NLP: N-Grams

Dan Garrette
dhg@cs.utexas.edu

October 3, 2013

1 Language Modeling Tasks

- Language idenfication / Authorship identification
- Machine Translation
- Speech recognition
- Optical character recognition (OCR)
- Context-sensitive spelling correction
- Predictive text (text messaging clients, search engines, etc)
- Generating spam
- Code-breaking (e.g. decipherment)

2 Kinds of Language Models

- Bag of words
- Sequences of words
- Sequences of tagged words
- Grammars
- Topic models

3 Language as a sequence of words

- What is the probability of seeing a particular sequence of words?
- Simplified model of langauge: words in order
- p(some words in sequence)

- p("an interesting story") ?
- p("a interesting story") ?
- p("a story interesting") ?

- $p(\text{next} \mid \text{some sequence of previous words})$
  - p("austin" | "university of texas at") ?
  - p("dallas" | "university of texas at") ?
  - p("giraffe" | "university of texas at") ?

Use as a language model

- Language ID: This sequence is most likely to be seen in what language?
- Authorship ID: What is the most likely person to have produced this sequence of words?
- Machine Translation: Which sentence (word choices, ordering) seems most like the target language?
- Text Prediction: What's the most likely next word in the sequence

Notation

- Probability of a sequence is a joint probability of words
- But the order of the words matters, so each gets its own feature
- p("an interesting story") $= p(w_1 = an,\ w_2 = interesting,\ w_3 = story)$
  - We will abbreviate this as p(an interesting story), but remember that the order matters, and that the words should be thought of as separate features
- p("austin" | "university of texas at")
  $= p(w_0 = austin \mid w_{-4} = university,\ w_{-3} = of,\ w_{-2} = texas,\ w_{-1} = at)$
  - So w_0 means "current word" and w_{-1} is "the previous word".
  - We will abbreviate this as p(austin|university of texas at), but remember that the order matters, and that the words should be thought of as separate features
- We will typically use the short-hand

4 Counting Words

- Terminology:
  - **Type:** Distinct word
    Token: Particular occurrence of a word
  - "the man saw the saw": 3 types, 5 tokens

What is a word? What do we count?

- Punctuation? Separate from neighboring words? Keep it at all?
- Stopwords?
- Lowercase everything?
- Distinct numbers vs. $\langle number \rangle$
- Hyphenated words?
- Lemmas only?
- Disfluencies? (um, uh)

\\\\ Day 2

5 Estimating Sequence Probabilities

- To build a statistical model, we need to set parameters.
- Our parameters: probabilities of various sequences of text
- Maximum Likelihood Estimate (MLE): of all the sequences of length N, what proportion are the relevant sequence?
- $p(\text{university of texas at austin})$
  $= p(w_1 = \text{university},\ w_2 = \text{of},\ w_3 = \text{texas},\ w_4 = \text{at},\ w_5 = \text{austin})$
  $= \frac{C(\text{“university of texas at austin”})}{C(\textit{all 5-word sequences})} = \frac{3}{25{,}000{,}000}$
- $p(\text{a bank}) = \frac{C(\text{“a bank”})}{C(\textit{all 2-word sequences})} = \frac{609}{50{,}000{,}000}$
- $p(\text{in the}) = \frac{C(\text{“in the”})}{C(\textit{all 2-word sequences})} = \frac{312{,}776}{50{,}000{,}000}$
- Long sequences are unlikely to have any counts:
  $p($*the university of texas football team started the season off right by scoring a touchdown in the final seconds of play to secure a stunning victory over the out-of-town challengers*$)$
  $= \frac{C(\text{... that sentence ...})}{C(\textit{all 30-word sequences})} = \mathbf{0.0}$
- Even shorter sentences may not have counts, even if they make sense, are perfecty grammatical, and not improbable that someone might say them:
  - “*university of texas in amarillo*”
- We need a way of estimating the probability of a long sequence, even though counts will be low.

Make a "naïve" assumption?

- With naïve Bayes, we dealt with this problem by assuming that all features were independent.
- $p(w_1 = \text{university},\ w_2 = \text{of},\ w_3 = \text{texas},\ w_4 = \text{in},\ w_5 = \text{amarillo}) =$
  $p(w = \text{university}) \cdot p(w = \text{of}) \cdot p(w = \text{texas}) \cdot p(w = \text{in}) \cdot p(w = \text{amarillo})$
- But this loses the difference between "university of texas in amarillo", which seems likely, and "university texas amarillo in of", which does not
- This amounts to a "bag of words" model

Use Chain Rule?

