Language Modeling

Michael Collins, Columbia University

Overview

- The language modeling problem
- Trigram models
- Evaluating language models: perplexity
- Estimation techniques:
  - Linear interpolation
  - Discounting methods

The Language Modeling Problem

- We have some (finite) vocabulary,
  say $\mathcal{V} = \{$the, a, man, telescope, Beckham, two, $\ldots\}$

- We have an (infinite) set of strings, $\mathcal{V}^{\dagger}$

  the STOP
  a STOP
  the fan STOP
  the fan saw Beckham STOP
  the fan saw saw STOP
  the fan saw Beckham play for Real Madrid STOP

The Language Modeling Problem (Continued)

- We have a *training sample* of example sentences in English

The Language Modeling Problem (Continued)

- We have a *training sample* of example sentences in English
- We need to "learn" a probability distribution p i.e., p is a function that satisfies

$$\sum_{x \in \mathcal{V}^\dagger} p(x) = 1, \quad p(x) \geq 0 \text{ for all } x \in \mathcal{V}^\dagger$$

The Language Modeling Problem (Continued)

- We have a *training sample* of example sentences in English
- We need to "learn" a probability distribution p i.e., p is a function that satisfies

$$\sum_{x \in \mathcal{V}^\dagger} p(x) = 1, \quad p(x) \geq 0 \text{ for all } x \in \mathcal{V}^\dagger$$

$p(\text{the STOP}) = 10^{-12}$
$p(\text{the fan STOP}) = 10^{-8}$
$p(\text{the fan saw Beckham STOP}) = 2 \times 10^{-8}$
$p(\text{the fan saw saw STOP}) = 10^{-15}$
$\ldots$
$p(\text{the fan saw Beckham play for Real Madrid STOP}) = 2 \times 10^{-9}$
$\ldots$

Why on earth would we want to do this?!

- **Speech recognition** was the original motivation.
  (Related problems are optical character recognition, handwriting recognition.)

Why on earth would we want to do this?!

- **Speech recognition** was the original motivation. (Related problems are optical character recognition, handwriting recognition.)

- The estimation techniques developed for this problem will be **VERY** useful for other problems in NLP

A Naive Method

- We have N training sentences
- For any sentence $x_1 \ldots x_n$, $c(x_1 \ldots x_n)$ is the number of times the sentence is seen in our training data
- A naive estimate:

$$p(x_1 \ldots x_n) = \frac{c(x_1 \ldots x_n)}{N}$$

Overview

- The language modeling problem
- Trigram models
- Evaluating language models: perplexity
- Estimation techniques:
  - Linear interpolation
  - Discounting methods

Markov Processes

- Consider a sequence of random variables $X_1, X_2, \ldots X_n$. Each random variable can take any value in a finite set $\mathcal{V}$. For now we assume the length n is fixed (e.g., $n = 100$).
- Our goal: model

$$P(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n)$$

First-Order Markov Processes

$$P(X_1 = x_1, X_2 = x_2, \ldots X_n = x_n)$$

First-Order Markov Processes

$$
\begin{aligned}
& P(X_1 = x_1, X_2 = x_2, \ldots X_n = x_n) \\
= \; & P(X_1 = x_1) \prod_{i=2}^{n} P(X_i = x_i | X_1 = x_1, \ldots, X_{i-1} = x_{i-1})
\end{aligned}
$$

First-Order Markov Processes

$$
\begin{aligned}
& P(X_1 = x_1, X_2 = x_2, \ldots X_n = x_n) \\
= \;& P(X_1 = x_1) \prod_{i=2}^{n} P(X_i = x_i | X_1 = x_1, \ldots, X_{i-1} = x_{i-1}) \\
= \;& P(X_1 = x_1) \prod_{i=2}^{n} P(X_i = x_i | X_{i-1} = x_{i-1})
\end{aligned}
$$

