

Log-Linear Models

Michael Collins, Columbia University

The Language Modeling Problem

- ▶ w_i is the i 'th word in a document
- ▶ Estimate a distribution $p(w_i | w_1, w_2, \dots, w_{i-1})$ given previous “history” w_1, \dots, w_{i-1} .
- ▶ E.g., $w_1, \dots, w_{i-1} =$

Third, the notion “grammatical in English” cannot be identified in any way with the notion “high order of statistical approximation to English”. It is fair to assume that neither sentence (1) nor (2) (nor indeed any part of these sentences) has ever occurred in an English discourse. Hence, in any statistical

Trigram Models

- ▶ Estimate a distribution $p(w_i | w_1, w_2, \dots, w_{i-1})$ given previous “history” $w_1, \dots, w_{i-1} =$

Third, the notion “grammatical in English” cannot be identified in any way with the notion “high order of statistical approximation to English”. It is fair to assume that neither sentence (1) nor (2) (nor indeed any part of these sentences) has ever occurred in an English discourse. Hence, in any statistical

- ▶ **Trigram estimates:**

$$\begin{aligned} q(\text{model} | w_1, \dots, w_{i-1}) &= \lambda_1 q_{ML}(\text{model} | w_{i-2} = \text{any}, w_{i-1} = \text{statistical}) + \\ &\quad \lambda_2 q_{ML}(\text{model} | w_{i-1} = \text{statistical}) + \\ &\quad \lambda_3 q_{ML}(\text{model}) \end{aligned}$$

where $\lambda_i \geq 0$, $\sum_i \lambda_i = 1$, $q_{ML}(y|x) = \frac{\text{Count}(x,y)}{\text{Count}(x)}$

Trigram Models

$$q(\text{model} | w_1, \dots, w_{i-1}) = \lambda_1 q_{ML}(\text{model} | w_{i-2} = \text{any}, w_{i-1} = \text{statistical}) + \\ \lambda_2 q_{ML}(\text{model} | w_{i-1} = \text{statistical}) + \\ \lambda_3 q_{ML}(\text{model})$$

- ▶ Makes use of only bigram, trigram, unigram estimates
- ▶ Many other “features” of w_1, \dots, w_{i-1} may be useful, e.g.,:

$q_{ML}(\text{model} \mid w_{i-2} = \text{any})$

$q_{ML}(\text{model} \mid w_{i-1} \text{ is an adjective})$

$q_{ML}(\text{model} \mid w_{i-1} \text{ ends in “ical”})$

$q_{ML}(\text{model} \mid \text{author} = \text{Chomsky})$

$q_{ML}(\text{model} \mid \text{“model” does not occur somewhere in } w_1, \dots, w_{i-1})$

$q_{ML}(\text{model} \mid \text{“grammatical” occurs somewhere in } w_1, \dots, w_{i-1})$

A Naive Approach

$$\begin{aligned} q(\text{model} | w_1, \dots w_{i-1}) = & \\ & \lambda_1 q_{ML}(\text{model} | w_{i-2} = \text{any}, w_{i-1} = \text{statistical}) + \\ & \lambda_2 q_{ML}(\text{model} | w_{i-1} = \text{statistical}) + \\ & \lambda_3 q_{ML}(\text{model}) + \\ & \lambda_4 q_{ML}(\text{model} | w_{i-2} = \text{any}) + \\ & \lambda_5 q_{ML}(\text{model} | w_{i-1} \text{ is an adjective}) + \\ & \lambda_6 q_{ML}(\text{model} | w_{i-1} \text{ ends in "ical"}) + \\ & \lambda_7 q_{ML}(\text{model} | \text{author} = \text{Chomsky}) + \\ & \lambda_8 q_{ML}(\text{model} | \text{"model" does not occur somewhere in } w_1, \dots w_{i-1}) + \\ & \lambda_9 q_{ML}(\text{model} | \text{"grammatical" occurs somewhere in } w_1, \dots w_{i-1}) \end{aligned}$$

This quickly becomes very unwieldy...

