Intro to Natural Language Processing

- Last class
  - Smoothing
    - Add-one (Laplacian)
    - Good-Turing
  - Unknown words
- Today:
  - Evaluating n-gram models
  - Combining estimators
    - (Deleted) interpolation
    - Backoff
  - HMM's for p-o-s tagging

Evaluating n-gram models

- Best way: extrinsic evaluation
  - Embed in an application and measure the total performance of the application
  - End-to-end evaluation
- Intrinsic evaluation
  - Measure quality of the model independent of any application
  - *Perplexity*
    - *Intuition: the better model is the one that has a tighter fit to the test data or that better predicts the test data*

Perplexity

For a test set $W = w_1 \, w_2 \ldots w_{N,}$

$$PP(W) = P(w_1 \, w_2 \ldots w_{N})^{-1/N}$$

The higher the conditional probability of the word sequence, the **lower** the perplexity.

Must be computed with models that have no knowledge of the test set.

- Next
  - Combining estimators
    - Interpolation
    - Backoff (won't cover this in detail…)

Combining estimators

- Smoothing methods
  - Provide the same estimate for all unseen (or rare) n-grams
  - Make use only of the raw frequency of an n-gram
- But there is an additional source of knowledge we can draw on --- the n-gram “hierarchy”
  - If there are no examples of a particular trigram, $w_{n-2}w_{n-1}w_n$, to compute $P(w_n|w_{n-2}w_{n-1})$, we can estimate its probability by using the bigram probability $P(w_n|w_{n-1})$.
  - If there are no examples of the bigram to compute $P(w_n|w_{n-1})$, we can use the unigram probability $P(w_n)$.
- For n-gram models, suitably combining various models of different orders is the secret to success.

Simple linear interpolation

- Construct a linear combination of the multiple probability estimates.
  - Weight each contribution so that the result is another probability function.

$$P(w_n \mid w_{n-2}w_{n-1}) = \lambda_3 P(w_n \mid w_{n-2}w_{n-1}) + \lambda_2 P(w_n \mid w_{n-1}) + \lambda_1 P(w_n)$$

  - Lambda’s sum to 1.
- Also known as (finite) *mixture models*
- *Deleted* interpolation
  - Each lambda is a function of the most discriminating context

Backoff (Katz 1987)

- Non-linear method
- The estimate for an n-gram is allowed to back off through progressively shorter histories.
- The most detailed model that can provide sufficiently reliable information about the current context is used.
- Trigram version (first try):

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

Final words…

- When smoothing, we usually ignore counts of 1
- Problems with backoff?
  - Probability estimates can change suddenly on adding more data when the back-off algorithm selects a different order of n-gram model on which to base the estimate.
  - Works well in practice.
- Good option: simple linear interpolation with MLE n-gram estimates plus some allowance for unseen words (e.g. Good-Turing discounting)