $\mathcal{M}[G]$ of a BN $G = (V, E)$ is an undirected graph over V that contains an undirected edge between X_i and X_j if
  1. there is a directed edge between them (in either direction)
  2. X_i and X_j are both parents of the same node

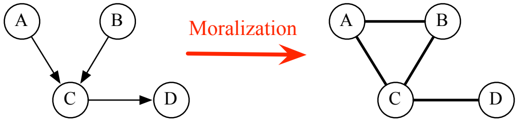


(term historically arose from the idea of "marrying the parents" of the node)

- The addition of the moralizing edges leads to the loss of some independence information, e.g., $A \rightarrow C \leftarrow B$, where $A \perp B$ is lost

Converting BNs to Markov networks

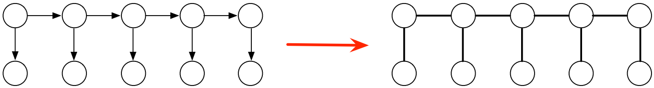
1. Moralize the directed graph to obtain the undirected graphical model:



2. Introduce one potential function for each CPD:

$$\phi_i(x_i, \mathbf{x}_{pa(i)}) = p(x_i \mid \mathbf{x}_{pa(i)})$$

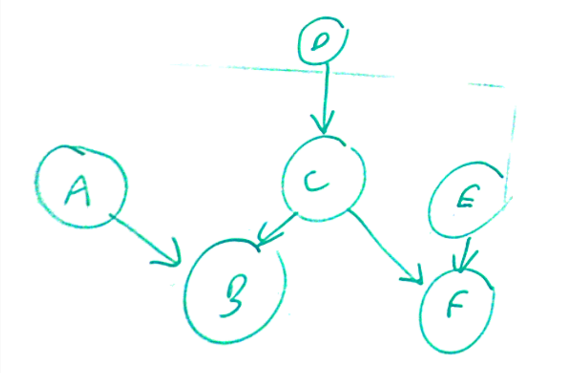
- So, converting a hidden Markov model to a Markov network is simple:



- For variables having > 1 parent, factor graph notation is useful

Board work

(3 min) On your own, convert the following Bayesian Network to an MRF



(example from whiteboard)

Conditional Random Fields

- **Conditional random fields** are undirected graphical models of conditional distributions $p(\mathbf{Y}|\mathbf{X})$
  - $\mathbf{Y}$ is a set of **target variables**
  - $\mathbf{X}$ is a set of **output variables**
- Potentials are functions of $\mathbf{X}$ and $\mathbf{Y}$

Conditional Random Fields

Formal Definition

- A CRF is a Markov network on variables $\mathbf{X} \cup \mathbf{Y}$, which specifies the conditional distribution

$$P(\mathbf{y} \mid \mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{c \in C} \phi_c(\mathbf{x}_c, \mathbf{y}_c)$$

  with partition function

$$Z(\mathbf{x}) = \sum_{\hat{\mathbf{y}}} \prod_{c \in C} \phi_c(\mathbf{x}_c, \hat{\mathbf{y}}_c).$$

- As before, two variables in the graph are connected with an undirected edge if they appear together in the scope of some factor
- The only difference with a standard Markov network is the normalization term – before marginalized over $\mathbf{X}$ and $\mathbf{Y}$, now only over $\mathbf{Y}$

Parameterization of CRFs

- Factors may depend on a large number of variables
- We typically parameterize each factor as a log-linear function,

$$\phi_c(\mathbf{x}_c, \mathbf{y}_c) = \exp\{\mathbf{w} \cdot \mathbf{f}_c(\mathbf{x}_c, \mathbf{y}_c)\}$$

- $\mathbf{f}_c(\mathbf{x}_c, \mathbf{y}_c)$ is a feature vector
- $\mathbf{w}$ is a weight vector which is typically learned – we will discuss this extensively in later lectures

Compare to our previous definition of a (linear-chain) CRF

CRF (two lectures ago)

