A ONE-PASS STRATEGY FOR KEYWORD SPOTTING AND VERIFICATION

Chak Shun LAI and Bertram E. SHI
Consumer Media Center/Human Language Technology Center, Department of Electrical and Electronic Engineering
Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, HONG KONG

ABSTRACT

One common method for keyword spotting in unconstrained speech is based upon a two pass strategy consisting of Viterbi-decoding to detect and segment possible keyword hits, followed by the computation of a confidence measure to verify those hits. In this paper, we propose a simple one-pass strategy where computation of the confidence measure is computed simultaneously with a Viterbi-like decoding stage. However, backtracking is not required, which when coupled with the need for only a single pass through the utterance significantly reduces the memory requirements of this algorithm. This feature makes it well suited for devices where processing power and memory are limited. Experimental results on a connected digits task show that performance of the decoding is comparable to that using a Viterbi search with backtracking. Experimental results on spotting days of the week in continuous speech indicate that the confidence measure calculated is effective in reducing the number of false alarms.

1. INTRODUCTION

Keyword spotting enables users to speak commands or requests without concern to the exact syntax of the request, as long as their utterance contains one or more of the keywords which the system has been designed to detect.

Many techniques for keyword spotting attack the problem with two passes through the data, e.g. [1]. The first pass uses Viterbi decoding to identify the boundaries of potential keywords in the utterance. The second part takes the boundaries of the keywords and the hypothesized identity of the keyword and computes a likelihood ratio between the hypothesized keyword and a filler model. This likelihood ratio measures the confidence in the keyword detection.

In this work, we propose a strategy which combines keyword detection and confidence measure computation in a single pass thorough the data without backtracking. Section 2 describes the algorithm. Section 3 shows experimental results from the algorithm on a connected digits task and on a task involving spotting the days of the week from continuous speech utterances.

2. ALGORITHM

For clarity, we first describe the algorithm as it would operate for utterances containing a single instance of one of K possible keywords. We then extend the algorithm to handle multiple instances of those keywords.

2.1 Isolated Keyword Spotting

The algorithm has some similarities with that proposed in [2]. In that work, each keyword is modelled by a concatenation of phoneme HMM. No filler, silence or garbage models are attached to the beginning or end. At each point in time, emission probabilities within each keyword are normalized by the emission probability of the best state to give a local score, which is then accumulated over time using a modified Viterbi algorithm to compute a normalized score for each state. The normalized score of the last state of each keyword serves as input to a subsequent decision stage. Because no backtracking is required, the algorithm has low computational and storage requirements, since only the last column of the trellis needs to be maintained in memory.

Our algorithm also uses a bank of keyword models consisting of HMMs with n states q_1 to q_n. However, each model has a filler model f_0 attached at the beginning and one model f_1 at the end. The filler models are single state multiple mixture HMMs trained on general speech and background[3]. It is important to note that the transition and emission probabilities for all of the filler models are identical. See Figure 1.

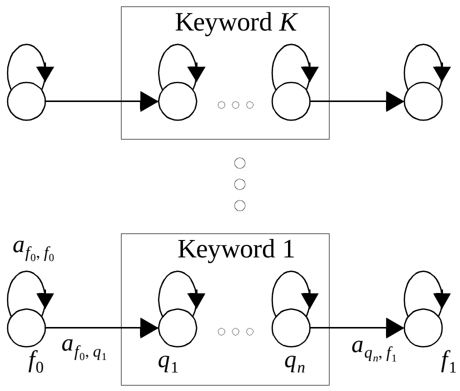

Figure 1. Word network used for isolated keyword spotting and verification.

The likelihoods of each of the keywords is evaluated independently using the standard Viterbi algorithm. In initializing the algorithm, we assume that only the initial filler model and the final filler model have non-zero probability:

$$P(s_1 = f_0) \neq 0 \text{ , } P(s_1 = f_1) \neq 0 \text{ and } \forall i, P(s_1 = q_i) = 0 \text{ .}$$

Backtracking is not required, so it is sufficient to store only the last column of the trellis. Instead, for each keyword model we compute only the log likelihood ratio between the state probability of the end filler model f_1 and the beginning filler model f_0:

$$LLR_{f_1 f_0}(t) = \log P(O_1^t, s_t = f_1) - \log P(O_1^t, s_t = f_0)$$

The following proposition indicates that this log likelihood ratio is closely related to the log likelihood ratio between the keyword and filler model used in many confidence measures:

