Some Applications of Natural Language Processing to the Field of Augmentative and Alternative Communication[1]

Kathleen F. McCoy and Patrick Demasco
Computer and Information Sciences Department and
Applied Science and Engineering Laboratories
University of Delaware/A.I.duPont Institute
Newark, DE 19716
mccoy@cis.udel.edu, demasco@asel.udel.edu

Introduction

Augmentative and Alternative Communication (AAC) is the field of study concerned with providing devices or techniques to augment the communicative ability of a person whose disability makes it difficult to speak in an understandable fashion. A variety of AAC devices and techniques exist today. Some are non-electronic word boards containing words and phrases in standard orthography and/or iconic representations. A user of such a non-electronic system points to locations on the board and depends on the listener to appropriately interpret the selection. Electronic word boards may use the same sorts of selectable items, but may also include speech synthesis. These presumably provide more independence for the user who does not need to rely on a partner to interpret the selections. However, these systems may place more burden on the user who must be aware of the actual strings associated with each selection and must ensure that the synthesized string be an appropriate English sentence. Since the system will only "speak" what has been selected, generally more selections are required per sentence and speed of selection becomes more crucial. Some common features used to improve access time include abbreviation expansion (where the user of the system memorizes a set of unique abbreviations for some set of words/phrases) and letter/word prediction (where the system attempts to predict the next word of input based on the first few letters; typically these predictions are displayed on the screen and may be accessed very easily). While the techniques that have been made possible with the advent of personal computers have certainly provided great benefits, in this paper we explore how the use of natural language processing techniques can be used to improve performance.

This work targets a population of AAC users who do not have language impairments. One can imagine that such a user would like their device to output well-formed sentences (in fact, naturally "think" in terms of such sentences). However, because of the nature of their physical impairment and the time that it takes to compose well-formed sentences, users of current devices must often settle for output that is not as desirable. In this work we take advantage of the regularities in natural language production to allow the system to act in an intelligent fashion. This intelligent fashion makes it easier for the user to cause the device to output well-formed text, but saves the user time and/or effort in entering that text.

We first discuss some limited background in the area of natural language processing, and then show how it can be applied to three application areas in augmentative communication: COMPAN-

1.This work has been supported by a Rehabilitation Engineering Research Center Grant from the National Institute on Disability and Rehabilitation Research (#H133E30010). Additional support has been provided by the Nemours Foundation. We thank Mark Jones, Christopher Pennington, Peter Vanderheyden, and Julie VanDyke for their work on various projects reported here.

SION (Demasco & McCoy, 1992; McCoy et al., 1994), intelligent word prediction, and intelligent abbreviation expansion. COMPANSION is a technique that uses primarily semantic (word meaning) information to expand a user's telegraphic input into a well-formed sentence. Intelligent word prediction takes the syntactic (and semantic) context into account in making predictions based on the first few letters of input. Intelligent abbreviation expansion allows the user to make up abbreviations "on the fly" and produces expansions appropriate to the linguistic context. Thus this technique is designed to relieve some of the cognitive load associated with memorizing a large set of unique abbreviations for each word. Each application area will be explained in light of how natural language processing is providing benefit to AAC.

Computer-Based Augmentative and Alternative Communication

A typical computer-based AAC system can be thought of a providing the user with a "virtual keyboard" which enables the user to select items to be output to a speech synthesizer or some other application. A virtual keyboard consists of two components: (1) a physical interface which is the physical method for activating the keyboard, and (2) a language set which contains the elements that may be selected. Both the physical interface and the language set must be tailored to an individual depending on his/her physical and cognitive circumstances and the task they are intending to perform.

For example, for people with severe physical limitations, access to the device might be limited to a single switch. A physical interface that might be appropriate in this case involves row-column scanning of the language set which is arranged (perhaps in a hierarchical fashion) as a matrix on the display. The user would make selections by appropriately hitting the switch when a visual cursor crosses the desired items. In row-column scanning the cursor first highlights each row moving down the screen at a rate appropriate for the user. When the cursor comes to the row containing the desired item, the user hits the switch causing the cursor to advance across the selected row highlighting each item in turn. The user hits the switch again when the highlighting reaches the desired item in order to select it.

Independent of the physical interface is the linguistic set which must also be tuned to the individual. For instance, the language set might contain letters, words, phrases, icons, pictures, etc...

The use of a computer-based AAC device generally has many trade-offs. Assuming a physical interface of row-column scanning, a language set consisting of letters would give the user the most flexibility, but would cause standard message construction to be very time consuming. On the other hand, a language set consisting of words or phrases might be more desirable from the standpoint of speed, but then the size of the language set would be much larger causing the user to take longer (on average) to access an individual member. In addition, if words or phrases are used, typically the words would have to be arranged in some hierarchical fashion, and thus there would be a cognitive load involved in remembering where the individual words and phrases were stored.

In this work we have been concerned with several aspects of the messages being composed by users: time, "correctness", and cognitive load. That is, we want to develop applications that will allow messages to be composed in a minimal amount of time yet we do not want to compromise the "correctness" of the language produced. At the same time, we do not want to impose a great cognitive burden on the user. Thus we want to enable the user to compose "normal" English in a reasonable amount of time without imposing an unrealistic cognitive load on the user.

