Token to Words

Expanding identified token to words

- numbers+type = word list
- homographs+type = words
- symbols broken down and pronounced
- unknown words: as word or letter sequence
(define (token_to_words token name)
  (cond
    ((string-matches name "[0-9]+’s") ;; e.g. 1950’s
     (item.set_feat token "token_pos" "year")
     (append
      (builtin_english_token_to_words token (string-before name "’s"))
      (list '(name "’s")(pos nnp))))
    ((string-matches name "[0-9]+-[0-9]+") ;; e.g. 12-14
     ;; just a simply word
     (builtin_english_token_to_words token name))))
Example token rule

for “$120 million”

(define (token_to_words token name)
  (cond
    ((and (string-matches name "\$[0-9,]+\(\.[0-9]+\)?")
      (string-matches (item.feat token "n.name")
        ".*illion.?"))
     (append
      (english_token_to_words token (string-after name ""))
      (list
       (item.feat token "n.name"))))
    ((and (string-matches (item.feat token "p.name")
               "\$[0-9,]+\(\.[0-9]+\)?")
       (string-matches name ".*illion.?"))
     (list "dollars"))
    (t
     (english_token_to_words token name))))
Text modes

If we know the type of text being synthesizing (e.g. email, Latex, HTML) we can tailor the processing.

□ mode specific tokenizing
□ using tokens to direct synthesis (emphasis, selecting voices etc.)
□ mode specific lexical items.
□ mode specific syntactic forms.

Explicit markup and/or Custom models
Festival text modes

Customizable modes for synthesis.
Each mode can have

- A (Unix) filter program to extract/delete information
- An `init_function` on entering the mode.
- An `exit_function` on exiting the mode.
An example text mode for email

A filter to extract , from line, subject and body from email message

#!/bin/sh
# Email filter for Festival tts mode
# usage: email_filter mail_message >tidied_mail_message
grep "^From: " $1
echo
grep "^Subject: " $1
echo
sed '1,/^$/ d' $1
setup mode specific token functions

(define (email_init_func)
  "Called on starting email text mode."
  (set! email_previous_t2w_func token_to_words)
  (set! english_token_to_words email_token_to_words)
  (set! token_to_words email_token_to_words))

(define (email_exit_func)
  "Called on exit email text mode."
  (set! english_token_to_words email_previous_t2w_func)
  (set! token_to_words email_previous_t2w_func))
(define (email_token_to_words token name)
  "Email specific token to word rules."
  (cond
    ((string-matches name "<.*@.*>")
     (append
      (email_previous_t2w_func token
       (string-after (string-before name "@") "<"))
      (cons
       "at"
       (email_previous_t2w_func token
        (string-before (string-after name "@") ">"))))))
((and (string-matches name ">")
  (string-matches (item.feat
defect "whitespace")
    "[ \t\n]*\n *")
    (voice_don_diphone)
  nil ;; return nothing to say
)
(t ;; for all other cases
  (if (string-matches (item.feat
defect "whitespace")
    ".*\n[ \n]*")
    (voice_rab_diphone))
  (email_previous_t2w_func token name))))
}
(set! tts_text_modes
  (cons
   (list
     'email ;; mode name
     (list ;; email mode params
       (list 'init_func email_init_func)
       (list 'exit_func email_exit_func)
       '(filter "email_filter")))
   tts_text_modes))
Alan W. Black writes on 27 November 1996:

> I’m looking for a demo mail message for Festival, but can’t seem to find any suitable. It should at least have some quoted text, and have some interesting tokens like a URL or such like.
>
> Alan

Well I’m not sure exactly what you mean but awb@cogsci.ed.ac.uk has an interesting home page at http://www.cstr.ed.ac.uk/~awb/ which might be what you’re looking for.

Alan

> PS. Will you attend the course?

I hope so

by for now
Reading addresses

Smith, Bobbie Q, 3337 St Laurence St, Fort Worth, TX 71611-5484, (817)839-3689
Anderson, W, 445 Sycamore Way NE, Lincoln, NE 98125-5108, (212)404-9988
Mark-up languages

- Building special text modes might be too difficult
- Need general method for general markup:
  - breaks, voice changing
  - pronunciations, date/time identifies
- All synthesizers include this but are incompatible
- Proposal of *general* method:
  - SGML/XML based
  - *basic* tags only
  - cf. JSML, VoiceXML
The boy saw the girl in the park with the telescope. The boy saw the girl in the park with the telescope.