A Second Example: Part-of-Speech Tagging

INPUT:

Profits soared at Boeing Co., easily topping forecasts on Wall Street, as their CEO Alan Mulally announced first quarter results.

OUTPUT:

Profits/**N** soared/**V** at/**P** Boeing/**N** Co./**N** ,/, easily/**ADV** topping/**V**
forecasts/**N** on/**P** Wall/**N** Street/**N** ,/, as/**P** their/**POSS** CEO/**N**
Alan/**N** Mulally/**N** announced/**V** first/**ADJ** quarter/**N** results/**N** ./.

N = Noun

V = Verb

P = Preposition

Adv = Adverb

Adj = Adjective

...

A Second Example: Part-of-Speech Tagging

Hispaniola/**NNP** quickly/**RB** became/**VB** an/**DT** important/**JJ**
base/**??** from which Spain expanded its empire into the rest of the
Western Hemisphere .

- There are many possible tags in the position **??**
{NN, NNS, Vt, Vi, IN, DT, ... }
- The task: model the distribution

$$p(t_i | t_1, \dots, t_{i-1}, w_1 \dots w_n)$$

where t_i is the i 'th tag in the sequence, w_i is the i 'th word

A Second Example: Part-of-Speech Tagging

Hispaniola/**NNP** quickly/**RB** became/**VB** an/**DT** important/**JJ**
base/**??** from which Spain expanded its empire into the rest of the
Western Hemisphere .

- The task: model the distribution

$$p(t_i | t_1, \dots, t_{i-1}, w_1 \dots w_n)$$

where t_i is the i 'th tag in the sequence, w_i is the i 'th word

- Again: many “features” of $t_1, \dots, t_{i-1}, w_1 \dots w_n$ may be relevant

$$q_{ML}(\text{NN} \mid w_i = \text{base})$$

$$q_{ML}(\text{NN} \mid t_{i-1} \text{ is JJ})$$

$$q_{ML}(\text{NN} \mid w_i \text{ ends in “e”})$$

$$q_{ML}(\text{NN} \mid w_i \text{ ends in “se”})$$

$$q_{ML}(\text{NN} \mid w_{i-1} \text{ is “important”})$$

$$q_{ML}(\text{NN} \mid w_{i+1} \text{ is “from”})$$

Overview

- ▶ Log-linear models
- ▶ Parameter estimation in log-linear models
- ▶ Smoothing/regularization in log-linear models

The General Problem

- ▶ We have some **input domain** \mathcal{X}
- ▶ Have a finite **label set** \mathcal{Y}
- ▶ Aim is to provide a **conditional probability** $p(y \mid x)$ for any x, y where $x \in \mathcal{X}$, $y \in \mathcal{Y}$

Language Modeling

- ▶ x is a “history” $w_1, w_2, \dots w_{i-1}$, e.g.,

Third, the notion “grammatical in English” cannot be identified in any way with the notion “high order of statistical approximation to English”. It is fair to assume that neither sentence (1) nor (2) (nor indeed any part of these sentences) has ever occurred in an English discourse. Hence, in any statistical

- ▶ y is an “outcome” w_i

Feature Vector Representations

- ▶ Aim is to provide a conditional probability $p(y \mid x)$ for “decision” y given “history” x
- ▶ A **feature** is a function $f_k(x, y) \in \mathbb{R}$
(Often **binary features** or **indicator functions** $f_k(x, y) \in \{0, 1\}$).
- ▶ Say we have m features f_k for $k = 1 \dots m$
 \Rightarrow A **feature vector** $f(x, y) \in \mathbb{R}^m$ for any x, y

Language Modeling

- ▶ x is a “history” w_1, w_2, \dots, w_{i-1} , e.g.,

Third, the notion “grammatical in English” cannot be identified in any way with the notion “high order of statistical approximation to English”.

