Indian Institute of Technology Kharagpur
Department of Computer Science & Engineering

CS60075
Natural Language Processing
Autumn 2020

Lecture 2B : Smoothing

Sep 10 2020

Sudeshna Sarkar

Readings

- Chapter 3 of Jurafsky & Martin (3^{rd} Ed)

https://web.stanford.edu/~jurafsky/slp3/3.pdf

- Chapter 6 of Eisenstein

https://github.com/jacobeisenstein/gt-nlp-class/blob/master/notes/eisenstein-nlp-notes.pdf

Unknown Words

- How to handle ***out of vocabulary*** (OOV) words?

1. Train a model that includes an explicit symbol for an unknown word (<UNK>)
    - Choose a vocabulary in advance and replace other words in the training corpus with **<UNK>.**
    - Replace the first occurrence of each word in the training data with <UNK>.

2. Character based models

Sample Perplexity Evaluation

- Models trained on 38 million words from the Wall Street Journal (WSJ) using a 19,979 word vocabulary.
- Evaluate on a disjoint set of 1.5 million WSJ words.

	Unigram	Bigram	Trigram
Perplexity	962	170	109

Empirical Observations

- A small number of events occur with high frequency
- A large number of events occur with low frequency
- Some of the zeroes in the table are low frequency events you haven't seen yet.
- Words follow a Zipfian distribution
  - Small number of words occur very frequently
  - A large number are seen only once
- Zipf’s law: a word’s frequency is approximately inversely proportional to its rank in the word distribution list

Smoothing

- Many rare (but not impossible) combinations never occur in training, so MLE incorrectly assigns zero to many parameters (***sparse data***).
- If a new combination occurs during testing, it is given a probability of zero and the entire sequence gets a probability of zero (i.e. infinite perplexity).
- In practice, parameters are ***smoothed*** (or ***regularized***) to reassign some probability mass to unseen events.
  - Adding probability mass to unseen events requires removing it from seen ones (***discounting***) in order to maintain a joint distribution that sums to 1.

Laplace (Add-One) Smoothing

- "Hallucinate" additional training data in which each possible N-gram occurs exactly once and adjust estimates.

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

N-gram: $$P(w_n \mid w_{n-N+1}^{n-1}) = \frac{C(w_{n-N+1}^{n-1}w_n)+1}{C(w_{n-N+1}^{n-1})+V}$$

V: the total number of possible (N–1)-grams (i.e. the vocabulary size for a bigram (n-gram) model).

- Tends to reassign too much mass to unseen events, so can be adjusted to add δ

Advanced Smoothing

- Improved smoothing for language models.
  - Interpolation
  - Backoff
  - Kneser-Ney
  - Class-based (cluster) N-grams

Model Combination

- As N increases, the power (expressiveness) of an N-gram model increases
  - *but* the ability to estimate accurate parameters from sparse data decreases

- A general approach is to combine the results of multiple N-gram models of increasing complexity (i.e. increasing N).

Interpolation

- Linearly combine estimates of N-gram models of increasing order.

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

- Learn proper values for λ_i by training to (approximately) maximize the likelihood of an independent *development* corpus.

Backoff

- Only use lower-order model when data for higher-order model is unavailable.
- Recursively back-off to weaker models until data is available.

$$P_{katz}(w_n \mid w_{n-N+1}^{n-1}) = \begin{cases} P^*(w_n \mid w_{n-N+1}^{n-1}) & \text{if } C(w_{n-N+1}^{n}) > 1 \\ \alpha(w_{n-N+1}^{n-1}) P_{katz}(w_n \mid w_{n-N+2}^{n-1}) & \text{otherwise} \end{cases}$$

- P* is a discounted probability estimate to reserve mass for unseen events and α's are back-off weights.

A Problem for N-Grams:
Long Distance Dependencies

- Syntactic dependencies
  - "The ***man*** next to the large oak tree near the grocery store on the corner **is** tall."
  - "The ***men*** next to the large oak tree near the grocery store on the corner **are** tall."

- Semantic dependencies
  - "The ***bird*** next to the large oak tree near the grocery store on the corner **flies** rapidly."
  - "The ***man*** next to the large oak tree near the grocery store on the corner **talks** rapidly."

Neural language model

$$P(w_t | w_{t-n}, \dots, w_{t-1}) = \cancel{\frac{C(w_{t-n}, \dots, w_t)}{C(w_{t-n}, \dots, w_{t-1})}} = f_\theta(w_{t-n}, \dots, w_{t-1})$$

- Parametric estimator
- We need numerical representation of words