CS544: Morphology

January 17, 2013

Zornitsa Kozareva
USC/ISI
Marina del Rey, CA
kozareva@isi.edu
www.isi.edu/~kozareva

REGULAR EXPRESSIONS

Regular Expressions

- A formal language for specifying text strings
- It requires:
  - a **pattern** that we want to search for
  - a **corpus** (text collection) to search through

Regular Expressions

- How can we search for any of these?
  - woodchuck
  - woodchucks
  - Woodchuck
  - Woodchucks

Figure from Dorr/Monz slides

Regular Expressions

RE	Match	Example
/woodchucks/	woodchucks	"link to woodchucks and"
/a/	'a'	"Anna stopped by Mona's"
/he said/	he said	"Where are you", he said.
/read/	read	"his reading is great"
/!/	!	"it's my #bday!

… but /woodchucks/ will not match /Woodchucks/

Regular Expressions

RE	Match	Example
/[wW]oodchucks/	woodchucks or Woodchucks	"Woodchucks and lemurs"
/[abc]/	'a', 'b' or 'c'	"In soldati"
/[0123456789] /	any digit	"His number is 3 no 504"

… what if I want to match any lowercase letter?

… what if I want to match any lowercase and uppercase letter?

Regular Expressions

Range

RE	Match	Example
/[A-Z]/	an uppercase letter	“call Mr. Smith”
/[a-z]/	a lowercase letter	“he is here”
/[0-9] /	any digit	“His number is 3 no 504”

[ ] and the – specify any one character in a range

Regular Expressions

Negation

RE	Match	Example
/[^A-Z]/	not an upper case	“Dr. Smith will call you”
/[^Ss]/	neither ‘S’ nor ‘s’	“Sam is here”
/[^\.] /	not a period	“Dr. Smith will call you”
/e^ /	either ‘e’ or ‘^’	“where is the ^ symbol”
a^b	the pattern a^b	“define a^b”

the carret ^ is used for negation or just as ^

Regular Expressions

Optional Character

RE	Match	Example
woodchucks?	woodchuck or woodchucks	"woodchuck"
colou?r	color or colour	"Brits like to say colour"

the ? means (0 or 1) instances of previous character

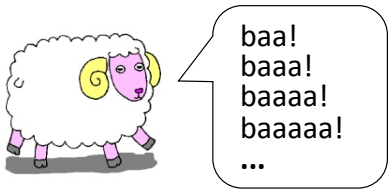
Regular Expressions

Optional Character

RE	Match	Example
woodchucks?	woodchuck or woodchucks	"woodchuck"
colou?r	color or colour	"Brits like to say colour"

the ? means '0' or '1' instances of previous character

Regular Expressions



Observations:
starts with b
followed by at least 2 a's
followed by !

the * means (0 or more) instances of previous character
baaa*!

the + means (1 or more) instances of previous character
baa+!

Regular Expressions

Wildcard

RE	Match	Example
/beg.n/	any character between beg and n	begin, began, begun

.* is used to capture any character

Capture any line in which the word "glory" appears two times
/glory.*glory/

Regular Expressions

- Anchors ^ and $
  - /^[A-Z]/ -> "Palo Alto"
  - /^[^A-Z]/ -> "¿verdad?" "really?"
  - /\.$/ -> "It is over."
  - /.$/ -> ?
- Boundaries \b and \B
  - /\bon\b/ -> "on my way" "Monday"
  - /\Bon\b/ -> "automaton"
- Disjunction |
  - /yours|mine/ -> "it is either yours or mine"

Slide from Dorr/Monz

Sanity Check

- Which word will not be captured by the RE

  Brit*[ea]?ne?y

  - Britney
  - Brittney
  - Britany
  - Britiney

Example from Jurafsky

Disjunction, Grouping, Precedence

- If we want to find

  Column 1 Column 2 Column 3 …

  ```
  /Column [0-9]+ */
  /(Column [0-9]+  +)*/
  ```