It is fair to assume that neither sentence (1) nor (2) (nor indeed any part of these sentences) has ever occurred in an English discourse.

Hence, in any statistical

- ▶ y is an “outcome” w_i

- ▶ Example features:

$$f_1(x, y) = \begin{cases} 1 & \text{if } y = \text{model} \\ 0 & \text{otherwise} \end{cases}$$

$$f_2(x, y) = \begin{cases} 1 & \text{if } y = \text{model and } w_{i-1} = \text{statistical} \\ 0 & \text{otherwise} \end{cases}$$

$$f_3(x, y) = \begin{cases} 1 & \text{if } y = \text{model, } w_{i-2} = \text{any, } w_{i-1} = \text{statistical} \\ 0 & \text{otherwise} \end{cases}$$

$$f_4(x, y) = \begin{cases} 1 & \text{if } y = \text{model}, w_{i-2} = \text{any} \\ 0 & \text{otherwise} \end{cases}$$

$$f_5(x, y) = \begin{cases} 1 & \text{if } y = \text{model}, w_{i-1} \text{ is an adjective} \\ 0 & \text{otherwise} \end{cases}$$

$$f_6(x, y) = \begin{cases} 1 & \text{if } y = \text{model}, w_{i-1} \text{ ends in "ical"} \\ 0 & \text{otherwise} \end{cases}$$

$$f_7(x, y) = \begin{cases} 1 & \text{if } y = \text{model}, \text{author} = \text{Chomsky} \\ 0 & \text{otherwise} \end{cases}$$

$$f_8(x, y) = \begin{cases} 1 & \text{if } y = \text{model}, \text{"model"} \text{ is not in } w_1, \dots, w_{i-1} \\ 0 & \text{otherwise} \end{cases}$$

$$f_9(x, y) = \begin{cases} 1 & \text{if } y = \text{model}, \text{"grammatical"} \text{ is in } w_1, \dots, w_{i-1} \\ 0 & \text{otherwise} \end{cases}$$

Defining Features in Practice

- ▶ We had the following “trigram” feature:

$$f_3(x, y) = \begin{cases} 1 & \text{if } y = \text{model}, w_{i-2} = \text{any}, w_{i-1} = \text{statistical} \\ 0 & \text{otherwise} \end{cases}$$

- ▶ In practice, we would probably introduce one trigram feature for every trigram seen in the training data: i.e., for all trigrams (u, v, w) seen in training data, create a feature

$$f_{N(u,v,w)}(x, y) = \begin{cases} 1 & \text{if } y = w, w_{i-2} = u, w_{i-1} = v \\ 0 & \text{otherwise} \end{cases}$$

where $N(u, v, w)$ is a function that maps each (u, v, w) trigram to a different integer

The POS-Tagging Example

- ▶ Each x is a “history” of the form $\langle t_1, t_2, \dots, t_{i-1}, w_1 \dots w_n, i \rangle$
 - ▶ Each y is a POS tag, such as NN, NNS, Vt, Vi, IN, DT, ...
 - ▶ We have m features $f_k(x, y)$ for $k = 1 \dots m$
-

For example:

$$\begin{aligned} f_1(x, y) &= \begin{cases} 1 & \text{if current word } w_i \text{ is base and } y = \text{Vt} \\ 0 & \text{otherwise} \end{cases} \\ f_2(x, y) &= \begin{cases} 1 & \text{if current word } w_i \text{ ends in ing and } y = \text{VBG} \\ 0 & \text{otherwise} \end{cases} \\ &\dots \end{aligned}$$

The Full Set of Features in Ratnaparkhi, 1996

- ▶ Word/tag features for all word/tag pairs, e.g.,

$$f_{100}(x, y) = \begin{cases} 1 & \text{if current word } w_i \text{ is base and } y = \text{Vt} \\ 0 & \text{otherwise} \end{cases}$$

