Language and Statistics II

Lecture 4: Log-Linear Models

(The Big Idea)

Noah Smith

Administrivia

- What lit review topics are we (you) thinking about?

(m+1)-gram Models, A Different Way

$$p\left(s_1^n\right) = \left(\prod_{i=1}^{n} \gamma\left(s_i \mid s_{i-m}^{i-1}\right)\right) \cdot \gamma\left(stop \mid s_{n-m+1}^{n}\right)$$

$$= \prod_{\langle \mathbf{s}_1,\ldots,\mathbf{s}_m,\mathbf{s}\rangle \in \left(\Sigma \cup \{\text{start},\text{stop}\}\right)^{m+1}} \gamma\left(\mathbf{s} \mid \mathbf{s}_1^m\right)^{\text{count}\left(\langle \mathbf{s}_1,\ldots,\mathbf{s}_m,\mathbf{s}\rangle ; s_1^n\right)}$$

$$= \exp \sum_{\langle \mathbf{s}_1,\ldots,\mathbf{s}_m,\mathbf{s}\rangle \in \left(\Sigma \cup \{\text{start},\text{stop}\}\right)^{m+1}} \text{count}\left(\langle \mathbf{s}_1,\ldots,\mathbf{s}_m,\mathbf{s}\rangle ; s_1^n\right) \cdot \log \gamma\left(\mathbf{s} \mid \mathbf{s}_1^m\right)$$

Log-Linear Models

$$p(\mathbf{x}) = \frac{\exp \vec{\theta} \cdot \vec{f}(\mathbf{x})}{Z(\vec{\theta})}$$

$$\log p(\mathbf{x}) \propto \vec{\theta} \cdot \vec{f}(\mathbf{x})$$

What are the features?

What are the parameters?

$$\exp \sum_{\langle \mathbf{s}_1, \ldots, \mathbf{s}_m, \mathbf{s} \rangle \in (\Sigma \cup \{\text{start}, \text{stop}\})^{m+1}} \text{count}\left(\langle \mathbf{s}_1, \ldots, \mathbf{s}_m, \mathbf{s} \rangle; s_1^n\right) \cdot \log \gamma\left(\mathbf{s} \mid \mathbf{s}_1^m\right)$$

Where is *Z*?

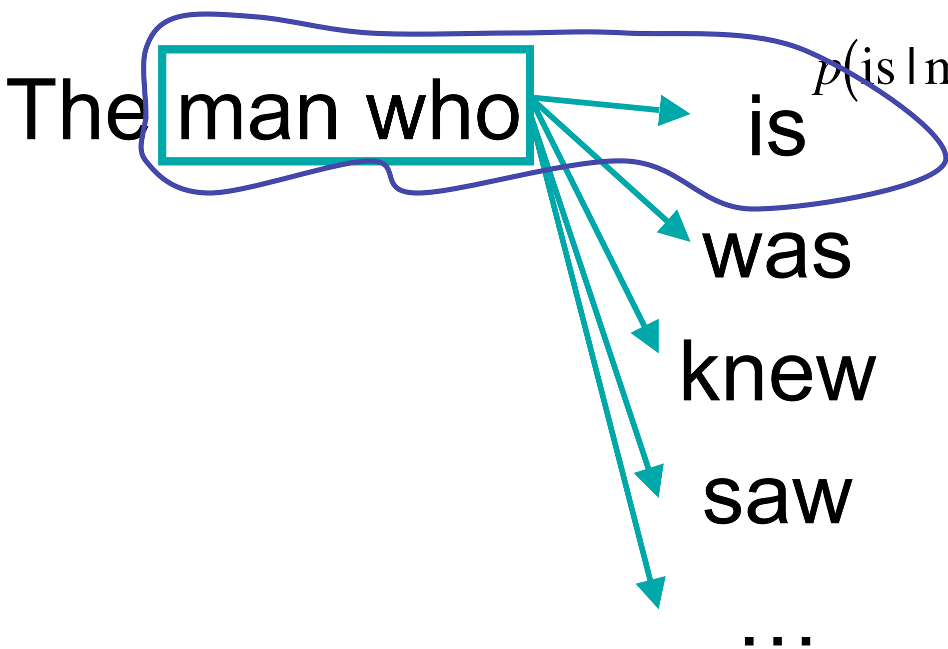
(m+1)-gram Model as a Log-Linear Model over the Next Word

$$p(s_1^n) = \prod_{i=1}^{n+1} p\left(s_i \mid s_{i-m}^{i-1}\right) = \prod_{i=1}^{n+1} \frac{\exp \vec{\theta} \cdot \vec{f}\left(s_{i-m}^{i}\right)}{\sum_{\mathbf{s} \in \Sigma} \exp \vec{\theta} \cdot \vec{f}\left(s_{i-m}^{i-1}, \mathbf{s}\right)}$$

$$f_{\vec{\mathbf{s}}}\left(\vec{s}\right) = \delta\left(\vec{\mathbf{s}}, \vec{s}\right)$$

