Speech Processing 11-492/18-492

Spoken Dialog Systems
Error handling
Evaluation
Deployment
Basic steps for machine conversation:

- Take speech to text (ASR)
- Extract meaning from text (Parsing)
- Interpret meaning, decide what to do next (DM)
- Generate text to convey decision (NLG)
- Speak text (TTS)

When these work well, actual conversations are possible

- Still can be too machine-like to be natural
S: Welcome to MyBus. Which itinerary do you want information for?
U: I need to go to the airport.
S: Where are you leaving from?
U: Downtown.
S: Just a minute. Let me check that for you.
S: There is a 28X leaving downtown at 7:03 PM. It will arrive at the airport at 7:37 PM.
S: You can say when is the next bus, when is the previous bus, start a new query, or goodbye.
U: When is the previous bus?
S: Just a second. There is a 28X leaving downtown at 4:20 PM. It will arrive at the airport at 4:56 PM.
S: You can say when is the next bus, when is the previous bus, start a new query, or goodbye.
U: Goodbye.
S: Thank you for using MyBus. Goodbye.
Of course, all those steps can be error-prone
  • In fact, rarely do all work 100% correctly

What happens when errors occur?
  • Degraded machine conversation

Errors can chain, compounding problems
  • Misrecognition → Bad parse → Wrong decision
S: Welcome to MyBus.
S: Which itinerary do you want information for?
U: uh… I wanna go from Pittsburgh Airport to the Cathedral of Learning.
S: Which itinerary do you want information for?
U: I said I wanna go from Pittsburgh Airport to the Cathedral of Learning.
S: Which itinerary do you want information for?
U: What?
S: Which itinerary do you want information for?
U: Well… I wanna go from Pittsburgh Airport to the…
S: Which itinerary do you want information for?
S: Just a minute. Let me check that for you.
U: What?
S: Which itinerary do you want…
S: Which itinerary do you want information for?
Overview

- **What to do with imperfect ASR results?**
  - Can anything useful be extracted or is there no meaning at all in what was recognized?

- **How can a conversation that’s going badly recover and end up successful?**

- **What approaches can be used for dialog decision making?**
Handling Imperfect Recognition

- Even with good recognition, you cannot blindly trust ASR output...

- Confidence annotation: Helios
  - Given the current state of the dialog, how confident is the system that the input matches the user’s intention?
  - Logistic regression based on speech, parsing, dialog features
  - Training from corpus of transcribed data
Grounding Concept Values

- **Grounding**: process where conversation participants establish common understanding

- For each understood concept, choose among 3 possible actions
  - Explicit confirmation: ask user a direct question, wait for a positive response before accepting
    - “To the airport. Is this correct?”
  - Implicit confirmation: repeat what was understood, accept unless user indicates it was wrong
    - “To the airport. Where are you leaving from?”
  - No action: silently accept without informing user

- Best choice can be situationally dependent
Enabling Confirmation in Olympus

Special Rosetta template prompts

```perl
$Rosetta::Templates::act{"implicit_confirm"} = {
   "origin_place" => "Leaving from <origin_place>.",
   ...
}
```
**Enabling Confirmation in Olympus**

**Confirmation policies config file:**
(Cconfigurations/DesktopSAPI/expl_impl.pol)

```
EXPLORATION_MODE=epsilon-greedy
EXPLORATION_PARAMETER=0.1
```

<table>
<thead>
<tr>
<th>State</th>
<th>ACCEPT</th>
<th>EXPL_CONF</th>
<th>IMPL_CONF</th>
</tr>
</thead>
<tbody>
<tr>
<td>INACTIVE</td>
<td>1</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CONFIDENT</td>
<td>-</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>UNCONFIDENT</td>
<td>-</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>GROUNDED</td>
<td>1</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

- Write Prompts
- Create Policies
- Attach Policies to Concepts
When defining concepts in the dialog manager, indicate which policy to apply:

```csharp
DEFINE_AGENCY( CPerformTask
    DEFINE_CONCEPTS(
        INT_USER_CONCEPT(query_type, "impl")
        STRING_USER_CONCEPT(origin_place, "expl_impl")
    ...
    )
    ...
    )
```
Handling Non-Understandings

- No meaning can be extracted from user input
  S: Where do you want to go?  
  U: (no parse)  
  S: ???

