

The Development of an Experimental Discrete Dictation Recognizer

FREDERICK JELINEK, FELLOW, IEEE

Invited Paper

This paper describes an experimental real-time recognizer of isolated word dictation implemented at the IBM Thomas J. Watson Research Center, on a system of commercially available computers and array processors. The recognizer's intended use is creation of office memoranda. It is based on a 5000-word vocabulary. A specially designed workstation enables the user to correct and edit the transcribed speech.

The paper outlines the self-organized, statistical approach underlying the basic algorithms of the recognizer. Results of several recognition experiments are then presented. The rest of the paper considers important issues in the future development of dictation recognizers, such as vocabulary selection, language model creation, and human factors.

I. INTRODUCTION

Since 1972 the Continuous Speech Recognition Group of IBM has been working on large-vocabulary speech recognition [1], [2]. In 1981 we decided to develop a real-time recognizer capable of handling dictation of office correspondence. The algorithms known to us at that time could not provide for real-time recognition of continuous speech on any feasible combination of commercially available processors, and it seemed prudent not to rely on self-implemented hardware. Thus we limited the proposed system to isolated word input. Preliminary simulation experiments [3], [4] suggested that discrete dictation with immediate text display would be a more productive text-creation mode than either handwriting or dictation on a standard recording machine, and that users preferred discrete input with large vocabularies to continuous input with small vocabularies. We expected that experience with the recognizer would provide valuable information about human factors of speech recognition.

In June 1984 we completed a real-time discrete dictation recognizer with a workstation that allows speech or keyboard editing of text (e.g., correction, insertion, etc.) which appears on a screen. The recognizer uses only algorithms also applicable to continuous speech recognition.

Our recognizer runs on the Speech Development System,

consisting of an IBM 4341 host machine and three Floating Point Systems 190L array processors, to which a workstation based on an Apollo DN-400 computer is attached. The speech signal is picked up by a Crown PZM-6S pressure-zone microphone placed freely on the desk. A mouse is used for pointing. The system is diagrammed in Fig. 1. Fig. 2 depicts the workstation, the microphone is above the mouse on the right.

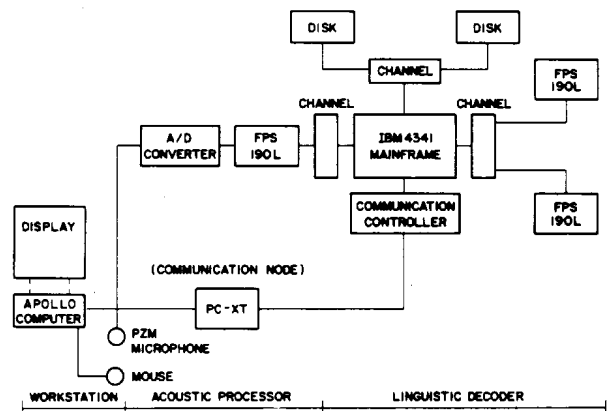


Fig. 1. Schematic of the hardware components of the Speech Development System.

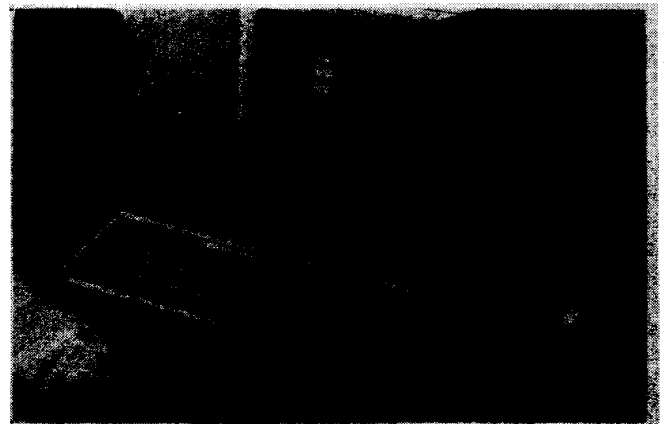


Fig. 2. The Speech Development System workstation. The microphone is on the right of the screen.

Manuscript received March 6, 1985; revised April 1, 1985.

Petitioner has added a romanette to page (i).
Otherwise, it leaves the original page numbering.

0018-9219/85/1100-1616\$01.00 ©1985 IEEE

MICROSOFT CORP.
EXHIBIT 1014

(Duplicate first page for exhibit labeling, per 37 C.F.R. § 42.63(d)(2)(ii).)

IEEE Xplore Repository Article ID: 1014, 1985

The Development of an Experimental Discrete Dictation Recognizer

FREDERICK JELINEK, FELLOW, IEEE

Invited Paper

This paper describes an experimental real-time recognizer of isolated word dictation implemented at the IBM Thomas J. Watson Research Center, on a system of commercially available computers and array processors. The recognizer's intended use is creation of office memoranda. It is based on a 5000-word vocabulary. A specially designed workstation enables the user to correct and edit the transcribed speech.

The paper outlines the self-organized, statistical approach underlying the basic algorithms of the recognizer. Results of several recognition experiments are then presented. The rest of the paper considers important issues in the future development of dictation recognizers, such as vocabulary selection, language model creation, and human factors.

I. INTRODUCTION

Since 1972 the Continuous Speech Recognition Group of IBM has been working on large-vocabulary speech recognition [1], [2]. In 1981 we decided to develop a real-time recognizer capable of handling dictation of office correspondence. The algorithms known to us at that time could not provide for real-time recognition of continuous speech on any feasible combination of commercially available processors, and it seemed prudent not to rely on self-implemented hardware. Thus we limited the proposed system to isolated word input. Preliminary simulation experiments [3], [4] suggested that discrete dictation with immediate text display would be a more productive text-creation mode than either handwriting or dictation on a standard recording machine, and that users preferred discrete input with large vocabularies to continuous input with small vocabularies. We expected that experience with the recognizer would provide valuable information about human factors of speech recognition.

In June 1984 we completed a real-time discrete dictation recognizer with a workstation that allows speech or keyboard editing of text (e.g., correction, insertion, etc.) which appears on a screen. The recognizer uses only algorithms also applicable to continuous speech recognition.

Our recognizer runs on the Speech Development System,

Manuscript received March 6, 1985; revised April 1, 1985.

