Natural Language Processing for Spoken Dialog Systems

Alex Sekula

Second Annual MIT PRIMES Conference, May 20, 2012
High Level Introduction

• Assistive technology
  ◦ Helps disabled people

• The Boston Home
  ◦ Nursing care facility
  ◦ Robotic Wheelchair
    • Works on its spoken dialog system

• Spoken dialog system makes things easier for the residents
  ◦ Can still be improved
Wheelchair Image
Introduction

• Spoken dialog system has two parts
  ○ Speech Recognition
  ○ Natural Language Processing (NLP)

• Goal of the project: Improve both parts of the dialog system using natural language processing
Dialog System

- Has certain actions that it can make (categories)
- Shows the weather, meals of the day, activities for the day, etc.
- Categories are made of two parts (sub-categories), some categories share a part
List of Dialog System Categories

- system wake_up
- system go_to_sleep
- time
- time
- activities today
- activities tomorrow
- activities monday
- activities tuesday
- activities wednesday
- activities thursday
- activities friday
- activities saturday
- activities sunday
- weather today
- weather tomorrow
- weather monday
- weather tuesday
- weather wednesday
- weather thursday
- weather friday
- weather saturday
- weather sunday
- weather three_day
- breakfast today
- breakfast tomorrow
- breakfast monday
- breakfast tuesday
- breakfast wednesday
- breakfast thursday
- breakfast friday
- breakfast saturday
- breakfast sunday
- lunch today
- lunch tomorrow
- lunch monday
- lunch tuesday
- lunch wednesday
- lunch thursday
- lunch friday
- lunch saturday
- lunch sunday
- dinner today
- dinner tomorrow
- dinner monday
- dinner tuesday
- dinner wednesday
- dinner thursday
- dinner friday
- dinner saturday
- dinner sunday
- voice_synthesizer audio_on
- voice_synthesizer audio_off
- voice_synthesizer interrupt
- voice_synthesizer speak_text_on_screen
- phone make_phone_call
- phone hang_up
- phone hold_call
- phone resume_call
- phone fullcreen_video
- phone unfullscreen_video
- phone show_contacts
- phone answer_phone
- confirmatory yes
- confirmatory no
- null yes_record
### Dialog System GUI

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mon, Apr 16, 2012</td>
<td>10:52 AM</td>
<td>Awake</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Voice On</td>
</tr>
<tr>
<td>Phone Available</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

You asked about today's activities.

**Today's activity schedule:**

- 09:30am: Coffee and News
- 01:00pm: 2nd Floor Store
- 01:45pm Movie
- 03:00pm: Bridge.
Part 1: Applying NLP to Speech Recognition

• Lower the error rate of the speech recognition with NLP
• Obtain NLP data from sentences
• Use data to classify it as "correctly transcribed" or "incorrectly transcribed"
Training Set

• Training set was needed to train the classifier

• The Boston Home residents created this set
  ○ Asked for a category
  ○ Marked off whether the transcription was correct
Extracting NLP Features

• Classifier was trained with NLP features
• Used the Stanford Parser to extract the NLP (part-of-speech) features
• Example of some features:
  ○ NN - Noun, singular
  ○ ADJP - Adjective phrase
  ○ PRP - Personal Pronoun
Stanford Parser Example

My/PRP$  dog/NN  also/RB  likes/VBZ  eating/VBG  sausage/NN  ./.

- PRP$ - Possessive personal pronoun
- NN - Noun singular
- RB - Adverb
- VBZ - Verb, 3rd person singular present
- VBG - Verb, present participle
Using the features with AdaBoost

• AdaBoost
  ○ An Adaptive Boosting algorithm
• Finds features that separate the correct and incorrect sentences the best
• Uses those features to train classifier
• Example:
  ○ FRAG (fragment) appeared more in incorrect sentences
  ○ Classifier would mark sentence down if it contained FRAG
Results of the Final Classifier

• Test set
  ◦ Original training set was split 80-20
  ◦ 80: Training set
  ◦ 20: Test set

• 67.7% (+/- 2%) correct on the test set
Improving NLP in the Dialog System

• Old NLP was simple
  ○ Used keyword searching
  ○ Could not ask more natural questions
• Goal: To enable understanding of more natural phrases
Approach

• Needed to associate words like "forecast" or "rain" with "weather"

• Scanned online sources that contain relevant words in the same place
  ○ Wikipedia
  ○ Twitter
  ○ Amazon Mechanical Turk
Scanning Online Sources

• Scanned online sources for each sub-category
  ○ Weather, Dinner, Lunch, etc.

