

CS 224S / LINGUIST 236 Speech Recognition and Synthesis

Dan Jurafsky

Lecture 4: Text Processing for TTS

IP Notice: lots of info, text, and diagrams on these slides comes (thanks!) from Alan Black's excellent lecture notes and from Richard Sproat's slides.

1/13/05

CS 224S Winter 2005

1

Outline

- Text Processing
 - Text Normalization
 - Homograph disambiguation
 - Part-of-speech tagging
- Letter-to-Sound Rules
 - (or "Grapheme-to-Phoneme Conversion")

1/13/05

CS 224S Winter 2005

2

Text Processing

- He stole \$100 million from the bank
- It's 13 St. Andrews St.
- The home page is <http://www.stanford.edu>
- Yes, see you the following tues, that's 11/12/01
- IV: four, fourth, I.V.
- IRA: I.R.A. or Ira
- 1750: seventeen fifty (date, address) or one thousand seven... (dollars)

1/13/05

CS 224S Winter 2005

3

Steps

- Identify tokens in text
- Chunk tokens
- Identify types of tokens
- Convert tokens to words

1/13/05

CS 224S Winter 2005

4

Step 1: identify tokens and chunk

- Whitespace can be viewed as separators
- Punctuation can be separated from the raw tokens
- Festival converts text into
 - ordered list of tokens
 - each with features:
 - its own preceding whitespace
 - its own succeeding punctuation

1/13/05

CS 224S Winter 2005

5

End-of-utterance detection

- Relatively simple if utterance ends in ?!
- But what about ambiguity of "."
- Ambiguous between end-of-utterance and end-of-abbreviation
 - My place on Forest Ave. is around the corner.
 - I live at 360 Forest Ave.
 - (Not "I live at 360 Forest Ave..")
- How to solve this period-disambiguation task?

1/13/05

CS 224S Winter 2005

6

CART

- Breiman, Friedman, Olshen, Stone. 1984. *Classification and Regression Trees*. Chapman & Hall, New York.
- Description/Use:
 - Binary tree of decisions, terminal nodes determine prediction ("20 questions")
 - If dependent variable is categorical, "classification tree",
 - If continuous, "regression tree"

1/13/05

CS 224S Winter 2005

Text from Richard Sproat 7

Determining end-of-utterance The Festival hand-built decision tree

```
((n.whitespace matches ".*\n.*\n[ \n]*") ; A significant break in text
(1))
((punc in ("?" ":" "!"))
(1))
((punc is ".")
; This is to distinguish abbreviations vs periods
; These are heuristics
((name matches "\\(.*\\.?.*\|[A-Z][A-Za-z]?[A-Za-z]?\\|etc\\)")
((n.whitespace is " ")
(0))
; if abbrev, single space enough for break
((n.name matches "[A-Z].*")
(1))
(0)))
((n.whitespace is " ") ; if it doesn't look like an abbreviation
((n.name matches "[A-Z].*") ; single sp. + non-cap is no break
(1))
(0)))
(1)))
(0)))
(1/13/05
```

CS 224S Winter 2005

8

The previous decision tree

- A dot with one or two letters is an abbrev
- A dot with 3 cap letters is an abbrev.
- An abbrev followed by 2 spaces and a capital letter is an end-of-utterance
- Non-abbrevs followed by capitalized word are breaks
- This fails for
 - Cog. Sci. Newsletter
 - Lots of cases at end of line.
 - Badly spaced/capitalized sentences

1/13/05

CS 224S Winter 2005

9

More sophisticated decision tree features

- Prob(word with "." occurs at end-of-s)
- Prob(word after "." occurs at begin-of-s)
- Length of word with "."
- Length of word after "."
- Case of word with ".": Upper, Lower, Cap, Number
- Case of word after ".": Upper, Lower, Cap, Number
- Punctuation after "." (if any)
- Abbreviation class of word with "." (month name, unit-of-measure, title, address name, etc)

1/13/05

CS 224S Winter 2005

10

From Richard Sproat slides

Learning DTs

- DTs are rarely built by hand
- Hand-building only possible for very simple features, domains
- Lots of algorithm for DT induction
- Covered in detail in CS 221 AI, CS 229 Machine Learning, etc
- I'll give quick intuition here

1/13/05

CS 224S Winter 2005

11

CART Estimation

- Creating a binary decision tree for classification or regression involves 3 steps:
 - Splitting Rules: Which split to take at a node?
 - Stopping Rules: When to declare a node terminal?
 - Node Assignment: Which class/value to assign to a terminal node?

