A New Verification-Based Fast-Match for Large Vocabulary Continuous Speech Recognition

Mohamed Afify, Feng Liu, Hui Jiang, *Member, IEEE*, and Olivier Siohan, *Member, IEEE*

***Abstract*—Acoustic fast-match is a popular way to accelerate the search in large-vocabulary continuous-speech recognition, where an efficient method is used to identify poorly scoring phonemes and discard them from detailed evaluation. In this paper we view acoustic fast-match as a verification problem, and hence develop an efficient likelihood ratio test, similar to other verification scenarios, to perform the fast match. Various aspects of the test like the design of alternate hypothesis models and the setting of phoneme look-ahead durations and decision thresholds are studied, resulting in an efficient implementation. The proposed fast-match is tested in a large vocabulary speech recognition task and it is demonstrated that depending on the decision threshold, it leads to 20–30% improvement in speed without any loss in recognition accuracy. In addition, it significantly outperforms a similar test based on using likelihoods only, which fails, in our setting, to bring any improvement in speed-accuracy trade-off. In a larger set of experiments with varying acoustic and task conditions, similar improvements are observed for the fast-match with the same model and setting. This indicates the robustness of the proposed technique. The gains due to the proposed method are obtained within a highly efficient 2-pass search strategy and similar or even higher gains are expected in other alternative search architectures.**

***Index Terms*—Fast-match, large vocabulary speech recognition, search algorithms, utterance verification.**

I. Introduction

THE last decade has witnessed increased interest in the verification problem among the speech research community. The term verification encompasses a wide range of technologies that have important practical applications. Simply stated, verification reduces to the acceptance or rejection of a certain claim. For example, speaker verification (SV) [1] deals with accepting or rejecting a speaker's identity using voice, verbal information verification (VIV) [2] relates to the approval or denial of user's speech password, and utterance verification (UV) [3] focuses on assessing, again for acceptance or rejection, the output of speech recognition systems. This interest has led to the development of new techniques that improved the performance of verification-based systems.

Manuscript received April 16, 2002; revised February 7, 2004. This work was conducted while the authors were with Bell Labs, Lucent Technologies. The Associate Editor coordinating the review of this manuscript and approving it for publication was Dr. Shrikanth Narayanan.

M. Afify is with BBN Technologies, Cambridge, MA 02138 USA (e-mail: mohamed_afify2001@yahoo.com).

F. Liu is with Avaya Labs, Basking Ridge, NJ 07920 USA (e-mail: fengliu@research.avayalabs.com).

H. Jiang is with the Department of Computer Science, York University, Toronto, ON, Canada M3J 1P3 (e-mail: hj@cs.yorku.ca).

O. Siohan is with the IBM T. J. Watson Research Center, Yorktown Heights, NY 10598 USA (e-mail: siohan@watson.ibm.com).

Digital Object Identifier 10.1109/TSA.2005.845809

By formulating the verification problem in a hypothesis testing framework where acceptance is associated to the null hypothesis, and rejection to the alternative hypothesis, it was shown that the optimal solution reduces to constructing a likelihood-ratio test (LRT) and comparing it to a threshold to arrive at the optimal decision. While this unified framework is applicable to the general verification problem, details for choosing and training the corresponding models for both hypotheses is problem-specific, and became an interesting research area in different application domains. Intuitively, the LRT can be viewed as an optimal normalization of a conventional likelihood-based procedure. In addition to the hypothesis testing framework, other alternatives exist for performing verification. These are mainly based on using posteriors, where the sum of all class probabilities is used as a normalization, or some form of approximate posterior, where, for example, the sum is replaced by maximum leading to a test that is very similar to beam pruning, which is widely used in search algorithms. In essence, these methods replace the alternative hypothesis score by some quantity calculated from the class models. For a more in-depth overview, we refer the reader to a recent review on the verification problem [4].

In the statistical approach to automatic speech recognition the best word sequence is chosen as the sequence which maximizes the a posteriori probability of the words given the observations [5]. Finding such a sequence requires in principle exploring every valid word sequence hypothesis in order to determine the best scoring one. Large vocabulary applications, on the order of several thousand words, result in a very large state-space defined by the language model, the acoustic-phonetic models of phones, and the pronunciation lexicon. Hence, for these applications, finding the best word sequence involves an expensive search in this state-space. Reducing this search space and hence accelerating the computation is one of the most active research areas in speech recognition. We refer the reader to [6]–[8] for an overview of the recent trends of the search problem in automatic speech recognition.

Look-ahead [7] techniques are among the popular ways for reducing the search space. In these methods we look-ahead in time using some acoustic and/or language model[1] probabilities to predict some hypotheses that will score poorly in the future, and hence discard them from detailed evaluation. In this article we are primarily concerned with the application of ideas from verification, as discussed above, to acoustic look-ahead techniques, also named acoustic fast-match or simply *fast-match*. Reference [7] contains an overview of language model look-ahead techniques. In the rest of this section we first review some

[1]Depending on using acoustic or language model look-ahead or both.

popular fast-match techniques, then motivate and outline the use of verification methods in fast-match design which will be discussed in detail in the rest of the paper.