The author is with the Continuous Speech Recognition Group, IBM Thomas J. Watson Research Center, Yorktown Heights, NY 10598, USA.

consisting of an IBM 4341 host machine and three Floating Point Systems 190L array processors, to which a workstation based on an Apollo DN-400 computer is attached. The speech signal is picked up by a Crown PZM-6S pressure-zone microphone placed freely on the desk. A mouse is used for pointing. The system is diagrammed in Fig. 1. Fig. 2 depicts the workstation, the microphone is above the mouse on the right.

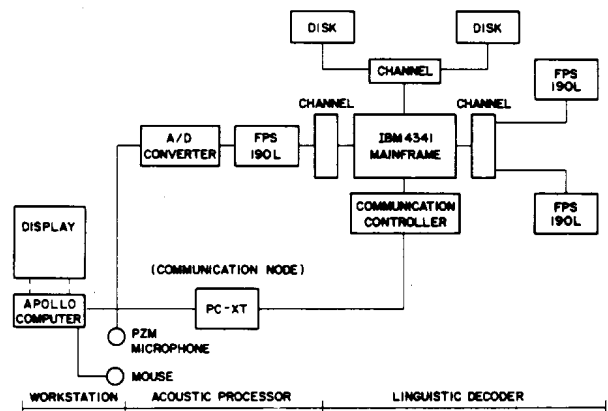


Fig. 1. Schematic of the hardware components of the Speech Development System.

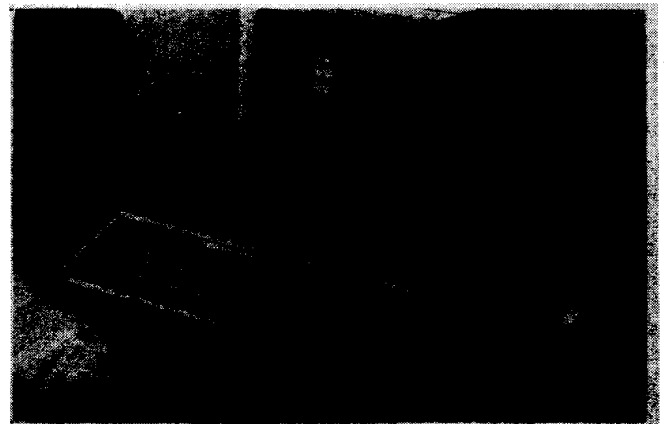


Fig. 2. The Speech Development System workstation. The microphone is on the right of the screen.

The vocabulary is limited to 5000 words and a spelling mode facilitates input of the complete text by speech. The system operates in a quiet office environment. A speech training sample 20 min long is required to adjust the recognizer to a new user. It is capable of keeping up with an average discrete speech rate of 90 words a minute, but a variable delay of about 3 words exists between the time a word is spoken and displayed on the screen, allowing the use of bidirectional context by the recognizer.

The next section is devoted to an overview of our approach to speech recognition. A more precise formulation was given elsewhere [1], [2]. In Section III we give results of experiments with our system. Section IV addresses the problems of vocabulary selection and personalization. Section V deals with achievement of better language models and measurement of their quality. Section VI concludes the paper with a discussion of desirable characteristics of practical recognizers for text creation and of our plans to achieve them.

II. A STATISTICAL APPROACH TO SPEECH RECOGNITION

We will now outline our approach to speech recognition. Let

$$W = w_1, w_2, \dots, w_n \quad (2.1)$$

denote a string of n words, and let A denote the acoustic evidence (data) on the basis of which the recognizer will make its decision about which words were spoken. If $P(W/A)$ denotes the probability that the words W were spoken given that the evidence A was observed, our recognizer is designed to decide in favor of a word string \hat{W} satisfying

$$P(\hat{W}/A) = \max_W P(W/A). \quad (2.2)$$

Decision criterion (2.2) is natural and generally acceptable. For a language whose spelling system is roughly phonetic, a dictation recognizer based on (2.2) will tend to minimize the number of corrections necessary to obtain the spoken text.

Application of Bayes' formula

$$P(W/A) = \frac{P(W)P(A/W)}{P(A)} \quad (2.3)$$

to the criterion (2.2) reveals the different research areas of speech recognition.

We must first determine the nature of the acoustic evidence A on which the recognizer's decision will be based. The transformation of the speech signal into the evidence A is called *acoustic processing*. In our recognizer (see Fig. 3), the acoustic signal is synchronously transformed by the

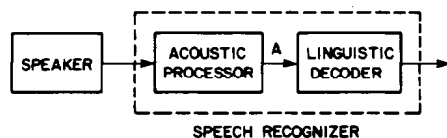


Fig. 3. The speech recognition system.

Acoustic Processor into a sequence of labels $A = a_1, a_2, \dots$ from an alphabet \mathcal{A} of size 200. Fig. 4 shows the acoustic processor schematically. Every 10 ms, the signal processor extracts a vector of 20 parameters from a 20-ms window of

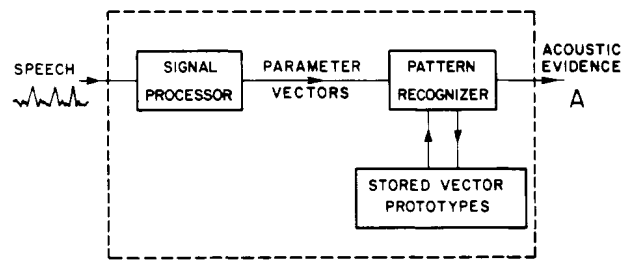


Fig. 4. The acoustic processor.

the speech input. This vector is compared to a set of 200 pre-stored prototype vectors and the label of the prototype that is closest to the parameter vector is put out [1], [2]. The design of our signal processor is based on current knowledge of the performance of the human ear [5]. The 200 prototypes are selected essentially by the method of vector quantization [6], [7]. The signal processor output parameter vectors corresponding to a 5-min long speech training sample are collected. Considered as points in a 20-dimensional space, they are partitioned into 200 clusters. The cluster centers become the stored prototypes which in our system are specific to each speaker.