- Many possible system responses:
  S: Where do you want to go?  
  S: Could you repeat that?  
  S: For example, you can say, “Downtown”.  
  S: Which route are you looking for?
  ...

...
Non-Understanding Policies

- **Repeat question**
  - May work if temporary channel issue caused ASR problems
  - Frustrating to user if continued

- **Provide example of what to say**
  - Can assist unfamiliar users
  - Annoys users who already said the example and weren’t understood

- **Change topic**
  - Gets user to talk about something else
  - Still have to get original question answered
Non-Understanding Policies

Two types of policies:

- **Handcrafted/deterministic**
  - Design a (small) space of dialog states
  - Set a utility to each action in each state

- **Data-driven**
  - Learn optimal weight based on collected dialogs
  - Exploration/exploitation trade-off at runtime
Create alternative versions of request template prompts:

"origin_place" => "Where are you leaving from?"
Create alternative versions of request template prompts:

"origin_place" => {
  "default" => "Where are you leaving from?",
  "explain_more" => "Right now, I need to know the stop or landmark from which you will be leaving.",
  "what_can_i_say" => "For example, you can say THE AIRPORT, or DOWNTOWN."},
Using Non-Understanding Strategies

Non-understanding policies config file:
(Configurations/DesktopSAPI/repeat.pol)

- EXPLORATION_MODE=greedy

- STATE[FIRST_NONU]
  - num_prev_nonu = 1
- END

- STATE[SUBSEQUENT_NONU]
  - num_prev_nonu > 1
- END

Write Prompts
Create Policies
Attach Policies to Agents
Using Non-Understanding Strategies

Non-understanding policies config file:
(Configuration/DesktopSAPI/repeat.pol)

<table>
<thead>
<tr>
<th></th>
<th>AREP</th>
<th>TYCS</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIRST_NONU</td>
<td>1</td>
<td>-</td>
</tr>
<tr>
<td>SUBSEQUENT_NONU</td>
<td>-</td>
<td>1</td>
</tr>
</tbody>
</table>
When defining subagents in the dialog manager, indicate which policy to apply:

```c
DEFINE_SUBAGENTS(
    SUBAGENT(RequestQuery, CRequestQuery, "tycs")
    SUBAGENT(RequestOriginPlace, CRequestOriginPlace, "arep_tycs")
...)
```
Dialog System Evaluation
What are the metrics?

- Task success
- Minimum call escalation
- Minimum call time
- Error recovery vs WER

What should I do to make it better
Spoken Dialog Challenge Goals

- Improve Spoken Dialog Systems
- Community wide effort
  - Accessible to new groups
- Find common task
  - Hard enough to be interesting
  - Easy enough that groups can build systems
- Test systems on common user base
  - Real users (not paid users)
- Provide framework for better evaluation
  - Not just task completion
Find bus schedules in Pittsburgh
  • CMU’s Let’s Go SDS
  • (120K live calls since 2005)

Answers calls via Port Authority
  • Weekdays 7pm-6am
  • Weekends 4:30pm-7am

Resources
  • Full source code for system
  • Full dialog logs
Build a complete system
  - From provided system
  - Or from your own system

Test an evaluation technique

Build a user simulator
SDC Timeline

- **Sep 2009** Announced
- **Nov 2009** Web seminars
- **Jan 2010** Data, system distributed
- **Jan-Apr 2010** Development
- **May 2010** Control Tests
- **Jul-Aug 2010** Live Tests
Test Scenarios

- **Control Tests**
  - 4 systems (sys1-4)
  - Around 100 calls
  - Provided scenarios
  - SDS Experts as callers
  - 4 systems (sys1, sys2, sys3, sys4)

- **Live Tests**
  - 14 days each (evenings/weekend)
  - Real callers who want bus information
  - 3 systems (sys1, sys3, sys4)
You want:

- Time you want to leave: 8:50 pm
- Any bus you can take before 9:45

You want:
Control Tests

- **Web based instructions**
  - To aligns calls and scenarios
- **8 calls per user**
- **Required international calls**
  - Remember time zones!
- **User pool is SDS researchers**
  - Unfamiliar with Pittsburgh buses
  - More non-native speech
  - Don’t care about next bus
  - Better/more forgiving to SDS
- **Finding callers is hard**
Measuring Success