- Precedence

Parenthesis	()
Counters	* + ? {}
Sequences and anchors	the ^my end$
Disjunction	|

Simple RE Example

- Find all cases of the word "the" in a text.
  - /the/

    Misses capitalized examples
  - /[tT]he/
    - Returns other or theology
  - /\b[tT]he\b/
  - /[^a-zA-Z][tT]he[^a-zA-Z]/
    - Misses sentence-initial "the"
  - /(^|[^a-zA-Z])[tT]he[^a-zA-Z]/

Errors

- The process we just went through was based on fixing two kinds of errors
  - False positives (Type I): matching strings that we should not have matched (there, then, other)
  - False negatives (Type II): not matching things that we should have matched (The)

Slide from Jurafsky

Errors

- We will be telling the same error story for different tasks, you will even experience it yourself in the homework
- Reducing the error rate for an application often involves two antagonistic efforts:
  - increasing accuracy or precision
    (minimizing false positives)
  - increasing coverage or recall
    (minimizing false negatives)

Slide from Jurafsky

More Complex RE Example

- Regular expressions for prices
- /$[0-9]+/
  - Doesn't deal with fractions of dollars
- /$[0-9]+\.[0-9][0-9]/
  - Doesn't allow $199, not word-aligned
- \b$[0-9]+(\.[0-9]0-9])?\b)

Slide from Jurafsky

Summary

- Regular expressions play a surprisingly large role
  - for hard tasks, we use machine learning classifiers
  - but complex sequences of regular expressions could still be a first thing to try, or we can use them to clean up and pre-process our data

Slide from Jurafsky

ELIZA

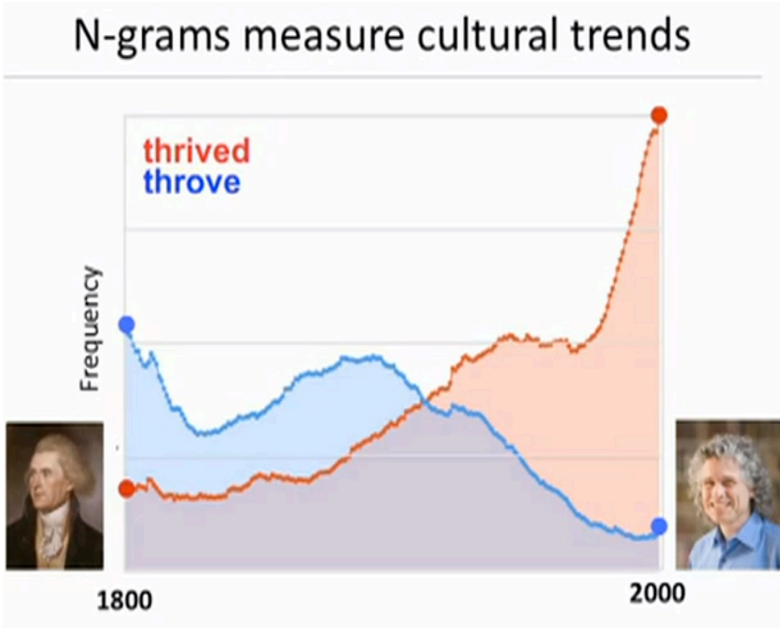
- ELIZA, a simple NLP program developed by (Weizenbaum, 1966)
- It simulates Rogerian psychologist and could carry on conversations

