Probability and Structure in Natural Language Processing

Noah Smith

Heidelberg University, November 2014
Introduction
Motivation

• Statistical methods in NLP arrived ~20 years ago and now dominate.

• Mercer was right: “There's no data like more data.”
  – And there's more and more data.

• Lots of new applications and new statistical techniques – it's formidable to learn and keep up with all of them.
Thesis

• Most of the main ideas are related and similar to each other.
  – Different approaches to decoding.
  – Different learning criteria.
  – Supervised and unsupervised learning.
• Umbrella: probabilistic reasoning about discrete linguistic structures.

• This is good news!
Plan

1. Graphical models and inference  Monday
2. Decoding and structures  Tuesday
3. Supervised learning  Wednesday
4. Hidden variables  Thursday
Exhortations

• The content is formal, but the style doesn't need to be.

• Ask questions!
  – Help me find the right pace.
  – Lecture 4 can be dropped/reduced if needed.
Lecture 1: Graphical Models and Inference
Random Variables

• Probability distributions usually defined by **events**
• Events are complicated!
  – We tend to *group* events by **attributes**
  – Person → Age, Grade, HairColor
• **Random variables** formalize attributes:
  – “Grade = A” is shorthand for event
    \[ \{ \omega \in \Omega : f_{\text{Grade}}(\omega) = A \} \]
• Properties of random variable X:
  – Val(X) = possible values of X
  – For discrete (categorical):
    \[ \sum_{x \in \text{Val}(X)} P(X = x) = 1 \]
  – For continuous:
    \[ \int_{x \in \text{Val}(X)} P(X = x) dx = 1 \]
  – Nonnegativity:
    \[ \forall x \in \text{Val}(X), P(X = x) \geq 0 \]
Conditional Probabilities

- After learning that $\alpha$ is true, how do we feel about $\beta$? $P(\beta \mid \alpha)$
Chain Rule

\[ P(\alpha \cap \beta) = P(\alpha)P(\beta \mid \alpha) \]

\[ P(\alpha_1 \cap \cdots \cap \alpha_k) = P(\alpha_1)P(\alpha_2 \mid \alpha_1) \cdots P(\alpha_k \mid \alpha_1 \cap \cdots \cap \alpha_{k-1}) \]
Bayes Rule

\[ P(\alpha \mid \beta) = \frac{P(\beta \mid \alpha)P(\alpha)}{P(\beta)} \]

\[ P(\alpha \mid \beta \cap \gamma) = \frac{P(\beta \mid \alpha \cap \gamma)P(\alpha \mid \gamma)}{P(\beta \mid \gamma)} \]

\( \gamma \) is an “external event”
Independence

• $\alpha$ and $\beta$ are **independent** if $P(\beta | \alpha) = P(\beta)$
  
  $P \rightarrow (\alpha \perp \beta)$

• **Proposition:** $\alpha$ and $\beta$ are **independent** if and only if $P(\alpha \cap \beta) = P(\alpha) \cdot P(\beta)$
Conditional Independence

• Independence is rarely true.

• \( \alpha \) and \( \beta \) are **conditionally independent** given \( \gamma \) if
\[
P(\beta \mid \alpha \cap \gamma) = P(\beta \mid \gamma)
\]

\[
P \rightarrow (\alpha \perp \beta \mid \gamma)
\]

**Proposition:** \( P \rightarrow (\alpha \perp \beta \mid \gamma) \) if and only if
\[
P(\alpha \cap \beta \mid \gamma) = P(\alpha \mid \gamma) P(\beta \mid \gamma)
\]
Joint Distribution and Marginalization

\[ P(\text{Grade, Intelligence}) = \]

<table>
<thead>
<tr>
<th></th>
<th>Intelligence = very high</th>
<th>Intelligence = high</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grade = A</td>
<td>0.70</td>
<td>0.10</td>
</tr>
<tr>
<td>Grade = B</td>
<td>0.15</td>
<td>0.05</td>
</tr>
</tbody>
</table>

- Compute the marginal over each individual random variable?
Marginalization: General Case

\[ p(X_1 = x) = \sum_{x_2 \in \text{Val}(X_2)} \cdots \sum_{x_n \in \text{Val}(X_n)} P(X_1 = x, X_2 = x_2, \ldots, X_n = x_n) \]

