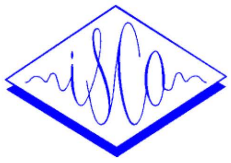

European Conference on
Speech Technology
Edinburgh, Scotland, UK
September 1987

DEMISYLLABLE BASED SPANISH NUMBER RECOGNITION EXPERIMENTS

José B. Mariño, C. Nadeu, E. Lleida *

ABSTRACT

The main features of our demisyllable based continuous speech recognition system (RAMSES) are showed. Special attention is paid to demisyllable definition and the syntactic constraints used with the dynamic programming algorithm; particularly, the grammatical inference method is described. Recognition scores are not included but they could be provided on request.

INTRODUCTION

Nowadays, syllables and demisyllables are under active investigation as phonetic unit for continous speech recognition systems; reasons for that are well known. As far as the Spanish language is concerned, the large number of syllables and the relatively small inventory of Spanish demisyllables (about 500 initial ones and 160 final ones) are two good reasons to prefer the demisyllable as recognition unit.

By using tools from connected speech recognition it has been built in our laboratory two alternative versions for a continous speech recognition system. The first one uses both templates and the time-warping approach and it is essentially speaker dependent; the second one is based on hidden Markov Models, and the decision is made from the maximum likelihood path. In both versions, the one-stage algorithm (ref. 1) performs the dynamic programming optimization, and the set of possible sentences is specified to the recognition system by syntactic constraints.

To test the performance of this recognition system, two vocabulary sets have been chosen. The first one is the Spanish number set from zero to one thousand; the telephone numbers pronounced in a Spanish natural way (3/21/69/20, three/twenty one/sixty nine/twenty) constitute the second set.

DEMISYLLABLE DEFINITION AND REFERENCE EXTRACTION

Every possible syllable is divided by the strong vowel into an initial demisyllable (/ids/-) and a final demisyllable; the cutting is made in an asymmetric way, like in ref. 2.

* Dpto. Teoría de la Señal y Comunicaciones. Universidad Politécnica de Catalunya. Apdo. 30.002, 08080 Barcelona, Spain.

This work is supported by the CAICYT grant number 21096/84.

This cutting allows the final demisyllable to account for the stress of the syllable; accordingly, we distinguish between stressed final demisyllables (-/sfds/) and unstressed final demisyllables (-/fds/).

The references for the demisyllables are taken from nonsense words, defined trying to provide a neutral phonetic context to each demisyllable. In general, initial demisyllables are collect from words like /ids/-pa (i.e. the word "cuapa" is used to obtain the demisyllable /cua/-); however, when the demisyllable can not be at the begining of the word, the following alternative construction is considered: pa/ids/-pa (i.e. paropa); in these nonsense words the stress is always made in the demisyllable part. Final demisyllables are taken from the word tap-/sfds/ or tap-/fds/ and the stress is made according to the stressed or nonstressed character of the demisyllable. In several special cases, the sound of the final demisyllable depends on the next syllable; in a such situation, the word used to extract the demisyllable is properly modified (i.e. we use "posba" when the realization of the phoneme s is voiced).

In the recognition process, the demisyllable reference will always be identical indepently from the word or phase where it appears. Possible alophonic or pronouncing alternatives for words or sentences are incorporated as different entries of the vocabulary set. The whole of demisyllables references is intended to cover every sound all over the vocabulary. For the number recognition experiments we use 39 initial demisyllables, 8 unstressed and 15 stressed final demisyllables.

GRAMMATICAL INFERENCE

Every word or phrase (number for our experiments) constitutes a string of demisyllables; legal strings can be described in terms of a finite state grammar (ref. 1), where each state corresponds to a demisyllable. In general, it could be necessary to use several different states for the same demisyllable in order to account for the different contexts in which the demisyllable may appear. Hereafter, we will call "copies" to these alternative states. A finite state grammar can be determined by the set of predecessors of every state. In the following paragraphs the algorithm we have used to inference the grammar, is outlined.

