n-gram models

Steve Renals
s.renals@ed.ac.uk

ICL — 16 October 2006

Grammatical and statistical approaches

- The rules governing the generation of linguistic events. Grammatical approaches are powerful in limited domains — but they are not always robust.
- The assignment of probabilities to linguistic events. Statistical approaches are more general but typically more shallow.
- Estimate the parameters of statistical models from large text corpora
- n-gram models — directly assign probabilities to word sequences

Two extremes...

> *Every time I fire a linguist the error rates go down.*

Fred Jelinek, former head of the IBM speech recognition research group (1988)

> *But it must be recognized that the notion "probability of a sentence" is an entirely useless one, under any known interpretation of the term.*

Noam Chomsky (1969)

Guess the next word

- Warning of big fall in house

Guess the next word

- Warning of big fall in house prices

Guess the next word

- Warning of big fall in house prices
- Variety reports Sean Connery may be

Guess the next word

- Warning of big fall in house prices
- Variety reports Sean Connery may be retiring

Guess the next word

- Warning of big fall in house prices
- Variety reports Sean Connery may be retiring
- Arsenal stranded in

Guess the next word

- Warning of big fall in house prices
- Variety reports Sean Connery may be retiring
- Arsenal stranded in Trondheim

Guess the next word

- Warning of big fall in house prices
- Variety reports Sean Connery may be retiring
- Arsenal stranded in Trondheim

Guessing the next word is an essential component of many tasks such as speech recognition, handwriting recognition, and context-sensitive spelling correction

How to make a good guess

- A word is easy to guess if it is more probable than any other word

How to make a good guess

- A word is easy to guess if it is more probable than any other word
- Order is important:

How to make a good guess

- A word is easy to guess if it is more probable than any other word
- Order is important:
  - Variety reports Sean Connery may be

How to make a good guess

- A word is easy to guess if it is more probable than any other word
- Order is important:
  - Variety reports Sean Connery may be
  - Variety Sean may reports Connery be

How to make a good guess

- A word is easy to guess if it is more probable than any other word
- Order is important:
  - Variety reports Sean Connery may be
  - Variety Sean may reports Connery be
  - be may Connery Sean reports Variety

How to make a good guess

- A word is easy to guess if it is more probable than any other word
- Order is important:
  - Variety reports Sean Connery may be
  - Variety Sean may reports Connery be
  - be may Connery Sean reports Variety
- Context helps:

How to make a good guess

- A word is easy to guess if it is more probable than any other word
- Order is important:
  - Variety reports Sean Connery may be
  - Variety Sean may reports Connery be
  - be may Connery Sean reports Variety
- Context helps:
  - ...

How to make a good guess

- A word is easy to guess if it is more probable than any other word
- Order is important:
  - Variety reports Sean Connery may be
  - Variety Sean may reports Connery be
  - be may Connery Sean reports Variety
- Context helps:
  - ...
  - Warning...

How to make a good guess

- A word is easy to guess if it is more probable than any other word
- Order is important:
  - Variety reports Sean Connery may be
  - Variety Sean may reports Connery be
  - be may Connery Sean reports Variety
- Context helps:
  - ...
  - Warning...
  - Warning of...

How to make a good guess

- A word is easy to guess if it is more probable than any other word
- Order is important:
  - Variety reports Sean Connery may be
  - Variety Sean may reports Connery be
  - be may Connery Sean reports Variety
- Context helps:
  - ...
  - Warning...
  - Warning of...
  - Warning of big...

How to make a good guess

- A word is easy to guess if it is more probable than any other word
- Order is important:
  - Variety reports Sean Connery may be
  - Variety Sean may reports Connery be
  - be may Connery Sean reports Variety
- Context helps:
  - ...
  - Warning...
  - Warning of...
  - Warning of big...
  - Warning of big fall...

