Lecture 11

General Virtual Assistants
Goal of this Lecture

• The course so far: Latest research in NLP
• Virtual assistants are an established commercial market
  • What is the virtual assistant ecosystem?
  • What is the user experience?
• What else does it take to build a virtual assistant?
  • Full pipeline: Voice → Semantic parsing → Execute
• Two architectures:
  • Alexa (duopoly, proprietary, privacy concerns)
  • Genie (research, open-source, privacy-protecting)
Discussion

• What are the most popular virtual assistants?
• What do people use it for?
• Which is good for IoTs?
• What do people like or not like about their assistants?
Some Statistics

- Smartspeakers: Tremendous early adoption
  - 50 millions in 2 years (Internet: 50 millions in 4 years)
- 123.5M adults in the US (out of 200M total) will use voice assistants at least once per month in 2022
- 42.7% adults will use a smartphone
- 32.2% adults will use a smart speaker
- 64M were monthly users of Amazon Echo last year (2x Google Home users)
Alexa

• 3 classes of functions
  • First class functions built by Alexa
  • Smart home devices (Alexa protocol)
  • 3rd party skills
Alexa 3rd Party: a Voice Web?

- Virtual assistant is the gateway to a voice web
- Alexa: 250,000 third-party skills
- From telling jokes, playing games, weather, answering questions, shopping, banking

![Graph showing the growth of Alexa Skills over years]
Alexa Skills vs WWW
In their 1\textsuperscript{st} 7 years

K sites

0 500 1000 1500 2000 2500 3000

years
1 2 3 4 5 6 7

Alexa Skills
WWW

https://www.internetlivestats.com/total-number-of-websites/
Google Home

- Google Assistant pre-installed on all Android phones
- Types of functions
  - 1\textsuperscript{st} class Google functions
  - Google Home protocol for smart homes
  - 3\textsuperscript{rd} party platforms: 1 million actions
    - Deprecated on June 13, 2023
      - announced on June 13, 2022
Quiz

• The 3rd platform is not growing as fast as the web
  • Why?

• What is the limitation to the growth of smart speakers?
Outline of This Lecture

- The virtual assistant ecosystem
- **What is the technology in a smart speaker?**
- Alexa’s platform
- How well does Alexa work?
- Genie: a research, private assistant
Comparing Alexa with the Web

1. Privacy
2. Poor User Experience: Discovery
3. Technology limit: Voice interfaces are much harder to create than webpages
   - Alexa provides the natural language capability
   - Alexa currently speaks 8 languages
   - Alexa has 10,000 developers
4. Monopolies vs. open system
   - Control over the platform; intermediate customer relationships
   - Self-preferencing Amazon’s products
   - Charges 30% of digital goods purchased on Alexa
SmartSpeakers

- A small cloud-connected device
  - All information is shipped to the cloud
    - Your credentials to all your accounts
    - Including the status of all your IoT devices
  - Responsible for handling voice inputs and outputs
    - Always on!
Sound Technology

- Acoustic Echo Cancellation
- Voice Activity Detection
- Wake Word Detection
- Ducking
- Speech-to-text
Sound Technology

- Acoustic echo cancellation (AEC)
- Problem: How to tell the device to stop while it is playing?
  - They play music, news, podcasts. (Words are even harder)
  - Echo: originally transmitted signal received after a delay
  - DSP (Digital signal processing) to remove the signal
  - Compute input (microphone) minus reference (agent output)
Sound Technology

- Voice Activity Detection: Is there speech?
  - Classify between noise and speech according to Gaussian Mixture Model
    - Compute similarity to speech spectrum
    - Compute similarity to noise spectrum
    - Choose most similar

Acoustic Echo Cancellation → Voice Activity Detection → Wake Word Detection → Ducking → Speech-to-text
Sound Technology

- Wake word detection
  - Always-on: Needs to run on local device: Privacy, efficiency
  - Accuracy is critical
    - Misfires are common and not always noticeable
  - Training with large speech-to-text corpora, sampled for the desired word (2 hours, Picovoice)
Sound Technology