Proposition 1:

Define $P(O_{t_i}^{t_f}, s_{t_f} = q_n | s_{t_i} = q_1)$ to be the likelihood of the observations from time t_i to t_f and the best path which starts in state q_1 and ends in state q_n of a keyword. Let $LLR(t_i, t_f)$ be the log likelihood ratio between the keyword and filler

$$LLR(t_i, t_f) = \log P(O_{t_i}^{t_f}, s_{t_f} = q_n | s_{t_i} = q_1)$$
$$-\log P(O_{t_i}^{t_f}, s_{t_f} = f_0 | s_{t_i} = f_0)$$

For any time t, define

$$LLR_{\max}(t) = \max_{1 < t_i, t_f < t} LLR(t_i, t_f)$$

Then,

$$LLR_{f_1 f_0}(t) = \max\{LLR_{\max} + \Theta_1, \Theta_2\}$$

where

$$\Theta_1 = \log \frac{a_{f_0, q_1} a_{q_n, f_1}}{a_{f_0, f_0}^2} \qquad \Theta_2 = \log \frac{P(s_1 = f_1)}{P(s_1 = f_0)}$$

and a_{f_0, q_1} is the state transition probability between the filler model f_0 and the first state of the keyword, a_{q_n, f_1} is the transition probability between the last state and the filler model f_1 and a_{f_0, f_0} is the transition probability from filler model f_0 to itself.

Proof:

Paths can begin only in either f_0 or f_1. However, by the model topology any path ending in f_0 must have started in f_0. Thus,

$$\frac{P(O_1^t, s_t = f_1)}{P(O_1^t, s_t = f_0)} = \max\left\{\frac{P(O_1^t, s_1 = f_0, s_t = f_1)}{P(O_1^t, s_t = f_0)}, \frac{P(O_1^t, s_1 = f_1, s_t = f_1)}{P(O_1^t, s_t = f_0)}\right\} \qquad (1)$$

Since we use a single state filler model, the state likelihood for the initial filler model is given by

$$P(O_1^t, s_t = f_0) = P(s_1 = f_0) P(O_1^t, s_t = f_0 | s_1 = f_0) \qquad (2)$$

where

$$P(O_1^t, s_t = f_0 | s_1 = f_0) = b_{f_o}(o_1)\left(\prod_{t=2}^{t} a_{f_0, f_0} b_{f_0}(o_t)\right)$$

or equivalently for any $1 < t_i, t_f < t$,

$$P(O_1^t, s_t = f_0) = P(O_1^{t_i - 1}, s_1 = f_0, s_{t_i - 1} = f_0) \times$$
$$a_{f_0, f_0} P(O_{t_i}^{t_f}, s_{t_f} = f_0 | s_{t_i} = f_0) \times \qquad (3)$$
$$a_{f_0, f_0} P(O_{t_f + 1}^{t}, s_t = f_0 | s_{t_f + 1} = f_0)$$

Since we assume that the output and transition probabilities for all filler models f_0 and f_1 are identical,

$$P(O_1^t, s_1 = f_1, s_t = f_1)$$
$$= P(s_1 = f_1) P(O_1^t, s_t = f_0 | s_1 = f_0) \qquad (4)$$

By definition,

$$P(O_{t_i}^{t_f}, s_{t_f} = q_n | s_{t_i} = q_1)$$
$$= \max_s \left\{ b_{q_1}(o_{t_i}) \left[\prod_{t = t_i + 1}^{t_f - 1} a_{s_{t-1}, s_t} b_{s_t}(o_t)\right] a_{s_{t_f - 1}, q_n} b_{q_n}(o_{t_f}) \right\}$$

By the optimality principle,

$$P(O_1^{t_f}, s_{t_f} = q_n | s_1 = f_0)$$
$$= \max_{1 < t_i < t}\{P(O_1^{t_i - 1}, s_{t_i - 1} = f_0 | s_1 = f_0) \times$$
$$a_{f_0, q_1} P(O_{t_i}^{t_f}, s_{t_f} = q_n | s_{t_i} = q_1)\}$$

and

$$P(O_1^t, s_1 = f_0, s_t = f_1)$$
$$= P(s_1 = f_0) \max_{1 < t_f < t}\{P(O_1^{t_f}, s_{t_f} = q_n | s_1 = f_0) \times$$
$$a_{q_n, f_1} P(O_{t_f + 1}^{t}, s_t = f_1 | s_{t_f + 1} = f_1)\}$$