Some English first and then some Spanish.
<LANGUAGE ID="SPANISH">Hola amigos.</LANGUAGE>
<LANGUAGE ID="NEPALI">Namaste</LANGUAGE>

Good morning My name is Stuart, which is spelled though some people pronounce it My telephone number is 2787.

I used to work in Buckleuch Place, but no one can pronounce that.

By the way, my telephone number is actually
SABLE: for marking emphasis

What will the weather be like today in Boston?
It will be <emph>rainy</emph> today in Boston.

When will it rain in Boston?
It will be rainy <emph>today</emph> in Boston.

Where will it rain today?
It will be rainy today in <emph>Boston</emph>.
But we need a richer markup

- SABLE is quite limited:
  - Now embodied in SSML, VoiceXML and JSML

- Concept to speech is richer:
  - translation and generation systems
  - Syntactic, Semantic
  - Anaphoric, Rhetorical, Speech act etc.

- Mark up should be:
  - abstract not low-level
  - e.g. type=question not
  - pitch rise at end
Data: four domains

nantc: press-wire news data
classifieds: real estate ads from on-line newspapers
pc110: palmtop mailing list (e-mail like)
rfr: rec.food.recipes USENET messages

<table>
<thead>
<tr>
<th>Corpus</th>
<th>nantc</th>
<th>ads</th>
<th>pc110</th>
<th>rfr</th>
</tr>
</thead>
<tbody>
<tr>
<td>total # tokens</td>
<td>4.3m</td>
<td>415k</td>
<td>264k</td>
<td>209k</td>
</tr>
<tr>
<td># NSWs</td>
<td>377k</td>
<td>180k</td>
<td>72k</td>
<td>46k</td>
</tr>
<tr>
<td>% NSW</td>
<td>8.8%</td>
<td>43.4</td>
<td>27.3</td>
<td>22.0</td>
</tr>
<tr>
<td>Token</td>
<td>Description</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
<td>----------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXPN</td>
<td>abbreviation, contractions</td>
<td>adv, N.Y, mph, gov’t</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSEQ</td>
<td>letter sequence</td>
<td>CIA, D.C, CDs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ASWD</td>
<td>read as word</td>
<td>CAT, proper names</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MSPL</td>
<td>misspelling</td>
<td>geography</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NUM</td>
<td>number (cardinal)</td>
<td>12, 45, 1/2, 0.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NORD</td>
<td>number (ordinal)</td>
<td>May 7, 3rd, Bill Gates III</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NTEL</td>
<td>telephone (or part of)</td>
<td>212 555-4523</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NDIG</td>
<td>number as digits</td>
<td>Room 101,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NIDE</td>
<td>identifier</td>
<td>747, 386, I5, PC110, 3A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NADDR</td>
<td>number as street address</td>
<td>5000 Pennsylvania, 4523 Forbes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NZIP</td>
<td>zip code or PO Box</td>
<td>91020</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NTIME</td>
<td>a (compound) time</td>
<td>3.20, 11:45</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NDATE</td>
<td>a (compound) date</td>
<td>2/2/99, 14/03/87 (or US) 03/14/87</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NYER</td>
<td>year(s)</td>
<td>1998 80s 1900s 2003</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MONEY</td>
<td>money (US or otherwise)</td>
<td>$3.45 HK$300, Y20,000, $200K</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMONY</td>
<td>money tr/m/billions</td>
<td>$3.45 billion</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRCT</td>
<td>percentage</td>
<td>75%, 3.4%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLNT</td>
<td>not spoken, word boundary</td>
<td>word boundary or emphasis character: M.bath, KENT*REALTY, <em>really</em>, ***Added</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PUNC</td>
<td>not spoken, phrase boundary</td>
<td>non-standard punctuation: “...” in DECIDE...Year, “<em><strong>” in $99,9K</strong></em>Whites</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FNSP</td>
<td>funny spelling</td>
<td>sllooooww, sh*t</td>
<td></td>
<td></td>
</tr>
<tr>
<td>URL</td>
<td>url, pathname or email</td>
<td><a href="http://apj.co.uk">http://apj.co.uk</a>, /usr/local, <a href="mailto:phj@teleport.com">phj@teleport.com</a></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NONE</td>
<td>token should be ignored</td>
<td>ascii art, formatting junk</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# Data: NSW distributions