http://nlp-addiction.com/eliza/

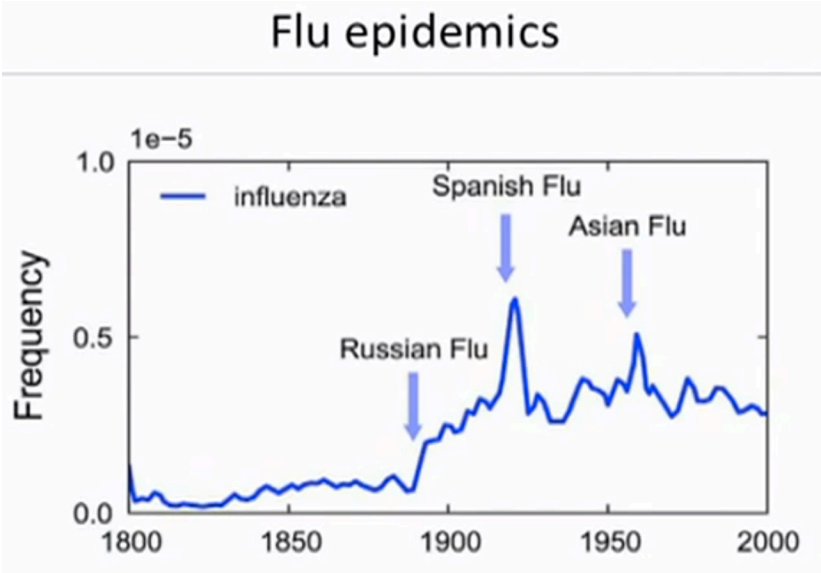
Behind the Scene

- ELIZA has a cascade of regular expression substitutions that each matched some part of the input line and changed them
- Examples:
  - Change all instance of *my* into *YOUR*
  - Change *I'm* to *YOU ARE*
  - Replace relevant patterns

```
s/.* I AM (depressed|sad) .*/I AM SORRY TO HEAR YOU ARE \1/
s/.* YOU ARE (depressed|sad) .*/WHY DO YOU THINK YOU ARE \1/
s/.* all .*/IN WHAT WAY/
s/.* always .*/CAN YOU THINK OF A SPECIFIC EXAMPLE/
```



http://www.youtube.com/watch?v=5l4cA8zSreQ



http://books.google.com/ngrams

- Searching for the singular and plural of *woodchuck* was easy, just add (*s*) *woodchucks*
- However, the singulars and plurals of

  fox -> foxes

  peccary ->peccaries

  goose -> geese

  fish -> fish

spelling rules: `-y` changes to `-i` and add `-es`

morphological rules: `fish` has null plural

`goose` is formed by changing vowels

MORPHOLOGY

Morphological Analysis

- *Morphology* studies the internal structure of words
- A *morpheme* is the smallest linguistic unit that has semantic meaning
- *Morphological Analysis* is the task of segmenting a word into its morphemes
  - carried => carry + ed (past tense)
  - disconnect => dis (not) + connect
  - foxes => fox +es

Morpholgy

- Words are composed of
  - ***stems****:* the core meaning bearing units
  - ***affixes***: bits and pieces that adhere to stems to change their meanings and grammatical functions
- Affixes are divided into:
  - prefixes (precede the stem)
  - suffixes (follow the stem)
  - infixes (inserted inside the stem)
  - circumfixes (precede and follow the stem)

Examples

- Circumfixes in German

  past participle of some verbs is formed by adding *ge-* at the beginning and *–t* to the end

  sagen (to say) -> *gesagt* (said)

- Infixes in Philippine language Tagalog

  stem *hingi* (borrow) and the affix *um* which marks agent of action produce *humingi*

Root-and-Pattern morphology

- In Hebrew, verb is constructed with a root composed of three consonants (CCC) and a templates for consonant and vowel ordering
- Ex: room *lmd* means *learn*, *study*
  - template CaCaC produces active voice
    lamad (he studied)
  - template CiCeC produces
    limed (he taught)
  - template CuCaC produces
    lumad (he was taught)

Can a word have more than one affix?