The basic idea of a fast-match (FM) is to look-ahead in time, using a computationally cheap method, to identify some phonemes with poor acoustic scores and discard them from the search space before detailed evaluation. In other words, rather than expanding the search network with some phone hypotheses that are likely to be pruned after detailed scoring, these hypotheses are pruned immediately, right after the fast-match scoring. Using a FM improves the speed of the search at the expense of risking the loss of the correct path. Hence a "good" fast-match should accelerate the computation with minimal loss of accuracy. Many fast-match algorithms were proposed in the speech recognition literature. These algorithms share the same basic idea but often differ in detail. Here we identify two general types of fast matches: global fast-match (GFM), and local fast-match (LFM). These two broad categories differ in their use of available information in making the fast-match decision. GFM combines the node score with the look-ahead score in making the pruning decision. Thus, it mimics the true search but uses a simpler network in calculating the fast-match score. For example, in [9] a Viterbi search is run ahead of the current time on a simple phoneme network using simple acoustic models. This search leads to a score for each phoneme. The total score of each potential path is then calculated based on its current score (coming from the search) plus the fast match score of the appropriate phoneme. The path is either pruned or considered for detailed evaluation depending on this combined score. On the other hand, in a LFM, only the local look-ahead score is used in making the fast-match pruning decision. For example, in phone deactivation pruning [10] the posterior probability of a phoneme in a fixed length look-ahead interval is used in taking a local pruning decision for the phoneme. Also in the channel-bank approach of [11] the phoneme likelihood in a fixed length window is compared to a threshold and used for pruning. Thus, the main difference between GFM and LFM lies in the use of the search score in taking the local pruning decision. The distinction between the two is important in our context because, as stated above, GFM mainly mimics the original search and can not easily host ideas like posteriors or verification, while LFM basically performs a local decision irrespective of the search results, and hence can accommodate easily such extensions.

This paper focuses on applying ideas from verification to local fast-match. At a certain time instant the fast-match makes a binary decision concerning the presence or absence of a certain phoneme. This type of binary decision problem is very similar to the verification problem. Using the current look-ahead information we want to accept or reject the hypothesis that a certain phoneme[2] will start at the current time. Using this reasoning the LFM readily fits in the verification framework discussed above. As stated at the beginning of this section a popular way to approach the verification problem is by using hypothesis testing framework. In this approach, the likelihood ratio of the null and alternative hypotheses is formed and compared to a threshold to make the required decision. Hence, we propose to use a similar likelihood ratio based procedure for constructing a LFM for large vocabulary speech recognition. In doing so we are mainly motivated by the success of the hypothesis testing framework in problems such as speaker and utterance verification [1], [3], where this methodology has proved to be superior to the more traditional approach that uses only likelihoods in making the decision. Thus, we believe that using such a likelihood ratio based verification scheme might be better than using only likelihood in designing the fast-match. Alternative approaches to this likelihood ratio based scheme, such as using posterior, exist in the verification literature, and have also been used for constructing local fast-match as in [10]. The comparison to these techniques, while interesting, is out of the scope of this paper and may be pursued in future work.

In addition to formulating a LFM in a hypothesis testing framework, and developing the corresponding likelihood ratio based LFM scheme, we consider design issues of the proposed approach. These include choosing and building appropriate models for the null and alternative hypotheses, calculating the decision thresholds, and calculating the look-ahead durations of each phoneme. In all design issues our main focus is on reducing the overhead incurred in calculating the fast match score while keeping the error introduced by the fast-match as small as possible. Considering these design considerations is important because while the hypothesis testing framework provides the necessary theoretical foundation, implementation details are application specific and should be reconsidered separately for each problem.

The paper is organized as follows. In Section II, we briefly overview hypothesis testing theory. We then show how to cast the acoustic fast-match as a hypothesis testing problem and hence develop a likelihood ratio based fast-match scheme in Section III. Section IV addresses design and implementation issues of the proposed method. Our basic decoder structure is briefly reviewed in Section V where it is shown how the fast-match is incorporated in the search mechanism. We present experimental results on a 20 K Japanese broadcast news task in Section VI. These results demonstrate the computational advantages introduced by the proposed fast-match compared to the case of not using a fast-match an dalso using a fast-match based on the likelihood score only. Finally, conclusions are drawn in Section VII.

II. Hypothesis Testing

This section gives a brief introduction to hypothesis testing theory. Hypothesis testing [12] is a general statistical framework for deciding among several hypotheses based on some observations. In particular, binary hypothesis testing chooses one among two hypotheses, usually referred to as the null (H_0) and alternative (H_1) hypotheses. In performing hypothesis testing two types of errors can occur. These are quantified by the probability of miss $p_M = Pr(\text{Say } H_1 | H_0 \text{ is true})$, and the probability of false alarm $p_F = Pr(\text{Say } H_0 | H_1 \text{ is true})$. Of course, one would like to minimize both probabilities, but there is a trade-off between them. Hence, we usually minimize p_M, or

[2]A word can be equivalently considered but we focus on phonemes in this work.

equivalently maximize $p_D = 1 - p_M$ subject to the constraint that p_F is less than some specified value.

A solution to this problem is given by the Neyman-Pearson Lemma. If the probability densities of the observation under both hypotheses are given, we first form the likelihood ratio, which is the ratio of the probability densities under the null and alternative hypotheses. This ratio is further compared to a threshold, and we decide in favor of H_0 if the ratio is greater than the threshold, and in favor of H_1 otherwise. The threshold is determined to satisfy the constraint on the false alarm probability. Because the logarithm is a monotonic function, the above procedure is usually implemented in terms of logarithms.

After this brief introduction to hypothesis testing we will show in the next section how we can formulate the fast-match as a hypothesis testing problem.

