SELECTION OF SPEECH RECOGNITION FEATURES USING A GENETIC ALGORITHM

L.A. Smith

Department of Computer Science
University of Waikato

ABSTRACT - A genetic algorithm was used to select a reduced set of features for a commercial speech recognizer. The algorithm was applied using a test set of 20 words spoken over the telephone by 22 speakers, with the recognition accuracy determining the fitness of competing feature sets. The recognizer correctly recognized 95.1% of the utterances using all 19 of the candidate features. The genetic algorithm found 2 sets of 16 features that recognized 95.3%, and a set of 17 features that recognized 95.4% of the test utterances. A second experiment found a set of feature weights, using all 19 candidates, with which the recognizer correctly identified 95.8% of the test utterances.

INTRODUCTION

Selecting a set of recognition features is a common problem in pattern recognition. The only known way to select the optimal set, resulting in the best possible recognition accuracy, is by trying all combinations of candidate features. Because exhaustive search is computationally intractable even for small sets of features, various means have been developed for choosing features, even though they don't guarantee an optimal set. Most of these methods are based on add-on or knock-out techniques, or on dynamic programming, but probabilistic methods have recently been applied to the problem with some success (Davis, 1987). The experiments described here apply one such technique in an attempt to reduce the number of features used by a commercial speech recognizer, thus enabling the recognizer to store a larger vocabulary.

Genetic algorithms are machine learning and optimization techniques based on biological concepts (Holland, 1975). Pioneered by John Holland, they attempt to mimic natural selection by using a population of competing solutions which evolve over a series of generations. The use of a population allows, effectively, a parallel search of the solution space and helps to avoid local minima. In an iterative process, generation *n* produces generation *n* +1 by reproduction among population members. Fitness, as determined by some objective measure, determines which population members are favored for reproduction. There are many variations on the genetic algorithm paradigm -- populations may or may not overlap and may or may not change in size from generation to generation; mutations may be used to help avoid local minima; limited "resources" may be shared among population members, which may develop different "species." The model used in the work presented here is a simple one based on Goldberg (1989); it is described below.

SPEECH RECOGNIZER

The recognizer is a form of nearest neighbor classifier developed for speaker independent recognition of isolated words or phrases (Smith *et al.*, 1990). An input word or phrase (the test token) is compared with a stored vocabulary (reference tokens) and is labeled as the closest matching vocabulary entry. The recognizer time normalizes speech based on prosodic and transitional cues (Scott *et al.*, 1989) so both test and reference tokens consist of 32 frames, where each frame is a vector of 19 feature coefficients (the term "feature", as used in pattern recognition, is any measure that helps distinguish one pattern from another; LPC, FFT, and cepstral coefficients are all "features" in this sense). The features referred to here are processed in ways beyond the scope of this paper, but they may be thought of as being separated into 5 bands. Three of the bands carry 4 autocorrelation coefficients each, and relate (roughly) to high, mid, and low frequencies. A fourth band is made up of averages taken over each of the first 3 bands, and 1 average taken over all 3 of the bands, while the fifth band carries 3 amplitude measures. All of these "features" are normalized across the utterance, with values ranging from 0 to 15.

Template Generation

During training, the recognizer creates a single composite template for each word in its vocabulary. The template for a given word or phrase is formed by calculating, for each feature coefficient, the mean and standard deviation across all training utterances for that word or phrase. Each template may be viewed as an n-dimensional hypersphere, where n is the number of template features, and where the boundaries are determined by the feature means and standard deviations.

Matching

During recognition, the template representing the input, or test, utterance is matched against each of the templates in the vocabulary. The score for each template is calculated by:

$$\sum_{i=1}^{V} \sum_{j=1}^{F} \left| \begin{array}{ll} d & (d > 0) \\ 0 & (d <= 0) \end{array} \right. ,$$

$$d = |v_{ij} - t_{ij}| - sd_{ij}$$

where V is the number of feature vectors in each template (32, in this case), F is the number of coefficients in each feature vector (19), v is the vocabulary template, t is the test template, and sd is the set of standard deviation vectors associated with the vocabulary template. The formula is similar to the L1, or city block, distance measure, but is modified by subtracting, for each vector coefficient, its standard deviation; if the result is greater than 0, then it is added to the cumulative score. The test token is classified according to the label of the vocabulary template with the lowest match score.

