Detection of ambiguous portions of signal corresponding to OOV words or misrecognized portions of input

Roxane Lacouture[1], *Yves Normandin*[2]
lacout@crim.ca, normandin@locus.ca

[1]Centre de recherche informatique de Montréal (CRIM)
[2]Locus Speech Corporation
1801, McGill College, suite 800
Montréal, Québec, Canada
H3A 2N4

ABSTRACT

One of the key problems for large-vocabulary ASR is the detection of unknown or misrecognized portions of the input. This paper presents results obtained using a local rejection algorithm. The algorithm is derived from the two-pass recognition algorithm by Murveit [3] and is used to detect misrecognized portions based on the number per frame of active words during the second pass. The hypothesis underlying the algorithm is that recognition on unexpected data, i.e. noise or out-of-vocabulary (OOV) words, is likely to result in activation of more words, since no word matches the data well; on the other hand, when the match is good, fewer words should be active.

The algorithm was tried on part of the WSJ 5K November 1993 test, in which there were no OOV words (3370 words in total) and on the digit-strings-only Macrophone data (14686 words of which 895 were OOV).

The results obtained indicate that our approach is promising, both for the detection of OOV words and misrecognized portions of the input. It may provide the base on which to build tools for dealing with these phenomena. These tools might include dialogue mechanisms based on the list of activated words corresponding to a rejected portion, display mechanisms such as reverse video or rescoring schemes.

1. INTRODUCTION

Although the performance of ASR systems is now very good even for very large vocabulary, there are still important problems which need to be resolved in some acceptable way. Among these problems are adaptation to unmatched acoustical conditions, the detection of unknown input and the detection of OOV words or misrecognized portions of input. In a previous paper [1], we presented solutions based on the same approach to the first two problems. This paper concentrates on the last problem.

The problem of detecting OOV words has been traditionally tackled by approaches similar to those used in keyword spotting. The use of garbage models [4] is one of the most popular of these approaches. However, the approach here is different. No special model is used. Instead each word is used as a garbage model for all other words, since it is the local score difference between the chosen word and the second (or more) best choice which is used as rejection criterion. The hypothesis behind this is that recognition on unexpected data is likely to result in many "medium" hypotheses instead of one "strong" hypothesis. The way this idea is implemented is by first computing a word lattice and then looking at the density of this lattice at each time frame.

2. THE LOCAL REJECTION ALGORITHM

The local rejection algorithm is derived from the two pass recognition algorithm by Murveit [3]. This algorithm consists of a forward pass followed by a backward pass. In the forward pass, all active word ending likelihoods are kept in a vector (forward scores). The number of scores kept is controlled by the value of the forward beam search. In our case we used two such beams: a general beam and an end word beam [2].

In the backward pass, the forward scores are combined with the backward scores to compute the "global score" of every active word pair. This global score is equivalent to the probability of the best word sequence containing this word pair. Then, the word pairs with global score within a certain beam width of the most likely sentence (known as the heuristic beam, or forward-backward beam) are used to build a lattice. This lattice is finally used to compute a density array on which is based the local rejection. The detailed steps of the whole process follow.

2.1. Forward/Backward Pass

The forward pass is a standard Viterbi in which

- at each frame t, α_{tm_i} the score at the end of each active word m_i is memorized;
- α_{max} the score of the forward pass solution is also memorized.

Then the backward pass, a standard Viterbi, is performed only on word pairs m_i and m_j for which

- $\alpha_{max} - \Delta_H < H + bigram(m_i, m_j) < \alpha_{max}$

where:

- Δ_H is called the heuristic beam,

- $H = \alpha_{tm_i} + \beta_{tm_j}$ is called the heuristic score,
- β_{tm_j} is the backward score at the beginning of word m_j at frame t.

2.2. Lattice Building

For each valid word pair (m_i,m_j) of the backward pass, a list of activation periods is kept. Each activation period contains:

- $sleft$:the start of the activation period between the two words
- $eleft$:the end of the activation period
- $bestright$:the activation time frame of m_j by a given m_x for which $\Delta(m_i, m_j)$ is maximum and where $\Delta(m_i, m_j)$ is defined as $H + bigram(m_i, m_j) - \alpha_{max}$:

The value of these variables is set for a given word pair (m_i,m_j) active at frame t according to the following rules:

- if an activation period already exists for these two words and $t = sleft - 1$, $sleft$ takes t as new value and if needed both $\Delta(m_i, m_j)$ and $bestright$ are modified
- if no activation period exists or $t \neq sleft - 1$ a new activation period is created for the same pair
- $bestright$ is obtained by memorizing, for each word, its best predecessor and start frame.

