Proc. Natl. Acad. Sci. USA
Vol. 92, pp. 9977–9982, October 1995
Colloquium Paper

This paper was presented at a colloquium entitled "Human–Machine Communication by Voice," organized by Lawrence R. Rabiner, held by the National Academy of Sciences at The Arnold and Mabel Beckman Center, in Irvine, CA, February 8–9, 1993.

Models of natural language understanding

MADELEINE BATES

BBN Systems and Technologies, 70 Fawcett Street, Cambridge, MA 02138

ABSTRACT This paper surveys some of the fundamental problems in natural language (NL) understanding (syntax, semantics, pragmatics, and discourse) and the current approaches to solving them. Some recent developments in NL processing include increased emphasis on corpus-based rather than example- or intuition-based work, attempts to measure the coverage and effectiveness of NL systems, dealing with discourse and dialogue phenomena, and attempts to use both analytic and stochastic knowledge. Critical areas for the future include grammars that are appropriate to processing large amounts of real language; automatic (or at least semi-automatic) methods for deriving models of syntax, semantics, and pragmatics; self-adapting systems; and integration with speech processing. Of particular importance are techniques that can be tuned to such requirements as full versus partial understanding and spoken language versus text. Portability (the ease with which one can configure an NL system for a particular application) is one of the largest barriers to application of this technology.

Natural language (NL) understanding by computer began in the 1950s as a discipline closely related to linguistics. It has evolved to incorporate aspects of many other disciplines (such as artificial intelligence, computer science, and lexicography). Yet it continues to be the Holy Grail of those who try to make computers deal intelligently with one of the most complex characteristics of human beings: language.

Language is so fundamental to humans, and so ubiquitous, that fluent use of it is often considered almost synonymous with intelligence. Given that, it is not surprising that computers have difficulty with natural language. Nonetheless, many people seem to think it should be easy for computers to deal with human language, just because they themselves do so easily.

Research in both speech recognition (i.e., literal transcription of spoken words) and language processing (i.e., understanding the meaning of a sequence of words) has been going on for decades. But quite recently, speech recognition started to make the transition from laboratory to widespread successful use in a large number of different kinds of systems. What is responsible for this technology transition?

Two key features that have allowed the development of successful speech recognition systems are (*i*) a simple general description of the speech recognition problem (which results in a simple general way to measure the performance of recognizers) and (*ii*) a simple general way to automatically train a recognizer on a new vocabulary or corpus. Together, these features helped to open the floodgates to the successful, widespread application of speech recognition technology. Many of the papers in this volume, particularly those by Makhoul and Schwartz (1), Jelinek (2), Levinson (3), Oberteuffer (4), Weinstein (5), and Wilpon (6) attest to this fact.

But it is important to distinguish "language understanding" from "recognizing speech," so it is natural to ask, why the same path has not been followed in natural language understanding. In natural language processing (NLP), as we shall see, there is no easy way to define the problem being solved (which results in difficulty evaluating the performance of NL systems), and there is currently no general way for NL systems to automatically learn the information they need to deal effectively with new words, new meanings, new grammatical structures, and new domains.

Some aspects of language understanding seem tantalizingly similar to problems that have been solved (or at least attacked) in speech recognition, but other aspects seem to emphasize differences that may never allow the same solutions to be used for both problems. This paper briefly touches on some of the history of NLP, the types of NLP and their applications, current problem areas and suggested solutions, and areas for future work.

A BRIEF HISTORY OF NLP

NLP has a long, diverse history. One way of looking at that history is as a sequence of application areas, each of which has been the primary focus of research efforts in the computational linguistics community, and each of which has produced different techniques for language understanding. A number of excellent references are available that survey the field in various ways (7–11).

In the 1950s, machine translation was the first area to receive considerable attention, only to be abandoned when it was discovered that, although it was easy to get computers to map one word string to another, the problem of translating between one natural language and another was much too complex to be expressible as such a mapping.

In the 1960s the focus turned to question answering. To "understand" and respond to typed questions, most NL systems used a strongly knowledge-based approach, attempting to encode knowledge for use by a system capable of producing an in-depth analysis of the input question. That analysis would then be used to retrieve the answer to the question from a database.

