NLP Week 2
Fun with Counting Words

Colin Cherry
Who is this guy?

- Dekang Lin’s PhD student
- From Dartmouth, Nova Scotia
- Works in Machine Translation
- One of two “stats people” in the group
- Favors empirical techniques that use linguistics for an edge
My job for this week:

- Prep you for the assignment
- Talk about Machine Translation, given time
Outline

- Why I study NLP (and you should)
- Working with corpora
  - Tokenization
  - Counting stuff
- Probability & Information Theory basics
Why study NLP?

NLP is:
- Needed
- Profitable
- Possible
- Interesting
NLP is Needed

- A lot of information is stored digitally, written in an unannotated human language

- This information is trapped without
  - A) Human intervention and interpretation
  - B) NLP

- We don’t have the time / money for a human to do it all for us
Example: How does Rotten Tomatoes work?

- *The Transporter* was downright adorable, a *Maxim*-ized cocktail of martial arts, stunt driving, and curiously circumspect sex -- nothing you could get roaring drunk on, more a predinner aperitif with which to toast how absurdly easy it is to entertain boys and young men 13 to 24 years of age.
Other examples

- Research / Medical abstracts
- Customer comments submitted by email
- Company manuals
- Captured documents
- Just plain old text
NLP is Profitable

- The companies that hold this data want to be able to get at it / manipulate it
- Other companies provide services helping others sift through data
  - Google
  - Yahoo
- Others still overcome the competition with neat features like grammar checkers
  - Microsoft
NLP is Possible

- By the end of the course, you should believe that we have enough:
  - Data
  - Stats
  - Linguistics
to do some really cool stuff
- And we don’t have to “solve AI” first
NLP is Interesting

- Multidisciplinary
  - Meet some really neat people
  - Learn some stuff outside of CS

- Often unsupervised
  - Wow yourself with how much a computer can learn just from text
Outline

- Why I study NLP (and you should)
  - Working with corpora
    - Tokenization
    - Counting stuff
  - Probability & Information theory basics
Working with corpora

- Corpus (corpora) = large collection(s) of text

- Different types:
  - Sources
  - Annotated / unannotated

- Statistical techniques are useful only because of corpora
Fun stuff you can do with an unannotated corpus:

- Learn a part of speech tagger
  - The <D> dog <N> jumps <V> high<A>
Fun stuff you can do with an unannotated corpus:

- Learn a part of speech tagger
- Learn word similarity
  - ‘Coke’ is similar to ‘Pepsi’
Fun stuff you can do with an unannotated corpus:

- Learn a part of speech tagger
- Learn word similarity
- Learn word clusters
  - ‘Coke’ is in a softdrink cluster, but also a company cluster and a drug cluster
Fun stuff you can do with an unannotated corpus:

- Learn a part of speech tagger
- Learn word similarity
- Learn word clusters
- Learn a paraphraser
  - X wrote Y == X is the author of Y
Fun stuff you can do with an unannotated corpus:

- Learn a part of speech tagger
- Learn word similarity
- Learn word clusters
- Learn a paraphraser
- Learn a pronoun resolver
  - He (John) likes his (John’s) car.
First, you have to tokenize

- A **token** is the smallest unit of text you’re interested in working with
- Tokenization separates text into tokens

  Luckily, he earned $2.50 doing yo-yo tricks. becomes:

  Luckily, he earned $2.50 doing yo-yo tricks.
Tokenization Approach

- Approach it just like you would a compiler tokenizer
  - Regular expression interpreters
  - Finite state machines
- Might need to make >1 pass
- Otherwise, traditional techniques work
Some concerns with tokenizing English: Periods

- Always separate a period from word that precedes it, unless it’s not followed by a space

- We would get U.S.A\. and etc\.  

- It’s hard to do perfectly unless we also solve the sentence boundary problem
  - Which is its own, non-trivial problem
Some concerns with tokenizing English: Apostrophes

- How do you want to handle contractions?
  - Don’t => do’n’t
  - Don’t => don’t

- What about I’m?
  - Imagine trying to parse “I’m right” with two tokens

- Also, what about possessive?
  - Dog’s = Dog’s or Dog’s?
  - What about Dogs’ (plural possessive)?
(Tokenization when you can’t count on white-space)

- New York, Nova Scotia, data base
  - Shouldn’t these be single tokens?