- Long sequences are sparse, short sequences are less so
- Break down long sequences using the chain rule
- $p(\text{university of texas in amarillo}) =$
  $p(\text{university}) \cdot p(\text{of} \mid \text{university}) \cdot p(\text{texas} \mid \text{university of}) \cdot p(\text{in} \mid \text{university of texas})$
  $\cdot\, p(\text{amarillo} \mid \text{university of texas in})$
- "p(seeing 'university') times p(seeing 'of' given that the previous word was 'university') times p(seeing 'texas' given that the previous two words were 'university of') ..."
- $p(\text{university}) = \frac{C(\text{university})}{\sum_{x \in V} C(x)} = \frac{C(\text{university})}{C(all\ \ words)}$; easy to estimate
  $p(\text{of} \mid \text{university}) = \frac{C(\text{university of})}{\sum_w C(\text{university } x)} = \frac{C(\text{university of})}{C(\text{university})}$; easy to estimate
  $p(\text{texas} \mid \text{university of}) = \frac{C(\text{university of texas})}{\sum_w C(u\text{university of } x)} = \frac{C(\text{university of texas})}{C(\text{university of})}$; easy to estimate
  $p(\text{in} \mid \text{university of texas}) = \frac{C(\text{university of texas in})}{\sum_w C(u\text{university of texas } x)} = \frac{C(\text{university of texas in})}{C(\text{university of texas})}$; easy to estimate
  $p(\text{amarillo} \mid \text{university of texas in}) = \frac{C(\text{university of texas in amarillo})}{\sum_w C(u\text{university of texas in } x)} = \frac{C(\text{university of texas in amarillo})}{C(\text{university of texas in})}$; **same problem**
- So this doesn't help us at all.

6 N-Grams

- We don't necessarily want a "fully naïve" solution
  - Partial independence: limit how far back we look
- "Markov assumption": future behavior depends only on recent history
  - k^{th}-order Markov model: depend only on k most recent states
- **n-gram**: sequence of n words
- **n-gram model**: statistical model of word sequences using n-grams.

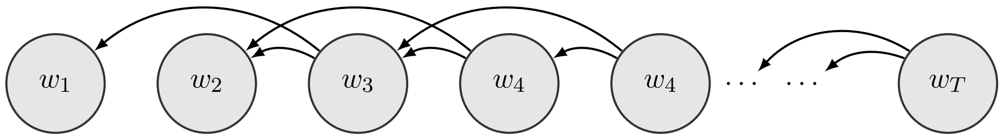


Figure 1: Trigram model showing conditional dependencies

- Approximate all conditional probabilities by only looking back n-1 words (conditioning only on the previous n-1 words)

- For n, estimate everything in terms of: $p(w_n \mid w_1, w_2, ..., w_{n-1})$
- $p(w_T \mid w_1, w_2, ..., w_{T-1}) \approx p(w_T \mid w_{T-(n+1)}, ..., w_{T-1})$
- p(university of texas in amarillo)
  - 5+ -gram:
    $p(\text{university}) \cdot p(\text{of} \mid \text{university}) \cdot p(\text{texas} \mid \text{university of}) \cdot p(\text{in} \mid \text{university of texas})$
    $\cdot\, p(\text{amarillo} \mid \text{university of texas in})$
  - 3-gram (trigram):
    $p(\text{university}) \cdot p(\text{f} \mid \text{university}) \cdot p(\text{texas} \mid \text{university of}) \cdot p(\text{in} \mid \text{of texas})$
    $\cdot\, p(\text{amarillo} \mid \text{texas in})$
  - 2-gram (bigram):
    $p(\text{university}) \cdot p(\text{of} \mid \text{university}) \cdot p(\text{texas} \mid \text{of}) \cdot p(\text{in} \mid \text{texas}) \cdot p(\text{amarillo} \mid \text{in})$
  - 1-gram (unigram) / bag-of-words model / full independence:
    $p(\text{university}) \cdot p(\text{of}) \cdot p(\text{texas}) \cdot p(\text{in}) \cdot p(\text{amarillo})$
- Idea: reduce necessary probabilities to an estimatable size.
- Estimating trigrams:

