New Improvements in Decoding Speed and Latency for Automatic Captioning

Jian Xue, Rusheng Hu and Yunxin Zhao

Department of Cmputer Science
University of Missouri, Columbia, MO 65211 USA
{jxwr7, rhe02}@mizzou.edu, zhaoy@missouri.edu

ABSTRACT

In this paper, we present new improvements in decoding speed and latency for automatic captioning in telehealth. Complementary local word confidence scores are used to prune uncompetitive search paths. Subspace distribution clustering hidden Markov modeling (SDCHMM) is used for fast generation of acoustic and local confidence scores, where overlap accumulative probability (OAP) is used to measure the similarity of Gaussian *pdf*'s in SDCHMM. We propose to use pre-backtrace based on detection of prosodic boundaries defined by unfilled pauses, filled pauses, as well as pitch contour to decrease latency. Experiments were conducted on a telehealth captioning task with vocabulary sizes of 21 K and 46 K. The proposed methods led to 33% improvement in decoding speed without loss of word accuracy, and to 3 folds of decrease in maximum latency with about 1.6% loss of word accuracy.

Index Terms: confidence-based pruning, P-value, decoding latency, pre-backtrace, prosodic features

1. INTRODUCTION

Decoding time efficiency has been an important issue in automatic speech recognition (ASR) systems. Usually, the time efficiency of a decoding engine is measured by the criterion of real-time factor. However, for interactive online conversational systems, such as telehealth automatic captioning [1], a system needs to provide other functions in addition to decoding, like speech stream separation, speaking rate estimation, confidence annotation, and so on. Since the overall processing time should be less than 1.0×real time, in such a system, a real-time decoding engine is not fast enough.

Decoding can be viewed as finding an optimal path in a search network. It is intractable to perform an exhaustive search through a huge network in large vocabulary continuous speech recognition (LVCSR) [2]. To make search feasible, the search space needs to be reduced by pruning. Beam pruning is probably the most important pruning criteria used in LVCSR decoders, which retains only paths with likelihood scores deviate from the score of the best partial path hypothesis within a beam width [3]. Another common pruning criterion is histogram pruning, which limits the number of active paths by retaining only a predefined number of best paths. Besides of beam and histogram prunings, quickly computing acoustic score and accessing language score is another way to speed up decoding.

Dynamic programming based decoding algorithms utilize backtracking and determine the best path, i.e., the recognition result, until reaching the end of the input. So the latency, i.e., the time lapse from a speaker starting to talk till recognition result being outputted, is at least the duration of the utterance. In telehealth, doctors can talk without a noticeable break for tens of seconds. Such a long utterance contains quite a few sentences and sometimes with many filled pauses. Our speech stream separation module [1] can detect low-energy un-filled pauses to break speech inputs into utterances, however, excessively long utterances remain in the separation output. Such long inputs to a decoder reduce decoding speed and increase latency, since as the inputs get longer, more search paths will be expanded and more memory will be consumed, and the time spent in waiting for recognition outputs will be increased. As the consequence, conversations between doctor and patient become sluggish.

Our decoding engine performs large vocabulary continuous speech recognition via one-pass time-synchronous Viterbi beam search with a tree-copy based search organization [12]. The current work focuses on improving our decoding engine to meet the requirement of online conversation captioning in telehealth. We propose three methods to speed up the decoding engine and decrease its latency. First, we propose to use complementary local word confidence scores to prune unpromising candidate paths during search, based on local word posterior probability score and word P-value [6]. Second, subspace distribution clustering hidden Markov modeling (SDCHMM) [5] is used to speed up the generation of acoustic scores and local word confidence scores, where in Gaussian density clustering for SDCHMM, we propose to use overlap accumulative probability (OAP) to measure the similarity of Gaussian *pdf*'s. Third, in order to decrease latency, we propose to use pre-backtrace to output recognition results incrementally prior to reaching the end of input by using as prosodic cues unfilled pauses, filled pauses, as well as pitch contour for pre-backtrace. It is worth noting that unlike read speech problems dealt with in [4], where low energy pauses were used as cues for pre-backtrace, in conversational speech, energy cue alone is insufficient and therefore more prosodic cues are needed.

