Beyond Misrecognitions and No Speech

By Rebecca Nowlin Green

At SpeechTEK West in San Francisco last year, a group of voice user interface designers participated in a workshop directed by James Larson of Larson Technical Services and Lizanne Kaiser of Genesys Telecommunications Laboratories. Participants developed recommendations for dealing with error messages in speech applications from four perspectives. This is the last in a series of four articles summarizing the recommendations reached by these experts. Contributors to this article were:

• Rebecca Nowlin Green, Nuance Communications;

• Lizanne Kaiser, Genesys Telecommunications Laboratories;

• Eduardo Olvera, Nuance Communications; and

• Greg Simsar, Syntellect

In the previous three articles in this series, you've read some ideas for how to move beyond the usual *I didn't get that* after a misrecognition error, or *I didn't hear anything* after a failure to detect speech. However, in all of those cases, some pretty tight constraints still existed. It has generally been assumed that designers will know only how much speech was detected by the recognizer, what was recognized, and what confidence score the recognizer assigned to the result. And while it's important to leverage a little more context, such as how many other errors have happened or what other recognition results might appear in the N-best list, we're frequently left wishing for even more information.

To illustrate how a lack of information can result in the same type of design we've been seeing since before the turn of the century, consider this scenario in which a mother at home is trying to give her account number to a speech recognition application:

System	*What's your account number?*
Caller	*Excuse me.* (Cough, cough, breath).
System	*I'm sorry, I didn't get that. Please say the number again.*
Caller	*OK, my account number is...* (Side speech: "Hey Mom! Where did you hide the peanut butter?" "Can't you see I'm on the phone?")
System	*Hmmm, I still didn't get it. Say your nine-digit account number once more.*
Caller	*One, eight, five...* (Doorbell, sigh). *Do you mind if I get the door?*
System	*Sorry I'm having so much trouble understanding. You can try calling back later, or hang on for the next available agent…*

Though the caller did nothing wrong, three errors occurred in the dialogue. In each exchange, the recognizer detected some speech but wasn't confident what it heard was a complete account number. We know this because the error messages said things like *I didn't get that* instead of *I didn't hear anything*. However, just because the system asked for an account number doesn't necessarily mean the caller gave a complete one. This design (like many) wrongly assumes that she did. That's why it keeps asking the caller to say her account number "again."

Knowing more about what went wrong helps to make it right. Imagine if the same caller had been speaking to an agent instead of an automated system. None of the errors would have caused any confusion at all, as in this scenario:

Agent	*What's your account number?*
Caller	*Excuse me.* (Cough, cough, breath).
Agent	*Wow, sounds like a nasty cough you have.*
Caller	*Yeah, I got it from my daughter in daycare. Anyway, my account number is...* (Side speech: "Hey Mom! Where did you hide the peanut butter?" "Can't you see I'm on the phone?")
Agent	*Kids have a great sense of timing, don't they?*
Caller	*Tell me about it. OK, that number is one, eight, five...* (Doorbell, sigh). *Do you mind if I get the door?*
Agent	*Sure, take your time…*

Despite the fact that the caller had the same troubles as in the first example, this conversation went much more smoothly because humans have the ability to understand speech that's not directly applicable to the question at hand, identify a broad array of environmental and non-speech audio, and react accordingly.

The designer can know more.

True, there will always be words and phrases a recognizer is not programmed to understand. Though it might not be easy, speech recognition engines can, or should, do several things that designers might want to start leveraging. We're not suggesting that voice systems begin offering health advice or child-rearing tips. However, we are suggesting that using certain pieces of information beyond the basics could result in better error handling. A few ideas include:

- Modeling common nonspeech sounds. There are already acoustic models of filler utterances, like "um" and "uh," just so we can identify and ignore them. Why not model other common sounds, like coughs and sneezes, barking dogs, doorbells, or honking horns, so that our designs react appropriately when they happen, too?
- Leveraging signal data. If another person (or a radio or television) is making sounds that can be picked up by the system, those sounds are likely to have less acoustic energy than the caller's voice. And the caller herself may turn her head or cover her mouthpiece when she's engaging in side speech. Why not compare the caller's volume from one utterance to the next and flag any significant changes?
- Using prosody. Pitch detection is already used for speech recognition of tonal languages, like Mandarin Chinese. Why not leverage prosodic information in other ways as well, such as to identify the slow hesitations of thought, the rate and volume of anger, or the rising intonation of questions like "Do you mind if I get the door?"
- Turning the timer on. Since a recognizer may already be logging data about utterance duration, why not compare that with the expected time it might take for a caller to answer a given question? It is likely that a too-short or too-long utterance from the caller either wasn't meant for the system to hear or wasn't really an answer to the question.

An innate human ability to do all four of the above tasks clued the agent into what was going on in our caller's environment. If similar information was easy to extract from the recognizer, speech system designers could be equally empowered. For example:

System	*What's your account number?*
Caller	*Excuse me.* (Cough, cough, breath).
System	*Whenever you're ready, I'd like to get that account number.*
Caller	*OK, my account number is...* (Side speech: "Hey Mom! Where did you hide the peanut butter?" "Can't you see I'm on the phone?")
System	*Hmmm, I'm not sure that was meant for me. I still need to get your nine-digit account number.*
Caller	*One, eight, five...* (Doorbell, sigh). *Do you mind if I get the door?*
System	*If you need me to wait a moment, say "Hold on." You could also try calling back later, or hang on for the next available agent…*

In this last example we strike a balance between human ability and what technology can arguably do. First, the system recognized the cough and didn't assume the caller tried to say the account number. Next, it detected the quieter volume and unexpected length of the side speech, and didn't assume the utterance was relevant to the conversation. Finally, it recognized the doorbell and anticipated the caller's need to step away. We didn't engage in small talk or understand exactly what was going wrong, but each response was much more relevant to the cause of the error and, therefore, made much more sense.

Two more things to consider are how each of these strategies complements the others and the intelligence they can provide about the overall interaction. Today's typical system is likely configured to count the number of errors allowed before triggering some kind of max-error condition (such as a transfer or disconnect). But imagine, for example, that once a system detects coughing at the beginning of the call, the design anticipates coughs for the remainder of the conversation. Certain related events, such as false barge-in or a lack of speech, may be planned for and not even be treated like an error at all.

These are just a few ideas for dealing with one scenario. Many more types of errors exist, as do many more tools to address them. Some are available now, while others, like emotion detection, are still works in progress.

Of course, there will always be things humans can do that speech systems cannot. That's the way it should be. But why not let our abilities to communicate with other humans in all sorts of contexts and environments also inspire us to push our designs to do better in some situations as well? In the end, we're all just trying to avoid the following scenario:

System	*I'm sorry, I didn't get that.*
Caller	*Well, duh! There was nothing to get.*