III. Proposed Fast-Match

This section formulates a fast-match as a hypothesis testing problem. For a fast-match the null and alternative hypotheses for phoneme α can be written as

$$H_0 : \alpha \text{ starts at time } t$$
$$H_1 : \alpha \text{ does not start at time } t.$$

The first step toward developing a likelihood ratio test for the above hypothesis testing problem is to define suitable probability distributions for both hypotheses. These probability distributions can be written as $P(X_t^{t+d_1}|\alpha \text{ starts at } t)$, and $P(X_t^{t+d_2}|\alpha \text{ does not start at } t)$, where X_a^b represents acoustic observations in the interval $[a, b]$, and d_1, and d_2 (not necessarily equal) are possible durations of both events. As phonemes are known to have variable length it is not possible to determine beforehand the values of both durations, and a search procedure, as in [9], can be used to determine the best end points at each time. On the other hand some types of fast matches, e.g., [10], and [11], use a fixed look-ahead interval. We adopt this fixed duration approach, and in this case the two distributions reduce to $P(X_t^{t+d_\alpha}|\alpha)$, and $P(X_t^{t+d_\alpha}|\bar{\alpha})$, where we set $d_1 = d_2 = d_\alpha$, and denote the models of the null and alternative hypotheses α, and $\bar{\alpha}$ respectively.

A note regarding the proposed fast-match is worth mentioning here. It is possible to define alternative fast match methods based on $P(\alpha|X_t^{t+d_\alpha})$, the posterior probability of α given the observations, and abandon the alternate hypothesis altogether. This approach was used in [10] where a neural network is used to supply the posterior value. In addition, the denominator of the posterior can be further approximated as $\max_\alpha P(X_t^{t+d_\alpha}|\alpha)$, this results in a test which is similar in spirit to beam pruning that is used in fast matches using a network search [9]. As mentioned in the introduction all these alternatives, including the proposed one, can be viewed as different normalizations of the likelihood score which can be considered as a baseline for all techniques. It is not usually possible to arrive at general conclusions of which type of normalization is better. However, if the problem is viewed in the context of hypothesis testing, as done in this paper, the likelihood ratio test is known to be optimal.

Now, assume the score of phoneme α at time t is denoted $S_t(\alpha)$, and that the lookahead duration of α is d_α. We define $S_t(\alpha)$ as the log probability that α starts at t and ends at $t + d_\alpha$. Hence, we write $S_t(\alpha) \equiv \log P(X_t^{t+d_\alpha}|\alpha)$, where $X_t^{t+d_\alpha}$ is the observation sequence in $[t, t + d_\alpha]$. The log likelihood ratio $L_t(\alpha)$ can be written as

$$L_t(\alpha) = S_t(\alpha) - S_t(\bar{\alpha}) \tag{1}$$

where $S_t(\bar{\alpha}) \equiv \log P(X_t^{t+d_\alpha}|\bar{\alpha})$, and $\bar{\alpha}$ stands for alternate hypothesis model of phoneme α, and hence represents the alternate hypothesis H_1. The likelihood ratio test reduces to accepting H_0 when $L_t(\alpha) > \eta_\alpha$ and deciding H_1 otherwise, where η_α is the decision threshold.

In order to calculate the null and alternate hypotheses scores we must first evaluate the corresponding probabilities. A natural choice would be to use hidden Markov models (HMMs) to calculate these probabilities. In addition, the score is calculated for every time instance, and hence it would be interesting to incrementally calculate the score at time t from the corresponding score at time $t - 1$. When using a fixed lookahead duration, the work in [11] develops an elegant recursion to incrementally calculate the phoneme scores for HMMs. Further, if a phoneme is represented by a 1-state HMM, or equivalently a Gaussian mixture model, the recursion reduces to the following very simple formula

$$S_t(\alpha) = S_{t-1}(\alpha) + \log p\left(x_{t+d_\alpha}|\alpha\right) - \log p(x_{t-1}|\alpha). \tag{2}$$

The probability $p(x_\tau|\alpha)$ can be calculated as

$$\begin{aligned} p(x_\tau|\alpha) &= \sum_{m=1}^{M} c_m \mathcal{N}(x_\tau, \mu_m, \Sigma_m) \\ &\approx \max_m c_m \mathcal{N}(x_\tau, \mu_m, \Sigma_m) \end{aligned} \tag{3}$$

where M is the number of Gaussian mixture components, and c_m, μ_m, and Σ_m, are the mixture weight, mean, and covariance of the m^{th} Gaussian component. The dependence of these quantities on the phoneme α has been dropped for convenience. We also note that the maximum approximation, widely used in calculating Gaussian mixtures, is used here for computational efficiency and is different in principle from the beam pruning type maximum approximation mentioned at the beginning of this section.

In turn, the likelihood ratio can also be incrementally calculated as

$$L_t(\alpha) = L_{t-1}(\alpha) + q\left(x_{t+d_\alpha}, \alpha\right) - q(x_{t-1}, \alpha) \tag{4}$$

where

$$q(x_\tau, \alpha) = \log p(x_\tau|\alpha) - \log p(x_\tau|\bar{\alpha}) \tag{5}$$

and $p(x_\tau|\bar{\alpha})$ is calculated as in (3) but using the parameters of the alternate hypothesis model. We note that in performing the Gaussian calculations in (3) we use the now popular Intel streaming SIMD extension (SSE) instruction set [13], which accelerates the Gaussian computations by about 50%.

At each time frame the proposed fast-match requires $2M$ Gaussian calculations, an addition, and a subtraction for each

phoneme. This represents a very small overhead compared to the Gaussian calculations for several thousand states, and the time alignment needed by the original search. We did not measure explicitly the fast-match overhead and its performance will be evaluated in the experiments. In the next section we will discuss implementation of the proposed fast-match.

IV. Implementation of the Fast-Match

In the previous section we showed how a fast match is formulated as a hypothesis testing problem and derived a simple test based on a fixed look-ahead duration and a Gaussian mixture model assumption, that can be incrementally calculated in each frame. However, several implementation issues for the successful application of the proposed test remain to be addressed. These include the definition of alternate hypothesis or anti-phoneme models, parameter estimation of the phoneme and anti-phoneme Gaussian mixture models, and determining the phoneme look-ahead durations and decision thresholds. These issues will be addressed in this section.