1/13/05

CS 224S Winter 2005

12

From Richard Sproat slides

Splitting Rules

- Which split to take a node?
- Candidate splits considered:
 - Binary cuts: for continuous ($-\infty < x < \infty$) consider splits of form:
 - $X \leq k$ vs. $x > k \forall k$
 - Binary partitions: For categorical $x \in \{1, 2, \dots\} = X$ consider splits of form:
 - $x \in A$ vs. $x \in X-A, \forall A \in X$

1/13/05

CS 224S Winter 2005

13

From Richard Sproat slides

Splitting Rules

- Choosing best candidate split.
 - Method 1: Choose k (continuous) or A (categorical) that minimizes estimated classification (regression) error after split
 - Method 2 (for classification): Choose k or A that minimizes estimated entropy after that split.

1/13/05

CS 224S Winter 2005

From Richard Sproat slides

Decision Tree Stopping

- When to declare a node terminal?
- Strategy (Cost-Complexity pruning):
 1. Grow over-large tree
 2. Form sequence of subtrees, $T_0 \dots T_n$ ranging from full tree to just the root node.
 3. Estimate "honest" error rate for each subtree.
 4. Choose tree size with minimum "honest" error rate.
- To estimate "honest" error rate, test on data different from training data (I.e. grow tree on 9/10 of data, test on 1/10, repeating 10 times and averaging (cross-validation)).

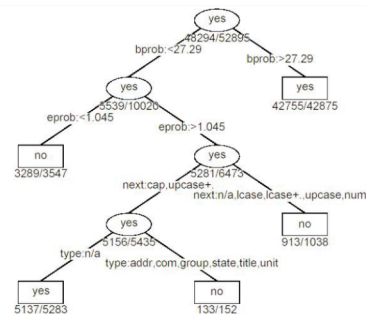
1/13/05

CS 224S Winter 2005

From Richard Sproat

15

Sproat EOS tree



1/13/05

CS 224S Winter 2005

From Richard Sproat slides

16

Summary on end-of-sentence detection

- **Best references:**
 - David Palmer and Marti Hearst. 1997. Adaptive Multilingual Sentence Boundary Disambiguation. Computational Linguistics 23, 2. 241-267.
 - David Palmer. 2000. Tokenisation and Sentence Segmentation. In "Handbook of Natural Language Processing", edited by Dale, Moisl, Somers.

1/13/05

CS 224S Winter 2005

17

Steps 3+4: Identify Types of Tokens, and Convert Tokens to Words

- **Pronunciation of numbers often depends on type:**
 - 1776 date: **seventeen seventy six.**
 - 1776 phone number: **one seven seven six**
 - 1776 quantifier: **one thousand seven hundred (and) seventy six**
 - 25 day: **twenty-fifth**

1/13/05

CS 224S Winter 2005

18

Festival rule for dealing with "\$1.2 million"

```
(define (token_to_words utt token name)
  (cond
    ((and (string-matches name "\\$[0-9,]+\\(\\.[0-9]+\\)?" )
          (string-matches (utt.streamitem.feats utt token "n.name")
                          ".*million.?"))
     (append
      (builtin_english_token_to_words utt token (string-after name "$"))
      (list
       (utt.streamitem.feats utt token "n.name"))))
    ((and (string-matches (utt.streamitem.feats utt token "p.name")
                          "\\$[0-9,]+\\(\\.[0-9]+\\)?" )
          (string-matches name ".*million.?"))
     (list "dollars"))
    (t
     (builtin_english_token_to_words utt token name))))
```