- **Non-empty calls**
  - User might just hang up

- **No information**
  - Call ends before any information given

- **“I don’t have that”**
  - No bus for those arrival/departure point
  - No bus at that time
  - Result can be **correct** or **incorrect**

- **Positive Output**
  - There is a bus from X to Y at time Z
  - Result can be **correct** or **incorrect**
## Control Test

<table>
<thead>
<tr>
<th></th>
<th>SYS1</th>
<th>SYS2</th>
<th>SYS3</th>
<th>SYS4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Calls</td>
<td>91</td>
<td>61</td>
<td>75</td>
<td>83</td>
</tr>
<tr>
<td>no_info</td>
<td>3.3%</td>
<td>37.7%</td>
<td>1.3%</td>
<td>9.6%</td>
</tr>
<tr>
<td>donthave</td>
<td>17.6%</td>
<td>24.6%</td>
<td>14.7%</td>
<td>9.6%</td>
</tr>
<tr>
<td>donthave_corr</td>
<td>68.8%</td>
<td>33.3%</td>
<td>100.0%</td>
<td>100.0%</td>
</tr>
<tr>
<td>donthave_incorr</td>
<td>31.3%</td>
<td>66.7%</td>
<td>0.0%</td>
<td>0.0%</td>
</tr>
<tr>
<td>pos_out</td>
<td>79.1%</td>
<td>37.7%</td>
<td>84.0%</td>
<td>80.7%</td>
</tr>
<tr>
<td>pos_out_corr</td>
<td>66.7%</td>
<td>78.3%</td>
<td>88.9%</td>
<td>80.6%</td>
</tr>
<tr>
<td>pos_out_incorr</td>
<td>33.3%</td>
<td>21.7%</td>
<td>11.1%</td>
<td>19.4%</td>
</tr>
<tr>
<td></td>
<td>SYS1</td>
<td>SYS3</td>
<td>SYS4</td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>------</td>
<td>------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>Total Calls</td>
<td>678</td>
<td>451</td>
<td>742</td>
<td></td>
</tr>
<tr>
<td>Non-empty calls</td>
<td>633</td>
<td>430</td>
<td>670</td>
<td></td>
</tr>
<tr>
<td>no_info</td>
<td>18.5%</td>
<td>14.0%</td>
<td>11.0%</td>
<td></td>
</tr>
<tr>
<td>donthave</td>
<td>26.4%</td>
<td>30.0%</td>
<td>17.6%</td>
<td></td>
</tr>
<tr>
<td>donthave_corr</td>
<td>47.3%</td>
<td>40.3%</td>
<td>37.3%</td>
<td></td>
</tr>
<tr>
<td>donthave_incorr</td>
<td>52.7%</td>
<td>59.7%</td>
<td>62.7%</td>
<td></td>
</tr>
<tr>
<td>pos_out</td>
<td>55.1%</td>
<td>56.0%</td>
<td>71.3%</td>
<td></td>
</tr>
<tr>
<td>pos_out_corr</td>
<td>86.8%</td>
<td>93.8%</td>
<td>91.6%</td>
<td></td>
</tr>
<tr>
<td>pos_out_incorr</td>
<td>13.2%</td>
<td>6.2%</td>
<td>8.4%</td>
<td></td>
</tr>
</tbody>
</table>
Control vs Live Tests

### Control

<table>
<thead>
<tr>
<th>Category</th>
<th>SYS1</th>
<th>SYS2</th>
<th>SYS3</th>
<th>SYS4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Calls</td>
<td>91</td>
<td>61</td>
<td>75</td>
<td>83</td>
</tr>
<tr>
<td>no_info</td>
<td>3.3%</td>
<td>37.7%</td>
<td>1.3%</td>
<td>9.6%</td>
</tr>
<tr>
<td>donthave</td>
<td>17.6%</td>
<td>24.6%</td>
<td>14.7%</td>
<td>9.6%</td>
</tr>
<tr>
<td>donthave_corr</td>
<td>68.8%</td>
<td>33.3%</td>
<td>100.0%</td>
<td>100.0%</td>
</tr>
<tr>
<td>donthave_incorr</td>
<td>31.3%</td>
<td>66.7%</td>
<td>0.0%</td>
<td>0.0%</td>
</tr>
<tr>
<td>pos_out</td>
<td>79.1%</td>
<td>37.7%</td>
<td>84.0%</td>
<td>80.7%</td>
</tr>
<tr>
<td>pos_out_corr</td>
<td>66.7%</td>
<td>78.3%</td>
<td>88.9%</td>
<td>80.6%</td>
</tr>
<tr>
<td>pos_out_incorr</td>
<td>33.3%</td>
<td>21.7%</td>
<td>11.1%</td>
<td>19.4%</td>
</tr>
</tbody>
</table>