The rest of the paper is organized as the following. Section 2 describes how to use the local word confidence scores of P-value and local word posterior probability (LWPP) as a path pruning criteria. Section 3 describes the use of OAP to cluster Gaussian *pdf*'s for SDCHMM to speed up the generation of acoustic score and local word confidence score. In section 4, the method of performing pre-backtrace for long inputs based on prosodic cues is described. In section 5 we present experimental results. We conclude the work in section 6.

2. CONFIDENCE-BASED PRUNING

Confidence-based pruning was proposed in [7] to guide the search for most promising paths in addition to beam and histogram prunings. In this technique, confidence scores play the role of an online filter that is applied to the word level partial hypotheses to aid the decision of whether to consider them for further path expansions or to discard them from the search space. In [7], only the feature of LWPP was used. Here we use two complementary confidence scores. One is word level P-value, another one is LWPP.

This work is supported in part by National Institutes of Health under the grant NIH 1 R01 DC04340-01 A2.

2.1 P-value

P-value was first used in confidence measure in [6]. For a one-dimensional Gaussian distribution $N(\mu, \sigma^2)$ with a known σ^2 and an unknown μ, one may test if μ equals μ_0 or not. Given an observation x, P-value is defined as $P_v(x) = Prob(|X-\mu| > |x-\mu| \,|\mu=\mu_0)$ which is the shaded area in Fig. 1. The larger $P_v(x)$ is, the higher confidence we have that μ equals μ_0.

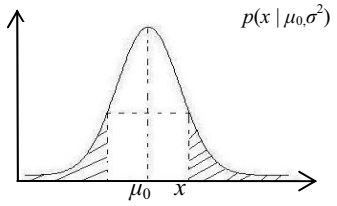

Fig. 1 P-value for a Gaussian distribution (shaded area)

For classification, we need to test if an observation x belongs to some class C. If the class-conditional distribution is a Gaussian distribution $N(\mu_0, \sigma^2)$, we can use $P_v(x)$ as a confidence feature.

In speech recognition, a tied state of context-dependent triphone HMMs is modeled by a multivariate Gaussian mixture density (GMD), and diagonal co-variance matrix is often used for the component Gaussian densities. We therefore define the P-value $P_{v,i}(\underline{x})$ for the ith n-dimensional Gaussian density of a GMD as the product of the P-values of individual dimensions. The P-value of an I-sized n-dimensional Gaussian mixture distribution is then defined as

$$P_v(\underline{x}) = \max_i(w_i P_{v,i}(\underline{x})) \quad \text{or} \quad P_v(\underline{x}) = \sum_{i=1\ldots I} w_i P_{v,i}(\underline{x}),$$

where w_i is the ith mixture weight. Our empirical evaluation showed that there was insignificant difference between these two definitions, and in the current work the latter definition is used. For a word w in a search path, denote x_t, $t = 1,2,\ldots,T$ as the corresponding acoustic observation sequence and M_{i_t}, $t = 1,2,\ldots,T$ as the corresponding model sequence. We then define the log P-value for the word w as

$$P_v(w) = \sum_{t=1}^{T} \log(P_v(x_t \mid M_{i_t})) \tag{1}$$

2.2 LWPP

The feature LWPP was proposed by Fu et al [8]. To define LWPP, the posterior probability of the state s_i conditioned on the observation x is defined as

$$p(s_i \mid x) = \frac{p(x \mid s_i)p(s_i)}{p(x)} = \frac{p(x \mid s_i)p(s_i)}{\sum_{s_j \in D} p(x \mid s_j)p(s_j)} \tag{2}$$

where D is the set of states survived after pruning. By assuming that the prior probabilities of all states are uniform, formula (2) is simplified as