How many terms?
Basic Concepts So Far

- **Atomic outcomes**: assignment of $x_1,\ldots,x_n$ to $X_1,\ldots,X_n$

- **Conditional probability**: $P(X, Y) = P(X) \ P(Y|X)$

- **Bayes rule**: $P(X|Y) = P(Y|X) \ P(X) / P(Y)$

- **Chain rule**: $P(X_1,\ldots,X_n) = P(X_1) \ P(X_2|X_1) \ldots \ P(X_k|X_1,\ldots,X_{k-1})$

- **Marginals**: deriving $P(X = x)$ from $P(X, Y)$
Sets of Variables

• **Sets** of variables $X, Y, Z$

• $X$ is independent of $Y$ given $Z$ if

$$P \rightarrow (X=x \perp Y=y | Z=z), \forall x \in \text{Val}(X), y \in \text{Val}(Y), z \in \text{Val}(Z)$$

• Shorthand:
  
  – Conditional independence: $P \rightarrow (X \perp Y | Z)$
  – For $P \rightarrow (X \perp Y | \emptyset)$, write $P \rightarrow (X \perp Y)$

• **Proposition:** $P$ satisfies $(X \perp Y | Z)$ if and only if

$$P(X,Y|Z) = P(X|Z) \cdot P(Y|Z)$$
Factor Graphs
Factor Graphs

• Random variable nodes (circles)
• Factor nodes (squares)
• Edge between variable and factor if the factor depends on that variable.
  – The graph is bipartite.
• A factor is a function from tuples of r.v. values to nonnegative numbers.

\[ P(X = \mathbf{x}) \propto \prod_j \phi_j(\mathbf{x}_j) \]
Two Kinds of Factors

• Conditional probability tables
  – E.g., \( P(X_2 \mid X_1, X_3) \)
  – Leads to Bayesian networks, causal explanations

• Potential functions
  – Arbitrary positive scores
  – Leads to Markov networks
Example: Bayesian Network

- The flu causes sinus inflammation
- Allergies *also* cause sinus inflammation
- Sinus inflammation causes a runny nose
- Sinus inflammation causes headaches
• The flu causes sinus inflammation
• Allergies also cause sinus inflammation
• Sinus inflammation causes a runny nose
• Sinus inflammation causes headaches

"Some local configurations are more likely than others."
"Some local configurations are more likely than others."
Example: Markov Network

- Swinging couples or confused students

\[ A \perp C \mid B, D \]
\[ B \perp D \mid A, C \]
\[ \neg B \perp D \]
\[ \neg A \perp C \]
Example: Markov Network

- Each random variable is a vertex.
- Undirected edges.
- **Factors** are associated with subsets of nodes that form cliques.
  - A factor maps assignments of its nodes to nonnegative values.
• In this example, associate a factor with each edge.
  – Could also have factors for single nodes!
Markov Networks

- **Probability distribution:**

\[
P(a, b, c, d) \propto \phi_1(a, b)\phi_2(b, c)\phi_3(c, d)\phi_4(a, d)
\]

\[
P(a, b, c, d) = \frac{\phi_1(a, b)\phi_2(b, c)\phi_3(c, d)\phi_4(a, d)}{\sum_{a', b', c', d'} \phi_1(a', b')\phi_2(b', c')\phi_3(c', d')\phi_4(a', d')}
\]

\[
Z = \sum_{a', b', c', d'} \phi_1(a', b')\phi_2(b', c')\phi_3(c', d')\phi_4(a', d')
\]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>(\varphi_1(A, B))</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>30</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>10</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>B</th>
<th>C</th>
<th>(\varphi_2(B, C))</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>100</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C</th>
<th>D</th>
<th>(\varphi_3(C, D))</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>100</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A</th>
<th>D</th>
<th>(\varphi_4(A, D))</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>100</td>
</tr>
</tbody>
</table>
Example: Markov Network