|  | Domains |  |
|---|---|---|---|---|
|  | nantc | classifieds | pc110 | rfr |
| ASWD | 83.49 | 28.64 | 64.60 | 72.36 |
| LSEQ | 9.10 | 3.00 | 22.60 | 2.11 |
| EXPN | 7.41 | 68.36 | 12.80 | 25.53 |

|  | Domains |  |
|---|---|---|---|---|
|  | nantc | classifieds | pc110 | rfr |
| NUM | 66.11 | 58.26 | 43.77 | 97.90 |
| NYER | 19.06 | 0.70 | 0.51 | 0.27 |
| NORD | 9.37 | 3.37 | 4.45 | 0.11 |
| NIDE | 2.24 | 5.83 | 37.41 | 0.47 |
| NTEL | 1.25 | 25.92 | 1.32 | 0.02 |
Hand labeling

- Each NSW presented in context
  - Three words either side

- One letter choice of TAG
  - or explicit expansion
  - splits “WinNT” \( \rightarrow \) “Win” “NT”

- Test of inter-labeler agreement
  - 3 labelers nantc, 2268 samples, \( \kappa = 0.81 \)
  - 9 labelers ads, 622 samples, \( \kappa = 0.84 \)

- Labeling held as XML markup

Today I bought a Sony

\(<\text{W NSW="LSEQ"> NP-F530</text></W><\text{W NSW="SPLT"></text><text{WS NSW="NUM"> 1350</text><text{WS NSW="EXPN">maH.</text><text{WS}></text></text></text>

Like your

\(<\text{W NSW="NIDE"> 550</text></W> it is slightly larger than the native

\(<\text{W NSW="LSEQ"> IBM</text></W> battery pack. It’s been

now on it’s first charge - I am charging in the

\(<\text{W NSW="LSEQ"> PC110.

</text>
Can we find NSWs?

- Tokens not in lexicon
- Plus
  - single character tokens
  - “punctuation”
  - common abbreviations (in lexicon)
- Misses homographic abbreviations/standard words
  - “sun”, “Jan”
  - also domain specific ones, “kit” and “named”

<table>
<thead>
<tr>
<th>Domain Dependent?</th>
<th>Detection Algorithm</th>
<th>Precision//Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>nantc</td>
</tr>
<tr>
<td>No</td>
<td>non-lexical</td>
<td>55/79</td>
</tr>
<tr>
<td>No</td>
<td>+ sct + abbrevs</td>
<td>44/93</td>
</tr>
<tr>
<td>Yes</td>
<td>++ abbrevs</td>
<td>39/93</td>
</tr>
</tbody>
</table>
Theoretical models

- Source-channel model:

\[ \hat{w} = \arg\max p(w|o) \]  \hspace{1cm} (1)

\[ = \arg\max p(o|w)p(w) \]  \hspace{1cm} (2)

- Direct approach:

\[ \hat{w} = \arg\max p(w|o) \]  \hspace{1cm} (3)
Architecture

pls wash your WS99 coff.
cup w/n-grams :)

Text
Tokenizer
Tokens
Split Tokens
Splitter
Tagged Tokens
Classifier
Word Lattices
Tag Expanders
Language Model
Best Words
Splitting

- whitespace separated tokens isn’t fine enough
- Further splitting is required:
  
  $1500\text{km} \rightarrow 1500\ \text{km}$  
  $\text{and/or} \rightarrow \text{and} / \text{or}$  
  $\text{WinNT} \rightarrow \text{Win NT}$

- Ideally deterministic, domain independent
- Simple regular expressions
Splitting

<table>
<thead>
<tr>
<th></th>
<th>NANTC</th>
<th>classifieds</th>
<th>pc110</th>
<th>RFR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recall</td>
<td>98.89</td>
<td>94.96</td>
<td>87.66</td>
<td>98.88</td>
</tr>
<tr>
<td>Precision</td>
<td>74.41</td>
<td>87.32</td>
<td>81.68</td>
<td>89.51</td>
</tr>
<tr>
<td>Split Correct</td>
<td>92.54</td>
<td>85.99</td>
<td>74.11</td>
<td>89.54</td>
</tr>
<tr>
<td>Total Correct</td>
<td>98.45</td>
<td>95.19</td>
<td>92.97</td>
<td>98.40</td>
</tr>
</tbody>
</table>

Misses:
- ESANDWICH, 3400sq.ft, xjack, 11/2

“False” positives:
- 1-3pm, w/d, R-Ariz, PC-110
Tag classification

Assign EXPN, NUM, NORD etc to NSWs:

□ domain independent features:
  – all caps, no vowels, numeric etc.