- Acoustic Echo Cancellation
- Voice Activity Detection
- Wake Word Detection
- Ducking
- Speech-to-text

- Ducking:
  - Lower the master output volume temporarily while listening
  - Improve listening accuracy
  - Lower the background volume temporarily while speaking
  - Agent speech more understandable
Sound Technology

- **Speech-to-Text:**
  - Typically done in the cloud
  - Inference on a large neural network trained with a massive amount of data
  - Accuracy varies across languages and accents
Outline of This Lecture

• The virtual assistant ecosystem
• What is the technology in a smartspeaker?
• Alexa’s 3rd platform
• How well does Alexa work?
• Genie: a research, private assistant
Alexa 3rd Party Design Goals
A second-class citizen

• **Users must enable skills to avoid interference**
  • 3rd party commands do not degrade 1st party commands
  • Multiple companies can provide same commands

• **Shallow semantic parsing: intents/slots**
  • Fast iteration
  • Simple connection to APIs
    (function invocation with parameters)
Amazon 3\textsuperscript{rd} Party Platform

<table>
<thead>
<tr>
<th>Skill</th>
<th>Skill + intent</th>
<th>Skill + intent + slots</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open Capitol1; Launch Palo Alto Health;</td>
<td>Ask U.S. Bank to get my checking account balance</td>
<td>Ask Ring Central to call 16501234567</td>
</tr>
</tbody>
</table>

Name of the company Identified by a fixed pattern ("open/launch/ask/tell X")

A choice of intents (API calls) Use a classifier Predict the most likely choice

Slots: typed parameters to APIs Tag the words according to a set of (predefined) types
Ask Yahoo what is the stock quote of AAPL?

When the market closed, Apple traded at $143.43 on the NASDAQ, up 0.34% since previous close.
How to Build a Skill?

1. User Interface:
   - Declare a list of intents and their associated slots
     - SearchRestaurant(loc, cuisine)
     - GetRestaurant(name)
   - Provide phrases for each intent (as many as possible)
     - “find me a good restaurant”
     - “find a restaurant in {loc}”
   - Choose type for each slot
     - loc : Location
   - If slot is not predefined, provide phrases for new slot types
     - name : “Mc Donalds”, “Burger King”, ..
     - cuisine: “American”, “Italian”..
How to Build a Skill?

2. Fulfillment (backend)
   • Implement HTTP server to be called by Alexa backend
   • Extract intent and slots from HTTP request
   • Map slots from natural language ("tomorrow") to API value ("2021-09-23T00:00:00Z")
   • Call any necessary APIs
   • Construct and return reply text to be spoken to the user
Outline of This Lecture

• The virtual assistant ecosystem
• What is the technology in a smartspeaker?
• Alexa’s 3rd platform
• How well does Alexa work?
• Genie: a research, private assistant
How Well Does Alexa Work?

• GDPR: Companies allow users to download their data

• Real world Alexa data
  (from mturk and word-of-mouth participants)

• Between 100 and 1000 transcripts for each user

• Preliminary study: 12 users
Expt With Privacy

Data uploading page

Can delete or flag each command
flag: only available to researchers
What Commands are People Issuing?

Top Commands (80% of total)
1. PlayMusic
2. TurnOffAppliance
3. TurnOnAppliance
4. Stop
5. SetNotification (timer)
6. QA
7. BrowseNotification
8. GetWeatherForecast
9. WakeWord
10. WhatTime
11. SearchItem
12. LaunchNativeApp
13. MusicControl
14. PlayStation
15. CancelNotification
16. SetVolume
17. Unknown
18. SetValue

- 6761 commands in total
- 235 unique commands
Error Rate Depends on Intent Types

Besides QA (question answering), PlayMusic, lesser used intents are more prone to errors (6.7% vs 26.7% on simple random sample at N=30).