In the 1970s interest broadened from database interfaces to other kinds of application systems, but the focus was still on the kinds of natural language that would be produced by a person interacting with a computer system—typed queries or commands issued one at a time by the person, each of which needed to be understood completely in order to produce the correct response. That is, virtually every word in the input had some effect on the meaning that the system produced. This tended to result in systems that, for each sentence they were given, either succeeded perfectly or failed completely.

The 1980s saw the first commercialization of research that was done in the previous two decades: natural language

The publication costs of this article were defrayed in part by page charge payment. This article must therefore be hereby marked "*advertisement*" in accordance with 18 U.S.C. §1734 solely to indicate this fact.

database interfaces and grammar and style checkers. For the first time, widespread interest began to be paid to systems that dealt with written language in paragraphs or larger chunks, instead of typed interactions. There were even some attempts to generate natural language, not just understand it.

Another change during this decade was the beginning of a redefinition of the fundamental goal of NL systems, which had always been to process every word of the input as deeply as necessary to produce an understanding of the sentence as a whole. Researchers began to think that this goal was not just difficult to achieve but perhaps impossible, and perhaps even unnecessary! Instead, partial understanding (in which some words of the input were completely ignored), which had been viewed as a failure and a problem that needed to be fixed, began to be seen as a meaningful and useful goal. It was discovered that systems which tried to extract at least some meaning from nearly every input could succeed better (at least for certain applications) than systems that tried (and often failed) to extract the complete meaning of every input. The old model of complete understanding or complete failure began to give way to the notion of partial correctness.

Today, in the 1990s, there is strong interest in a wide spectrum of tasks that require NL processing. Spoken language systems (SLSs) (which combine speech recognition with language understanding), language generation, message processing (the term used for systems that deal with bodies of written language in a noninteractive way, including document indexing and retrieval, text classification, and contents scanning, which is also called data extraction), and interactive NL interfaces are all important research areas. Even machine translation has come full circle and is now being reinvestigated using the results of more than 30 years of research, although this time around there is interest in doing speech translation (e.g., a translating telephone) as well as text. Reports of current research in all these areas can be found in the journal of the Association for Computational Linguistics and in the proceedings of various workshops that are included in the bibliography.

The emphasis has changed over the years not only in the type of applications that are of interest but also the "reality" of the language studied. Originally, toy problems with examples of language made up by researchers and linguists were all that a system could be expected to handle. But the development of large sharable corpora (often with additional attached information such as part-of-speech assignment, or structure, or question andanswer) has revolutionized the study of language understanding. Now it is considered absolutely necessary for good research to examine "real" language, preferably a large linguistic corpus obtained using a real application in as natural a setting as possible. Many of these corpora and some systems to process them are available through the Linguistic Data Consortium at the University of Pennsylvania, the Consortium for Lexical Research at New Mexico State University, and the Treebank Project at the University of Pennsylvania.

WHY IS NLP DIFFICULT?

One way to illustrate the problems of NL processing is to look at the difference between the fundamental goals of a speech recognition (SR) system and an NL system. As illustrated in Fig. 1, SR is well defined in terms of input and output. The input is a speech signal, and the output is a word string (possibly a set or lattice of alternative word strings, possibly with scores or probabilities attached). Despite the fact that there are a few nontrivial problems in deciding what is a word, it is fairly easy for two or more speech researchers to come to agreement on what is considered a word (e.g., that BOOK and BOOKS are two different words and that AIR_FARE is a collocation) and on metrics for evaluating the quality of SR systems.

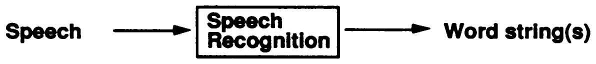

FIG. 1. Input/output for speech recognition is easy to define.

The word error rate, which incorporates insertions, deletions, and substitutions, has been the generally accepted metric for many years; it is widely accepted, easy to apply, and works so well that there is little reason for the speech research community to change it. Because the SR task is so well defined, it is fairly easy to tell whether an SR system is doing a good job or not, and it is very easy to tell, given two different SR systems with identical input, which performs better.

Computational linguists envy this straightforward problem definition and unambiguous criterion for success! It is quite a different matter in NL processing, which is extremely difficult precisely because the input/output characteristics of an NLP system are varied, hard to specify, difficult to get common agreement on, and resistant to the development of easily applied evaluation metrics.

The range of possible inputs to an NLP is quite broad. Language can be in a variety of forms, such as the (possibly imperfectly recognized) output of an SR system, paragraphs of text (possibly containing some material that is not natural language), commands that are typed directly to a computer system, etc.