VOCABULARY

The vocabulary used in this experiment was the digits (zero - nine) and 10 control words: enter, erase, help, yes, no, go, repeat, rubout, start, and stop. Each word was uttered over the telephone by 22 speakers (11 male and 11 female). Speakers were asked to say each word 3 times; some speakers did not say each word 3 times or, occasionally, said a word outside the vocabulary rather than the correct word. Furthermore, obvious errors by the automatic endpoint routine were discarded as being irrelevant to the problem of improving the feature set used for recognition. The speech data base used for these experiments, then, consisted of 1001 utterances out of the 1452 possible.

Because the volume of data available was only marginally adequate for creating speaker independent vocabularies, 22 separate vocabularies were created, with each speaker's utterances being tested using a vocabulary created over the other 21 speakers.

GENETIC ALGORITHM

The genetic algorithm used here is based on Goldberg (1989). Feature weightings are coded as a 19 bit string, corresponding to vector coefficients numbered from 0 to 18. If a bit is on (value 1), then the corresponding feature is used in the recognition match; if the bit is off (0), then the feature is not used in the match. Raw fitness of the individual string is simply the percentage of utterances correctly recognized using the features specified by the string. The raw fitness is linearly scaled, after Goldberg (1989), using a formula of the form f' = af + b, where f' is the scaled fitness, f is the raw fitness, and a and b are coefficients chosen such that the average fitness is maintained across scaling. As an example, in generation 0 of the first experiment, string 0000011111110100110 used features 1, 2, 5, and 7 through 13 (the rightmost bit is position 0), and correctly recognized 890 out of 1001 utterances for a raw fitness value of 0.889, which was scaled to 0.965. Scaling the population fitness helps prevent the algorithm from settling on locally optimal results in 2 ways. Early in the run, it helps maintain population diversity by keeping a few individuals from crowding others out ("premature convergence"). Late in the run, when almost all individuals are successful, scaling helps distinguish those individuals that are slightly more fit than the others.

New generations are created by the 3 basic genetic operators: selection, crossover, and mutation. Generations do not overlap, meaning that each generation is, in principle, made up of new individuals with none carried over from the previous generation. As will be seen, however, some of the members of the new generation will be copies of their "parents."

Selection is based on fitness, with each member of the population expected to contribute a number of offspring to the new generation equal to its scaled fitness. The member of generation 0 described above, for example, is expected to contribute 0.965 offspring to generation 1; viewed another way, 0.965 is the probability that the individual will be selected for reproduction. Selection is by the stochastic remainder method, in which an individual with a scaled fitness of 1.5 will surely contribute 1 offspring; .5 is the probability that the individual will contribute an additional offspring.

Crossover, the main operator for creating new strings from pairs of selected individuals, incorporates randomness in 2 ways. First, crossover only occurs with a given probability (0.5 in these experiments). When crossover does not occur, both selected individuals are copied into the new generation without change. Second, when crossover does occur, the crossover point for the two individuals is randomly chosen, and the 2 offspring are built by concatenating the pieces of the 2 parents. For example, the strings 11111 and 00000 (length 5 for illustration), when crossed at position 3 produce the 2 offspring 11000 and 00111. In order to implement nonoverlapping populations which are the same size in each generation, each pair of selected individuals always produces 2 offspring.

Mutation is a genetic operator designed to introduce a degree of random noise into the procedure by occasionally changing the value of 1 bit, chosen at random, from an individual. The purpose is, again, to help avoid locally optimal solutions. Mutation is usually applied with some low probability, but was used somewhat differently in this experiment. Here, mutation was always applied to the second offspring when parents selected for crossover were exact copies of one another. The reason for applying mutation in this manner is the desire to introduce a new string whenever crossover occurs; crossing 2 copies of the same string will only produce 2 exact copies of the string. The mutation operator selects, at random, one bit of the chosen string and inverts it. For example, the string 01010, mutated at position 2, would result in 01110.