Returning to (2.3), it is next necessary to create an *acoustic model* describing statistically the interaction between the speaker and the acoustic processor. The acoustic model allows us to compute the probability $P(A/W)$ that the acoustic processor will put out the label string A if the speaker says the word sequence W . In our system, the acoustic model is built up out of models of pronunciation of phonetic symbols. In brief, the model for a word string W consists of a concatenation of models of the individual words w_i , and these in turn are made up of a concatenation of models of the phonetic symbols defining the basic word pronunciations.

To each word w of the vocabulary \mathcal{W} there corresponds a baseform $B(w) = b_1, \dots, b_k$ consisting of a string of symbols b_i from a phonetic alphabet \mathcal{B} . The baseform models the basic pronunciation of the word w .¹ To each symbol b of \mathcal{B} there corresponds a Markov source (hidden Markov chain) [1], [2], [8] which is an abstract model of the response of the acoustic processor to the act of "pronouncing" the phonetic symbol b . The transitions between states are labeled by letters of the alphabet \mathcal{A} (Fig. 5). As the source changes state, it outputs the label attached to the transition it takes. The acoustic model of any word w of the vocabulary \mathcal{W} is obtained by concatenating the Markov sources corresponding to the symbols of the baseform $B(w)$ of the

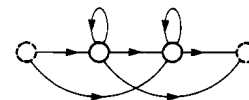


Fig. 5. Structure of a Markov model of acoustic label generation resulting from "pronunciation" of phonetic symbols. Each solid arc represents 200 labeled transitions, one for each letter of the output alphabet \mathcal{A} . The transition probabilities are estimated during the training process and differ for different phonetic symbols.

¹The phonetic alphabet \mathcal{B} contains several symbols corresponding to various modes of silence and so in isolated word dictation the baseforms both start and end with substrings of these silence symbols.

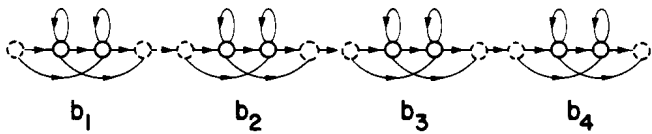


Fig. 6. Markov model of a word whose baseform is b_1, b_2, b_3, b_4 .

word (see Fig. 6). The probability $P(A/W)$ is then computed as follows: A composite Markov source for the string W is obtained by concatenating the Markov sources corresponding to the n words of $W = w_1, \dots, w_n$. $P(A/W)$ is the probability that the composite source will produce the string A when it is placed in its initial state and left running until it reaches the final state.²

The various Markov sources (Fig. 5) corresponding to the phonetic symbols b of the alphabet \mathcal{B} are characterized by the different probabilities with which these sources take various transitions labeled by the letters a of the alphabet \mathcal{A} . While the baseforms $B(w)$ used by our recognizer are speaker independent (although intended for the standard American dialect), the transition probabilities of sources are specific to the speaker. They are estimated by the Forward-Backward algorithm [1], [2], [9] from data A generated by the speaker's reading a prescribed text of 100 sentences, resulting in a 20-min speech sample.

Next, in order to use (2.3) it is necessary to construct a *language model* computing the probability $P(W)$ that the speaker will wish to say the word string W . Now

$$P(W) = \prod_{i=1}^n P(w_i/w_{i-1}, \dots, w_1). \quad (2.4)$$

But the probability values $P(w_i/w_{i-1}, \dots, w_1)$ cannot in reality depend on the full apparent set of i parameters, since the number of these values N^i for even a moderate vocabulary size N would be too large to be estimated, stored, or retrieved. Hence the various possible conditioning histories w_1, \dots, w_{i-1} must be partitioned into a manageable number of equivalence classes. The classification would most fruitfully be based on syntactic and semantic information. The selection of the exact classification scheme, and its use in determining the probability values from a large amount of text, is an unsolved problem that will claim increasing attention of researchers.

In our system, all histories ending in the same two words w_{i-2}, w_{i-1} are considered equivalent, so that the factors $P(w_i/w_{i-1}, \dots, w_1)$ of (2.4) are defined equal to $P(w_i/w_{i-1}, w_{i-2})$. Unfortunately, these probabilities cannot be approximated directly by relative frequencies obtained by counting trigrams occurring in some large text, since the vast majority of possible English word trigrams will not take place even in very large databases. In fact, the number of different trigrams of a 5000-word vocabulary is 1.25×10^{11} , which is far bigger than any conceivable database.

Our current method of estimating the required probabilities was worked out by S. Katz [10], based on Good's elaboration [11] of an argument by Turing. We present here a rudimentary version that contains the main idea. We first develop an estimate of probabilities $P(w_2/w_1)$ and then use it to compute $P(w_3/w_2, w_1)$.

²A computationally efficient algorithm for computing $P(A/W)$ was given by Bahl and Jelinek [28].

Consider two samples of text generated by the same source. Referring to the first sample, let N be the total number of bigrams, let $N(w_1, w_2)$ be the number of times the words w_1, w_2 occur adjacent to each other, and let $N(w_1)$ be the number of times w_1 takes place. Finally, let r be the number of different pairs that occur exactly once (i.e., such that $N(w_1, w_2) = 1$). Turing argues that r/N is a good estimate of the probability that a random pair of words selected from the second sample will be one never seen in the first. Then it is reasonable to estimate the required conditional probability by the formula

$$P(w_2/w_1) = \begin{cases} \left(1 - \frac{r}{N}\right) \times \frac{N(w_1, w_2)}{N(w_1)}, & \text{if } N(w_1, w_2) > 0 \\ \left(\frac{r}{N}\right) \times \frac{N(w_2)}{K_2(w_1)}, & \text{otherwise} \end{cases} \quad (2.5a)$$

where $K_2(w_1)$ is a normalizing factor that assures that the probabilities estimated by (2.5a) add up to 1 when summed over all the words w_2 . The method is referred to as one of "backing off" from bigram to unigram estimation when bigram data are not available.

Let s be the number of trigrams w_1, w_2, w_3 occurring exactly once in the training text. Then application of backing off approach leads straightforwardly to the formula

$$P(w_3/w_2, w_1) = \begin{cases} \left(1 - \frac{s}{N}\right) \times \frac{N(w_1, w_2, w_3)}{N(w_1, w_2)}, & \text{if } N(w_1, w_2, w_3) > 0 \\ \left(\frac{s}{N}\right) \times \frac{P(w_3/w_2)}{K_3(w_1, w_2)}, & \text{otherwise} \end{cases} \quad (2.5b)$$

where $P(w_3/w_2)$ is computed by (2.5a) and $K_3(w_1, w_2)$ is a normalizing factor.