  $p(\text{university}) \cdot p(\text{of} \mid \text{university}) \cdot p(\text{texas} \mid \text{university of}) \cdot p(\text{in} \mid \text{of texas})$
  $\cdot\, p(\text{amarillo} \mid \text{texas in})$

  $$= \frac{C(\text{university})}{\sum_{x \in V} C(x)} \cdot \frac{C(\text{university of})}{\sum_{x \in V} C(\text{university } x)} \cdot \frac{C(\text{university of texas})}{\sum_{x \in V} C(\text{university of } x)} \cdot \frac{C(\text{of texas in})}{\sum_{x \in V} C(\text{of texas } x)} \cdot \frac{C(\text{texas in amarillio})}{\sum_{x \in V} C(\text{texas in } x)}$$

  $$= \frac{C(\text{university})}{C(\textit{all words})} \cdot \frac{C(\text{university of})}{C(\text{university})} \cdot \frac{C(\text{university of texas})}{C(\text{university of})} \cdot \frac{C(\text{of texas in})}{C(\text{of texas})} \cdot \frac{C(\text{texas in amarillio})}{C(\text{texas in})}$$
- All of these should be easy to estimate!
- Other advantages:
  - Smaller n means fewer parameters to store. Means less memory required. Makes a difference on huge datasets or on limited memory devices (like mobile phones).

7 N-Gram Model of Sentences

- Sentences are sequences of words, but with starts and ends.
- We also want to model the likelihood of words being at the beginning/end of a sentence.

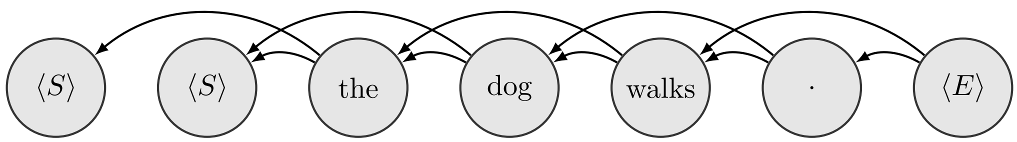


Figure 2: Trigram Model for the sentence "the dog walks ."

- Append special "words" to the sentence
  - n–1 '$\langle S \rangle$' symbols to beginning
  - only one '$\langle E \rangle$' to then end needed
    - $p(\langle E \rangle \mid . \langle E \rangle) = 1.0$ since $\langle E \rangle$ would always be followed by $\langle E \rangle$.
- "the man walks the dog ." (trigrams)
  - Becomes "$\langle S \rangle$ $\langle S \rangle$ the man walks the dog . $\langle E \rangle$"
  - $p(\langle S \rangle\ \langle S \rangle$ the man walks the dog . $\langle E \rangle) =$
    $p(\text{the} \mid \langle S \rangle\ \langle S \rangle)$
    $\cdot\, p(\text{man} \mid \langle S \rangle\ \text{the})$
    $\cdot\, p(\text{walks} \mid \text{the man})$
    $\cdot\, p(\text{the} \mid \text{man walks})$
    $\cdot\, p(\text{dog} \mid \text{walks the})$
    $\cdot\, p(. \mid \text{the dog})$
    $\cdot\, p(\langle E \rangle \mid \text{dog} .)$
- Can be generalized to model longer texts: paragraphs, documents, etc:
  - Good: can model ngrams that cross sentences (e.g. $p(w_0 \mid .)$ or $p(w_0 \mid ?)$)
  - Bad: more sparsity on $\langle S \rangle$ and $\langle E \rangle$

8 Sentence Likelihood Examples

Example dataset:

```
<S> the dog runs . <E>
<S> the dog walks . <E>
<S> the man walks . <E>
<S> a man walks the dog . <E>
<S> the cat walks . <E>
<S> the dog chases the cat . <E>
```

Sentence Likelihood with Bigrams:

$$p(\langle S \rangle \text{ the dog walks . } \langle E \rangle)$$

$$= p(\text{the} \mid \langle S \rangle) \cdot p(\text{dog} \mid \text{the}) \cdot p(\text{walks} \mid \text{dog}) \cdot p(. \mid \text{walks}) \cdot p(\langle E \rangle \mid .)$$

$$= \frac{C(\langle S \rangle \text{ the})}{\sum_{x \in V - \langle E \rangle} C(\langle S \rangle\ x)} \cdot \frac{C(\text{the dog})}{\sum_{x \in V} C(\text{the } x)} \cdot \frac{C(\text{dog walks})}{\sum_{x \in V} C(\text{dog } x)} \cdot \frac{C(\text{walks .})}{\sum_{x \in V} C(\text{walks } x)} \cdot \frac{C(.\ \langle E \rangle)}{\sum_{x \in V} C(.\ x)}$$

$$= \frac{C(\langle S \rangle \text{ the})}{C(\langle S \rangle)} \cdot \frac{C(\text{the dog})}{C(\text{the})} \cdot \frac{C(\text{dog walks})}{C(\text{dog})} \cdot \frac{C(\text{walks .})}{C(\text{walks})} \cdot \frac{C(.\ \langle E \rangle)}{C(.)}$$

$$= \frac{5}{6} \cdot \frac{4}{7} \cdot \frac{1}{4} \cdot \frac{3}{4} \cdot \frac{6}{6} = 0.83 \cdot 0.57 \cdot 0.25 \cdot 0.75 \cdot 1.0 = \mathbf{0.089}$$

