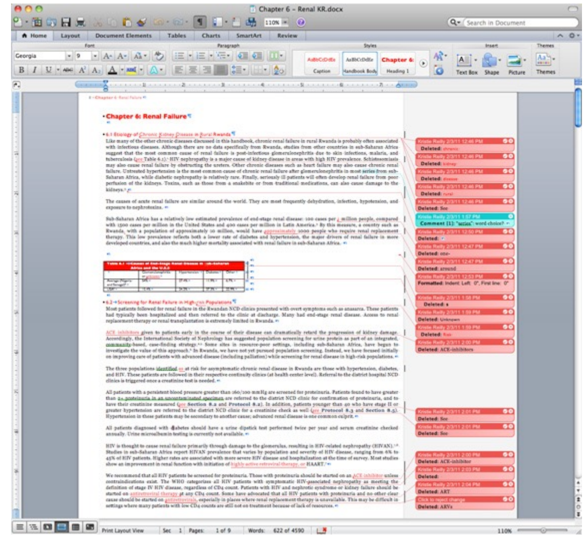
Homework

- Please leave lab computers on (but log out)
- On github.com
  - Class materials at http://github.com/Ling583/notes
  - Read *Speech and Language Processing* Chapter 2: "Regular expressions, text normalization, edit distance"
- On datacamp.com
  - Finish *Intro to Python for Data Science* and *Introduction to Shell for Data Science,* and *Intermediate Python for Data Science*
- Turn in text collection assignment

Revision control



Revision control

- SCCS, RCS
  - One central repository shared by all users
  - Revision history
  - Check-in/check-out file access

Revision control

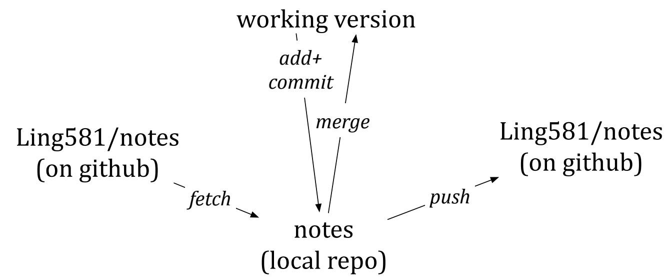
- CVS, SVN
  - One central repository
  - Each user has their own working version
  - Merge and commit
  - Conflict detection

Revision control

- git, mercurial
  - Each user has their own repository and working version
  - Distributed VCS: clone, push, fetch, pull
  - Bare repos
  - Fork, pull requests