First-Order Markov Processes

$$
\begin{aligned}
& P(X_1 = x_1, X_2 = x_2, \ldots X_n = x_n) \\
= \; & P(X_1 = x_1) \prod_{i=2}^{n} P(X_i = x_i | X_1 = x_1, \ldots, X_{i-1} = x_{i-1}) \\
= \; & P(X_1 = x_1) \prod_{i=2}^{n} P(X_i = x_i | X_{i-1} = x_{i-1})
\end{aligned}
$$

The first-order Markov assumption: For any $i \in \{2 \ldots n\}$, for any $x_1 \ldots x_i$,

$$P(X_i = x_i | X_1 = x_1 \ldots X_{i-1} = x_{i-1}) = P(X_i = x_i | X_{i-1} = x_{i-1})$$

Second-Order Markov Processes

$$P(X_1 = x_1, X_2 = x_2, \ldots X_n = x_n)$$

Second-Order Markov Processes

$$
\begin{aligned}
& P(X_1 = x_1, X_2 = x_2, \ldots X_n = x_n) \\
= \quad & P(X_1 = x_1) \times P(X_2 = x_2 | X_1 = x_1) \\
& \times \prod_{i=3}^{n} P(X_i = x_i | X_{i-2} = x_{i-2}, X_{i-1} = x_{i-1})
\end{aligned}
$$

Second-Order Markov Processes

$$
\begin{aligned}
& P(X_1 = x_1, X_2 = x_2, \ldots X_n = x_n) \\
= \; & P(X_1 = x_1) \times P(X_2 = x_2 | X_1 = x_1) \\
& \times \prod_{i=3}^{n} P(X_i = x_i | X_{i-2} = x_{i-2}, X_{i-1} = x_{i-1}) \\
= \; & \prod_{i=1}^{n} P(X_i = x_i | X_{i-2} = x_{i-2}, X_{i-1} = x_{i-1})
\end{aligned}
$$

(For convenience we assume $x_0 = x_{-1} = *$, where * is a special "start" symbol.)

Modeling Variable Length Sequences

- We would like the length of the sequence, n, to also be a random variable
- A simple solution: always define $X_n =$ STOP where STOP is a special symbol

Modeling Variable Length Sequences

- We would like the length of the sequence, n, to also be a random variable
- A simple solution: always define $X_n = \text{STOP}$ where STOP is a special symbol
- Then use a Markov process as before:

$$
\begin{aligned}
& P(X_1 = x_1, X_2 = x_2, \ldots X_n = x_n) \\
= & \prod_{i=1}^{n} P(X_i = x_i | X_{i-2} = x_{i-2}, X_{i-1} = x_{i-1})
\end{aligned}
$$

(For convenience we assume $x_0 = x_{-1} = *$, where * is a special "start" symbol.)

Trigram Language Models

- A trigram language model consists of:
  1. A finite set $\mathcal{V}$
  2. A parameter $q(w|u,v)$ for each trigram u, v, w such that $w \in \mathcal{V} \cup \{\text{STOP}\}$, and $u, v \in \mathcal{V} \cup \{\text{*}\}$.

Trigram Language Models

- A trigram language model consists of:
  1. A finite set $\mathcal{V}$
  2. A parameter $q(w|u, v)$ for each trigram u, v, w such that $w \in \mathcal{V} \cup \{\text{STOP}\}$, and $u, v \in \mathcal{V} \cup \{*\}$.

- For any sentence $x_1 \ldots x_n$ where $x_i \in \mathcal{V}$ for $i = 1 \ldots (n-1)$, and $x_n = \text{STOP}$, the probability of the sentence under the trigram language model is

$$p(x_1 \ldots x_n) = \prod_{i=1}^{n} q(x_i | x_{i-2}, x_{i-1})$$

where we define $x_0 = x_{-1} = *$.

