An Introduction to IBM Natural Language Understanding

An IBM White Paper

This white paper was written for anyone who is interested in learning about Natural Language Understanding (NLU) technology, particularly as it is applied within the realm of voice applications. It is broad in scope, aimed at anyone from a company CEO to an application developer.

Executive Overview

Businesses face many challenges today, not the least of which is finding ways to operate more efficiently. Take customer service as an example. It is an expensive proposition, and the less time a customer spends speaking directly with a customer service representative, the better it is for the enterprise (shorter calls cost the less money). Many businesses have thus adopted a self-service approach to customer service, through web-based solutions or telephone keypad driven Interactive Voice Response (IVR) systems. But businesses can operate even more efficiently with *Natural Language Understanding* (NLU) systems. Through a more natural interaction with an IVR system or a web site, users can understand how to complete their transactions more quickly and more easily. The speed of the transaction should improve, which can reduce call time and thus decrease the cost of providing customer service. Decreased call times also can increase call transaction volume.

Another challenge facing businesses today is customer satisfaction and retention. Most customers are familiar with and use IVR-based self-service systems. But they also desire anything that talks to them to behave more like a human in terms of understanding, conversational ability, and personality. Providing the end user with a pleasurable, comfortable experience is also important.

Natural Language Understanding combines breakthrough research and development in the fields of voice recognition, linguistics, statistics, human factors, and artificial intelligence. The technology can make the interaction between people and computer applications more intuitive and effective. When NLU is combined with speech recognition, the user is able to speak in a more unstructured, conversational style, resulting in a more comfortable and productive user experience.

As you can probably imagine, the technology behind NLU is quite complex. Let's look at the following example and see how an NLU application might work:

System:	*"Welcome to our Mutual Funds Trading System. How can I help you?"*
User:	*"I'd like to buy more Fidelity please"*
System:	*"You currently own 500 shares of Fidelity Magellan. How many shares would you like to buy?"*
User:	*"What's the going price?"*
System:	*"The current price of Fidelity Magellan is ... What else can I do for you?*
User:	*"Hmmmmm...I think I'll buy 300 more"*
System:	*"Purchasing 300 shares of Fidelity Magellan at market price...please say yes to confirm."*

User:	*"No, make that 2000 dollars"*
System:	*"Purchasing 2000 dollars of Fidelity Magellan at market price...please say yes to confirm."*
User:	*"Do it."*
System:	*"400 shares of Fidelity Magellan purchased at market price. What else can I do for you?"*

When the user says "I'd like to buy more Fidelity please," the software must ***extract the meaning*** from what was said. Clearly, there are extra words in this sentence -- words that don't effect the meaning of the sentence -- and the software must extract the pertinent words from the sentence to determine its meaning. In this sentence, the key concepts are "buy" (an action), "more" (implying a fund the user already owns), and "Fidelity" (a fund company).

As you can see from its response, the application determined that the user owns the Fidelity Magellan fund and asks how many shares the user would like to buy. In this example, the user responds NOT with the number of shares to buy, but with a totally different question, "What's the going price?" ***The user is in control*** of the conversation, not the application. At this point, the application must "switch gears" and obtain the current price of a fund. From the ***conversation history*** (that is, what was previously said by the user), the application knows that the user wants the price of Fidelity Magellan.

The next thing the user says is "Hmmmmm...I think I'll buy 300 more." Again, the application must extract meaning from this sentence; in this case, the key words are "buy" and "300." Again, the application can use conversation history to determine the fund to buy (Fidelity Magellan).

When the application confirms the purchase of 300 shares, the user responds "No, make that 2000 dollars," and the application confirms the purchase of $2000 worth of Fidelity Magellan. The user responds "Do it," and the application completes the transaction.

From the user's perspective, the process seems simple. Interacting with the application should be very similar to interacting with a live human agent (in this case, a mutual funds broker). And that's the whole point of NLU -- allow the user to get things done for himself/herself, but in a much more pleasant, productive fashion than previous systems have allowed.

NLU Components

Conceptually, NLU technology is comprised of two components:

- **Natural Language Dialog** - The way in which a user can provide input to the application and the way the application responds to the user.
- **Natural Language Processing** - The work done by the software to process what was said by the user. Essentially, this means that meaning is extracted from the dialog and from the conversation history.

Let's examine these two components a little more closely.