CRF trained to maximize:

$$P(\boldsymbol{y}|\boldsymbol{x}) = \frac{\exp(score(\boldsymbol{x}, \boldsymbol{y}))}{\sum_{\boldsymbol{y}'} \exp(score(\boldsymbol{x}, \boldsymbol{y}'))} = \frac{\exp(score(\boldsymbol{x}, \boldsymbol{y}))}{Z}$$

Using Forward algorithm, we can compute:

$$Z = \sum_{\boldsymbol{y}'} e^{score(\boldsymbol{x}, \boldsymbol{y}')} = \sum_{\mathbf{y}} \prod_{i=1}^{n} e^{s(\mathbf{x}, i, y_i, y_{i-1})}$$

$score$ is a sum of “local parts”:

$$score(\boldsymbol{x}, \boldsymbol{y}) = \textstyle\sum_{i=1}^{n} s(\mathbf{x}, i, y_i, y_{i-1})$$

$e^{s(\mathbf{x}, i, y_i, y_{i-1})}$ **are the potential functions**

Conditional random fields

$$P(y \mid x, \beta) = \frac{\exp(\Phi(x,y)^\top \beta)}{\sum_{y' \in \mathcal{Y}} \exp(\Phi(x,y')^\top \beta)}$$

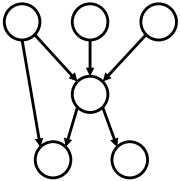
Feature vector scoped over the entire input and label sequence

$$\Phi(x,y) = \sum_{i=1}^{n} \phi(x, i, y_i, y_{i-1})$$

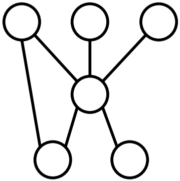
ϕ is the same feature vector we used for local predictions using MEMMs

Three Types of Graphical Models

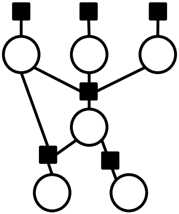
Directed Graphical Model



Undirected Graphical Model



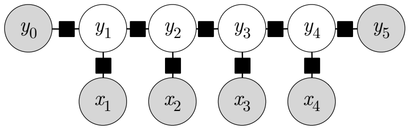
Factor Graph



Factor graphs

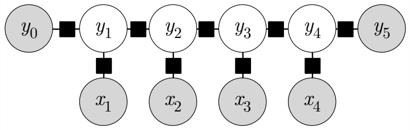
Factor Graph Example

$$\hat{\mathbf{y}} = \underset{\mathbf{y}}{\operatorname{argmax}} \sum_{i=1}^{n} \log(p(x_i|y_i)) + \log(p(y_i|y_{i-1}))$$



- Like Bayesian networks, **factor graphs** are a graphical model
- Each box represents a **local factor**, which is a function that depends on the R.V.s it is connected to
- The total score is the product (or sum) of the factors

Factor Graphs



- Like Bayesian networks, **factor graphs** are a graphical model
- Each box represents a **local factor**, which is a function that depends on the R.V.s it is connected to
- The total score is the product (or sum) of the factors

$$S(\mathbf{x}) = \prod_s \psi_s(\mathbf{x}_s)$$

where s runs over the factors and $\mathbf{x}_s$ denotes the subset of variables in factor s

How General Are Factor Graphs?

- Factor graphs can be used to describe
  - Markov Random Fields (undirected graphical models)
    - i.e., log-linear models over a tuple of variables
  - Conditional Random Fields
  - Bayesian Networks (directed graphical models)

- *Inference* treats all of these interchangeably.
  - Convert your model to a factor graph first.
  - Pearl (1988) gave key strategies for *exact* inference:
    - **Belief propagation**, for inference on *acyclic* graphs
    - **Junction tree algorithm**, for making *any* graph acyclic (by merging variables and factors: blows up the runtime)