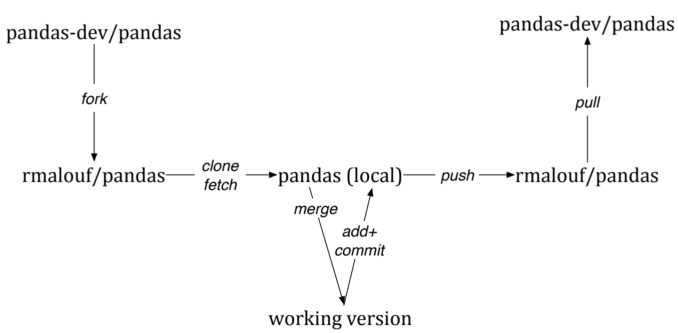
Revision control

- GitHub workflow



Revision control

- GitHub workflow

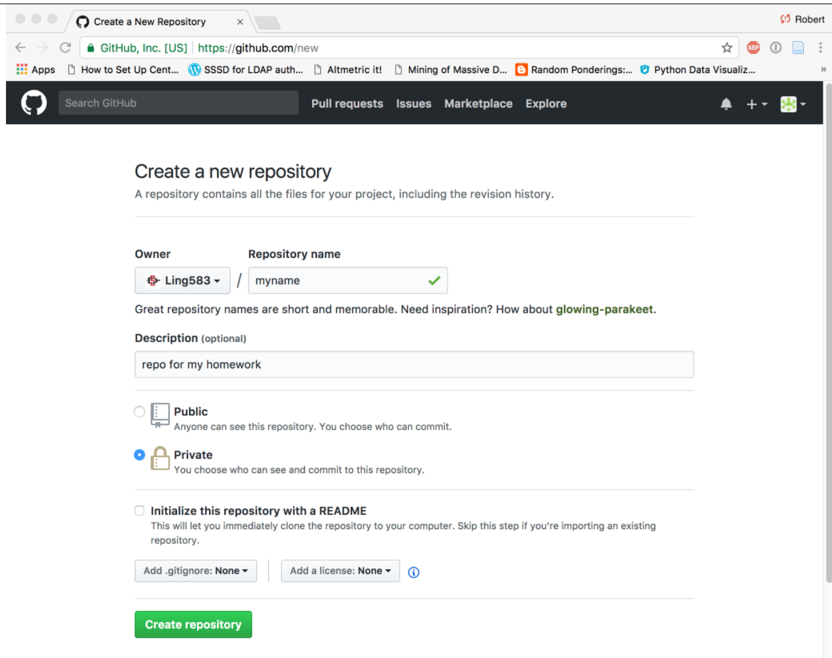


GitHub

- Configure local git (only need to do this once)

```
git config --global user.name “My Name"
git config --global user.email my.name@example.com
git config --global push.default simple
```

- Create GitHub repo for yourself (also only need to do this once)



GitHub

- Make a local copy of your repo

```
git clone https://github.com/Ling583/myname.git
```

- Commit changes from working copy to local repo

```
git add .
git commit -m "turn in hw 1"
```

- Send changes from local repo to GitHub

```
git push
```

- Get changes from GitHub to local repo and working copy

```
git pull
```

Homework

- Include notebooks, other code, and data files
- Each notebook should have a title, description, and your name at the top
- Notebooks should also include text describing that you did and why, and also interpreting your results
- For the text assignment, put all files in a sub-folder called hw1

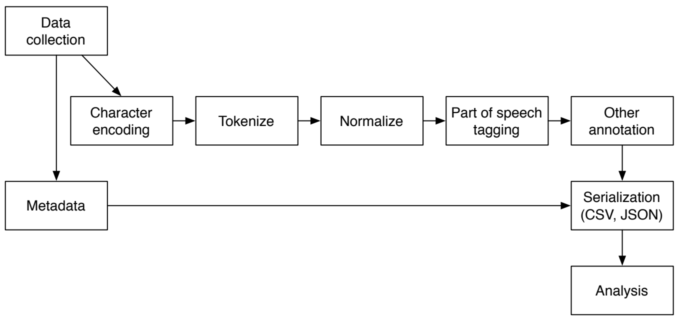
Homework

- Keywords for top 10 commenters

```
visarga : starspace, simulators, relational, relations, eating, automata, fei, simulator, sim
ulation, voices

ajmooch : ajbrock, freezeout, batchrenorm, anneal, smashv, convs, bw, hypernet, crop, hyperpa
rams

NicolasGuacamole : generalisation, receptive, lasagne, learnmachinelearning, dilated, boostin
g, boundary, surely, semantic, insight

alexmlamb : authorship, autoregressive, ali, alex, alignment, anonymous, epsilon, professors,
forcing, icml

epicwisdom : liberties, ontology, gmail, unethical, accused, sexism, likewise, emails, consci
ousness, conscious

darkconfidantislife : ding, gabor, processors, googlenet, nm, processor, analog, chip, moveme
nt, convnets

bbsome : gn, martens, h_min, hessian, ir, hf, llvm, nevertheless, kfac, additionally

gwern : scott_e_reed, fredkin, webvision, kanal, danbooru, gwern, gaydar, tank, multilevel, a
le

olBaa : qualia, adult, baby, intelligent, consciousness, defining, room, chinese, vec, turing

phobrain : raf, skot, phobrain, gallery, ions, histograms, pics, cookie, dna, siamese
```