The input language might be given to the system a sentence at a time or multiple sentences all at once. It might not be sentences at all in the sense of complete grammatical units but could be fragments of language or a mixture of sentences and fragments. The input might be grammatical, nearly grammatical, or highly ungrammatical. It might contain useful cues like capitalization and punctuation or (particularly if the input comes from a speech processor) all punctuation and even the sentence boundaries might be missing.

There is not even a good way to refer to the language that is input to an NL system. "Sentence" implies grammaticality, or at least unity of the words involved, and also implies a fairly small number of words. "Utterance" implies speech but does not imply any degree of completeness or grammar. "Word string" might be better, but it seems to exclude speech input. I will use "input" as a general term that includes all the types of language input mentioned here.

The ultimate output from a system that incorporates an NLP might be an answer from a database, a command to change some data in a database, a spoken response, or some other action on the part of the system. But these are the output of the system as a whole, not the output of the NLP component of the system—a very important distinction.

This inability to specify a natural, well-defined output for an NLP system will cause problems in a variety of ways, as we shall see more of below.

Another example of why language processing is difficult is illustrated by a recent Calvin and Hobbes cartoon:

Calvin:	I like to verb words.
Hobbes:	What?
Calvin:	I take nouns and adjectives and use them as verbs. Remember when "access" was a thing? Now it's something you do. It got verbed.
Calvin:	Verbing weirds language.
Hobbes:	Maybe we can eventually make language a complete impediment to understanding.

Understanding what Calvin meant by "Verbing weirds language" stretches the limits of human language performance. One of the reasons that NL is challenging to computational linguists is its variety. Not only are new words frequently introduced into any natural language, but old words are constantly reused with new meanings (not always accompanied by new morphology).

WHAT IS IN AN NLP SYSTEM?

There are many good overviews of NL processing, including those by Allen (7), Gazdar and Mellish (8), Smith (9), and Winograd (10), and state-of-the-art research results, including those of Bates (12) and Bates and Weischedel (13). Fig. 2 shows a generic NLP system and its input-output variety. Fig. 3 shows a typical view of what might be inside the NLP box of Fig. 2. Each of the boxes in Fig. 3 represents one of the types of processing that make up an NL analysis.

Most NL systems have some kind of preprocessor that does morphological analysis, dictionary lookup, and lexical substitutions (to normalize abbreviations, for example), and part-of-speech assignment. The order in which these processes are performed, the techniques used to perform them, and the format of the result are highly idiosyncratic.

Syntax

Syntactic processing is without doubt the most mature field of study in the NL field. Some think of syntax as just a way of checking whether the input is well formed, but in fact syntactic analysis has at least two uses. One is to simplify the process of subsequent components as they try to extract meaning from the input. A second use of syntactic analysis is to help detect new or unusual meanings.

Without syntactic analysis it might be possible to use semantic probabilities to determine that a string containing "boy" and "dog" and "bit" means that a dog bit a boy, but syntax makes it easy to determine who bit whom in the input "boy bit dog." Calvin's observation that "Verbing weirds language" can be understood only by using morphological and syntactic cues, not by semantics alone.

Various syntactic formalisms have been developed and implemented in great detail, including mathematical analyses of their expressive power. Most are useful; all give incomplete accounts of the wide range of NL phenomena (as one linguist put it, "All grammars leak."). Augmented transition networks (a procedural language with most of the simplicity of context-free grammars but able to capture many context-sensitive aspects as well) were once quite popular, but in the mid 1980s a shift began toward declarative formalisms, such as the combination of context-free rules with unification.

Semantics

Semantics is a more serious problem. The output of the semantic component is the "meaning" of the input. But how can this "meaning" be expressed? Not by anything as simple as a sequence of words. Many different "meaning representation languages" have been developed in an attempt to find a language that has the appropriate expressive power, but there is no uniform semantic representation language that can represent the meaning of every piece of NL.

Indeed, some would argue that there is no such thing as "the (i.e., unique) meaning" of a string of words, because the

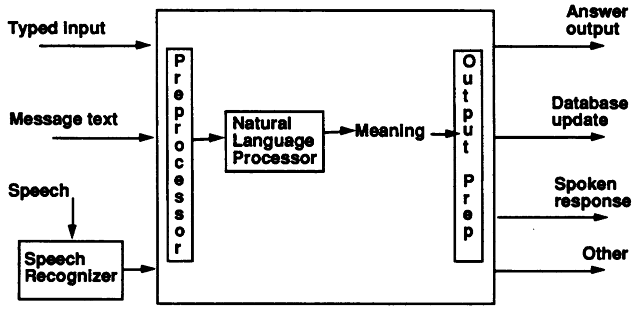

FIG. 2. A generic NL system.

