Speech Processing 15-492/18-492

Speech Synthesis
Pronunciation
Letter to Sound rules
Speech Synthesis

- **Linguistic Analysis**
  - Pronunciations
  - Prosody
Part of Speech Tagging

- **Find the most likely tag for each word**
  - Most words only have one tag (92% correct)

- **Context often defines tag type**
  - “The project” vs “To project”

- **Use HMM Part of Speech tagger**
  - But need data to train it (English PennTreeBank)
Poor Man’s PoS Tagger

- **Hand list “function” word types**
  - (determiners a an the this)
  - (conjunctions and or but)
  - (pp in on to)
  - (content everything else)

- **Better than nothing**
  - Easy to do on new languages
Pronunciation Lexicon

- **List of words and their pronunciation**
  - (“pencil” n (p eh1 n s ih l))
  - (“table” n (t ey1 b ax l))

- **Need the right phoneme set**

- **Need other information**
  - Part of speech
  - Lexical stress
  - Other information (Tone, Lexical accent …)
  - Syllable boundaries
Homograph Representation

- **Must distinguish different pronunciations**
  - (“project” n (p r aa1 jh eh k t))
  - (“project” v (p r ax jh eh1 k t))
  - (“bass” n_music (b ey1 s))
  - (“bass” n_fish (b ae1 s))

- **ASR multiple pronunciations**
  - (“route” n (r uw t))
  - (“route(2)” n (r aw t))
Pronunciation of Unknown Words

- How do you pronounce new words?
- 4% of tokens (in news) are new.
- You can’t synthesis them without pronunciations.
- You can’t recognize them without pronunciations.
- Letter-to-Sounds rules.
- Grapheme-to-Phoneme rules.
Hand written rules

- \([\text{LeftContext}] X [\text{RightContext}] -> Y\)
- e.g.
  - \(c [h \ r] -> k\)
  - \(c [h] -> ch\)
  - \(c [i] -> s\)
  - \(c -> k\)
LTS: Machine Learning Techniques

- **Need an existing lexicon**
  - Pronunciations: words and phones
  - But different number of letters and phones

- **Need an alignment**
  - Between letters and phones
  - checked -> ch eh k t
LTS: alignment

- **checked** -> **ch eh k t**

<table>
<thead>
<tr>
<th>c</th>
<th>h</th>
<th>e</th>
<th>c</th>
<th>k</th>
<th>e</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td>ch</td>
<td>_</td>
<td>eh</td>
<td>k</td>
<td>_</td>
<td>_</td>
<td>t</td>
</tr>
</tbody>
</table>

- **Some letters go to nothing**

- **Some letters go to two phones**
  - **box** -> **b aa k-s**
  - **table** -> **t ey b ax-l -**
Find alignment automatically

- **Epsilon scattering**
  - Find all possible alignments
  - Estimate $p(L,P)$ on each alignment
  - Find most probable alignment

- **Hand seed**
  - Hand specify allowable pairs
  - Estimate $p(L,P)$ on each possible alignment
  - Find most probable alignment

- **Statistical Machine Translation (IBM model 1)**
  - Estimate $p(L,P)$ on each possible alignment
  - Find most probable alignment
Not everything aligns

- 0, 1, and 2 letter cases
  - e -> epsilon “moved”
  - x -> k-s, g-z “box” “example”
  - e -> y-uw “askew”

- Some alignments aren’t sensible
  - dept -> d ih p aa r t m ax n t
  - cmu -> s iy eh m y uw
Training LTS models

- Use CART trees
  - One model for each letter
- Predict phone (epsilon, phone, dual phone)
  - From letter 3-context (and POS)
- # # # c h e c c -> ch
- # # c h e c k -> _
- # c h e c k e -> eh
- c h e c k e d -> k
LTS results

- **Split lexicon into train/test 90%/10%**
  - i.e. every tenth entry is extracted for testing

<table>
<thead>
<tr>
<th>Lexicon</th>
<th>Letter Acc</th>
<th>Word Acc</th>
</tr>
</thead>
<tbody>
<tr>
<td>OALD</td>
<td>95.80%</td>
<td>75.56%</td>
</tr>
<tr>
<td>CMUDICT</td>
<td>91.99%</td>
<td>57.80%</td>
</tr>
<tr>
<td>BRULEX</td>
<td>99.00%</td>
<td>93.03%</td>
</tr>
<tr>
<td>DE-CELEX</td>
<td>98.79%</td>
<td>89.38%</td>
</tr>
<tr>
<td>Thai</td>
<td>95.60%</td>
<td>68.76%</td>
</tr>
</tbody>
</table>
For letter V:
if (n.name is v)
    return _
if (n.name is #)
    if (p.p.name is t)
        return f
    return v
if (n.name is s)
    if (p.p.p.name is n)
        return f
    return v
return v
But we need more than phones

- **What about lexical stress**
  - \( p r \ a a1 j \ eh \ k t \) -> \( p r \ a a j \ eh1 k t \)

- **Two possibilities**
  - A separate prediction model
  - Join model – introduce \( eh/eh1 \) (BETTER)

<table>
<thead>
<tr>
<th></th>
<th>LTP+S</th>
<th>LTPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>L no S</td>
<td>96.36%</td>
<td>96.27%</td>
</tr>
<tr>
<td>Letter</td>
<td>---</td>
<td>95.80%</td>
</tr>
<tr>
<td>W no S</td>
<td>76.92%</td>
<td>74.69%</td>
</tr>
<tr>
<td>Word</td>
<td>63.68%</td>
<td>74.56%</td>
</tr>
</tbody>
</table>
Does it really work

- **40K words from Time Magazine**
  - 1775 (4.6%) not in OALD
  - LTS gets 70% correct (test set was 74%)

<table>
<thead>
<tr>
<th></th>
<th>Occurs</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Names</td>
<td>1360</td>
<td>76.6</td>
</tr>
<tr>
<td>Unknown</td>
<td>351</td>
<td>19.8</td>
</tr>
<tr>
<td>US Spelling</td>
<td>57</td>
<td>3.2</td>
</tr>
<tr>
<td>Typos</td>
<td>7</td>
<td>0.4</td>
</tr>
</tbody>
</table>
Dialect Lexicons

- **Need different lexicons for different dialects**
  - US, UK, Indian, Australia, Europeans

- **Build dialect independent lexicons**
  - **Dialect independent vowels** ("key-vowels")
    - The vowel in **coffee** and **conference**
    - Map to **aa** in US, and **o** in the UK
  - **Post-vocalic r** in UK English
    - Car -> k aa
  - **Specific words**
    - Leisure, route, tortoise, poem
Post-lexical Rules

- **Sometime you need context**
- “the” as dh ax or dh iy
  - The banana and The apple
- **R-insertion in UK English**
  - Car door vs car alarm
- **Liaison in French**
  - Petit vs Petit ami
Summary

★ Linguistic analysis
  - Part of speech tagging
  - Pronunciation
    ☒ Phones, stress, (syllables)
    ☒ Letter to sound rules
  - Post lexical rules