The backing off formulas of Katz [10] are more subtle than those of (2.5), but they too depend on the frequencies of word trigrams, bigrams, and unigrams, computed from a very large (25 million words) office correspondence text. Section V examines additional aspects of language modeling.

Our system actually uses two language models. In addition to the one described above, another smaller one exists to support spelling recognition. It has the form (2.5), but its "words" belong to a vocabulary of size 76 consisting of letters, digits, punctuation marks, and nine control words. Its basic relative frequencies were estimated from a sample of 5 million characters of office text.

So far, the discussion of the recognizer components has taken care of the numerator of (2.3). Since the denominator is not a function of W , it need not be evaluated to find the maximizing sequence \hat{W} satisfying (2.2). So the remaining problem is that of *hypothesis search* for \hat{W} . From the combinatorics involved, it is clear that this search cannot be exhaustive even for a very moderate vocabulary size N and sequence length n . It is necessary to severely limit the search to only a very small fraction of the N^n possible word strings W .

A method that is successful is an adaptation of stack sequential decoding studied extensively by Information Theory [12]. We have described it elsewhere [1], [2]; here we

just outline it. The hypothesis tree appropriate to the search has a root node from which stem N branches, one for each word in the vocabulary. From each branch there stem N more branches, etc. The problem of speech recognition can be thought of as one of finding a complete tree path (that corresponds to one definite string of words W) that is most probable given the acoustic evidence A (see (2.2)). The idea is to conduct a left to right search of the acoustic evidence A , comparing it to various subpaths (of different length) of the word hypothesis tree. The search is efficient if relatively few paths are examined.

Let $L(W)$ be an evaluation function of the path corresponding to the word string W that depends appropriately on $P(W)$ and $P(A/W)$. Assume that L has the property that if $L(W) > L(W')$ then W is more likely than W' to be the beginning of the spoken word sequence resulting in A , regardless of the relative lengths of W and W' . Such a function is described in a previous paper [2]. A good search strategy is as follows:

- 1) Arrange words w of the vocabulary \mathcal{V} in decreasing order of $L(w)$, creating a stack.
- 2) Let W^* be the path on top. Remove it and insert into the stack its extensions $W' = [W^*, w]$ for all vocabulary words w , keeping the stack ordered by the values $L(W)$.
- 3) If the last word of the path W^* on top of the stack corresponds to an end of sentence marker, and if W^* accounts for the complete evidence A , decide that W^* was the sentence spoken. Else go to step 2).

The algorithm above can serve as a basis for a practical search procedure because the function L has the property that the value $L(W')$ is a simple update of $L(W^*)$ that depends essentially only on w and on that portion of acoustics A corresponding to w .

As it stands in step 2), the algorithm requires the evaluation of L , called *the detailed match*, for N different extensions W' of W^* . For large vocabularies this represents too much work. Thus the list of extensions of W^* must be reduced further. One way to do that is to split the acoustics A into the front part A^* accounted for by W^* , and the remaining tail part A' , and then to confine the detailed match to a small subset of words w that could possibly account for the beginning of the tail A' . The determination of the subset is called *the fast match*, and many techniques are possible.

In one method of fast match the vocabulary is pre-clustered into subsets of acoustically similar words [13]. Each subset is associated with a word "centroid." During recognition the data tail A' is matched against the centroids. The word extension list for W^* is made up of subsets having centroids whose match exceeds a pre-determined threshold.

Another option is to first carry out the match $P(A'/w)$ over all words w of the vocabulary, but in an approximate, fast manner [14]. The extension list is then made up only of those words whose approximate match exceeds a threshold.

The processor configuration of the recognition system of Fig. 1 mirrors the research categories discussed in this section. The array processor on the left (connected to the microphone) is devoted to acoustic processing, the remaining two array processors carry out the detailed and fast matches, and the host IBM 4341 system computes the language model probabilities, conducts the hypothesis search, and communicates the recognition results to the workstation.

III. THE PERFORMANCE OF THE YORKTOWN DICTATION RECOGNIZER [15]

The experiments described here tested the recognizer under three conditions:

- 1) prerecorded speech
- 2) speech read in real time
- 3) speech produced spontaneously in real time.

A total of five speakers, including one female, were represented.

Each speaker trained the system by reading 100 sentences containing 1107 words (about 20 min of speech). Next, in the same recording session, the speaker read 50 test sentences containing roughly 591 words, which we call the "prerecorded speech." Both training and test sentences were chosen from our database of office correspondence (the test sentences, however, were kept separate from the data used to estimate language model parameters).

The speech read in real time and the spontaneous speech were collected over the course of 2 months in half-hour dictation sessions. Within each session, the first 20 test sentences (of the 50 mentioned above) were read; this constituted the "read speech." The "spontaneous speech" consisted of actual memos dictated by the speakers, and averaged 150 words per memo. Each time a new topic was chosen by the speaker.

In the next section we discuss how the 5000 words of the system's vocabulary were chosen. The recognition results on words of this vocabulary are shown in Table 1 (words

Table 1 Word Error Rate of the Real-Time Recognizer

Speaker	Prerecorded Speech (%)	Read Speech (%)	Spontaneous Speech (%)
JC	2.7	3.1	7.8
BF ¹	3.0	4.3	6.7
MP	1.4	3.3	3.7
SD	2.0	3.8	6.2
LB	1.0	1.0	4.1
Average	2.0	3.1	5.7

¹Female speaker.

not in the vocabulary have been omitted from the data). As can be seen, performance on prerecorded speech is better than on live speech, and performance on read speech is better than that on spontaneous speech. The performance on prerecorded speech is high because the test and training data were collected during the same session. Performance on spontaneous speech may be lower than that on read speech for a number of reasons. First, the error rate on spontaneous speech reflects not only dictated sentences but also efforts by the speaker to experiment with the system. For example, a speaker might try to correct a recognition error by redictating portions of the memo and the same error is likely to recur. Second, the language model may be doing a poorer job of predicting words in spontaneous speech than for "normal" office correspondence, and, in fact, the perplexity³ of text dictated spontaneously by our subjects is higher than that of the test data. Also, any word not in the vocabulary places the language

³Perplexity is a basic measure of complexity of text relative to the language model used [16], [17]. It is further discussed in Section V.

model automatically in a wrong state for prediction of the next word. Finally, our training method does not reflect the change in speaker acoustic style from read to spontaneous speech.