□ domain dependent features:
  – alphabetic sub-classifier for EXPN, ASWD and LSEQ

Tested CART and Maximum Entropy models
Alphabetic tag sub-classification

NSW tag $t$ for alphabetic observations $o$

NATO: ASWD, PCMCIA: LSEQ, frplc: EXPN

\[ p(t|o) = \frac{p_t(o|t)p(t)}{p(o)} \]

where $t \in \{ASWD, LSEQ, EXPN\}$.

- $p_t(o|t)$ estimated by a letter trigram model
  \[ p_t(o|t) = \prod_{i=1}^{N} p(l_i|l_{i-1}, l_{i-2}), \]

- $p(t)$ prior from data or uniform
- normalized by
  \[ p(o) = \sum_t p_t(o|t)p(t) \]
## Alphabetic tag sub-classification

LLM features are fed into overall classifier through 6 features

| Token | $p(\text{ASWD}|o)$ | $p(\text{LSEQ}|o)$ | $p(\text{EXPN}|o)$ | $p_{\text{max}}$ | $t_{\text{max}}$ | diff 1-2 |
|-------|---------------------|--------------------|--------------------|-----------------|-----------------|----------|
| mb    | 0.0001              | 0.0038             | **0.9962**         | 0.9962          | EXPN            | 0.9924   |
| Grt   | 0.0024              | 0.0000             | **0.9976**         | 0.9976          | EXPN            | 0.9952   |
| NBA   | 0.0017              | **0.9983**         | 0.0000             | 0.9983          | LSEQ            | 0.9966   |
| Cust  | **0.5456**          | 0.0000             | 0.4544             | 0.5456          | ASWD            | 0.0912   |
Using LLM features alone

<table>
<thead>
<tr>
<th>Domain</th>
<th>NANTC</th>
<th>ads</th>
<th>pc110</th>
<th>RFR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>83.9[ASWD]</td>
<td>80.53[EXPN]</td>
<td>63.77[ASWD]</td>
<td>69.98[ASWD]</td>
</tr>
<tr>
<td>Uniform</td>
<td>88.92</td>
<td>98.5</td>
<td>90.83</td>
<td>97.36</td>
</tr>
<tr>
<td>Unigram</td>
<td>95.72</td>
<td>98.74</td>
<td>92.27</td>
<td>97.92</td>
</tr>
</tbody>
</table>
# Full tag classification

<table>
<thead>
<tr>
<th>Accuracy</th>
<th>NANTC</th>
<th>ads</th>
<th>pc110</th>
<th>RFR</th>
</tr>
</thead>
<tbody>
<tr>
<td>No LLM Feats</td>
<td>97.7</td>
<td>92.7</td>
<td>90.9</td>
<td>97.3</td>
</tr>
<tr>
<td>All LLM feats</td>
<td>98.1</td>
<td>93.5</td>
<td>91.8</td>
<td>96.8</td>
</tr>
</tbody>
</table>
Algorithmic expansions

- SLNT, NONE: expand to nothing
- ASWD, PUNC: expand to themselves
- LSEQ: as letters
- NUM: expands integers, floats, roman to string of words
- NORD: expands to ordinals
- NYER: as number pairs (except 00 and 000)
- NADDR, NZIP, NTEL, NDATE, NTIME: specific expanders
- NIDE: letters as letters, numbers as pairs
- MONEY, BMONY: as currency
- PRCT: as NUM with “percent”
- EMAIL, URL: treated ASWD (though should not be)
- MSPL, FNSP, OTHER: treated ASWD (though should not be), never predicted
EXPN expansions

How to find the expansion of an abbreviation:
– “wbfpl” → “wood burning fireplace”
– “BR” → “bedroom”
– “Fl” → “Florida” or “Floor”

Not simple lists:
– 32 different abbrevs for “bedroom”
– Productive: SQH, SB, Newingtn

In *supervised* case use labelled expansions
error rate:

without language model 6.7%
without language model 4.8%
What about *unsupervised case*?

