Language Models

Computational Linguistics I: Jordan Boyd-Graber

University of Maryland

September 23, 2013

COLLEGE OF
INFORMATION
STUDIES

Slides adapted from Philipp Koehn

Roadmap

- Why do we need language models?
- Definition of language models
- Estimating probability distributions
- Evaluating language models
- Dealing with zeroes

Outline

Language models

- **Language models** answer the question: *How likely is a string of English words good English?*
- Autocomplete on phones and websearch
- Creating English-looking documents
- Very common in machine translation systems
  - Help with reordering

$$p_{\text{lm}}(\text{the house is small}) > p_{\text{lm}}(\text{small the is house})$$

  - Help with word choice

$$p_{\text{lm}}(\text{I am going home}) > p_{\text{lm}}(\text{I am going house})$$

N-Gram Language Models

- Given: a string of English words $W = w_1, w_2, w_3, ..., w_n$
- Question: what is $p(W)$?
- Sparse data: Many good English sentences will not have been seen before

$\rightarrow$ Decomposing $p(W)$ using the chain rule:

$$p(w_1, w_2, w_3, ..., w_n) = p(w_1)\, p(w_2|w_1)\, p(w_3|w_1, w_2)...p(w_n|w_1, w_2, ...w_{n-1})$$

(not much gained yet, $p(w_n|w_1, w_2, ...w_{n-1})$ is equally sparse)

Markov Chain

- **Markov assumption**:
  - only previous history matters
  - limited memory: only last k words are included in history (older words less relevant)

  $\longrightarrow$ k**th order Markov model**
- For instance 2-gram language model:

$$p(w_1, w_2, w_3, ..., w_n) \simeq p(w_1)\, p(w_2|w_1)\, p(w_3|w_2)...p(w_n|w_{n-1})$$

- What is conditioned on, here w_{i-1} is called the **history**

Outline

How good is the LM?

- A good model assigns a text of real English W a high probability
- This can be also measured with cross entropy:

$$H(W) = \frac{1}{n} \log p(W_1^n)$$

- Or, **perplexity**

$$\text{perplexity}(W) = 2^{H(W)}$$

Comparison 1–4-Gram

word	unigram	bigram	trigram	4-gram
i	6.684	3.197	3.197	3.197
would	8.342	2.884	2.791	2.791
like	9.129	2.026	1.031	1.290
to	5.081	0.402	0.144	0.113
commend	15.487	12.335	8.794	8.633
the	3.885	1.402	1.084	0.880
rapporteur	10.840	7.319	2.763	2.350
on	6.765	4.140	4.150	1.862
his	10.678	7.316	2.367	1.978
work	9.993	4.816	3.498	2.394
.	4.896	3.020	1.785	1.510
</s>	4.828	0.005	0.000	0.000
average				
perplexity	265.136	16.817	6.206	4.758

Outline

How do we estimate a probability?

- Suppose we want to estimate $P(w_n = \text{“}home\text{”} | h = \text{go})$.

How do we estimate a probability?

- Suppose we want to estimate $P(w_n = \text{“}home\text{”} | h = \texttt{go})$.

home	**home**	big	with	to
big	with	to	and	money
and	**home**	big	and	**home**
money	**home**	and	big	to

How do we estimate a probability?

- Suppose we want to estimate $P(w_n = \text{“}home\text{”} | h = \text{go})$.

home	**home**	big	with	to
big	with	to	and	money
and	**home**	big	and	**home**
money	**home**	and	big	to

- Maximum likelihood (ML) estimate of the probability is:

$$\hat{\theta}_i = \frac{n_i}{\sum_k n_k} \tag{1}$$

Example: 3-Gram

- Counts for trigrams and estimated word probabilities

the red (total: 225)

word	c.	prob.
cross	123	0.547
tape	31	0.138
army	9	0.040
card	7	0.031
,	5	0.022

  - 225 trigrams in the Europarl corpus start with **the red**
  - 123 of them end with **cross**

$\longrightarrow$ maximum likelihood probability is $\frac{123}{225} = 0.547$.

