A day in the life with speech recognition, machine learning, & IOT

David Boloker
CTO
Emerging Technologies, IBM
boloker@us.ibm.com

Ben Sechrist
Emerging Technologies, IBM
@BenDSechrist

Mark VanderWiele
Distinguished Engineer
Emerging Technologies, IBM
@MarkVanderWiele

#IBMBluemix
The user experience is changing

Search and Apps

Voice as the New Command Line

Ask and Receive

Analyze and Notify

ChatBOTs

Teach and Learn

Personal Assistants

Mean time to know or act

Conversation
Looking forward

We will no longer have to learn to use the machine, the machine will learn from listening to us.

We will converse naturally within our own digital world to:

• Ask questions
• Control devices
• Collaborate more naturally
• Purchase goods and services using “Conversational Commerce”
• Carry out our daily tasks - a personal assistant
• Learn, adapt, and extend our digital world

Re-imaging Enterprises - IT & LoB - where a majority of B2C interactions with new user experiences are initiated thru voice and text

• Conversational apps will eclipse the totality of application created to-date
Speech Scenarios:

Command and Control:
- “Make my drone fly” “turn on my oven”

Question and Answer:
- “What can you do”

Ask and Receive a Variable Response:
- “send” Response: “done” “sure” “got it”

Conversations:
- “I would like to remodel my kitchen”

Conversation with Reasoning:
- “Diagnose my problem”
New Job Roles - Interaction Designer, Conversation and Dialog, The Next Phase Of Designing Chatbot Personalities, Conversation Techniques For Designers...
Conversational Application - architectural flow

Device Characteristics?

- Always on listening for key/wakeup words, immediate voice activation, and voice as a command line.
  - some push to talk, others wakeup and command, others go into conversational mode
- Respond in < 2 seconds
- Balance between on device operations vs cloud
  - based on device capabilities
- mic config varies - Single directional mic to mems array, noise reduction, beam forming, voice detection, biometrics…

"The last “next” mile of device interface and analytics"
Aways listening (for wakeup words) low cost open device

https://developer.ibm.com/recipes/tutorials/connect-a-simplelink-wifi-cc3200-launchpad-to-iotf-


cc3200 + audio booster pack ~ $60

Audio booster pack
- mic, line in, DSP, codec, $30

CC3200 launchpad
- Built in Wifi
- Simple Link Wifi Config via iPhone app
- on board Accelerometer
- 2 temp sensors (board+object)
- 2 buttons
- < $30
- Arduino like dev environment & TI CodeComposer
  - http://www.energia.nu
  - 3.3V VCC, GPIOs, 1.5v analog read

Optional HW for Wakeup words and Voice Biometrics
Additional work on far field beam forming & noise reduction
Demo 1: Voice Command/Control

Simple voice to text

- Match command to known list of actions (look for key words, nouns, verbs)
- Respond with text
- Generate response
- Generate MQTT command to control device
Demonstration 2: A day in the Life with Speech, ML, and IOT

Using speech recognition and cloud technologies to:

Demo:

- **Monitor** and help plan my day’s activities, including waking me up.
- **Summarize** my portfolio and News
  - Send to my phone or car.
- **Analyze** business data, IOT Data
- **Search and follow** hot topics
- **Plan and track** my projects, shop, get help
- **Display, Analyze & Visualize my devices** around the world
- **Securely control devices** around the world (with voice)
- **Create reusable conversations** from my interactions, creating a “verbal mashup”
Demo 3: Use Voice to command a device and analyze output

You can do this at home!

Live conversation with a device

- Control other devices in the room - BLE - ex BB8
- Take a picture
- Analyze the picture and tell me about it
- Analyze what has been happening at my front door

“Wakeup Watson” - SnowBoy
“Can you make my robot change color”
“What do you see on my front porch?”

“Taking a picture.”

