Augmented and alternative NLP techniques for augmentative and alternative communication

Ann Copestake
Center for the Study of Language and Information,
Stanford University,
Ventura Hall,
Stanford, CA 94305,
USA
aac@csli.stanford.edu

Abstract

Current communication devices designed for non-speaking users are inadequate to support conversation because the speed with which a user can input information is typically very limited. We describe some practical work on word prediction, and discuss its limitations as a technique for speeding up free text entry. We then outline an alternative approach, currently under development, which combines prediction with a constrained technique for natural language generation.

1 Introduction

The work described in this paper concerns the communication needs of people who cannot speak because of motor disabilities. It is possible to build prosthetic devices for such users by linking a suitable physical interface with a speech synthesizer, so that text or other symbolic input can be converted to speech. However, while speech rates in normal conversation are around 150-200 words per minute (wpm), and skilled typists can achieve rates of 30-40 wpm[1], conditions which impair physical ability to speak usually cause more general loss of motor function and typically speech prosthesis users can only output at best 10-15 wpm using a keyboard, with much lower rates if direct letter selection is not possible. This prevents natural conversation, not simply because of the time which is taken but because the delays completely disrupt the usual processes of turn-taking. Thus the other speaker finds it hard to avoid interrupting the prosthesis user.

This problem can potentially be alleviated in two ways: by improving the design of the physical interface (keyboard, head-stick, head-pointer, eye-tracker etc) or by minimizing the input that is required for a given output. The research described here concentrates on the latter aspect, although the utility of various techniques is partially dependent on the interface.

Techniques which have been used in augmentative and alternative communication (AAC) often involve the use of alternative symbol systems for input, in particular Minspeak (Baker, 1982). However, the work reported here was prompted by the needs of an individual who has lost the ability to speak due to amyotrophic lateral sclerosis (ALS or Lou Gehrig's disease). Such users would prefer to continue using their original language, rather than to learn an alternative symbol system. Several commercial AAC systems which take text input exist, but we found that these had a variety of drawbacks for our user. In particular, most are dedicated to speech output and cannot be used to aid writing text or email. There are also limitations in compatibility with particular software or hardware, and restrictions in the physical interfaces. A system was developed at CSLI which would run on a standard laptop while still allowing the use of other software (email, Web browser etc). The initial version of this system incorporates word prediction, as described in the next section, and also a small number of fixed text utterances, accessible via dedicated keys or menus. Experience with this suggested that an approach which allowed for more flexible combination of fixed and free text might have advantages. This is outlined in §3.

2 Experiments with word prediction

Prediction techniques have been extensively used in AAC, see, for example, the review in Darragh and

[1]Some typists can copy text much faster than this, but constructing text takes more time, even with informal text such as email. Some people who spend a lot of time contributing to online forums have reported typing speeds which are considerably higher than this range, but they still cannot approach normal conversation speeds.

Witten (1992). The basic technique behind word prediction is to give the user a choice of the words (or words and phrases) which are calculated to be the most likely, based on the previous input. The choices are usually displayed as some sort of menu: if the user selects one of the items, that word is output, but if no appropriate choice is present, the user continues entering letters. Prediction ordering is based on the user's previous input, and so the system can automatically adapt to the individual user. Unknown strings are added to the database, thus allowing them to be predicted subsequently. For text input, the simplest technique is to use the initial letters of a word as context, and to predict words on the basis of their frequencies. This basic approach was implemented in the first version of our system.

In order to experiment with different algorithms, we ran simulations using 26,000 words of data logged from the daily speech of one user. The collected data was split into training and test sets (10% of total data). We used the following scoring method to measure performance:

$$(1 - \frac{\text{(keystrokes + menu selections)}}{\text{keystrokes needed without prediction}}) * 100$$

For example, choosing *table* after inputting 't' 'a' would give a score of 50%, since a space is automatically output after the word. This scoring method is an idealization but is a reasonably accurate predictor of keystroke savings for our user.

We should emphasize that a saving in keystrokes will not correspond to an equivalent reduction in time taken to construct a message, since there is a cognitive cost in searching a menu for the desired word. Prediction is only useful when text input is very slow and difficult — even someone using a headstick may not find any advantage in it. For our user, however, each movement is not only slow, but tiring and somewhat painful, so keystroke saving is a useful measurement of performance. Prediction also has the side-effect of reducing misspellings and typographic errors, which is useful because these often make the synthesized speech incomprehensible.