The Application of NLP

The fields of Natural Language Processing and Computational Linguistics attempt to capture regularities in natural (i.e., human) languages in an effort to enable a machine to communicate effectively with a human conversational partner (Allen, 1987; Allen, 1995; Gazdar & Mellish, 1989; Grishman, 1986). Three major areas of research in these fields (syntax, semantics, and pragmatics) deal with regularities of language at different levels. Various techniques have been developed within each which will be useful for application to various AAC technologies.

Syntax

The syntax of a language captures how the words can be put together in order to form sentences that "look correct in the language" (Allen, 1987). Syntax is intended to capture structural constraints imposed by language which are independent of meaning. For example, it is the syntax of the language that makes:

"I just spurred a couple of gurpy fliffs."

seem like a reasonable sentence, but makes

"Spurred fliff I couple a gurpy."

seem ill-formed.

Processing the syntax of a language generally involves two components: 1) a *grammar* which is a set of rules that refer to word categories (e.g., noun, verb) and various morphological endings (e.g., +S for plural, +ING) that capture the allowable syntactic strings in a language and; 2) a *parser* which is a program that, given a grammar and a string of words, determines whether the string of words adheres to the grammar. (See (Allen, 1987; Allen, 1995; Gazdar & Mellish, 1989; Winograd, 1983) for examples of various parsing formalisms and grammars.)

Using a grammar and parser an AAC system would be able to: 1) determine whether or not the utterance selected by the user was well-formed syntactically, 2) determine valid sequences of word categories that could form a well-formed sentence, 3) given a partial sentence typed by the user, determine what categories of words could follow as valid sentence completions, 4) determine appropriate morphological endings on words (e.g., that a verb following the helping-verb "have" must be in its past participle form), and 4) determine appropriate placement of function words which must be added for syntactic reasons (e.g., that certain nouns must be preceded by an article, that the actor in a passive sentence is preceded by the word "by").

Current use of Syntax in AAC. Syntactic knowledge is being successfully applied in a number of AAC projects. For example, several word prediction systems use syntactic information to limit the words predicted to those which could follow the words given so far in a syntactically valid sentence (Swiffin et al., 1987; VanDyke et al., 1992; VanDyke, 1991). To some extent, many grammar checkers available today and systems aimed toward language tutoring (e.g., (Suri & McCoy, 1993a; Suri & McCoy, 1993b; Newell et al., 1990; Wright et al., 1992)) also use syntactic information, though there is still great room for improvement.

Semantics

The area of semantics deals with the regularity of language which comes from the meanings of individual words and how the individual words in a sentence form a meaningful whole. A problem in semantics is the fact that many words in English have several meanings (e.g., "bank" may refer to the edge of a river or to a financial institution). In Computational Linguistics the use of selectional restrictions (Katz & Fodor, 1963), case frames (Fillmore, 1968; Fillmore, 1977), and preference semantics (Wilks, 1975) is based on the idea that the meanings of the words in a sentence are mutually constraining and predictive (Small & Rieger, 1982). When the words of a sentence are taken as a whole, the meanings of the individual words can become clear.

Consider the sentence "*John put money in the bank*." Here the financial institution meaning of *"bank"* can be inferred from the meaning of the verb "*put*" (which expects a thing to be put and a location to put it in) and the fact that *"money"* is the appropriate kind of object to be put in a financial institution.

Note that in order to take advantage of semantics, a natural language processing system must (1) have rules (selectional restrictions, case frames) which capture the expectations from individual words (e.g., "eat" is a verb that generally requires an animate agent and an object which can be classified as a food-item), and (2) have a knowledge base that contains concepts that are classified according to their meanings (e.g., "apples" are food-items, "John" is a person, and "people" are animate).

Current Use of Semantic Information in AAC. The AAC system that has taken the most semantic information into account is the *Compansion* system described below. Semantic information is also a main component of the PROSE (Waller et al., 1992) system developed at the University of Dundee. PROSE is intended to give the user access to prestored phrases/sentences/stories which can be accessed according to their semantic content. The basic idea is that sets of phrases, stories, sentences etc. will be input by the user (in advance) along with some semantic information about their content. PROSE will then store this information in an intelligent way according to the semantic information given. The system will then retrieve the pre-stored material, based on minimal prompting by the user, in semantically appropriate contexts.

Pragmatics

Pragmatic information refers to the broad context in which language and communication takes place (Allen, 1987; Joshi et al., 1981; Levinson, 1983). Situational context and previous exchanges produce conversational expectations about what is to come next. Natural language processing has concerned itself with developing computational mechanisms for capturing these same expectations in a computer.

Current Use of Pragmatics in AAC. The majority of the AAC work that takes advantage of pragmatic information has come from the University of Dundee. Their CHAT system (Alm et al., 1987) is a communication system that models typical conversational patterns. For example, a conversation generally has an opening consisting of some standardized greeting, a middle, and a standardized closing. The system gives users access to standard openings and closings (at appropriate times). In addition (for the middle portion of a conversation) it provides a number of "checking" or "fill" phrases (e.g., "OK", "yes") which are relatively content free but allow the user to participate more fully in the conversation.

The TALKSBACK system (Waller et al., 1990; Waller et al., 1991; Waller et al., 1992) incorporates user modeling issues. It takes as input some parameters of the situation (e.g., the conversational partners, topics, social situation) and predicts (pre-stored) utterances the user is likely to want based on the input parameters. For example, if the user indicates a desire to ask a question about school to a particular classmate, the system might suggest a question such as "What did you think of the geography lesson yesterday?". In other words, the system attempts to use the parameters input by the user to select utterances that are pragmatically appropriate.

