1 Admistrivia

• Who are you? Please answer the following questions
  – Name, program, degree
  – Background in stat
  – Background in linguistics
  – What did you hear this class was about? (Since it wasn’t announced, I’m curious what the network of students has produced for a course description)

• Readings you should be doing on your own as we go along
  – Book: JM = *Speech and Language processing* by Jurafsky and Martin.
– Foundations of statistical natural language processing by Manning and Schutze. (Suggested if you have a weaker statistics background.)

– Language log (Liberman’s blog)

• Lots of homework: mostly R, some math.

• Final project: choice of two

  – Paper replication: Analysis and write up and present some linguistic data. If you have a particular domain in mind, talk to me soon and we can start connecting you with data. Basically you should mimic some existing paper fairly carefully—but say with new data. semester.

  – Two “language log” blog posts and one appendix for one of the posts. You can replicate one of his posts (clearly one with data) if you like. The other should be original (or better still, both can be original). For one of the two posts, write up a technical appendix looking at the data more statistically than would be meaningful for language log.

Let me know which you plan on doing. If the language log, you should do your first one before Nov 1st and your second one before Dec 12th.
2 Introduction: Example (Flu)

- Classic epidemiology: count number of cases doctors report for flu each week.
  - Model speed of reporting (different for different areas)
  - Predict growth to make up for delay in data
  - Make forecast of current level of flu

- Google Flu:
  - Take searches for “flu” as ground truth
  - Look for phrases that correlate with it: runny nose, etc
  - Count the number of searches for these terms
  - Don’t bother with delay since you have daily/hourly searches

3 Introduction: General cases

- Medical: machine reading of reports.

- Credit cards: Machine reading bills and predicting fraud / bankruptcy.

- Biology: Machine reading of abstracts to find which genes are discussed in the article.

- Education: Machine reading of course evaluations.
• Teaching: Automatic writing of questions for reading comprehension. OK, this maybe more statistical use in linguistics rather than the other way around. So let’s turn to those.

• Call center ("you are on a recorded line.")

• Call handling (google information)

4 Statistics in linguistics

• Best voice recognition are based on probabilistic models

• Best POS taggers are statistical

• Parsing use statistics now adays

• Word disambiguation / word meaning are statistical

• We are in the “Rise of Machine learning” era according to JM.

5 Why now?

• Chomsky didn’t use data

• In 70s/80s, people might train systems on 50k words. Similar to expecting a baby to talk at age, 5 hours.

• Now we can process a billion words. Close to a lifetime’s worth. So we can start finding the same patterns that humans can encrypt in language.
6 The bright and beautiful future

- In 1951 Turing wrote his piece on smart computers. He said in 50 years we would have smart computers. Hence the title of the Arthur C Clark book: 2001: A space Odyssey. “I’m sorry Dave, I’m afraid I can’t do that.”

- Understand text is the final problem in science: once it can be solved, computers will start to be really smart. (Suppose you had the time to read the entire Wikipedia and understand the whole thing. That is the starting point of any intelligent computer.)

- Turing guess was a shot in the dark. Now we have better estimates of human memory: say about $10^{15}$ bits, or $10^{14}$ bytes, or 100 Terabytes (Or TibiBytes for the purists). You can buy Terabyte disks now. A terabyte of memory would be about one thousand dollars (2009 estimate).

- In 20 years, you will have about 100 Terabyte computers on your desk. In 25 years, your “ipod” will have that much memory.

- So, peg the singularity at 2029 for a first guess.
Unix tools: Finding the most common words in Alice in Wonderland

The following is a classic text for unix tools for linguistic analysis:

Let’s figure out some statistics for Alice in Wonderland.

- We can look at the first few words:
  
  head <alice_in_wonderland.txt

- Or the last few
  
  tail <alice_in_wonderland.txt

- Or see more:
  
  tail -100 <alice_in_wonderland.txt

- Using the tr command (use man tr to see help).
  
  tr " " "#" alice_in_wonderland.txt

- Yikes that is a lot of output
  
  tr " " "#" alice_in_wonderland.txt |tail -100

- Ok, one word per line
Now we can sort it and use uniq -c to get word frequency:

```
tr " " "\n" alice_in_wonderland.txt |sort | uniq -c | tail -100
```

And for the most common words:

```
tr " " "\n" alice_in_wonderland.txt |sort | uniq -c | tail -100
```

### 8 How about bi-grams?

We need two more tricks. A “lag” operator, namely tail --lines=+2 and a paste command to glue two files together. This leads to the following command:

```
paste <(tr " " "\n" < alice_in_wonderland.txt) \
   <(tr " " "\n" <alice_in_wonderland.txt|tail --lines=+2) \
   |sort |uniq -c |sort -n |tail -100
```

This uses some cute stuff from zsh. We could do it more simply with the following:

```
tr " " "\n" < alice_in_wonderland.txt > word_per_line.tmp
tail --lines=+2 word_per_line.tmp > lag_word_per_line.tmp
paste word_per_line.tmp lag_word_per_line.tmp > bigrams.tmp
sort bigrams.tmp | uniq -c | sort -n > frequent_bigrams
rm word_per_line.tmp
```
rm lag_word_per_line.tmp
rm bigrams.tmp
tail - 100 frequent_bigrams