$p(\langle S \rangle$ the cat walks the dog . $\langle E \rangle)$

$$= p(\text{the} \mid \langle S \rangle) \cdot p(\text{cat} \mid \text{the}) \cdot p(\text{walks} \mid \text{cat}) \cdot p(\text{the} \mid \text{walks}) \cdot p(\text{dog} \mid \text{the}) \cdot p(. \mid \text{dog}) \cdot p(\langle E \rangle \mid .)$$

$$= \frac{C(\langle S \rangle \text{ the})}{\sum_{x \in V - \langle E \rangle} C(\langle S \rangle\ x)} \cdot \frac{C(\text{the cat})}{\sum_{x \in V} C(\text{the } x)} \cdot \frac{C(\text{cat walks})}{\sum_{x \in V} C(\text{cat } x)} \cdot \frac{C(\text{walks the})}{\sum_{x \in V} C(\text{walks } x)} \cdot \frac{C(\text{the dog})}{\sum_{x \in V} C(\text{the } x)} \cdot \frac{C(\text{dog .})}{\sum_{x \in V} C(\text{dog } x)} \cdot \frac{C(.\ \langle E \rangle)}{\sum_{x \in V} C(.\ x)}$$

$$= \frac{C(\langle S \rangle \text{ the})}{C(\langle S \rangle)} \cdot \frac{C(\text{the cat})}{C(\text{the})} \cdot \frac{C(\text{cat walks})}{C(\text{cat})} \cdot \frac{C(\text{walks the})}{C(\text{walks})} \cdot \frac{C(\text{the dog})}{C(\text{the})} \cdot \frac{C(\text{dog .})}{C(\text{dog})} \cdot \frac{C(.\ \langle E \rangle)}{C(.)}$$

$$= \frac{5}{6} \cdot \frac{1}{7} \cdot \frac{1}{1} \cdot \frac{1}{4} \cdot \frac{4}{7} \cdot \frac{1}{4} \cdot \frac{6}{6} = 0.83 \cdot 0.14 \cdot 1.0 \cdot 0.25 \cdot 0.57 \cdot 0.25 \cdot 1.0 = \mathbf{0.004}$$

$p(\langle S \rangle$ the cat runs . $\langle E \rangle)$

$$= p(\text{the} \mid \langle S \rangle) \cdot p(\text{cat} \mid \text{the}) \cdot p(\text{runs} \mid \text{cat}) \cdot p(. \mid \text{runs}) \cdot p(\langle E \rangle \mid .)$$

$$= \frac{C(\langle S \rangle \text{ the})}{\sum_{x \in V} C(\langle S \rangle\ x)} \cdot \frac{C(\text{the cat})}{\sum_{x \in V} C(\text{the } x)} \cdot \frac{C(\text{cat runs})}{\sum_{x \in V} C(\text{cat } x)} \cdot \frac{C(\text{runs .})}{\sum_{x \in V} C(\text{runs } x)} \cdot \frac{C(.\ \langle E \rangle)}{\sum_{x \in V} C(.\ x)}$$

$$= \frac{C(\langle S \rangle \text{ the})}{C(\langle S \rangle)} \cdot \frac{C(\text{the cat})}{C(\text{the})} \cdot \frac{C(\text{cat runs})}{C(\text{cat})} \cdot \frac{C(\text{runs .})}{C(\text{runs})} \cdot \frac{C(.\ \langle E \rangle)}{C(.)}$$

$$= \frac{5}{6} \cdot \frac{1}{7} \cdot \frac{\mathbf{0}}{1} \cdot \frac{1}{1} \cdot \frac{6}{6} = 0.83 \cdot 0.14 \cdot \mathbf{0.0} \cdot 1.0 \cdot 1.0 = \mathbf{0.000}$$

- Longer sentences have lower likelihoods.
  - This makes sense because longer sequences are harder to match exactly.
- Zeros happen when an n-gram isn't seen.

\\ Day 3

9 Handling Sparsity

How big of a problem is sparsity?