- Probability distribution:

\[ P(a, b, c, d) \propto \phi_1(a, b) \phi_2(b, c) \phi_3(c, d) \phi_4(a, d) \]

\[ P(a, b, c, d) = \frac{\phi_1(a, b) \phi_2(b, c) \phi_3(c, d) \phi_4(a, d)}{\sum_{a', b', c', d'} \phi_1(a', b') \phi_2(b', c') \phi_3(c', d') \phi_4(a', d')} \]

\[ Z = \sum_{a', b', c', d'} \phi_1(a', b') \phi_2(b', c') \phi_3(c', d') \phi_4(a', d') \]

\[ = 7,201,840 \]
Example: Markov Network

- **Probability distribution:**

$$P(a, b, c, d) \propto \phi_1(a, b)\phi_2(b, c)\phi_3(c, d)\phi_4(a, d)$$

$$P(a, b, c, d) = \frac{\phi_1(a, b)\phi_2(b, c)\phi_3(c, d)\phi_4(a, d)}{\sum_{a', b', c', d'} \phi_1(a', b')\phi_2(b', c')\phi_3(c', d')\phi_4(a', d')}$$

$$Z = \sum_{a', b', c', d'} \phi_1(a', b')\phi_2(b', c')\phi_3(c', d')\phi_4(a', d')$$

$$= 7,201,840$$

$P(0, 1, 1, 0) = \frac{5,000,000}{Z} = 0.69$
Example: Markov Network

- Probability distribution:

\[ P(a, b, c, d) \propto \phi_1(a, b) \phi_2(b, c) \phi_3(c, d) \phi_4(a, d) \]

\[ P(a, b, c, d) = \frac{\phi_1(a, b) \phi_2(b, c) \phi_3(c, d) \phi_4(a, d)}{\sum_{a', b', c', d'} \phi_1(a', b') \phi_2(b', c') \phi_3(c', d') \phi_4(a', d')} \]

\[ Z = \sum_{a', b', c', d'} \phi_1(a', b') \phi_2(b', c') \phi_3(c', d') \phi_4(a', d') \]

= 7,201,840

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>( \phi_1(A, B) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>30</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>10</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>B</th>
<th>C</th>
<th>( \phi_1(B, C) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>100</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C</th>
<th>D</th>
<th>( \phi_1(C, D) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>100</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A</th>
<th>D</th>
<th>( \phi_1(A, D) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>100</td>
</tr>
</tbody>
</table>

\[ P(1, 1, 0, 0) = \frac{10}{Z} = 0.0000014 \]
Independence and Structure

• There's a *lot* of theory about how BNs and MNs encode conditional independence assumptions.
  – **BNs**: A variable X is independent of its non-descendants given its parents.
  – **MNs**: Conditional independence derived from “Markov blanket” and *separation* properties.
  – Local configurations can be used to check *all* conditional independence questions; almost no need to look at the values in the factors!
Independence Spectrum

\[ \prod_{i} \phi_i(x_i) \]

full independence assumptions

\[ \phi(\mathbf{x}) \]

everything is dependent

various graphs
Products of Factors

• Given two factors with different scopes, we can calculate a new factor equal to their products.

\[ \phi_{product}(x \cup y) = \phi_1(x) \cdot \phi_2(y) \]
Products of Factors

• Given two factors with different scopes, we can calculate a new factor equal to their products.
Factor Maximization

• Given $X$ and $Y$ ($Y \notin X$), we can turn a factor $\varphi(X, Y)$ into a factor $\psi(X)$ via maximization:

$$\psi(X) = \max_Y \varphi(X, Y)$$

• We can refer to this new factor by $\max_Y \varphi$. 
Factor Maximization

Given \( X \) and \( Y \) (\( Y \notin X \)), we can turn a factor \( \phi(X, Y) \) into a factor \( \psi(X) \) via maximization:

\[
\psi(X) = \max_Y \phi(X, Y)
\]
Factor Marginalization

• Given $X$ and $Y$ ($Y \notin X$), we can turn a factor $\varphi(X, Y)$ into a factor $\psi(X)$ via marginalization:

$$\psi(X) = \sum_{y \in \text{Val}(Y)} \phi(X, y)$$
Factor Marginalization

- Given \( X \) and \( Y \) (\( Y \notin X \)), we can turn a factor \( \varphi(X, Y) \) into a factor \( \psi(X) \) via marginalization:

\[
\psi(X) = \sum_{y \in \text{Val}(Y)} \varphi(X, y)
\]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>( \varphi(A, B, C) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.9</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0.3</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1.1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1.7</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0.4</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0.7</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1.1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.2</td>
</tr>
</tbody>
</table>