$$p(s_i \mid x) = \frac{p(x \mid s_i)}{\sum_{s_j \in D} p(x \mid s_j)} \tag{3}$$

Assuming for the word w the state sequence and the observation sequence to be $s_{i_1},\ldots, s_{i_T}$ and $x_1, \ldots x_T$, the LWPP of w is defined as:

$$LWPP(w) = \log[\prod_{t=1}^{T} p(s_{i_t} \mid x_t)] \tag{4}$$

Although $P_v(w)$ and $LWPP(w)$ both provide confidence information for the word w, they are different. P-value captures information of distribution spreadness more effectively, while $LWPP(w)$ catches the difference of acoustic scores among the path candidates. A competitive path should have both large $P_v(w)$ and $LWPP(w)$.

In the search process, at every time frame t, each word w that reaches its final state will be evaluated by confidence scores of $P_v(w)$ and $LWPP(w)$. These two values reflect how well the hypothesized word can account for the acoustic evidence. The better the acoustic evidence is supported by a hypothesized word, the higher the confidence that the word is in fact correct. If $P_v(w)$ or $LWPP(w)$ is smaller than their predefined thresholds, the corresponding path will be pruned.

3. SDCHMM

SDCHMM was first introduced by Bocchieri and Mak [5], with the aim of building compact acoustic models. Since in LVCSR, the number of physical models is very large, computing acoustic scores and P-values consume a lot of time, especially P-value, where the time needed to compute a P-value is several times more than that for computing an acoustic score. Although we use table-lookup to speed up the computation of P-values, prior to lookup we need to normalize the observation $\underline{x}$ for each physical model to have zero mean and unit standard deviation, which is still a heavy burden due to the large number of physical models. If SDCHMM is used, then the Gaussian density scores of all continuous density hidden Markov models (CDHMMs) can be approximated by certain combinations of a small number of subspace distribution prototypes. In the decoding process, all these subspace Gaussian log likelihoods and log P-values can be precomputed once for each dimension at the beginning of each frame, and their values are stored in lookup tables. Then for each Multivariate Gaussian in a physical model, the log likelihoods and log P-values can be computed as the summation of precomputed values in different dimensions together with the log mixture weight. So SDCHMM is computationally efficient.

To convert CDHMM to SDCHMM, a clustering of Gaussian densities in each feature dimension is made and therefore a proper measure of similarities of the *pdf*'s is needed. The classification-based Bhattacharyya distance (CBB) was used in [5]. Li et al proposed Kullback-Leibler divergence (KLD) for SDCHMM [9]. For Gaussian distributions, it was found that KLD overstates the difference between Gaussian densities when they have very different variances [11].

OAP was first proposed in [11], where it was used for Frame Discrimination (FD) training. Here we use OAP [11] as the similarity measure for clustering a Gaussian *pdf* f to a Gaussian prototype g, which is defined as:

$$S(f,g) = \int_{-\infty}^{+\infty} \min(f,g)dx \tag{5}$$

The value $S(f, g)$ is between 0 and 1. If f and g are the same, then $S(f, g)$ equals 1. For two Gaussian *pdf*s, $S(\cdot,\cdot)$ equals one of the shaded areas in Fig. 2.

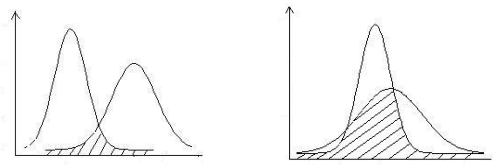

Fig. 2 Definition of OAP for two Gaussian pdfs.