Generation 0 is initialized by randomly setting all individuals bits to 1 or 0 with equal probability, with the constraint that there must be no matching individuals in the first generation -- that is, generation 0 is guaranteed to consist of 30 unique individuals.

RESULTS

Figure 1 shows the maximum, minimum, and mean recognition accuracies over 36 generations. The experiment was continued to 50 generations, but no significant evolution occurred after generation 30; the mean hovered between 95.1% and 95.2% while the maximum remained at 95.4%. It is obvious that the algorithm very quickly dispenses with poor encodings and that maximum, minimum, and mean quickly converge. Figure 2 shows the maximum scores with the scale expanded.

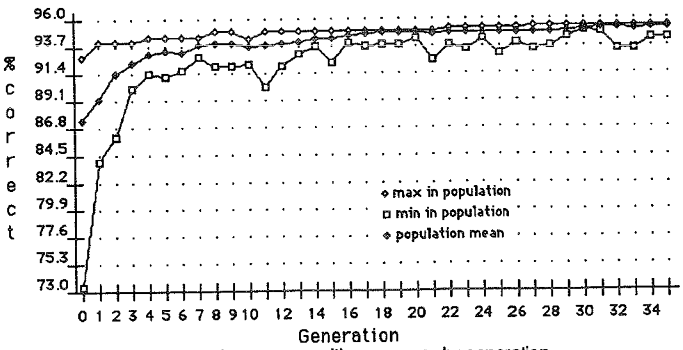

Figure 1. Maximum, minimum, and mean recognition accuracy by generation.

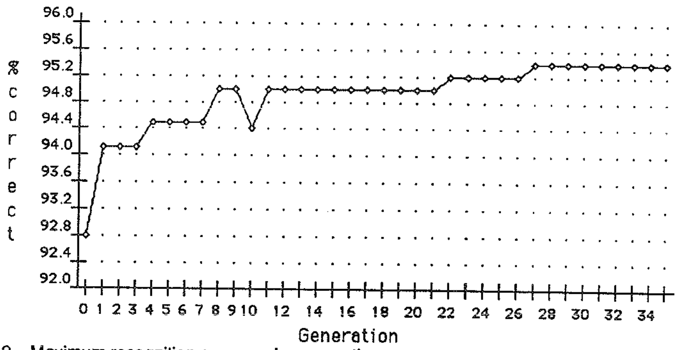

Figure 2. Maximum recognition accuracy by generation.

The goal of the experiment was to reduce the set of features without seriously degrading the accuracy of the recognizer. The baseline score (recognition accuracy using all features) was 95.1%; surprisingly, the genetic algorithm found sets of features that actually improved the recognition score over the test data. The feature set achieving the top score, 95.4%, used 17 features. In addition, the genetic algorithm found 2 different sets of 16 features that recognized 95.3% of the test utterances and a set of 18 features that scored 95.2%. All 3 of the reduced feature sets discarded 1 autocorrelation coefficient in the low frequency band, as well as the average coefficient value of the mid frequency band. One of the 16 feature solutions discarded the frame average value, while the other did away with a logarithmic amplitude measure.

Figure 3 illustrates the operation of the algorithm, showing the number of unique individuals in each generation.

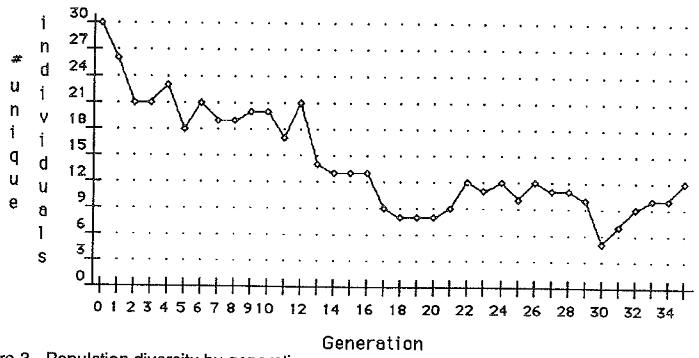

Figure 3. Population diversity by generation.