• Put all of these words into files

• Used different online sources to find relevant words
  ○ Wikipedia Articles
  ○ Twitter Tweets
  ○ Amazon Mechanical Turk Data
Online Sources

• Wikipedia
  ○ Sub-category was most often title

• Twitter
  ○ Sub-category was a word in the tweet

• Amazon Mechanical Turk
  ○ Users were asked to type in ways that they would ask for a category
  ○ Restriction: No sub-categories in the sentence
  ○ Example
    ▪ Prompt: "Ask for 'weather today' without using the words 'weather' or 'today'"
    ▪ Potential response: "What is the forecast right now?"
Using Tf*Idf for Word Association

• To find a word's most relevant sub-category, tf*idf was used.

• Term frequency-Inverse Document Frequency
  ○ Used to score how relevant a sub-category is to a word.
  ○ Finds the best association of that word.
**Tf*Idf**

- For a word to a document in a set of documents

- **Term frequency**
  - Input word count divided by total word count

- **Inverse document frequency**
  - Takes the log\_10 (total files / total files containing that word)

- **Tf*Idf**
  - Multiplies the tf and the idf
Tf*Idf Example

• Example: "The Brown Cow"
• "The"
  ○ tf - Generally high
  ○ Most likely all files contain this word
    ▪ idf = log_10(1) = 0
  ○ tf*idf = 0
• "Brown" or "Cow"
  ○ tf - Smaller
  ○ idf - Not zero
  ○ tf*idf - Not zero, both have impact on total tf*idf
Categorizing an Input Sentence

• For each sub-category, tf*idf was used on each word in the input sentence
• Total tf*idf = sum of word tf*idfs
• Both sub-category scores were added to find score of the entire category
Incorporating Keyword Searching

- Phrases containing sub-categories should definitely be categorized correctly
- Used keyword searching of each sub-category
- Prioritized the keyword searching

- e.g "What is the forecast for monday?"
  - All categories containing the sub-category "monday" were considered
Example Results

• Input: "what is the forecast for monday"
• Results:

```
lunch monday  1000.00118833
activities monday 1000.00138955
breakfast monday 1000.00149459
dinner monday  1000.001626
weather monday  1000.00299411
```

Best: weather monday

(results only show top 5 categories)
Another Example

• Input: "pizza" (using Twitter)
• Results (top 3):

```
breafast monday
0.000639681442312
```

```
dinner today
0.000640723296498
```

```
breafst today
0.000682054366294
```
Getting Results

• Two different test sets
  • My test set
    ○ Listed 5-10 sentences per category
    ○ Unbiased
    ○ No restrictions
  • Amazon Mechanical Turk test set
    ○ Used the training set as a test set (80-20 split)
    ○ Keyword searching would perform 0% ideally
## Results

(percent of correctly classified sentences in each test set)

<table>
<thead>
<tr>
<th></th>
<th>Keyword Searching</th>
<th>Wikipedia</th>
<th>Twitter</th>
<th>AMT Training Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>My Test Set</td>
<td>69.1</td>
<td>79.5</td>
<td>76.1</td>
<td>74.9</td>
</tr>
<tr>
<td>AMT Test Set</td>
<td>9.9</td>
<td>37.4</td>
<td>23.5</td>
<td>63.0</td>
</tr>
</tbody>
</table>
Special thanks to...

- The MIT PRIMES program
- Slava Gerovitch
- Pavel Etingof
- Tanya Kovanova
- William Li, my mentor