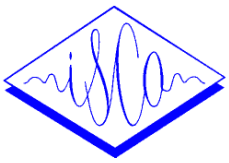

2nd International Conference on
Spoken Language Processing (ICSLP 92)
Banff, Alberta, Canada
October 12-16, 1992

SECONDARY PROCESSING USING SPEECH SEGMENTS FOR AN HMM WORD SPOTTING SYSTEM

Herbert Gish *Kenney Ng* *J. Robin Rohlicek*

BBN Systems and Technologies
10 Moulton Street 6/4b
Cambridge MA 02138 USA

ABSTRACT

In this paper we describe a secondary processing algorithm designed to improve word spotting performance by reducing the sensitivity of a primary HMM word spotting system to false alarms while maintaining high recognition accuracy. The concept behind the algorithm is to rescore the putative events hypothesized by the primary word spotter by generating a "secondary" score, which is designed to discriminate between true keyword occurrences and false alarms, and then combining it with the original HMM score. The secondary processor makes use of variable duration speech segments produced by a deterministic acoustic segmentation algorithm. The secondary processing algorithm is evaluated on a keyword spotting task using the Road Rally Database. Performance is shown to improve significantly over that of the baseline word spotting system with the greatest improvements taking place at low false alarm rates.

1. INTRODUCTION

The putative events generated by our HMM continuous speech keyword spotting system [3] consist of true keyword hits and false alarm events. Each putative event has a posterior probability score which indicates its "likelihood" of being a true keyword hit. This HMM score is currently the only information available in ranking the putative events to try to distinguish true hits from false alarms. As a result, false alarms that score well and true hits that score poorly limit word spotting performance. One approach to try to compensate for these HMM scoring anomalies is to incorporate additional information that can separate true keywords and high scoring false alarms. Generating this additional information is the goal of the secondary processing system. Putative events produced by the primary word spotter are used to train keyword specific segmental models to discriminate between true keywords and false alarms. These models are then used to generate "secondary" scores for new putative events which are then combined with the "primary" HMM scores to perform rescoring.

An overview of the secondary processing system is shown in the block diagram in Figure 1. Feature analysis and acoustic segmentation are preprocessing stages that are performed on both the training and testing data. The resulting speech features and segment boundaries for the training data are then used, in conjunction with the marked training events, to extract out the true keyword and false alarm segments for training the segment models in the secondary processor. Once trained, these models are then used to evaluate and rescore putative events from the new testing data. In the next Section, the acoustic segmentation algorithm used to generate the speech segments is described. Section 3 then discusses how these speech segments are modeled and used to rescore new putative events. And finally, some experimental results are presented in Section 4.

2. ACOUSTIC SEGMENTATION

Unlike the primary HMM word spotter, which uses frames of spectral features generated at constant intervals as the basic unit of speech, the secondary processor uses variable duration acoustic segments as the basic speech unit. The segments are generated using a data driven acoustic segmentation algorithm based in part on the work described in [1]. The segmentation is performed as a preprocessing step and results in a deterministic set of segment boundaries. The goal of the segmentation is the grouping together of acoustically similar adjacent frames into segmental units. The motivation for a segmental approach is that by working with a collection of frames we can make better discriminations than if we worked with individual frames.

The first step in the segmentation process is the computation of a discontinuity score as a function of time based on a similarity measure between neighboring frames of spectral features. The discontinuity score is computed as the likelihood ratio:

$$L(t) = \frac{p(f(x(t), x(t')) \mid \text{frames from same segment})}{p(f(x(t), x(t')) \mid \text{frames from different segments})}, \quad (1)$$

where $f(x(t), x(t'))$ is a statistic that measures the similarity between the two neighboring speech frames, $x(t)$ and $x(t')$.

In the current segmentation algorithm, discontinuity scores using three different similarity statistics are computed in an attempt to incorporate multiple sources of information into the boundary decision making process. The first similarity statistic compares spectral features across the entire frequency range; the second examines features in narrower overlapping frequency bands; and the third looks at the ratio of low to high frequency energy. The bottom three plots in Figure 2 show the three discontinuity scores generated for the speech utterance "a secondary dirt track." Notice that the discontinuity scores are high at points of large spectral change and small in regions of steady state.