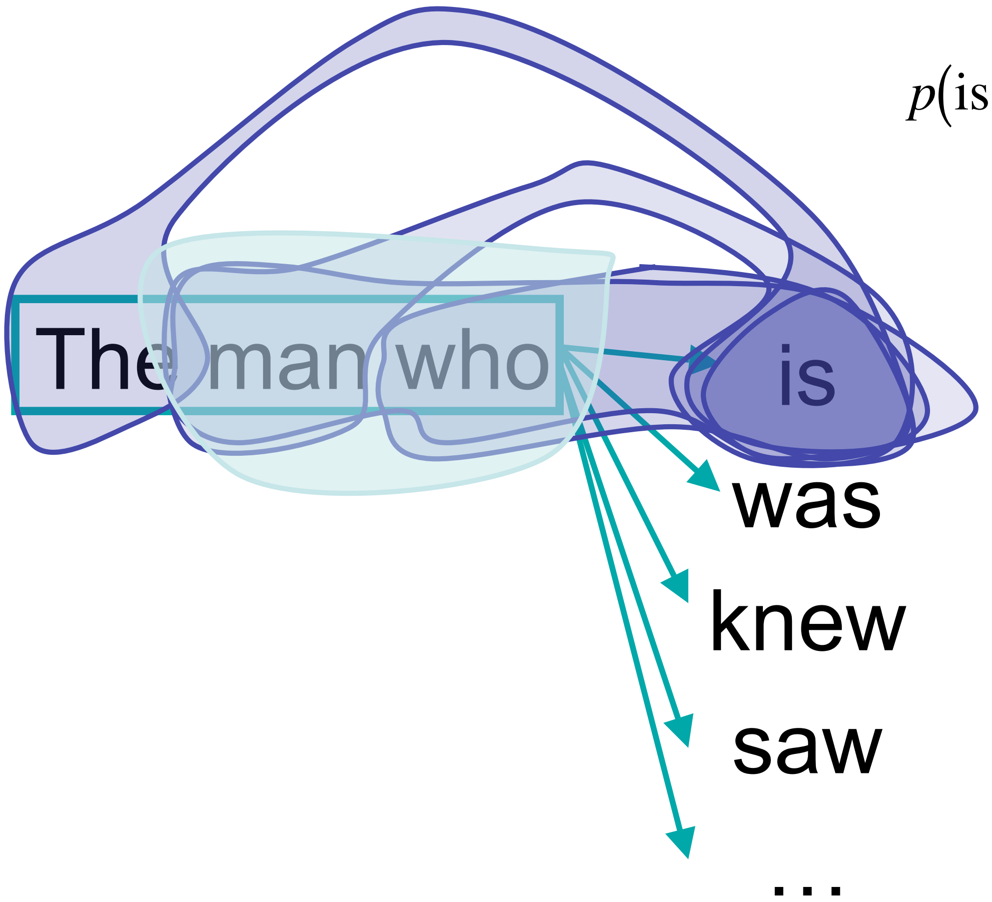
$$\theta_{\langle \mathbf{s}, \mathbf{s}_1, \ldots, \mathbf{s}_m \rangle} = \log \gamma\left(\mathbf{s} \mid \mathbf{s}_1, \ldots, \mathbf{s}_m\right)$$

Distributions over “the next word”



$$p(\text{is} \mid \text{man who}) = \frac{\exp \vec{f}(\text{man who is}) \cdot \vec{\theta}}{\sum_{w} \exp \vec{f}(\text{man who } w) \cdot \vec{\theta}}$$

Adding features



$$p(\text{is} \mid \text{history}) = \frac{\exp \vec{f}(\text{history is}) \cdot \vec{\theta}}{\sum_w \exp \vec{f}(\text{history } w) \cdot \vec{\theta}}$$

Watch Out!

- Nobody said nothin' 'bout "max ent" (yet)!
- We haven't talked about estimation (picking θ values from data).
- Don't worry about that yet.

Bait: Feature Brainstorm

Switch

How are we going to …

- Pick the best sequence in a (possibly weighted) lattice?
- Sum up over sequences (e.g., for minimum expected-loss decoding)?

- We need analogs to **Dijkstra**'s algorithm …
- (In fact, we will need such algorithms for *training*, too!)

What Makes Log-Linear Models Difficult

- *Z* (easy … so far!)
- Decoding with “big” or “long-distance” features
- Training is generally expensive

About the θs

- In a log-linear model, each θ can take any real value at all.