The optimal prototype, $g(\mu_p, \sigma_p^2)$, of a cluster of N Gaussian distributions, $f(\mu_i, \sigma_i^2)$, $i = 1 \ldots N$, is obtained by the following formulas:

$$\mu_p = \frac{1}{N}\sum_{i=1}^{N}\mu_i \tag{6}$$

$$\sigma_p^2 = \frac{1}{N}\sum_{i=1}^{N}\{\sigma_i^2 + (\mu_i - \mu_p)^2\} \tag{7}$$

To initialize the iterative k-means clustering procedure for converting CDHMMs to SDCHMMs with K subspace Gaussian prototypes per dimension, we first split all the Gaussians into K supspaces, based on the mean values of the Gaussians. In each dimension, we order all the M Gaussians to be clustered with the increasing order of $\mu_i's$. We assign the first M/K Gaussians to cluster 1, the second M/K Gaussians to cluster 2, and so on, and compute the Gaussian prototype of each cluster with formulas (6) and (7). Subsequently, the algorithm iterates between cluster assignment and prototype update. In cluster assignment, each Gaussian is assigned to the cluster producing maximal $S(\cdot,\cdot)$. In prototype update, the formulas (6) and (7) are used. Usually after a few iterations the Gaussian prototypes will converge.

4. PRE-BACKTRACE DECODING

It is widely acknowledged that prosodic features play an important role in the definition and hence the detection of syntactic boundaries. Since most phonological rules are constrained to operate within a phrase, it is reasonable to consider pre-backtrace at syntactic boundaries which are often associated with prosodic boundaries.

Unfilled pause (or pause) is a well recognized feature of prosodic boundary. In spontaneous speech, on the other hand, filled-pause, such as AH, UM, is also a promising feature. Many filled pauses last for sufficient durations and therefore they serve as potentially good places for pre-backtrace. F_0 contour represents intonation. Dips in F_0 contours often coincide with prosodic boundaries and therefore they can be used in addition to energy for detection of syntactic boundaries. In [10], the F_0 contour is segmented at dips (minima) in the energy contour of the same speech utterance, and rules are applied to generate candidate syntactic boundaries at dips in the F_0 contour. We detect dips in F_0 contour by following the rules of [10]. We use autocorrelation of residues of linear prediction coding (order 16) to produce F_0. For each speech segment marked by a pair of dips in the energy contour, we fit the pitch contour by a recursive line with the method of least square error. If the error averaged over the segment exceeds a threshold, the segment is divided into two parts at the point where the maximum error occurs and a recursive line is fitted again for each part. The process iterates until we find a recursive line for each segment. All dips in F_0 contour obtained by this procedure are treated as potential prosodic boundaries.

With the aim of reducing latency in our decoding engine, we detect syntactic boundaries based on detections of short unfilled pauses, filled pauses, and dips in F_0 contour during one-pass speech decoding and use the cues to perform pre-backtrace. In acoustic modeling, one HMM model was trained for short pause, and nine HMM models were trained for nine patterns of filled pauses. In decoding, if short pause states rank as the best hypothesis consecutively with a duration exceeding a threshold, then the boundary is accepted; similarly, if the last state of a filled pause HMM ranks as the best hypothesis consecutively for several time frames, then the boundary is accepted; finally, if a prosodic boundary is detected at some point due to a dip in F_0 contour, and the last state of some word ranks as the best hypothesis for several consecutive frames, then the boundary is also accepted. Once a prosodic boundary is accepted, the decoder starts backtracking, finds the best partial path and outputs it. Decoding then continues by using the fixed word history that includes the last several words on the best partial path just produced.

5. EXPERIMENTAL RESULTS

Experiments were conducted on the telehealth captioning system, using five doctors' datasets, Dr. 1 through Dr. 5. The improvements on speed and latency are made on the decoding engine TigerEngine 1.1 [12]. The captioning task has a vocabulary size of 46,489, with 3.07% of vocabulary words being medical terms. Five speaker dependent (SD) acoustic models were trained with one for each doctor. Similarly, five mixture trigram language models were trained for individual doctors. Test set perplexities for the five doctors Dr. 1 through Dr. 5 were 115.51, 84.49, 75.63, 116.84, and 107.12, respectively. For details of the captioning system and data, please refer to [1]. The task vocabulary size was 46k except for the case of Table 2, where 21k was used for faster decoding speed.