\text{“summing out” B}

<table>
<thead>
<tr>
<th>A</th>
<th>C</th>
<th>( \psi(A, C) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>2.0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>2.0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1.5</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0.9</td>
</tr>
</tbody>
</table>
Factor Marginalization

- Given $X$ and $Y$ ($Y \notin X$), we can turn a factor $\varphi(X, Y)$ into a factor $\psi(X)$ via marginalization:

$$\psi(X) = \sum_{y \in \text{Val}(Y)} \varphi(X, y)$$

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>$\varphi(A, B, C)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.9</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0.3</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1.1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1.7</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0.4</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0.7</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1.1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>$\psi(A, B)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1.2</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>2.8</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1.1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1.3</td>
</tr>
</tbody>
</table>

"summing out" $C$
Factor Marginalization

• Given \( X \) and \( Y \) (\( Y \notin X \)), we can turn a factor \( \varphi(X, Y) \) into a factor \( \psi(X) \) via marginalization:

\[
\psi(X) = \sum_{y \in \text{Val}(Y)} \varphi(X, y)
\]

• We can refer to this new factor by \( \sum_Y \varphi \).
Marginalizing Everything?

• Take a factor graph’s “everything factor” by multiplying all of its factors.
• Sum out all the variables (one by one).

• What do you get?
Factors Are Like Numbers

- Products are commutative: $\phi_1 \cdot \phi_2 = \phi_2 \cdot \phi_1$
- Products are associative:
  
  \[
  (\phi_1 \cdot \phi_2) \cdot \phi_3 = \phi_1 \cdot (\phi_2 \cdot \phi_3)
  \]
- Sums are commutative: $\sum_X \sum_Y \phi = \sum_Y \sum_X \phi$
  (max, too).
- Distributivity of multiplication over marginalization and maximization:

  \[
  X \notin \text{Scope}(\phi_1) \quad \Rightarrow \quad \sum_X (\phi_1 \cdot \phi_2) = \phi_1 \cdot \sum_X \phi_2
  \]

  \[
  \max_X (\phi_1 \cdot \phi_2) = \phi_1 \cdot \max_X \phi_2
  \]
Inference
Querying the Model

• Inference (e.g., do you have allergies or the flu?)

• What's the best explanation for your symptoms?

• Active data collection (what is the next best r.v. to observe?)
A Bigger Example: Your Car

• The car doesn't start.
• What do we conclude about the battery age?
• 18 random variables
• $2^{18}$ possible scenarios
Inference: An Ubiquitous Obstacle

• Decoding is inference (lecture 2).
• Learning is inference (lectures 3 and 4).

• Exact inference is #P-complete.
  – Even approximations within a given absolute or relative error are hard.
Probabilistic Inference Problems

Given values for some random variables \((X \subseteq V)\) ...

- **Most Probable Explanation**: what are the *most probable* values of the *rest* of the r.v.s \(V \setminus X\)?

(More generally ...)

- **Maximum A Posteriori (MAP)**: what are the most probable values of *some* other r.v.s, \(Y \subseteq (V \setminus X)\)?

- Random *sampling* from the posterior over values of \(Y\)
- Full *posterior* over values of \(Y\)
- *Marginal* probabilities from the posterior over \(Y\)

- **Minimum Bayes risk**: What is the \(Y\) with the lowest expected cost?
- **Cost-augmented decoding**: What is the most *dangerous* \(Y\)?
Approaches to Inference

- **Inference**
  - **Exact**
    - Variable elimination
    - Dynamic program'ing
    - ILP
    - MCMC
    - Gibbs
  - **Approximate**
    - Randomized
      - Importance sampling
      - Randomized search
      - Simulated annealing
      - Mean field
    - Variational
      - Loopy belief propagation
      - Mean field
    - Deterministic
      - LP relaxations
      - Dual decomp.
      - Beam search

**Hard inference methods; soft inference methods; methods for both**
Exact Marginal for Y

• This will be a generalization of algorithms you may already have seen: the *forward* and *backward* algorithms.

• The general name is *variable elimination*.