- ▶ Spelling features for all prefixes/suffixes of length ≤ 4 , e.g.,

$$f_{101}(x, y) = \begin{cases} 1 & \text{if current word } w_i \text{ ends in ing and } y = \text{VBG} \\ 0 & \text{otherwise} \end{cases}$$

$$f_{102}(h, t) = \begin{cases} 1 & \text{if current word } w_i \text{ starts with pre and } y = \text{NN} \\ 0 & \text{otherwise} \end{cases}$$

The Full Set of Features in Ratnaparkhi, 1996

- Contextual Features, e.g.,

$$f_{103}(x, y) = \begin{cases} 1 & \text{if } \langle t_{i-2}, t_{i-1}, y \rangle = \langle \text{DT}, \text{JJ}, \text{Vt} \rangle \\ 0 & \text{otherwise} \end{cases}$$

$$f_{104}(x, y) = \begin{cases} 1 & \text{if } \langle t_{i-1}, y \rangle = \langle \text{JJ}, \text{Vt} \rangle \\ 0 & \text{otherwise} \end{cases}$$

$$f_{105}(x, y) = \begin{cases} 1 & \text{if } \langle y \rangle = \langle \text{Vt} \rangle \\ 0 & \text{otherwise} \end{cases}$$

$$f_{106}(x, y) = \begin{cases} 1 & \text{if previous word } w_{i-1} = \textit{the} \text{ and } y = \text{Vt} \\ 0 & \text{otherwise} \end{cases}$$

$$f_{107}(x, y) = \begin{cases} 1 & \text{if next word } w_{i+1} = \textit{the} \text{ and } y = \text{Vt} \\ 0 & \text{otherwise} \end{cases}$$

The Final Result

- ▶ We can come up with practically any questions (*features*) regarding history/tag pairs.
- ▶ For a given history $x \in \mathcal{X}$, each label in \mathcal{Y} is mapped to a different feature vector

$$f(\langle \text{JJ, DT, } \langle \text{Hispaniola, } \dots \rangle, 6 \rangle, \text{Vt}) = 1001011001001100110$$

$$f(\langle \text{JJ, DT, } \langle \text{Hispaniola, } \dots \rangle, 6 \rangle, \text{JJ}) = 0110010101011110010$$

$$f(\langle \text{JJ, DT, } \langle \text{Hispaniola, } \dots \rangle, 6 \rangle, \text{NN}) = 0001111101001100100$$

$$f(\langle \text{JJ, DT, } \langle \text{Hispaniola, } \dots \rangle, 6 \rangle, \text{IN}) = 0001011011000000010$$

...

Parameter Vectors

- ▶ Given features $f_k(x, y)$ for $k = 1 \dots m$,
also define a **parameter vector** $v \in \mathbb{R}^m$
- ▶ Each (x, y) pair is then mapped to a “score”

$$v \cdot f(x, y) = \sum_k v_k f_k(x, y)$$

Language Modeling

- ▶ x is a “history” $w_1, w_2, \dots w_{i-1}$, e.g.,
Third, the notion “grammatical in English” cannot be identified in any way with the notion “high order of statistical approximation to English”. It is fair to assume that neither sentence (1) nor (2) (nor indeed any part of these sentences) has ever occurred in an English discourse. Hence, in any statistical
- ▶ Each possible y gets a different score:

$$\begin{array}{ll} v \cdot f(x, model) = 5.6 & v \cdot f(x, the) = -3.2 \\ v \cdot f(x, is) = 1.5 & v \cdot f(x, of) = 1.3 \\ v \cdot f(x, models) = 4.5 & \dots \end{array}$$

Log-Linear Models

- ▶ We have some input domain \mathcal{X} , and a finite label set \mathcal{Y} . Aim is to provide a conditional probability $p(y \mid x)$ for any $x \in \mathcal{X}$ and $y \in \mathcal{Y}$.
- ▶ A feature is a function $f : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$
(Often binary features or indicator functions $f_k : \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$).
- ▶ Say we have m features f_k for $k = 1 \dots m$
 \Rightarrow A feature vector $f(x, y) \in \mathbb{R}^m$ for any $x \in \mathcal{X}$ and $y \in \mathcal{Y}$.
- ▶ We also have a **parameter vector** $v \in \mathbb{R}^m$
- ▶ We define

$$p(y \mid x; v) = \frac{e^{v \cdot f(x, y)}}{\sum_{y' \in \mathcal{Y}} e^{v \cdot f(x, y')}}$$