- Assume expanded form somewhere in corpus
- Build letter deletion model from known EXPNs
  - CART predicts prob of letter deletion (88% accuracy)
  - convert CART to WFST
  - compute
    \[
    [SW \circ A \circ NSW]^{-1}
    \]
    (4)
  - build a WFST for weighted lattice of possible expansions of a potential NSW.
Unsupervised prediction of expansions

1. All singleton SWs + bigrams $> 3$ times: $33\%$ error rate
2. as 1 plus standard abbrevs: $24\%$
3. as 2 but
   - expand on training set
   - use language model
   - select most frequent expansion alone: $19.9\%$
4. as 3 but
   - select best 2 and reestimate probs: $19.9\%$
Further issues in EXPN expansions

1. Need better model of expansion:

   OEPN   OPEN PERENNIAL
   DALLIN  DAVID ALLAIN
   MASHPEE MARSH PROPERTIES
   SEAVIEW  SEASONAL VIEWS
   WIGET   WITHGUESTS

2. Current ignoring case (unsupervised)

3. What is *likely* to be abbreviated
   \(- p(t|w): BTW \rightarrow {\text{because the windows}}\)
Language Modeling

- Grand schemes:
  - trigger models
  - maximum entropy

- Simple smoothed backed off trigrams

- Applied to pseudo-words:
  ... lives at 123 Norman St. ...
  ... lives at NADDR Norman St. ...
Baseline results

**LDC tools**: LDC text conditioning tools

**Festival**: 1.4.0 released text analyzer

<table>
<thead>
<tr>
<th></th>
<th>LDC tools</th>
<th>Festival</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TER</td>
<td>WER</td>
</tr>
<tr>
<td>nantc</td>
<td>–</td>
<td>2.88</td>
</tr>
<tr>
<td>classifieds</td>
<td>–</td>
<td>30.81</td>
</tr>
<tr>
<td>pc110</td>
<td>–</td>
<td>22.36</td>
</tr>
<tr>
<td>rfr</td>
<td>–</td>
<td>9.06</td>
</tr>
</tbody>
</table>
Domain dependent model

- domain independent splitter
- CART tag classifier with letter language model features
- EXPNs by WFST
- Language model

<table>
<thead>
<tr>
<th></th>
<th>festival</th>
<th></th>
<th>m4</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TER</td>
<td>WER</td>
<td>TER</td>
<td>WER</td>
</tr>
<tr>
<td>nantc</td>
<td>1.00</td>
<td>1.38</td>
<td>0.39</td>
<td>0.82</td>
</tr>
<tr>
<td>classifieds</td>
<td>30.09</td>
<td>33.48</td>
<td>7.00</td>
<td>9.71</td>
</tr>
<tr>
<td>pc110</td>
<td>14.37</td>
<td>32.62</td>
<td>3.66</td>
<td>9.25</td>
</tr>
<tr>
<td>rfr</td>
<td>6.28</td>
<td>16.19</td>
<td>0.94</td>
<td>2.07</td>
</tr>
</tbody>
</table>
Removing components

**m4.nolm**: no language model (most prob EXPN)

**m4.noef**: no letter language models feats

**m4.noeflm**: no LM and no LLM feats

<table>
<thead>
<tr>
<th></th>
<th>m4</th>
<th>m4.nolm</th>
<th>m4.noef</th>
<th>m4.noeflm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TER</td>
<td>WER</td>
<td>TER</td>
<td>WER</td>
</tr>
<tr>
<td>nantc</td>
<td>0.39</td>
<td>0.82</td>
<td>0.39</td>
<td>0.81</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>classifieds</td>
<td>7.00</td>
<td>9.71</td>
<td>6.82</td>
<td>9.70</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rfr</td>
<td>0.94</td>
<td>2.07</td>
<td>0.93</td>
<td>2.06</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Giving truth

