A Fast Approximate Acoustic Match for Large Vocabulary Speech Recognition

L. R. Bahl, S. V. De Gennaro, P. S. Gopalakrishnan, R. L. Mercer

Speech Recognition Group, IBM Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, NY 10598, USA

ABSTRACT
In a large vocabulary speech recognition system, a great deal of computer time is spent performing detailed acoustic matches of words in the vocabulary. In order to run in real time on a modest amount of hardware, it is important that these detailed matches be performed only on words which have a reasonable probability of being correct. We describe a scheme for rapidly obtaining an approximate acoustic match of all of the words in the vocabulary in such a way as to ensure that the correct word is, with high probability, one of a small number of words examined in detail.

We give experimental results showing the effectiveness of the fast match that we describe for a number of talkers. In particular, we give the average rank of the correct word, the average number of words suggested, and the number of times that the correct word is not among the words suggested.

1. INTRODUCTION
In a large vocabulary speech recognition system, a great deal of computer time is spent performing detailed acoustic matches of words in the vocabulary. In order to run in real time on a modest amount of hardware, it is important that these detailed matches be performed only on words which have a reasonable possibility of being correct.

We describe here the method used in a speech recognition system developed at IBM. Other aspects of the system and its performance have been reported elsewhere[1, 2, 3]. Briefly, the system employs hidden Markov models of the words in the vocabulary to perform a detailed acoustic examination of the utterance and a trigram model of the language to guide its search for the most probable interpretation of the utterance as a sequence of words from the vocabulary. During a training session, the talker reads a script of approximately 1100 words from which we estimate the parameters of the hidden Markov models using several iterations of the forward-backward algorithm. We estimate the parameters of the trigram language model from roughly 200 million words of English text.

In the next section, we describe a rapid calculation which allows us to preview each word of the vocabulary to gauge its acoustic feasibility and in the following section, we incorporate the language model to limit the resulting list of acoustically reasonable words to words which are not only acoustically reasonable but also linguistically acceptable. In section 4, we give results showing the effectiveness of the complete fast match for a number of talkers.

2. ACOUSTIC FAST MATCH
The probability that a particular acoustic unit produces a given sequence of phonemes is the sum over all complete paths in the Markov model for the unit of the a priori probability of the path times the probability of the phoneme string given the path. A path through a Markov model is simply a string of arcs, \( a_1, a_2, \ldots, a_n \) and a phoneme string is a sequence of vector quantization labels. Thus, for a unit \( u \) and phoneme string \( f = f_1, f_2, \ldots, f_n \) we have"
more acoustic units. For very brief acoustic units, the effect of
replacing \( p(t) | a \) by \( m(t) \) will be small and so \( F_p(t) \) will be a
good approximation to \( D_p(t) \). At the same time, each word
will be made up of many acoustic units and, as a result, computing
the fast match score will not be much faster than computing the
detailed match score for the word. If the acoustic units are long,
then replacing \( p(t) | a \) by \( m(t) \) may be a drastic over-estimate
but computing the acoustic fast match score for a word will be
much faster than computing the detailed match score. A
successful fast match, therefore, depends on a choice of acoustic
unit which allows sufficiently rapid computation while not
rendering the acoustic fast match score useless. In the results
that we describe below, we have used the phone as the acoustic
unit for the fast match computation. With this choice, the fast
match score for a word can be computed between 30 and 40
times as quickly as the detailed match score for the word.

We refer to the sequence of phones that makes up a word as the
phonetic baseform for the word. Many words in a large
vocabulary will begin with the same initial sequence of phones.
As pointed out by Klovstaad[4], we can profitably arrange the
phonetic baseforms for all of the words in the vocabulary into a
tree as shown schematically in Figure 1. Each of the leaves of
the tree corresponds to a word. Each of the other nodes in the
tree corresponds to a common initial phonetic sequence of the
baseforms at the leaves of the subtree that depends from it.

Given a feneme string \( f \), consider the problem of determining
the acoustic fast match score for each of the leaves of the
phonetic tree. Let \( t \) be a node of the tree and let \( u(t), \ldots, u(w) \) be
the corresponding phone sequence. We write \( F(t) \) as a short-hand
notation for \( F_{\text{phones}}(u(t), \ldots, u(w)) \). We call the vector \( (F(t), F(w), \ldots) \)
the end-time distribution for node \( t \). Let \( t \) and \( w \) be two nodes in the
tree and let \( s \) be the daughter of \( t \) with phone sequence \( u(t) \ldots u(s) \).
Then

\[
F_s = \sum_{k=0}^{l} F_k F_s(t_k). \tag{5}
\]

We can organize an efficient computation of the end-time
distribution for each node by taking the end-time distribution of
the root to be \((1, 0, 0, \ldots)\) and then applying equation (5) to
obtain the end-time distributions of all of the daughters of the
root and so forth. The \( n \)th element of the end-time distribution
for a leaf is then the acoustic fast match score for the leaf.