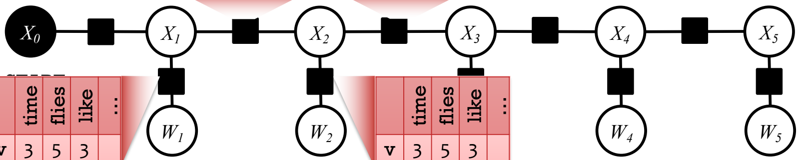
Factors have local opinions (≥ 0)

Each black box looks at *some* of the tags X_i and words W_i

	v	n	p	d
v	1	6	3	4
n	8	4	2	0.1
p	1	3	1	3
d	0.1	8	0	0

	v	n	p	d
v	1	6	3	4
n	8	4	2	0.1
p	1	3	1	3
d	0.1	8	0	0

Note: We chose to reuse the same factors at different positions in the sentence.



	time	flies	like	…
v	3	5	3	
n	4	5	2	
p	0.1	0.1	3	
d	0.1	0.2	0.1	

	time	flies	like	…
v	3	5	3	
n	4	5	2	
p	0.1	0.1	3	
d	0.1	0.2	0.1	

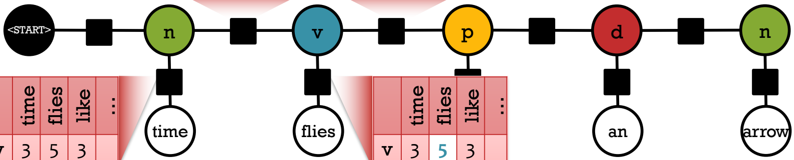
Markov Random Field (MRF)

Joint distribution over tags X_i <u>and</u> words W_i
The individual factors aren't *necessarily* probabilities.

$$p(\text{n, v, p, d, n, time, flies, like, an, arrow}) = \frac{1}{Z}(4 * 8 * 5 * 3 * \ldots)$$

	v	n	p	d
v	1	6	3	4
n	8	4	2	0.1
p	1	3	1	3
d	0.1	8	0	0

	v	n	p	d
v	1	6	3	4
n	8	4	2	0.1
p	1	3	1	3
d	0.1	8	0	0



	time	flies	like	…
v	3	5	3	
n	4	5	2	
p	0.1	0.1	3	
d	0.1	0.2	0.1	

	time	flies	like	…
v	3	5	3	
n	4	5	2	
p	0.1	0.1	3	
d	0.1	0.2	0.1	

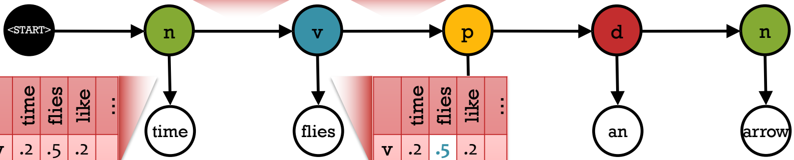
Bayesian Networks

But sometimes we *choose* to make them probabilities.
Constrain each row of a factor to sum to one. Now $Z = 1$.

$$p(\text{n, v, p, d, n, time, flies, like, an, arrow}) = \cancel{\frac{1}{Z}}(.3 * .8 * .2 * .5 * \ldots)$$