- In English words have no more than 4 or 5 affixes
  - *rewrites*
  - *unbelievably*
- In Turkish words can have up to 9 or 10 affixes

  uygarlastiramadiklarimizdanmissinizcasina

  (behaving) as if you are among those whom we could not civilize

  uygar `civilized' + las `become'
  + tir `cause' + ama `not able'
  + dik `past' + lar 'plural'
  + imiz 'p1pl' + dan 'abl'
  + mis 'past' + siniz '2pl' + casina 'as if'

Stemming

- Reduce terms to their "roots" before indexing
- "Stemming" is crude chopping of "affixes"
  - language dependent
  - e.g., ***automate(s), automatic, automation*** all reduced to ***automat***.

for example compressed and compression are both accepted as equivalent to compress.

for exampl compress and compress ar both accept as equival to compress

Slide from Chris Manning

Blindly stripping affixes can produce strange results …

- `preempt -> empt`
- `news -> new`
- `pretended -> tend`
- `hardly -> hard`
- `glasses -> glass`
- `Mrs ->Mr`
- `Easter ->East`

Porter's algorithm

- Commonest algorithm for stemming English
- A sequence of phases
- Each phase consists of a set of rules
  - *sses → ss*
  - *ies → i*
  - *ational → ate (e.g. relational->relate)*
  - *tional → tion*
  - *ing -> ø if stem has vowel (e.g. motoring->motor)*
  - Some rules only apply to multi-syllable words
    - *replacement → replac*
    - *cement → cement*

Slide from Chris Manning

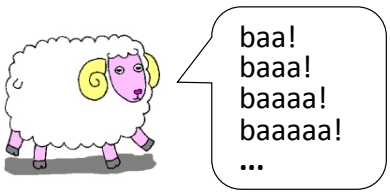
Morphology with FSA

- Morphological analysis is often performed with finite-state transducers (FST)
- An FST is a finite-state automaton that maps between two sets of symbols
- An FST can be viewed as a *generator* or a *recognizer* between pairs of strings

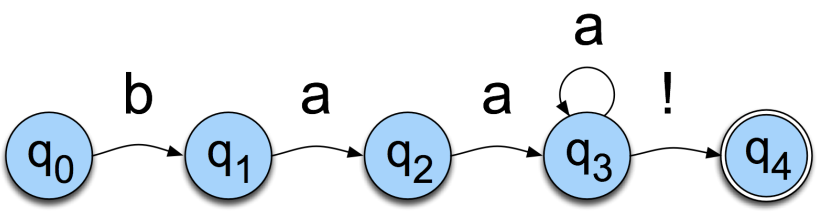
Finite State Automata (FSA)

- Even regular expressions can be viewed as a textual way of specifying the structure of finite-state automata

FSAs as Graphs

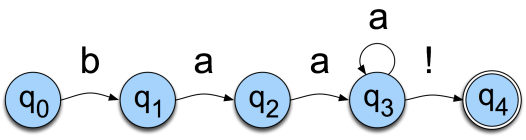



```
Regular Expression
/baa+!/
```



Sheep FSA

- We can say the following things about this machine
  - It has 5 states
  - b, a, and ! are in its alphabet
  - q_0 is the start state
  - q_4 is an accept state
  - It has 5 transitions



More Formally

- You can specify an FSA by enumerating the following things
  - The set of states: Q
  - A finite alphabet: Σ
  - A start state
  - A set of accept/final states
  - A transition function that maps QxΣ to Q

Does Stemming Improve NLP systems?

- Machine Translation system needs to know that the Spanish word

  quiero ('I want')

  quieres ('you want')

  are both related to querer 'want'

Does Stemming Improve NLP systems?

- Information Retrieval
  - (Krovetz, 1993) showed that improvement is very small and often stemming is not used in search engines
  - Improvements were noticed for small document collections, however the larger the collection the higher the chance for finding the exact word from the query
  - Note the experiment was done mainly on English texts, results could differ based on the language

TOKENIZATION, LEMMATIZATION, SEGMENTATION

Tokenization

- Applications
  - Information retrieval
  - Information extraction (detecting named entities)
  - Spell-checking
  - …
- 3 tasks
  - Segmenting/tokenizing words in text
  - Normalizing word formats
  - Segmenting sentences in text

Why not just periods and white-space?