Why the name?

$$\log p(y \mid x; v) = \underbrace{v \cdot f(x, y)}_{\text{Linear term}} - \underbrace{\log \sum_{y' \in \mathcal{Y}} e^{v \cdot f(x, y')}}_{\text{Normalization term}}$$

Overview

- ▶ Log-linear models
- ▶ Parameter estimation in log-linear models
- ▶ Smoothing/regularization in log-linear models

Maximum-Likelihood Estimation

- ▶ Maximum-likelihood estimates given training sample $(x^{(i)}, y^{(i)})$ for $i = 1 \dots n$, each $(x^{(i)}, y^{(i)}) \in \mathcal{X} \times \mathcal{Y}$:

$$v_{ML} = \operatorname{argmax}_{v \in \mathbb{R}^m} L(v)$$

where

$$L(v) = \sum_{i=1}^n \log p(y^{(i)} \mid x^{(i)}; v) = \sum_{i=1}^n v \cdot f(x^{(i)}, y^{(i)}) - \sum_{i=1}^n \log \sum_{y' \in \mathcal{Y}} e^{v \cdot f(x^{(i)}, y')}$$

Calculating the Maximum-Likelihood Estimates

- Need to maximize:

$$L(v) = \sum_{i=1}^n v \cdot f(x^{(i)}, y^{(i)}) - \sum_{i=1}^n \log \sum_{y' \in \mathcal{Y}} e^{v \cdot f(x^{(i)}, y')}$$

- Calculating gradients:

$$\begin{aligned} \frac{dL(v)}{dv_k} &= \sum_{i=1}^n f_k(x^{(i)}, y^{(i)}) - \sum_{i=1}^n \frac{\sum_{y' \in \mathcal{Y}} f_k(x^{(i)}, y') e^{v \cdot f(x^{(i)}, y')}}{\sum_{z' \in \mathcal{Y}} e^{v \cdot f(x^{(i)}, z')}} \\ &= \sum_{i=1}^n f_k(x^{(i)}, y^{(i)}) - \sum_{i=1}^n \sum_{y' \in \mathcal{Y}} f_k(x^{(i)}, y') \frac{e^{v \cdot f(x^{(i)}, y')}}{\sum_{z' \in \mathcal{Y}} e^{v \cdot f(x^{(i)}, z')}} \\ &= \underbrace{\sum_{i=1}^n f_k(x^{(i)}, y^{(i)})}_{\text{Empirical counts}} - \underbrace{\sum_{i=1}^n \sum_{y' \in \mathcal{Y}} f_k(x^{(i)}, y') p(y' \mid x^{(i)}; v)}_{\text{Expected counts}} \end{aligned}$$

Gradient Ascent Methods

- ▶ Need to maximize $L(v)$ where

$$\frac{dL(v)}{dv} = \sum_{i=1}^n f(x^{(i)}, y^{(i)}) - \sum_{i=1}^n \sum_{y' \in \mathcal{Y}} f(x^{(i)}, y') p(y' | x^{(i)}; v)$$

Initialization: $v = 0$

Iterate until convergence:

- ▶ Calculate $\Delta = \frac{dL(v)}{dv}$
- ▶ Calculate $\beta_* = \operatorname{argmax}_{\beta} L(v + \beta \Delta)$ (Line Search)
- ▶ Set $v \leftarrow v + \beta_* \Delta$

Conjugate Gradient Methods

- ▶ (Vanilla) gradient ascent can be very slow
- ▶ Conjugate gradient methods require calculation of gradient at each iteration, but do a line search in **a direction which is a function of the current gradient, and the previous step taken.**
- ▶ Conjugate gradient packages are widely available
In general: they require a function

$$\text{calc_gradient}(v) \rightarrow \left(L(v), \frac{dL(v)}{dv} \right)$$

and that's about it!