1/13/05

CS 224S Winter 2005

19

Rule-based versus machine learning

- **As always, we can do things either way, or more often by a combination**
- **Rule-based:**
 - Simple
 - Quick
 - Can be more robust
- **Machine Learning**
 - Works for complex problems where rules hard to write
 - Higher accuracy in general
 - But worse generalization to very different test sets
- **Real TTS and NLP systems**

1/13/05

Often use aspects of both

20

Machine learning method for Text Normalization

- From 1999 Hopkins summer workshop "Normalization of Non-Standard Words"
- Sproat, R., Black, A., Chen, S., Kumar, S., Ostendorf, M., and Richards, C. 2001. Normalization of Non-standard Words, Computer Speech and Language, 15(3):287-333
- **NSW examples:**
 - Numbers:
 - 123, 12 March 1994
 - Abbreviations, contractions, acronyms:
 - approx., mph, ctrl-C, US, pp, lb
 - Punctuation conventions:
 - 3-4, +/-, and/or
 - Dates, times, urls, etc

1/13/05

CS 224S Winter 2005

21

How common are NSWs?

- Varies over text type
- Word not in lexicon, or with non-alphabetic characters:

Text Type	% NSW
novels	1.5%
press wire	4.9%
e-mail	10.7%
recipes	13.7%
classified	17.9%

1/13/05

CS 224S Winter 2005

22

From Alan Black slides

How hard are NSWs?

- **Identification:**
 - Some homographs "Wed", "PA"
 - False positives: OOV
 - **Realization:**
 - Simple rule: money, \$2.34
 - Type identification+rules: numbers
 - Text type specific knowledge (in classified ads, BR for bedroom)
 - **Ambiguity (acceptable multiple answers)**
 - "D.C." as letters or full words
 - "MB" as "meg" or "megabyte"
- 250

1/13/05

CS 224S Winter 2005

23

Step 1: Splitter

- Letter/number conjunctions (WinNT, SunOS, PC110)
- Hand-written rules in two parts:
 - Part I: group things not to be split (numbers, etc; including commas in numbers, slashes in dates)
 - Part II: apply rules:
 - At transitions from lower to upper case
 - After penultimate upper-case char in transitions from upper to lower
 - At transitions from digits to alpha
 - At punctuation

1/13/05

CS 224S Winter 2005

24

From Alan Black Slides

Step 2: Classify token into 1 of 20 types

- EXPN: abbrev, contractions (adv, N.Y., mph, gov't)
- LSEQ: letter sequence (CIA, D.C., CDs)
- ASWD: read as word, e.g. CAT, proper names
- MSPL: misspelling
- NUM: number (cardinal) (12,45,1/2, 0.6)
- NORD: number (ordinal) e.g. May 7, 3rd, Bill Gates II
- NTEL: telephone (or part) e.g. 212-555-4523
- NDIG: number as digits e.g. Room 101
- NIDE: identifier, e.g. 747, 386, I5, PC110
- NADDR: number as street address, e.g. 5000 Pennsylvania
- NZIP, NTIME, NDATE, NYER, MONEY, BMONY, PRCT, URL, etc
- SLNT: not spoken (KENT*REALTY)

1/13/05

CS 224S Winter 2005

25

More about the types

- 4 categories for alphabetic sequences:
 - EXPN: expand to full word or word seq (fplc for fireplace, NY for New York)
 - LSEQ: say as letter sequence (IBM)
 - ASWD: say as standard word (either OOV or acronyms)
- 5 main ways to read numbers:
 - Cardinal (quantities)
 - Ordinal (dates)
 - String of digits (phone numbers)
 - Pair of digits (years)
 - Trailing unit: serial until last non-zero digit: 8765000 is "eight seven six five thousand" (some phone numbers, long addresses)
 - But still exceptions: (947-3030, 830-7056)

1/13/05

CS 224S Winter 2005

26

Type identification algorithm

- Create large hand-labeled training set and build a DT to predict type
- Example of features in tree for subclassifier for alphabetic tokens:
 - $P(t|o) = p(o|t)p(t)/p(o)$
 - $P(o|t)$, for t in ASWD, LSQW, EXPN (from trigram letter model)

$$p(o|t) = \sum_{i=1}^N p(l_i | l_{i-1}, l_{i-2})$$
 - $P(t)$ from counts of each tag in text
 - $P(o)$ normalization factor