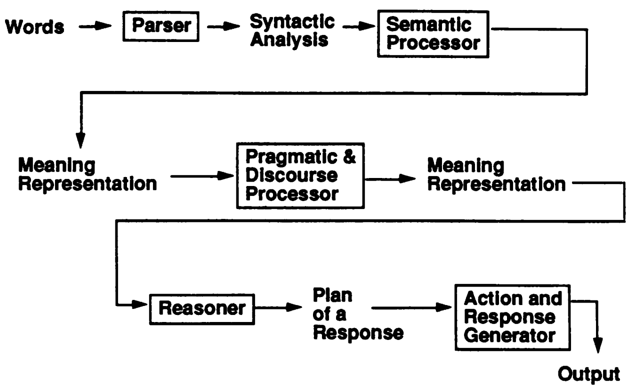

FIG. 3. A pipeline view of the components of a generic NL system.

meaning can be greatly influenced by the context in which the words are used and by the purpose the words are intended to achieve.

Three general kinds of semantic representations are in wide use: propositional logic (most frame-based semantic representations are equivalent to this, since they do not allow quantification); First- Order Predicate Logic (FOPL, which does allow quantifiers); and various representations that can handle expressions not representable in FOPL (see, e.g., R. Montague, ref. 14).

First-order predicate logic is a good representation for some types of meaning, but it is unnecessarily complex for simple applications where quantifiers do not frequently occur, and it is not rich enough for others. And even FOPL requires prespecification of the atomic concepts from which the FOPL expressions are built. Even if it were possible to express the meaning of any sentence in FOPL, the meaning would not be unambiguous.

Semantic processing methodologies are often chosen to match the characteristics of a particular application domain. For database access, meanings can generally be expressed in some form of predicate logic. For updating a database with information extracted from a body of text, it is crucial to be able to characterize in advance the kinds of information to be extracted; this allows the operation of the semantic component to be guided in part by the (very narrow) range of possible meanings.

Even when the problem is limited to a single application, it is very hard to get people to agree on the form of the output that a semantic component should produce.

Discourse and Pragmatics

Modeling context, and using context appropriately, is one of the least well understood and most difficult aspects of NLP. Unlike context in speech, which is quite localized in time, NL context is all pervasive and extremely powerful; it can reach back (or forward) hundreds of words. It is the difficult task of the discourse and pragmatics component to determine the referents of pronouns and definite noun phrases and to try to understand elliptical sentence fragments, dropped articles, false starts, misspellings, and other forms of nonstandard language, as well as a host of other long-range language phenomena that have not even been adequately characterized much less conquered.

The pragmatic component must alter itself as a result of the meaning of previous inputs. Unlike speech, where the context that influences a particular bit of input is very close by, NL context can span multiple sentences, multiple paragraphs, or even multiple documents. Some NL expressions are forward referencing, so the relevant context is not always prior input.

But feedback across sentences is not limited to pragmatics alone. The reasoning component, or even the output generator might need to change the discourse state so that, for example, subsequent input will be understood in the context of the output that was returned to the user. This feedback is extremely important for NLP applications because real language rarely occurs in isolated sentences.

Reasoning, Response Planning, and Response Generation

For straightforward inputs (e.g., a query or command like "List the flights from Boston to Topeka that leave on Saturday morning"), it is usually possible to come up with a representation that captures the literal meaning well enough to answer the question or carry out the command, but sometimes additional reasoning is necessary.

For example, what should be done with the following input to a travel agent system: "I want to go from Pittsburgh to Boston"? Should that be interpreted as a request to list the flights from Pittsburgh to Boston, or should the effect merely be to change the discourse state so that subsequent queries will take into account the intended itinerary ("When is the first flight on Saturday morning?"), or should it cause the system to plan and produce a response to clarify the user's goals ("Do you want to see all the flights?")?