### Live

<table>
<thead>
<tr>
<th>Category</th>
<th>SYS1</th>
<th>SYS3</th>
<th>SYS4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Calls</td>
<td>678</td>
<td>451</td>
<td>742</td>
</tr>
<tr>
<td>Non-empty calls</td>
<td>633</td>
<td>430</td>
<td>670</td>
</tr>
<tr>
<td>no_info</td>
<td>18.5%</td>
<td>14.0%</td>
<td>11.0%</td>
</tr>
<tr>
<td>donthave</td>
<td>26.4%</td>
<td>30.0%</td>
<td>17.6%</td>
</tr>
<tr>
<td>donthave_corr</td>
<td>47.3%</td>
<td>40.3%</td>
<td>37.3%</td>
</tr>
<tr>
<td>donthave_incorr</td>
<td>52.7%</td>
<td>59.7%</td>
<td>62.7%</td>
</tr>
<tr>
<td>pos_out</td>
<td>55.1%</td>
<td>56.0%</td>
<td>71.3%</td>
</tr>
<tr>
<td>pos_out_corr</td>
<td>86.8%</td>
<td>93.8%</td>
<td>91.6%</td>
</tr>
<tr>
<td>pos_out_incorr</td>
<td>13.2%</td>
<td>6.2%</td>
<td>8.4%</td>
</tr>
</tbody>
</table>
## Call Lengths

<table>
<thead>
<tr>
<th></th>
<th>Length (s)</th>
<th>Turns/call</th>
<th>Words/turn</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYS1 control</td>
<td>155</td>
<td>18.29</td>
<td>2.87 (2.84)</td>
</tr>
<tr>
<td>SYS1 live</td>
<td>111</td>
<td>16.24</td>
<td>2.15 (1.03)</td>
</tr>
<tr>
<td>SYS2 control</td>
<td>147</td>
<td>17.57</td>
<td>1.63 (1.62)</td>
</tr>
<tr>
<td>SYS3 control</td>
<td>96</td>
<td>10.28</td>
<td>2.73 (1.94)</td>
</tr>
<tr>
<td>SYS3 live</td>
<td>80</td>
<td>9.56</td>
<td>2.22 (1.14)</td>
</tr>
<tr>
<td>SYS4 control</td>
<td>154</td>
<td>14.70</td>
<td>2.25 (1.78)</td>
</tr>
<tr>
<td>SYS4 live</td>
<td>126</td>
<td>11.00</td>
<td>1.63 (0.77)</td>
</tr>
</tbody>
</table>
## Completion Results

<table>
<thead>
<tr>
<th></th>
<th>SYS1</th>
<th>SYS3</th>
<th>SYS4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control</td>
<td>64.9% (5.0%)</td>
<td>89.4% (3.6%)</td>
<td>74.6% (4.8%)</td>
</tr>
<tr>
<td>Live</td>
<td>60.3% (1.9%)</td>
<td>64.6% (2.3%)</td>
<td>71.9% (1.7%)</td>
</tr>
</tbody>
</table>

- Live rates lower than Control rates
- Different system orders between Control and Live
## Completion vs WER

### Completion

<table>
<thead>
<tr>
<th></th>
<th>SYS1</th>
<th>SYS3</th>
<th>SYS4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control</td>
<td>64.9% (5.0%)</td>
<td>89.4% (3.6%)</td>
<td>74.6% (4.8%)</td>
</tr>
<tr>
<td>Live</td>
<td>60.3% (1.9%)</td>
<td>64.6% (2.3%)</td>
<td>71.9% (1.7%)</td>
</tr>
</tbody>
</table>

### Word Error Rate

<table>
<thead>
<tr>
<th></th>
<th>SYS1</th>
<th>SYS3</th>
<th>SYS4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control</td>
<td>38.4</td>
<td>27.9</td>
<td>27.5</td>
</tr>
<tr>
<td>Live</td>
<td>43.8</td>
<td>42.5</td>
<td>35.7</td>
</tr>
</tbody>
</table>
Observations