1/13/05

CS 224S Winter 2005

27

Type identification algorithm

- Hand-written context-dependent rules:
 - List of lexical items (Act, Advantage, amendment) after which Roman numbers read as cardinals not ordinals
- Classifier accuracy:
 - 98.1% in news data,
 - 91.8% in email

1/13/05

CS 224S Winter 2005

28

Step 3: expanding NSW Tokens

- **Type-specific heuristics**
 - ASWD expands to itself
 - LSEQ expands to list of words, one for each letter
 - NUM expands to string of words representing cardinal
 - NYER expand to 2 pairs of NUM digits...
 - NTEL: string of digits with silence for punctuation
 - Abbreviation:
 - use abbrev lexicon if it's one we've seen
 - Else use training set to know how to expand
 - Cute idea: if "eat in kit" occurs in text, "eat-in kitchen" will also occur somewhere.

1/13/05

CS 224S Winter 2005

29

4 steps to Sproat et al. algorithm

- 1) **Splitter** (on whitespace or also within word ("AltaVista"))
- 2) **Type identifier**: for each split token identify type
- 3) **Token expander**: for each typed token, expand to words
 - Deterministic for number, date, money, letter sequence
 - Only hard (nondeterministic) for abbreviations
- 4) **Language Model**: to select between alternative pronunciations

1/13/05

CS 224S Winter 2005

30

From Alan Black slides

Homograph disambiguation

- **19 most frequent homographs, from Liberman and Church**

use	319	survey	91
increase	230	project	90
close	215	separate	87
record	195	present	80
house	150	read	72
contract	143	subject	68
lead	131	rebel	48
live	130	finance	46
lives	105	estimate	46
protest	94		

- **Not a huge problem, but still important**

1/13/05

CS 224S Winter 2005

31

POS Tagging for homograph disambiguation

- **Many homographs can be distinguished by POS**

- use y u w s	y u w z
- close k l o w s	k l o w z
- house h a w s	h a w z
- live l a y v	l i h v
- REcord	reCORD
- INsult	inSULT
- OBject	obJECT
- OVERflow	overFLOW
- DIScount	disCOUNT
- CONtent	conTENT

1/13/05

CS 224S Winter 2005

32

Part of speech tagging

- **8 (ish) traditional parts of speech**
 - Noun, verb, adjective, preposition, adverb, article, interjection, pronoun, conjunction, etc
 - This idea has been around for over 2000 years (Dionysius Thrax of Alexandria, c. 100 B.C.)
 - Called: parts-of-speech, lexical category, word classes, morphological classes, lexical tags, POS
 - We'll use POS most frequently

1/13/05

CS 224S Winter 2005

33

POS examples

- **N** noun chair, bandwidth, pacing
- **V** verb study, debate, munch
- **ADJ** adj purple, tall, ridiculous
- **ADV** adverb unfortunately, slowly,
- **P** preposition of, by, to
- **PRO** pronoun I, me, mine
- **DET** determiner the, a, that, those

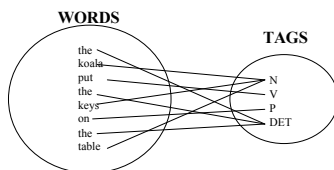
1/13/05

CS 224S Winter 2005

34

POS Tagging: Definition

- The process of assigning a part-of-speech or lexical class marker to each word in a corpus:



1/13/05

CS 224S Winter 2005

35

POS Tagging example

WORD	tag
the	DET
koala	N
put	V
the	DET
keys	N
on	P
the	DET
table	N

1/13/05

CS 224S Winter 2005

36

Open and closed class words

- **Closed class: a relatively fixed membership**
 - Prepositions: of, in, by, ...
 - Auxiliaries: may, can, will had, been, ...
 - Pronouns: I, you, she, mine, his, them, ...
 - Usually **function words** (short common words which play a role in grammar)
- **Open class: new ones can be created all the time**
 - English has 4: Nouns, Verbs, Adjectives, Adverbs
 - Many languages have all 4, but not all!
 - In Lakhota and possibly Chinese, what English treats as adjectives act more like verbs.