Text annotation

- Text processing often uses a 'pipeline' model (spacy)



Tokenization

- Tokenization is the dividing up of a sequence of characters into units (e.g., words, sentences):

  "Stop!" Dr. John shouted.

  becomes:

  ||"|Stop|!|"|Dr.|John|shouted|.||
- First step required before most text analysis
- Often dismissed as uninteresting, but not as easy as it looks

Words

- Generally finding word boundaries is straightforward (in English!) using spaces and punctuation
- Often contractions are treated as multiple words for tagging or parsing:

  can't → ca|n't
  they'll've → they|'ll|'ve
- Multiword expressions like *a little* or *in addition* can be treated as a single unit

PTB tokenization

- Most punctuation is split from adjoining words
- Double quotes (`"`) are changed to doubled single forward- and backward- quotes (` `` ` and `''`)
- Verb contractions and the Anglo-Saxon genitive of nouns are split into their component morphemes, and each morpheme is tagged separately.

```
children's → children 's
parents' → parents '
won't → wo n't
gonna → gon na
I'm → I 'm
```

PTB tokenization

- There are some subtleties for hyphens vs. dashes, ellipsis dots (`...`) and so on, but these often depend on the particular corpus or application of the tagged data.
- Bracket-like characters are converted to special 3-letter sequences to avoid confusion with parse brackets. These tokens in POS files:

  `( ) [ ] { }`

  become, in parsed files:

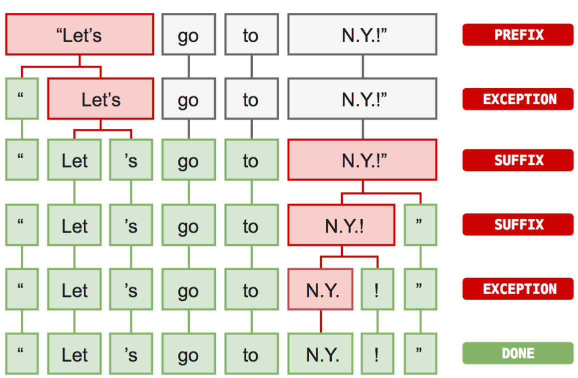
  `-LRB- -RRB- -RSB- -RSB- -LCB- -RCB-`

PTB tokenization

- Original sed implementation "does a decent enough job on most corpora, once the corpus has been formatted into one-sentence-per-line"
- nltk.word_tokenizer is a python reimplementation
- REPP is a special purpose language for defining tokenizers
- All of these work via a chain of search-and-replace operations

PTB tokenization

- spacy tokenizer uses prefix, infix, suffix, and exception rules



Tokenization

- Other alphabetic languages have their own conventions, but work more or less the same
- Languages that don't mark word boundaries in the orthography are more difficult
- MaxMatch:

  Input: 他特别喜欢北京烤鸭 "He especially likes Peking duck"
  Output: 他 特别 喜欢 北京烤鸭
  He especially likes Peking duck

  Input: `wecanonlyseeashortdistanceahead`
  Output: `we canon l y see ash ort distance ahead`

Sentences

- Identifying sentence boundaries is hard
- Periods in English are ambiguous, and may serve many functions:
  - end of sentence
  - abbreviations
  - decimal points
  - ellipses
- Other sentence-like units aren't clearly marked (lists, captions, etc.)

Sentence breaks

- Simplest rule: Every period is a sentence break
- Clearly wrong for titles, abbreviations, decimal numbers
- In the Brown corpus, there are 52,511 sentences ending in a period or question mark, and 3,569 contain a non-terminal period.
- So, the simple rule yields 93.20% correct (Grefenstette 1999)

Sentence breaks

- We can improve on the simple rule by identifying known classes of exceptions
- Numbers, which can include periods as decimal points, are easy to recognize:

  $123,456.78
  98.761%

- If we rule out decimal points as possible sentence boundaries, we fix 229 errors. This leaves 3,340 incorrect sentences, or 93.64% correct.