In the remainder of this paper we discuss three AAC applications under development at the Applied Science and Engineering Laboratories of the University of Delaware and the A.I. duPont Institute that rely on NLP.

COMPANSION

The Compansion system (Demasco & McCoy, 1992; McCoy et al., 1994) is designed to work with a word-based language set system. Compansion allows the user to select a telegraphic input consisting of uninflected content words. This input is expanded by the system into a syntactically (and semantically) well-formed sentence.[1] Thus Compansion allows the user to "speak" grammatically well-formed sentences but greatly reduces the number of keystrokes necessary to select those sentences. This is done without increasing the cognitive load required for selection.

The Compansion system uses both syntactic and semantic knowledge in its processing. The system consists of three major phases:

(1) A "word order" parser relies on a syntactic grammar which captures the regularity of the expected telegraphic input. This parser is responsible for grouping words into sentence-sized chunks and indicating each word's part of speech (e.g., noun, verb). In addition, the word order parser is responsible for attaching modifiers (e.g., compound possessives, adjectives, and adverbs) to the word they are most likely modifying.

(2) A "semantic" parser reasons about the meaning of the content words (the modifiers are hidden from this processing) of each sentence-sized chunk and develops a semantic representation of the sentence. The semantic reasoning is the most sophisticated aspect of this system.

(3) A translator/generator takes the output of the semantic parser and generates an English sentence using a syntactic grammar of English.

Consider the following example handled by the system:

Input: `5 apple eat John`
Output: `5 apples were eaten by John.`[2]

The word order parser is responsible for identifying the part of speech of each input word and for passing main sentence components off to the semantic parser for processing. It first identifies `5` as an adjective which is modifying the noun to its right. Next it identifies `eat` as the main verb and `John` as its noun object. The word order parser "packages" the input (hiding the modifiers from the next phase of processing) and hands off the noun-verb-noun "sentence" to the semantic parser.

The semantic parser is responsible for determining how the given words could fit into a well-formed semantic structure. The semantic parser determines that John (because he is human) is doing the eating, and that the apples (an edible item) are actually the things being eaten.

This semantic analysis (along with the original input string) is then passed to the translator/generator[3] which attempts to generate a syntactically well-formed sentence that captures the intended meaning. It is in this phase that plural endings, function words, etc. are added to the orig-

1. The system may come up with more than one possible sentence for a given input. The desired possibility could then be selected with an additional keystroke (for example).
2. Note each of the underlined words represent a savings of at least one word selection to the user.

inal telegraphic input.

The Word Order Parser

The word order parser uses a lexicon that indicates part of speech information for each word and a grammar that captures the syntactic regularity of telegraphic speech. These are used to identify the syntactic category of each input word (e.g., noun, verb) and to attach modifiers (e.g., adjectives and adverbs) to the word that they modify. For instance consider the following:

Simple Sentences - most sentences follow a noun-verb-noun pattern which can be used to disambiguate words (e.g., watch) that can play either a noun or a verb role. Different verbs may require different structures. E.g., *`give`* or *`put`* would expect two nouns following them, while intransitive verbs such as *`die`* would not require any.

Embedded Sentences - some verbs (e.g., *`think`*, *`believe`*) may take sentential complements. The word order parser identifies the complement and sends it off to the semantic parser as a single unit.

Adjectives - in English adjectives (in noun phrases) occur to the left of the word that they modify. Thus, for example, in the input *`mary put book big table`*, the parser would associate the adjective *`big`* with *`table`* (and not with *`book`*).

Adverbs - may occur either to the left or right of the verb they modify and thus either attachment may be possible. Adverbs are most difficult when the sentence contains several verbs.

Compound Nouns - are two or more nouns which occur next to each other in the input string. The user might intend that these nouns (1) be joined by a conjunction (as in `John Mary like dance`), (2) represent a possessive (as in `John dog run away`), (3) are "unrelated" and serve as the fillers of different cases in the semantics (as in `John give Mary book`). Case (3) is handled by a sentence pattern associated with verbs requiring multiple objects. The first two cases require some semantic reasoning to distinguish with any accuracy. Reasoning includes rules such as "like" things are conjunctions (e.g., `john mary` would be assumed a conjunction), and animates may possess inanimate things (e.g., `john hat` would be assumed to be `john's hat`).

Semantic Parser

The semantic parser (McCoy et al., 1990a; McCoy et al., 1990b) takes the noun and verb words identified by the word order parser and attempts to fit these items into a well-formed semantic structure. Because our system has little syntactic information, its semantic processing is more extensive than that found in most standard NLP systems.

Semantic Representation. The output of the parser is a "logical form" meaning of the intended sentence. The representation itself is modeled after that described in (Fillmore, 1968; Fillmore, 1977). In our representation the verb is central to the meaning of the sentence and the nouns are said to play a "role" or to "fill a case" with respect to the verb. The cases available are small in number, and typically adhere to some semantic restrictions. Each case may occur at most once in the representation for a given sentence. For instance, we use the following set: AGEXP (AGent/EXPeriencer) is the object doing the action. THEME is the object being acted upon, while INSTR is the object or tool etc. used in performing the action of the verb. GOAL can

---

3. The sentence generator used by the system was written by Michael Elhadad at Columbia University and is based on the functional unification paradigm (Elhadad, 1991).

be thought of as a receiver, which is not to be confused with BENEF, the beneficiary of the action. For example, in “John gave a book to Mary for Jane”, “Mary” is the GOAL while “Jane” is the BENEF. We also have a LOC case which captures the location in which the situation is taking place (this case may be further decomposed into TO-LOC, FROM-LOC, and AT-LOC), and TIME which captures time information (this case may also be further decomposed).