A. Design of Anti-Phoneme Models

Choosing appropriate anti-phoneme models to account for the alternate hypothesis and hence to be used in calculating the likelihood ratio is known to be a very important design aspect in all verification problems. A general trend in their design is to consider either class specific models or a shared model, often referred to as background model. Both avenues were explored in this work, i.e., we considered both a common background model for all phonemes, and a separate anti-model for each phoneme. While it can be argued that the background model contains information from the phoneme of interest this model has been widely employed, and can be considered as representing the whole structure of acoustic space in contrast to phoneme models which are more related to the phoneme of interest. In initial experiments we obtained similar results, in terms of speed-accuracy trade-off, for both the general background model and the phoneme specific models. Given that the latter almost doubles the number of fast-match computations a single background model was used in all subsequent experiments. We should note here that in other verification scenarios, e.g., utterance verification [3], it was found that specific anti-models significantly outperform a general background model.

B. Training of Phoneme and Anti-Phoneme Models

Once we arrive at a definition for anti-phoneme models it remains to estimate their parameters, in addition to the phoneme models parameters, from training data. This is done as follows. First, a set of training utterances, e.g., those used in building the acoustic models, is first segmented using forced alignment into phoneme units. The training data for each class is then defined by collecting all segments belonging to this class. For example, for building a phoneme model all segments belonging to this phoneme are pooled, and for constructing a general background model all training data are put together, while for creating phoneme specific anti-models all data not belonging to the required phoneme are stacked. The collected data represent the training set for the class of interest. An interesting in-search strategy for collecting alternate hypothesis training data was recently proposed in [14], and may be considered for future work, in place of the fixed segmentation approach taken here.

Once the training data of each class is collected, as discussed above, maximum likelihood (ML) estimation can be used to estimate its Gaussian mixture model. ML estimation is implemented using the segmental K-means algorithm [15] to estimate the mixture weights, means, and diagonal covariances of each model. For simplicity, all models are assumed to have the same number of components. Experimental results for varying the number of components will be given in Section VI.

An alternative to ML estimation for estimating model parameters is the minimum classification error (MCE) learning framework. MCE learning estimates the parameters by minimizing a smooth estimate of the classification error using the generalized probabilistic descent (GPD) algorithm. The application of MCE to Gaussian mixture models is a special case of the algorithm for continuous density HMM and can be found in [16]. In other verification problems, e.g., utterance verification [3], and speaker verification [17] it was found that the use of a discriminative training algorithm significantly outperforms an ML trained baseline. In our application, i.e., verification for fast-match, we did not find, in initial experiments, a significant gain, in terms of speed-accuracy trade-off, from using MCE in building the Gaussian mixture phoneme and anti-phoneme models. Thus, in the rest of the article we report results for ML trained models.

C. Calculation of Look-Ahead Duration

As discussed in Section III the implementation of the fast match requires the calculation of a look-ahead duration d_α for each phoneme. In this work we use a very simple method for calculating this look-ahead duration. After the segments belonging to each phoneme are identified using forced alignment of the training data, the look-ahead duration is computed as the average duration of these segments.

D. Calculation of Decision Thresholds

Determining the phone decision thresholds η_α is an important issue in designing the fast-match. In general hypothesis testing problems, these thresholds are used to balance the trade-off between the miss and false alarm probabilities. For example in some applications the equal error rate is used to determine their values. For fast-match, a miss will lead to a recognition error while a false alarm will result in additional search effort. As we are mainly interested in minimizing the recognition errors introduced by the fast-match we choose the decision threshold to provide controlled coverage of the area under the correct score density. This is done in the following simple way.

We start from the segmented acoustic training data, and the Gaussian mixture models. Then for each phoneme we evaluate the score, as in (1), of all segments in the training set belonging to this phoneme α. We calculate the mean score μ_α and the standard deviation σ_α of the score of these segments. These represent the mean and the standard deviation of the distribution of the likelihood ratio for correct phone segments in the training

data. Then we control the coverage of the correct score density by choosing the threshold as lying at a multiple of the standard deviation below the mean. Thus, during the test we reject those segments whose score lies at multiple of the standard deviation below the mean. Hence, the threshold is calculated as $\eta_\alpha = \mu_\alpha - n\sigma_\alpha$. Here n is used to trade off the speed and accuracy. A large n leads to a slower but more accurate search and vice versa. The choice of appropriate n will be addressed in the experimental evaluation. We also note that when implementing a likelihood based fast-match, as will be given in Section VI for comparison purpose, the same procedure is used for determining the thresholds except for using the likelihood score instead of the likelihood ratio.

V. Basic Decoder Structure

The proposed fast-match can be applied to many of the well known search strategies in continuous speech recognition. In this section we briefly overview our two pass decoder and show how to integrate the fast-match in this search.

The basic idea of the two pass search is to limit the potential search space to a set of most likely candidates using a fast search strategy and simple acoustic and language models during the first pass. These candidates are then rescored using detailed acoustic and language models in the second pass. This two pass strategy is adopted in many speech recognition systems. In particular, our search structure is similar to that in [18]. We will describe this two pass search below and more details can be found in [19].

The first pass of our 2-pass decoder uses a single static tree search structure, left biphone acoustic models, and a bigram language model. Based on these ingredients a classical Viterbi beam search is used, where at each time instant the top K candidates are propagated back into the root of the tree. The list of the best word candidates at each time instant is kept for later use in the backward pass. The forward pass results in a set of likely words ending at each time frame. This is followed by a backward pass whose search space is constrained by the results of the first pass. The backward pass uses within-word triphone models, and a backward trigram language model. The backward pass is based on a Viterbi beam search on the constrained search space.