Sentence breaks

- Abbreviations are another large class of exceptions:
  - Single letter: G. Gordon Liddy c. 1910
  - Alternating letters and periods: U.S. i.e. m.p.h.
  - Capital letter followed by consonants: Mr. St. Assn.
- Results:

Pattern	Correct	Error	End
`[A-Za-z]\.`	1,323	30	14
`[A-Za-z]\.([A-Za-z0-9]\.)+`	626	0	63
`[A-Z][^aeiou]+\.`	1,927	33	26
Total	3,876	63	103

- Overall accuracy increases to 97.7%

Sentence breaks

- We can extract “likely abbreviations” from the corpus itself: words that end in a period followed by a comma, semi-colon, question mark, lower-case letter, number, or capitalized word ending in a period.
- Exclude from this any word which appears somewhere else without a period
- Results:

Pattern (unique)	**Total (unique)**	**Correct (unique)**	**Incorrect**
Likely abbrev	947	239	718
Not excluded	231	197	34

- Combined with the first two rules from the last method, this gets 51,642 out of 52,511 (98.35%) sentences correct.

Sentence breaks

- Printed dictionaries contain lists of common abbreviations
- Combined method:
  - if a candidate is followed by a lowercase letter, comma, or semi-colon, then it is an abbreviation
  - if it occurs in a list of common abbreviations, then it is an abbreviation
  - otherwise, it is a sentence boundary
- This gets 52,023 out of 52,511 sentences right, or 99.07% accuracy.

Sentence breaks

- Summary of Grefenstette’s (1999) results:

Method	**Errors**	**Accuracy**
All periods	3,869	93.20%
Numbers	3,340	93.64%
Patterns	1,229	97.66%
Corpus	869	98.35%
Dictionary	488	99.07%

- Moral #1: the more you know, the better you will do
- Moral #2: perfect accuracy is (usually) impossible

Tagging

- Part-of-speech tagging assigns a grammatical category to tokens in a corpus
- Since words may potentially occur as more than one part of speech, tagging is a limited kind of disambiguation:

The	*representative*	*put*	*the*	*chairs*	*on*	*the*	*table*	*.*
DET	NOUN	VERB	DET	NOUN	PREP	DET	NOUN	PERIOD

- Tagging can be done by hand, automatically, or as a combination of the two.

Tagging

- Various tag sets:

	CLAWS5	Brown	Penn	ICE
she	PNP	PPS	PRP	PRON(pers,sing)
was	VBD	BEDZ	VBD	AUX(pass,past)
told	VVN	VBN	VBN	V(ditr,edp)
that	CJT	CS	IN	CONJUNC(subord)
the	AT0	AT	DT	ART(def)
journey	NN1	NN	NN	N(com,sing)
might	VM0	MD	MD	AUX(modal,past)
kill	VVI	VB	VB	V(montr,infin)
her	PNP	PPO	PRP	PRON(poss,sing)
.	PUN	.	.	PUNC(per)

Tagging

- Tag sets differ greatly in the number and kind of distinctions they make.
  - Brown 179
  - Penn treebank 45
  - CLAWS1 132
  - CLAWS2 166
  - CLAWS5 65
  - London-Lund 197
- Tagsets are language and application dependent.