1/13/05

CS 224S Winter 2005

37

Open class words

- **Nouns**
 - Proper nouns (Stanford University, Boulder, Neal Snider, Margaret Jacks Hall). English capitalizes these.
 - Common nouns (the rest). German capitalizes these.
 - Count nouns and mass nouns
 - Count: have plurals, get counted: goat/goats, one goat, two goats
 - Mass: don't get counted (snow, salt, communism) ("two snows")
- **Adverbs: tend to modify things**
 - **Unfortunately**, John walked home **extremely slowly yesterday**
 - Directional/locative adverbs (here, home, downhill)
 - Degree adverbs (extremely, very, somewhat)
 - Manner adverbs (slowly, slinkily, delicately)
- **Verbs:**
 - In English, have morphological affixes (eat/eats/eaten)

1/13/05

CS 224S Winter 2005

38

Closed Class Words

- **Idiosyncratic**
- **Examples:**
 - prepositions: on, under, over, ...
 - particles: up, down, on, off, ...
 - determiners: a, an, the, ...
 - pronouns: she, who, I, ..
 - conjunctions: and, but, or, ...
 - auxiliary verbs: can, may should, ...
 - numerals: one, two, three, third, ...

1/13/05

CS 224S Winter 2005

39

POS tagging: Choosing a tagset

- There are so many parts of speech, potential distinctions we can draw
- To do POS tagging, need to choose a standard set of tags to work with
- Could pick very coarse tagsets
 - N, V, Adj, Adv.
- More commonly used set is finer grained, the "UPenn TreeBank tagset", 45 tags
 - PRP\$, WRB, WP\$, VBG
- Even more fine-grained tagsets exist

1/13/05

CS 224S Winter 2005

40

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

1/13/05 CS 224S Winter 2005 41

Using the UPenn tagset

- The/DT grand/JJ jury/NN commented/VBD on/IN a/DT number/NN of/IN other/JJ topics/NNS ./.
- Prepositions and subordinating conjunctions marked IN ("although/IN I/PRP..")
- Except the preposition/complementizer "to" is just marked "to".

1/13/05

CS 224S Winter 2005

42

POS Tagging

- Words often have more than one POS:
 - The *back* door = JJ
 - On my *back* = NN
 - Win the voters *back* = RB
 - Promised to *back* the bill = VB
- The POS tagging problem is to determine the POS tag for a particular instance of a word.

1/13/05

CS 224S Winter 2005

43

How hard is POS tagging? Measuring ambiguity

Unambiguous (1 tag): 38,857
Ambiguous (2-7 tags): 8,844

Number of tags	Count	Examples
2 tags	6,731	
3 tags	1621	
4 tags	357	
5 tags	90	
6 tags	32	
7 tags	6	well, set, round, open, fit, down
8 tags	4	's, half, back, a
9 tags	3	that, more, in

1/13/05

CS 224S Winter 2005

44

Write rules to eliminate tags

Eliminate VBN if VBD is an option when VBN|VBD follows "<start> PRP"

			NN		
			RB		
	VBN		JJ		VB
PRP	VBD	TO	VB	DT	NN
She	promised	to	back	the	bill

1/13/05

CS 224S Winter 2005

49

Stochastic Tagging

- Intuition: assign each word "most probable" tag
- Simplest way to define "most probable"
 - Collect a training corpus
 - Choose tag which is most frequent for that word in training corpus
 - I.e., chose tag such that $p(\text{tag}|\text{word})$ is high
 - Of all the times that "use" occurred in a training corpus, what percentage was it V, what percentage N? Choose higher probability tag.
- Does it work?
 - Achieves: 90%! But we can do better:
 - How? Context: "to use": use is V; "the use of": use is N