**m4.nosplt:** uses hand labeled splits

**m4.nost:** uses hand labeled splits and actual tags

<table>
<thead>
<tr>
<th></th>
<th>m4</th>
<th>m4.nosplt</th>
<th>m4.nost</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TER</td>
<td>WER</td>
<td>TER</td>
</tr>
<tr>
<td>nantc</td>
<td>0.39</td>
<td>0.82</td>
<td>0.20</td>
</tr>
<tr>
<td>classifieds</td>
<td>7.00</td>
<td>9.71</td>
<td>5.40</td>
</tr>
<tr>
<td>pc110</td>
<td>3.66</td>
<td>9.25</td>
<td>2.58</td>
</tr>
<tr>
<td>rfr</td>
<td>0.94</td>
<td>2.07</td>
<td>0.59</td>
</tr>
</tbody>
</table>
Cross-domain models

**m4.domin**: nantc models

**m4.dominE**: nantc models with domain EXPNs

<table>
<thead>
<tr>
<th></th>
<th>festival</th>
<th>m4</th>
<th>m4.domin</th>
<th>m4.dominE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TER</td>
<td>WER</td>
<td>TER</td>
<td>WER</td>
</tr>
<tr>
<td>nantc</td>
<td>1.00</td>
<td>1.38</td>
<td>0.39</td>
<td>0.82</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.39</td>
<td>0.82</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.39</td>
<td>0.82</td>
</tr>
<tr>
<td>classifieds</td>
<td>30.09</td>
<td>33.48</td>
<td>7.00</td>
<td>9.71</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>25.20</td>
<td>29.11</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>19.69</td>
<td>21.18</td>
</tr>
<tr>
<td>pc110</td>
<td>14.37</td>
<td>32.62</td>
<td>3.66</td>
<td>9.25</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>12.35</td>
<td>18.69</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>12.09</td>
<td>18.07</td>
</tr>
<tr>
<td>rfr</td>
<td>6.28</td>
<td>16.19</td>
<td>0.94</td>
<td>2.07</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2.71</td>
<td>4.66</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2.32</td>
<td>4.14</td>
</tr>
</tbody>
</table>
Unsupervised domain models

Building models from unlabeled data

- Label tokens with nantc CART tag classifier
- Relabel alphabetics with best LLM prediction
- Build EXPN expander from plain text and labeled EXPNs
- Build words with best EXPN expansion
- Build LM from full expanded words
- Run with multiple EXPNs and LM to choose

<table>
<thead>
<tr>
<th></th>
<th>TER</th>
<th>WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>m4</td>
<td>7.00</td>
<td>9.71</td>
</tr>
<tr>
<td>us1.lm</td>
<td>12.50</td>
<td>13.40</td>
</tr>
<tr>
<td>us1.nolm</td>
<td>12.64</td>
<td>13.50</td>
</tr>
<tr>
<td>us2.EXPNlist</td>
<td>10.58</td>
<td>13.51</td>
</tr>
<tr>
<td>m4.dominE</td>
<td>19.69</td>
<td>21.18</td>
</tr>
</tbody>
</table>
NSW model for new domains

- Models for specific domains
- Standard text analyzers fail
- Can build models from unlabeled data

57 ST E/1st & 2nd Ave Huge
drmn 1 BR 750+ sf, lots of sun &
clsts. Sundeck & Indry facils. Askg
$187K, maint $868, util
incl. Call Bkr Peter 914-428-9054.
Results

- Marked up databases
- Tools to help label databases
- Tools and methods for building models
- 4 domain models
- Text expander better than LDC or Festival
- Tools and methods for building unsupervised models
But what if there are no spaces?

- Chinese, Japanese etc. don’t use whitespace
- But still need to tokenize
Some techniques

Requires lexicon of words

☐ Take longest match in lexicon (that gives partition)

☐ or find

$$\hat{w} = \arg\max_w p(w|)$$ (5)

☐ Lattice of all possible partitions and find most probable
Number pronunciation

In languages with gender, declensions etc.

1 niño → un niño (one boy)
1 niña → una niña (one girl)

1 hermano → un hermano (one brother)
1 hermana → una hermana (one sister)

Can’t just look at a/o ending letter

1 país → un país (one country)
1 ra’iz → una ra’iz (one root)

Slavic languages have many variations for numbers making it harder.
End of Text Analysis

From strings of characters to lists of words

☐ Tokenize string of chars
☐ Chunk into utterance sized chunks
☐ Identify token types (homographs, numbers etc)
☐ Expand tokens with token to word rules