The system has also been tested on a number of speakers with foreign accents (British English, Persian, Russian, Czech); on average, there were again half as many word errors.

IV. SELECTION OF A RECOGNITION VOCABULARY AND ITS PERSONALIZATION

We define a word by its spelling. Thus two differently spelled inflections of the same stem constitute different words. Different meanings or parts of speech of the same spelling (homographs) constitute the same word. The 5000 words of our vocabulary were selected in two steps. First a basic subvocabulary of size M was constructed, consisting of words whose availability was thought necessary, such as recognizer control words, numbers (e.g. "thousand," "million," etc.), dates (e.g., "Monday," "November," etc.), and the first and last names of some of our colleagues at IBM Research. The remaining 5000 - M words of the vocabulary were chosen to be those that have the highest minimal count in three databases (OC1, OC2, and OC3) and do not belong to the subvocabulary. This selection method left out words that have little universal use and are peculiar to a particular database.

We are in possession of three additional databases. The first, PDB, consists of 2 million words of correspondence by a single colleague at the T. J. Watson Research Center. The second, MNG1, contains written memoranda from one of us, from the director of research, and from the chairman of the corporation. The third database MNG2 was created from transcribed spontaneous dictation of memoranda by four Research staff members. Our 5000-word vocabulary covered about 93 percent of words in all three of these text collections. That is, with probability 0.07 a random word selected from any of the texts does not belong to the vocabulary. As a consequence, if no words are spelled, the error rate of our recognizer will exceed 7 percent regardless of how good it is otherwise. Moreover, when a word outside the vocabulary is spoken, the resulting recognition error forces the language model into a wrong state, increasing the probability of an error in the next word.

A substantially better text coverage by the vocabulary is clearly desirable. It must be achieved by a combination of vocabulary size increase and personalization. Inspection of standard references on frequencies of English words [18], [19] clearly indicates that a large fixed size alone will not do the job. In fact, quite familiar words such as "admonition" or "deluded" occupy a position higher than 86 000 in the American Heritage list [19]. And this does not even take into account technical words particular to the individual user's field! On the other hand, it turns out that the total vocabulary of all works of Shakespeare is only 29 000 [20], and the vocabulary of a 1 million-word large varied office text by many authors (database OC1) is 19 000, so the problem is not a hopeless one.

Using the previously described method, R. L. Mercer [21] constructed vocabularies of varying larger sizes. Table 2 shows the PDB coverage when names and acronyms occurring in it have been disregarded. The selection rule for the

Table 2 Static Coverage of the Personal Database PDB as a Function of Vocabulary Size

Vocabulary Size	Text Coverage (%)
5000	92.5
10 000	95.9
15 000	97.0
20 000	97.6

20 000 word vocabulary had to be modified because the OC database contains fewer than that many different words.

Our method of fixed vocabulary construction may be satisfactory if the intended user belongs to some easily identifiable category (e.g., employee of the data processing industry) for which text databases are available. Still, it is hardly acceptable that the recognizer's use would have to be identified before the system could be adequately equipped. The only way out seems to be some more practical method of personalization or a fixed vocabulary of really gigantic size. The latter would have to be synthetic, consisting of inflections and derivations based on a substantially smaller set of building blocks (stems? morphemes?). Henry Kucera estimates that vocabulary compression by a factor of 6 to 10 may be attainable in this way [20].

A crude method of personalization is achieved by maintaining during dictation a dynamically varying vocabulary consisting at any given moment of the last N different words used. The coverage is then defined as the asymptotic probability that the word spoken next already belongs to the vocabulary. Table 3 gives the result for the PDB database [21] when names and abbreviations have been ex-

Table 3 Dynamic Coverage of the Personal Database PDB as a Function of Vocabulary Size (The last column indicates the amount of text processed before the number of different words contained in it is equal to the vocabulary size.)

Vocabulary Size	Text Coverage (%)	Text Size to Reach Coverage
5000	95.5	56 000
10 000	98.2	240 000
15 000	99.0	640 000
20 000	99.5	1 300 000

cluded. The second column is the good news (99-percent coverage by a 15 thousand-word vocabulary), while the third is the bad: It takes 613 thousand words of text before a set of 15 thousand different words is assembled. This then is not a practical way of vocabulary personalization, except for the most prolific of writers. It may, however, result in a satisfactory vocabulary for an entire installation site if the text produced by all the resident users is pooled.

There are at least four different ways in which a vocabulary is particular to a speaker. First is the peculiar active vocabulary reflecting his habits of expression, possibly related to his level of education. Next is the general domain of his discourse (e.g., data processing, musicology, medical reports, etc.) containing technical expressions, cliches, and such. Then, there are the names, addresses, and other information that reflects the user's needs. Finally, there is the vocabulary needed for the document at hand. A large, fixed vocabulary can take care of the first category and

conceivably of some part of the second as well. The rest must be obtained through active participation by the user.

In the discussion above we have not considered the problem of language modeling. We constructed our trigram language model (see Section III and [1], [2]) using the 25 million word text database OC3. The trigrams from OC3 made up of words in our vocabulary covered 84 percent of correspondingly restricted trigrams in the test databases PDB and MNG1, while the coverage of the same texts by trigrams from the 1 million word database OC1 was only 55 percent.

A trigram language model for a personalized vocabulary cannot be constructed solely from the text created by the user. He will never dictate enough. The only conceivable way is to extract the model from an already stored trigram collection appropriate to a large vocabulary that includes the personalized one. It is obvious that the more the personalized vocabulary of size N differs from the N most frequent words of the training corpus, the less will the trigrams in the dictated text be covered by trigrams collected from the training text. Therefore, a truly gigantic training corpus would have to be used as a basis for a satisfactory dynamic (personalized) language model.⁴

The need for a more general method of language model construction is thus clear. It seems that the trigram idea can be preserved only if it is based on equivalence between words whose statistical properties are similar. For instance, one could envision having 5000 categories g , one for each word in the basic vocabulary for which a trigram model $P(g_3/g_2, g_1)$ was constructed based on a training corpus representative of English. Every word w of the total potential vocabulary would then be classified as belonging to one of the 5000 categories. The probabilities of use of words w in category g , $P(w/g)$, could be estimated. The actual language model for a large or personalized vocabulary would then function using the trigram formula

$$P(w_3/w_2, w_1) = P(w_3/g_3) \times P(g_3/g_2, g_1) \quad (4.1)$$

where g_i , $i = 1, 2, 3$, are the categories to which the words w_i belong.