Table 1 shows the recognition results for SDCHMM, using the discussed three methods to measure similarities of *pdf*'s.

Table 1. Comparison of word accuracy between SDCHMM and CDHMM

	Word Accuracy			
	Full-space model	Subspace model		
		CBB	KLD	OAP
Dr. 1	80.28%	80.41%	80.50%	81.14%
Dr. 2	74.08%	73.90%	73.76%	74.05%
Dr. 3	71.82%	73.98%	73.55%	74.41%
Dr. 4	79.32%	79.66%	79.13%	79.54%
Dr. 5	82.12%	81.42%	81.85%	82.17%
Average	77.52%	77.87%	77.76%	78.26%

From Table 1 we see that OAP obtained better results than CBB and KLD, except for Dr. 4's dataset, where CBB got the best result. We also observe that the average performance of each subspace model is better than the original full-space model, which is mainly due to our insufficient training data in training a large set of CDHMMs. Subspace modeling made up for the deficiency of training data by reducing the large set of parameters of CDHMMs to a smaller set of parameters of SDCHMM.

In order to measure the effects of Confidence-based pruning (CBP) and SDCHMM, we summarize the recognition results and speeds on Dr. 1's dataset, before and after using Confidence-based pruning and SDCHMM. Here SDCHMM used OAP, and the baseline used CDHMMs. The thresholds for confidence-based pruning were empirically set to be -80 for $P_v(w)/T$ and -9 for $LWPP(w)/T$, with T the duration of word w.

Table 2. Performance of Confidence-based pruning and SDCHMM

	Word accuracy	Speed (RT factor)
Baseline	79.45%	1.2
CBP(P-value + LWPP)	79.35%	1.4

SDCHMM		80.29%	1.0
CBP+ SDCHMM	P-value only	80.25%	0.9
	LWPP only	80.27%	0.9
	Both	80.25%	0.8

From Table 2 we observe that by using Confidence-based pruning and SDCHMM, the speed of decoding engine is improved from 1.2 real-time to 0.8 real-time. But when only confidence-based pruning was used, the speed was even slower than baseline, since computing P-value took up more time than that saved by pruning of uncompetitive paths. Furthermore, in confidence-based pruning, when only P-value or LWPP was used, the speeds were both about 0.9 real-time. Therefore it is meaningful to use both as the confidence scores for path pruning.

Table 3 compares recognition results obtained by using the proposed pre-backtrace method and the baseline, where the latter used OAP based SDCHMM. The pre-backtrace duration thresholds based on unfilled pause, pitch, and filled pauses were empirically set to be 200ms, 100ms, and 50 ms, respectively.

Table 3. Recognition performance of baseline and pre-backtrace

	Word Accuracy	
	Baseline	Pre-backtrace
Dr. 1	81.14%	78.61%
Dr. 2	74.05%	73.32%
Dr. 3	74.41%	72.29%
Dr. 4	79.54%	79.32%
Dr. 5	82.17%	79.76%
Average	78.26%	76.66%

We observe that pre-backtrace decreased word accuracy somewhat, since it is possible to prune away the best path prematurely. Fig. 3 shows the distributions of latencies without and with pre-backtrace. From Fig. 3 we see that pre-backtrace decreased latency greatly. Without pre-backtrace, the maximal latencies ranged from 8.4 seconds to 38.6 seconds, depending on the speaking style of different doctors, and the average latencies were from 2.4 seconds to 5.7 seconds. After pre-backtrace, the average latencies were from 2.1 seconds to 4.5 seconds, and the maximal latency dropped to 12.9 seconds.