The graph in Figure 1 shows the keystroke savings achieved with the basic method compared with an enhanced method which takes some account of context. Results are shown with varying menu sizes. Our user actually works with a menu size of 10, but clearly much larger menu sizes than this are impractical: the graph shows sizes up to 20 because this crudely approximates results that might be achievable with a better predictor with smaller menus. The simplest way to add contextual information is to temporarily increase weights of recently seen words (recency). We found this improved prediction rates by an additional 0.9% at a menu size of 8. The rest of the improvement for the enhanced method is due to the use of part of speech bigrams. [2]

2.1 Part of speech bigrams

Given the great improvements that have been made in speech recognition by using Hidden Markov Models (HMMs), it is natural to expect that these techniques would be beneficial for word prediction. However existing text corpora do not make good models for the speech of our user, and the amount of training data which we can collect is insufficient to extract reliable word-based trigrams. 26,000 words of data represents around three months of input, and much more data would be necessary to collect useful trigrams (vocabulary size in the 26,000 word sample is over 3,000). One possible solution is to back off to broader categories, so we investigated the use of part of speech (POS) transition probabilities extracted from existing tagged corpora (Penn Treebank). However, this actually led to a degradation in performance with the corpora we tried, because they were a poor model for our data: we suspect this is because our user makes frequent use of questions, imperatives and interjections. We therefore decided to derive transition probabilities by tagging our collected data.

Somewhat surprisingly the Treebank corpora turned out to be a good model for our data with respect to the most likely POS associated with a word and we obtained about 92% tagging accuracy simply by choosing the most frequent tag on this basis. We could not improve on this with the taggers we tried, possibly because of the small size of our training sample, and the very short length of most of the utterances. However, we have not investigated this in detail because we would expect improved tagging accuracy to have only a small effect on transition probabilities and hence on prediction performance. Furthermore, not having to use a tagger makes it much simpler to implement a practical system which adapts to the user and to the type of text.

The data shown in Figure 1 are based on a set of 80 tags, since we expanded the initial tagset by adding individual tags for some frequent words. Thus we are effectively using a combination of POS and word transition probabilities. Expanding the tagset in this way improves prediction performance: for instance distinguishing between personal pronouns allows greatly improved prediction of verb agreement. The only hand-coding involved was in reviewing and

[2] A more detailed description of some of the work described in this section is in Copestake (1996).

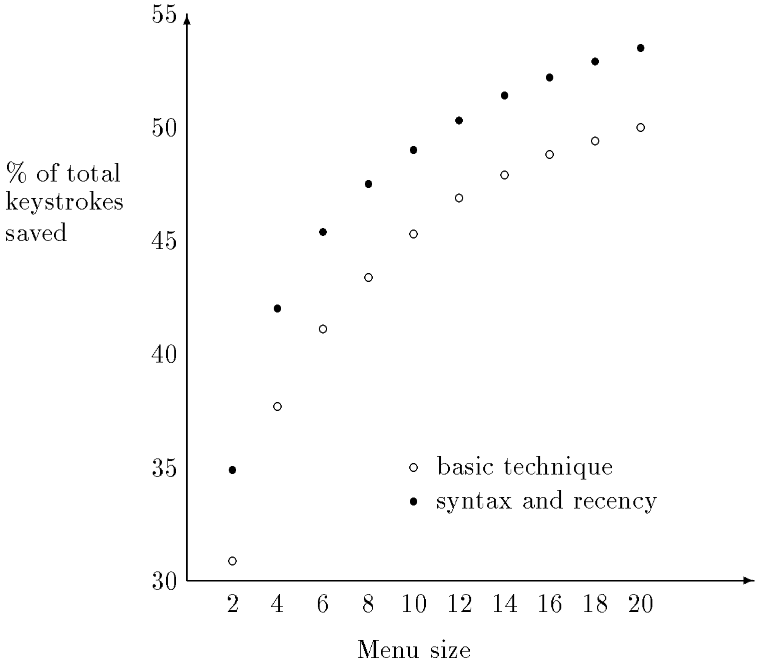

Figure 1: Prediction results for various menu sizes

revising the tags for the most frequent 50 words in the collected data. Unknown words are treated as proper names. Misspellings occur at a rate of roughly 1% in our collected data, usually involving words which were not in the lexicon at the time the corpus was collected and were therefore not predicted.[3] We expect to be able to achieve slightly better results by further expanding the tagset by using semantic tags of the sort discussed in §3 to improve the prediction rates for nouns (and perhaps verbs). We could perhaps also improve results by using trigrams rather than bigrams.

