Probability and Information Theory for Language Modeling

- Statistical vs. Symbolic NLP
- Elementary Probability Theory
- Language Modeling
- Information Theory

Statistical Linguistics

- Statistical approaches are clearly useful for engineering tasks.
- Are statistical approaches appropriate for scientific study?
  - Language Acquisition
  - Language Change
  - Language Variation

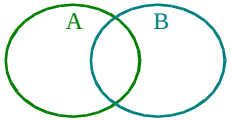
Statistical Linguistics: Adult Monolingual Speaker

- Error tolerance
- Language comprehension:
  - An average sentence has a huge number of "possible" syntactic structures.
  - Defining which analysis is correct is not a computational problem.
- Possible Counter:
  - These are "performance issues"
  - But note: "performance issues" is not the same as "computational issues"

Elementary Probability Theory

- Terminology:
  - Experiment ⇒ a repeatable process
  - Sample ⇒ a possible outcome
  - Sample space ⇒ all samples for an experiment
  - Event ⇒ a set of samples
  - Probability distribution ⇒ assigns a probability to each sample.
  - P(A) ⇒ $\sum_{x \in A} P(x)$
  - Uniform Distribution ⇒ All samples are equi-probable.

Elementary Probability Theory (2)



- Prior probability of event *A* is *P(A)*
- We are told that *B* is true
- Now the probability of *A* is *P(A and B)/P(B)*
- This is the conditional probability of A given B:
  - P(A|B) = P(A∩B)/P(B)

Elementary Probability Theory (3)

- Multiplication Rule:
  - P(A∩B) = P(A|B)P(B) = P(B|A)P(A)
- Bayes rule:
  - P(A|B) = P(B|A)P(A)/P(B)
- Independance:
  - P(A|B) = P(A)
  - P(B|A) = P(B)
  - P(A∩B) = P(A)P(B)

Bayes Theorem: Example

- Event A = a sentence contains a certain linguistic construction
  - P(A) = 0.001
- Event B = our program reports that it found the linguistic construction
  - P(B|A) = 0.9 (true positive)
  - P(B|¬A) = 0.1 (false positive)
- Suppose the program finds the construction. What is the chance that it is correct?
  - P(A|B) = P(B|A)P(A)/P(B)

Language Modeling

- Goal: Find the probability of a "text"
  - "text" can be a word, an utterance, a document, etc.
- Texts are generated by an *unknown* probability distribution
- Use language model to capture *a priori* information about the likelihood of a text.
  - We are more likely to predict a text with a higher *a priori* probability.

Statistical Inference

- Use training data to make inferences about the unknown distribution.
- Define a class of candidate probability distributions.
  - Divide texts into equivalance classes.
  - Examples: $P(w) = P(w.type)$
    $P(w_i) = P(w_i.type|w_{i-1}.type)$
- Select the "best" candidate
  - Use training data to evaluate each candidate probability distribution.

Why Do Language Modeling?

- Speech recognition
  - Predict word sequences that are more likely.
- Spelling correction
  - Suggest words that are more likely.
- Machine translation
  - Suggest translations that are more likely.
- Generation
  - Language models help us generate "likely" sentences.

Output Forms

- Each text generates an *output form*, which we can directly observe.
  - Speech recognition: a sequence of sounds
  - Spelling correction: a sequence of characters
  - Machine translation: a source language text
- Texts and their output forms are generated by an unknown distribution:
  $P(text, output)$
- Find the most likely text for an output form:
  $argmax_{text} P(text|output)$

Finding the Source Text

- Bayes Rule:

$$P(text|output) = \frac{\overbrace{P(text)}^{\text{Language Model}} P(output|text)}{P(output)}$$

- Recovering the underlying form:

$$argmax_{text} P(text|output) =$$

$$argmax_{text} \frac{P(text)P(output|text)}{P(output)} =$$

$$argmax_{text} P(text)P(output|text)$$

Divide & Conquor

- *P(output)* has a very large sample space.
  - We can not directly model *P(output)*.
- Reduce the problem of modeling *P(output)* to simpler estimation problems, whose solutions can be combined.
  - Estimate the probability of a text one word at a time:

$$P(w_1, w_2, \ldots, w_n) = \prod_{i=1}^{n} P(w_i | w_1, \ldots, w_{i-1})$$

Equivalance Classes

- $P(w_n | w_1, \ldots, w_{n-1})$ has a large sample space.
- Divide $P(w_n | w_1, \ldots, w_{n-1})$ into *equivalance classes*
  - Example: $P(w_n | w_1, \ldots, w_{n-1}) \cong P(w_n | w_{n-1})$
- Estimate the probability of each equivalance class.
  - Count the number of training instances in each equivalance class.
  - Use these counts to estimate the probability for each equivalance class.