...“I see what appears to be a person.”
Demo 4: Voice Command/Control

Build a conversation live

<table>
<thead>
<tr>
<th></th>
<th>qcon_examplew</th>
<th>Intents</th>
<th>Entities</th>
<th>Dialog</th>
<th>Improve</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Create new</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>#color_light</td>
<td>#greeting</td>
<td>Good afternoon</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>#color_light</td>
<td>Can you make my light blue?</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>#greeting</td>
<td>Good afternoon</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Create new: qcon_examplew

Intents: #color_light
Entities: Can you make my light blue?
Dialog: Good afternoon
Improve: None
Using Customization based on conversations

Speech to Text

Streamed voice wav

On device operations vs on cloud

EndPoint Devices

"Big Brother"-always listening

input voice wav

reply wav

Behind The Scenes

Demo - Services used, basic architecture

Custom Node App - Deployed in Bluemix - Orchestrate between services "Hal9000"

Domain Services

Standard Services

Health

MQTT commands/status

Text to Speech

text

JSON

Tone Analyzer

Discover, understand, and revise the language tones in text

Visual Recognition

AlchemyData News

Language Translator

Weather Company Data

Use the Weather Company Data for IBM Bluemix service to incorporate weather

Dave Boloker, Mark VanderWiele, & Ben Sechrist
IBM Emerging Technologies
Watson service pipeline

What did you say?

Speech to Text

Using Customization based on conversations

What do your really want?

What is your current **context**?
What was your **Intent**?
What **Object/Entity**?

Conversation

- Use response (Q/A)
- Randomize response
- Personalize
- Use context
- Generate response

Are you upset?

Tone Analyzer

- Add empathy
- Escalate
Understanding Speech to Text

Bolt on simple speech front end - direct command/control

Ouch, match text returned to cmd != strcmp
Speech code - evolution to custom model

1. Speech to Text API

2. Match text to CMD
   - Guess vs Final?
   - Word confidence score
   - Conversations & Context?
   - Phonetic matching?
   - Acronyms?
   - Nouns n verbs?
   - Utterances?

Retrain the speech to text service

I said “my drone is Bryan's bebop”
I got “my German is Brian’s Bieber”

“Can you ask my drone to fly please”
“Can you ask my drum the fly place”

To increase the probability within context of matching a command we use a custom speech model

You can use multiple custom models within a conversation
Creating a Custom Speech Model

1. Create custom model
   - POST /api/v1/customizations
     
     ```
     {
       "name": "Custom Model",
       "base_model_name": "en-US_BroadbandModel",
       "description": "Custom Model for QCON"
     }
     ```
   
   • Response 201
   ```
   {  
     "customization_id": "abcdefghijklmnopqrstuvwxyz"
   }
```

2. Add corpus
   - POST /api/v1/customizations/{id}/corpora/{name}
     
     • Corpus file
   
   • Response 201

3. Wait for corpus to be analyzed
   - GET /api/v1/customizations/{id}/corpora
     
     ```
     {  
       "corpora": [
         ...
         {  
           "name": {name},
           ...
           "status": "analyzed"
         }
       ...
     ]
     ```

4. Train model
   - POST /api/v1/customizations/{id}/train
     
     • Response 200

5. Wait for model to be trained
   - GET /api/v1/customizations/{id}
     
     ```
     {  
       "status": "ready"
     }
     ```

http://www.ibm.com/watson/developercloud/speech-to-text/api/v1/
Example Code

Create Custom Model

```javascript
const request = require('request);

const WATSON_API = 'https://stream.watsonplatform.net/speech-to-text/api/v1/';

request({
  baseUrl: WATSON_API,
  uri: '/customizations',
  method: 'POST',
  json: true,
  auth: {
    user: '',
    pass: ''
  }
}, (err, res, body) => {
  if (err || res.statusCode !== 201) {
    // Handle error
  } else {
    // Grab customization id from json body
  }
});
```
Example Code