V. LANGUAGE MODEL PERPLEXITY AND THE SHANNON GAME

There is, of course, nothing to recommend the trigram language model except its simplicity and ease of construction from training text. Just how good is it? Information theory measures the information content of sources by their entropy H [22]. Since in speech recognition the source can be thought of as generating word sequences, then by definition

$$H = - \lim_{N \rightarrow \infty} \left(\frac{1}{N} \right) \times \log_2 P(w_1, w_2, \dots, w_N) \quad (5.1)$$

where N is the size of the sample text generated by the source.

We see from (5.1) that to compute the entropy H one would need to know the actual probabilities $P(w_1, w_2, \dots, w_N)$ of strings of the language. These probabilities

⁴If we think of the personalized vocabulary as consisting of a common core together with a personal set, the words in the latter will on average be longer and thus acoustically more distinguishable. This may compensate for weak prediction by the language model of words outside the core.

are, in practice, unknowable. At best we can have their estimates $\hat{P}(w_1, w_2, \dots, w_N)$ which the language model provides to the recognizer. Thus from the point of view of the recognizer, the difficulty of recognition of a given text is measured by its *logprob* (LP)

$$LP = - \lim_{N \rightarrow \infty} \left(\frac{1}{N} \right) \times \log_2 \hat{P}(w_1, w_2, \dots, w_N). \quad (5.2)$$

It is easy to show [22] that $LP \geq H$, assuming properly ergodic behavior of the text generating source. In the case of the trigram language model

$$LP = - \lim_{N \rightarrow \infty} \left(\frac{1}{N} \right) \times \sum_{i=1}^N \log_2 P(w_i/w_{i-1}, w_{i-2}). \quad (5.3)$$

It is intuitively more satisfying to measure the difficulty of a recognition task relative to a given language model by the value of its *perplexity* (PP) defined by [16], [17]

$$PP = 2^{LP}. \quad (5.4)$$

A basic argument of Information Theory shows that to the first order of approximation (which disregards the acoustic distances between particular words) the task can be thought to be as difficult as would the recognition of a language with PP equally likely words. Thus perplexity gives the correctly computed "branching factor" of the language model.

In practice, any estimation of the perplexity depends on the sample text whose length N is finite. The perplexity of our trigram model when applied to the databases OC1 and MNG2 is 70 and 128, respectively. A different model will approximate $P(w_1, w_2, \dots, w_N)$ in a different way and will thus lead to a different perplexity that will be lower if the model is better.

Entropy measures the intrinsic difficulty of the task, logprob its difficulty relative to the language model used. The difference between logprob and entropy, if it were known, would indicate the scope of potential language model improvement. Folklore has it that in a famous paper [23] Shannon estimated the entropy of English to be 1 bit per letter, i.e., assuming the average word length to be 5.5 letters, about 5.5 bits per word. Actually, based on a sample text, Shannon provided bounds on logprob of natural language models intrinsic to human subjects. His bounds place logprob between 0.6 and 1.3 bits, and thus perplexity between 10 and 142. We will outline below his method that depends on guessing a text. His results would be valid only if the text they were based on was representative and large enough for his statistical estimates to converge. Representativity of English can never be established, and the test passage Shannon chose was much too short. But given enough perseverance by human subjects, valid results for the narrow genre of office correspondence could be obtained. Also, introspection during guessing may provide clues to the construction of more effective language models.

In the Shannon method, the subject is asked to guess the text, letter by letter (including the space marker between words, punctuation, etc.). He is advised about the correctness of each guess. When he finds the first letter, he starts guessing the second, then the third, etc. . . . Let r_i be

the number of tries it takes to guess the i th letter. Then assuming that given the same state of knowledge the subject would always guess in the same order, the text can be perfectly recovered by the subject from the guess order sequence $r_1, r_2, \dots, r_i, \dots$. The sequence is thus a perfect encoding of the text. Hence the logprob of $r_1, r_2, \dots, r_i, \dots$ is the same as the logprob of the text. Let $Q(n)$ be the relative frequency with which the number n appears in the order sequence. Then it is easy to show [22] that

$$H^+ = - \sum_n [Q(n) \times \log Q(n)] \quad (5.5)$$

is an upper bound on the logprob of the encoding $r_1, r_2, \dots, r_i, \dots$ and therefore on the logprob of the text. If m is the average number of letters per word in the text, then 2^{mH^+} is an upper bound on its perplexity. Shannon also derived a lower bound formula on text logprob. Cover and King later obtained a direct estimate of text logprob by replacing the guesses in Shannon's procedure by bets [24].

Stephen and Vincent Della Pietra have automated the Shannon procedure [23] on a computer [25]. The resulting Shannon game has several additional features. The text is chosen at random from a database. It is possible to have the estimate converge faster by having part of the text revealed from the start. The information contained in n -grams can be measured by selecting at random the word to be guessed and revealing to the subject exactly n preceding words. The subject is also provided with two aids. Consistent with the letters guessed so far, the possible words of the vocabulary are shown together with their probabilities as computed by the trigram language model (given the preceding two words of the text). Another display panel shows the letter probabilities of the guess to be made next, as computed by the language model consistent with the revealed past and the guesses already made for the current letter position. The computer keeps the running H^+ score and compares it to the score that would have been incurred had the subject been guided strictly by the language model.

As already pointed out, our aim in playing the Shannon game is not to estimate the human perplexity of office correspondence, but to discover what manner of information is used by humans in word prediction. We hope that this will help us design the structure of future language models. Even keeping in mind that H^+ is only an upper bound, it is already clear that the trigram language model can be substantially improved. In the Shannon game humans beat the trigram model by factors of 3 or more in perplexity (2^{mH^+}). The advantage of humans seems mostly based on their ability to use relevant information found contained in a text passage that often considerably precedes the currently guessed letters.