- Alice's Adventures in Wonderland
  - Vocabulary (all word types) size: $V = 3{,}569$
  - Distinct bigrams: 17,149; $\frac{17{,}149}{|V|^2}$, or 99.8% of possible bigrams unseen
  - Distinct trigrams: 28,540; $\frac{17{,}149}{|V|^3}$, or 99.9999994% of possible trigrams unseen
- If a sequence contains an unseen ngram, it will have likelihood zero: an impossible sequence.
- Many legitimate ngrams will simply be absent from the corpus.
- This does not mean they are impossible.

- Even ungrammatical/nonsense ngrams should not cause an entire sequence's likelihood to be zero.
- Many others will be too infrequent to estimate well.

Add-λ Smoothing

- Add some constant λ to every count, including unseen ngrams
- V is the Vocabulary — all posslb "next word" types — including $\langle E\rangle$ (if necessary $n > 1$)
  - Don't need $\langle S\rangle$ because it will never be the 'next word'
- $p(w_0 \mid w_{1-n} \ldots w_{-1}) = \frac{C(w_{1-n} \ldots w_{-1}\ w_0)+\lambda}{\sum_{x\in V}(C(w_{1-n} \ldots w_{-1}\ x)+\lambda)} = \frac{C(w_{1-n} \ldots w_{-1}\ w_0)+\lambda}{(\sum_{x\in V} C(w_{1-n} \ldots w_{-1}\ x))+\lambda|V|} = \frac{C(w_{1-n} \ldots w_{-1}\ w_0)+\lambda}{C(w_{1-n} \ldots w_{-1})+\lambda|V|}$
  - Add $|V|$ to denominator to account for the fact that there is an extra count for every x
- In practice it over-smoothes, even when $\lambda < 1$
- Example dataset:

```
<S> <S> the dog runs . <E>
<S> <S> the dog walks . <E>
<S> <S> the man walks . <E>
<S> <S> a man walks the dog . <E>
<S> <S> the cat walks . <E>
<S> <S> the dog chases the cat . <E>
```

$V = \{a, cat, chases, dog, man, runs, the, walks, ., \langle E\rangle\}$
$|V| = 10$

Sentence Likelihood with Trigrams:

$p(\text{"the cat runs ."})$

$= p(\langle S\rangle\ \langle S\rangle \text{ the cat runs . } \langle E\rangle)$

$= p(\text{the} \mid \langle S\rangle\ \langle S\rangle) \cdot p(\text{cat} \mid \langle S\rangle \text{ the}) \cdot p(\text{runs} \mid \text{the cat}) \cdot p(. \mid \text{cat runs}) \cdot p(\langle E\rangle \mid \text{runs .})$

$= \frac{C(\langle S\rangle\ \langle S\rangle\ \text{the})}{\sum_{x\in V}(C(\langle S\rangle\ \langle S\rangle\ x)+1)} \cdot \frac{C(\langle S\rangle\ \text{the cat})+1}{\sum_{x\in V}(C(\langle S\rangle\ \text{the}\ x)+1)} \cdot \frac{C(\text{the cat runs})+1}{\sum_{x\in V}(C(\text{the cat}\ x)+1)} \cdot \frac{C(\text{cat runs .})+1}{\sum_{x\in V}(C(\text{cat runs}\ x)+1)} \cdot \frac{C(\text{runs . }\langle E\rangle)+1}{\sum_{x\in V}(C(\text{runs . }x)+1)}$

$= \frac{C(\langle S\rangle\ \langle S\rangle\ \text{the})+1}{C(\langle S\rangle\ \langle S\rangle)+(|V|-1)} \cdot \frac{C(\langle S\rangle\ \text{the cat})+1}{C(\langle S\rangle\ \text{the})+|V|} \cdot \frac{C(\text{the cat runs})+1}{C(\text{the cat})+|V|} \cdot \frac{C(\text{cat runs .})+1}{C(\text{cat runs})+|V|} \cdot \frac{C(\text{runs . }\langle E\rangle)+1}{C(\text{runs .})+|V|}$

$= \frac{5+1}{6+\mathbf{9}} \cdot \frac{1+1}{5+10} \cdot \frac{\mathbf{0}+1}{\mathbf{2}+10} \cdot \frac{\mathbf{0}+1}{\mathbf{0}+10} \cdot \frac{1+1}{1+10}$

$= 0.40 \cdot 0.13 \cdot \mathbf{0.08} \cdot \mathbf{0.10} \cdot 0.18 = \mathbf{0.000081}$