Counting words

- The best way to guess the next word is to first try the most probable, then the second-most probable, and so on
- So we need to estimate the probability of a word

Counting words

- The best way to guess the next word is to first try the most probable, then the second-most probable, and so on
- So we need to estimate the probability of a word
- To estimate word probabilities we can count words—how many tokens of each type?
- More generally we can count n-grams not just words

Two simple approaches to language modelling

Sequence Estimate the probability of a word given the recent sequence of words (eg *resonance* likely to follow *nuclear magnetic*)
Used for speech recognition language modelling, tagging, machine translation

Topic Estimate the probability of a word given the distribution of words in a document (eg *Brown* likely to occur in a document containing *Blair Prime Chancellor Minister Downing leadership*)
Used for text retrieval, document classification

n-grams

- An n-gram is a sequence of n words

n-grams

- An n-gram is a sequence of n words
- Eg: warning of big fall in house prices

n-grams

- An n-gram is a sequence of n words
- Eg: warning of big fall in house prices
  - Unigrams: warning ; of ; big ; fall ; in ; house ; prices

n-grams

- An n-gram is a sequence of n words
- Eg: warning of big fall in house prices
  - Unigrams: warning ; of ; big ; fall ; in ; house ; prices
  - Bigrams: warning of ; of big ; big fall ; fall in ; in house ; house prices

n-grams

- An n-gram is a sequence of n words
- Eg: warning of big fall in house prices
  - Unigrams: warning ; of ; big ; fall ; in ; house ; prices
  - Bigrams: warning of ; of big ; big fall ; fall in ; in house ; house prices
  - Trigrams: warning of big ; of big fall ; big fall in; fall in house ; in house prices

n-grams

- An n-gram is a sequence of n words
- Eg: warning of big fall in house prices
  - Unigrams: warning ; of ; big ; fall ; in ; house ; prices
  - Bigrams: warning of ; of big ; big fall ; fall in ; in house ; house prices
  - Trigrams: warning of big ; of big fall ; big fall in; fall in house ; in house prices
  - 4-grams: warning of big fall ; of big fall in ; big fall in house ; fall in house prices

Example: BBC news transcripts

The THISL corpus of transcribed BBC TV and radio news programmes, containing 7,488,445 word tokens.
Counts and relative frequencies of eight most frequent words:

word	count	rel freq.	word	count	rel freq.
the	394 481	0.0527	and	133 962	0.0179
to	240 001	0.0320	as	109 217	0.0146
a	225 506	0.0301	be	84 020	0.0112
in	177 997	0.0238	that	69 265	0.0092

Probability of a word sequence

- wonderland is an infrequent word... unless we have just seen the words alice in

Probability of a word sequence

- wonderland is an infrequent word... unless we have just seen the words alice in
- We can better estimate the probability of a word if we take into account the word sequence

Probability of a word sequence

- wonderland is an infrequent word... unless we have just seen the words alice in
- We can better estimate the probability of a word if we take into account the word sequence
- Consider a string of N words: $w_1, w_2, w_3, \ldots, w_{N-1}, w_N$.
- Decompose the probability of the string as follows

$$
\begin{aligned}
P(w_1, w_2, w_3, \ldots, w_{N-1}, w_N) &= \\
&P(w_1)P(w_2|w_1)P(w_3|w_1, w_2)P(w_4|w_1, w_2, w_3)\ldots \\
\ldots P(w_{N-1}|w_1, w_2, \ldots, w_{N-2})&P(w_N|w_1, w_2, \ldots, w_{N-2}, w_{N-1})
\end{aligned}
$$

n-gram approximation

- Taking all the previous words into account results in a huge table of probabilities

n-gram approximation

- Taking all the previous words into account results in a huge table of probabilities
- The Markov assumption — consider only the previous (n-1) words

n-gram approximation

- Taking all the previous words into account results in a huge table of probabilities
- The Markov assumption — consider only the previous (n-1) words
- Bigram (n=2) has one word of context:

$$P(w_1, w_2, w_3, \ldots, w_{N-1}, w_N) = P(w_1)P(w_2|w_1)P(w_3|w_2)\ldots P(w_{N-1}|w_{N-2})P(w_N|w_{N-1})$$

$$P(w_3|w_1, w_2) \sim P(w_3|w_2)$$
$$P(w_N|w_1, w_2, \ldots, w_{N-2}, w_{N-1}) \sim P(w_N|w_{N-1})$$

n-gram approximation

- Taking all the previous words into account results in a huge table of probabilities
- The Markov assumption — consider only the previous (n-1) words
- Bigram (n=2) has one word of context:

$$P(w_1, w_2, w_3, \ldots, w_{N-1}, w_N) = P(w_1)P(w_2|w_1)P(w_3|w_2)\ldots P(w_{N-1}|w_{N-2})P(w_N|w_{N-1})$$

$$P(w_3|w_1, w_2) \sim P(w_3|w_2)$$

$$P(w_N|w_1, w_2, \ldots, w_{N-2}, w_{N-1}) \sim P(w_N|w_{N-1})$$

- View a bigram as a simple Markov chain — or a (weighted) finite state machine with a state for each word

Example bigrams

$P(\bullet|fast)$ in the ICSI meetings corpus.

bigram	prob	log(prob)
fast [end-sent]	0.202	-0.695
fast enough	0.049	-1.312
fast forward	0.030	-1.530
fast and	0.027	-1.571
fast because	0.019	-1.723
fast that	0.012	-1.934
fast the	0.011	-1.952

Example bigrams

$P(\bullet|fast)$ in the ICSI meetings corpus.

bigram	prob	log(prob)
fast [end-sent]	0.202	-0.695
fast enough	0.049	-1.312
fast forward	0.030	-1.530
fast and	0.027	-1.571
fast because	0.019	-1.723
fast that	0.012	-1.934
fast the	0.011	-1.952

Bigram probabilities are usually very small — often use log(prob) to avoid floating point underflow

Estimating bigram probabilities

- The relative frequency estimate of a bigram is given by:

$$p(w|v) = \frac{c(v,w)}{\sum_{w'} c(v,w')} = \frac{c(v,w)}{c(v)}$$

$c(v, w)$ is the frequency (count) of word pair (v, w)

Estimating bigram probabilities

- The relative frequency estimate of a bigram is given by:

$$p(w|v) = \frac{c(v, w)}{\sum_{w'} c(v, w')} = \frac{c(v, w)}{c(v)}$$

  $c(v, w)$ is the frequency (count) of word pair (v, w)
- Consider a vocabulary of 50 000 words (typical for a speech recognition system): 2.5×10^9 possible bigrams; 1.25×10^{14} possible trigrams. Therefore most trigrams and bigrams will not be observed in a given corpus
- For a given corpus, $c(v, w) = 0$ for most word pairs, hence most n-grams estimated in this way will be 0!

Estimating n-gram probabilities

- Notation: w_{N-n+1}^{N-1} represents $(n-1)$ words of context, $w_{N-(n-1)}, w_{N-(n-2)}, \ldots, w_{N-1}$ $[N-(n-1) = N-n+1]$

Estimating n-gram probabilities

- Notation: w_{N-n+1}^{N-1} represents $(n-1)$ words of context, $w_{N-(n-1)}, w_{N-(n-2)}, \ldots, w_{N-1}$ $\quad [N-(n-1) = N-n+1]$
- General estimate of n-gram probabilities:

$$p(w_N | w_{N-n+1}^{N-1}) = \frac{c(w_{N-n+1}^{N-1}, w_N)}{c(w_{N-n+1}^{N-1})}$$

Estimating n-gram probabilities

- Notation: w_{N-n+1}^{N-1} represents $(n-1)$ words of context, $w_{N-(n-1)}, w_{N-(n-2)}, \ldots, w_{N-1}$ $\quad [N-(n-1) = N-n+1]$
- General estimate of n-gram probabilities:

$$p(w_N | w_{N-n+1}^{N-1}) = \frac{c(w_{N-n+1}^{N-1}, w_N)}{c(w_{N-n+1}^{N-1})}$$