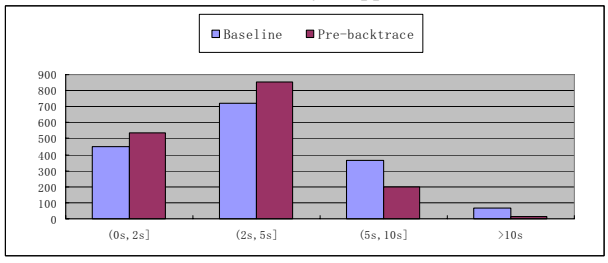

Fig. 3 Distributions of latencies

In order to verify the effect of using F_0 contour in pre-backtrace, we provide the results of using only unfilled pauses or filled pauses to detect boundaries for pre-backtrace. This test was done on Dr. 1's dataset. Table 4 summarizes the results.

Table 4. Comparison of using unfilled pause , filled pause and F_0 contour in detection of prosodic boundary

	Word Accuracy	Latency	
		Average	Max.
200 ms pause	79.24%	3.5s	9.6s
150 ms pause	76.23%	2.7s	7.5s
Filled pause	80.55%	3.8s	9.2s
200 ms pause + filled pause + F_0 contour	78.61%	3.1s	7.7s
Baseline	81.14%	4.0s	11.0s

We observe that unfilled pause feature is effective on latency but introduces a large loss on word accuracy, filled pause feature introduces less errors but has less effect on latency, and the best tradeoff in accuracy and latency appears to be achieved with using all three prosodic features.

6. CONCLUSION

In this paper we propose three new methods to improve the speed and latency performance of a speech decoding engine for telehealth captioning. Complementary word confidence scores were used to prune uncompetitive paths. SDCHMM was used for fast generations of acoustic and local confidence scores with an improved clustering similarity measure. Prosodic features of unfilled pause, filled pause, and pitch contour were used to detect syntactic boundaries to decrease latency in recognition outputs of an online LVCSR system. By combining these three methods, our decoding engine achieved faster decoding speed and lower latency.

REFERENCE

[1] Y. Zhao et al, "An Automatic captioning system for telemedicine," in Proc. of *ICASSP* 2006, May 2006.

[2] X. L. Aubert, "An overview of decoding techniques for large vocabulary continuous speech recognition," *Computer Speech and Language*, Vol. 16, pp. 89-114, 2002.

[3] S. Ortmanns and H. Ney, "Look-ahead techniques for fast beam search," *Computer Speech and Language*, Vol. 14, pp.15-32, 2000.

[4] T. Kawahara, H. Nanjo, S. Furui, "Automatic transcription of spontaneous lecture speech," *ASRU. IEEE Workshop*, pp. 186-189, 2001.

[5] E. Bocchieri and B. K.-W. Mak, "Subspace distribution clustering Hidden Markov Model," *IEEE Transactions on Speech and Audio Processing*, Vol. 9, No. 3, pp. 264-275, March 2001.

[6] J. Xue and Y. Zhao, "Random forest-based confidence annotation using novel features from confusion network," in Proc. of *ICASSP* 2006, May 2006.

[7] S. Abdou and M.S. Scordilis, "Beam search pruning in speech recognition using a posterior probability-based confidence measure," *Speech Communication*, Vol. 42, pp. 409-428, 2004.

[8] Y. Fu and L. Du, "Combination of multiple predictors to improve confidence measure based on local posterior probabilities," *Proc. ICASSP*, pp. 93-96, 2005.

[9] X-B. Li, F. K. Soong, T. A. Myrvoll and R-H. Wang, "Optimal clustering and non-uniform allocation of Gaussian kernels in scalar dimension for HMM compression," *Proc. ICASSP*, pp. I669-I772, 2005.

[10] K. Hirose, A. Sakurai and H. Honno, "Use of prosodic features in the recognition of continuous speech," *Proc. ICSLP*, pp. 1123-1126, 1994.

[11] D. Povey and P.C. Woodland, "Frame discrimination training of HMMs for large vocabulary speech recognition," *Proc. ICASSP*, pp. 333-336, 1999.

[12] X. Li and Y. Zhao, "A fast and memory-efficient N-gram language model lookup method for large vocabulary continuous speech recognition," to appear in *Computer Speech & Language*, 2006.