Example: 3-Gram

- Counts for trigrams and estimated word probabilities

the red (total: 225)

word	c.	prob.
cross	123	0.547
tape	31	0.138
army	9	0.040
card	7	0.031
,	5	0.022

  - 225 trigrams in the Europarl corpus start with **the red**
  - 123 of them end with **cross**

  $\longrightarrow$ maximum likelihood probability is $\frac{123}{225} = 0.547$.

- Is this reasonable?

The problem with maximum likelihood estimates: Zeros

- If there were no occurrences of "bageling" in a history go, we'd get a zero estimate:

$$\hat{P}(\text{“bageling”}|\ \text{go}) = \frac{T_{\text{go, “bageling”}}}{\sum_{w' \in V} T_{\text{go},w'}} = 0$$

- $\rightarrow$ We will get $P(\text{go}|d) = 0$ for any sentence that contains go bageling!
- Zero probabilities cannot be conditioned away.

How do we estimate a probability?

- In computational linguistics, we often have a *prior* notion of what our probability distributions are going to look like (for example, non-zero, sparse, uniform, etc.).
- This estimate of a probability distribution is called the maximum a posteriori (MAP) estimate:

$$\theta_{\text{MAP}} = \text{argmax}_\theta f(x|\theta)g(\theta) \tag{2}$$

Add-One Smoothing

- Equivalent to assuming a **uniform** prior over all possible distributions over the next word (you'll learn why in CL2)
- But there are many more unseen n-grams than seen n-grams
- Example: Europarl 2-bigrams:
  - $86,700$ distinct words
  - $86,700^2 = 7,516,890,000$ possible bigrams
  - but only about $30,000,000$ words (and bigrams) in corpus

How do we estimate a probability?

- Assuming a **sparse Dirichlet** prior, $\alpha < 1$ to each count

$$\theta_i = \frac{n_i + \alpha_i}{\sum_k n_k + \alpha_k} \tag{3}$$

- α_i is called a smoothing factor, a pseudocount, etc.

How do we estimate a probability?

- Assuming a **sparse Dirichlet** prior, $\alpha < 1$ to each count

$$\theta_i = \frac{n_i + \alpha_i}{\sum_k n_k + \alpha_k} \tag{3}$$

- α_i is called a smoothing factor, a pseudocount, etc.
- When $\alpha_i = 1$ for all i, it's called "Laplace smoothing"

How do we estimate a probability?

- Assuming a **sparse Dirichlet** prior, $\alpha < 1$ to each count

$$\theta_i = \frac{n_i + \alpha_i}{\sum_k n_k + \alpha_k} \tag{3}$$

- α_i is called a smoothing factor, a pseudocount, etc.
- When $\alpha_i = 1$ for all i, it's called "Laplace smoothing"
- What is a good value for α?
- Could be optimized on held-out set to find the "best" language model

Example: 2-Grams in Europarl

Count	Adjusted count		Test count
c	$(c+1)\frac{n}{n+v^2}$	$(c+\alpha)\frac{n}{n+\alpha v^2}$	t_c
0	0.00378	0.00016	0.00016
1	0.00755	0.95725	0.46235
2	0.01133	1.91433	1.39946
3	0.01511	2.87141	2.34307
4	0.01888	3.82850	3.35202
5	0.02266	4.78558	4.35234
6	0.02644	5.74266	5.33762
8	0.03399	7.65683	7.15074
10	0.04155	9.57100	9.11927
20	0.07931	19.14183	18.95948

Example: 2-Grams in Europarl

Count	Adjusted count		Test count
c	$(c+1)\frac{n}{n+v^2}$	$(c+\alpha)\frac{n}{n+\alpha v^2}$	t_c
0	0.00378	0.00016	0.00016
1	0.00755	0.95725	0.46235
2	0.01133	1.91433	1.39946
3	0.01511	2.87141	2.34307
8	0.03399	7.65683	7.15074
10	0.04155	9.57100	9.11927
20	0.07931	19.14183	18.95948

Can we do better?

In higher-order models, we can learn from similar contexts!