An Example

For the sentence

the dog barks STOP

we would have

$$\begin{aligned} p(\text{the dog barks STOP}) &= q(\text{the}|\text{*, *}) \\ &\quad \times q(\text{dog}|\text{*, the}) \\ &\quad \times q(\text{barks}|\text{the, dog}) \\ &\quad \times q(\text{STOP}|\text{dog, barks}) \end{aligned}$$

The Trigram Estimation Problem

Remaining estimation problem:

$$q(w_i \mid w_{i-2}, w_{i-1})$$

For example:

$$q(\text{laughs} \mid \text{the, dog})$$

The Trigram Estimation Problem

Remaining estimation problem:

$$q(w_i \mid w_{i-2}, w_{i-1})$$

For example:

$$q(\text{laughs} \mid \text{the, dog})$$

A natural estimate (the "maximum likelihood estimate"):

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

$$q(\text{laughs} \mid \text{the, dog}) = \frac{\text{Count(the, dog, laughs)}}{\text{Count(the, dog)}}$$

Sparse Data Problems

A natural estimate (the "maximum likelihood estimate"):

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

$$q(\text{laughs} \mid \text{the, dog}) = \frac{\text{Count(the, dog, laughs)}}{\text{Count(the, dog)}}$$

Say our vocabulary size is $N = |\mathcal{V}|$, then there are N^3 parameters in the model.

e.g., $N = 20,000 \quad \Rightarrow \quad 20,000^3 = 8 \times 10^{12}$ parameters

Overview

- The language modeling problem
- Trigram models
- Evaluating language models: perplexity
- Estimation techniques:
  - Linear interpolation
  - Discounting methods

Evaluating a Language Model: Perplexity

- We have some test data, m sentences

$$s_1, s_2, s_3, \ldots, s_m$$

Evaluating a Language Model: Perplexity

- We have some test data, m sentences

$$s_1, s_2, s_3, \ldots, s_m$$

- We could look at the probability under our model $\prod_{i=1}^{m} p(s_i)$. Or more conveniently, the *log probability*

$$\log \prod_{i=1}^{m} p(s_i) = \sum_{i=1}^{m} \log p(s_i)$$

Evaluating a Language Model: Perplexity

- We have some test data, m sentences

$$s_1, s_2, s_3, \ldots, s_m$$

- We could look at the probability under our model $\prod_{i=1}^{m} p(s_i)$. Or more conveniently, the *log probability*

$$\log \prod_{i=1}^{m} p(s_i) = \sum_{i=1}^{m} \log p(s_i)$$

- In fact the usual evaluation measure is *perplexity*

$$\text{Perplexity} = 2^{-l} \quad \text{where} \quad l = \frac{1}{M} \sum_{i=1}^{m} \log p(s_i)$$

and M is the total number of words in the test data.

Some Intuition about Perplexity

- Say we have a vocabulary $\mathcal{V}$, and $N = |\mathcal{V}| + 1$ and model that predicts

$$q(w|u, v) = \frac{1}{N}$$

  for all $w \in \mathcal{V} \cup \{\text{STOP}\}$, for all $u, v \in \mathcal{V} \cup \{*\}$.

- Easy to calculate the perplexity in this case:

$$\text{Perplexity} = 2^{-l} \quad \text{where} \quad l = \log \frac{1}{N}$$

  $\Rightarrow$

$$\text{Perplexity} = N$$

Perplexity is a measure of effective "branching factor"

Typical Values of Perplexity

- Results from Goodman ("A bit of progress in language modeling"), where $|\mathcal{V}| = 50,000$
- A trigram model: $p(x_1 \ldots x_n) = \prod_{i=1}^{n} q(x_i | x_{i-2}, x_{i-1})$. Perplexity $= 74$

Typical Values of Perplexity

- Results from Goodman ("A bit of progress in language modeling"), where $|\mathcal{V}| = 50,000$
- A trigram model: $p(x_1 \ldots x_n) = \prod_{i=1}^{n} q(x_i | x_{i-2}, x_{i-1})$. Perplexity $= 74$
- A bigram model: $p(x_1 \ldots x_n) = \prod_{i=1}^{n} q(x_i | x_{i-1})$. Perplexity $= 137$