1/13/05

CS 224S Winter 2005

50

HMM Tagger

- Intuition: Pick the most probable tag sequence for a series of words

$$\hat{t}_1^n = \operatorname{argmax}_{t_1^n} P(t_1^n | w_1^n)$$

- $\operatorname{argmax}_x f(x)$ means "the x such that $f(x)$ is maximized"
- But how to make the right-hand side operational?
- Use Bayes' rule: $P(x | y) = \frac{P(y | x)P(x)}{P(y)}$
- Substituting:

$$\hat{t}_1^n = \operatorname{argmax}_{t_1^n} \frac{P(w_1^n | t_1^n)P(t_1^n)}{P(w_1^n)}$$

1/13/05

CS 224S Winter 2005

51

HMM Tagger: fundamental equations

$$\hat{t}_1^n = \operatorname{argmax}_{t_1^n} \frac{P(w_1^n | t_1^n)P(t_1^n)}{P(w_1^n)}$$

- Since the word sequence is constant:

$$\hat{t}_1^n = \operatorname{argmax}_{t_1^n} P(w_1^n | t_1^n)P(t_1^n)$$

↑ likelihood ↑ prior

- Still too hard to compute directly

1/13/05

CS 224S Winter 2005

52

HMM Tagger: Two simplifying assumptions

$$\hat{t}_1^n = \operatorname{argmax}_{t_1^n} P(w_1^n | t_1^n) P(t_1^n)$$

- Prob of word independent of other words and their tags:

$$P(w_1^n | t_1^n) \approx \prod_{i=1}^n P(w_i | t_i)$$

- Prob of tag is only dependent on previous tag

$$P(t_1^n) \approx \prod_{i=1}^n P(t_i | t_{i-1})$$

- **Combining:**

$$\hat{t}_1^n = \operatorname{argmax}_{t_1^n} P(t_1^n | w_1^n) \approx \operatorname{argmax}_{t_1^n} \prod_{i=1}^n P(w_i | t_i) P(t_i | t_{i-1})$$

1/13/05

CS 224S Winter 2005

53

Estimating these probabilities

- **Determiners precede nouns in English, so expect P(NN|DT) to be high**

$$P(t_i | t_{i-1}) = \frac{C(t_{i-1}, t_i)}{C(t_{i-1})}$$

- In tagged 1-million word Brown corpus:

$$P(\text{NN}|\text{DT}) = C(\text{DT}, \text{NN}) / C(\text{DT}) = 56509 / 116454 = .49$$

$$P(w_i | t_i) = \frac{C(t_i, w_i)}{C(t_i)}$$

- $P(\text{is}|\text{VBZ}) = C(\text{VBZ}, \text{is}) / C(\text{VBZ}) = 10073 / 21627 = .47$

1/13/05

CS 224S Winter 2005

54

An example

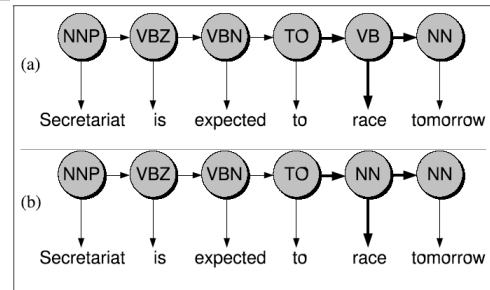
- Secretariat/NNP is/BEZ expected/VBN to/TO **race/VB** tomorrow/NR
- People/NNS continue/VB to/TO inquire/VB the/AT reason/NN for/IN the/AT **race/NN** for/IN outer/JJ space/NN