Is the decision to treat the input as a command, or merely as information to the system, really part of the "NL-understanding" process, or part of the backend process that takes place after understanding, and is independent of it? The same questions can be asked about the response planner and response generator components in Fig. 2. Is it a part of the NL processing (a part that just is not used when the input comes from text instead of an interactive user) or is it part of the post-NLP system? Computational linguists do not agree on where to draw these boundaries or on how to represent the information that passes between them.

Simplifying the Problem

Not every NLP system has or needs all of the components shown in Fig. 3. Some systems have a vestigial parser and attempt to extract meaning without using much if any syntactic information. Others combine the syntactic and semantic processing into one indistinguishable process. Some applications require little if any pragmatic or discourse processing.

A few systems attempt to leave out almost all of the components shown there and bravely try to go directly from words to a reasoner (perhaps an expert system) that attempts to produce a meaningful response without a detailed linguistic analysis at any level. This is no more extreme than eliminating the reasoner and response generator in applications where there are only a few allowable outputs.

Inside the NLP box, not every system has or needs all the pieces described above. In short, all NL systems work by simplifying some aspects of the problem they are trying to solve. The problem can be simplified on the input side (e.g., just typed questions to a database system) or on the output side (e.g., extract from multiple paragraphs of newspaper text just three pieces of information about company mergers; extract from a single spoken utterance one of six possible commands to an underlying display system). These problem simplifications result in the simplification or elimination of one or more of the components shown above.

Progress in developing NLP systems will likely depend on training and evaluation (as has been the case with speech processing), but the multiplicity of components, each with its own input/output behavior that is not commonly agreed upon has made progress very difficult.

Another View

Another way to look at the NLP problem, instead of boxes in sequential order, is as a series of independent processes, each of which uses particular kinds of knowledge bases and each of which contributes to an overall understanding of the input. This architecture is illustrated in Fig. 4.

In this view the lexical processor would use a dictionary to help it transform the input words into a structure with more meaning; the syntactic processor would use a grammar of the language; the semantic processor would use semantic interpretation rules and a domain model of concepts and relationships that defines the domain the system can understand; and discourse and pragmatics might use a task model that specifies the user's goals and the portions of those goals that have been achieved by previous inputs.

All of these knowledge sources are available to a process called the "understanding search," rather like the "recognition search" of speech recognition. It produces one or more outputs, such as an ordered list of possible meanings, perhaps with probabilities attached. See Marcus (15) and Moore (16) in this volume for more detailed descriptions of related work.

One advantage of this view of the problem is that it permits a new and very important component to be added: a learning algorithm that populates the knowledge sources by an automatic (or semiautomatic) process and an appropriately annotated corpus, as shown in Fig. 5. The use of a single common understanding search process provides the framework for using all of the knowledge sources in ways that are similar enough for the results to be combined; in the old pipelined architecture (Fig. 2), it would be much harder to have a uniform way of expressing the results of each component and thus much harder to develop a learning component for each of the knowledge sources.

The cooperating process view of language understanding holds the promise that the knowledge sources needed for NLP in a new domain can be created automatically. If this is true, it should become much easier to make an NLP system for a new domain by starting from a suitably annotated corpus and populating the knowledge sources so that an understanding search process could take place.

NLP systems based on this model are currently being developed, but it is too soon to assess their overall success. There are a number of methods for learning the parameters necessary to predict the part of speech of an unknown word in text with only a 2 to 3% error rate. No systems yet create syntactic rules completely automatically, but some do use syntactic probabilities that have been trained on corpora. Semantics and domain model extraction are harder yet, but enough effort is being expended in this direction that it seems justified to expect considerable progress in these areas in the near future.

It is also possible to add probabilities to syntactic and semantic rules so that the most probable parse or the most likely interpretation is produced first. The basic operation of one system (17) is to determine the most probable set of

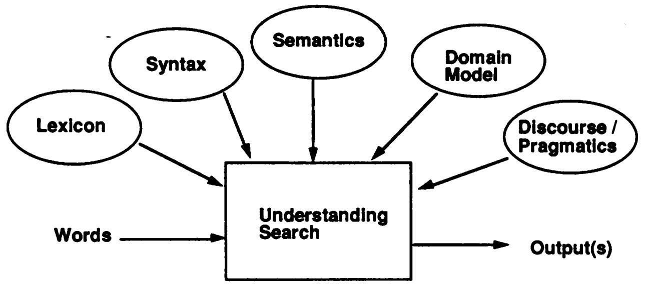

FIG. 4. A cooperating process view of a generic NLP system.