2.2 The limitations of word prediction

It is difficult to compare our results with those in the word prediction literature, because of the lack of common corpora and of an agreed standard of measurement. However our results appear at least comparable with those previously reported for an adult English vocabulary. It seems to be very difficult to achieve keystrokes savings much above 50%. It is sometimes assumed that estimates of entropy (e.g., Shannon's estimate that English is 75% redundant, Brown et al's (1992) upper bound of 1.75 bits per character for printed English) are directly applicable to word prediction and imply that much better keystroke savings could be achieved, in principle. However, the relationship between entropy and word prediction is somewhat complex. Firstly, if a standard keyboard or equivalent device is used to enter letters, then no advantage is being taken of techniques such as Huffman encoding which reduce the number of bits required per letter (although such encodings are relevant to the use of binary switches). Secondly, consider the trivial example of a "language" which consists of 25 equiprobable "words" each of 4 letters, written without spaces, all of which have the common prefix ZZZ (i.e., ZZZA, ZZZB ... ZZZY). This language is 75% redundant but standard left-to-right prediction will not work well (20% keystroke saving with a menu size of 8 with the algorithm used here). A language with words AZZZ, BZZZ ... ZZZY has the same entropy, but allows much better prediction performance (56% saving).[4] To make the comparison between entropy and prediction, we have to consider the information contributed by each input. Of course, in this artificial language, the Z's actually carry no information.

As a rough comparison of word prediction and entropy figures for English, note that in order to

[3]There are some cases where words are deliberately misspelled in order to get better output from the synthesizer, such as *coyote* spelled *kiote*.

[4]The relevance of this to natural languages is that simple left-to-right prediction does not work well if there are a large number of possible suffixes.

achieve 50% keystroke savings with input which, like that of our user, averages 4 letters per word, it is necessary to predict the word on average after 1.5 letters have been input (assuming that the space is predicted). Assuming a letter can be Huffman encoded in 4 bits and choosing from a menu of 8 items is 3 bits, we have an entropy figure of 1.8 bits per character, close to Brown et al's result. For 60% savings, the figure is 1.4 bits. This admittedly crude calculation suggests that it may be unrealistic to expect much better than 50-60% keystroke savings on free text with a usable menu size.

We believe that although prediction techniques are robust and flexible, by themselves they cannot offer the improvement in text input speed necessary to allow natural conversation using a text-to-speech system. Work at the University of Dundee (e.g., Alm et al, 1992; Todman and Alm, this volume) has shown that the extensive use of fixed text for sequences such as greetings and prestored narratives is beneficial in AAC. We would like to extend this to the potentially much wider range of situations where partially predefined strings are appropriate. We are therefore investigating an alternative approach, making use of *cogeneration*, a novel natural language generation technique, which allows the flexible combination of free and fixed text. This is discussed in the next section.

3 The cogeneration approach

Traditionally, natural language generation has been seen as the inverse of parsing, where the input is some sort of meaning representation, such as predicate calculus expressions. This is inappropriate for AAC, since formal meaning representation languages are hard to learn and anyway tend to be more verbose than their natural language counterparts. Instead, in cogeneration, input is partially specified as a series of text units by the user, and the job of the generator is to combine these units into grammatical, coherent sentences which are idiomatic, appropriately polite and so on. To accomplish this, the generator has to be able to order the text units, add inflections and insert extra words (both function and content words). Cogeneration thus builds on work on *compansion* (e.g., Demasco and McCoy, 1992; McCoy, this volume).

Cogeneration involves a combination of grammatical, statistical, and template-based constraints. For AAC devices the templates are designed for particular dialogue situations. The choice of template is made by the user, and the interface provides slots which the user instantiates with text. In many cases, slots will be optional or have default fillers, constructed according to context and previous inputs. We assume that instantiation of the slots can be aided by word and phrase prediction, conditioned on slot choice. Prediction should be much more effective than with free text, since the slots will provide fine-grained syntactic and semantic constraints. The cogenerator operates by combining the constraints specified by the template(s) with the general constraints of the grammar to produce an output sentence, guided by statistical information.