Knowledge About Nouns. In order to determine which noun is playing which role, the system must have access to the possible meanings of each noun it may encounter. To capture this, our nouns are arranged in a hierarchical meaning representation. Thus, for example, `hammer` is represented as a `T-Box` (something typically found in a tool-box) which is-a `tool` which is-a `physical-object`. `John` is represented as a `human` which is an `animate-object`. The granularity of this representation determines how precise the system reasoning can be.

Knowledge About Verbs. The system must also have information about expected roles and fillers associated with each verb it may encounter. Most of this information is hierarchically arranged in our Verb Hierarchy and is represented as a set of preferences concerning the semantic cases expected by the verb and the type of nouns that can fill these cases. Our system uses three kinds of preferences: Case Filler Preferences, Case Importance Preferences, and Higher-Order Case Preferences.[1]

In addition to preferences, our system also employs a set of idiosyncratic case constraints. These constraints are orgthogonal to the hierarchical organization of verbs and are thus attached to individual words. The constraints dictate mandatory and forbidden cases for individual verbs.

Both the preferences and constraints used by the system are discussed below. Taken together, they are used to rule out certain semantic interpretations of the input and to rate acceptable interpretations against each other so that the system can select the interpretation most likely intended.

Case Filler Preferences. The case filler preferences are most similar to what has previously been used in semantic reasoning in NLP systems. These preferences indicate preferred fillers of a particular verb case. Motivated by Preference Semantics (Wilks, 1975), the preferences indicate the kinds of objects that could fill a particular role along with an indication of how desirable each possible filler type is. For example, the preference for filling the BENEFiciary case for most verbs is: ((human 3) (organization 2) (animate 2)). This basically should be read as, given the choice, a human should fill this role, but an organization or an animate object are also reasonable. No other types of objects may fill the BENEF case.

Case Importance Preferences. While the above preference tells us what kind of object may fill a particular case role, it does not tell us anything about what *cases* are more important to fill for a particular verb. This is captured by the Case Importance Preferences.

For example, with material verbs such as `hit`, it seems much more likely that the role of THEME (the item being hit) will be filled than that of BENEF (the person/thing for whom the hitting is being done). To represent this, a higher value (3) is given as the preference of filling the THEME case, while a lower value (1) is given as the preference for filling the BENEF case.

Having this preference specified allows the system to handle input such as: `[John hit Mary]` which most human readers would take to mean that `John` is the agent of a hitting action

1. Note that only the first of these preferences is normally captured in most Natural Language Processing systems. The other two will be motivated below.

where `Mary` is the Theme or object being hit. The problem is that if the system only takes the case *filler* preferences into account, then this reading cannot be preferred over the reading of `Mary` playing the BENEFiciary case (as in “John hit for Mary"). This is because, in our system, `Mary` (being a physical object) would be given a “3” rating for playing the THEME case. `Mary` (being a human) would also be given a “3” rating for playing the BENEF case. Thus neither reading would be preferred over the other. The case *importance* preferences allow the “John hit Mary” reading to be preferred by indicating that it is more important to fill the THEME case of `hit` than it is to fill the BENEF case.

Higher-Order Case Preferences. While the above heuristics have proven to be quite useful, there are situations in which they fall short in that the scope of the heuristic is limited to a single case. However, the heuristic value of an interpretation can be affected by a combination of case bindings. Higher-order case preferences heuristics are intended to account for interactions between various cases and their fillers. These preferences must be applied to a full interpretation of the input words.

To see how multiple cases can interact to affect an interpretation, consider the following: if a non-human animate (e.g., dog) is the AGENT of a material process (e.g., `eat`), it is quite unlikely that an INSTRrument is being used (e.g., dogs do not typically eat with spoons). Note that it is perfectly reasonable to have either role instantiated by such fillers *in isolation* (e.g., dogs can eat and people can eat with spoons). In the semantic parser the interaction of these fillers of cases is captured by a rule which subtracts from the overall goodness of an interpretation when these conditions are found. The idea is that by subtracting such a value, a more reasonable interpretation will rise to the top.

Idiosyncratic Case Constraints. The idiosyncratic case constraints are not used to heuristically compare interpretations, but are used to constrain what interpretations are considered. They differ from the semantic case preferences in that they are more definite, and in that they are placed directly on the verbs themselves and are not inherited down the verb hierarchy.

These constraints are intended to capture the notion that some cases on a verb are mandatory and others forbidden. These are idiosyncratic in nature because they do not seem to have any relation to the general semantics of the verb hierarchy. Two semantically similar types may have different idiosyncratic features. For example, the parser encodes the words *eat* and *swallow* as semantically equivalent. However, *swallow* cannot typically have an instrument, while *eat* may.

The mandatory and forbidden features seem to follow the syntactic classification of verbs as transitive, intransitive, and bitransitive (although finer grained distinctions are made in our system). For example, a mandatory THEME feature on the verb *hit* (a transitive verb) requires that the THEME be filled. On the other hand, *hit* cannot accommodate a GOAL. This is captured with a forbidden GOAL feature. Typically intransitive verbs forbid the filling of the THEME case: *die* cannot have a theme. Words other than bitransitives typically forbid filling the GOAL case: *give* can have a GOAL, but *hit* cannot. The most common situation is that of the verb neither forbidding nor requiring a particular case.