Typical Values of Perplexity

- Results from Goodman ("A bit of progress in language modeling"), where $|\mathcal{V}| = 50,000$
- A trigram model: $p(x_1 \ldots x_n) = \prod_{i=1}^{n} q(x_i | x_{i-2}, x_{i-1})$. Perplexity $= 74$
- A bigram model: $p(x_1 \ldots x_n) = \prod_{i=1}^{n} q(x_i | x_{i-1})$. Perplexity $= 137$
- A unigram model: $p(x_1 \ldots x_n) = \prod_{i=1}^{n} q(x_i)$. Perplexity $= 955$

Some History

- Shannon conducted experiments on entropy of English i.e., how good are people at the perplexity game?

  C. Shannon. Prediction and entropy of printed English. Bell Systems Technical Journal, 30:50–64, 1951.

Some History

Chomsky (in *Syntactic Structures* (1957)):

> *Second, the notion "grammatical" cannot be identified with "meaningful" or "significant" in any semantic sense. Sentences (1) and (2) are equally nonsensical, but any speaker of English will recognize that only the former is grammatical.*
>
> *(1) Colorless green ideas sleep furiously.*
>
> *(2) Furiously sleep ideas green colorless.*
>
> *...*
>
> *... 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 model for grammaticalness, these sentences will be ruled out on identical grounds as equally 'remote' from English. Yet (1), though nonsensical, is grammatical, while (2) is not. ...*

Overview

- The language modeling problem
- Trigram models
- Evaluating language models: perplexity
- Estimation techniques:
  - Linear interpolation
  - Discounting methods

Sparse Data Problems

A natural estimate (the "maximum likelihood estimate"):

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

$$q(\text{laughs} \mid \text{the, dog}) = \frac{\text{Count(the, dog, laughs)}}{\text{Count(the, dog)}}$$

Say our vocabulary size is $N = |\mathcal{V}|$, then there are N^3 parameters in the model.

e.g., $N = 20,000 \quad \Rightarrow \quad 20,000^3 = 8 \times 10^{12}$ parameters

The Bias-Variance Trade-Off

- Trigram maximum-likelihood estimate

$$q_{\mathsf{ML}}(w_i \mid w_{i-2}, w_{i-1}) = \frac{\mathsf{Count}(w_{i-2}, w_{i-1}, w_i)}{\mathsf{Count}(w_{i-2}, w_{i-1})}$$

- Bigram maximum-likelihood estimate

$$q_{\mathsf{ML}}(w_i \mid w_{i-1}) = \frac{\mathsf{Count}(w_{i-1}, w_i)}{\mathsf{Count}(w_{i-1})}$$

- Unigram maximum-likelihood estimate

$$q_{\mathsf{ML}}(w_i) = \frac{\mathsf{Count}(w_i)}{\mathsf{Count}()}$$

Linear Interpolation

- Take our estimate $q(w_i \mid w_{i-2}, w_{i-1})$ to be

$$
\begin{aligned}
q(w_i \mid w_{i-2}, w_{i-1}) = \quad & \lambda_1 \times q_{\mathsf{ML}}(w_i \mid w_{i-2}, w_{i-1}) \\
& +\lambda_2 \times q_{\mathsf{ML}}(w_i \mid w_{i-1}) \\
& +\lambda_3 \times q_{\mathsf{ML}}(w_i)
\end{aligned}
$$

  where $\lambda_1 + \lambda_2 + \lambda_3 = 1$, and $\lambda_i \geq 0$ for all i.