Natural Language Dialog

There are essentially three types of interactions -- or dialog styles -- that are available to software applications:

- Directed dialog
- Mixed initiative dialog
- Natural language dialog

In a **directed dialog**, the application is in control of the interaction. It directs the user to perform a specific task by asking for information at each turn of the dialog, expecting specific words or phrases in response. Here is a simple example of a directed dialog application:

System:	*"Welcome to the IBM Directory Dialer. Please say the location"*
User:	*"West Palm Beach"*
System:	*"West Palm Beach...Please say the name"*
User:	*"Kim Kemble"*
System:	*"West Palm Beach...Kim Kemble...Dialing..."*

You can see that this application directs the user to provide very specific responses at each turn in the dialog. First, the application asks the user to say the location. When a valid location is provided, the application then asks the user to say the name of the person. With those two pieces of information, the application can then dial the correct telephone number.

You can design very effective, highly usable directed dialog applications. Remember, though, that the application is in control in this type of interaction, not the user, and the user is allowed to say only a very specific set of words and phrases in a very specific order. Some users want

more control and flexibility in their interaction with the application...That's where the other two dialog styles come into play.

The second dialog style is known as **mixed initiative**. This style of interaction allows the user to have as much or as little control as he or she desires. In a mixed initiative dialog, users can provide more input than what is being requested at a particular turn in the dialog, and in a different order than it may be expected in. Take the example shown here:

System:	*"Welcome to the IBM Directory Dialer. Please say the location"*
User:	*"West Palm Beach...Kim Kemble"*
System:	*"West Palm Beach...Kim Kemble...Dialing..."*

The application requests a location, but the user responds with both a location and name. In a mixed initiative dialog, that's OK! The application grabs both pieces of information and continues from there -- in this case, it completes the transaction, because all of the required information (a location and a name) was provided in a single utterance.

In this same example, if the caller had said "Kim Kemble...West Palm Beach", the application should still be able to decipher the utterance correctly, even though it is in a different order than expected. As you can imagine, it takes much more intelligence in the application to support a mixed initiative dialog, and more complexity in the design of your application grammars as well, but the usability payoff can be significant.

The final dialog style is known as **natural language conversation**. In a natural language dialog, the user enjoys a more unstructured, conversational interaction with the application, as illustrated in the following example:

System:	*"Welcome to the IBM Directory Dialer. How can I help you?"*
User:	*"I'd like to call Kim Kemble in Palm Beach please"*
System:	*"West Palm Beach - Kim Kemble. Is this correct?*
User:	*"I need her cell phone number"*
System:	*"The cell phone number is 561-465-8336. What else can I do?"*
User:	*"Well, I think I'll just leave her a message."*
System:	*"Dialing West Palm Beach - Kim Kemble - voice mail..."*

Even though it's a simple interaction, you can see that the caller is speaking more naturally. In the first utterance ("*I'd like to call Kim Kemble in Palm Beach please*"), the application must extract the pertinent information ("*Kim Kemble*") and the location ("*Palm Beach*"). It must also determine that "Palm Beach" is really "West Palm Beach."

A natural language application also uses conversation history to decipher an utterance. In the second utterance ("*I need her cell phone number*"), the application must again extract the

pertinent information -- in this case, "*cell phone number*." It also determines that "*her*" is referring to Kim Kemble, since that is the context within which the user was previously speaking.

Natural Language Processing

As you've seen in the previous examples, NLU applications offer a lot more flexibility in the way the user provides input. The user can say things many different ways, allowing the context of the conversation to augment what is said. As a result of an NLU interaction, the speech recognition and meaning extraction challenges are considerable.

NLU also allows the user to provide more than one piece of information at a time. Repeat users of a voice-driven application will know ahead of time all the choices that are needed to complete a transaction and will want to accelerate the process. Problems associated with entering multiple pieces of information in a single "turn" include arbitrary subsets of necessary choices as well as highly varied syntax for specifying the information. The speech recognition and information extraction problems are again more difficult because of the greater variety of things that can be said.

NLU capability (and mixed initiative dialog, for that matter) extends system design complexity significantly. Speech recognition must allow a much broader variety of input. Additional technology must be employed to be able to resolve abstractions, glean the meaning of arbitrary inputs, and to allow complex transitions from one type of transaction to another. At the same time, conversational history must be used to "inherit" parameters.

There are two approaches to how NLU can be implemented:

- Grammar-based NLU
- Statistical NLU

Let's examine grammar-based NLU first. In this type of system, all potential user utterances (words, phrases, and sentences) must be explicitly predefined to the system in the form of a grammar (or multiple grammars). Grammars are much more than just a list of the possible words and phrases. There are syntactical rules and expressions that can be employed to define variability and flexibility in what can be said.