• After we see it for the marginal, we'll see how to use it for the MAP.
Inference Example

• Goal: \( P(D) \)
Inference Example

• Let’s calculate \( P(B) \) first.
Inference Example

- Let’s calculate $P(B)$ first.

$$P(B) = \sum_{a \in \text{Val}(A)} P(A = a)P(B \mid A = a)$$
Inference Example

• Let’s calculate $P(B)$ first.

$$P(B) = \sum_{a \in \text{Val}(A)} P(A = a)P(B \mid A = a)$$

• Note: C and D don’t matter.
Inference Example

- Let’s calculate $P(B)$ first.

$$P(B) = \sum_{a \in \text{Val}(A)} P(A = a) P(B \mid A = a)$$

<table>
<thead>
<tr>
<th>B</th>
<th>$P(B) = \varphi_B(B)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A</th>
<th>$P(A) = \varphi_A(A)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>$P(B \mid A) = \varphi_{AB}(A, B)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
Inference Example

• New model in which A is eliminated; defines $P(B, C, D)$
Inference Example

- Same thing to eliminate $B$.

\[ P(C) = \sum_{b \in \text{Val}(B)} P(B = b)P(C | B = b) \]
Inference Example

- New model in which B is eliminated; defines P(C, D)
Simple Inference Example

• Last step to get $P(D)$:

$$P(D) = \sum_{c \in \text{Val}(C)} P(C = c)P(D | C = c)$$

<table>
<thead>
<tr>
<th>D</th>
<th>$P(D) = \varphi_D(D)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C</th>
<th>P(C) = $\varphi_C(C)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

| C | D | P(D | C) = $\varphi_{CD}(C, D)$ |
|---|---|--------------------------------|
| 0 | 0 |                                |
| 0 | 1 |                                |
| 1 | 0 |                                |
| 1 | 1 |                                |
Simple Inference Example

• Notice that the same step happened for each random variable:
  – We created a new factor over the variable and its “successor”
  – We summed out (marginalized) the variable.

\[
P(D) = \sum_{a \in \text{Val}(A)} \sum_{b \in \text{Val}(B)} \sum_{c \in \text{Val}(C)} P(A = a)P(B = b | A = a)P(C = c | B = b)P(D | C = c)
\]

\[
= \sum_{c \in \text{Val}(C)} P(D | C = c) \sum_{b \in \text{Val}(B)} P(C = c | B = b) \sum_{a \in \text{Val}(A)} P(A = a)P(B = b | A = a)
\]
That Was Variable Elimination

• We reused computation from previous steps and avoided doing the same work more than once.
  — Dynamic programming à la forward algorithm!
• We exploited the graph structure (each subexpression only depends on a small number of variables).
• Exponential blowup avoided!
What Remains

• Variable elimination in general
• The maximization version (for MAP inference)
• A bit about approximate inference
Eliminating One Variable

Input: Set of factors $\Phi$, variable Z to eliminate
Output: new set of factors $\Psi$

1. Let $\Phi' = \{\varphi \in \Phi \mid Z \in \text{Scope}(\varphi)\}$
2. Let $\Psi = \{\varphi \in \Phi \mid Z \notin \text{Scope}(\varphi)\}$
3. Let $\psi$ be $\sum_Z \prod_{\varphi \in \Phi} \varphi$
4. Return $\Psi \cup \{\psi\}$
Example

• Query:
  \[ P(\text{Flu} \mid \text{runny nose}) \]

• Let's eliminate \( H \).
Example

• Query:
  \[ P(\text{Flu} \mid \text{runny nose}) \]

• Let's eliminate H.
  1. \( \Phi' = \{\phi_{SH}\} \)
  2. \( \Psi = \{\phi_F, \phi_A, \phi_{FAS}, \phi_{SR}\} \)
  3. \( \psi = \sum_H \prod_{\varphi \in \Phi'} \varphi \)
  4. Return \( \Psi \cup \{\psi\} \)
Example

• Query: 
  \( P(\text{Flu} \mid \text{runny nose}) \)

• Let's eliminate \( H \).
  1. \( \Phi' = \{ \varphi_{SH} \} \)
  2. \( \Psi = \{ \varphi_F, \varphi_A, \varphi_{FAS}, \varphi_{SR} \} \)
  3. \( \psi = \sum_H \varphi_{SH} \)
  4. Return \( \Psi \cup \{ \psi \} \)
Example