Quiz: Why are QA and PlayMusic hard?
Errors Can (And Do) Happen Anywhere
Error Taxonomy Based on Prior Work and Virtual Assistant Systems
Each type of error has a different implication

Trigger: Wake word detection
ASR: Automatic Speech Recognition
OOD: Out-of-domain
Error Distribution

What is the most annoying?  
What is the worst category of errors?
Error Distribution

Surprise! Erroneous triggers are all saved in the user history!
Error Distribution

Hidden errors
- Trigger error – no response
  Why is it still in the user history?
- ASR error-correct response

Visible errors
- Intent/slot detection: semantic parsing
- Trigger error
  - incorrect response
The Full Spectrum of the Problem

Response generation and text-to-speech are less prone to errors

However, on the input side:

Trigger
Automatic speech recognition (ASR)
Out-of-domain detection
Even if we get the text right and it is in-domain:
Semantic parsing (or intent/slots)
Named entity detection
Some inputs are truly ambiguous

Error handling in dialogue agents has been studied for decades,
still no practical fixes
Summary of Experiment

1. Intents have a long tail
2. Different intents have very different error rates
3. Lesser used intents are more prone to errors
   (6.7% vs 26.7% on simple random sample at N=30).
4. A spectrum of errors: trigger, ASR, parsing
Discussion

• How should Alexa be improved?

• What does it take to get voice widely adopted?
Outline of This Lecture

• The virtual assistant ecosystem
• What is the technology in a smart speaker?
• Alexa’s 3rd platform
• How well does Alexa work?
• Genie: a research, private assistant
Genie Virtual Assistant

Demonstrates different design choices

- Open-source and public repository of skills: Thingpedia
- Conversational agents trained with synthesized data
  - long-tail coverage
  - less human labor intensive
- Privacy: Genie can run on local devices
  - Credentials. IoT are all local
- Programmable in natural language
  - Basics in this lecture
  - More advanced functions in the next
Alexa 3rd Party: A Dispatch Model

Results are simply to be spoken

Ask Yahoo what is the stock quote of AAPL?

Intent Classification

Fulfillment

Speech-to-text

Text-to-speech

When the market closed, Apple traded at $143.43 on the NASDAQ, up 0.34% since previous close.
Genie Tasks: A Function Execution Model

What is the stock quote of AAPL?

- Speech-to-text
  - Semantic Parsing
    - ThingTalk Execution
      - Current: $143.43
        - Last close: $142.94
        - Exchange: Nasdaq
- Text-to-speech
  - Language Generation
    - Policy
      - Apple is trading at $143.43 on the NASDAQ, up 0.34% since previous close.
# Execution vs. Dispatch Models

<table>
<thead>
<tr>
<th>Genie: Execution Model</th>
<th>Alexa: Dispatch Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Used by Genie for 1&lt;sup&gt;st&lt;/sup&gt; and 3&lt;sup&gt;rd&lt;/sup&gt; party skills</td>
<td>Used by Alexa &amp; Google Assistant on 3&lt;sup&gt;rd&lt;/sup&gt; party skills</td>
</tr>
<tr>
<td>Declaration of full signature of functions: typed inputs and typed return results</td>
<td>Declaration of typed inputs only</td>
</tr>
<tr>
<td>Skill returns typed results in ThingTalk</td>
<td>Skill returns display text</td>
</tr>
<tr>
<td>Supports composition: Return results can be used as inputs</td>
<td>Does not support composition</td>
</tr>
<tr>
<td>Response generator can adjust output according to the context</td>
<td>Canned output per intent</td>
</tr>
</tbody>
</table>
Genie Function Model

- Two kinds of grounding primitives
  - Queries: Return a query result (DB or API); No side effect
  - Actions: Perform an action, possibly returning a result. With potential side effect

<table>
<thead>
<tr>
<th>Twitter</th>
<th>Natural Language</th>
<th>API Signatures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Queries</td>
<td>get tweets from Stanford</td>
<td>home_timeline(), … author==“Stanford”</td>
</tr>
<tr>
<td>Actions</td>
<td>tweet “Stanford won!”</td>
<td>post (status)</td>
</tr>
</tbody>
</table>
Skills