Semantic Parser Logic. Essentially, the semantic parser begins with an empty frame where all of the roles are represented, but unfilled. Based on information associated with the main verb (identified in the word-order parser phase of the system) the parser removes roles from the empty frame that are forbidden for the verb. In addition, the semantic case preferences for the main verb are added to the resulting frame. The parser next considers all of the objects in the input string. Using the information from the case filler preferences, it recognizes what objects can play what roles with respect to the verb.

Next the semantic parser uses the case filler preference information to calculate all "legally" filled case frames (i.e., all case frames in which each word of input is used to fill at most one case, a word filling a given case is appropriate according to the case filler preferences, no case has more than one filler, and the mandatory and forbidden constraints are adhered to). A heuristic value is calculated for each filled frame based on the semantic preferences associated with the verb.The interpretation with the highest heuristic value is considered the best guess of what the user intended.

Processing Example. Consider the input: `[apple eat John.` The word-order parser would pass these words as a package into the semantic parser along with an indication that `eat` is the main verb and the other words in the sentence are nouns. From the information associated with the verb `eat`, we know the GOAL case is forbidden. Also we know that the THEME and AGEXP case are mandatory. Next, the parser locates `eat` in the verb hierarchy to retrieve the associated preferences.

The case importance and the case filler information from the verb `eat` are listed below. The first line indicates that 4 points are awarded if the AGEXP case is filled (a case importance preference), and the rest of that entry represents the case filler preference information. 3 points are awarded if AGEXP is filled by a communicator (either a human or an organization), 2 points are awarded for another animate object (e.g., a dog), and 2 points are awarded if this case is filled by an ergative object (e.g., a truck). The other cases are specified in the same way.

AGEXP 4--to fill	3 Communicator
	2 Animate
	2 Ergative-Object
THEME 3--to fill	4 Food
	3 Solid
	2 Ingestible
	1 Physical
INSTR 3--to fill	4 FoodT (spoon)
	3 T-Box (hammer)
	2 Tool (rock)
	1 Solid
BENEF 1--to fill	3 Human
	2 Organization
	2 Animate
LOC 1--to fill	3 Place (Boston)
TIME 1--to fill	4 Timed (yesterday)

At this point we have accumulated top-down information about the case frame interpretation by considering the verb. Next, in a bottom-up fashion different interpretations will be considered by attempting to fit the nouns into the case frame in every possible way. Below shows what point values will be awarded when each object is assigned each of its possible roles. For instance, *John* would be given a value of 3 for filling the AGEXP role, 1 for THEME etc.

Apple	THEME 4	INSTR 1	GOAL <Forbidden>	
John	AGEXP 3	THEME 1	BENEF 3	GOAL <Forbidden>

With this information, the parser fits the objects into the case frame in every possible way abiding by the mandatory and forbidden cases and allowing only one object to fill any given role.[1] Under these constraints, we generate only one possibility which corresponds to the sentence "The apple is eaten by John". The corresponding semantic representation along with its heuristic rating is shown below:

(25 DECL
(VERB (LEX EAT))
(AGEXP (LEX JOHN))
(THEME (LEX APPLE)))

In cases where more than one possible semantic representation is computed by the system, the highest rated interpretation would be passed to the translator/generator which would output its natural language form. The user may either select it or request the next highest interpretation etc.

Some Companion System Examples:

Passive constructions are chosen when needed by semantics. The system attempts to generate sentences that maintain the word order given by the user.

Many apple eat John
=> Many apples are eaten by John. AND Many of the apples are eaten by John.

The intended semantics may be ambiguous. In such situations the user may be given several options:

John believe Mary
=> John believes Mary. AND John is believed by Mary,

Notice that a question mark will cause a question to be generated by the system which will add appropriate question words for a standard Yes/No question:

Mary want John go store? => Does Mary want John to go to the store?

Appropriate case marking prepositions are added to form well-formed sentences when needed.

John give ball Mary => John gives the ball to Mary.

John give Mary ball => John gives Mary the ball.

John break window hammer => John breaks the window with the hammer.

John go Newark => John goes to Newark.

Default verb and AGEXP together may greatly speed communication rate:

very tired=> I am very tired.

In the context of a question, the default AGEXP is "you":

tired?=> Are you tired?

Verbs that take various kinds of sentential complements are handled:

John contemplate buy new car => John contemplates buying the new car.

John teacher know he not want learn write?
=> Does John's teacher know that he doesn't want to learn to write.

---

1. Recall that conjunctions are handled in the word-order parser and are hidden from the semantic parser processing.

Appropriate tense may be chosen from a “time” word indicated in the sentence. Tense may also be selected using special keys. If no time information is included, a default (present) tense is chosen for the first sentence. Subsequent sentences maintain the tense used in the previous sentence.

John go store yesterday => John went to the store yesterday.