For example, a request template might have slots for requested action and for the requestee. Suppose the user input *open kitchen window* into the requested action slot and that the requestee slot defaulted to *you*. The system might plausibly generate *Please could you open the kitchen window* using fixed text associated with the request template. The user would be given the option of making the output more urgent and less polite, which might result in *Open the kitchen window!*. One significant advantage of the cogeneration technique is that extra information is available to guide the speech synthesizer, allowing more appropriate intonation, prosody and even volume.

Cogeneration involves three different types of knowledge source: a grammar and lexicon, statistical information about collocations and preferred syntactic structures, application- and context-dependent templates. We do not have space here to give details of all these aspects of the cogeneration system, and it would be inappropriate given that development is in its early stages. However, one area which we will examine in slightly more detail, because of its relevance to the work on word prediction discussed above, is the proposed use of semantic categories in the grammar and in the statistical component.

Consider the example given above, where the user inputs *open kitchen window* as the requested action in a request template. Some of the actions required in the course of accepting and processing this input are:

1. Predicting words (e.g. *window* is more likely in this context than *work*).

2. Recognizing *kitchen window* as a unit for generation, so it does not get split up, and preceding it with a plausible determiner.

3. Giving the utterance the correct stress (compounds vary in stress in a way that partly depends on meaning: contrast *cotton bag* meaning *bag made out of cotton* and *bag for cotton*).

Obviously the requirements may differ to some extent for other grammatical constructions: for instance, we have to recognize not just noun-noun compounds but conventionalized adjective-noun combinations (e.g., *social security*). In general, achieving this sort of analysis requires some statistical information about the words in the input and their collocations. One possibility would be to use n-grams based on words (or word stems), but even assuming that we could extract useful information from one of the existing large text corpora rather than an AAC-specific one, the data is still likely to be too sparse for all but the most frequent words. For example, *kitchen window* occurs only once in the approximately one million word LOB corpus, and does not occur at all in the similarly sized portion of the Wall Street Journal distributed as part of the Penn Treebank.

The alternative strategy which we have adopted is to back off to semantic classes for infrequent or unseen collocations. For example, *kitchen* might be classified as **space-loc**, which is intended to encompass locations which have significant spatial extent, and *window* as **figure-ground** (following Pustejovsky (1995) who uses this nomenclature because *window* belongs to a class of words which can refer either to an opening or its filler). The semantic categories are arranged in a hierarchy. The intention is that these categories would improve prediction and support a form of partial parsing: e.g., we could detect that *kitchen window* was a plausible constituent belonging to a particular class of noun-noun compounds, even if our corpus contained no instances of *kitchen window* itself, because the class of **space-loc figure-ground** compounds would be recorded, based on combining data on the frequencies of all compounds which fit this schema (*bedroom window*, *bathroom door*, etc). Similarly, determiner choice will depend to some extent on the verb governing the noun phrase. For instance, although *the* is much more frequent than *a/an* in most corpora, following *buy* the reverse is usually true. But again, available corpora are too small or too unbalanced to determine this information for less frequent verbs, and it is necessary to consider verb classes instead (Levin's (1993) **verbs of obtaining** would be a relevant class for this example).

It may sometimes be necessary to do lexical disambiguation to support generation, but often this is not required. For instance, it would not be necessary to distinguish between house windows and windows on a computer screen if *open window* were input to the request template considered above, because the generated utterance would be the same in either case. Classes such as **figure-ground** are intended to capture some types of systematic polysemy. The use of rather broad categories also reduces the extent to which words are ambiguous with respect to semantic category. We think this is an advantage compared to the use of classes which are more tightly linked to real world knowledge. For example, *window* is given four senses in WordNet (Miller, 1990), corresponding to the usage for buildings, vehicles, envelopes and computers (the metaphorical *window of opportunity* use is omitted). However, for our purposes we suspect that it is unnecessary to distinguish these uses. In general, the granularity of the lexical semantic classes has to be sufficiently coarse to enable reliable statistics to be obtained, but also should not introduce unnecessary ambiguity. For example, having a class **vehicle-part** might be counter-productive, because it would lead to words such as *engine* being ambiguous between their use in vehicles and in stationary objects, which is unwarranted linguistically. For our current purposes, even distinguishing between less related usages such as the luggage use of *trunk* and the (American English) part-of-car use is probably unnecessary, since a more general class, such as **container**, would capture most of the relevant behaviour of both.