Tag	Description	Example	Tag	Description	Example
CC	Coordin. Conjunction	*and, but, or*	SYM	Symbol	+,%, &
CD	Cardinal number	*one, two, three*	TO	“to”	*to*
DT	Determiner	*a, the*	UH	Interjection	*ah, oops*
EX	Existential ‘there’	*there*	VB	Verb, base form	*eat*
FW	Foreign word	*mea culpa*	VBD	Verb, past tense	*ate*
IN	Preposition/sub-conj	*of, in, by*	VBG	Verb, gerund	*eating*
JJ	Adjective	*yellow*	VBN	Verb, past participle	*eaten*
JJR	Adj., comparative	*bigger*	VBP	Verb, non-3sg pres	*eat*
JJS	Adj., superlative	*wildest*	VBZ	Verb, 3sg pres	*eats*
LS	List item marker	*1, 2, One*	WDT	Wh-determiner	*which, that*
MD	Modal	*can, should*	WP	Wh-pronoun	*what, who*
NN	Noun, sing. or mass	*llama*	WP$	Possessive wh-	*whose*
NNS	Noun, plural	*llamas*	WRB	Wh-adverb	*how, where*
NNP	Proper noun, singular	*IBM*	$	Dollar sign	$
NNPS	Proper noun, plural	*Carolinas*	#	Pound sign	#
PDT	Predeterminer	*all, both*	“	Left quote	‘ or “
POS	Possessive ending	*’s*	”	Right quote	’ or ”
PRP	Personal pronoun	*I, you, he*	(	Left parenthesis	[, (, {, <
PRP$	Possessive pronoun	*your, one’s*	)	Right parenthesis	],), }, >
RB	Adverb	*quickly, never*	,	Comma	,
RBR	Adverb, comparative	*faster*	.	Sentence-final punc	. ! ?
RBS	Adverb, superlative	*fastest*	:	Mid-sentence punc	: ; ... – -
RP	Particle	*up, off*			

Figure 5.6 Penn Treebank part-of-speech tags (including punctuation).

Tagging

- Examples:

The/DT grand/JJ jury/NN commented/VBD on/IN a/DT number/NN of/IN other/JJ topics/NNS ./.

There/EX are/VBP 70/CD children/NNS **there/RB**

Although/IN preliminary/JJ findings/NNS were/VBD **reported/VBN** more/RBR than/IN a/DT year/NN ago/IN ,/, the/DT latest/JJS results/NNS appear/VBP in/IN today/NN **’s/POS** New/NNP England/NNP Journal/NNP of/IN Medicine/NNP ,/,

Mrs./NNP Shaefer/NNP never/RB got/VBD **around/RP** to/TO joining/VBG

All/DT we/PRP gotta/VBN do/VB is/VBZ go/VB **around/IN** the/DT corner/NN

Chateau/NNP Petrus/NNP costs/VBZ **around/RB** 250/CD

Tagging

- More challenges

  cotton/NN *sweater*/NN
  income-tax/JJ *return*/NN
  the/DT *Gramm-Rudman*/NP Act/NP

  Chinese/NN *cooking*/NN
  Pacific/NN *waters*/NNS

  They were married/VBN *by the Justice of the Peace*.
  At the time, she was already married/JJ.

Ambiguity

- Word types in the Brown corpus

Tags	Types
1	47,328
2	7,186
3	1,146
4	265
5	87
6	27
7	12
>7	6

- In the Brown corpus, 15.7% of the word types are ambiguous, but more 78.6% of the word tokens are ambiguous

Stochastic tagging

- Stochastic taggers take advantage of uneven tag probabilities
- If we want to assign a sequence of tags $t_1, \ldots, t_n$ to a sequence of words $w_1 \ldots w_n$, we need to maximize:

  $\mathrm{P}(t_1, \ldots, t_n \mid w_1, \ldots, w_n)$
- We construct a probability model using some annotated data.
- Typically, the annotated data available is divided into **training**, **validation**, and **test** data.