Overview

- ▶ Log-linear models
- ▶ Parameter estimation in log-linear models
- ▶ Smoothing/regularization in log-linear models

Smoothing in Log-Linear Models

- Say we have a feature:

$$f_{100}(x, y) = \begin{cases} 1 & \text{if current word } w_i \text{ is base and } y = \text{Vt} \\ 0 & \text{otherwise} \end{cases}$$

- In training data, base is seen 3 times, with Vt every time
- Maximum likelihood solution satisfies

$$\sum_i f_{100}(x^{(i)}, y^{(i)}) = \sum_i \sum_y p(y \mid x^{(i)}; v) f_{100}(x^{(i)}, y)$$

- $\Rightarrow p(\text{Vt} \mid x^{(i)}; v) = 1$ for any history $x^{(i)}$ where $w_i = \text{base}$
- $\Rightarrow v_{100} \rightarrow \infty$ at maximum-likelihood solution (most likely)
- $\Rightarrow p(\text{Vt} \mid x; v) = 1$ for any test data history x where $w = \text{base}$

Regularization

- Modified loss function

$$L(v) = \sum_{i=1}^n v \cdot f(x^{(i)}, y^{(i)}) - \sum_{i=1}^n \log \sum_{y' \in \mathcal{Y}} e^{v \cdot f(x^{(i)}, y')} - \frac{\lambda}{2} \sum_{k=1}^m v_k^2$$

- Calculating gradients:

$$\frac{dL(v)}{dv_k} = \underbrace{\sum_{i=1}^n f_k(x^{(i)}, y^{(i)})}_{\text{Empirical counts}} - \underbrace{\sum_{i=1}^n \sum_{y' \in \mathcal{Y}} f_k(x^{(i)}, y') p(y' | x^{(i)}; v)}_{\text{Expected counts}} - \lambda v_k$$

- Can run conjugate gradient methods as before
- Adds a penalty for large weights

Experiments with Regularization

- ▶ [Chen and Rosenfeld, 1998]: apply log-linear models to language modeling: Estimate $q(w_i \mid w_{i-2}, w_{i-1})$
- ▶ Unigram, bigram, trigram features, e.g.,

$$f_1(w_{i-2}, w_{i-1}, w_i) = \begin{cases} 1 & \text{if trigram is (the, dog, laughs)} \\ 0 & \text{otherwise} \end{cases}$$

$$f_2(w_{i-2}, w_{i-1}, w_i) = \begin{cases} 1 & \text{if bigram is (dog, laughs)} \\ 0 & \text{otherwise} \end{cases}$$

$$f_3(w_{i-2}, w_{i-1}, w_i) = \begin{cases} 1 & \text{if unigram is (laughs)} \\ 0 & \text{otherwise} \end{cases}$$

$$q(w_i \mid w_{i-2}, w_{i-1}) = \frac{e^{f(w_{i-2}, w_{i-1}, w_i) \cdot v}}{\sum_w e^{f(w_{i-2}, w_{i-1}, w) \cdot v}}$$

Experiments with Gaussian Priors

- ▶ In regular (unregularized) log-linear models, if all n-gram features are included, then it's equivalent to maximum-likelihood estimates!

$$q(w_i \mid w_{i-2}, w_{i-1}) = \frac{\text{Count}(w_{i-2}, w_{i-1}, w_i)}{\text{Count}(w_{i-2}, w_{i-1})}$$

- ▶ [Chen and Rosenfeld, 1998]: with regularization, get very good results. Performs as well as or better than standardly used “discounting methods” (see lecture 2).
- ▶ Downside: computing $\sum_w e^{f(w_{i-2}, w_{i-1}, w) \cdot v}$ is **SLOW**.