Because \( m(t) \) is never greater than 1 and usually quite a bit less,
successive elements of the end-time distribution for a given node
are progressively smaller. For those nodes corresponding to
phone sequences that are not acoustically close to the true phone
sequence, the decay will be much more rapid than for those
corresponding to phone sequences which are phonetically
dense to the true phone sequence. It is a waste of time to pursue
paths in the tree which do not match well and so we would like
to be able to judge from the end-time distribution at a node
whether we should continue the calculation for its daughters.
To do this, we estimate, during training, \( e(t) \), the expected value for
\( m(t) \) when \( u \) is the correct phone. We can then compare each
end-time distribution to the expected end-time distribution,
(1, \( e(t) \), \( e(t_1) \), \( e(t_2) \), \ldots), and abandon the node if the end-time
distribution is falling too quickly. The expected end-time
distribution depends only on the feneme sequence and so can be
computed without any knowledge of what phones are actually
present.

By computing the acoustic fast match scores on a tree and
curtiling the computation for nodes which do not compare
favorably with the expected end-time distribution, we are able to
reduce the total computation by a factor of about three so that
allegedly the computation of the acoustic fast match score is
about 100 times as fast as the computation of the detailed match
score and produces a list of several hundred words which are
candidates for further investigation.

3. INCLUDING THE LANGUAGE MODEL

The computation outlined in the previous section gives us a list
of words which are acoustically similar to the correct word. The
list may, however, include words which are unlikely on linguistic
grounds. For example, if the sentence up to some point runs ...
... of the' and the next word is see, then the list of acoustically
similar words might well include such linguistically unreasonable
members as the, she, and me. We combine the acoustic fast
match score with a score obtained from our trigram language
model to obtain a complete fast match score which incorporates
both acoustic evidence and linguistic evidence to rank possible
next words. Let \( p(w) \) be the \( a \text{ priori} \) probability assigned by the
language model to the word \( w \) following the last two words of
the sentence thus far. Then the complete fast match score, \( c(w) \), is
given by

\[
c(w) = \max_i \{ c_i(w^n) \}.
\]

The parameter \( \alpha \), here, which we choose between 0 and 1, allows
us to alter the relative importance of the linguistic and acoustic
contributions to the complete fast match score. As we shall see
in the next section, we can use the complete fast match score to
provide a list of a few tent of candidate words among which the
correct word can be found with very high probability.

4. RESULTS

We have used the fast match scheme described above for four
male talkers using the 20,000 word vocabulary and the
recognition system of reference [1]. Each talker read a script of
1696 words with pauses between the words. Figure 2 is a table
showing the performance of the acoustic fast match alone for
these 1696 words. The average list length across all four talkers
is 852 words among which the correct word is found, on average,
99.8% of the time. Figure 3 is a table showing the performance
of the complete fast match for the four talkers. Here, we have
used the trigram language model to further prune the acoustic
fast match list. The average list length across the four talkers
is 14.6 words and the correct word is in the list 99.3% of the time,
usually in position 1 or 2. When the correct word is not on the
fast match list, we are sure to make an error in decoding since
only these words are explored in the detailed match. We note,
however, that not all of these errors can be attributed to the fast
match. Approximately half the time when a word is not on the
fast match list, the detailed match would have rejected the word
even if the fast match had suggested it. These are places where
either the language model will not permit the word even though
the detailed match score is good, or the detailed match score
itself is bad. Thus, the use of the complete fast match for these
four talkers increases the error rate by no more than .35%.

ACKNOWLEDGEMENT

We would like to thank the other members of the Speech
Recognition group at the IBM Research Center for their help
and encouragement.

REFERENCES

1. A. Averbach, L. Bahl, R. Bakis, P. Brown, G. Daggert, S.
Das, K. Davie, S. De Gennaro, P. de Souza, E.
Epstein, D. Fraleigh, F. Jelinek, B. Lewis, R. Mercer,
M. Moorehead, A. Nadas, D. Nahamoo, M. Picheny, G.
Shihman, P. Spinelli, D. Van Compernolle, and H.
Willems. "Experiments with the TANGORA 20,000 word
speech recogniser". Proceedings of the 1987 IEEE
International Conference on Acoustics, Speech, and Signal
Processing, Dallas, Texas, pages 701-704, April 1987.
2. L.R. Bahl, P.F. Brown, P.V. de Souza, R.L. Mercer,
and M.A. Picheny. "Acoustic Markov models used in the


---

**Figure 1. Fast Match Search Tree**

<table>
<thead>
<tr>
<th>Talker 1</th>
<th>Talker 2</th>
<th>Talker 3</th>
<th>Talker 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average list length</td>
<td>794.6</td>
<td>1157.8</td>
<td>682.8</td>
</tr>
<tr>
<td>Correct word not in list</td>
<td>0.41% (7/1696)</td>
<td>0.06% (1/1696)</td>
<td>0.12% (2/1696)</td>
</tr>
</tbody>
</table>

**Figure 2. Acoustic Fast Match Performance.**

<table>
<thead>
<tr>
<th>Talker 1</th>
<th>Talker 2</th>
<th>Talker 3</th>
<th>Talker 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average rank of correct word</td>
<td>1.37</td>
<td>1.41</td>
<td>1.27</td>
</tr>
<tr>
<td>Average list length</td>
<td>14.47</td>
<td>16.16</td>
<td>12.38</td>
</tr>
<tr>
<td>Correct word not in list</td>
<td>0.82% (14/1696)</td>
<td>0.77% (13/1696)</td>
<td>0.24% (4/1696)</td>
</tr>
</tbody>
</table>

**Figure 3. Complete Fast Match Performance.**