Let's see what this means in terms of an example. How many ways can someone ask for the price of a mutual fund?

"What's the price of ..."
"Give me the price of ..."

"Gimme the price of ..."
"Can I have the price of ...?"
"I'd like to know the price of ..."
"What's ... going for these days?"
"What's the market value of ...?"
"Tell me the price of ..."
"Get me the latest quote on ..."
"I need the current price of ..."

And so on. You can define all of these sentences (and much, much more) as possibilities in a grammar. And you're going to want to do that -- to reflect all of the things that your users are going to say. You can imagine, though, that the grammar is going to get quite complex quite quickly, and that effective grammar design requires significant linguistic and human factors experience.

Furthermore, the main limitation of a grammar is that if the user says something (a word, phrase or sentence) that is not explicitly defined in the grammar, no matter how much flexibility and variability you've allowed for, it will not be recognized. For example, if the caller says *"Can I PLEASE have the price of ..."* and that was not defined in the grammar, it would not be recognized by the speech recognition software. And that defeats the purpose of NLU.

There is a technique known as *word spotting* where grammars are used to extract possible meanings from utterances. This technique allows greater flexibility in what the user can say. However, because the recognition and meaning extraction functions in word spotting systems are so tightly coupled, the extent of the meanings that can be extracted in this type of grammar-based system is restricted.

Now let's look at Statistical NLU systems. These systems employ a *model* of utterances rather than a specification of all potential utterances. These models are typically derived from actual conversational data. By definition, a model is a smaller object that represents a larger object. In the case of statistical NLU, the larger object that the model represents is an entire language domain. Mutual funds, travel, or weather are all language domains.

The statistical NLU model contains enough information to effectively represent the larger language domain, but it does not contain every possible utterance. Using artificial intelligence, linguistic knowledge, and conversation history, a statistical NLU system can correctly process and interpret the correct meaning from unanticipated utterances (whereas grammar-based systems require exact, verbatim utterances that have a specific match within the grammar).

Thus statistical NLU, which separates the recognition vocabulary and phrase construction from the meaning extraction function enables a broader input set. It allows handling ambiguous meanings, inferred meanings, context based evaluation, and many other constructs and

techniques which provide a richer understanding of the recognized text than is possible with grammar-based systems.

NLU Application Development

Without question, NLU technology adds complexity to the development of voice applications. This complexity can affect the creation of the language model, the application development environment, and the application execution environment. Even so, NLU can provide benefits to both the enterprise and the end user that far outweigh the costs of implementation.

Let's look at both the grammar-based NLU and statistical NLU in terms of how each approach affects the development of an NLU application.

Creating the language model

An NLU dialog is created by specifying a language model. The *language model* defines the speech recognition domain for the system. It defines the potential utterances that may be spoken by the user and recognized by the system. This language model may be specified in one of two ways: either by developing grammars or by developing a statistical model.

As discussed in the previous section, grammar-based NLU requires considerable effort be invested in developing a syntactical specification -- a grammar or set of grammars -- that defines all of the potential utterances the user can say and the system can recognize. To enable even a moderate amount of flexibility in the way users can provide input, grammars can become quite complex quite quickly. Furthermore, in the end, if the user says something that's not defined within the grammar, the application is not going to recognize what the user said, and that is contrary to the objectives of an NLU system.

It is very difficult to account for every possible way the user is going to say something. Although explicit grammars can be used to create NLU systems, creating such grammars requires very high linguistic expertise. Anyone wanting a personal feeling for this should look at a grammar designed simply to accept natural numbers or amounts. Furthermore**,** the more flexibility and variability that is allowed in the input, the longer it takes to develop the corresponding grammars. This can increase the timeline to implement and test the application, thus increasing the overall project cost.

This is not to imply that NLU language models are not complex in themselves. However, in the statistical NLU system, the language model creation process is more automated than the grammar-based approach, where complicated grammars are hand-crafted. The NLU language

model is created using a *statistical training process* based on actual conversational data. By automating the process, these complex models can be generated without the heavy dependence on expensive linguistic skills required for grammar based NLU solutions.

Development environment

Grammar-based NLU application development closely matches the way voice applications are developed today. There are grammar development skills already available in the industry, albeit typically simpler in nature than what is required for NLU capability. Thus, grammar-based NLU generally results in a shorter initial implementation timeline than statistical NLU applications.