VI. FUTURE ASPECTS OF TEXT CREATION BY VOICE

Gould and Boies [26] "observed that handwriting was the main method by which principals composed. (They) had the intuitive belief that dictation was potentially a superior method of composition. Dictation is potentially five times faster than writing, on the basis of estimates of maximum writing and speaking rates when composition is not required (see [27]). Dictation may also be qualitatively superior: potentially faster transfer of ideas from limited capacity human working memory to a permanent record may reduce forgetting attributable to interference or decay."

Those who could pay for it have been creating text by voice for a long time. Originally by direct dictation to a secretary, later also by using conventional dictating (recording) equipment. This method of text creation is linear and does not allow for easy review and immediate modification of what has been said. Insertion, change of phrasing, or text reorganization while dictating are not possible. The job cannot be completed in one sitting, there are delays, and the convenience, work load, office hours, and various habits of the typist must be taken into account.

Word processing via computer terminals or personal computers gives more control over the process to the user, but the natural human inclination to communicate by voice rather than keyboard is sacrificed. We hope that automatic dictation recognition will solve the problem. Two conditions must be fulfilled: the speech recognizer must be reliable, and the man-machine interface must be a convenient one.

The interface problem, shared with word processing, is not a trivial one. Perhaps due to early childhood experiences (I believe the cause is deeper), hands-on editing with paper and pencil seems preferable. It is easy to direct the typist to rearrange text by circling it, or to indicate text insertion, correction, deletion, or indentation. Pointing by a mouse or a touch-sensitive screen, or text movement via marking the beginning and end of a section followed by cursor specification of the place of insertion, are not as convenient. To get a really good interface, it may prove necessary to use writing tablets and achieve automatic recognition of hand-produced diacritical marks and possibly of handwriting as well. Clearly, the recognizer must be an integral part of the total office support system that includes facilities such as electronic mail or database access. The machine should not take up extra space or be otherwise intrusive. Thus most of the feedback it gives should be visual and not acoustic.

To gain reliability, accuracy, and ease of use, the IBM project will attempt to enhance four facets of recognition: vocabulary text coverage, noise immunity, speaker independence, and continuous speech. The first problem was discussed in Section IV. The remaining three will now be addressed.

Our pressure-zone microphone picks up all sounds in its vicinity and the recognizer interprets them as part of the dictation. We consider this a worthwhile price to pay for eliminating the head mounted microphone that usually inconveniences users of speech recognizers. Nevertheless, the appearance of extraneous text on the screen is extremely annoying, as is the somewhat increased error rate due to background noise. It should be possible to adjust our signal processing to filter out speech by others than the dictator, as well as noise bursts such as telephone ringing, dropped books, rustling of paper, or door slams. Tracking of the speaker's position by a microphone array may prove part of the answer. Improved recognition algorithms should take care of higher ambient noise levels.

Our training procedure currently consists of reading of 100 sentences of average length of 12 words. Thus about 20 min of speech is needed. The sentences were selected at random from our office text and somewhat adjusted with the aim of covering all basic sound combinations. We do not expect to change this approach when we increase the vocabulary. The current training text contains only 700 different words. The necessary statistical parameters of the

speaker's personal acoustic model are estimated off-line using the Forward-Backward algorithm [1], [2], [9], as are the 200 prototype vectors. It takes a day or so of elapsed time to sign a new speaker onto the recognizer. This delay will be drastically cut when we implement the training algorithm on our real-time system, but the sign-on will still be time consuming. It is clearly desirable to create a system capable of recognizing without any training any standard American speaker with tolerable accuracy. The performance should improve as the recognizer is used until the best achievable speaker accuracy rate is reached.

As the results of Section III indicate, we have encountered only moderate problems in recognizing spontaneous (as opposed to read) speech. However, with rare exceptions, to achieve satisfactory results, speakers whose native tongue is not English must pronounce the entire vocabulary several times during training. Even then the recognition performance is slightly worse. This we think is due mainly to the inconsistency and occasional self-consciousness of the accented speech. For instance, F. Jelinek occasionally (both during training and recognition) remembers to voice the final "s" in plurals, pronounces the word "of" as 'ov' (rather than his usual 'of'), or attempts to adjust his 'th' sound to make it conform to what he imagines is its correct pronunciation. At other times he tries to remember how he pronounced particular words during training. Such efforts lead mostly to trouble. We do not know yet how to handle these difficulties. It is unfortunately possible that the first commercial dictation recognizers will not perform adequately for some nonnative speakers.

We believe that isolated word dictation will be acceptable for text creation, but the desirability of recognizing continuous speech is self-evident. This is how people speak. Our algorithms do not make any explicit use of the pauses between words and are therefore not limited to isolated word recognition. Unfortunately, even assuming that our methods prove sufficient to overcome problems such as co-articulation, an increase of 6 to 10 times in computing speed will be required to achieve real-time recognition of continuous speech. Our current Speech Development System is not powerful enough. Wanting real-time recognition as we do, we will need to use a new generation of array processors or reluctantly base the development of algorithms on special-purpose hardware. In any case, much progress in work station human factors, noise immunity, vocabulary text coverage, and speaker independence will be necessary before text creation by natural speech becomes everyday reality.

ACKNOWLEDGMENT

The work reported here was carried out individually and collectively by present and past members of the Continuous Speech Recognition Group of the IBM T. J. Watson Research Center. Very significant contributions were made by J. Cocke. All the ideas presented that are of any value resulted from exceptionally effective collaboration by these colleagues.

The present Group members are A. Averbuch, L. Bahl, R. Bakis, P. Brown, A. Cole, G. Daggett, S. Das, K. Davies, S. De Gennaro, P. de Souza, E. Epstein, D. Fraleigh, I. Halpin, F. Jelinek, S. Katz, B. Lewis, H. Meleis, R. Mercer, A. Nadas, D. Nahamoo, M. Picheny, G. Shichman, and P. Spinelli.