The discontinuity scores are next run through a local maxima and thresholding procedure to generate three sets of initial segment boundaries, one for each discontinuity score. These initial boundaries are then combined using a simple voting scheme to produce an intermediate set of boundaries. A final agglomeration step then merges similar adjacent segments and generates the final segmentation. For the example in Figure 2, the resulting segment boundaries are shown overlaid on the spectrogram.

After the segments are formed, a set of sufficient statistics is computed for each segment. These statistics consist of a trajectory of the speech features, the covariance around that trajectory, and the number of frames in the segment. In the experiments described in this paper, a constant mean trajectory is used, although a more general trajectory model of the features is possible.

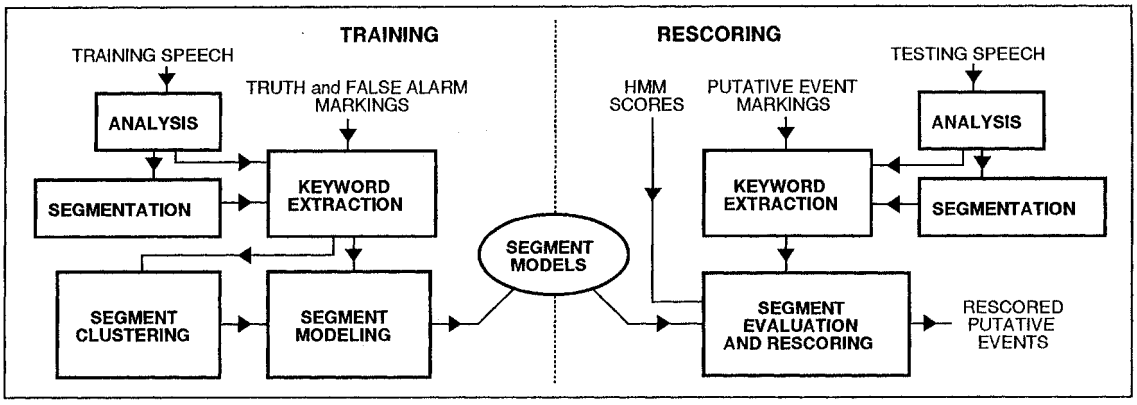

Figure 1: Block Diagram of the Secondary Processor Training and Rescoring Systems.

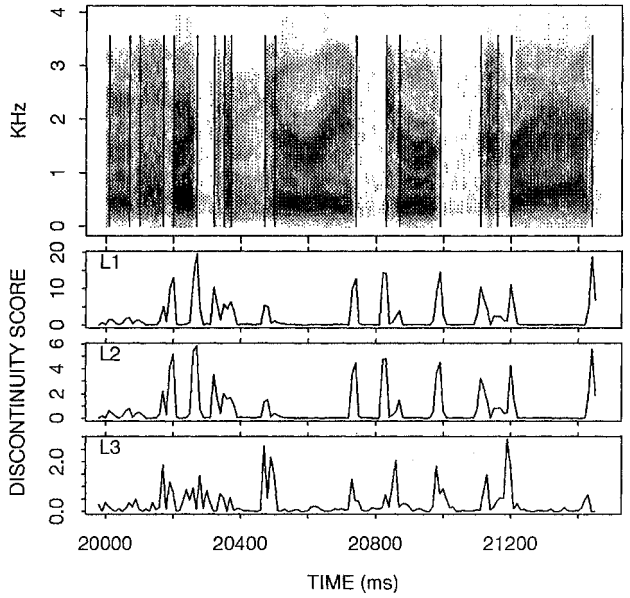

Figure 2: Illustration of the Discontinuity Scores and Segment Boundaries Generated by the Segmentation Algorithm.

3. SEGMENTAL PROCESSING

After representing the speech by variable duration acoustic segments, the next step is the modeling of these segments for use in the secondary processing algorithm.

In the secondary processor, two mixture models for each keyword are created using labeled putative events from the training set. One mixture model is for the segments from true keyword occurrences, and the other is for the segments from false alarm events. The two mixture models are trained separately using different portions of the training set. The truth model is trained using only the true occurrences, while the false alarm model is trained using only the false alarms. In addition to modeling the characteristics of the segments, first-order transition probabilities between the segments, and word duration probabilities are also estimated for the true events and false alarms. The goal in training the secondary processor is to build models that can be used to discriminate true keyword events from false alarms.

A detailed description of the training process for the segment models is given in the next two sections. Although we focus on the training of the true keyword model, the false alarm model is also trained using the same set of procedures.