	v	n	p	d
v	.1	.4	.2	.3
n	.8	.1	.1	0
p	.2	.3	.2	.3
d	.2	.8	0	0

	v	n	p	d
v	.1	.4	.2	.3
n	.8	.1	.1	0
p	.2	.3	.2	.3
d	.2	.8	0	0

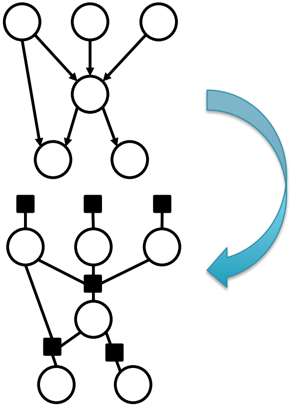


	time	flies	like	…
v	.2	.5	.2	
n	.3	.4	.2	
p	.1	.1	.3	
d	.1	.2	.1	

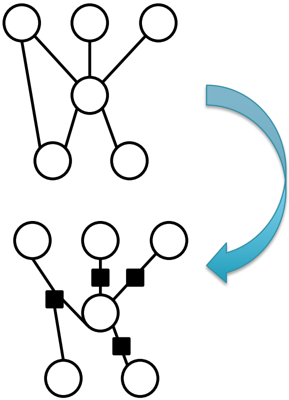
	time	flies	like	…
v	.2	.5	.2	
n	.3	.4	.2	
p	.1	.1	.3	
d	.1	.2	.1	

Converting to Factor Graphs

Each conditional and marginal distribution in a **directed GM** becomes a factor

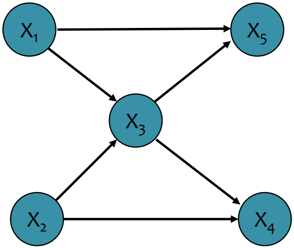


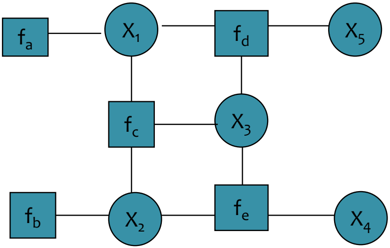
Each maximal clique in an **undirected GM** becomes a factor



Factor Graph Examples

- Example 1



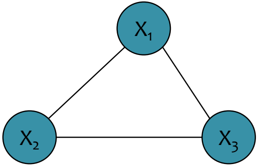


$P(X_1)$ $P(X_2)$ $P(X_3|X_1,X_2)$ $P(X_5|X_1,X_3)$ $P(X_4|X_2,X_3)$

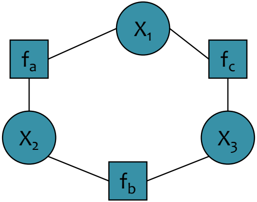
$f_a(X_1)$ $f_b(X_2)$ $f_c(X_3,X_1,X_2)$ $f_d(X_5,X_1,X_3)$ $f_e(X_4,X_2,X_3)$

Factor Graph Examples

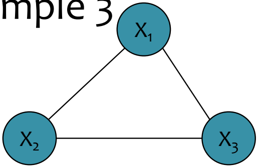
- Example 2



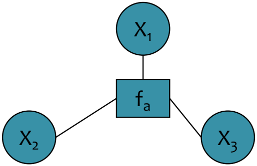
$\psi(x_1,x_2,x_3) = f_a(x_1,x_2)f_b(x_2,x_3)f_c(x_3,x_1)$



- Example 3



$\psi(x_1,x_2,x_3) = f_a(x_1,x_2,x_3)$



Board work

(3 min) On your own, write down the factor graph for the linear chain CRF we had from before:

$$P(\boldsymbol{y}|\boldsymbol{x}) = \frac{\exp(score(\boldsymbol{x}, \boldsymbol{y}))}{\sum_{\boldsymbol{y}'} \exp(score(\boldsymbol{x}, \boldsymbol{y}'))} = \frac{\exp(score(\boldsymbol{x}, \boldsymbol{y}))}{Z}$$

$$Z = \sum_{\boldsymbol{y}'} e^{score(\boldsymbol{x}, \boldsymbol{y}')} = \sum_{\mathbf{y}} \prod_{i=1}^{n} e^{s(\mathbf{x}, i, y_i, y_{i-1})}$$