- If the context was never seen, then the distribution is uniform:

$$\begin{aligned} p_{+\lambda}(w_0 \mid w_{-2}, w_{-1}) &= \frac{C(w_{-2}\ w_{-1}\ w_0) + \lambda}{\left(\sum_{x\in V} C(w_{-2}\ w_{-1}\ x)\right) + \lambda \cdot |V|} \\ &= \frac{0+\lambda}{\left(\sum_{x\in V} 0\right) + \lambda \cdot |V|} \\ &= \frac{0+\lambda}{0+\lambda\cdot|V|} = \frac{\lambda}{\lambda\cdot|V|} = \frac{1}{1\cdot|V|} = \frac{1}{|V|} \end{aligned}$$

- Since $\langle S \rangle$ can never be followed by $\langle E \rangle$
  - $p(\langle E \rangle \mid \langle S \rangle\ \langle S \rangle) = 0$
  - The denominator $\sum_{x \in V} C(\langle S \rangle\ \langle S \rangle\ x)$ only gets $|V|$-1 smoothing counts
- $\langle S \rangle$ not included in V because we can't transition to it.
- More smoothing on less common ngrams
  - With smoothing, we have counts for any possible type following "runs .":
    $p(\langle E \rangle \mid \text{runs .}) = \frac{1+1}{1+10} = \frac{2}{11} = 0.18$
    $p(\text{the} \mid \text{runs .}) = \frac{0+1}{1+10} = \frac{1}{11} = 0.09$
    * Counts of "runs ." are very low (only 1 occurrence), so estimates are bad
    * Bad estimates means more smoothing is good
    * MLE of "runs . $\langle E \rangle$" is 1.0, add-1 smoothed becomes 0.18
      MLE of "runs . the" is 0.0, add-1 smoothed becomes 0.09
    * Original difference of 1.0 becomes difference of 0.09!
    * This makes sense because our estimates are so bad that we really can't make a judgement about what could possibly follow "runs ."
  - Contexts with higher counts have less smoothing:
    $p(\langle E \rangle \mid \text{walks .}) = \frac{3+1}{3+10} = \frac{4}{13} = 0.31$
    $p(\text{the} \mid \text{walks .}) = \frac{0+1}{3+10} = \frac{1}{13} = 0.08$
    * Counts of "walks ." are higher (3 occurrence), so estimates are better
    * Better estimates means less smoothing is good
    * MLE of "walks . $\langle E \rangle$" is 1.0, add-1 smoothed becomes 0.31
      MLE of "walks . the" is 0.0, add-1 smoothed becomes 0.08
    * Original difference of 1.0 becomes difference of 0.23
    * Remains a larger gap than we saw for "runs . x"
    * This makes sense because our estimates are better, so we can be more sure that "walks ." should only be followed by $\langle E \rangle$
- Disadvantages of add-λ smoothing:
  - Over-smoothes.

Good-Turing Smoothing

- Estimate counts of things you haven't seen from counts of things you have
- Estimate probability of things which occur c times with the probability of things which occur $c+1$ times
- $c^* = (c+1)\frac{N_{c+1}}{N_c}$

  $p^*_{GT}(\textit{things with freq 0}) = \frac{N_1}{N}$
- $p_{GT}(w_0 \mid w_{1-n} \ldots w_{-1}) = \frac{C^*(w_{1-n}\ \ldots\ w_{-1}\ w_0)}{C^*(w_{1-n}\ \ldots\ w_{-1})}$

Knesser-Ney Smoothing

- Intuition: interpolate based on "openness" of the context
- Words seen in more contexts are more likely to appear in others
- Even if we haven't seen w_0 following the context, if the context is "open" (supports a wide variety of "next words"), then it is more likely to support w_0
- Boost counts based on $|\{x : C(w_{1-n} \ldots w_{-1}\ x) > 0\}|$, the number of different "next words" seen after $w_{1-n} \ldots w_{-1}$

\\ Day 4

Interpolation

- Mix n-gram probability with probabilities from lower-order models

$$\begin{aligned}\hat{p}(w_0 \mid w_{-2}\ w_{-1}) = &\quad \lambda_3 \cdot p(w_0 \mid w_{-2}\ w_{-1}) \\ &+ \lambda_2 \cdot p(w_0 \mid w_{-1}) \\ &+ \lambda_1 \cdot p(w_0)\end{aligned}$$

- λ_i terms used to decide how much to smooth
- $\sum_i \lambda_i = 1$ (still a valid probability distribution, because they are proportions
- Use *dev* dataset to tune λ hyperparameters
- Also useful for combining models trained on different data:
  - Can interpolate "customized" models with "general" models
  - Baseline English + regional English + user-specific English
  - Little in-domain data, lots of out-of-domain

Stupid Backoff

- If p(n-gram)=0, just use p((n-1)-gram)
- Does *not* yield a valid probability distribution
- Works shockingly well for huge datasets

10 Out-of-Vocabulary Words (OOV)

Add-λ

- If ngram contains OOV item, assume count of λ, just like for all other ngrams.
- Probability distributions become invalid. We can't know the full vocabulary size, so we can't normalize counts correctly.