The fast-match is invoked during the forward pass. As mentioned above the forward pass uses a single static tree to perform a Viterbi beam search. At each time instant the set of active nodes in the tree are expanded to give rise to a set of new arcs, as represented in Fig. 1. According to the lexicon each arc has a certain phoneme label. Typically, many of these arcs are expanded to be pruned later by the beam search. The fast-match can be invoked to overcome this problem. At each time instant the fast-match is run and each phoneme is either accepted or rejected according to the fast-match test. An arc is expanded only if the corresponding phoneme passes the fast-match test. This results in huge savings because the rejection of a phoneme will lead to rejection of all arcs carrying this phoneme identity.

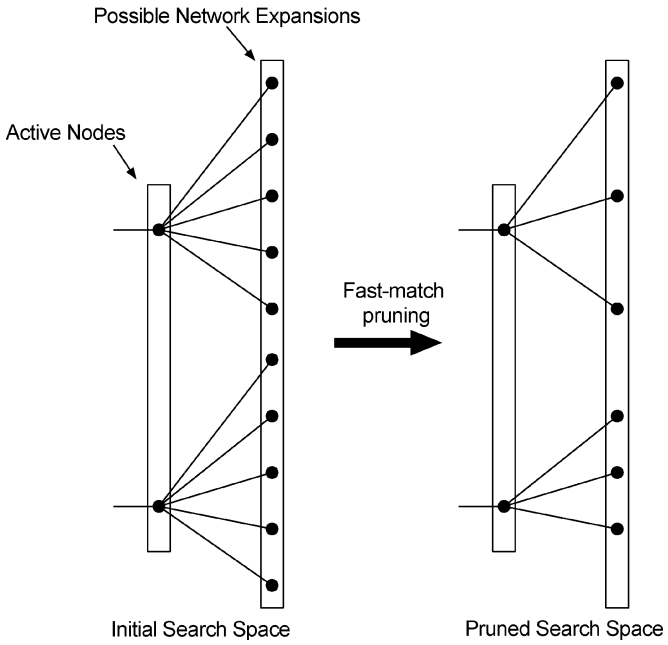

Fig. 1. Principle of the fast-match algorithm: before costly evaluation of all possible network expansions, the fast-match is used to prune unlikely hypotheses. Only the remaining hypotheses will be scored using detailed acoustic models.

VI. Experimental Evaluation

The proposed algorithm is tested on a Japanese broadcast news transcription task, whose vocabulary is drawn from 20 000 words. Training and test speech data in addition to the trigram language model are provided by the Japan Broadcasting Corporation (NHK).

We will report results on two series of experiments. First, we will illustrate the behavior of the fast match algorithm on a small development set, and describe how tuning the threshold affects the performance of the system. Then, given the best setting, we will evaluate the performance and robustness of the fast-match on a much larger dataset including various acoustic environments.

In the first series of experiments, the training data consists of 90 h of speech. Context dependent hidden Markov models are built using decision tree clustering [20]. The feature space is 39-dimensional consisting of 12 cepstrum coefficients, energy, and their first and second order derivatives. The test set consists of 162 utterances from male speakers, in a clean studio environment. The test set perplexity is about 34 and the out-of-vocabulary (OOV) rate is 0.76%. Our baseline system runs in about 0.79 times real-time, for a word error rate (WER) of 4.04%. That corresponds to a system that has been already highly optimized for speed.

The fast-match model is trained on the same acoustic data used for training the acoustic model. The phoneme models are trained from segmented data using the segmental K-means training algorithm. The background model is trained from the data of all phonemes and has the same size as the phoneme models. The look-ahead duration of each phoneme is taken as its average duration in the training data. We evaluated various design parameters of the fast-match namely the Gaussian

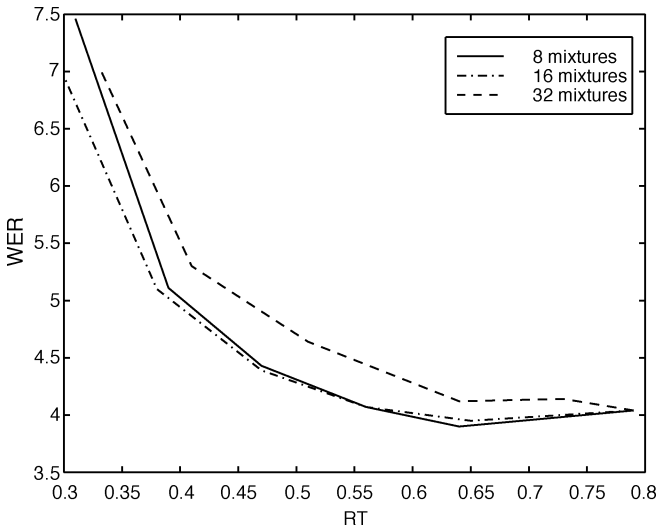

Fig. 2. Percentage word error rate (WER) and real time factor (RT) for the fast-match with mixture sizes 8, 16, and 32, and for different thresholds on the development set. Likelihood ratio scores are used here for the fast-match. The threshold is varied as $\mu - n\sigma$ where $n \in \{4, 3.5, 3, 2, 1\}$, and higher n indicate slower but more accurate search. The baseline result with 4.04% WER and 0.79 RT is also shown.

