Talking to Machines and Being Heard: Getting Started With Speech Recognition

LumenVox® Speech Understood

by Dave Rich for UX Magazine
http://uxmag.com/articles/talking-to-machines-and-being-heard-a-speech-recognition-primer

Speech recognition presents an exciting and dynamic set of challenges and opportunities for UX designers. With the mass-market reception of consumer technologies such as Apple's Siri and the near-omnipresence of speech in telephone applications, speech recognition is a computer–human interface many people interact with daily. Speech applications range from self-service telephone systems such as banking applications, to mobile applications that allow users to speak commands and compose messages with their voice. In the future, we can expect to see many different applications integrate speech recognition in some form. The time is near when speech will be the most universal user interface.

An Overview of Speech Technology Jargon

Like many technologies, speech recognition has its own special lingo. Here are some of the basic terms:

Automatic Speech Recognition (ASR) and Text-To-Speech (TTS)

An ASR is a piece of software that turns spoken words into written text, and a TTS is software does the opposite, turning text into synthesized speech. They both tend to run in connection with an Interactive Voice Response (IVR) platform, which automates interactions with telephone callers via a media server.

Speaker Independent and Speaker Dependent

The majority of speech recognition used in commercial applications (including most mobile applications and telephone IVR systems) is speaker independent, which means the speaker is not known to the system before the interaction begins. Two common speaker-independent applications are phone banking and flight arrival/departure systems. Speaker independence simply means that the software is designed to understand a wide range of users.

Speaker dependent software, on the other hand, is used primarily for dictation with a PC and requires that a user "train" the system over time by speaking into it and correcting its mistakes. One popular brand of dictation software is Dragon Naturally Speaking.

While early generations of mobile ASR used speaker-dependent software, this is becoming increasingly rare as speech recognition moves off of mobile devices and into the cloud. Today, most speech interface design opportunities come in the IVR and mobile spaces, which are almost entirely built around speaker-independent dialogues that lead a user through an interaction.

Short Utterances vs. Natural Language Understanding (NLU)

When designing a VUI, it is important to consider the type of input, or utterances, expected from users. The simplest types of utterances are simple commands: "Main menu," "Checking account balance," "Transfer to operator," etc. This kind of speech application is often called directed dialog, as the system directs the user to give simple responses, e.g.,"You can say 'checking account,' 'savings account,' or 'main menu.'"

More complex applications make use of natural language understanding, which tends to be substantially more difficult to design and test. These applications are marked by long, natural sentences in response to very open-ended prompts such as, "Please state the reason for your call."

Not all ASRs provide the same capabilities and some may be limiting and constraining to the VUI. For example, an ASR that does not support NLU capabilities will obviously not allow for a very open-ended dialog. This interplay between technology and VUI design choices illustrates an important point for the designer working with speech: building a great user experience with speech recognition requires a blend of

technical knowledge of the science of speech technology with an understanding of the art of spoken human interface design.

The Human Factor: The Art of Creating a Great User Experience

There are many verbal communication factors to consider when building a VUI compared to a traditional GUI design. For instance, when users are presented with a web page, they are largely constrained and guided by the GUI design. They can click on links, hover over items, enter text with the keyboard, etc., making it relatively easy for the designer to anticipate the types of actions a user will attempt. With speech, however, users have spent their whole lives speaking naturally and may have very limited exposure to speech recognition applications. The highly contextual nature of conversational speech, where humans use intelligence and context to fill in the gaps of what's actually spoken in a conversation, make building good VUIs more difficult.

This must be kept in mind when creating a good spoken experience for users. An important consideration is the careful creation of audio prompts. Both the wording and the tone of an audio prompt will affect how users respond. Consider the following dialog:

> **SYSTEM:** Did you want to speak with "John Boston" or "Don Austin?"
>
> **USER:** Yes.

In this case, the user may be responding "Yes" without listening to the entire prompt, but the result is ambiguous. The designer of the dialog likely intended to get a response indicating which person the user was trying to speak with, but instead constructed a question where a yes/no response is grammatically accurate. A better way to construct the question is:

> **SYSTEM:** We have a "John Boston" and a "Don Austin." Which one would you like to speak with?
>
> **USER:** Don Austin.