$e^{s(\mathbf{x}, i, y_i, y_{i-1})}$ are the potential functions

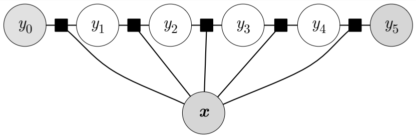
Board work

(3 min) On your own, write down the factor graph for the linear chain CRF we had from before:

$$P(\boldsymbol{y}|\boldsymbol{x}) = \frac{\exp(score(\boldsymbol{x},\boldsymbol{y}))}{\sum_{\boldsymbol{y}'}\exp(score(\boldsymbol{x},\boldsymbol{y}'))} = \frac{\exp(score(\boldsymbol{x},\boldsymbol{y}))}{Z}$$

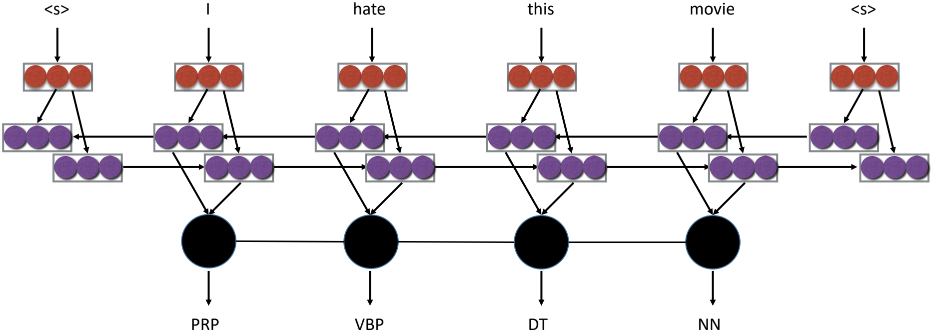
$$Z = \sum_{\boldsymbol{y}'} e^{score(\boldsymbol{x},\boldsymbol{y}')} = \sum_{\mathbf{y}} \prod_{i=1}^{n} e^{s(\mathbf{x},i,y_i,y_{i-1})}$$

$e^{s(\mathbf{x},i,y_i,y_{i-1})}$ are the potential functions



Revisiting our neural CRF example

BiLSTM-CRF for Sequence Labeling



Potential Functions

- $\psi_i(y_{i-1}, y_i, X) = \exp\left(W^T T(y_{i-1}, y_i, X, i) + U^T\ S(y_i, X, i) + b_{y_{i-1}, y_i}\right)$
  - Using neural features in DNN:
    $\psi_i(y_{i-1}, y_i, X) = \exp\left(W_{y_{i-1}, y_i}^T F(X, i) + U_{y_i}^T\ F(X, i) + b_{y_{i-1}, y_i}\right)$
    - Number of parameters: $O(|Y|^2 d_F)$
  - Simpler version:
    $\psi_i(y_{i-1}, y_i, X) = \exp\left(W_{y_{i-1}, y_i} + U_{y_i}^T F(X, i) + b_{y_{i-1}, y_i}\right)$
    - Number of parameters: $O(|Y|^2 + |Y| d_F)$

CRF Training & Decoding

- $P(Y|X) = \frac{\prod_{i=1}^{L} \psi_i(y_{i-1}, y_i, X)}{\sum_{Y'} \prod_{i=1}^{L} \psi_i(y'_{i-1}, y'_i, X)} = \frac{\prod_{i=1}^{L} \psi_i(y_{i-1}, y_i, X)}{Z(X)}$
- Training: computing the partition function Z(X)

$$Z(X) = \sum_{Y} \prod_{i=1}^{L} \psi_i(y_{i-1}, y_i, X)$$

- Decoding

$$y^* = argmax_Y P(Y|X)$$

Go through the output space of Y which grows exponentially with the length of the input sequence.