2.3. Density Array

The following pseudo-code is used to fill out a density array using the activation data cumulated in the second pass. It uses a stack to store activation word period with which is associated the following information: $start$, end (frame numbers) and wno (word number). b is a special symbol indicating the beginning of the signal, e a special symbol indicating the end of the signal.

```
for all (b,m_j) activation data
    push_on_stack(0, bestright, m_j)
while stack_not_empty
    pop (start, end, wno);
    fill_density_array(start, end);
    if (wno != e)
        for all ((wno,m_j) activation data
            if (sleft(wno, m_j) < end < eleft(wno, m_j))
            push_on_stack(end, bestright, m_j)
```

2.4. Rejection Criteria

Rejection is performed using the segmentation of the top-scoring word sequence hypothesis and the density array. That is, for each word of this solution, the number of frames for which the word count density is higher than a given value d is computed. Then, if the ratio between this number of frames and the total duration of that word in frames is higher than a given threshold (0.6 in our test), the word is replaced by a special alt symbol.

The alt symbol is not counted as an insertion and can match any number of words in the reference string in order to decrease the total error rate (see section 3.2. for concrete examples). Clearly the use of the alt symbol is only a mechanism to estimate the error rate. In real applications, other more appropriate treatment of the rejected words would have to be built. Section 4. gives some examples.

3. RESULTS

This local rejection algorithm was applied to two databases: the digit data of Macrophone and part of the WSJ 5K November test data 1993.

3.1. Digit Macrophone Data Results

The Macrophone digit data were used to compare our local rejection algorithm to a more or less classical garbage model approach. In this approach, a one distribution garbage model is used. The garbage model unique distribution is computed at each frame. Its value is equal to the maximum distribution for that frame minus a given threshold. That means that its value varies from one sentence to the other and inside the same sentence. The smaller the threshold, the more rejection will occur, the bigger the threshold, the less rejection will occur.

In the test using the Macrophone digit data, words like “area code”, “dash”, “tack” were considered OOV, i.e. the acceptable vocabulary was reduced to include only the 10 digits. There were 895 of them. The HMMs used were discrete tree based states tied distribution.

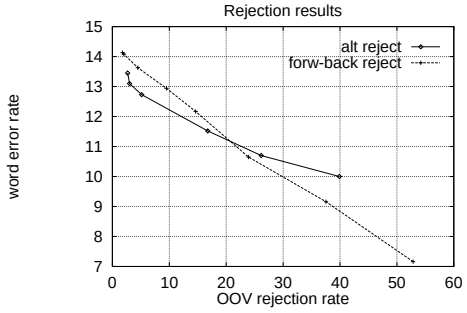

Figure 1: Comparison between the OOV detection performance of the lattice backward approach and the garbage approach. The error rate is obtained by subtracting the number of OOV from the total

number of words, by counting missed OOV words as insertions and non-OOV words as deletions.

In figure 1, we compare the performance of the two methods in detecting OOV words. As can be seen, both methods obtain more or less similar results. But OOV word detection is not the only evaluation criterion to consider, so in figure 2, we also looked at the non-OOV word rejection.

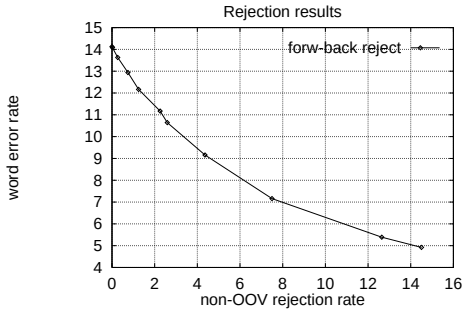

Figure 2: Detection of non-OOV by the lattice backward approach. The error rate is obtained as in figure 1.

However, the non-OOV words results may be misleading. The fact that a non-OOV is rejected by the algorithm does not mean the algorithm is wrong. It may be that this vocabulary word is not at all the one which has been pronounced. In that case, its rejection is not really a mistake. We thus decided to compute the error and rejection rate in another way, which is detailed in the next section.

3.2. WSJ Results

Since the distinction between OOV words and non-OOV words is no longer relevant, the rejection rate is simply defined as the proportion of words in the reference string which are in a "bad match" portion of the signal. The word error rate is then computed only on the remaining portion of the signal. The example below illustrates the process:

```
Spoken:
UNDER THE OLD PLAN FEDERAL EMPLOYEES WEREN'T
ELIGIBLE FOR SOCIAL SECURITY BENEFITS

Solution without rejection:
UNDER THE OLD PLAN FEDERAL EMPLOYEES WHO
AREN'T ELIGIBLE FOR SOCIAL SECURITY BENEFITS
          1 ins 0 del & 1 subs
Solution with rejection:
UNDER THE OLD PLAN FEDERAL EMPLOYEES alt
ELIGIBLE FOR SOCIAL SECURITY BENEFITS
          0 ins 0 del & 0 subs
```