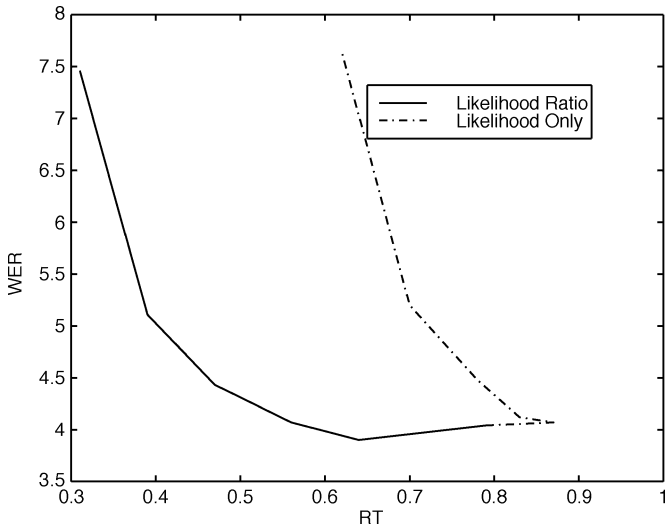

Fig. 3. Percentage word error rate (WER) and real time factor (RT) for the fast-match with mixture size 8 and for different thresholds on the development set. Both likelihood ratio and likelihood scores are used here for the fast-match for comparison. The threshold is varied as $\mu - n\sigma$ where $n \in \{4, 3.5, 3, 2, 1\}$ and higher n indicate slower but more accurate search. The baseline result with 4.04% WER and 0.79 RT is also shown.

mixture size, and the value of threshold. The Gaussian mixture size is set to 8, 16, and 32 while the threshold is taken as $\mu - n\sigma$ as discussed in Section IV, and n takes the values $n \in \{4, 3.5, 3, 2, 1\}$. Large n indicates slower but more accurate search and vice versa. The results are given in Fig. 2. The figure shows that the performance, in terms of accuracy-speed trade-off, is almost insensitive to the Gaussian mixture size in the range 8–32. It also demonstrates the importance of the threshold n in balancing the accuracy-speed trade-off. A value of n around 3.0–3.5 accelerates the search by about 20%–30% while keeping almost the same accuracy.

The next set of experiments, shown in Fig. 3, compares the performance, for different threshold values,[3] of both the likelihood ratio and likelihood based fast-match. The size of the Gaussian mixture in this case is set to 8, and the thresholds are taken the same as above. The results clearly indicate the superiority of using the likelihood ratio based fast-match. In fact, the likelihood based fast-match fails in bringing any improvement in terms of accuracy-speed trade-off in our setting. This is in contrast to other results, e.g., [11], where a likelihood based algorithm was capable of providing reasonable improvements.

Some discussions and comments on the presented results will be given below. First, it can be argued that, in contrast to the presented results, the use of likelihood based fast-match proved successful in other contexts. Some of these results are obtained in a GFM setting (as discussed in the paper), e.g., [9], where the global network score is used in the fast-match decision, and are not directly comparable to our approach. While it is difficult to decide which strategy (local or global) is better for implementing a fast-match, a major advantage of a LFM is its computational simplicity, and the ability to enhance it using posteriors [10], or likelihood ratio as done in this work. It is not clear how these ideas could be integrated in a GFM.[4] In other situations which employ a LFM approach, e.g., [11], reasonable improvements were obtained using likelihoods in contrast to the results presented here but this was done in a different search architecture which makes them not directly comparable. Second, it may seem more reasonable to compare the proposed fast-match to other more elaborate schemes like using posteriors [10], or using a normalization based on the maximum over all phonemes,[5] instead of comparing to the likelihood. However, as discussed above, in many instances likelihood based fast-match showed improvements which qualifies it as a candidate for comparison. In addition, all this schemes may be viewed as different normalizations of the likelihood that were used not only in fast-match but also in different applications of verification technology. Comparing them remains to be an interesting issue in future work.

TABLE I
LIST OF ALL EVALUATION CONDITIONS AND NUMBER OF TEST UTTERANCES PER CONDITION

Environment	# Test utterances
Field Report	148
Noisy News	56
Spontaneous	40
Sports (clean)	80
Sports (noisy)	117
Studio News	274
Weather	39

In a second series of experiments, we illustrate the performance of the proposed fast-match algorithm on a much larger data set, including various acoustic environments, listed in Table I. This set is intended to measure the robustness of the proposed method to changing acoustic and task conditions. The

[3]As noted before thresholds are calculated as $\mu - n\sigma$, and they are phoneme dependent. Also we would like to emphasize that μ and σ are different for both likelihood ratio and likelihood fast-match as discussed in Section IV.

[4]In [21] a method for calculating the posterior of sentences using a word graph, in the context of utterance verification, is given but it is too complex to be used for fast-match.

[5]As suggested by a reviewer.

TABLE II
WORD ERROR RATE (WER) AND REAL-TIME (RT) FACTOR ON NHK EVALUATION TEST SET WITH AND WITHOUT FAST-MATCH

Environment	Fast-match off		Fast-match on	
	WER	RT	WER	RT
Field Report	22.0	2.19	23.1	1.66
Noisy News	2.9	1.15	2.1	0.89
Spontaneous	26.5	1.85	24.7	1.32
Sports (clean)	6.7	1.21	7.5	0.86
Sports (noisy)	15.2	2.63	15.5	2.17
Studio News	2.7	0.98	2.8	0.72
Weather	5.9	1.06	4.6	0.80
Average	11.7	1.58	11.5	1.20

test perplexity varies from less than 10 to about 80 depending on the environment, with OOV rates ranging from 0.25% to 2.5%. The acoustic models used for recognition are build on about 170 hours of training data. The fast-match models are the same as in the previous experiments, and the threshold is set to $\mu - 3.5\sigma$.