Linear Interpolation (continued)

Our estimate correctly defines a distribution (define $\mathcal{V}' = \mathcal{V} \cup \{\mathsf{STOP}\}$):

$\sum_{w \in \mathcal{V}'} q(w \mid u, v)$

Linear Interpolation (continued)

Our estimate correctly defines a distribution (define $\mathcal{V}' = \mathcal{V} \cup \{\mathsf{STOP}\}$):

$$\sum_{w \in \mathcal{V}'} q(w \mid u, v)$$

$$= \sum_{w \in \mathcal{V}'} \left[\lambda_1 \times q_{\mathsf{ML}}(w \mid u, v) + \lambda_2 \times q_{\mathsf{ML}}(w \mid v) + \lambda_3 \times q_{\mathsf{ML}}(w)\right]$$

Linear Interpolation (continued)

Our estimate correctly defines a distribution (define $\mathcal{V}' = \mathcal{V} \cup \{\mathsf{STOP}\}$):

$\sum_{w \in \mathcal{V}'} q(w \mid u, v)$

$= \sum_{w \in \mathcal{V}'} [\lambda_1 \times q_{\mathsf{ML}}(w \mid u, v) + \lambda_2 \times q_{\mathsf{ML}}(w \mid v) + \lambda_3 \times q_{\mathsf{ML}}(w)]$

$= \lambda_1 \sum_w q_{\mathsf{ML}}(w \mid u, v) + \lambda_2 \sum_w q_{\mathsf{ML}}(w \mid v) + \lambda_3 \sum_w q_{\mathsf{ML}}(w)$

Linear Interpolation (continued)

Our estimate correctly defines a distribution (define $\mathcal{V}' = \mathcal{V} \cup \{\mathsf{STOP}\}$):

$$\sum_{w \in \mathcal{V}'} q(w \mid u, v)$$

$$= \sum_{w \in \mathcal{V}'} [\lambda_1 \times q_{\mathsf{ML}}(w \mid u, v) + \lambda_2 \times q_{\mathsf{ML}}(w \mid v) + \lambda_3 \times q_{\mathsf{ML}}(w)]$$

$$= \lambda_1 \sum_w q_{\mathsf{ML}}(w \mid u, v) + \lambda_2 \sum_w q_{\mathsf{ML}}(w \mid v) + \lambda_3 \sum_w q_{\mathsf{ML}}(w)$$

$$= \lambda_1 + \lambda_2 + \lambda_3$$

Linear Interpolation (continued)

Our estimate correctly defines a distribution (define $\mathcal{V}' = \mathcal{V} \cup \{\mathsf{STOP}\}$):

$$\sum_{w \in \mathcal{V}'} q(w \mid u, v)$$

$$= \sum_{w \in \mathcal{V}'} [\lambda_1 \times q_{\mathsf{ML}}(w \mid u, v) + \lambda_2 \times q_{\mathsf{ML}}(w \mid v) + \lambda_3 \times q_{\mathsf{ML}}(w)]$$

$$= \lambda_1 \sum_w q_{\mathsf{ML}}(w \mid u, v) + \lambda_2 \sum_w q_{\mathsf{ML}}(w \mid v) + \lambda_3 \sum_w q_{\mathsf{ML}}(w)$$

$$= \lambda_1 + \lambda_2 + \lambda_3$$

$$= 1$$

Linear Interpolation (continued)

Our estimate correctly defines a distribution (define $\mathcal{V}' = \mathcal{V} \cup \{\mathsf{STOP}\}$):

$\sum_{w \in \mathcal{V}'} q(w \mid u, v)$

$= \sum_{w \in \mathcal{V}'} [\lambda_1 \times q_{\mathsf{ML}}(w \mid u, v) + \lambda_2 \times q_{\mathsf{ML}}(w \mid v) + \lambda_3 \times q_{\mathsf{ML}}(w)]$

$= \lambda_1 \sum_w q_{\mathsf{ML}}(w \mid u, v) + \lambda_2 \sum_w q_{\mathsf{ML}}(w \mid v) + \lambda_3 \sum_w q_{\mathsf{ML}}(w)$

$= \lambda_1 + \lambda_2 + \lambda_3$

$= 1$

(Can show also that $q(w \mid u, v) \geq 0$ for all $w \in \mathcal{V}'$)

How to estimate the λ values?