1/13/05

CS 224S Winter 2005

55

An example of two tag sequences

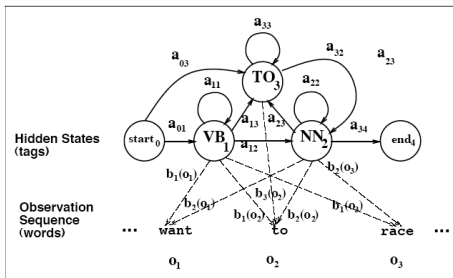


1/13/05

CS 224S Winter 2005

56

Picture of HMM



1/13/05

CS 224S Winter 2005

57

Parameters to define an HMM

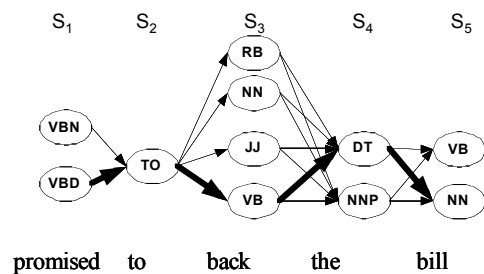
- **States:** a set of states $Q = q_1, q_2, \dots, q_N$
- **Transition probabilities:** set of probabilities $A = a_{01}, a_{02}, \dots, a_{n1}, \dots, a_{nn}$ where a_{ij} represents prob of transitioning from state i to state j .
- **Observation likelihoods:** set of probs $B = b_i(o_t)$, expressing prob of observation o_t being generated from state i .
- **Special non-emitting states.**

1/13/05

CS 224S Winter 2005

58

Viterbi Algorithm



1/13/05

CS 224S Winter 2005

59

Evaluation

- The result is compared with a manually coded "Gold Standard"
 - Typically accuracy reaches 96-97%
 - This may be compared with result for a baseline tagger (one that uses no context).
- Important: 100% is impossible even for human annotators.

1/13/05

CS 224S Winter 2005

60

Summary

- Part of speech tagging plays important role in TTS
- Most algorithms get 96-97% tag accuracy
- Not a lot of studies on whether there is remaining error rates

1/13/05

CS 224S Winter 2005

61

Word pronunciations

- Now that you've tried doing this by hand!

1/13/05

CS 224S Winter 2005

62

Lexicons and Lexical Entries

- You can explicitly give pronunciations for words
 - Each lg/dialect has its own lexicon
 - You can lookup words with
 - (lex.lookup WORD)
 - You can add entries to the current lexicon
 - (lex.add.entry NEWENTRY)
 - Entry: (WORD POS (SYL0 SYL1...))
 - Syllable: ((PHONE0 PHONE1 ...) STRESS)
 - Example:
'("cepstra" n ((k eh p) l) ((s t r aa) 0)))

1/13/05

CS 224S Winter 2005

63

Converting from words to phones

- Two methods:
 - Dictionary-based
 - Rule-based (Letter-to-sound=LTS)
- Early systems, all LTS
- MITalk was radical in having huge 10K word dictionary
- Now systems use a combination
- CMU dictionary: 127K words
 - <http://www.speech.cs.cmu.edu/cgi-bin/cmudict>

1/13/05

CS 224S Winter 2005

64

Dictionaries aren't always sufficient

- **Unknown words**
 - Seem to be linear with number of words in unseen text
 - Mostly person, company, product names
 - But also foreign words, etc.
- **So commercial systems have 3-part system:**
 - Big dictionary
 - Special code for handling names
 - Machine learned LTS system for other unknown words

1/13/05

CS 224S Winter 2005

65

Letter-to-Sound Rules

- **Festival LTS rules:**
- (LEFTCONTEXT [ITEMS] RIGHTCONTEXT = NEWITEMS)
- **Example:**
 - (# [c h] C = k)
 - (# [c h] = ch)
- **# denotes beginning of word**
- **C means all consonants**
- **Rules apply in order**
 - "christmas" pronounced with [k]
 - But word with ch followed by non-consonant pronounced [ch]
- E.g., "choice"

1/13/05

CS 224S Winter 2005

66

What about stress: practice

- **Generally**
- **Pronounced**
- **Exception**
- **Dictionary**
- **Significant**
- **Prefix**
- **Exhale**
- **Exhalation**
- **Sally**