- Open-source skills stored in publicly available Thingpedia
- **Manifest: Definition of a skill**
  1. **User interface:** Mapping of words to queries and actions
  2. **Fulfillment:** Implementing queries and actions
    - **Alexa:** Vendors set up http servers according to Alexa’s protocol
    - **Genie:** Works with protocols of vendors
const Tp = require('thingpedia');

module.exports = class DB extends Tp.BaseDevice {

  async query(query, env) {
    const sql = compile(query);
    return executeSQL(sql);
  }

  query: ThingTalk query as abstract syntax tree
  env: ThingTalk runtime (command ID, session ID)

  Compile: ThingTalk to DB query (e.g. sql, sparql)

  Execute: Execute the query and return the results
Fulfillment: Standard Protocols

- **Import** `loader` brings in library `loader` (a fulfillment library)
- **RSS**: Really Simple Syndication
  - Standardized content distribution method for news, blogs

```java
class @com.washingtonpost {
    import loader from @org.thingpedia.rss();
    ...
}
```

This is an RSS feed
Needs only RSS URL
const Tp = require('thingpedia');
module.exports = class Weather extends Tp.BaseDevice {
    async get_forecast({ location, date }, env) {
        // make HTTP request
        return [{
            status: ..., 
            temperature: ..., 
            wind_speed: ...
        }];
    }
};
Accounts

- Many skills need accounts: Spotify, email, IoT, FB
- Common authentication scheme: OAuth
  - IETF (Internet Engineering Task Force)
    RFC (Request for comments) 6749
  - Lets a third-party application obtain limited access to an HTTP service
  - User is redirected to login page on 3rd party website
  - Then redirected back to Genie with auth material
OAuth in Genie

class @com.spotify {
    import config from @org.thingpedia.oauth2(
        authorize="https://accounts.spotify.com/authorize"^^tt:url,
        get_access_token="https://accounts.spotify.com/api/token"^^tt:url,
        get_profile="https://api.spotify.com/v1/me"^^tt:url,
        profile=["id", "display_name", "product"]
    );
    ...
}

^^: RDF syntax of assigning a type to a string

import config: how the user configures the skill
Find my favorite songs by Taylor Swift

@com.spotify.get_item_from_library(), contains(artists, "spotify:artist:06HL4z0CvFAxyc27GXpf02" ^^com.spotify:artist("Taylor Swift"))
## Differences between Genie & Alexa

<table>
<thead>
<tr>
<th>Genie</th>
<th>Alexa</th>
</tr>
</thead>
<tbody>
<tr>
<td>Research assistant (&lt; 10 people)</td>
<td>Very successful product (10K employees)</td>
</tr>
<tr>
<td>Connect to IoT devices with existing protocols</td>
<td>Proprietary Amazon IoTs</td>
</tr>
<tr>
<td>Home Assistant has 1000 devices</td>
<td>Incompatible with Google, Apple, …</td>
</tr>
<tr>
<td>Connect to existing network resources</td>
<td>Companies put up Alexa servers</td>
</tr>
<tr>
<td>Privacy: Open-source software</td>
<td>Proprietary, full control of platform, 30%</td>
</tr>
<tr>
<td>Can be run locally (parsing in cloud currently)</td>
<td>tax on online digital goods</td>
</tr>
<tr>
<td>Conversational semantic parsing</td>
<td>Semantic parsing for single commands</td>
</tr>
<tr>
<td>Advanced features: programmable, extensible, federated sharing (next lecture)</td>
<td>intent classification for 3rd party</td>
</tr>
<tr>
<td></td>
<td>?</td>
</tr>
</tbody>
</table>
Summary

• How it works?
  • Sound processing → Semantic Parsing → fulfillment
• Voice is in its nascency, and it is already big!
• Glimpse of the future: in Alexa’s 3rd party platform
• Will we have a closed voice web
  or an open World Wide Voice?