Combining these two equations, we obtain

$$P(O_1^t, s_1 = f_0, s_t = f_1)$$
$$= \max_{1 < t_i, t_f < t}\{P(O_1^{t_i - 1}, s_1 = f_0, s_{t_i - 1} = f_0) \times$$
$$a_{f_0, q_1} P(O_{t_i}^{t_f}, s_{t_f} = q_n | s_{t_i} = q_1) \times \qquad (5)$$
$$a_{q_n, f_1} P(O_{t_f + 1}^{t}, s_t = f_1 | s_{t_f + 1} = f_1)\}$$

Combining (3) and (5)

$$\frac{P(O_1^t, s_1 = f_0, s_t = f_1)}{P(O_1^t, s_t = f_0)}$$
$$= \left(\frac{a_{f_0, q_1} a_{q_n, f_1}}{a_{f_0, f_0}^2}\right) \max_{1 < t_i, t_f < t} \frac{P(O_{t_i}^{t_f}, s_{t_f} = q_n | s_{t_i} = q_1)}{P(O_{t_i}^{t_f}, s_{t_f} = f_0 | s_{t_i} = f_0)}$$

Combining (2) and (4)

$$\frac{P(O_1^t, s_1 = f_1, s_t = f_1)}{P(O_1^t, s_t = f_0)} = \frac{P(s_1 = f_1)}{P(s_1 = f_0)}$$

Taking logarithms of both sides of (1) does not affect the max operation. Thus

$$\log P(O_1^t, s_t = f_1) - \log P(O_1^t, s_t = f_0)$$
$$= \max\{LLR_{\max}(t) + \Theta_1, \Theta_2\}$$

QED

The quantity $LLR_{\max}(t)$ is the maximum value of the log likelihood between the keyword and filler model evaluated over all possible keyword boundaries up to time t. Thus, this algorithm allows us to search simultaneously for the boundaries of the keyword and to evaluate the log likelihood between the keyword and filler over those boundaries in a single pass through the data. The two quantities Θ_1 and Θ_2 set a threshold on the minimum value of $LLR_{\max}(t)$ which can be measured by $LLR_{f_1f_0}(t)$. If $LLR_{\max}(t) < \Theta_2 - \Theta_1$, then $LLR_{f_1f_0}(t) = \Theta_2$. Otherwise, $LLR_{f_1f_0}(t)$ is affine in $LLR_{\max}(t)$. Note that while Θ_1 is set by the model parameters, the value of Θ_2 can be controlled by the choice of the relative probabilities of starting in the beginning and ending filler states.

One possible way to do keyword spotting for an utterance which is known to contain at most one keyword is to evaluate $LLR_{f_1f_0}(T)$ for each keyword, where T is the length of the utterance. The keyword with the maximum score could be chosen as the putative keyword within that utterance and the value of $LLR_{f_1f_0}(T)$ for that keyword used as a confidence measure. However, instead of pursing this approach, we describe in the next section an approach which enables us to detect multiple or repeated keywords within an utterance.

2.2 Multiple Keyword Spotting

The architecture is shown in Figure 2. As in the isolated keyword detection case, each keyword/filler block runs Viterbi scoring independently to compute $LLR_{f_1f_0}(t)$. These scores are then passed to the decision block which performs both detection and verification.

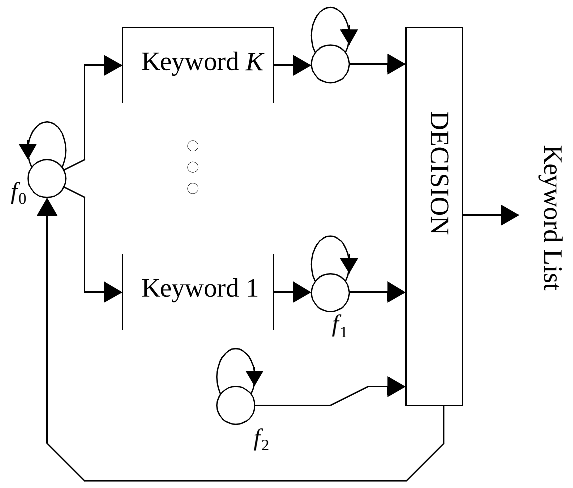

Figure 2. Network used for multiple keyword spotting and verification.