3.1. Segment Clustering

The first step in the training of the true keyword model is the initialization of the segment mixture model. This initialization is accomplished by first forming a distance matrix between all segments from the true keyword occurrences in the training set, and then using it to cluster the segments. The clustered segments are then used to calculate the mean vectors and covariance matrices needed to initialize the components of the mixture model.

The formation of the distance matrix is based on a generalized likelihood ratio test between pairs of segments. Given two segments, we form the following hypothesis test:

- H_o: the segments were generated by the same model, and
- H_1: the segments were generated by different models.

Gaussian models are used in forming this hypothesis test, and the test statistic depends only on the sufficient statistics associated with each of the segments. If we let $\lambda(r, s)$ denote the likelihood ratio for the test applied to segments r and s, then the distance between the segments is defined as:

$$d(r, s) = -\log \lambda(r, s) . \quad (2)$$

This equation is discussed in detail in [2], under the assumption that the probability models are Gaussian. Refer to that paper for details about the distance expression.

Once calculated, the distance matrix serves as input to an agglomerative, hierarchical clustering procedure from which N clusters of segments are automatically generated. N is an indication of the number of different types of segments in the keyword and is used to determine the number of components to use in the keyword mixture model. Each segment cluster is then used to initialize the corresponding component in the mixture model.

3.2. Segment Modeling

This N component mixture model is a model of the spectral distributions of the different types or classes of segments in the true occurrences of the keyword. It can be written in the form:

$$p(s) = \sum_{i=1}^{N} \alpha_i p_i(s; \mu_i, \Sigma_i) , \quad (3)$$

where α_i is the *a prior* probability that a segment is from segment class i. The observation s *is a segment* and is represented by $(\bar{s}, S, n)$: the sample mean of the features, the sample covariance, and the number of frames in the segment, respectively. Recall that this triplet is the set of sufficient statistics for the segment.

The probability of a segment is the probability of the frames in the segment, and is simply a function of the sufficient statistics of the segment. Employing a Gaussian model for the frames in the segment and assuming the frames in a segment are statistically independent gives for component i:

$$p_i(s; \mu_i, \Sigma_i) = (2\pi)^{-\frac{dn}{2}} |\Sigma_i|^{-\frac{d}{2}} \times \exp\left[-\frac{n}{2}\mathrm{tr}\left[\Sigma_i^{-1}S\right] - \frac{n}{2}(\bar{s}-\mu_i)'\Sigma_i^{-1}(\bar{s}-\mu_i)\right]. \quad (4)$$

The mean, μ_i, and covariance, Σ_i, for the different mixture components are the parameters initialized from the segment clustering procedure. Once the mixture model is initialized, it is trained by iterating over all the segments from the true keyword occurrences in the training set using the Expectation-Maximization (EM) algorithm. Currently, we run through 10 iterations of the EM algorithm to train the mixture model.

As mentioned above, in addition to modeling the characteristics of the segments, first-order transition probabilities between the segments, and keyword duration probabilities are also estimated. This results in a true keyword model that is a composite of a segment mixture model, a segment transition model, and a keyword duration model. A composite model for the false alarms is also constructed following a similar set of procedures.

3.3. Segment Evaluation and Rescoring

Once the true and false alarm composite models in the secondary processor for a particular keyword are trained, they can be used to rescore new putative events of that keyword. The first step in the rescoring procedure is the computation of a secondary score, S_s, for each putative event, E. This score is computed as the log likelihood ratio between the probability that the putative event came from the truth model, and the probability that the putative event came from the false alarm model:

$$S_s = -\log\left[\frac{S_{\text{true}}}{S_{\text{false}}}\right] = -\log\left[\frac{p(E \mid \text{truth model})}{p(E \mid \text{false model})}\right]. \quad (5)$$

Both the S_{true} and S_{false} probability scores are computed using a dynamic programming (DP) search that finds the most probable (highest scoring) path through the segments of the putative event given the truth and false alarm composite models, respectively. The DP search takes into account the likelihood of each segment in the putative event belonging to each class in the mixture model, the transition probabilities from one segment class to another, and the duration model of the keyword.

After the secondary scores are generated, they are combined with the HMM scores, S_h, from the primary word spotter to produce a new set of "combined" scores for the putative events:

$$S_c = \tilde{S}_h + \tilde{S}_s = \frac{S_h}{S_h^{(90)} - S_h^{(10)}} + \frac{S_s}{S_s^{(90)} - S_s^{(10)}}, \quad (6)$$

where $S^{(n)}$ is the n^{th} percentile value of the set of scores, S.