$\langle unk \rangle$

- Create special token $\langle unk \rangle$
- Create a fixed lexicon L
  - All types in some subset of training data?
  - All types appearing more than k times?
- $V = L + \langle unk \rangle, \quad |V| = |L| + 1$
- Before training, change any word not in L to $\langle unk \rangle$
- Then train as usual as if $\langle unk \rangle$was a normal word
- For new sentence, again replace words not in L with $\langle unk \rangle$before using model
- Probabilities containing $\langle unk \rangle$ measure likelihood with *some rare word*
- Problem: the "rare" word is no longer rare since there are many $\langle unk \rangle$tokens
  - Ngrams with $\langle unk \rangle$will have higher probabilities than those with any particular rare word
  - Not so bad when comparing same sequence under multiple models. All will have inflated probabilities.
  - More problematic when comparing probabilities of different sequences under the same model
    * p(i **totes** know) < p(i **totally** know) < p(i $\langle$**unk**$\rangle$ know)

11 Evaluation

Extrinsic

- Use the model in some larger task. See if it helps.
- More realistic
- Harder

Intrinsic

- Evaluate on a test corpus
- Easier

Perplexity

- Intrinsic measure of model quality
- How well does the model "fit" the test data?
- How "perplexed" is the model whe it sees the test data?
- Measure the probability of the test corpus, normalize for number of words.
- $PP(W) = \sqrt[|W|]{\frac{1}{p(w_1\ w_2\ ...\ w_{|W|})}}$
- With individual sentences: $PP(s_1, s_2, ...) = \sqrt[(\sum_i |s_i|)]{\frac{1}{\prod_i p(s_i)}}$

12 Generative Models

- Generative models are designed to model how the data *could have been generated.*
- The best parameters are those that would most likely generate the data.
- MLE maximizes that likelihood that the training data was generated by the model.
- As such, we can *actually generate* data from a model.
- Trigram model:
  - General:

    For each sequence:
    1. Sample a word w_0 according to $w_0 \sim p(w_0)$
    2. Sample a second word w_1 according to $w_1 \sim p(w_1 \mid w_0)$
    3. Sample a next word w_k according to $w_k \sim p(w_k \mid w_{k-2}\ w_{k-1})$
    4. Repeat step 3 until you feel like stopping.
  - Sentences:

    For each sentence:
    1. Sample a word w_0 according to $w_0 \sim p(w_0 \mid \langle S \rangle\ \langle S \rangle)$
    2. Sample a second word w_1 according to $w_1 \sim p(w_1 \mid w_0\ \langle S \rangle)$
    3. Sample a next word w_k according to $w_k \sim p(w_k \mid w_{k-2}\ w_{k-1})$
    4. Repeat until $\langle E \rangle$ is drawn.
- Longer n generates more coherent text
- Too-large n just ends up generating sentences from the training data because most counts will be 1 (no choice of next word).
- Naïve Bayes was a generative model too!