Suppose that in an utterance of length T, the optimal breakpoints determined by $LLR_{\max}(T)$ are given by t_i and t_f. Suppose also that $LLR_{\max}(T) > \Theta_2 - \Theta_1$. Proposition 1 indicates that the $LLR_{f_1f_0}(t)$ is constant for all t such that $t_f < t \le T$. This suggests that we can detect the endpoints of a keyword by detecting regions of length at least $T_{\min}$ where $LLR_{f_1f_0}(t)$ is constant.

By itself, this strategy could not detect multiple instances of a keyword because as $LLR_{f_1f_0}(t)$ increases, it simultaneously raises the threshold for detection of keywords with starting points $t_i > t$. This is because $LLR_{f_1f_0}(t)$ plays the role of Θ_2 in Proposition 1 for those keywords. To handle this, we propose a strategy to reset $LLR_{f_1f_0}(t)$ after detection of a keyword.

Another problem in applying this algorithm to the detection of multiple keywords is removal of overlapping keywords, which is not an issue for two pass strategies due to the Viterbi decoding. We handle this by treating the most recently detected keyword as a tentative result which is either discarded if an overlapping keyword with higher confidence is detected later, or validated if the end of the utterance is reached or another non-overlapping keyword is found.

Our algorithm is implemented based on the token-passing model[4]. At each frame t, tokens are propagated from one state to the next. When a token is propagated from f_0 to the first state q_1 of any keyword, it contains not only the starting point of the path (i.e. the path id) and the best score, but also the offset between $\log P(O_1^{t-1}, s_{t-1} = f_2)$ and $\log P(O_1^{t-1}, s_{t-1} = f_0)$, which we denote by $LLR_{f_2f_0}(t-1)$. This is required for correct computation of the confidence measure due to the reset operation.

The operation of the decision block in integrating the tokens passed from the keyword models and is described in detail below:

For each frame t and each keyword k,

1. Compute $LLR_{f_1f_0}(t)$.
2. Detect possible keyword endpoints by checking for each keyword whether $LLR_{f_0f_1}(t)$ is greater than Θ_2 and whether it has been constant for at least $T_{\min}$ frames.
3. If any endpoints are detected, choose the one with highest confidence measure calculated by the difference between $LLR_{f_1f_2}(t)$ and the offset from the token. Otherwise go to the next frame.
4. Compare the result with that stored as TEMP.
   a. If TEMP is empty, store the keyword along with its start point, end point and confidence measure as TEMP.
   b. If the keyword does not overlap TEMP, change TEMP to a permanent result and store the keyword as TEMP.
   c. If the keyword overlaps TEMP and has a higher confidence measure, then replace TEMP with the new keyword.
5. Pass the token of the detected keyword back to f_0 and set
   $$\log P(O_1^t, s_t = f_0) = \log P(O_1^t, s_t = f_1) + \Theta_2$$
   This resets the threshold for all subsequent keywords.

3. EXPERIMENTS AND RESULTS

We carried out two experiments to demonstrate the effectiveness of the proposed algorithm. In both experiments, we used 39 element feature vectors with 39 elements consisting of 12 MFCC, frame energy and delta and acceleration coefficients computed from 26 filter bank coefficients. Frame size was 25ms. Frame shift was 10ms.

The first experiment is designed to measure performance degradation due to the lack of backtracking in the one-pass algorithm, in comparison with the results by the Viterbi decoder supplied by HTK. We use the TIDIGITS corpus. The eleven keywords used were the digits “zero” to “nine” and “oh.” In this case, none of the utterances contain any out of vocabulary words. Thus, the experiment is specifically tailored to test the decoding performance.