1/13/05

CS 224S Winter 2005

67

Stress rules in LTS

- **English famously evil: one from Allen et al 1987**
- $V \rightarrow [1\text{-stress}] / X_C^* \{V\text{short } C\ C^*|V\} \{[V\text{short } C^*|V]\}$
- **Where X must contain all prefixes:**
- **Assign 1-stress to the vowel in a syllable preceding a weak syllable followed by a morpheme-final syllable containing a short vowel and 0 or more consonants (e.g. difficult)**
- **Assign 1-stress to the vowel in a syllable preceding a weak syllable followed by a morpheme-final vowel (e.g. oregano)**

1/13/05

CS 224S Winter 2005

68

Modern method: Learning LTS rules automatically

- Induce LTS from a dictionary of the language
- Black et al. 1998
- Applied to English, German, French
- Two steps: alignment and (CART-based) rule-induction

1/13/05

CS 224S Winter 2005

69

Alignment

- Letters: c h e c k e d
- Phones: ch _ eh _ k _ t
- Black et al Method 1:
 - First scatter epsilons in all possible ways to cause letters and phones to align
 - Then collect stats for $P(\text{letter}|\text{phone})$ and select best to generate new stats
 - This iterated a number of times until settles (5-6)
 - This is EM (expectation maximization) alg

1/13/05

CS 224S Winter 2005

70

Alignment

- Black et al method 2
- Hand specify which letters can be rendered as which phones
 - C goes to k/ch/s/sh
 - W goes to w/v/f, etc
- Once mapping table is created, find all valid alignments, find $p(\text{letter}|\text{phone})$, score all alignments, take best

1/13/05

CS 224S Winter 2005

71

Alignment

- Some alignments will turn out to be really bad.
- These are just the cases where pronunciation doesn't match letters:
 - Dept d ih p aa r t m ah n t
 - CMU s iy eh m y uw
 - Lieutenant l eh f t eh n ax n t (British)
- Also foreign words
- These can just be removed from alignment training

1/13/05

CS 224S Winter 2005

72

Building CART trees

- Build a CART tree for each letter in alphabet (26 plus accented) using context of +-3 letters
- # # # c h e c -> ch
- c h e c k e d -> _
- This produces 92-96% correct LETTER accuracy (58-75 word acc) for English

1/13/05

CS 224S Winter 2005

73

Improvements

- Take names out of the training data
- And acronyms
- Detect both of these separately
- And build special-purpose tools to do LTS for names and acronyms

1/13/05

CS 224S Winter 2005

74

Names

- Big problem area is names
- Names are common
 - 20% of tokens in typical newswire text will be names
 - 1987 Donnelly list (72 million households) contains about 1.5 million names
 - Personal names: McArthur, D'Angelo, Jiminez, Rajan, Raghavan, Sondhi, Xu, Hsu, Zhang, Chang, Nguyen
 - Company/Brand names: Infinit, Kmart, Cytoc, Medamicus, Inforte, Aaon, Idexx Labs, Bebe

1/13/05

CS 224S Winter 2005

75

Names

- Methods:
 - Can do morphology (Walters -> Walter, Lucasville)
 - Can write stress-shifting rules (Jordan -> Jordanian)
 - Rhyme analogy: Plotsky by analogy with Trotsky (replace tr with pl)
 - Liberman and Church: for 250K most common names, got 212K (85%) from these modified-dictionary methods, used LTS for rest.
 - Can do automatic country detection (from letter trigrams) and then do country-specific rules

1/13/05

CS 224S Winter 2005

76

(Publishable) Final Project Ideas

- Incorporating Alan Black's list of "Main errors left in Festival"
 - 1) LTS rules failing on novel forms
 - 2) Foreign proper names often fail (extend Litjens and Black 2001)
 - 3) Wrong POS in newspaper headlines
 - 4) Homograph disambig still sucks (POS not enough info for e.g., wind w ih n d/w ay n d)
 - 5) Better sentence boundary disambig, extending Palmer and Hearst.

1/13/05

CS 224S Winter 2005

77

Speaking of Final projects

- Eurospeech-2005 conference
- Big bi-annual speech conference (ASR, TTS, speaker recognition, dialogue systems, you name it)
- 4 page papers
- Submission deadline: April 8
- <http://www.interspeech2005.org/>

1/13/05

CS 224S Winter 2005

78