- Hold out part of training set as "validation" data

How to estimate the λ values?

- Hold out part of training set as "validation" data
- Define $c'(w_1, w_2, w_3)$ to be the number of times the trigram (w_1, w_2, w_3) is seen in validation set

How to estimate the λ values?

- Hold out part of training set as "validation" data
- Define $c'(w_1, w_2, w_3)$ to be the number of times the trigram (w_1, w_2, w_3) is seen in validation set
- Choose $\lambda_1, \lambda_2, \lambda_3$ to maximize:

$$L(\lambda_1, \lambda_2, \lambda_3) = \sum_{w_1, w_2, w_3} c'(w_1, w_2, w_3) \log q(w_3 \mid w_1, w_2)$$

such that $\lambda_1 + \lambda_2 + \lambda_3 = 1$, and $\lambda_i \geq 0$ for all i, and where

$$\begin{aligned} q(w_i \mid w_{i-2}, w_{i-1}) = \quad & \lambda_1 \times q_{\mathsf{ML}}(w_i \mid w_{i-2}, w_{i-1}) \\ & + \lambda_2 \times q_{\mathsf{ML}}(w_i \mid w_{i-1}) \\ & + \lambda_3 \times q_{\mathsf{ML}}(w_i) \end{aligned}$$

Allowing the λ's to vary

- Take a function Π that partitions histories
  e.g.,

$$\Pi(w_{i-2}, w_{i-1}) = \begin{cases} 1 & \text{If Count}(w_{i-1}, w_{i-2}) = 0 \\ 2 & \text{If } 1 \leq \text{Count}(w_{i-1}, w_{i-2}) \leq 2 \\ 3 & \text{If } 3 \leq \text{Count}(w_{i-1}, w_{i-2}) \leq 5 \\ 4 & \text{Otherwise} \end{cases}$$

- Introduce a dependence of the λ's on the partition:

$$\begin{aligned} q(w_i \mid w_{i-2}, w_{i-1}) = \quad & \lambda_1^{\Pi(w_{i-2}, w_{i-1})} \times q_{\mathsf{ML}}(w_i \mid w_{i-2}, w_{i-1}) \\ & + \lambda_2^{\Pi(w_{i-2}, w_{i-1})} \times q_{\mathsf{ML}}(w_i \mid w_{i-1}) \\ & + \lambda_3^{\Pi(w_{i-2}, w_{i-1})} \times q_{\mathsf{ML}}(w_i) \end{aligned}$$

  where $\lambda_1^{\Pi(w_{i-2}, w_{i-1})} + \lambda_2^{\Pi(w_{i-2}, w_{i-1})} + \lambda_3^{\Pi(w_{i-2}, w_{i-1})} = 1$,
  and $\lambda_i^{\Pi(w_{i-2}, w_{i-1})} \geq 0$ for all i.

Overview

- The language modeling problem
- Trigram models
- Evaluating language models: perplexity
- Estimation techniques:
  - Linear interpolation
  - Discounting methods

Discounting Methods

- Say we've seen the following counts:

x	Count(x)	$q_{\mathsf{ML}}(w_i \mid w_{i-1})$
the	48	
the, dog	15	15/48
the, woman	11	11/48
the, man	10	10/48
the, park	5	5/48
the, job	2	2/48
the, telescope	1	1/48
the, manual	1	1/48
the, afternoon	1	1/48
the, country	1	1/48
the, street	1	1/48

- The maximum-likelihood estimates are high (particularly for low count items)