Statistical methods based on training systems from actual conversational input provide a more "cookbook" approach to the development of voice applications. NLU systems not only provide tools to create the language model, as discussed, but also provide tools that tie together all the aspects of a voice application, including speech recognition, meaning extraction and dialog control.

Statistical NLU systems requires a larger initial system implementation than grammar-based solutions for new domains. Data collected is highly domain specific (weather application data is not usable for a stock application). Thus, it typically takes longer to develop a pilot system than does a grammar-based solution.

Application execution environment

As grammar complexity increases, the size of the grammars also increases. The system requirements to support these grammars grow as well. Thus, there is a potential limit to the size of grammars that can be handled, and there are also performance tradeoffs to consider. For example, response time from the speech recognition engine increases with the size and complexity of the active grammars, thus resulting in processing delays that may be perceivable by the end user. Recognition accuracy may degrade with larger and more complex grammars, not to mention the problem we've discussed several times -- that if the user says something that's not in the grammar, it won't be recognized. Finally, the number of concurrent speech recognition engines you can run per CPU may be limited as a result of the size of the grammars the application supports. At a certain point of complexity, then, grammar-based NLU systems hit a limit.

On the other hand, NLU systems typically have greater initial system requirements to support the execution environment because of the complexity of the data extraction and conversation history. However, the language model can be extended without affecting the overall system footprint.

The Opportunity for NLU

While it is clear that NLU brings many benefits to the enterprise and to the end user, not all applications are suitable for NLU. Many simple applications do not warrant the investment in NLU due to the minimal incremental value that NLU would bring to the business. This coupled with the limited development resources typically available for such applications would make NLU implementation prohibitive.

Additionally, NLU can provide the attraction of deploying the latest technology, which appeals to many businesses as a differentiator. It can also appeal to the end user as a way to use the latest technological advancements.

IBM Solutions

IBM Natural Language Understanding (NLU) technology allows a user to interact with computer applications in much the same way as they would when dealing with a person. IBM NLU adds a natural dimension to speech technology by supporting unstructured, conversational dialog rather than requiring the user to speak specific commands. It puts the user in control of the interaction, eliminating long menu options, keypad-driven transactions, and hard-to-remember commands for easy, transparent access to information and services.

With IBM NLU, the user interacts with the system in much the same manner as when conversing with another human. This means that a very large vocabulary -- a virtually unlimited set of words and phrases -- can be spoken.

IBM NLU systems use statistically-based models which provide more flexibility and robustness than traditional methods. IBM's NLU eliminates the limitations of speech recognition grammars that are typically used in speech applications today and provides tools that can make the application development process easier and quicker.

With statistical language modeling supporting wider recognition, and statistical parsing allowing more flexible meaning extraction from user utterances, the third pillar of the IBM NLU solution is built upon sophisticated dialog management. A form-based dialog manager provides the flexibility to process natural dialog containing interjected context changes, ambiguous information, extraneous information, and complicated interdependencies. The form dialog manager provides the call flow designer with automatic selection of the most appropriate prompts, integrated confirmation of spoken, inferred, and inherited values, computational support to infer the user’s goals so that information can be recognized prior to actual need, and

many other features needed when the user is in control of the dialog, rather than when the system is directing the dialog.

IBM can provide services to help you realize the benefits of this state-of-the-art technology. For more information, please visit our Web site at ibm.com/software/voice or e-mail any inquiries to Talk2Me@us.ibm.com.

For a demonstration of IBM's NLU technology in action, call the Mutual Funds demo at 1-877-842-8642 (or outside of the U.S., call 1-972-402-5963).

Conclusion

The last decade has seen an explosion in the deployment of IVR systems and, even more recently, in the deployment of speech recognition solutions. NLU technology can take the convenience and usability of these systems to another level by enabling a more intuitive, comfortable and productive interaction for end user, which can increase customer satisfaction and retention.

NLU solutions offer significant benefits to both the enterprise and the end user. They allow users to not only get things done for themselves (self-service means decreased cost of business), but also to get those things done in a manner that's comfortable and pleasant.

Natural Language Understanding (NLU) technology is also leading-edge, and businesses offering NLU solutions will be able to differentiate themselves from their competitors.

Notices

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes may be periodically made to the information herein; these changes will be incorporated in new editions of the publication, if any. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM publications are provided for convenience only and do not in any manner serve as an endorsement of those Web sites or publications. This information is not part of this white paper and should be used or viewed at your own risk.

IBM is a trademark or registered trademarks of International Business Machines Corporation in the United States, other countries or both.

© Copyright IBM 1993-2001. All Rights Reserved.