- Mr. Sherwood said reaction to Sea Containers' proposal has been "very positive." In New York Stock Exchange composite trading yesterday, Sea Containers closed at $62.625, up 62.5 cents.
- "I said, 'what're you? Crazy?' " said Sadowsky. "I can't afford to do that."

What's a word?

- I do uh main- mainly business data processing
  - Fragments (main-)
  - Filled pauses (uh)
- Are cat and cats the same word?
- Some terminology
  - **Lemma**: a set of lexical forms having the same stem, major part of speech, and rough word sense
    - cat and cats = same lemma
  - **Wordform**: the full inflected surface form.
    - cat and cats = different wordforms
  - Token/Type

Difference between Tokens and Types

Rose is a rose is a rose.

- tokens are concrete (10)
- types are abstract and unique (3 or 4)

How Many Words?

- The Switchboard corpus of American telephone conversation:
  - 2.4 million wordform tokens
  - ~20,000 wordform types
- Brown et al (1992) large corpus of text
  - 583 million wordform tokens
  - 293,181 wordform types
- Shakespeare:
  - 884,647 wordform tokens
  - 31,534 wordform types
- Let N = number of tokens, V = vocabulary = number of types
- General wisdom: V > O(sqrt(N))

Zipf's law

- In book "Human Behavior and the Principle of Least Effort" Zipf's argues that people will act so as to minimize their probable average rate of work
  - speaker's effort is conserved by having a small vocabulary of common words
  - hearer's effort is lessened by having a large vocabulary of individually rarer words (so that messages areess ambiguous)

Zipf's Law

- If we list the words by their frequency of occurrence, we can explore the relationship between the frequency of a word *f* and the rank *r* which is its position in the list as

  f*r=k

Zipf's Law

Word	Freq. (f)	Rank (r)	$f \cdot r$	Word	Freq. (f)	Rank (r)	$f \cdot r$
the	3332	1	3332	turned	51	200	10200
and	2972	2	5944	you'll	30	300	9000
a	1775	3	5235	name	21	400	8400
he	877	10	8770	comes	16	500	8000
but	410	20	8400	group	13	600	7800
be	294	30	8820	lead	11	700	7700
there	222	40	8880	friends	10	800	8000
one	172	50	8600	begin	9	900	8100
about	158	60	9480	family	8	1000	8000
more	138	70	9660	brushed	4	2000	8000
never	124	80	9920	sins	2	3000	6000
Oh	116	90	10440	Could	2	4000	8000
two	104	100	10400	Applausive	1	8000	8000

Table *1.3* Empirical evaluation of Zipf's law on Tom *Sawyer*.

Issues in Tokenization

- How should we tokenize the following:
  - Finland's capital
  - Hewlett-Packard (one token or two)
  - State-of-the-art (break up or not)
  - San Francisco, New York (one token or two)
  - Ph.D., m.p.h (words with punctuations)

Slide from Chris Manning

Tokenization: Language Issues

- French

  L'ensemble should it be one token or two
  - ***L*** ? ***L'*** ? ***Le*** ?
  - Want ***l'ensemble*** to match with ***un ensemble***

- German noun compounds are not segmented

  lebensversicherungsgesellschaftsangestellter

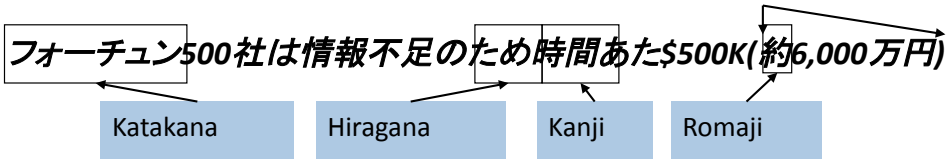
  life insurance company employee

  Note: German retrieval systems benefit greatly from a **compound splitter**

Slide from Jurafsky

Tokenization: Language Issues

- Chinese and Japanese no spaces between words:
  - 莎拉波娃现在居住在美国东南部的佛罗里达。
  - 莎拉波娃 现在 居住 在 美国 东南部 的 佛罗里达
  - Sharapova now lives in US southeastern Florida
- Further complicated in Japanese, with multiple alphabets intermingled
  - dates/amounts are in multiple formats



End-user can express query entirely in hiragana!

