Speech II

Larry Rudolph

clearly speech is import for mobile applications. It is also important for "human-centric" applications. We thus spend two lectures on the topic. Unfortunately, we will not have time for a full problem set, but that does not mean this is not important. It only means that I miss allocated problem set assignments.

Speech Recognition

- Audio Wave Form => divide into spectral landmarks
- Phonemes => divide into words in vocabulary
- Words => divide into sentences in grammar
- Sentences => satisfy dialog management
- *These divisions are probabilistic and ambiguous but get refined (less ambiguity as progress from step to step.*

This is a bit of review. Each of the main processing steps reduces the ambiguity. This is a valuable trick that is exploited in many "natural" human communication recognition algorithms.

Clearly, the smaller the set of phones, words, or valid sentences, the more the ambiguity is reduced. Another aid is the "distance" between the valid words. It is probably easier to distinguish "affirmative" from "negative" than it is to distinguish "yes" from "no"

The main challenge for one building a speech recognition system is to find a convient way to define these sets.

Speaker Independent; Domain Dependent

- What is a domain?
  - a vocabulary (words)
  - sentences
- How to define words?
  - English spelling and pronunciation
- How to define sentences
  - Grammar

defining phonemes and words in general is a solved problem. There are plenty of online dictionaries. But, a huge dictionary means lots of mistakes.

Grammar

- What is a grammar?
  - a set of terminals
    - A, B, ...
  - a set of non-terminals $a, b, c, \ldots$
  - a set of rules or productions
    - a == B | b A
    - b == a | NULL
    - a sample sentence: B A A A
    - a --> b A --> a A --> b A A --> a A A ...
- Can you explain this to Grandma?
  - would probably use examples

Grammars are taught in computer science as part of a course in computability, programming languages, or compilers. Linguists also use them frequently. This is a very small and tiny example.

A more user-friendly Grammar

- Attributes
  - think of them as: terminals
    - actually, a non-terminal that goes to a terminal
  - For example
    - A set of terminals: lights, microwave, toaster, vcr, tv
    - These are all "objects"
    - So, "object" would be an attribute
  - Another example
    - dining room, living room, kitchen
    - "room" is the attribute

Rules

- Speechbuilder, an SLS tool, calls them "actions"
  - No complicated productions
- Each action is an example sentence
  - Sentence contains
    - an "action" terminal
    - zero or more attributes
    - optional words
  - E.g. Turn on the lights
    - "lights" is an example of an "object" attribute
    - "on" is an example of an "onoff" attribute
    - "turn" is an "action"

Speechbuilder is a simple web-based tool that lets one build grammars in an easier way by defining most of the defaults. Not only that, the vocabulary is also implicitly defined.

Example after reduction

All sentences for action turn	Action: turn — What gets sent to application
turn all the lights off	action=turn&frame=(object=lights, onoff=off)
turn off all the lights	action=turn&frame=(object=lights, onoff=off)
can you turn all the lights off	action=turn&frame=(object=lights, onoff=off)
can you turn off the living room lights	action=turn&frame=(object=lights, onoff=off,room=living+room)
can you turn off all the lights	action=turn&frame=(object=lights, onoff=off)
can you turn off the lights in the living room	action=turn&frame=(object=lights, onoff=off,room=living+room)
can you turn the living room lights off	action=turn&frame=(object=lights, onoff=off,room=living+room)
turn off the living room lights	action=turn&frame=(object=lights, onoff=off,room=living+room)
can you turn the lights in the living room off	action=turn&frame=(object=lights, onoff=off,room=living+room)
turn off the lights in the living room	action=turn&frame=(object=lights, onoff=off,room=living+room)
turn the living room lights off	action=turn&frame=(object=lights, onoff=off,room=living+room)

Given an example sentence, look for keywords in the sentence. They are examples or instances of actions or objects. all the other words get skipped over.