- Query:
  \( P(\text{Flu} \mid \text{runny nose}) \)

- Let's eliminate \( H \).
  1. \( \Phi' = \{ \phi_{SH} \} \)
  2. \( \Psi = \{ \phi_F, \phi_A, \phi_{FAS}, \phi_{SR} \} \)
  3. \( \psi = \sum_H \phi_{SH} \)
  4. Return \( \Psi \cup \{ \psi \} \)

<table>
<thead>
<tr>
<th>S</th>
<th>H</th>
<th>( \phi_{SH}(S, H) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0.8</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0.2</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0.1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0.9</td>
</tr>
</tbody>
</table>

\[
\begin{array}{c|c|c}
S & \psi(S) \\
0 & 1.0 \\
1 & 1.0 \\
\end{array}
\]
Example

• Query:
  P(Flu | runny nose)

• Let's eliminate H.
  1. $\Phi' = \{\varphi_{SH}\}$
  2. $\Psi = \{\varphi_F, \varphi_A, \varphi_{FAS}, \varphi_{SR}\}$
  3. $\psi = \sum_H \varphi_{SH}$
  4. Return $\Psi \cup \{\psi\}$

<table>
<thead>
<tr>
<th>S</th>
<th>H</th>
<th>$\varphi_{SH}(S, H)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0.8</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0.2</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0.1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0.9</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>S</th>
<th>$\psi(S)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.0</td>
</tr>
<tr>
<td>1</td>
<td>1.0</td>
</tr>
</tbody>
</table>
Example

• Query: 
  \( P(\text{Flu} \mid \text{runny nose}) \)

• Let's eliminate H.

• We can actually ignore the new factor, equivalently just deleting H!
  – Why?
  – In some cases eliminating a variable is really easy!

\[
\begin{array}{c|c|c}
S & \psi(S) \\
0 & 1.0 \\
1 & 1.0 \\
\end{array}
\]
Example

• Query: \( P(\text{Flu} \mid \text{runny nose}) \)

• H is already eliminated.

• Let's now eliminate S.
Example

• Query:
P(Flu | runny nose)

• Eliminating S.
  1. $\Phi' = \{\varphi_{SR}, \varphi_{FAS}\}$
  2. $\Psi = \{\varphi_{F}, \varphi_{A}\}$
  3. $\psi_{FAR} = \sum_S \prod_{\varphi \in \Phi'} \varphi$
  4. Return $\Psi \cup \{\psi_{FAR}\}$
Example

• Query:
P(Flu | runny nose)

• Eliminating S.
1. \( \Phi' = \{ \varphi_{SR}, \varphi_{FAS} \} \)
2. \( \Psi = \{ \varphi_{F}, \varphi_{A} \} \)
3. \( \psi_{FAR} = \sum_{S} \varphi_{SR} \cdot \varphi_{FAS} \)
4. Return \( \Psi \cup \{ \psi_{FAR} \} \)
Example

• Query:
  \[ P(\text{Flu} \mid \text{runny nose}) \]

• Eliminating S.
  1. \( \Phi' = \{ \varphi_{SR}, \varphi_{FAS} \} \)
  2. \( \Psi = \{ \varphi_F, \varphi_A \} \)
  3. \( \psi_{FAR} = \sum_S \varphi_{SR} \cdot \varphi_{FAS} \)
  4. Return \( \Psi \cup \{ \psi_{FAR} \} \)
Example

• Query:
  $P(\text{Flu} \mid \text{runny nose})$

• Finally, eliminate A.
Example

• **Query:**
  \[ P(\text{Flu} \mid \text{runny nose}) \]

• **Eliminating A.**
  1. \( \Phi' = \{ \varphi_A, \varphi_{\text{FAR}} \} \)
  2. \( \Psi = \{ \varphi_F \} \)
  3. \( \psi_{\text{FR}} = \sum_A \varphi_A \cdot \psi_{\text{FAR}} \)
  4. Return \( \Psi \cup \{ \psi_{\text{FR}} \} \)
Example

• Query:
  \[ P(\text{Flu} \mid \text{runny nose}) \]

• Eliminating A.
  1. \( \Phi' = \{ \varphi_A, \varphi_{\text{FAR}} \} \)
  2. \( \Psi = \{ \varphi_F \} \)
  3. \( \psi_{\text{FR}} = \sum_A \varphi_A \cdot \psi_{\text{FAR}} \)
  4. Return \( \Psi \cup \{ \psi_{\text{FR}} \} \)
Chain, Again

• Goal: $P(D)$
• Earlier, we eliminated A, then B, then C.
Chain, Again

- **Goal:** $P(D)$
- Earlier, we eliminated $A$, then $B$, then $C$.
- Let’s start with $C$. 