Maximum Likelihood Estimation

- Predict the probability of an equivalance class using its *relative frequency* in the training data:

$$P(x) = \frac{C(x)}{N}$$

  - C(x) = frequency of x in the training data
  - N = number of training instances

Problems with MLE

- Underestimates the probability for unseen data
  - *C(x)=0*
  - Maybe we just didn't have enough training data.
- Overestimates the probability for rare data
  - *C(x)=1*
  - Estimates based on one training sample are unreliable.

Statistical Estimators

- Use the training data to form a more advanced estimate of P(x)
- Laplace, Lidstone, and ELE:
  - Reserve a small portion of the probability distribution for unseen data.
  - See Manning pp. 202-205
- Held out estimation:
  - Use a small amount of held-out data to decide the probability of unseen data.

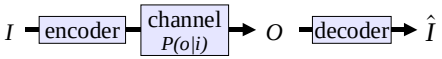
Statistical Estimators: nltk

- *nltk.probability.ProbDistI* defines an interface for probability distributions.
- Probability distributions are typically constructed from frequency distributions:

```
>>> pdist = ELEProbDist(fdist)
>>> print pdist.prob('the')
0.02
>>> print pdist.cond_prob('the', SetEvent('the', 'a'))
0.6
```

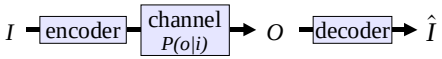
- See the nltk reference documentation for more information.

Noisy Channel Model



- Channel introduces "noise."
- Task: optimize *throughput* and *accuracy*
  - More redundancy -> More accuracy
  - Less redundancy -> More throughput
- Many statistical NLP problems can be thought of as *decoding problems*.
  - No control over encoding

Noisy Channel Model (2)



- Example: optical character recognition
  - *i* = actual text
  - *o* = text with mistakes
  - *P(i)* = language model
  - *P(o|i)* = model of OCR errors
- See Manning Table 2.2 (p. 71)

Entropy

- The *information content* of a probability distribution.
- The average length of the message needed to transmit the outcome.

$$H(X) = -\sum_{x} \left(P(x) \log P(x) \right)$$

- Example: fair 8-sided die

$$H(X) = -\sum_{x=1}^{8} \left(P(x) \log P(x) \right) = -\sum_{x=1}^{8} \left(\frac{1}{8} \log \frac{1}{8} \right)$$

$$= -8 \left(\frac{1}{8} \log \frac{1}{8} \right) = -\log \frac{1}{8} = 3$$

Example: Simplified Polynesian

Letter	p	t	k	a	i	u
P(Letter)	1/8	1/4	1/8	1/4	1/8	1/8

- Per-letter entropy:

$$H(X) = -\sum \left(P(x) \log P(x) \right)$$

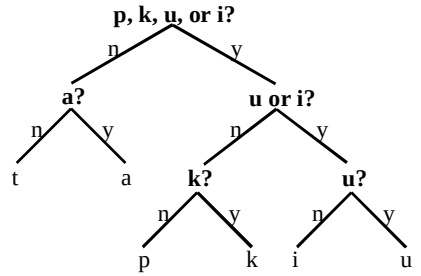
$$= -\left(4 \times \frac{1}{8} \log \frac{1}{8} \; plus \; 2 \times \frac{1}{4} \log \frac{1}{4} \right)$$

$$= 2.5 bits$$

Optimal Binary Encodings

- On average, how many yes/no questions do you need to ask to find the outcome?
  - Example questions:
    - Is the letter *t* or *a*?
    - Is the letter a consonant?
- Binary encoding: each bit is a yes/no question.
- Entropy is the average message length, using this encoding.

Example: Simplified Polynesian (2)



- Optimal encoding for Simplified Polynesian:

Letter	p	t	k	a	i	u
Encoding	100	00	101	01	110	111

Relative Entropy

- **Relative entropy measures the difference between two probability distributions.**

$$D(p\|q) = \sum_{x \in X} p(x) \log \frac{p(x)}{q(x)}$$

  - **If we encode the actual data using the language model, how many *extra bits* do we use, on average?**
- **Use relative entropy to measure performance:**
  - *p* = **actual distribution (from test data)**
  - *q* = **language model**
  - **performance** = *D(p||q)*

Readings

- **Manning 1, 2**
- **Abney #1**