STEP 1: The legal strings of the vocabulary are processed sequentially in order to establish the necessary copies for every demisyllable and the corresponding sets of predecessors P and successors S; the latter is used only for purposes of the inference algorithm. The processing rules are:

rule 1: If a demisyllable appears several times in a string, every occurrence must be associated to a different copy.

rule 2: A copy may be shared by several strings, only if every string exhibits the same substring from the demisyllable copy to a specified ending of the string (i.e. the shared substring must always be predecessor or always be successor of the demisyllable).

Once the copies of every demisyllable are determined, we can inmediately obtain the sets of predecessor and successors.

STEP 2: In general, the application of step 1 overestimates the number of necessary copies for each demisyllable; the following lemma bounds this number.

lemma: Let be n_p the number of different predecessors of a demisyllable and n_s the number of different sucessors. If n is the lowest of n_p and n_s, it is possible to redefine the copies in such a way that their number will not exceed n.

To achieve this goal it is enough to use the following:

rule 3: Let be n equal to n_p (n_s); each new copy is defined with only one predecessor (successor) and with a set of successors (predecessors) determined as the union of the successor (predecessor) sets of every old copy that exhibits the predecessor (successor) defining the new copy.

In several cases, further reduction of copies can be achieved by using the following:

rule 4: Two copies that exhibit the same set of predecessors (successors) may be merged in only one copy, which successor (predecessor) set will be the union of sets of the original copies.

The modification of copy definition make us to update the P and S sets of all demisyllable copies of the grammar. The algorithm ends when no reduction can be accomplished. Figure 1 illustrates the exposed grammatical inference procedure.

DEMISYLLABLE LENGTH VARIABILITY CONSIDERATION

One of the main sources of recognition errors is the variability of syllable duration in natural or continuous speech. Although the dynamic programming algorithm provides a reasonable treatement of this problem, we have implemented in our system two aditional strategies to account for it:

a) considering various references for every final demisyllable;

b) allowing the optimum path of the dynamic programming algorithm to skip several frames at the end of an initial demisyllable and at the begining of the following final demisyllable.

RESULTS

Unexpected problems have precluded the generation of the data base necessary to evaluate the performance of our recognition system. As a consequence, at present we can not afford significant results. At the conference we will show the recognition scores for one speaker provided by the version with templates; the results of our preliminary experiments are encouraging, because testing 20 numbers from zero to one hundred no error has taken place. The final results will be provided on request.

REFERENCES

/1/ H. Ney.
"The use of a One-stage dynamic programming algorithm for connected word recognition".
IEEE Trans. ASSP-32, no. 2, pp. 263-271 (1984).

/2/ A.E. Rosenberg et al.
"Demisyllable-based isolated word recognition system".
IEEE Trans. ASSP-31, no. 3, pp. 713-725 (1983).

VOCABULARY LEGAL STRINGS: abc, abd, eabc, fabc, gabd.

STEP 1

	P	copy	S
. a^1 b^1 c^1 /	.	a^1	b^1
. a^1 b^1 d^1 /	e^1	a^2	b^2
. e^1 a^2 b^2 c^1 /	g^1	a^3	b^3
. f^1 a^2 b^2 c^1 /	a^1	b^1	c^1, d^1
. g^1 a^3 b^3 d^1 /	a^2	b^2	c^1
	a^3	b^3	d^1

STEP 2

P	copy	S
.	a^1	b^1, b^2
e^1	a^2	b^1
g^1	a^3	b^2
a^1, a^2	b^1	c^1
a^1, a^3	b^2	d^1

first iteration

P	copy	S
., e^1	a^1	b^1
., g^1	a^2	b^2
a^1	b^1	c^1
a^2	b^2	d^1

second iteration

Figure 1. Example of grammatical inference, (.=begin, /=end, P=predecessor set, S=successor set).