Intelligent Word Prediction

Word prediction is a technique that is used with letter-based AAC systems (i.e., with systems whose language set consists of letters). Most current approaches to word prediction rely on statistical data to determine what words to present based on the first (few) letters of input (Newell et al., 1992). This statistical data includes information such as frequency (what are the most frequent words that begin with the typed prefix) or recency (what are the most recently selected words that begin with the typed prefix) or some combination of the two. A typical prediction system might act in the following way:

```
Given Input: I went to the t
Presented predictions: the,to,that
```

Notice that the words predicted are very frequently occurring, however, none of them make much sense in syntactic context. In particular, it is not the case that a “`the`” could follow a “`the`” in a standard English sentence. Therefore these predictions are likely to not be very useful to the user. Rather than just relying on the statistical data, our intelligent word prediction system uses information from syntax to restrict the predictions to those which are appropriate to the syntactic context. Our long term goal is to apply semantic and pragmatic information as well.

Consider how the use of syntactic information might cause the predicted items to be of the appropriate syntactic category:

```
Given Input: I went to the t
Presented predictions: table, town, thing
```

The syntactic information may also be useful in selecting word with appropriate morphological endings. For example, a verb following a help verb BE must be in its progressive form. If verbs are predicted in that context, the presented verbs would comply to the syntactic rule:

```
Given Input: I am t
Presented predictions: talking,thinking,telling,tired
```

Syntactic Information in Previous Word Prediction Systems

The first system to take syntactic information into account in word prediction was Syntax-PAL developed at the University of Dundee (Swiffin et al., 1987). This system did not take full syntactic information into account, bur rather used a transition probability on the categories of word pairs. For instance, the following transition table might be used to describe the probability of a noun being followed by various other word categories:

CAT(word-1)	CAT(word)	PROB
Noun	Verb	60%
Noun	Prep	25%
Noun	Noun	7%
Noun	Adj	3%

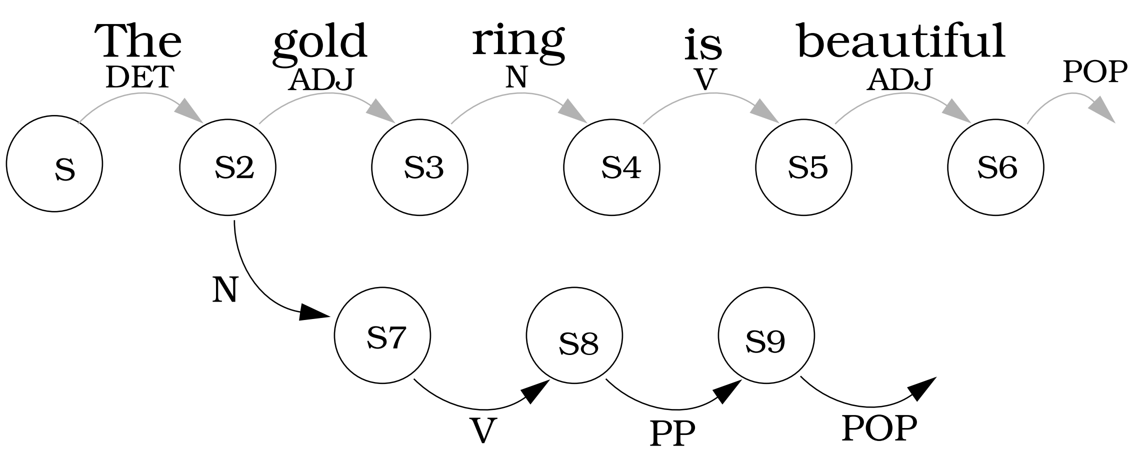
Syntax-Pal would score the various words predicted according to the following formula:

```
Score=Freq(word)*Prob(Cat(word-1),Cat(word)))
```

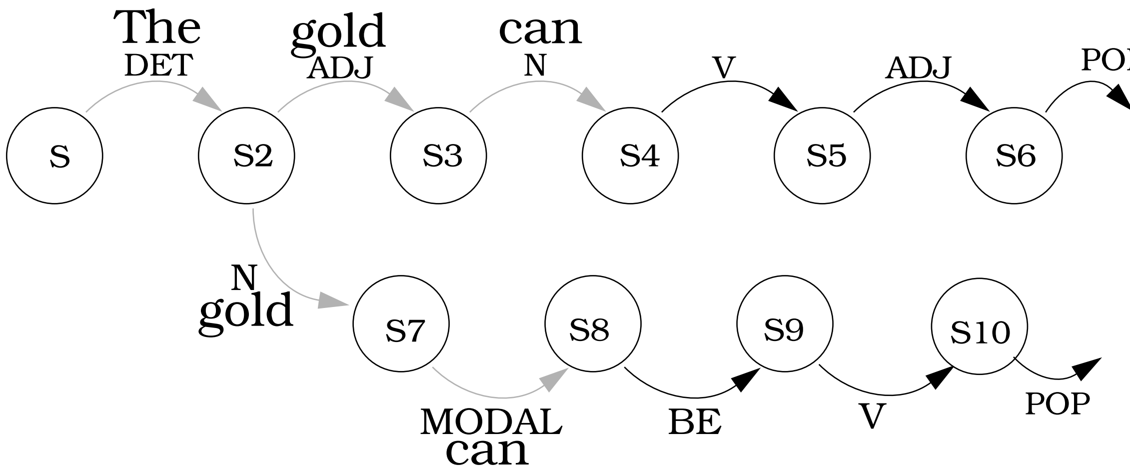
While this was a step in the right direction, it was lacking. For instance, the transition probabilities are suspect for several reasons. First, they require that each word be assigned to exactly one category. But many words may be assigned to multiple categories, and the various possible categories of a word may dramatically affect the predictions. Second, it did not take the whole sentence context into account. For example, the Noun-Verb probability is quite high in Pal, however, after the main verb of the sentence has been seen, the probability of a Verb following a Noun must be dramatically lower. Third, (related to the second point), the sentence type might also affect the probabilities. For instance, in declarative sentences the probability of a Verb following a Verb is probably quite low, however, this is more common in some wh-questions such as `Who do you` <u>`think saw`</u> `Bill?`