Viterbi Algorithm

- $\pi_t(y|\mathrm{X})$ is the partition of sequence with length equal to t and end with label y:

$$\pi_t(y|X) = \sum_{y_i,\ldots,y_{t-1}} \left(\prod_{i=1}^{t-1} \psi_i(y_{i-1}, y_i, X)\right) \psi_t(y_{t-1}, y_t = y, X)$$

$$= \sum_{y_{t-1}} \psi_t(y_{t-1}, y_t = y, X) \sum_{y_i,\ldots,y_{t-2}} \left(\prod_{i=1}^{t-2} \psi_i(y_{i-1}, y_i, X)\right) \psi_{t-1}(y_{t-2}, y_{t-1}, X)$$

$$= \sum_{y_{t-1}} \psi_t(y_{t-1}, y_t = y, X) \pi_{t-1}(y_{t-1}|X)$$

- Computing partition function $Z(X) = \sum_y \pi_L(y|X)$

Viterbi Algorithm

- Decoding is performed with similar dynamic programming algorithm
- Calculating gradient: $l_{ML}(X, Y; \theta) = -\log P(Y|X; \theta)$

$$\frac{\partial l_{ML}(X, Y; \theta)}{\partial \theta} = F(Y, X) - E_{P(Y|X;\theta)}[F(Y, X)]$$

  - Forward-backward algorithm (Sutton and McCallum, 2010)
    - Both $P(Y|X; \theta)$ and $F(Y, X)$ can be decomposed
    - Need to compute the marginal distribution:

$$P(y_{i-1} = y', y_i = y|X; \theta) = \frac{\alpha_{i-1}(y'|X)\psi_i(y', y, X)\beta_i(y|X)}{Z(X)}$$

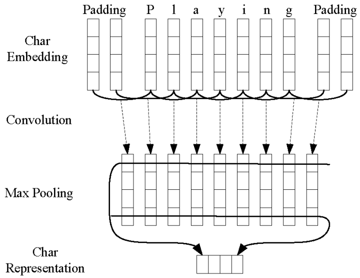
  - Not necessary if using DNN framework (auto-grad)

Case Study: BiLSTM-CNN-CRF for Sequence Labeling (Ma et al, 2016)

- Goal: Build a truly end-to-end neural model for sequence labeling task, requiring no feature engineering and data pre-processing.
- Two levels of representations
  - Character-level representation: CNN
  - Word-level representation: Bi-directional LSTM

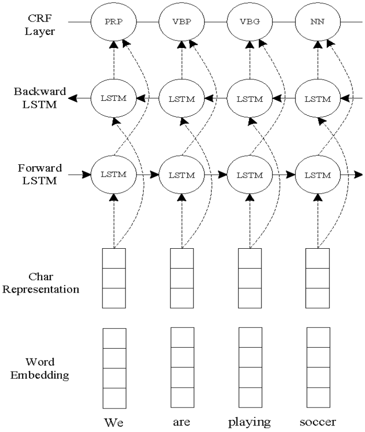
CNN for Character-level representation

- We used CNN to extract morphological information such as prefix or suffix of a word



Bi-LSTM-CNN-CRF

- We used Bi-LSTM to model word-level information.
- CRF is on top of Bi-LSTM to consider the co-relation between labels.



Training Details

- Optimization Algorithm:
  - SGD with momentum (0.9)
  - Learning rate decays with rate 0.05 after each epoch.
- Dropout Training:
  - Applying dropout to regularize the model with fixed dropout rate 0.5
- Parameter Initialization:
  - Parameters: Glorot and Bengio (2010)
  - Word Embedding: Stanford's GloVe 100-dimentional embeddings
  - Character Embedding: uniformly sampled from $[-\sqrt{\frac{3}{dim}}, +\sqrt{\frac{3}{dim}}]$, where $dim = 30$

Experiments

Model	POS		NER					
	Dev	Test	Dev			Test		
	Acc.	Acc.	Prec.	Recall	F1	Prec.	Recall	F1
BRNN	96.56	96.76	92.04	89.13	90.56	87.05	83.88	85.44
BLSTM	96.88	96.93	92.31	90.85	91.57	87.77	86.23	87.00
BLSTM-CNN	97.34	97.33	92.52	93.64	93.07	88.53	90.21	89.36
BLSTM-CNN-CRF	97.46	97.55	94.85	94.63	94.74	91.35	91.06	91.21

End

We stopped here.