Each keyword was modelled by a 9 state single mixture HMM. The filler model was single state with four mixtures. The same 3850 utterances from the TIDIGITS training corpus spoken by both male and female speakers were used to train both the keywords and the filler model. The algorithm was tested on a separate set of 3850 utterances from the TIDIGITS testing corpus. The word network used by the Viterbi decoder consisted of a free digit loop with filler models at the beginning and the end. Insertion penalties were tuned to optimize performance. Table 1 shows that the performance of the proposed algorithm is comparable to that obtained that from Viterbi decoding.

Gender	**Viterbi Accuracy (%)**	**One Pass Accuracy (%)**
Male	98.0	97.6
Female	97.8	97.4
Average	97.9	97.5

TABLE 1. Performance Comparison of the Proposed Algorithm with Viterbi Decoding and Back Tracking on a connected digits task using gender independent models.

The second experiment was designed to validate the efficacy of the confidence measure computed by the algorithm. We chose 1081 utterances from the WSJ1 database containing days of week, corresponding to about 2.5 hours of speech. The distribution of keywords are listed in Table 2. Note that some of the utterances contain more than one keyword. Each keyword model was constructed by concatenating phoneme models trained on the WSJ1 corpus and not specifically for this task. Each phoneme model was three state left-to-right and single mixture. Approximately one hour of speech (500 utterances) randomly chosen from the WSJ1 corpus was used for training the filler model, which was single state with four mixtures. Figure 3 plots the detection rate versus the false alarms per keyword per hour.

Keyword	**Count**	**Keyword**	**Count**
MONDAY	317	FRIDAY	436
TUESDAY	293	SATURDAY	116
WEDNESDAY	165	SUNDAY	68
THURSDAY	176	**TOTAL**	**1571**

TABLE 2. Keyword distribution of the testing data used in the second experiment

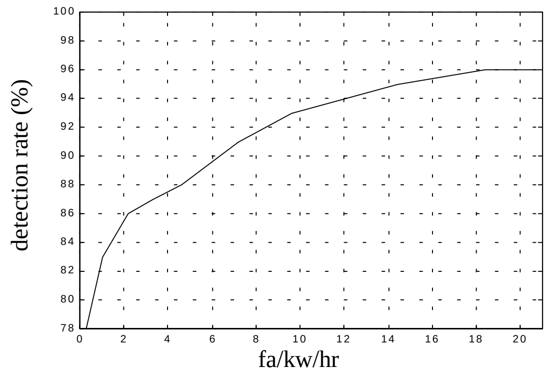

Figure 3. ROC for spotting 7 days of the week from continuous speech utterances from the WSJ1 corpus.

4. CONCLUSION

In this paper, we have presented an algorithm that detects multiple occurances of keywords and computes confidence measures associated with each detection in a single pass through the utterance. Despite the lack of backtracking, the results from running the spotting algorithm on connected digit strings yield accuracies comparable to that obtained using a Viterbi decoder. Experimental results in spotting days of the week indicate that the confidence measure calculated can be effective in reducing the number of false alarms. Due to the low memory and computational requirements, this algorithm may be well suited for applications where these resources are limited.

ACKNOWLEDGEMENTS

This research was supported by the Hong Kong Government Industry Department under the Innovation and Technology Fund Grant AF/264/97 and by the Hong Kong Research Grants Council under grant number RGC CA97/98.EG02.

REFERENCES

[1] R. C. Rose, “Keyword detection in conversational speech utterances using hidden Markov model based continuous speech recognition,” *Computer, Speech and Language*, vol. 9, pp. 309-333, 1995.

[2] J. Junkawitsch, L. Neubauer, H. Hoge, G. Ruske, “A New Keyword Spotting Algorithm with Pre-Calculated Optimal Thresholds,” *Proc. Fourth Intl. Conf. on Spoken Language Processing*, vol.4, pp.2067-70, Philadelphia, PA, 1996.

[3] J. G. Wilpon, L. R. Rabiner, C. H. Lee, E. R. Goldman, “Automatic Recognition of Keywords in Unconstrained Speech Using Hidden Markov Models,” *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 38, no. 11, pp. 1870-1878, Nov. 1990

[4] S. J. Young, N. H. Russel, and J. H. S. Thornton, “Token Passing: a Simple Conceptual Model for Connected Speech Recognition System, Cambridge University Engineering Department, Tech, Report No. TR. 38, July, 1989