Intelligence by Brainhat[sm]

White Paper:

Natural Language Processing to Improve Speech Recognition

Natural Language Processing to Improve Speech Recognition

Natural language processing (NLP) promises to dramatically improve speech recognition by adding missing elements of context and intelligence to dialog. But it doesn't stop there; good NLP can take us beyond improvements in the human interface to new possibilities for programming and communicating with machines.

Imagine I called you on the phone this afternoon:

me: Hey! How's it going?
you: Who's this?
me: It's Kevin. Are you watching the game tonight?
you: Oh hi, Kevin. No. I have to go to my kid's school play.

Now imagine another conversation:

me: Hate. How's it going?
you: Who's this?
me: Cave-in. Are you watching the game tonight?
you: Oh hi, Kevin. No. I have to go to my kid's school play.

You would have understood me even though I deliberately substituted a word here and there. The reason you would understand me is that the conversation takes place in your mind just as much as it does on the telephone. You *interpret* what you hear based upon your expectations, the course of the conversation and your goals for the exchange. In other words, you used knowledge to improve the recognition rate.

Statistics and Constraints

The application of knowledge to improve the understanding capability of machines comes in several forms, possibly used in combination. Samplings of large corpra of dialog can provide a speech engine a statistical model of likely combinations of words and phrases. Given the N-best interpretations of something the user has said, the engine can consult the model to make a best guess. The models can be tuned for different applications to favor different interpretations, as would be the case with a medical corpus as opposed to, say, a legal corpus. Statistical samplings are especially important when an (almost) "*anything goes"* dictation capability is desired, as would be the case in medical or legal transcription. The statistical model embodies knowledge of what is likely to be said, and what is not.

Limiting the possibilities to just a handful of known interpretations can reign in the errors in a different fashion. Unlike a dictation grammar, a *context free grammar* (CFG) as is used within VoiceXML and CRM telephony applications describes all the possible combinations of words the application will accept at a given prompt. A CFG provides exceptional precision when the number of possibilities is small and when the utterances are limited to simple responses. However, CFGs are often employed without statistical models; if the grammar is enlarged to cover general dialog then the recognition rates of a CFG fall sharply.

Knowledge

We may one day liken the stochastic and set-limiting approaches to improving speech recognition discussed above the way we compare the operation of the brain stem to that of the rest of the brain. Knowledge-based dialog promises dramatic recognition rate improvements. But how do we get there? Roger Shank's work at Yale in the 1970s proposed the notion of a script--a blueprint for the course a conversation might take. In a restaurant, for instance, we know that the waiter or waitress is carrying silverware to make up a place-setting; we don't expect the waiter to plunge a knife in our back. Likewise, we know how a conversation about ordering dinner might go, largely because we have done it before.

Crafting a natural language dialog within a subject area--a domain--means identifying the elements of the domain and the scripts associated with it. Taking an order for a sweater, for instance, includes an exchange of information about size, color, material and cost. If we wish to delegate the job of conducting the order-taker to a machine, we need to provide it the vocabulary, script, goals and background knowledge of sweaters.

The concept of scripts lives on in natural language processing endeavors today, sometimes under other monikers such as "micro-theories" or "scenarios." But the goal is the same: deep understanding of the situation at hand and a sense for what comes next. Like your auditory senses providing your brain with a rough representation of something heard, a speech engine will provide a natural language package with the N-best interpretations or a lattice representing what the speaker has said. It will be up to the NLP component to "understand," or to push back to the engine for a better interpretation.

How might one go about instructing an NLP package to improve speech recognition? As a prerequisite, we would need a package that can perform deep understanding and knowledge representation. Likewise, it would need a capability of running inferences so that it could decide for itself whether an utterance makes sense, or whether it needs clarification. And it would need a composte of domain-specific knowledge to work from. Like you, it would need to be able to ask itself: "was I expecting that?" "Did that make sense?" "Are we talking about the same things?"

Given an N-best list of interpretations, our natural language operating system, Brainhat, can engage in this kind of speculation. If the interpretation is too outlandish, it will ask the user if that is really what they meant to say.

The next step is to take advantage of the semantic value of spoken language; we care about what the user *meant*, not what he said. Consider "you're pulling my leg" as a substitute for "I don't believe you." Even if a speech engine recognizes the utterance with no errors, it will not result in a meaningful semantic interpretation. Inferences are the key to understanding idioms at a semantic level: "if the user says that someone is pulling the user's leg then the user does not believe someone." By the same mechanism, one can give the NLP platform a belief system to accompany the domain; things the user says may be colored by the platform's own goals.

Beyond Improved Recognition

As you might guess, NLP capability at this level can provide more than an improved human interface. With a broad base of knowledge and an inferencing capability, the NLP component can become the application platform itself. This is why Brainhat's base product is called a *natural language operating system*. It is an active execution environment: the NLP engine can decide what comes next and what side effects should occur. The system might ask a question, send a piece of mail, perform a database look-up, make a phone call, etc. By handling natural language events from many sources all at once, the system becomes a clearinghouse for language-based events, hence an *operating system.*

In a well-crafted environment, the programming interface, like the application interface, should be natural language. Consider how you might "program" a kid working his first day at your ice cream parlor. You would explain: "the lever pumps the cones... the shots are in the bucket under the sink... never pump ice cream into your mouth..." Because we expect the kid to be able to interact with our customers in English, it isn't too much to ask that he be able to understand English explanations of the job at hand. Likewise, we should expect that our NLP platform should be able to understand our English description of the task at hand and follow our instructions. Brainhat provides a programming environment at this level. The basic ontology and grammar are programmed in a specific format, but the background knowledge and inferences are described in simple English.

Where We Are Going

We often talk about natural language processing and speech recognition in separate breaths, but the reality is that they are part and parcel to the same challenge. For speech recognition to approach Hollywood proportions--or even to meet today's expectations--will require an appreciation for the deep underlying meanings and implications of the spoken word. And this can only come from knowledge-based computing environment, like Brainhat.

About Brainhat

Brainhat Corporation is dedicated to the advancement of knowledge-based, natural language computing that allows humans and machines to interact in a more natural manner. Its Brainhat operating system provides the first platform to bring natural language programming and a natural language interface to commercial markets. The privately held company is located in East Hartford, Connecticut. For more information and online demos, visit www.brainhat.com.

111 Founders Plaza, 15th Floor, East Hartford, CT 06108 Ph: (860) 610-0606 Fax: (860) 610-0625