Stochastic tagging

- The simplest statistical tagger picks the most frequent tag for a given word
- The probability of a tag given a word can be estimated from training data:

$$P(\text{tag}|\text{word}) = \frac{\#\ \text{times word occurs with tag}}{\#\ \text{times word occurs}}$$

- Depending on the text and the tagset, this tagger can give as high as 91% word accuracy
- Since it is so simple, it is usually considered the **baseline**
- The 'human ceiling' is about 98%

HMM taggers

- If we make a few simplifying assumptions, HMMs provide a tool for adding context:
  - a word's tag only depends on the previous word's tag (**Markov** property)
- From the training data, we need to collect probabilities for tag bigrams:

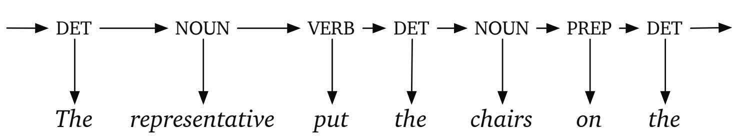
$$P(t_2|t_1) = \frac{C(t_1\,t_2)}{C(t_1)}$$

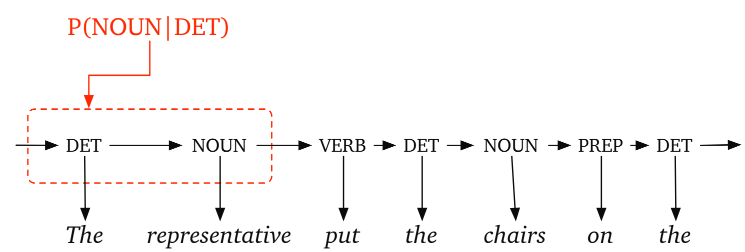
- And we need:

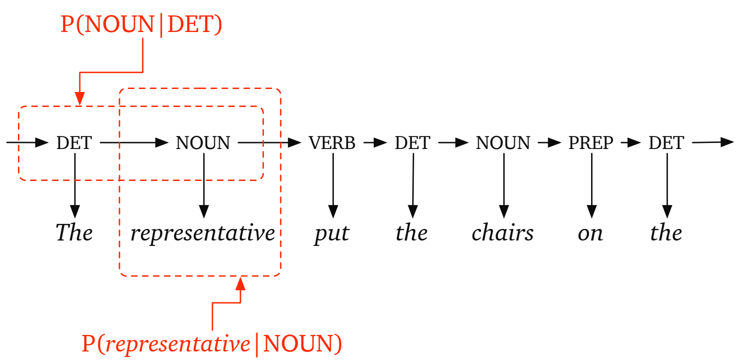
$$P(w|t) = \frac{C(w/t)}{C(t)}$$

HMM taggers

$$\begin{aligned}
\hat{t}_1,\ldots,\hat{t}_n &= \operatorname*{argmax}_t P(t_1,\ldots,t_n|w_1,\ldots,w_n) \\
&= \operatorname*{argmax}_t \frac{P(t_1,\ldots,t_n)\,P(w_1,\ldots,w_n|t_1,\ldots,t_n)}{P(w_1,\ldots,w_n)} \\
&= \operatorname*{argmax}_t P(t_1,\ldots,t_n)\,P(w_1,\ldots,w_n|t_1,\ldots,t_n) \\
&\approx \operatorname*{argmax}_t \prod_i P(t_i|t_{i-1})\,\prod_i P(w_i|t_i) \\
&= \operatorname*{argmax}_t \prod_i P(t_i|t_{i-1})\,P(w_i|t_i)
\end{aligned}$$







HMM taggers

- Markov models can be extended to take more context to account (bigrams, trigrams, etc.)
- Problems with Markov taggers:
  - unknown words
  - long-distance relations
  - independence assumptions
- More sophisticated statistical models are needed.

HMM taggers

- Weischedel, et al.'s (1993) "tri-tag" model used trigram probabilities:

$$P(t_1, \ldots, t_n) = P(t_1) \times P(t_2|t_1) \times \prod_{i=3}^{n} P(t_i|t_{i-1}t_{i-2})$$

- For known words, $P(w_i|t_i)$ is estimated in the usual way
- For unknown words:

$$P(w_i|t_i) = P(\text{unknown}|t_i) \times P(\text{capital}|t_i) \times P(\text{hyphen}|t_i) \times \prod_j P(\text{ending}_j|t_i)$$

- Works pretty well (85% unknown word accuracy), despite dubious independence assumptions