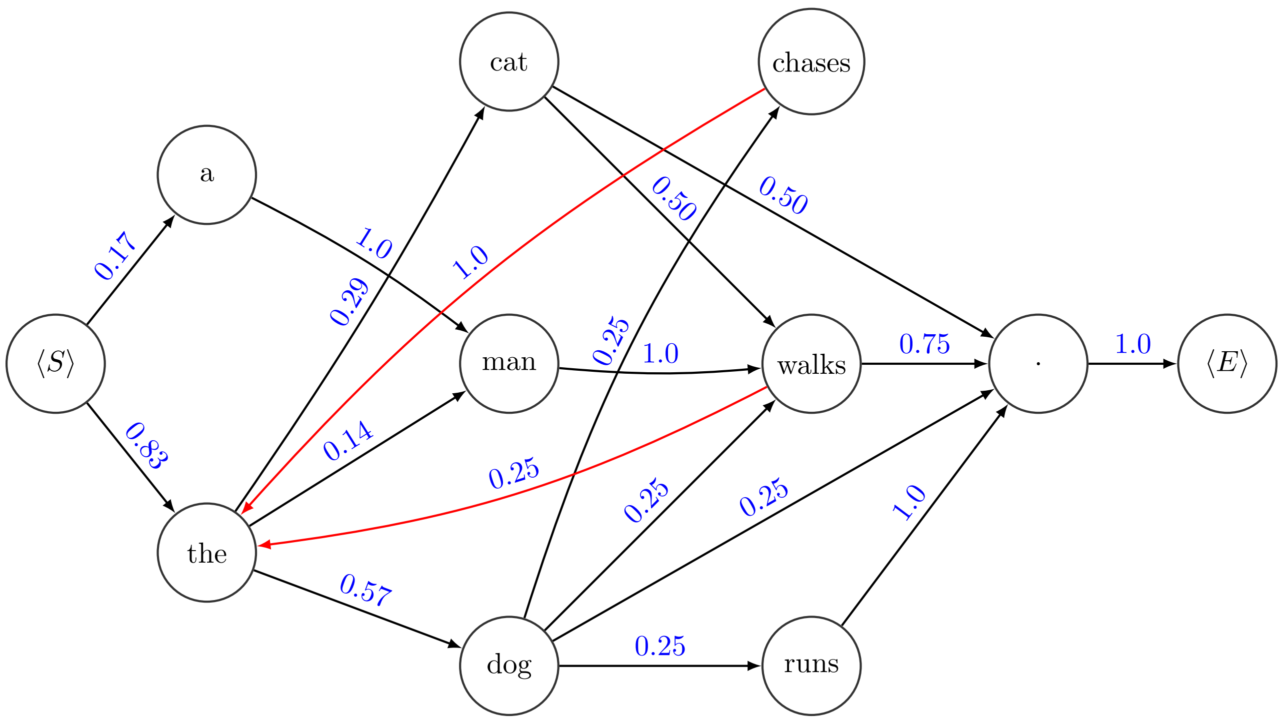


Figure 3: Finite state machine. Missing arrows are assumed to be zero probabilities. With smoothing, there would be an arrow in *both directions* between *every* pair of words.

For each instance:

1. Sample a label l according to $l \sim p(Label = l)$
2. For each feature F: sample a value v according to $v \sim p(F = v \mid Label = l)$

- We will see many more generative models throughout this course

```
p(S -> the) = 0.83
p(S -> a) = 0.17

p(the -> cat) = 0.29
p(the -> dog) = 0.57
p(the -> man) = 0.14

p(a -> man) = 1.00

p(man -> walks) = 1.00

p(cat -> walks) = 0.50
p(cat -> .) = 0.50

p(dog -> chases) = 0.25
p(dog -> runs) = 0.25
p(dog -> .) = 0.25
p(dog -> walks) = 0.25

p(chases -> the) = 1.00

p(runs -> .) = 1.00

p(walks -> the) = 0.25
p(walks -> .) = 0.75

p(. -> <E>) = 1.00
```

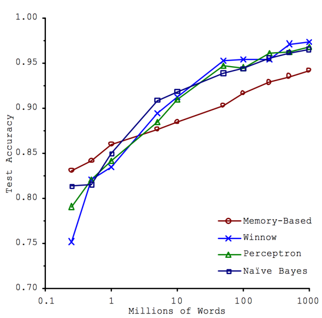
13 How much data?

Choosing n

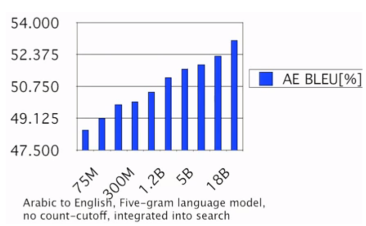
- Large n
  - More context for probabilities:
    $p(\text{phone})$ vs
    $p(\text{phone} \mid \text{cell})$ vs
    $p(\text{phone} \mid \text{your cell})$ vs
    $p(\text{phone} \mid \text{off your cell})$ vs
    $p(\text{phone} \mid \text{turn off your cell})$
  - Long-range dependencies
- Small n
  - Better generalization
  - Better estimates
  - Long-range dependencies

How much training data?

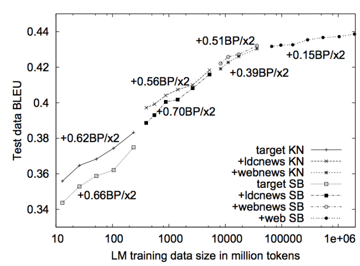
- *As much as possible.*
- More data means better estimates
- Google N-Gram corpus uses 10-grams



(a) Data size matters more than algorithm (Banko and Brill, 2001)



(b) Results keep improving with more data (Norvig: Unreasonable ...)



(c) With enough data, stupid backoff approaches Kesser-Ney accuracy (Brants et al., 2007)

14 Citations

Some content adapted from:

- `http://courses.washington.edu/ling570/gina_fall11/slides/ling570_class8_smoothing.pdf`