For simplicity, a straightforward sum is currently used to combine the two scores, although a more complex and potentially better performing combination of the scores could be trained. The simple inter-percentile normalization in Equation 6 is used to make the HMM and secondary scores compatible with each other so that their sum is meaningful. An alternative scheme is to normalize the scores by their variance. The resulting new combined scores, S_c, are the final product of the secondary processing algorithm. These scores are then used to reorder the set of putative events.

4. EXPERIMENTS

4.1. Experimental Paradigm

The secondary processing algorithm is evaluated on a twenty keyword spotting task using the Road Rally Database. The four data sets used in the experiments are described in Table 1. The marked keyword occurrences from the read paragraph speech of the 28 male speakers in the Waterloo portion of the database (NIST Training Set) are used to train the primary HMM word spotter [3]. There is a total of 2796 true keyword occurrences in the 55 minutes of speech in this part of the database.

All 32 male conversational speech files from the Stonehenge portion of the database are then processed with the trained primary word spotter in order to generate putative events for training and testing the secondary processor. Putative events from the 12-speaker Stonehenge Training Set (NIST Augmented Training Set) are used to train the secondary processor. The remaining 20 speakers make up two separate test sets: the 10-speaker Stonehenge Test Set 1 (NIST Standard Test Set), and the 10-speaker Stonehenge Test Set 2 (NIST Augmented Test Set). Putative events from these two test sets are then rescored using the trained secondary processor.

Although there are twenty keywords in this spotting task, only the eleven keywords listed in Table 2 are rescored with the secondary processor. The reason not all twenty of the keywords are rescored is because the primary HMM word spotter did not produce any false alarm putative events for nine of the keywords in the 12-speaker Stonehenge Training Set. As a result, there was not enough data to train the secondary processor models for those words. In the set of eleven keywords that are rescored, notice that all four of the monosyllabic keywords, which are among the most difficult to recognize, are included.

chester	conway	interstate	look
middleton	minus	mountain	road
thicket	track	want	

Table 2: List of the Eleven Keywords Processed with the Secondary Processing Algorithm.

The input features used by both the primary HMM word spotter and the secondary processor are normalized mel-warped LPC cepstra ($nc_1, \ldots, nc_{10}$), and cepstral derivatives ($dc_0, \ldots, dc_{10}$) computed from 300-3300 Hz bandlimited speech waveform signals. The cepstra are normalized to reduce their variability due to channel and speaker differences [3]. The features used in the acoustic segmentation algorithm are 16 mel-spaced samples of a log LPC power spectrum (ie., spectral bands).

4.2. Experimental Results

Figure 3 shows the ROC (Receiver Operating Curve) performance curves of the primary HMM word spotter with and without secondary processing for the 10-speaker Stonehenge Test Set 1. Composite detection rate in percent for all twenty keywords (P_d) is plotted against false alarm rate in false alarms per keyword per hour (fa/kw/hr). We note that word spotting performance after rescoring the putative events with the secondary processor (○)

Data Set Label	Duration	# Speakers	Speaker IDs	Purpose
Waterloo males	55 min.	28	wm29–wm56	HMM Training
Stonehenge Training males	34 min.	12	sm03c–sm10c, sm13c–sm16c	Secondary Training
Stonehenge Test-1 males	26 min.	10	sm33c–sm41c, sm43c	Test Set 1
Stonehenge Test-2 males	25 min.	10	sm49c–sm57c, sm59c	Test Set 2

Table 1: Description of the Road Rally Data Sets Used in the Secondary Processing Experiments.

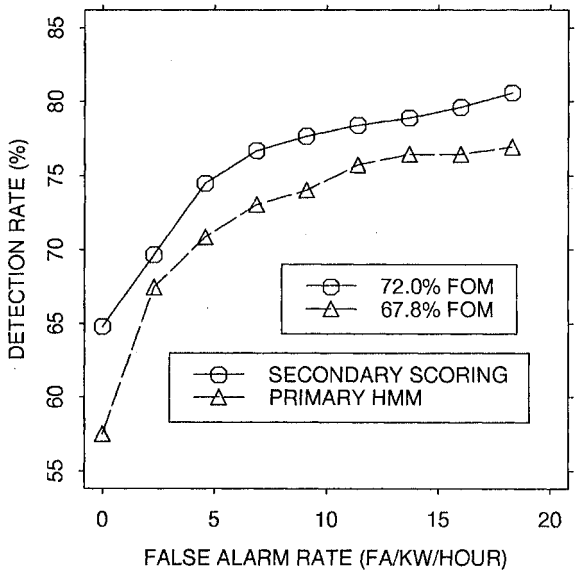

Figure 3: ROC Performance Curves for the 10 Speaker Stonehenge Test Set 1 (NIST Standard Test Set).