- This ratio is referred to as the relative frequency
- Estimating probabilities with relative frequencies is an example of maximum likelihood estimation

Estimating n-gram probabilities

- Notation: w_{N-n+1}^{N-1} represents $(n-1)$ words of context, $w_{N-(n-1)}, w_{N-(n-2)}, \ldots, w_{N-1}$ $\quad [N-(n-1) = N-n+1]$
- General estimate of n-gram probabilities:

$$p(w_N | w_{N-n+1}^{N-1}) = \frac{c(w_{N-n+1}^{N-1}, w_N)}{c(w_{N-n+1}^{N-1})}$$

- This ratio is referred to as the relative frequency
- Estimating probabilities with relative frequencies is an example of maximum likelihood estimation
- Jurafsky and Martin (2nd ed: sec 4.3.1; 1st ed: p.202–206) for examples of n-gram generation of text

The Zero Probability Problem

- If an event has a zero probability then we are saying it can never occur!
- Since probabilities sum to 1 this is equivalent to saying that the probabilities of observed n-grams are over-estimated
- Solution: Smooth the n-gram probabilities so that every event has probability greater than zero
  - Discounting — reserve some probability for unseen events
  - Smoothing with lower-level n-grams — use the most precise model allowed by the data

Laplace's law — add one

Consider estimating a unigram probability $P(w_i)$ (vocabulary size is V, total number of word tokens is M):

- Unsmoothed maximum likelihood estimate:

$$P_{ML}(w_i) = \frac{c(w_i)}{\sum_{x=1}^{V} c(w_x)} = \frac{c(w_i)}{M}$$

Laplace's law — add one

Consider estimating a unigram probability $P(w_i)$ (vocabulary size is V, total number of word tokens is M):

- Unsmoothed maximum likelihood estimate:

$$P_{ML}(w_i) = \frac{c(w_i)}{\sum_{x=1}^{V} c(w_x)} = \frac{c(w_i)}{M}$$

- Could just add one to each count (so no more zero counts) and renormalize:

$$P_{LAP}(w_i) = \frac{c(w_i)+1}{\sum_{x=1}^{V}(c(w_x)+1)} = \frac{c(w_i)+1}{M+V}$$

Laplace's law — add one

Consider estimating a unigram probability $P(w_i)$ (vocabulary size is V, total number of word tokens is M):

- Unsmoothed maximum likelihood estimate:

$$P_{ML}(w_i) = \frac{c(w_i)}{\sum_{x=1}^{V} c(w_x)} = \frac{c(w_i)}{M}$$

- Could just add one to each count (so no more zero counts) and renormalize:

$$P_{LAP}(w_i) = \frac{c(w_i) + 1}{\sum_{x=1}^{V}(c(w_x) + 1)} = \frac{c(w_i) + 1}{M + V}$$

- This does not work very well, particularly if there are a lot of unseen events (eg if applied to bigram or trigram estimation)
- Better results if $\lambda < 1$ is added to the counts (eg $\lambda = 1/2$)

Discounting

- Discounting schemes reduce, or discount, the probability estimates of observed events
- The freed probability mass is used for unseen events

Discounting

- Discounting schemes reduce, or discount, the probability estimates of observed events
- The freed probability mass is used for unseen events
- What is the best way to estimate the probability of an unseen event? — Look at the distribution of events seen precisely once!

Discounting

- Discounting schemes reduce, or discount, the probability estimates of observed events
- The freed probability mass is used for unseen events
- What is the best way to estimate the probability of an unseen event? — Look at the distribution of events seen precisely once!
- Many discounting schemes: Good-Turing, Witten-Bell, Absolute. All work similarly in practice (although there are some sophistications in their exact implementation).

