BABBLER 2.0

CRYSTAL QIAN (CQIAN@)

INTRODUCTION

GOAL

I built Babbler 2.0, a friendly bot that uses vision and speech recognition systems to recognize its friends, their emotions, and the world around it.

This was originally built as a spoken word performance piece for the House of Sound show. On January 14th, 2017, I walked around around Woolworth Center with Babbler and a large speaker system for two hours. During this time, Babbler commentated on things that it saw and met members of the community.

However, Babbler can also be used in other fun ways. Have Babbler narrate a movie for you! Put on a blindfold and have Babbler guide you through an unknown environment! (This is not recommended.)

The focus of 2.0 is on human interaction. I noticed at House of Sound that people were most attracted to the part of Babbler that greeted and recognized people, and not so much the part that yelled out observations about the environment.

Babbler 2.0 can meet and greet people by name. Ultimately, it's a commentary on some of the everyday superficial interactions we experience.

PREVIOUS WORK

I started playing around with the idea of a "babbling machine" when thinking about translations from sight to sound and how we process the world around us.

To that end, Babbler 1.0 was built. It used a camera device to take in a live video stream, and "spoke" through the speakers of a computer in a woman's voice.

Here are some of its features:

Facial detection and recognition. Babbler 1.0 can predict the age and gender of people detected. It also has a limited memory, and will comment things like "I don't think we've met" or "We've met before."

Emotion recognition. Babbler 1.0 can predict the emotions of people detected. For example, it might notice if someone is angry, sad, happy, or surprised.

Object recognition. When not meeting people, Babbler 1.0 would babble about objects that it has seen. This was an adaptation of key terms extracted from frames using Microsoft's Cognitive Services. (Image from sample analysis shown below.)

Here's a demo of Babbler 1.0 in action. This was a successful idea in that the Babbler executed most of ideas I had in mind (the three features listed above).

There's a slew of technical issues from 1.0 that are fixed in 2.0, including timed-out API calls, broken speech issues, repetitive terms, etc.

Aside from that, I wanted to improve the human interaction portion of the project, so 1.0 fails in that its interaction is limited to facial detection, facial recognition, and emotion recognition.

2.0's main addition is vocal recognition and a focus on conversation. We use these things to further personalize a user's experience with the Babbler.

APPROACH / METHODOLOGY

APPROACH

Most of the recognition APIs were provided by Microsoft Cognitive Services, and the program was written in C#.

This program works best in an environment with fast Internet access; at the House of Sound performance, Babbler would sometimes stop speaking when the Internet connection timed out. Having multiple people in attendance at the time of the performance would also be ideal, due to the nature of the project.

METHODOLOGY

We implemented the following pieces to create Babbler 2.0.

API Interaction. We use a Face API to get the age and gender of faces that are detected. An Emotion API gauges emotions, and a Vision API returns terms found from images. I played around with some other options for Cognitive Services, but Microsoft had the widest breadth of products that were easily implemented in an application. Also, I found the term detection to be more accurate; for example, instead of "person face" as found by Google's detection system, I'd get "girl happy."

Commentary. To give Babbler a personality, I hard-coded a variety of different responses depending on certain situations, all randomly triggered with various probabilities. This might not have been the most elegant way to achieve this goal, but I couldn't find better alternatives.

For example, when Babbler detects that you are happy, it might comment "you are happy," but there's also a slight probability that it will say one of a few predetermined comments, like "I'm glad that you are happy because I am happy too." These responses had to be carefully timed.

Live feed processing. Babbler 1.0 took still frames at fixed intervals (3 seconds, 5 seconds, 7 seconds) and analysed what was seen at that point. There were some problems with this; for example, speech was inconsistent. For some frames, a lot of speech would be generated. For other frames, no speech. (We cache words spoken in the past time frame so there isn't excessive repetition). Additionally, since we're limited to a certain amount of API calls per second, we would run into our limit quickkly.

Our new approach is more elegant; it uses video movement detection to trigger frame checks, and only requests frames after the current speech is done.

Additionally, we take care to keep the queue of words to say relatively short, so that speech lines up approximately with the current frame being shown.

Persistent memory. Babbler 2.0 remembers the names and the faces of people that it has met. We keep a structure to store these things. There is a limit, however; to keep the program from getting too slow each time we detect a face, Babbler 2.0 is limited to remembering 20 faces at a time.

Speech detection output. When implementing a prototype of the Babbler using Google's products, I had to send "how do I pronounce this?" requests to Google Translate to create some kind of speech. The current approach uses Bing Speech, which provides a clean API and a variety of voices that can be used to speak text input. We can also use Bing Speech to capture the names of people being met.

Speech recognition: We decided to ultimately not implement speaker recognition. To train our system to remember someone's voice, we needed to the person to say a specific phrase three times for calibration. Since we wanted people to have a natural conversation experience with Babbler, we decided that speaker recognition as not beneficial towards this goal.

RESULTS / CONCLUSION

RESULTS

Since the final product was an application used to communicate with people, there aren't strict quantitative measurements used to evaluate this project.

However, I got some fun feedback on Babbler 1.0 at House of Sound, for example: "It's weird that we get so excited about the computer detecting things in our environment when we know what's already there, so it's not offering anything new, you know?"

I think that it brought joy to many people at this performance, so in my book, this was a success.

I shared a demo video on Facebook and it received positive feedback as well; the commentary part, although not the most graceful piece of code, received a lot of attention for its humor, so I think this was a much needed part.

I'm looking forward to demo-ing the new version on Thursday and receiving additional feedback.

DISCUSSION

To create a simple conversation bot with the features that our bot has, I think that the approach we took was the most straightforward and easiest to understand.

If we were to seriously develop this into a product or local application, I'd make improvements like localized facial detection and recognition to improve speeds.

There are many interesting potential applications for this. We could keep it outside of a house to detect and announce visitors, or keep it in a classroom to greet and remember the names of students (taking attendance, etc.). The advantage of this is that no additional identification or verification is needed on the side of the user; students don't need to use a badge or card for this process.

If we wanted to create an application highlighting the more problematic implications of facial recognition, we could pull up information about the person speaking when we ask for that peron's name, using facial recognition and existing images of that person to validate that the person is telling the truth. This might be an interesting next step.

For example, if Blabber 3.0 scanned my face and got my name, after verifying my face with, say, my Facebook profile picture, it might ask, "are you from Kansas?" since a Google result might indicate that I went to high school in Kansas.

CONCLUSION

In all, I think that this application is successful in sparking questions about how we have conversations. I noticed that I have the same conversation with so many people throughout the day, that follows the lines of "Hello! How are you? We haven't talked in a while. Let's grab a meal sometime." These conversations are largely immemorable.

By having this kind of conversation with Blabber 2.0, users might be more aware of the thoughtlessness put in these trite conversations and think about what needs to be done in the future to stimulate more effective conversation.