Outline

Back-Off

- In given corpus, we may never observe
  - **Scottish beer drinkers**
  - **Scottish beer eaters**
- Both have count 0

  $\longrightarrow$ our smoothing methods will assign them same probability
- Better: backoff to bigrams:
  - **beer drinkers**
  - **beer eaters**

Interpolation

- Higher and lower order n-gram models have different strengths and weaknesses
  - high-order n-grams are sensitive to more context, but have sparse counts
  - low-order n-grams consider only very limited context, but have robust counts
- Combine them

$$\begin{aligned} p_I(w_3|w_1, w_2) = \quad & \lambda_1 \, p_1(w_3) \\ & \times \lambda_2 \, p_2(w_3|w_2) \\ & \times \lambda_3 \, p_3(w_3|w_1, w_2) \end{aligned}$$

Back-Off

- Trust the highest order language model that contains n-gram

$$p_n^{BO}(w_i|w_{i-n+1},...,w_{i-1}) =$$
$$= \begin{cases} \alpha_n(w_i|w_{i-n+1},...,w_{i-1}) \\ \qquad \text{if } \text{count}_n(w_{i-n+1},...,w_i) > 0 \\ d_n(w_{i-n+1},...,w_{i-1})\, p_{n-1}^{BO}(w_i|w_{i-n+2},...,w_{i-1}) \\ \qquad \text{else} \end{cases}$$

- Requires
  - adjusted prediction model $\alpha_n(w_i|w_{i-n+1},...,w_{i-1})$
  - discounting function $d_n(w_1,...,w_{n-1})$

Diversity of Histories

- Consider the word **York**
  - fairly frequent word in Europarl corpus, occurs 477 times
  - as frequent as **foods**, **indicates** and **providers**

  $\longrightarrow$ in unigram language model: a respectable probability
- However, it almost always directly follows **New** (473 times)
- Recall: unigram model only used, if the bigram model inconclusive
  - **York** unlikely second word in unseen bigram
  - in back-off unigram model, **York** should have low probability

Kneser-Ney Smoothing

- Kneser-Ney smoothing takes diversity of histories into account
- Count of histories for a word

$$N_{1+}(\bullet w) = |\{w_i : c(w_i, w) > 0\}|$$

- Recall: maximum likelihood estimation of unigram language model

$$p_{ML}(w) = \frac{c(w)}{\sum_i c(w_i)}$$

- In Kneser-Ney smoothing, replace raw counts with count of histories

$$p_{KN}(w) = \frac{N_{1+}(\bullet w)}{\sum_{w_i} N_{1+}(w_i w)}$$

Interpolated Back-Off

- Back-off models use only highest order n-gram
  - if sparse, not very reliable.
  - two different n-grams with same history occur once → same probability
  - one may be an outlier, the other under-represented in training
- To remedy this, always consider the lower-order back-off models
- Adapting the α function into interpolated α_I function by adding back-off

$$\alpha_I(w_n|w_1,...,w_{n-1}) = \alpha(w_n|w_1,...,w_{n-1}) + d(w_1,...,w_{n-1})\, p_I(w_n|w_2,...,w_{n-1})$$

- Note that d function needs to be adapted as well

Evaluation

Evaluation of smoothing methods:

Perplexity for language models trained on the Europarl corpus

Smoothing method	**bigram**	**trigram**	**4-gram**
Good-Turing	96.2	62.9	59.9
Witten-Bell	97.1	63.8	60.4
Modified Kneser-Ney	95.4	61.6	58.6
Interpolated Modified Kneser-Ney	94.5	59.3	54.0

Reducing Vocabulary Size

- For instance: each number is treated as a separate token
- Replace them with a number token num
  - but: we want our language model to prefer

$$p_{\text{lm}}(\text{I pay 950.00 in May 2007}) > p_{\text{lm}}(\text{I pay 2007 in May 950.00})$$

  - not possible with number token

$$p_{\text{lm}}(\text{I pay num in May num}) = p_{\text{lm}}(\text{I pay num in May num})$$

- Replace each digit (with unique symbol, e.g., @ or **5**), retain some distinctions

$$p_{\text{lm}}(\text{I pay 555.55 in May 5555}) > p_{\text{lm}}(\text{I pay 5555 in May 555.55})$$

Summary

- Language models: *How likely is a string of English words good English?*
- N-gram models (Markov assumption)
- Perplexity
- Count smoothing
- Interpolation and backoff

In Class . . .

- Any remaining questions for first homework
- Working through Knesser-Ney example
- Discussing homework assignment: building bigram language models