There are many more of these sorts of human factors considerations—more than will fit in this article. Among But some of the other significant considerations in developing a good speech-driven VUI include the need to understand "turn-taking" in conversation between the speech system and the user, the cognitive limits that constrain how many choices a user can keep in his head at once, and how speech patterns change when a user becomes frustrated.

The Technical Aspects: The Science of Designing Good Grammars

A designer must also keep in mind the technical aspects of speech technology. After prompt choices, the most important element in speech design is what is known as a grammar. Grammars are requirements of speaker-independent software, and they essentially provide a structured list of the words and phrases that can be recognized at any given time. Grammars constrain the word choices that can be understood by the system. Speech cannot be recognized by the ASR unless it is contained within the grammar.

Designing good grammars is a challenge. Recognition accuracy tends to decrease as the number of word choices in a grammar increase because the more options available for the ASR, the greater the chance for it to make a mistake. The ideal grammar is big enough to cover the range of things users are likely to say while being small enough to be accurate. Good grammars flow naturally from good application and prompt design, as illustrated above. Well-worded prompts tend to elicit specific responses from users, allowing for more compact grammars.

Tuning: The Key to Accuracy

An important part of a designer's job when working with speech recognition is to be involved in the speech tuning process. This is a process where recorded audio from users of the application is captured and then transcribed manually. Using a tuning tool, these transcripts are compared to the recognition results from the ASR to understand how well the system is performing.

Tuning is an extremely important task because it reveals problems in the overall application and in the individual dialog sections that are known as "states." It can illustrate that users are getting lost in a complex interaction or that a specific grammar doesn't include enough variations on a phrase. A designer must therefore understand the science of tuning (including how to gather statically significant samples of data) and the design of VUIs to be able to find and fix any problems.

Error Handling: Avoiding Frustrated Users

As a designer's sophistication with ASR technologies increases, so does his ability to improve a user's experience. ASR engines that decode the audio into probable words provide a speech application with a wealth of data such as

confidence scores (a numeric representation of how likely the answer returned by the ASR matches what the user said) and n-best lists (a list of possible alternative things the user may have said in the event that the top result from the ASR choices is incorrect). These allow a designer to perform complex error handling that is often seamless to the user and leads to less of the dreaded "I'm not sure what you said," type of re-prompting. Apple's Siri product, for example, employs these tricks combined with metadata about a user's location and search habits to provide an elegant experience that rarely needs to ask for clarification.

Getting Started: How Long Will It Take?

The good news is that elegant speech applications are easier than ever to build and deploy thanks to a broad range of platform and toolset choices, mature standards, and established best practices. Budgeting the design and development of a speech application is tricky, but the complexity of a solution can usually be gauged roughly by looking at the number of interactions with the user in a standard use case and the complexity of each interaction. A yes/no question generally requires one turn (or response) with the user and is quite simple, while capturing a street address may require five turns with the user, some of which are quite complex.

In addition to the initial design and development time, developers must budget for tuning time. The industry generally recommends that about 40% of the initial design and development time be budgeted for tuning. Tuning is performed after the initial deployment and is usually done in multiple cycles, including a round that is done just after deployment. Other tuning rounds follow after the application has been running for a while with live users. Over time the need to tune decreases.

A new designer should generally budget a couple of weeks of dedicated time for designing a relatively simple speech application such as a 5-6 interaction directed-dialog IVR, with about the same amount of time dedicated to post-deployment tuning.

"Computer, compute to the last digit the value of pi." – Spock

Just as Hollywood has imaginatively shown us how easy communication should be with machines, simplicity only comes with good design. Since most computer applications have limited functions, a good speech interface needs to handle only a narrow slice of potential conversations that addresses the special input and output needs of that application. The art of speech user interface design is to craft clear questions and fully anticipate the range of potential responses at each stage of a structured conversation. While speech technology can handle variations in speaker accent, speed, volume, and tone, the UX designer is thoughtful about how to handle the many variations in situational and emotional context, and in word choice. Creating a good interface is easier than computing the last digit of pi; you merely need to focus on a limited application, obtain some assistance, and use tuning tools to improve it over time. This will get you down the road to a great speech application.