- $\theta_j < 0$: feature *j* gets penalized (event is less likely)
- $\theta_j = 0$: has no effect
- $\theta_j > 0$: feature *j* gets a bonus (event is more likely)

- $64,000 question: how do we pick the θs? (Next week)

Question

- Last week, we talked about choosing a path through an unweighted or weighted lattice, using an (*m*+1)-gram model.
- To do this, we used dynamic programming.
- What changes, if the model is log-linear (still based on (*m*+1)-grams) instead of a classical Markov model?

Claim

- HMMs are log-linear models, too.

$$p\left(c_1^n, s_1^n\right) = \left(\prod_{i=1}^{n} \eta\left(s_i \mid c_i\right) \cdot \gamma\left(c_i \mid c_{i-m}^{i-1}\right)\right) \cdot \gamma\left(\text{stop} \mid c_{n-m+1}^{n}\right)$$

$$p\left(c_1^n \mid s_1^n\right) = \frac{1}{p\left(s_1^n\right)}\left(\prod_{i=1}^{n} \eta\left(s_i \mid c_i\right) \cdot \gamma\left(c_i \mid c_{i-m}^{i-1}\right)\right) \cdot \gamma\left(\text{stop} \mid c_{n-m+1}^{n}\right)$$

$$= \frac{1}{Z\left(\vec{\theta}, s_1^n\right)} \prod_{i=1}^{n+1} \exp \vec{f}\left(c_1^i, s_1^n\right) \cdot \vec{\theta}$$

Log-linear HMMs

- In a standard trigram HMM, two feature **schemata** (or **templates**).

$$f_{\langle \mathbf{c},\mathbf{c}' \rangle \to \mathbf{c}''}\left(c_1^i, s_1^n\right) = \begin{cases} 1 \text{ if } c_{i-2} = \mathbf{c} \wedge c_{i-1} = \mathbf{c}' \wedge c_i = \mathbf{c}'' \\ 0 \text{ otherwise} \end{cases}$$

$$f_{\mathbf{c} \to \mathbf{s}}\left(c_1^i, s_1^n\right) = \begin{cases} 1 \text{ if } c_i = \mathbf{c} \wedge s_i = \mathbf{s} \\ 0 \text{ otherwise} \end{cases}$$

$$\vec{f}\left(c_1^i, s_1^n\right) = \vec{f}\left(c_{i-2}^i, s_i\right)$$

This fact succinctly encodes an independence assumption!

Ratnaparkhi (1996)

$$\vec{f}\left(c_1^i, s_1^n\right) = \vec{f}\left(c_{i-2}^i, s_{i-1}^{i+1}\right)$$

- Current word, current tag
- Tag trigram, tag bigram, tag unigram
- Current word prefix, current tag
- Current word suffix, current tag
- Previous word, current tag
- Next word, current tag

- Conjoined features: $f_i = f_j \wedge f_k$

Another point about Ratnaparkhi (1996)

- Orthogonal to the model: decoding
- Ratnaparkhi used a beam search
  - In principle, exact decoding is possible!
  - Why did he use a beam?

Food for Thought

- Where do the features come from?
  - Too many features: overfit
  - Too few good features: don't learn
  - Ratnaparkhi: cutoffs.
- Good models = good features + good weight training.
- Consider:
  - Every log-linear model on structures **x** actually includes **all** possible features of **x**.
  - Most of them have weight $\theta = 0$.

Global Log-Linear Models

- Instead of predicting “the next word” given the history …
- Build one big model that scores the whole sequence and normalizes *once*.

- Rosenfeld (1994) - language modeling
- Lafferty et al. (2001) - HMM-style models

(m+1)-gram Model as a Log-Linear Model over Sequences

$$p(s_1^n) = \frac{\exp \vec{\theta} \cdot \vec{f}\left(s_1^n\right)}{Z\left(\vec{\theta}\right)} = \frac{\exp \vec{\theta} \cdot \vec{f}\left(s_1^n\right)}{\sum_{\vec{s} \in \Sigma^*} \exp \vec{\theta} \cdot \vec{f}\left(\vec{s}\right)}$$

$$f_{\langle \mathbf{s}, \mathbf{s}_1, \ldots, \mathbf{s}_m \rangle}\left(s_1^n\right) = \mathrm{count}\left(\left\langle \mathbf{s}, \mathbf{s}_1, \ldots, \mathbf{s}_m \right\rangle; s_1^n\right)$$

$$\theta_{\langle \mathbf{s}, \mathbf{s}_1, \ldots, \mathbf{s}_m \rangle} = \log \gamma\left(\mathbf{s} \mid \mathbf{s}_1, \ldots, \mathbf{s}_m\right)$$

Summary & Ad

- Log-linear models:
  - Any features you want!
  - What the weights mean
  - The models we already know are examples
  - Some of the issues/choices/concerns
- Next time:
  - Training the weights (estimation)
  - Why they're called "max ent" models
  - Selecting the features