Generation 0 begins with 30 unique individuals. As the more successful individuals begin "taking over" the population, the number of unique individuals declines, dropping as low as 5 before recovering due to forced mutation.

The propensity of a few individuals to take over the population is both the strength and the weakness of genetic algorithms -- the algorithm works by convergence toward better and better solutions, but, as the algorithm proceeds, more and more overhead is devoted to simply keeping the good solutions rather than to searching out new solutions. Figure 4 illustrates the rate at which new solutions were tried, showing the number of unique individuals in each generation which were not seen in any previous generation.

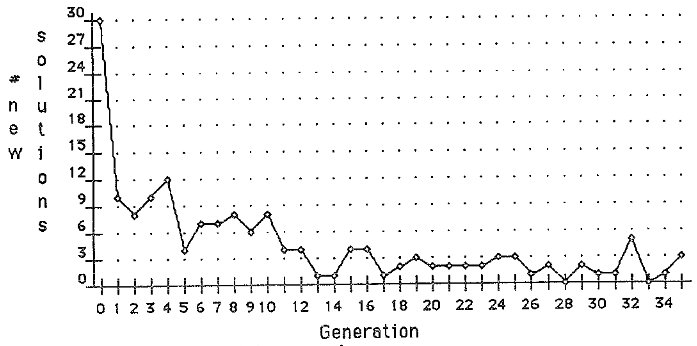

Figure 4. Number of new individuals by generation.

Generation 13 tried only 1 new solution, and 2 later generations had no new solutions to try. Of course, it should be noted that the generations which tried no new solutions occurred after the best solutions had appeared, and it may be that it was not possible to make any progress after that point. Still, consideration of figure 4 suggests that some measures might be taken to force the algorithm to break new ground. This might be as simple as using a larger population, or perhaps memory might be used to limit reincarnation.

A second experiment, using a similar genetic algorithm to determine feature weights, did not discard any features, but found sets of weights that performed even better than the binary weights found by the first experiment, with a top recognition score of 95.8% over the same speech data. The weights were allowed to vary from 0 to 3. The most successful solutions (95.7 - 95.8%) assigned the highest weights to high frequency autocorrelation coefficients and to amplitude measures.

CONCLUSIONS

The goal of the experiment was purely practical -- to find a reduced set of features for speaker independent word recognition. The genetic algorithm met the goal, finding 2 sets of 16 and 1 set of 17 features which performed slightly better than the original set of 19 features. Reduction of features from 19 to 16, along with some minor modifications in the way vocabularies are stored, will enable the recognizer to expand its vocabulary from 160 to 200 words or phrases.

A second experiment, which allowed feature weights to vary from 0 to 3, confirmed the finding of the first experiment that, for this recognizer, the most significant information is carried by high frequency autocorrelation coefficients. In addition, the second experiment placed increased weight on amplitude measures.

The utility of genetic algorithms in selecting sets of features and weightings is unquestionable. It seems, considering the reluctance of the algorithm to break new ground (figure 4), that further experiments should use larger populations; furthermore, in order to determine a set of features and weights which are universally applicable, the genetic algorithm must be applied to larger and more diverse sets of words and speakers.

REFERENCES

Davis, L. ed. (1987) *Genetic Algorithms and Simulated Annealing,* (Pitman: London).

Goldberg, D. E. (1989) *Genetic Algorithms in Search, Optimization, and Machine Learning,* (Addison-Wesley: Reading, MA).

Holland, J.H. (1975) *Adaptation in Natural and Artificial Systems,* (University of Michigan Press: Ann Arbor, MI).

Smith, L.A., Scott, B.L., Lin, L.S. and Newell, J.M. (1990) *Template adaptation in a hypersphere word classifier,* Proc. IEEE Int. Conf. Acoust., Speech, and Signal Processing.

Scott, B.L., Lin, L.S., Newell, J.M. and Smith,L.A. (1989) *Improvements to a speaker independent algorithm,* J. Acoust. Soc. Am. **85,** S56.