Using a Full Grammar of English to Follow all Parses in Parallel

Our solution to the problem has been to create a full-blown grammar of English in a formalism that allows several parses to be pursued in parallel (VanDyke et al., 1992; VanDyke, 1991). This was done by implementing a grammar in a transition network formalism. In such a formalism, parsing a sentence means "walking" the network taking arcs which correspond to the category of a word that is encountered. Consider the following network and how it would be "walked" in parsing the sentence: `The gold ring is beautiful.`



This standard formalism (which pursues one parse at a time) can be augmented to follow all parses at once: `The gold can`



Consider a partial sentence such as “the gold”. Since “gold” can either be an adjective or a noun, both of these parses would be followed by the predictor. This would allow the predictor to predict both Noun and Adjectives following “gold” accounting for both of the following sentences: (1) `The gold can is full.` and (2) `The gold can be moved tomorrow.`[1]

Our current work on this project includes adding probabilities to the arcs to indicate which is more likely (and thus to order predictions along the most likely arcs before those on less likely arcs).

Future Directions: Additional Linguistic Knowledge

The current implementation of intelligent word prediction relies on syntactic information only. However, our long term goal includes adding other types if linguistic information into the process. For example, if one adds semantic information such as that associated with the verb cases described in the semantic parser for the Compansion system, the word predictor might restrict its predictions to those of the appropriate semantic type. For instance, if there is a syntactic indication in the sentence that the “location” case is being typed, the word predictor can restrict its predictions to being locations:

```
Given Input: I went to the t
Presented predictions: table,town,train
```

Semantic information might also restrict the types of nouns appropriate for a given adjective. For example:

```
Given Input: The fast t
Presented predictions: train,tiger,turtle
```

In addition to semantic information, pragmatic information might also come into play in this process. For instance, knowing where the conversation was taking place might affect the predictions:

```
London Context:
```

1. Notice, as is shown in the figure, the parser could not differentiate between the N and ADJ reading of “gold” until quite late in the sentence.

```
        Given Input: I went to the t
        Presented predictions: train,theatre,tower
University of Delaware Context
        Given Input: I went to the t
        Presented predictions: talk,test,teacher
```

The conversational partner might also affect the predictions

```
Conversational Partner: teacher
        Given Input: Could you give me a b
        Presented Predictions: book,binder,break
Conversational Partner: bartender
        Given Input: Could you give me a b
        Presented Predictions: beer,bourbon,brandy
```

Intelligent Abbreviation Expansion

Abbreviation Expansion is another common AAC technique used with letter-based AAC systems. In such a system the user must define a set of abbreviations in advance and memorize the abbreviations associated with each word in the abbreviation set. While entering text, if a user types an abbreviation, the system simply looks the abbreviation expansion up in a table and the expansion replaces the abbreviation entered by the user.

This technique has proven to be very useful, but requires an incredible cognitive load on the user. Our notion is to use “flexible” abbreviation expansion (Stum, 1991; Stum et al., 1991; Stum & Demasco, 1992). Our system allows the user to type abbreviations on the fly -- as they think of them while typing text. The assumption is that a user will employ a rather small set of rules in creating an abbreviation. For example, the abbreviation might be a truncation of the desired word, or it might be the word with all vowels deleted, or some combination of these two rules. Because the number of rules for creating reasonable abbreviations is small and somewhat stable, if the user has a dictionary of all possible words to be used and the set of abbreviation rules, it can generate all possible expansions of the abbreviation employed by the user. The system must then have a mechanism for deciding which of these expansions was the one desired by the user -- but then this is very similar to the word prediction problem described previously. We propose that a scoring function be used that takes into account statistical data (e.g., word frequency) as well as syntactic, semantic, and pragmatic data. In addition, since the system uses a set of abbreviation rules to expand the abbreviation, there may also be a score associated with each rule indicating how often the user employs the rule used to create each expansion.

References

Allen, J. (1987). *Natural language understanding*. CA: Benjamin/Cummings.

Allen, J. (1995). *Natural language understanding*. Redwood City, CA: Benjamin/Cummings, second edition.

Alm, N., Newell, A., & Arnott, J. (1987). A communication aid which models conversational patterns. In R. Steele & W. Gerrey (Eds.), *Proceedings of the Tenth Annual Conference on Rehabilitation Technology* (pp. 127–129). Washington, DC: RESNA.

Demasco, P. W. & McCoy, K. F. (1992). Generating text from compressed input: An intelligent interface for people with severe motor impairments. *Communications of the ACM*, *35*(5), 68–78.

Elhadad, M. (1991). *FUF: The universal unifier user manual version 5.0*. Technical report, Columbia University, Computer Science Department.

Fillmore, C. J. (1968). The case for case. In E. Bach & R. Harms (Eds.), *Universals in Linguistic Theory* (pp. 1–90). New York: Holt, Rinehart, and Winston.