Results are reported in Table II in terms of word error rate and real-time factor, with and without activating the fast-match. On average, over all environments, the fast match leads to about 0.2% absolute reduction of the word error, for a 20% to 30% speed-up of the decoding time. This illustrates the robustness of the proposed fast-match: the fast-match models are easy to build and are effective in many acoustic environments, while the fast-match thresholds require little tuning.

VII. CONCLUSION

This paper presents an acoustic fast-match for large-vocabulary speech recognition. The basic idea is to view the fast-match as performing an accept/reject decision at the phoneme level, and hence cast it as a verification problem. This naturally leads to the use of hypothesis testing framework, widely used for other verification tasks, to develop an optimal solution to the problem as a likelihood ratio test performed at each frame. Further, by employing the idea of LFM, as defined in the paper, it is shown that at the current frame the test can be incrementally calculated from the previous frame using very simple computations. This is again very important because it may turn out, see [9], that a fast-match may do a very good job in performing the accept/reject decision in terms of reducing the number of visited arcs while it does not lead to similar reduction in the total recognition time due to the overhead incurred in its computation.

In addition to the link established with the verification problem, which we believe is one of the important contributions of this work, the proposed method has links with other LFM strategies proposed in the literature. A natural choice is to use the likelihood as a basis of the fast-match decision as done in [11]. In this context the likelihood ratio provides an optimal form of normalization of the likelihood score which leads to large performance improvements as shown in the experimental comparisons done in the paper. In addition it offers robustness as evidenced in the multi-environment experiments carried out in Section VI. The likelihood ratio is not the only possible normalization of the likelihood, and using the posteriors, which employ the sum of probabilities of all classes as a normalization, or some approximate posteriors are valid alternatives. For example, [10] uses posteriors to develop a very interesting and efficient fast-match. The comparison between different normalization procedures in this context, and also for the general verification problem remain an interesting issue for future work.

It is well known that the successful application of the hypothesis testing framework to a certain verification problem requires, in addition to theoretical developments, some design issues related to defining and building alternate hypothesis models. These considerations were addressed for the proposed fast-match. In particular, we studied the definition (phoneme specific versus general), and training (ML vs MCE) of alternate hypothesis phoneme models, choice of phoneme look-ahead durations, and decision thresholds. Interestingly, results regarding these aspects came in contrast to general practice in verification problems as discussed in Section IV. We believe this is due to the specific nature of the fast-match where simple models are favored and where the price of a miss is higher than a false alarm.

The proposed fast-match is applied in the first pass of a 2-pass search algorithm. When used in the search, for a 20 K Japanese broadcast news task, the fast-match leads to about 20%–30% speed-up in the overall search with no degradation in accuracy. In addition, the improvements using the same fast-match models and settings carry over to a more diverse testing setup with different environmental conditions and different acoustic models which highlights the robustness of the proposed technique. We would like to point out that these improvements are obtained for a highly optimized search strategy that uses a single static tree in the first pass and it may be expected that more savings can obtained in the context of the more conventional multiple tree Viterbi search or other search strategies.

ACKNOWLEDGMENT

The authors would like to thank NHK Science and Technology Labs, Japan, for providing them with their databases. Insightful comments from the reviewers, which improved the presentation of the paper, are appreciated.

REFERENCES

[1] A. E. Rosenberg, J. DeLong, C.-H. Lee, B.-H. Juang, and F. K. Soong, "The use of cohort normalized scores for speaker verification," in *Proc. Int. Conf. on Spoken Language Processing*, Banff, AB, Canada, 1992, pp. 599–602.

[2] Q. Li, B.-H. Juang, Q. Zhou, and C.-H. Lee, "Automatic verbal information verifiction for user authentication," *IEEE Trans. Speech Audio Process.*, vol. 8, no. 5, pp. 585–596, Sep. 2000.

[3] M. G. Rahim, C.-H. Lee, and B.-H. Juang, "Discriminative utterance verification for connected digits recognition," *IEEE Trans. Speech Audio Process.*, vol. 5, no. 3, pp. 266–277, May 1997.

[4] H. Jiang, "Confidence measures for speech recognition: A survey," Tech. Rep., Dept. Comput. Sci., York Univ., York, U.K., [Online] Available:, http://www.cs.yorku.ca/techreports/2003/CS-2003-06.html, May 2003.

[5] L. R. Bahl, F. Jelinek, and R. L. Mercer, "A maximum likelihood approach to continuous speech recognition," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 5, no. 2, pp. 179–190, Mar. 1983.

[6] F. Jelinek, *Statistical Method for Speech Recognition*. Cambridge, MA: MIT Press, 1997.

[7] H. Ney and S. Ortmanns, "Progress in dynamic programming search for LVCSR," *Proc. IEEE*, vol. 88, no. 8, pp. 1224–1240, Aug. 2000.