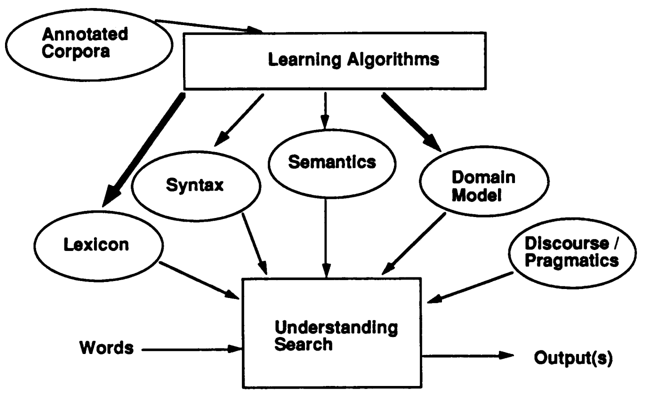

FIG. 5. Learning from an annotated corpus.

concepts (word senses) and the semantic links between those concepts, given a set of local linguistic structures (called grammatical relations) and the a priori likelihood of semantic links between concepts. This domain-independent statistical search strategy works as effectively on fragments as on grammatical sentences, producing a meaning expression output for almost any input. Learning algorithms will follow. Currently, no attempt is being made to use these statistical techniques for discourse or pragmatic processing.

One factor that limits progress in using this model is that the cost of producing an appropriately annotated corpus for NL processing is significantly higher than the cost of producing an annotated corpus for speech processing. In the case of speech, annotation consists basically of transcription, which can be performed by almost anyone who knows the language being spoken. But for NL, annotation is mostly done by experts. Some lexical and syntactic annotations can be done by carefully trained people (who are not experts).

The semantic annotation that was performed for the Air Travel Information System (ATIS) data in the Advanced Research Project Agency's (ARPA) Spoken Language Systems program (18) is an example of a corpus that was expensive to collect and annotate, and it is not even useful for some types of research (e.g., pragmatics). Efforts are currently under way in the computational linguistics community to develop some kinds of annotation that are both useful and inexpensive.

HOW CAN NL SYSTEMS BE APPLIED AND EVALUATED?

In speech there are a few factors, such as vocabulary size, perplexity, and training data, that roughly determine the performance of an SR system for a particular domain. In NLP there are a large number of such factors, not all easily quantifiable: vocabulary size, number of concepts in the domain, number of relations between those concepts, amount of training data, type of annotations available, task complexity, amount of ambiguity, amount of ungrammaticality, errors in input, etc.

We need a metric similar to perplexity but one that takes into account the number of possible concepts and relations and the amount of overlap (potential ambiguity) among them. Ideally, such a metric would take into account domain and task complexity as well as syntactic and semantic complexity.

Even without a way to measure the difficulty of the input, there are a variety of ways to evaluate the performance of NL systems. Evaluation is a necessary part of any system, whether developed for research or for application. In the past few years, several methodologies have emerged for evaluating particular kinds of NL systems—spoken language systems (SLS) and message-processing systems foremost among them. The notion of domain-independent understanding and evaluation, while being actively explored by the research community, is only one of several approaches to evaluation. The methodologies that have actually been used thus far vary from one type of application (message processing, question answering, translation, information retrieval, etc.) to another.

The methodologies allow comparative evaluation of different NL systems, as well as tracking of the progress of a single NL system as it evolves over time. For example, the methodology for the SLS program can be used for both spoken language systems (with speech input) or just NL systems (by omitting the SR component and giving the NL system a word string as input). The SLS methodology works by comparing the predetermined "right answer" (the canonical answer) to answers that are produced when different SLS systems are given identical inputs and are required to use identical data bases (19). The "understanding error rate" is the percentage of utterances that are either answered incorrectly or not answered at all.

One of the current ways to evaluate NL systems (a way that has been very successful in several recent ARPA-sponsored evaluations of both text and spoken language) is to look at the output produced by a system with an NL component and try to determine whether the output is correct or incorrect. (A better metric might be appropriate or inappropriate; that is a topic of considerable discussion.) Either way, the complexity and variety of the types of output that can be produced by a system make evaluation based on output extremely difficult.