Absolute discounting

- Subtract a constant k from each non-zero count and redistribute over unseen events (zero counts)

Absolute discounting

- Subtract a constant k from each non-zero count and redistribute over unseen events (zero counts)
- Let $c(e)$ be the count of event e, and $\hat{c}(e)$ be the discounted count:

$$\hat{c}(e) = \begin{cases} c(e) - k & \text{if } c(e) > 0 \\ \dfrac{k}{u_0} \times \sum_r u_r & \text{if } c(e) = 0 \end{cases},$$

  where u_i is the number of events with a count of i.
- The value of k is typically based on u_1 and u_2, eg $k \sim u_1/(u_1 + 2u_2)$

Absolute discounting

- Subtract a constant k from each non-zero count and redistribute over unseen events (zero counts)
- Let $c(e)$ be the count of event e, and $\hat{c}(e)$ be the discounted count:

$$\hat{c}(e) = \begin{cases} c(e) - k & \text{if } c(e) > 0 \\ \dfrac{k}{u_0} \times \displaystyle\sum_r u_r & \text{if } c(e) = 0 \end{cases},$$

  where u_i is the number of events with a count of i.
- The value of k is typically based on u_1 and u_2, eg $k \sim u_1/(u_1 + 2u_2)$
- Forms the basis of Kneser-Ney discounting (widely used in machine translation and speech recognition)

Interpolation

- Linearly combine n-gram models — discounted estimates of lower-order n-grams more reliable than directly estimating probabilities of unseen higher-order n-grams.

Interpolation

- Linearly combine n-gram models — discounted estimates of lower-order n-grams more reliable than directly estimating probabilities of unseen higher-order n-grams.
- For a trigram (u, v, w), smoothly estimate $p(w|u, v)$ as:

$$p_{int}(w|u,v) = \lambda_3 \hat{p}(w|u,v) + \lambda_2 \hat{p}(w|v) + \lambda_1 \hat{p}(w) + \lambda_0$$

Such that

$$\sum_i \lambda_i = 1$$

Interpolation

- Linearly combine n-gram models — discounted estimates of lower-order n-grams more reliable than directly estimating probabilities of unseen higher-order n-grams.
- For a trigram (u, v, w), smoothly estimate $p(w|u, v)$ as:

$$p_{int}(w|u, v) = \lambda_3 \hat{p}(w|u, v) + \lambda_2 \hat{p}(w|v) + \lambda_1 \hat{p}(w) + \lambda_0$$

  Such that

$$\sum_i \lambda_i = 1$$

- Estimate λ to maximize the likelihood of a held-out corpus (separate from the main training corpus)
- λ can be context-independent or a function of the history

Backing off

- Rather than combining different order models, choose the most appropriate n-gram level
- Probability mass reserved from discounting is then partitioned among lower-order n-grams (and so on, recursively)
- In interpolation always use lower order n-gram information
- In backoff if the trigram counts are above a threshold (eg 1) only use the trigram estimates

Language modelling

- In speech recognition, many word sequences can match the acoustics reasonably well (especially in noisy conditions)
- Can constrain the problem by giving more weight to more probable word sequences
- Combine acoustic model (matching word sequence with the acoustics) with language model (probability of a word sequence).
- Language model: estimate $P(w_1, \ldots, w_n)$ using n-grams
- This component is used in all large vocabulary speech recognition systems

Context-sensitive spelling correction

- Homophones: Their is a house in New Orleans
- Typos: Three is a house in New Orleans
- An n-gram model is likely to have:

$$P(is|there) > P(is|their)$$
$$P(is|there) > P(is|three)$$

- Use this intuition to design a context-sensitive spelling corrector

Summary

- **Reading:** Jurafsky and Martin, chapter 6
- Statistical models of language by directly considering the probabilities of word sequences — n-grams
- The zero probability problem — estimating the probabilities of unseen words and and word sequences
- Discounting and smoothing
- Lots more about n-grams next semester in Empirical Methods in NLP