We have found that it is possible to use WordNet as a knowledge source to semi-automatically derive semantic classes of the appropriate granularity, even though our semantic hierarchy does not correspond to the WordNet taxonomy. Effectively we regard WordNet as a source of information which we can exploit about word groups. For example, we can identify nodes which are superordinate to a group of senses which should be given the same code, such as *room* (sense 1) for the category **space-loc**. By associating the category with a relatively general node, we can automatically classify a large number of words with a fair degree of reliability. Because the semantic hierarchy does not correspond in a simple way to WordNet, a particular category may have to be associated with several disjoint WordNet subhierarchies, and it is necessary to allow for exceptions.

We are in the process of developing and refining the semantic classification for nouns and verbs. As a first step, we intend to see whether these categories can be used to improve the prediction rate for free text entry, by using the semantic classes to expand the tagset described in §2.1. We expect to use these categories directly in the symbolic grammar, to control preposition selection and to provide schemata for compound nouns, for instance.

4 Conclusions and future work

We have described a prediction system which can adapt to a user's vocabulary and syntax with fairly small amounts of data. The techniques described are all ones which can be used on the fly, although for efficiency it might be desirable to do morphological analysis and n-gram frequency at times when the system is not being actively used (e.g. overnight). We have discussed the limitations of the efficiency of prediction, and introduced the idea of cogeneration which combines free text entry with fixed text associated with templates.

Our prediction work has proven practical utility since our user selects predictions at close to the maximal level (i.e., it is rare for a predicted possibility to be missed). It is difficult to determine how much saving in utterance generation time results from prediction, but it is clear that it considerably reduces physical effort. Our user will not accept any system which does not incorporate prediction. In contrast, our work on cogeneration is at a very early stage. We aim to build the cogeneration system in a modular fashion that allows the reuse of knowledge sources. For example, we expect the semantic categories described in the previous section to be useful in prediction outside the context of a full cogeneration system and also to allow better output from the speech synthesizer, e.g., in the pronunciation of homographs, such as *bow*, in stress placement for compounds etc. We believe that such flexibility is necessary to maximize the chances of NLP research having practical utility for AAC systems.

Acknowledgements

This material is in part based upon work supported by the National Science Foundation under grant number IRI-9612682.

References

Alm, Norman, John L. Arnott and Alan F. Newell (1992) 'Prediction and conversational momentum in an augmentative communication system', *Communications of the ACM,* **35(5)**, 47–57.

Baker, B. R. (1982) 'Minspeak', *Byte,* **7(9)**, 186–202.

Brown, P. F., S.A. DellaPietra, V.J. DellaPietra, J.C. Lai and R.L. Mercer (1992) 'An estimate of an upper bound for the entropy of English', *Computational Linguistics,* **18(1)**, 31–40.

Copestake, A. (1996) 'Applying natural language processing techniques to speech prostheses', *Proceedings of the AAAI Fall Symposium on developing assistive technology for people with disabilities,* MIT, Cambridge, MA.

Darragh, J. J. and I. H. Witten (1992) *The reactive keyboard,* Cambridge University Press.

Demasco, Patrick W. and Kathleen F. McCoy (1992) 'Generating text from compressed input:an intelligent interface for people with severe motor impediments', *Communications of the ACM,* **35(5)**, 68–78.

Levin, B. (1993) *English verb classes and alternations,* University of Chicago Press, Chicago.

Miller, George (1990) 'WordNet: An on-line lexical database', *International Journal of Lexicography,* **3**, 235–312.

Newell, Alan F., John L. Arnott, Alistair Y. Cairns, Ian W. Ricketts and Peter Gregor (1995) 'Intelligent systems for speech and language impaired people: a portfolio of research' in Edwards, Alistair D. N. (ed.), *Extra-Ordinary Human-Computer Interaction,* Cambridge University Press, pp. 83–101.

Pustejovsky, J. (1995) *The Generative Lexicon,* MIT Press, Cambridge, MA.