These evaluations are labor intensive and somewhat difficult to specify and carry out, but they are very important to the community of people doing research in this area. Current NL systems (components of SLS systems, operating in the ATIS domain, with vocabularies of around 2000 to 3000 words) achieve an understanding error rate of about 6%, which appears to be quite close to the threshold of real utility for applications. Detailed descriptions of the methodology, as well as the underlying databases and annotated corpora for the ATIS domain (as well as many other NL corpora), are available from the Linguistic Data Consortium at the University of Pennsylvania.

CONCLUSIONS

What is the current state of the art in NL processing? In question-answering domains such as database interfaces, the understanding error rate is about 5 to 10%. The amount of effort needed to bring an NL system to this level of performance is still more substantial than we would like. In fact, it is currently the major bottleneck to the availability of NLP applications.

Portability can be defined as the ability to make an NL system (for a particular type of application, such as a database) usable in a new domain (with a new vocabulary, a new set of semantic concepts and relations). A system could be considered portable if it were possible to achieve moderate performance (perhaps 15 to 20% error) in a new domain using some automatic methods and a few person-weeks of human effort. The goal of portability is good performance with moderate effort.

The portability problem will probably be cracked by work that is being done in several areas simultaneously. Automatic learning (training) based on annotated corpora holds substantial promise. In addition, NL systems need to be adaptable to their users (i.e., a user should be able to tell a system when it understood something incorrectly and what the right interpretation is).

Other challenges in addition to portability include scaling up from demonstration to real applications; increasing robustness (how systems deal with unexpected novel input); feedback (what kind of help the system can give a user when an interpretation goes wrong); and determining what is an acceptable level of performance for a new NLP system.

As has been the case in speech processing, the biggest payoff comes when machines can perform at or near the level of human performance. NL systems still have a long, long way to go, but the goal (for limited domains) will soon be within our grasp. The result will be a paradigm shift that will enable designers and developers of many types of systems (but particularly interactive systems) to incorporate NLP into their systems. Users will begin to expect their systems to understand spoken or typed commands and queries, to be able to classify bodies of text, and to extract various kinds of information from bodies of text.

1. Makhoul, J. & Schwartz,R. (1995) *Proc. Natl. Acad. Sci. USA* **92,** 9956–9963.
2. Jelinek, F. (1995) *Proc. Natl. Acad. Sci. USA* **92,** 9964–9969.
3. Levinson, S. E. (1995) *Proc. Natl. Acad. Sci. USA* **92,** 9953–9955.
4. Oberteuffer, J. A. (1995) *Proc. Natl. Acad. Sci. USA* **92,** 10007–10010.
5. Weinstein, C. J. (1995) *Proc. Natl. Acad. Sci. USA* **92,** 10011–10016.
6. Wilpon, J. G. (1995) *Proc. Natl. Acad. Sci. USA* **92,** 9991–9998.
7. Allen, J. (1987) *Natural Language Understanding* (Benjamin/Cummings, Menlo Park, CA).
8. Gazdar, G. & Mellish, C. (1989) *Natural Language Processing in LISP* (Addison–Wesley, Reading, MA).
9. Smith, G. W. (1991) *Computers and Human Language* (Oxford Univ. Press, Oxford).
10. Weischedel, R. M., Carbonell, J., Grosz, B., Marcus, M., Perrault, R. & Wilensky, R. (1990) *Annu. Rev. Comput. Sci.* Vol. **4**.
11. Winograd, T. (1983) *Language as a Cognitive Process* (Addison–Wesley, Reading, MA).
12. Bates, M., ed. (1993) in *Proceedings of the ARPA Workshop on Human Language Technology* (Princeton).
13. Bates, M. & Weischedel, R. M., eds. (1993) *Challenges in Natural Language Processing* (Cambridge Univ. Press, Cambridge, U.K.).
14. Montague, R. (1970) *Synthese* **22,** 68–94.
15. Marcus, M. (1995) *Proc. Natl. Acad. Sci. USA* **92,** 10052–10059.
16. Moore, R. C. (1995) *Proc. Natl. Acad. Sci. USA* **92,** 9983–9988.
17. Bobrow, R., Ingria, R. & Stallard, D. (1992) in *DARPA Speech and Natural Language Workshop* (Harriman, NY).
18. Hirschman, L., *et al.* (1993) in *Proceedings of the ARPA Workshop on Human Language Technology* (Princeton).
19. Pallett, D., *et al.* (1993) in *Proceedings of the ARPA Workshop on Human Language Technology* (Princeton).