Slide from Jurafsky

Word Segmentation in Chinese

- Words composed of characters
- Characters are generally 1 syllable and 1 morpheme
- Average word is 2.4 characters long
- Standard segmentation algorithm:
  - Maximum Matching

Maximum Matching Word Segmentation Algorithm

Given: a wordlist of Chinese and a string

Algorithm:

1) start a pointer at the beginning of the string
2) find the longest word in dictionary that matches the string starting at pointer
3) move the pointer over the word in string
4) go to 2

Slide from Jurafsky

English failure example (Palmer 00)

the table down there

thetabledownthere

Theta bled own there

- But works astonishingly well in Chinese
  - 莎拉波娃现在居住在美国东南部的佛罗里达。
  - 莎拉波娃 现在 居住 在 美国 东南部 的 佛罗里达
- Modern algorithms better still:
  - probabilistic segmentation
  - using "sequence models" like HMMs

Normalization

- For IR, indexed text & query terms must have the same form
  - ***U.S.A.*** should match ***USA***
- We most commonly implicitly define equivalence classes of terms
  - e.g., by deleting periods in a term
- Alternative is to do asymmetric expansion:
  - Enter: ***window*** Search: ***window, windows***
  - Enter: ***windows*** Search: ***Windows, windows, window***
  - Enter: ***Windows*** Search: ***Windows***
- Potentially more powerful, but less efficient

Case Folding

- For IR, best to lower case everything, since users will use lowercase regardless of 'correct' capitalization (exception upper case in middle of sentence)
  - ***General Motors***
  - ***Fed*** vs. ***fed***
  - ***SAIL*** vs. ***sail***
- For sentiment analysis, MT, IE case is helpful ("US" versus "us" is important)

Slide from Chris Manning

Lemmatization

- Lemmatization implies doing "proper" reduction to dictionary headword form
- Example:
  - *am, are, is → be*
  - *car, cars, car's, cars' → car*

The boy's cars are different colors

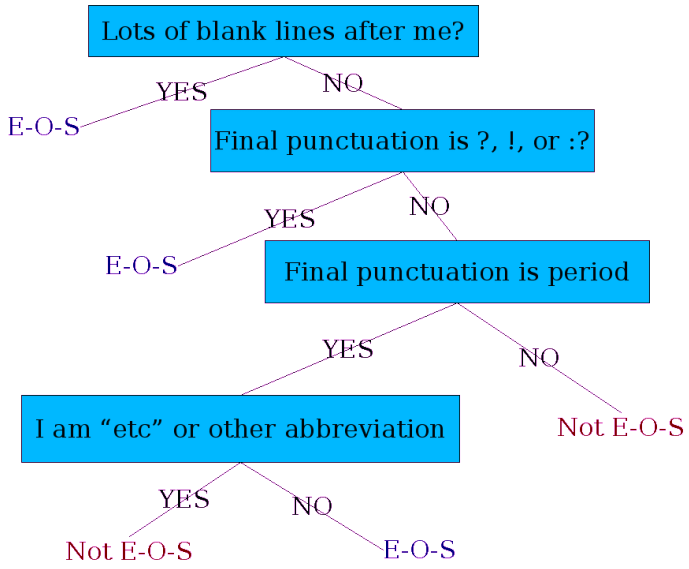
The boy car be different color

Sentence Segmentation

- !, ? relatively unambiguous
- Period "." is quite ambiguous
  - Sentence boundary
  - Abbreviations like *Inc.*, *Dr.* or *Ph.D.*
- General idea is to build a binary classifier that
  - looks at a "."
  - decides EndOfSentence/NotEOS
  - could be hand-written rules, sequences of regular expressions, or machine-learning

Slide from Jurafsky

Determining if a word is end-of-utterance: a Decision Tree



Slide from Jurafsky