Discounting Methods

- Now define "discounted" counts, $\text{Count}^*(x) = \text{Count}(x) - 0.5$
- New estimates:

x	$\text{Count}(x)$	$\text{Count}^*(x)$	$\frac{\text{Count}^*(x)}{\text{Count}(\text{the})}$
the	48		
the, dog	15	14.5	14.5/48
the, woman	11	10.5	10.5/48
the, man	10	9.5	9.5/48
the, park	5	4.5	4.5/48
the, job	2	1.5	1.5/48
the, telescope	1	0.5	0.5/48
the, manual	1	0.5	0.5/48
the, afternoon	1	0.5	0.5/48
the, country	1	0.5	0.5/48
the, street	1	0.5	0.5/48

Discounting Methods (Continued)

- We now have some "missing probability mass":

$$\alpha(w_{i-1}) = 1 - \sum_{w} \frac{\text{Count}^*(w_{i-1}, w)}{\text{Count}(w_{i-1})}$$

e.g., in our example, $\alpha(the) = 10 \times 0.5/48 = 5/48$

Katz Back-Off Models (Bigrams)

- For a bigram model, define two sets

$$\begin{aligned} \mathcal{A}(w_{i-1}) &= \{w : \text{Count}(w_{i-1}, w) > 0\} \\ \mathcal{B}(w_{i-1}) &= \{w : \text{Count}(w_{i-1}, w) = 0\} \end{aligned}$$

- A bigram model

$$q_{BO}(w_i \mid w_{i-1}) = \begin{cases} \frac{\text{Count}^*(w_{i-1}, w_i)}{\text{Count}(w_{i-1})} & \text{If } w_i \in \mathcal{A}(w_{i-1}) \\ \alpha(w_{i-1}) \frac{q_{\text{ML}}(w_i)}{\sum_{w \in \mathcal{B}(w_{i-1})} q_{\text{ML}}(w)} & \text{If } w_i \in \mathcal{B}(w_{i-1}) \end{cases}$$

where

$$\alpha(w_{i-1}) = 1 - \sum_{w \in \mathcal{A}(w_{i-1})} \frac{\text{Count}^*(w_{i-1}, w)}{\text{Count}(w_{i-1})}$$

Katz Back-Off Models (Trigrams)

- For a trigram model, first define two sets

$$\begin{aligned}\mathcal{A}(w_{i-2}, w_{i-1}) &= \{w : \text{Count}(w_{i-2}, w_{i-1}, w) > 0\} \\ \mathcal{B}(w_{i-2}, w_{i-1}) &= \{w : \text{Count}(w_{i-2}, w_{i-1}, w) = 0\}\end{aligned}$$

- A trigram model is defined in terms of the bigram model:

$$q_{BO}(w_i \mid w_{i-2}, w_{i-1}) = \begin{cases} \frac{\text{Count}^*(w_{i-2}, w_{i-1}, w_i)}{\text{Count}(w_{i-2}, w_{i-1})} & \text{If } w_i \in \mathcal{A}(w_{i-2}, w_{i-1}) \\ \frac{\alpha(w_{i-2}, w_{i-1}) q_{BO}(w_i \mid w_{i-1})}{\sum_{w \in \mathcal{B}(w_{i-2}, w_{i-1})} q_{BO}(w \mid w_{i-1})} & \text{If } w_i \in \mathcal{B}(w_{i-2}, w_{i-1}) \end{cases}$$

where

$$\alpha(w_{i-2}, w_{i-1}) = 1 - \sum_{w \in \mathcal{A}(w_{i-2}, w_{i-1})} \frac{\text{Count}^*(w_{i-2}, w_{i-1}, w)}{\text{Count}(w_{i-2}, w_{i-1})}$$

Summary

- Three steps in deriving the language model probabilities:
  1. Expand $p(w_1, w_2 \ldots w_n)$ using Chain rule.
  2. Make Markov Independence Assumptions
     $p(w_i \mid w_1, w_2 \ldots w_{i-2}, w_{i-1}) = p(w_i \mid w_{i-2}, w_{i-1})$
  3. Smooth the estimates using low order counts.

- Other methods used to improve language models:
  - "Topic" or "long-range" features.
  - Syntactic models.

It's generally hard to improve on trigram models though!!