Add Corpus to Model

```javascript
const request = require('request'),
    fs = require('fs');

const WATSON_API = 'https://stream.watsonplatform.net/speech-to-text/api/v1/';
const customization_id = 'abcd1234';
const corpus_name = 'corpus1';

fs.createReadStream('corpus1.txt').pipe(request({
    baseUrl: WATSON_API,
    uri: '/customizations/' + customization_id + '/corpora/' + corpus_name,
    method: 'POST',
    json: true,
    auth: {
        user: '',
        pass: ''
    }
}), (err, res, body) => {
    if (err || res.statusCode !== 201) {
        // Handle error
    }
});
```
Train Custom Model

```javascript
const request = require('request');

const WATSON_API = 'https://stream.watsonplatform.net/speech-to-text/api/v1/';
const customization_id = 'abcd1234';

request(
  {
    baseUrl: WATSON_API,
    uri: '/customizations/' + customization_id + '/train',
    method: 'POST',
    json: true,
    auth: {
      user: '',
      pass: ''
    }
  }, (err, res, body) => {
    if (err || res.statusCode !== 201) {
      // Handle error
    }
  });
```
Using Custom Model in Speech-to-Text

<table>
<thead>
<tr>
<th>Returned</th>
<th>Watson</th>
<th>Watson Custom Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>why did you get me out so early</td>
<td>80.30%</td>
<td>87.20%</td>
</tr>
<tr>
<td>Accuracy</td>
<td>61.29%</td>
<td>100.00%</td>
</tr>
<tr>
<td>Latency</td>
<td>3132 ms</td>
<td>3046 ms</td>
</tr>
<tr>
<td>why did you get me up so early</td>
<td>36.80%</td>
<td>66.50%</td>
</tr>
<tr>
<td>Accuracy</td>
<td>66.67%</td>
<td>100.00%</td>
</tr>
<tr>
<td>Latency</td>
<td>3069 ms</td>
<td>2044 ms</td>
</tr>
<tr>
<td>asking there's</td>
<td>41.00%</td>
<td>70.40%</td>
</tr>
<tr>
<td>Accuracy</td>
<td>28.57%</td>
<td>100.00%</td>
</tr>
<tr>
<td>Latency</td>
<td>2417 ms</td>
<td>2408 ms</td>
</tr>
<tr>
<td>other</td>
<td>4.00%</td>
<td>98.10%</td>
</tr>
<tr>
<td>Accuracy</td>
<td>0.00%</td>
<td>100.00%</td>
</tr>
<tr>
<td>Latency</td>
<td>2919 ms</td>
<td>2064 ms</td>
</tr>
<tr>
<td>other</td>
<td>3.50%</td>
<td>99.00%</td>
</tr>
<tr>
<td>Accuracy</td>
<td>100.00%</td>
<td>100.00%</td>
</tr>
<tr>
<td>Latency</td>
<td>2227 ms</td>
<td>1664 ms</td>
</tr>
</tbody>
</table>
Building a conversation with Watson Conversations

Intents

Create new

#check-temperature
get my wine temperature

#welcome

Add a new user example...

- good afternoon
- good morning
- hello
- hello my friend
- hi
# Building a conversation with Watson Conversations

### Entities

<table>
<thead>
<tr>
<th>My entities</th>
<th>System entities</th>
</tr>
</thead>
</table>

**Create new**

<table>
<thead>
<tr>
<th>@Places</th>
<th>Add a new value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Home</td>
<td>Air, house, inside</td>
</tr>
<tr>
<td>Wine</td>
<td>Cabernet, Cabernet Sauvignon, Chardonnay, Fermented Grapes, Grapes</td>
</tr>
</tbody>
</table>
Building a conversation with Watson Conversations

Dialog

Conversation starts

#welcome

Watson Response

"output": {
  "text": {
    "values": [
      "hello",
      "hi",
      "howdy",
      "#welcome"
    ]
  }
}