Rejection rate of this sentence is 2/13, its original error rate 2/13 and its resulting error rate 0 (**WHO AREN'T** has been correctly rejected). Note that the error rate may also increase. For example, in the possible other solution given below, the rejection rate of the sentence is 1/13 and its resulting error rate 2/12 which is worse (**THE** has been falsely rejected).

```
Solution with rejection:
UNDER alt OLD PLAN FEDERAL EMPLOYEES WHO
AREN'T ELIGIBLE FOR SOCIAL SECURITY BENEFITS
          1 ins 0 del & 1 subs
```

Using this way of estimating the rejection and error rate, the results of using the density array algorithm on 212 sentences of the 5k 1993 November test (3370 words) are illustrated in figure 3. A rejection rate of 5.76% with an error rate of 19.18% yielded a 17.26% reduction in word error in relation to an original 23.18% error rate (= (23.18-19.18)/23.18*100), while a rejection rate of 23.62% with an error rate of 10.18% yielded a 56.08% reduction.

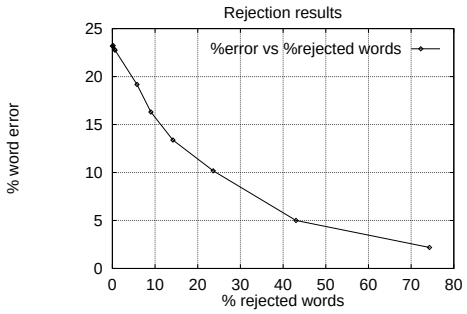

Figure 3: Rejection results on WSJ data.

The HMMs used were discrete tree based states tied models trained (MLE training) on SI-284.

We can then also better illustrate (figure 4) the real impact of the approach on the Macrophone data.

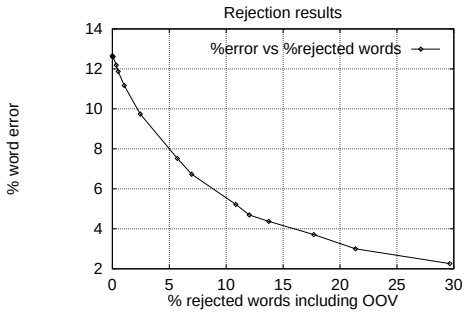

Figure 4: Rejection results on Macrophone data.

On the digit-only Macrophone data, a rejection rate of 5.71% with an error rate of 7.52% yielded a 40.60% reduction in word error rate (= (12.66-7.52)/12.66*100), and a rejection rate of 21.36% with an error rate of 3.00%, a 76.30% reduction in word error rate in relation to an original 12.66% error rate.

4. FUTURE WORK

The next logical step will be to assess the potential of this local rejection algorithm for making corrections. Since the main advantage of the density array approach is due to the fact that it can offer alternatives to the rejected portion of signal, it would be logical to see if those alternatives may be used to correct the misrecognized portions. One approach would be to rescore, using higher level knowledge like trigrams, the different alternatives found in the lattice. Although lattice rescoring is common practice, it is always demanding in time since the entire lattice has to be rescored. The rescoring of only portions of it could accelerate the process.

In more interactive applications, dialogue mechanisms based on the list of activated words corresponding to a rejected portion or display mechanisms such as reverse video and pop-up list could be considered.

Finally, in hopes of increasing its rejection capability, the same rejection algorithm could be coupled with better lattice building algorithms such as, for example, the word-dependent N-best [3].

5. REFERENCES

[1] R.Lacouture, Y. Normandin, H.M. Cung, "Using Two Pass Recognition for Adaptation or Rejection in Real-Time Applications", *IEEE Automatic Speech Recognition Workshop*, pp. 171-172, 1995.

[2] R. Lacouture, Y. Normandin, "Efficient Lexical Access Strategies", *Eurospeech 93*, Vol. 3, pp. 1537-1540, 1993.

[3] R. Schwartz, S. Austin, "A Comparison of Several Approximate Algorithms for Finding Multiple (N-BEST) Sentence Hypotheses", *ICASSP 91*, pp. 701-704.

[4] H. Murveit, J. Butzberger, V. Digalakis, M. Weintraub, "Large-Vocabulary Dictation Using SRI's Decipher Speech Recognition System: Progressive Search Techniques", *ICASSP 93*, pp. 319-322, 1993.

[5] J.G. Wilpon, L.R. Rabiner, C.H. Lee, E.R. Goldman, "Automatic Recognition of Keywords in Unconstrained Speech Using Hidden Markov Models", *IEEE Trans. on ASSP*, Vol. 38, No 11, pp. 1870-1878, Nov. 1990.