The past group members are T. Ancheta, J. K. Baker, J. M. Baker, S. Chang, J. Cohen, P. Cohen, N. R. Dixon, M. Fitzgerald, G. Freeman, R. Freitas, M. Garrett, G. Hatfield, G. Heidorn, C. Junker, P. Loewner, L. Loh, F. Mintzer, J. Mommens, E. M. Mueckstein, L. Mullin, J. Raviv, R. Riekert, J. Robinson, J. Rosenfeld, R. Sadr, H. Silverman, C. Tappert, D. Teaney, and A. Wadia.

We have all benefited from the help given us by our visitors F. Adler, R. Ambrosio, E. Black, S. Bozic, L. Braidia, G. Brown, M. Brown, D. Bustamante, L. Butler, W. Chang, P. Chen, P. Corsi, S. Della Pietra, V. Della Pietra, A. M. Derouault, I. Feerst, R. Findlay, T. Fine, D. Francis, L. Fridman, A. Fronistas, E. Goldwasser, S. Haltsonen, S-S. Huang, T. Im, T. Kaneko, V. Khazatsky, S. Kuo, D. Lee, B. Lotto, J. Lucassen, A. Martelli, M. Morf, A. Nobel, M. Okhochi, A. Pickholtz, J. Pitrelli, S. Pombra, L. Powers, N. C. Rabin, S. Rao, M. Rentmeesters, C. Richardson, M. Roberts, S. Scarci, M. Scott, S. Soudoplatoff, E. Strum, K. Toshioka, and L. Wilcox.

REFERENCES

- [1] F. Jelinek, "Continuous speech recognition by statistical methods," *Proc. IEEE*, vol. 64, no. 4, pp. 532-556, Apr. 1976.
- [2] L. R. Bahl, F. Jelinek, and R. L. Mercer, "A maximum likelihood approach to continuous speech recognition," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-5, no. 2, pp. 179-190, Mar. 1983.
- [3] J. D. Gould, J. Conti, and T. Hovanyecz, "Composing letters with a simulated listening typewriter," *Commun. ACM*, vol. 26, no. 4, pp. 295-308, Apr. 1983.
- [4] E. Goldwasser, an unpublished memorandum, 1980.
- [5] J. R. Cohen, "Application of a sensor-Neural model to speech recognition," to be published.
- [6] H. Abut, R. M. Gray, and G. Rebolledo, "Vector quantization of speech and speech-like waveforms," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-30, no. 3, pp. 423-435, June 1982.
- [7] A. Nadas, R. L. Mercer, L. R. Bahl, R. Bakis, P. S. Cohen, A. G. Cole, F. Jelinek, and B. L. Lewis, "Continuous speech recognition with automatically selected acoustic prototypes obtained by either bootstrapping or clustering," in *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing* (Atlanta, GA, Apr. 1981), pp. 1153-1155.
- [8] J. D. Ferguson, "Hidden Markov analysis: An introduction," in J. D. Ferguson, Ed., *Hidden Markov Models for Speech*. Princeton, NJ: IDA-CRD, Oct. 1980, pp. 8-15.
- [9] L. E. Baum, "An inequality and associated maximization technique in statistical estimation of probabilistic functions of Markov processes," *Inequalities*, vol. 3, no. 1, pp. 1-8, 1972.
- [10] S. Katz, "Recursive M-Gram language model via a smoothing of Turing's formula," a forthcoming paper.
- [11] I. J. Good, *The Estimation of Probabilities: An Essay on Modern Bayesian Methods*. Cambridge, MA: MIT Press, Mar. 1965.
- [12] F. Jelinek, "A fast sequential decoding algorithm using a stack," *IBM J. Res. Devel.*, vol. 13, pp. 675-685, Nov. 1969.
- [13] D. P. Huttenlocher and V. W. Zue, "A model of lexical access from partial phonetic information," in *Proc. ICAASP 84*, vol. 2, pp. 26.4.1-26.4.4, Mar. 1984.
- [14] T. Kaneko and N. R. Dixon, "A hierarchical decision approach to large vocabulary discrete utterance recognition," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-31, no. 5, pp. 1061-1066, Oct. 1983.
- [15] A. Averbuch et al., "A real-time, isolated-word, speech recognition system for dictation transcription," in *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing* (Tampa, FL, Mar. 1985).
- [16] F. Jelinek, R. L. Mercer, L. R. Bahl, and J. K. Baker, "Perplexity—A measure of difficulty of speech recognition tasks," presented at the 94th Meet. Acoustical Society of America, Miami Beach, FL, Dec. 15, 1977.

- [17] M. M. Sondhi and S. E. Levinson, "Computing relative redundancy to measure grammatical constraint in speech recognition tasks," in *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing* (Tulsa, OK, Apr. 1978), pp. 409–412.
- [18] W. N. Francis and H. Kucera, *Frequency Analysis of English Usage*. Boston, MA: Houghton-Mifflin, 1982.
- [19] J. B. Carroll, P. Davies, and B. Richman, *Word Frequency Book*. New York, NY: American Heritage, 1971.
- [20] H. Kucera, personal communication.
- [21] R. L. Mercer, personal communication.
- [22] F. Jelinek, *Probabilistic Information Theory*. New York, NY: McGraw-Hill, 1968.
- [23] C. E. Shannon, "Prediction and entropy of printed English," *Bell Syst. Tech. J.*, vol. 30, pp. 50–64, 1951.
- [24] T. M. Cover and R. C. King, "A convergent gambling estimate of the entropy of English," *IEEE Trans. Informat. Theory*, vol. IT-24, no. 4, pp. 413–420, July 1978.
- [25] S. Della Pietra and V. Della Pietra, personal communication.
- [26] J. D. Gould and S. J. Boies, "Human factors challenges in creating a principal support office system—The speech filing system approach," *ACM Trans. Office Inform. Syst.*, vol. 1, no. 4, pp. 273–298, Oct. 1983.
- [27] ———, "Writing, dictating, and speaking letters," *Science*, vol. 201, pp. 1145–1147, 1978.
- [28] L. R. Bahl and F. Jelinek, "Decoding for channels with insertions, deletions, and substitutions with applications to speech recognition," *IEEE Trans. Informat. Theory*, vol. IT-21, no. 4, pp. 404–411, July 1975.



ed to: Klarquist Sparkman LLP. Downloaded on June 19, 2025 at 17:00:02 UTC from IEEE
The Speed Development System workstation; the
microphone is on the right of the screen.