Domain XML example

```
<class name="object" type="Key">
    <entry>(television | tv) {television}</entry>
    <entry>lights</entry>
    <entry>microwave</entry>
    <entry>toaster</entry>
    <entry>v c r {VCR}</entry>
</class>
```

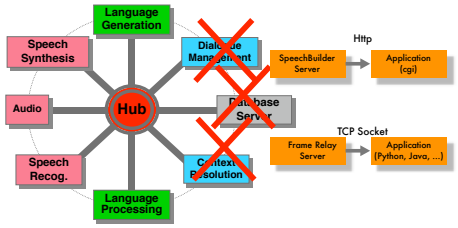
Domain XML

```
<class name="onoff" type="Key">
   <entry>lit {on}</entry>
   <entry>off</entry>
   <entry>on</entry>
</class>
<class name="turn" type="Action">
   <entry>[can you] [please] turn all the lights off</entry>
   <entry>[can you] [please] turn off all the lights</entry>
   <entry>[can you] [please] turn off the (living room lights | lights in the living room)</entry>
   <entry>[can you] [please] turn the (living room lights | lights in the living room) off</entry>
</class>
<class name="status" type="Action">
   <entry>([can you] [please] tell me | do you know) (what | which) lights are on</entry>
   <entry>([can you] [please] tell me | do you know) if the (lights in the kitchen | kitchen lights) are on</entry>
   <entry>(is | are) the (dining room television | tv in the living room) On or Off</entry>
   <entry>(is | are) the (dining room television | tv in the living room) on</entry>
</class>
<class name="good_bye" type="Action">
   <entry>good bye</entry>
   <entry>later</entry>
</class>
<class name="room" type="Key">
   <entry>dining room</entry>
   <entry>kitchen</entry>
   <entry>living room</entry>
</class>
```

Speechbuilder

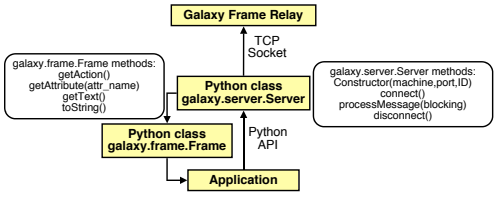
- Galaxy is the speech recognition system
- Speechbuilder is a tool to develop a domain for galaxy
- Real speech recognizers take a lot of work and detailed knowledge of all the components.
- Speechbuilder is great for prototyping

Galaxy's Components



Speechbuilder API

- Galaxy meaning representation provided through frame relay
- Applications connect via TCP sockets
- API provided in Python, Java, Perl



"Goals/Peebles" Approach

- Steve Ward has a research effort in pervasive computing (will discuss later)
- Package up: audio file + grammar
  - send to server and get back parse
- Process the "parse" locally

Simple Speech Recognition

Larry Rudolph

Speechbuilder +/-

- Speechbuilder is a great tool, but
  - Grammar is limited
  - Forces use of “action” and “attribute”
  - “Domain” has lots of components
    - complicated compile of domain pieces -- vocabulary, natural language, scoring

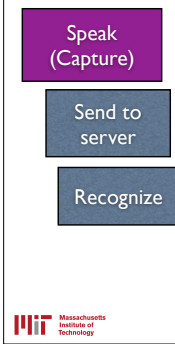
Simpler than S-Builder

- Simple:
  - send audio to server
  - receive string in the domain
  - but what about domain?
- open a connection with server w/ domain

Streaming Audio

- Long process:
  - each step can be seconds.
- Streaming can speed things up (a lot)

Streaming Audio



- Start sending audio before user is done speaking
- Start recognition before audio done arriving
- Still must wait for return

Streaming recognition

- Recall, recognition proceedings in several stages:
  - waveform to phonemes
  - phonemes to words
  - words to sentences
  - Natural language filtering

Much of this can be pipelined

NL requires whole sentence

Speed up process

- Do not do natural language filtering
- Do not do very limited vocabulary
  - (what should one do with extraneous words -- fastest is to recognize them since ignoring them slows down parsing)

more speed

- pipeline whole process.
- what about grammar parse?
  - do a Virturbi search
  - get back confidence levels

This needs lots of comments.....to be filled in soon