- Chinese
  - Has no spaces
  - One approach is statistical processing - similar to phrase discovery in English

- Tokenization is often task-specific
Outline

- Why I study NLP (and you should)
  - Working with corpora
    - Tokenization
    - Counting stuff
- Probability & Information theory basics
Let’s start counting stuff!

■ What do we want to count?
■ Let’s start with words:
  – **Token**: an individual occurrence of a word
  – **Type**: the word itself

“Bernie the dog likes swimming in the lake” has 8 tokens, but only 7 types (the type ‘the’ occurs twice)
What makes two tokens the same type?

- Do we want to reduce all words to lowercase so ‘The’ == ‘the’?
  - Depends on the task
    - sparsity vrs coarsness
  - Happy medium: lowercase all words that begin sentences or are in titles

- What about ‘sing’ and ‘sings’
  - Do we want to stem to the lemma or keep the word form?
A little more terminology

- If we count how often a given type occurs in our corpus, we’ll call that a **unigram** count:
  - ‘the’ occurs 311,818 times
  - ‘elmo’ occurs 1 time
A little more terminology

- If we count how often a given type occurs in our corpus, we’ll call that a **unigram** count:
  - ‘the’ occurs 311,818 times
  - ‘elmo’ occurs 1 time
- Pairs of types will be called **bigrams**
- Triplets of types will be **trigrams**
- Then 4-grams, 5-grams and so on…
Looking at those counts

- Types in text are distributed unevenly
- Some types will occur very frequently
  - In a large corpus (80,000 types)
  - ‘the’ occurs 300,000 times
  - ‘of’ occurs 150,000 times
  - ‘waffles’ occurs once
- 35,000 other types also occur only once
Zipf’s Law

- Zipf observed this:
  - Let $r$ be the rank of a type by frequency
  - Let $f$ be the type’s frequency

$$f \propto \frac{1}{r}$$

or

$$f \cdot r = k$$

- Tied to “Principle of Least Effort”
Consequences

- The most frequent word will occur twice as much as the second most frequent
- The 50th most frequent word will occur 3 times more often than the 150th word
- Isn’t perfect - in fact, it’s a long way off
- Important intuition:
  - You will have a small number of high frequency types, and a huge number of low frequency types
  - Almost all types are rare
Outline

- Why I study NLP (and you should)
  - Working with corpora
    - Tokenization
    - Counting stuff
  - Probability & Information theory basics
Some quick Probability basics

- $p(x)$ is the probability of an event $x$ drawn from some universe of events.

- $p(x,y)$ is the probability of two events $x$ and $y$ occurring at the same time
  - Called joint probability

- If the events $x$ and $y$ are independent:
  - $p(x,y) = p(x) * p(y)$
Conditional Probabilities

$$p(x \mid y) = \frac{p(x, y)}{p(x)}$$

- How likely is $x$ once we know $y$ has happened?
- If $x$ and $y$ are independent:
  - $p(x \mid y) = p(x)$
Information Theory

- Entropy measures the information in a random variable
  \[
  H(X) = \sum_{x \in X} p(x) \log \frac{1}{p(x)}
  \]
- More random = more entropy
- Also the average length of a message containing the outcome of the variable
Conditional Entropy & Mutual Information

\[ H(Y \mid X) = \sum_{x \in X} \sum_{y \in Y} p(x, y) \log \frac{1}{p(y \mid x)} \]

Mut. info: \( I(Y; X) \) is the reduction of uncertainty in \( Y \) due to \( X \):

\[ I(Y; X) = H(Y) - H(Y \mid X) \]

\[ = \sum_{x \in X} \sum_{y \in Y} p(x, y) \log \frac{p(x, y)}{p(x)p(y)} \]
Outline

- Why I study NLP (and you should)
  - Working with corpora
    - Tokenization
    - Counting stuff
  - Probability & Information theory basics
Next lecture: Collocations

- We put simple corpus-based counting together with a few easy statistical measures and we (kind of) get a phrase finder!