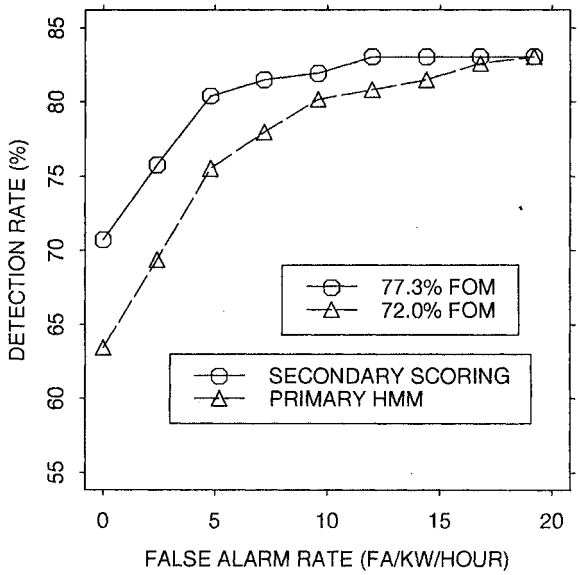

Figure 4: ROC Performance Curves for the 10 Speaker Stonehenge Test Set 2 (NIST Augmented Test Set).

is better than the performance obtained when using just the primary HMM scores alone (△). At 0 fa/kw/hr, P_d improves from 57.5% to 64.8% and in terms of Figure of Merit (FOM), which is the average P_d from 0 to 10 fa/kw/hr, performance improves from 67.8% to 72.0% FOM.

On the 10-speaker Stonehenge Test Set 2, similar performance improvements can be seen as shown in Figure 4. Once again, ROC performance curves are plotted for the primary HMM word spotter with (○) and without (△) secondary processing. For this test set, P_d at 0 fa/kw/hr improves from 63.4% to 70.7%, and FOM improves from 72.0% to 77.3% FOM when secondary processing is performed.

For both test sets, we see a significant performance improvement when putative events hypothesized by the primary HMM word spotter are rescored using the secondary processing algorithm. Although performance improves over a wide range of false alarm rates, the largest improvements take place at low false alarm rates, which is the region of most interest.

5. SUMMARY

In this paper, we describe a secondary processing algorithm that rescores putative events hypothesized by a primary word spotter in an attempt to separate true events from false alarms to improve word spotting performance. The secondary processing algorithm makes use of variable duration speech segments produced by a deterministic acoustic segmentation algorithm.

The secondary processor generates supplemental (secondary) scores for the putative events and combines them with the original HMM scores to form new combined scores which are then used to reorder the putative events. Each secondary score is generated by evaluating the segments of the putative event on trained keyword-specific truth and false alarm segment models.

The secondary processing algorithm is evaluated on two sets of conversational test speech from the Road Rally Database. Word spotting performance is shown to improve significantly over that of the baseline word spotting system with the largest gains occurring at low false alarm rates.

ACKNOWLEDGMENT

This work was sponsored by the U. S. Department of Defense.

REFERENCES

1. J. Cohen, "Segmenting speech using dynamic programming" in *JASA*, 69(5), May 1981, pp. 1430-1437.
2. Herbert Gish, Man Hung Siu, and J. Robin Rohlicek, "Segregation of Speakers for Speech Recognition and Speaker Identification" in *IEEE ICASSP* 1991, pp. 873-876.
3. J. Robin Rohlicek, William Russell, Salim Roukos, and Herbert Gish, "Continuous Hidden Markov Modeling for Speaker-Independent Word Spotting," in *IEEE ICASSP* 1989, pp. 627-630.
4. Man Hung Siu, George Yu, and Herbert Gish, "An Unsupervised, Sequential, Learning Algorithm for the Segmentation of Speech Waveforms with Multiple Speakers" in *IEEE ICASSP* 1992, pp. (II)189-192.