Anything else

#check-temperature

Enter something to test your bot
Pick Your Device

IBM IoT Foundation

Recipes
developer.ibm.com/recipes/

ARM mbed
BeagleBone with SensorTag
SimpleLink™ Wi-Fi® CC3200 LaunchPad
Intel Galileo
Raspberry Pi
Arduino Uno with Wi-Fi Shield
Device Simulator
Cloud services: Analytics, Voice, Storage, Security, Message Storage/Delivery, Wiring/Logic, Views…

Voice Activation/Control

MQTT

Devices: Connect
Publish-Status/Description
Subscribe - Commands

Device Connect & Control - Journey/Experience

Device Brokers

HomeKit

HealthKit

JSON
Wire new flows for your device
Device control - from speech to command

Publish

```json
{
  "d": {
    "id": "5B937D56-2E75-5293-BE2B-CB17C2EA539B",
    "name": "David's Home : iDevicesNightLight",
    "data": {
      "hue": {
        "writable": true,
        "step": 1,
        "max": 360,
        "value": 220,
        "format": "number",
        "min": 0
      },
      "on": {
        "value": true,
        "writable": true,
        "format": "bool"
      }
    },
    "location": {
      "lng": -71.15152086101887,
      "lat": 42.29974632421209
    },
    "iso": "2016-04-11T10:54:29.317-0400"
  }
}
```

MQTT

Subscribe

cmd/set

```json
{
  "d": {
    "id": "5B937D56-2E75-5293-BE2B-CB17C2EA539B",
    "on": true,
    "hue": 160
  }
}
```
MQTT

MQTT is simple to implement

Connect
Subscribe
Publish
Unsubscribe
Disconnect

client = new Messaging.Client(hostname, port, clientId);
client.onMessageArrived = messageArrived;
client.onConnectionLost = connectionLost;
client.connect({ onSuccess: connectionSuccess });

function connectionSuccess() {
    client.subscribe("planets/earth");
    var msg = new Messaging.Message("Hello world!");
    msg.destinationName = "planets/earth";
    client.publish(msg);
}

function messageArrived(msg) {
    console.log(msg.payloadString);
    client.unsubscribe("planets/earth");
    client.disconnect();
}

Eclipse Paho JavaScript MQTT client
MQTT

Quality of Service for reliable messaging

Publish to topic iot-2/evt/<event-type-id>/fmt/json

Subscribe to topic iot-2/cmd/<event_id>/fmt/json

QoS 0
at most once
- doesn’t survive failures
- never duplicated

QoS 1
at least once
- survives connection loss
- can be duplicated

QoS 2
exactly once
- survives connection loss
- never duplicated
The Future
What if…

Personal assistants or other **devices could learn**

The **conversational style of interaction** with devices is more than just asking the device to perform a set of static tasks.

We could **teach the devices new things through conversation**, combining tasks from an endless set of rich content components.

We are **programming by example, the example in this case is in the form of conversation** - do what I say! AND learn what I do!

In many ways, the **zero UI of a conversational interaction pattern is much easier**.

What better way to prescribe is there than to describe...
Why PaaS? What is Bluemix? - Sign up for a free trial

**(PaaS)** - for rapidly building, managing, and running cloud based applications and services of all types without worrying about the underlying infrastructure. Program in your choice of language.

**(IBM’s Bluemix)** - Built on open-standards and open source technologies: Cloud Foundry, OpenStack, MQTT, docker,…

State of the Art User Interface

**Services Catalog** containing Services/APIs for Mobile, Data, Enterprise data connectors, Cognitive, Analytics, Social and any callable Rest based service

Multiple flavors - **public, dedicated, on-premise, hybrid**
Thank you for your time

www.bluemix.net

Tweet   #IBMBluemix   @qconsf   @MarkVanderwiele
for more info see the following blogs

Jon Kaufman:  jkaufman.io
https://github.com/watson-developer-cloud/company-insights

Steve Atkin:  stevenatkin.com

Niklas Heidlof  http://heidloff.net

James Thomas  http://jamesthom.as/blog/categories/bluemix/

#IBMBluemix