Fillmore, C. J. (1977). The case for case reopened. In P. Cole & J. M. Sadock (Eds.), *Syntax and Semantics VIII: Grammatical Relations* (pp. 59–81). New York: Academic Press.

Gazdar, G. & Mellish, C. (1989). *Natural language processing in lisp, an introduction to computational linguistics*. New York: Addison-Wesley Publishing Company.

Grishman, R. (1986). *Computational linguistics: An introduction*. Cambridge; New York: Cambridge University Press.

Joshi, A., Webber, B., & Sag, I., Eds. (1981). *Elements of discourse understanding*. Cambridge: Cambridge University Press.

Katz, J. J. & Fodor, J. A. (1963). The structure of semantic theory. *Language*, *39*, 170–210.

Levinson, S. (1983). *Pragmatics*. Cambridge: Cambridge University Press.

McCoy, K., Demasco, P., Jones, M., Pennington, C., & Rowe, C. (1990a). Applying natural language processing techniques to augmentative communication. In H. Karlgren (Ed.), *COLING90: Proceedings of the Thirteenth International Conference on Computational Linguistics* (pp. 413–415). Helsinki: Association for Computational Linguistics Universitas Helsingiensis.

McCoy, K. F., Demasco, P., Jones, M., & Pennington, C. (1990b). A domain independent semantic parser for companion. In J. J. Presperin (Ed.), *Proceedings of the Thirteenth Annual RESNA Conference* (pp. 187–188). Washington, D.C.: RESNA Press.

McCoy, K. F., Demasco, P. W., Jones, M. A., Pennington, C. A., Vanderheyden, P. B., & Zickus, W. M. (1994). A communication tool for people with disabilities: Lexical semantics for filling in the pieces. In *Proceedings of the ASSETS '94* (pp. ?). Marina del Ray, CA.

Newell, A. F., Arnott, J. L., Beattie, W., & Brophy, B. (1992). Effect of "{PAL{" word prediction system on the quality and quantity of text generation. *Augmentative and Alternative Communication*, (pp. 304–311).

Small, S. & Rieger, C. (1982). Parsing and comprehending with word experts (a theory and its realization). In W. G. Lehnert & M. H. Ringle (Eds.), *Strategies for Natural Language Processing* (pp. 89–147). Hillsdale, NJ: Lawrence Erlbaum Associates.

Stum, G. M. (1991). *Automatic abbreviation generation*. Technical Report 92-01, Department of Computer and Information Sciences, University of Delaware, Newark, DE.

Stum, G. M. & Demasco, P. (1992). Flexible abbreviation expansion. In J. J. Presperin (Ed.), *Pro-*

ceedings of the RESNA International '92 Conference (pp. 371–373). Washington, D.C.: RESNA Press.

Stum, G. M., Demasco, P. W., & McCoy, K. F. (1991). Automatic abbreviation generation. In J. J. Presperin (Ed.), *Proceedings of the Fourteenth Annual RESNA Conference* (pp. 97–99). Washington, D.C.: RESNA Press.

Suri, L. Z. & McCoy, K. F. (1993a). *Correcting discourse-level errors in a CALL system for second language learners*. Technical report 94-02, Department of Computer and Information Sciences, University of Delaware, Newark, DE.

Suri, L. Z. & McCoy, K. F. (1993b). *A methodology for developing an error taxonomy for a computer assisted language learning tool for second language learners*. Technical report 93-16, Department of Computer and Information Sciences, University of Delaware, Newark, DE.

Swiffin, A. L., Arnott, J. L., & Newell, A. F. (1987). The use of syntax in a predictive communication aid for the physically impaired. In R. Steele & W. Gerrey (Eds.), *Proceedings of the Tenth Annual Conference on Rehabilitation Technology* (pp. 124–126). Washington, DC: RESNA.

VanDyke, J., McCoy, K., & Demasco, P. (1992). Using syntactic knowledge for word prediction. Presented at ISAAC-92. Abstract appears in Augmentative and Alternative Communication, 8.

VanDyke, J. A. (1991). Word prediction for disabled users: Applying natural language processing to enhance communication. Thesis for honors bachelor of arts in cognitive studies, University of Delaware, Newark, DE.

Waller, A., Alm, N., & Newell, A. (1990). Aided communication using semantically linked text modules. In J. J. Presperin (Ed.), *Proceedings of the 13th Annual RESNA Conference* (pp. 177–178). Washington, D.C.: RESNA.

Waller, A., Broumley, L., & Newell, A. (1992). Incorporating conversational narratives in an AAC device. Presented at ISAAC-92. Abstract appears in Augmentative and Alternative Communication, 8.

Waller, A., Broumley, L., Newell, A. F., & Alm, N. (1991). Predictive retrieval of conversational narratives in an augmentative communication system. In *Proceedings of 14th Annual Conference* Kansas City, MI: RESNA.

Wilks, Y. (1975). A preferential, pattern-seeking, semantics for natural language inference. *Artificial Intelligence*, *6*, 53–74.

Winograd, T. (1983). *Language as a cognitive process, volume 1: Syntax*. Reading, Ma: Addison-Wesley Publishing Company.

Wright, A., Beattie, W., Booth, L., Ricketts, W., & Arnott, J. (1992). An integrated predictive wordprocessing and spelling correction system. In J. Presperin (Ed.), *Proceedings of the 15th Annual RESNA Conference* (pp. 369–370). Washington, DC: RESNA.