NLP

Introduction to NLP

Language models (Part 2)

Smoothing

- If the vocabulary size is $|V|=1M$
  - Too many parameters to estimate even a unigram model
  - MLE assigns values of 0 to unseen (yet not impossible) data
  - Let alone bigram or trigram models
- Smoothing (regularization)
  - Reassigning some probability mass to unseen data

Smoothing

- How to model novel words?
  - Or novel bigrams?
- Distributing some of the probability mass to allow for novel events
- Add-one (Laplace) smoothing:
  - Bigrams: $P(w_i|w_{i-1}) = (c(w_{i-1},w_i)+1)/(c(w_{i-1})+V)$
  - This method reassigns too much probability mass to unseen events
- Possible to do add-k instead of add-one
- Both of these don't work well in practice

Advanced Smoothing

- Good-Turing
  - Try to predict the probabilities of unseen events based on the probabilities of seen events
- Kneser-Ney
- Class-based n-grams

Example

- Corpus:
  - cat dog cat rabbit mouse fish fish mouse hamster hamster fish turtle tiger cat rabbit cat dog dog fox lion
- What is the probability the next item is “mouse”?
  - P_{MLE} (mouse) = 2/20
- What is the probability the next item is “elephant” or some other previously unseen animal?
  - Trickier
  - Is it 0/20?
  - Note that P (that the next animal is unseen) > 0
  - Therefore we need to discount the probabilities of the animals that have already been seen
  - P_{MLE} (mouse) < 2/20

Good Turing

- Actual counts c
- N_r = number of n-grams that occur exactly c times in the corpus
- N_0 = total number of n-grams in the corpus
- Revised counts c*
  - $c^* = (c+1)\, N_{c+1}/N_c$

Example

- Corpus:
  - cat dog cat rabbit mouse fish fish mouse hamster hamster fish turtle tiger cat rabbit cat dog dog fox lion
- Counts
  - C(cat) = 4
  - C(dog) = 3
  - C(fish) = 3
  - C(mouse) = 2
  - C(rabbit) = 2
  - C(hamster) = 2
  - C(fox) = 1
  - C(turtle) = 1
  - C(tiger) = 1
  - C(lion) = 1
- N1=4, N2=3, N3=2, N4=1

Example (cont'd)

- N1=4, N2=3, N3=2, N4=1
- Revised counts $c^* = (c+1)\, N_{c+1}/N_c$
  - C*(cat) = 4
  - C*(dog) = (3+1) x 1/2 = 2
  - C*(mouse) = (2+1) x 2/3 = 2
  - C*(rabbit) = (2+1) x 2/3 = 2
  - C*(hamster) = (2+1) x 2/3 = 2
  - C*(fox) = (1+1) x 3/4 = 6/4
  - C*(turtle) = (1+1) x 3/4 = 6/4
  - C*(tiger) = (1+1) x 3/4 = 6/4
  - C*(lion) = (1+1) x 3/4 = 6/4
  - C*(elephant) = N1/N = 4/20
- Note that these counts don't necessarily add to 1, so they still need to be normalized.
  - P*(lion) = 6/4 / 20 = 6/80

Dealing with Sparse Data

- Two main techniques used
  - Backoff
  - Interpolation

Backoff

- Going back to the lower-order n-gram model if the higher-order model is sparse (e.g., frequency >= 1)
- Learning the parameters
  - From a development data set

Interpolation

- If $P'(w_i|w_{i-1},w_{i-2})$ is sparse:
  - Use $\lambda_1 P'(w_i|w_{i-1},w_{i-2})+\lambda_2 P'(w_i|w_{i-1})+\lambda_3 P'(w_i)$
- Better than backoff
- See [Chen and Goodman 1998] for more details

NLP