- **Control vs Live is different**
  - Addressing real users is different
  - Control callers speak longer, don’t give up

- **Word error rate matters**
  - But for real users not so much for control

- **Confirmation vs Call Length**
  - Most correct calls vs most correct calls per hour

- **WER Important**
  - But SYS3/SYS4 equal in Control different in Live
  - SYS4 had more controlled dialogs (longer)
  - SYS3 had much shorter dialogs
Further Analysis

- **Automatic Measures**
  - Can we get automatic measures

- **Too much to hand label in real systems**
  - Cannot label every day
  - Need daily automatic measure
  - WER, completion rate etc not accurate enough
  - Can we find something better

- **Full database release**
  - [http://dialrc.org/data/sdc2010](http://dialrc.org/data/sdc2010)
  - Control and Live Test Logs
  - Transcription of all turns
  - Hand labeled success rate
Dialog System Deployment
From the lab to real world

- Developing a real usable system
  - Strategy to deployment
  - Engineering issues
  - Stability issues

- Call analysis
  - Finding issues in calls
From the lab to the real world

- Build system that works for you
- Have locals (developers) use it
  - Analyze their usage and build a new system
- Have subset of target users use it
  - Analyze their usage and build a new system
- Have target users use it
  - Analyze their usage and build a new system
- Over time after successful deployment
  - Analyze usage and build a new system
Improving a system

- **Improving components**
  - Acoustic models (channel-appropriate data)
  - Language models (what people actually say)
  - Grammar models (how to interpret it)
  - Dialog strategies (how people get what they want)
  - New functionality (asking for things you could give)
Improving Acoustic models

- **Collect date of target audience using system**
  - It’s from the right channel
  - It’s the right distribution of vocabulary
  - It’s the right distribution of dialects
  - It’s the right distribution of style

- **Collect over time**
  - You may have different users and different times of the week.
Collecting real data:
- What words and phrases are used
  - (Are people polite, rude, include greetings …)
- How do people actually ask for things you offer

Grammar additions
- Modify your grammar to deal with what real users actually say
**Dialog Strategies and New Functions**

- **What are people doing**
  - Analyze dialog states and look for novel events
    - (e.g. adding “next” and “previous” bus)
- **What are they saying when you tell them you can’t help.**
  - What functionality are they asking for
How well is my system working

- Number of callers
- Measuring success
  - (Real measuring of success)
- Number of Turns
- Number of User Complaints
- User satisfaction
- Proportion of first time callers
- Call through to operators
User satisfaction

- **Caring beyond “success”**
  - Good success, painful success

- **How to detect it**
  - **Word level information**
    - Politeness (or lack off)
  - **Prosodic information**
    - Detect frustration, anger etc
  - **Human labeling**
    - Subsample the data and estimate it
What is happening in your 100K calls?

Estimate success

Count number of turns

Count number of known errors

Trace different dialog state sequences
  • Are some more likely to fail.

What should you do next to improve task success.
Call Analysis

- **Cannot listen to them all**
  - Need optimal way to sub-sample
- **Find the “interesting” calls**
- **Who are your users**
  - First time callers, repeat callers
  - What classes succeed
  - What classes fail
USAir, Amtrak, AT&T ....

They wish to minimize operator calls

- % of calls dealt with automatically
- (successfully or not ?)
Dialog Users

- They want the service to work
- They want it to work in an obvious way
- It must be better than waiting for a human
- It must be able to deal with their task
- It must get better for them
System Designer

- **Must be easy to maintain**
  - Not require re-design every day

- **Must be fast**
  - The more calls you serve the money you make
  - The faster calls are the less equipment you need

- **Must adapt to the needs**
  - Customer and users
Must be reliable hardware/software

Can it deal with (near) simultaneous calls

Can it deal with very long calls

POTS is not a very stable system
  • Hard to detect hang up

It has to run 24/7

You must detect hangs automatically
  • Not easy on some operating systems
Cost per call
- What is the average length of a call
- How many simultaneous calls per machine

Can it scale to 10, 100, 1,000 …

Can you deal with call volume
- What are the peaks
- What are the down times

Can your (Amazon) Cloud deal with that?