[8] N. Deshmukh, A. Ganapathiraju, and J. Picone, "Hierarchical search for large-vocabulary conversational speech recognition: working toward a solution to the decoding problem," *IEEE Signal Processing Mag.*, vol. 16, no. 5, pp. 84–107, Sep. 1999.
[9] S. Ortmans, A. Eiden, H. Ney, and N. Coenen, "Look-ahead techniques for fast beam search," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, Munich, Germany, Apr. 1997, pp. 1783–1786.
[10] S. Renals, "Phone deactivation pruning in large vocabulary continuous speech recognition," *IEEE Signal Processing Lett.*, vol. 3, no. 1, pp. 4–6, 1996.
[11] P. S. Gopalakrishnan, D. Nahamoo, M. Padmanabhan, and M. A. Picheny, "A channel bank based phone detection strategy," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, Adelaide, Australia, Apr. 1994, pp. 161–164.
[12] H. L. Van Trees, *Detection, Estimation, and Modulation Theory. Part I*. New York: Wiley, 1968.
[13] *Intel Architecture Software Developer's Manual*, 1999.
[14] H. Jiang, F. K. Soong, and C.-H. Lee, "A dynamic in-search data selection method with its applications to acoustic modeling and utterance verification," *IEEE Trans. Speech Audio Process.*, to be published.
[15] L. R. Rabiner, J. G. Wilpon, and B.-H. Juang, "A segmental K-means training procedure for connected word recognition," *AT&T Bell Labs Tech. J.*, vol. 65, no. 3, pp. 21–31, 1986.
[16] B.-H. Juang, W. Chou, and C.-H. Lee, "Minimum classification error rate methods for speech recognition," *IEEE Trans. Speech Audio Process.*, vol. 5, no. 3, pp. 257–265, May 1997.
[17] A. E. Rosenberg, O. Siohan, and S. Parthasarathy, "Speaker verification using minimum verification error training," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, Seattle, WA, May 1998.
[18] L. Nguyen and R. Schwartz, "Single-tree method for grammar-directed search," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, Phoenix, AZ, 1999.
[19] O. Siohan, A. Ando, M. Afify, H. Jiang, C.-H. Lee, Q. Li, F. Liu, K. Onoe, F. K. Soong, and Q. Zhou, "A real-time japanese broadcast news closed-captioning system," in *Proc. Eur. Conf. Speech Communication Technology*, Aalborg, Denmark, Sep. 2001, pp. 495–498.
[20] W. Reichl and W. Chou, "Robust decision tree state tying for continuous speech recognition," *IEEE Trans. Speech Audio Process.*, vol. 8, no. 5, pp. 555–566, Sep. 2000.
[21] F. Wessel, R. Schluter, K. Macherey, and H. Ney, "Confidence measures for large vocabulary continuous speech recognition recognition," *IEEE Trans. Speech Audio Process.*, vol. 9, no. 3, pp. 288–298, Mar. 2001.

Feng Liu received the B. Eng., M. Eng., and Ph.D. degrees in electrical engineering, in 1991, 1994, and 1998, respectively, all from Xian University, China.

From 1998 to 2000, he worked as a Postdoctoral Fellow in Tsinghua University, Beijing, China. He was with the ECE Department, Duke University, Durham, NC, as a Postdoc from March to June 2000. From June 2000 to May 2001, he was a Consultant at the Multimedia Communication Research Lab, Bell Labs, Lucent Technologies, Murray Hill, NJ. From June 2001 to May 2002, he was with the CAIP Center, Rutgers University, New Brunswick, NJ. Since July 2002, he has been with Avaya Labs Research, Avaya, Inc., Basking Ridge, NJ, as a Research Scientist. His current research interests include speech recognition, natural language understanding, dialogue systems, multimodal interaction, and converged communications.

Hui Jiang (M'00) received the B.Eng. and M.Eng. degrees from University of Science and Technology of China (USTC) and the Ph.D. degree from the University of Tokyo, Tokyo, Japan, in September 1998, all in electrical engineering.

From October 1998 to April 1999, he worked as a Researcher in the University of Tokyo. From April 1999 to June 2000, he was with Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, as a Postdoctoral Fellow. From 2000 to 2002, he was with Dialogue Systems Research, Multimedia Communication Research Lab, Bell Labs, Lucent Technologies Inc., Murray Hill, NJ. In 2002, he joined the Department of Computer Science, York University, Toronto, ON, Canada, as an Assistant Professor. His current research interests include all issues related to speech recognition and understanding, especially robust speech recognition, utterance verification, adaptive modeling of speech, spoken language systems, and speaker recognition/verification.

Mohamed Afify was born in Cairo, Egypt, in 1964. He received the B.Sc. (with distinction), M.Sc., and Ph.D. degrees from the Department of Electronics and Communications of Cairo University in 1987, 1992, and 1995, respectively.

From 1989 to 1996, he was a Research Associate at the National Telecommunication Institute, Cairo. From 1996 to 1998, he was a Postdoctoral Research Fellow with the Speech Recognition Group at INRIA Lorraine, Nancy, France. From 1998 to 2000, he was Assistant Professor with the Department of Electrical Engineering, Fayoum Branch, Cairo University. From 2000 to 2002, he worked for the Dialogue Systems Research Department, Bell Laboratories, Murray Hill, NJ. Since 2002, he has been Associate Professor with the Faculty of Information and Computers, Cairo University. In March 2004, he joined BBN Technologies, Cambridge, MA. His research interests are in statistical modeling, pattern recognition, and digital signal processing with a particular emphasis on their application to speech and language processing.

Olivier Siohan (M'97) received the M.S. degree in electrical engineering in 1991 from Supelec Electrical Engineering Institute, France. He received the M.S. and Ph.D. degrees in computer science, both from the University of Nancy, France, in 1991 and 1995, respectively.

In 1995, he joined AT&T Bell Laboratories in a postdoctoral position to carry out research on robust speech recognition. From 1996 to 1998, he was with AT&T Labs working on speaker recognition. From 1998 to 2002, he was a Member of Technical Staff at Lucent Technologies—Bell Labs, Murray Hill, NJ, where he led a broadcast news speech recognition project. In 2003, he moved to IBM T. J. Watson Research Center, Yorktown Heights, NY, as a Research Staff Member. His major research interests are speech and speaker recognition, acoustic modeling, speaker adaptation, and noise robustness. He is the author of over 45 articles in these areas.