<table>
<thead>
<tr>
<th>$A$</th>
<th>$B$</th>
<th>$P(B \mid A) = \varphi_{AB}(A, B)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$B$</th>
<th>$C$</th>
<th>$P(C \mid B) = \varphi_{BC}(B, C)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$C$</th>
<th>$D$</th>
<th>$P(D \mid C) = \varphi_{CD}(C, D)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Chain, Again

• Goal: \( P(D) \)
• Earlier, we eliminated A, then B, then C.
• Let’s start with C.
Chain, Again

• Eliminating C.

| B | C | P(C | B) = ϕ_{BC}(B, C) |
|---|---|------------------------|
| 0 | 0 | 0                      |
| 0 | 1 | 0                      |
| 1 | 0 | 1                      |
| 1 | 1 | 1                      |

| C | D | P(D | C) = ϕ_{CD}(C, D) |
|---|---|------------------------|
| 0 | 0 | 0                      |
| 0 | 1 | 1                      |
| 1 | 0 | 1                      |
| 1 | 1 | 1                      |

<table>
<thead>
<tr>
<th>B</th>
<th>C</th>
<th>D</th>
<th>ϕ_{BCD}(B, C, D)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Chain, Again

- Eliminating C.

<table>
<thead>
<tr>
<th>B</th>
<th>C</th>
<th>D</th>
<th>$\varphi_{BCD}(B, C, D)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>B</th>
<th>D</th>
<th>$\psi(B, D)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Chain, Again

- Eliminating B will be similarly complex.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>$P(B \mid A) = \varphi_{AB}(A, B)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>B</th>
<th>D</th>
<th>$\psi(B, D)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A</th>
<th>$P(A) = \varphi_A(A)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
Variable Elimination: Comments

• Can prune away all non-ancestors of the query variables.
• Ordering makes a difference!
What about Evidence?

• So far, we've just considered the posterior/marginal $P(Y)$.

• Next: conditional distribution $P(Y \mid X = x)$.

• It's almost the same: the additional step is to reduce factors to respect the evidence.
Example

• Query: $P(\text{Flu} \mid \text{runny nose})$

• Let's reduce to $R = \text{true}$ (runny nose).

\[
\begin{array}{c|c|c}
S & R & \varphi_{SR} (S, R) \\
0 & 0 & 0 \\
0 & 1 & 0 \\
1 & 0 & 1 \\
1 & 1 & 1 \\
\end{array}
\]

\[
\begin{array}{c|c|c}
S & R \mid \varphi'_S (S) \\
0 & 1 & 1 \\
1 & 1 & 1 \\
\end{array}
\]
Example

• Query: 
  $P(\text{Flu} \mid \text{runny nose})$

• Let's reduce to $R = \text{true (runny nose)}$. 

\[
\begin{array}{c|c|c}
S & R & \varphi'_S (S) \\
\hline
0 & 1 & \\
1 & 1 & \\
\end{array}
\]
Example

• Query: 
  \( P(\text{Flu} \mid \text{runny nose}) \)

• Now run variable elimination all the way down to one factor (for F).
Example

• Query: 
  \( P(\text{Flu} \mid \text{runny nose}) \)

• Now run variable elimination all the way down to one factor (for F).
Example

• Query: $P(\text{Flu} \mid \text{runny nose})$

• Now run variable elimination all the way down to one factor (for F).

Eliminate A.
Example

- Query: \( P(\text{Flu} \mid \text{runny nose}) \)

- Now run variable elimination all the way down to one factor (for F).

Take final product.
Example

• Query:
  \( P(\text{Flu} \mid \text{runny nose}) \)

• Now run variable elimination all the way down to one factor.

