April 30, 2008

Chapter 6

N-grams; CFG preview

Overview

- N-gram review
- Leftovers: Witten-Bell, hapax legomena
- Backoff
- Entropy and perplexity
- Factored language models
- CFG preview (if time)

N-gram models: Probabilities of sequences of words

- Ideally: Probability of word N, given the presence of words 1 through N-1.
- Approximation: Probability of word N depends only on the last M words.
- Considers only word sequence and not other structure
- We *estimate* probabilities by observing frequencies
- Data sparsity: zero frequency does not entail zero probability

  Address this with smoothing and backoff

Estimating bigram probabilities

- Count bigram occurrences in some corpus, and divide by the bigram frequencies of the first word.

  $P(w_n \mid w_{n-1}) = \frac{C(w_{n-1}w_n)}{C(w_{n-1})}$

- Maximum Likelihood Estimation (MLE): Estimates 'true' probabilities as those that make the training set most likely (but not necessarily any other corpus)

Hapax legomena

- Singular: hapax legomenon
- OED: A word or form of which only one instance is recorded in a literature or an author.
- From Greek for “thing once said”

Witten-Bell Discounting (1/2)

- The probability for zero frequency N-grams is modeled as the probabiilty of seeing an N-gram for the first time.
- The proportion of N-gram tokens that are first occurrences is T (seen types) divided by N (seen tokens).
- The total probability mass to redistribute to unseen N-grams is T/(N+T).

    Demoninator is one event for each token plus one event for each new type

    MLE of a new event occurring

Witten-Bell Discounting (2/2)

- We could divide this redistributed probability mass equally among unseen N-grams.

  Z = number of unseen N gram types

  $P_i^*(unseen) = T/(Z(N+T))$

  $P_1^*(seen) = \text{Count}(i)/(N+T)$

- More common: Do this on a per-history (prefix) basis

  T = number of seen types with a given history (prefix)

  Z = number of unseen types with a given history (prefix)

Backoff

- If we have no information about the frequency of an N-gram, we might still have information about its component N-1 grams.

  $P(w_i \mid \; w_{i-2}, w_{i-1}) \approx \; P(w_i \mid \; w_{i-1})$

  if $\text{Count}(w_{i-2}, w_{i-1}, w_i) = 0$

- But this adds probability mass

- We must discount the probabilities as well, and spread the left-over probability mass to the lower-order N-grams we back off to.

Backoff with Discounting

- Use your preferred discounting scheme to get new p* for the full-length N-grams
- On an (N-1-gram) prefix-by-prefix basis, redistribute the left-over probability mass to lower-order N-grams
- Suppose we want to get a probability for *seven dogs bark*, which does not appear in the training set.
- Find the probability mass left over from discounting *seven dogs X*
- Find what share of that to assign to *dogs bark*

Backoff with Discounting

$$\hat{P}(w_i \mid w_{i-2}, w_{i-1}) = \begin{cases} P(w_i \mid w_{i-2}, w_{i-1}) & \text{if } C(w_{i-2}, w_{i-1}, w_i) > 0 \\ \alpha_1 P(w_i \mid w_{i-1}) & \text{if } C(w_{i-2}, w_{i-1}, w_i) = 0 \\ & \text{and } C(w_{i-1}, w_i) > 0 \\ \alpha_2 P(w_i) & \text{otherwise} \end{cases}$$

Deleted Interpolation

- We can also make use of lower-order N-gram probabilities even if the higher-order counts aren't zero.
- Weight the probabilities from the N-gram, the N-1-gram etc, with weights that sum to 1.
- Train weights from a held-out set of data (dev set)

  Why?
- Train weights on a prefix-by-prefix basis

Random Variables

- Random Variable: A function that maps events onto numbers
- We might have a random variable X, which ranges over the results of flipping a coint, and maps "heads" to 1 and "tails" to 0.
- This is a convenient way to talk about the set of possibilities.
- A model assigns a probability to each value of X
- Notational shortcut $p(x) = P(X = x)$

Entropy (1/4)

- A measure of the information of a probability distribution (in bits): How surprising is each event?
  $$H(X) = -\sum_x p(x)\log_2 p(x)$$
- The number of bits, on average, needed to encode a value of X
- The entropy of a fair eight-sided die is
  $$-(8 \times\ .125 \times\ \log .125) = -\log .125 = 3\text{bits}$$

Entropy (2/4)

- The entropy of an unfair die with the distribution

  $\{\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{64}, \frac{1}{64}, \frac{1}{64}, \frac{1}{64}\}$

  is

  $-\frac{1}{2}\log\frac{1}{2} - \frac{1}{4}\log\frac{1}{4} - \frac{1}{8}\log\frac{1}{8} - \frac{1}{16}\log\frac{1}{16} - \frac{1}{64}\log\frac{1}{64} - \frac{1}{64}\log\frac{1}{64} - \frac{1}{64}\log\frac{1}{64} - \frac{1}{64}\log\frac{1}{64} = 2\text{bits}$

- Why does the fair die have higher entropy?

Entropy (3/4)

- In general, smooth distributions have higher entropy than lumpy ones
- Pointwise entropy: How surprised our model is at seeing the next word in a sequence = $-\log m(w|h)$
- The average of this for a long sequence of words, generated according to the probability distribution p is the cross-entropy of m and p, written as $H(p, m)$

Entropy (4/4)

- Better models have lower cross-entropy
- $H(p, m)$ is an upper bound on $H(p)$ ($H(p) \leq H(p, m)$)
- You don't need to know p to compare the entropy of different m!

  You just need some data generated by p.

Perplexity

- Models are generally evaluated using *perplexity* rather than cross-entropy.
- Preplexity $= 2^H$
- A measure of 'surprise': on average, we are as surprised as we would be if we had to choose between 2^H possibilities.
- Better models have lower perplexity

Interim summary

- Backoff is a technique for using lower-order N-grams to fill in when the higher-order ones are unattested
- To keep things summing to 1, backoff is combined with discounting
- Entropy is a measure of the amount of information contained in a signal
- Cross-entropy allows us to compare models on how well they approximate the "true" probability distribution
- In comparing models, always use held-out data

N-grams and typological variation

- English is likely to be more n-gram friendly than average

    Relatively fixed word order

    Relatively simple morphology (low lemma:wordform ratio)

- Alternatives (for English and other languages)

    Dependency n-grams

    Factored language models (Bilmes & Kirchhoff 2003)

Overview

- N-gram review
- Leftovers: Witten-Bell, hapax legomena
- Backoff
- Entropy and perplexity
- Factored language models
- CFG preview (if time)