\( \varphi_F \cdot \psi_F \)
Additional Comments

• Runtime depends on the size of the *intermediate* factors.
• Hence, variable elimination ordering matters a lot.
  – But it's NP-hard to find the best one.
  – For MNs, *chordal graphs* permit inference in time linear in the size of the original factors.
  – For BNs, *polytree* structures do the same.
• If you can avoid “big” intermediate factors, you can make inference linear in the size of the original factors.
Variable Elimination for Conditional Probabilities $P(Y \mid X = x)$

Input: Graphical model on $V$, set of query variables $Y$, evidence $X = x$

Output: factor $\varphi$ and scalar $\alpha$

1. $\Phi = \text{factors in the model}$
2. Reduce factors in $\Phi$ by $X = x$
3. Choose variable ordering $\pi$ on $Z = V \setminus Y \setminus X$
4. $\varphi = \text{Variable-Elimination}(\Phi, Z, \pi)$
5. $\alpha = \sum_{z \in \text{Val}(Z)} \varphi(z)$
6. Return $\varphi, \alpha$
Getting Back to NLP

• Traditional structured NLP models were sometimes chosen for these properties.
  – HMMs, PCFGs (with a little work)
  – But not: IBM model 3

• To decode, we need MAP inference for decoding!

• When models get complicated, need approximations!
From Marginals to MAP

• Replace factor marginalization steps with maximization.
  – Add bookkeeping to keep track of the maximizing values.

• Add a traceback at the end to recover the solution.

• This is analogous to the connection between the forward algorithm and the Viterbi algorithm.
  – Ordering challenge is the same.
Variable Elimination
(Max-Product Version with Decoding)

Input: Set of factors $\Phi$, ordered list of variables $Z$ to eliminate

Output: new factor

1. For each $Z_i \in Z$ (in order):
   - Let $(\Phi, \psi_{Z_i}) = \text{Eliminate-One}(\Phi, Z_i)$

2. Return $\prod_{\varphi \in \Phi} \varphi$, Traceback($\{\psi_{Z_i}\}$)
Eliminating One Variable
(Max-Product Version with Bookkeeping)

Input: Set of factors $\Phi$, variable $Z$ to eliminate
Output: new set of factors $\Psi$

1. Let $\Phi' = \{ \varphi \in \Phi \mid Z \in \text{Scope}(\varphi) \}$
2. Let $\Psi = \{ \varphi \in \Phi \mid Z \notin \text{Scope}(\varphi) \}$
3. Let $\tau$ be $\max_Z \prod_{\varphi \in \Phi'} \varphi$
   - Let $\psi$ be $\prod_{\varphi \in \Phi'} \varphi$ (bookkeeping)
4. Return $\Psi \cup \{ \tau \}, \psi$
Traceback

Input: Sequence of factors with associated variables: \((\psi_{Z_1}, ..., \psi_{Z_k})\)

Output: \(z^*\)

• Each \(\psi_Z\) is a factor with scope including \(Z\) and variables eliminated \textit{after} \(Z\).

• Work backwards from \(i = k\) to \(1\):
  – Let \(z_i = \text{arg max}_z \psi_{Z_i}(z, z_{i+1}, z_{i+2}, ..., z_k)\)

• Return \(z\)
About the Traceback

• No extra (asymptotic) expense.
  – Linear traversal over the intermediate factors.

• The factor operations for both sum-product VE and max-product VE can be generalized.
  – Example: get the K most likely assignments
Variable Elimination Tips

• Any ordering will be correct.
• Most orderings will be too expensive.
• There are heuristics for choosing an ordering.
  – If the graph is chain-like, work from one end toward the other.
(Rocket Science: True MAP)

- Evidence: $X = x$
- Query: $Y$
- Other variables: $Z = V \setminus X \setminus Y$

$$y^* = \arg \max_{y \in \text{Val}(Y)} P(Y = y \mid X = x)$$

$$= \arg \max_{y \in \text{Val}(Y)} \sum_{z \in \text{Val}(Z)} P(Y = y, Z = z \mid X = x)$$

- First, marginalize out $Z$, then do MAP inference over $Y$ given $X = x$

- This is not usually attempted in NLP, with some exceptions.
Parting Shots

• You will probably never implement the general variable elimination algorithm.
• You will rarely use exact inference.
• Understand the inference problem would look like in exact form; then approximate.
  – Sometimes you get lucky.
  – You’ll appreciate better approximations as they come along.