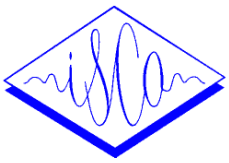

4th European Conference on
Speech Communication and Technology
EUROSPEECH '95
Madrid, Spain, September 18-21, 1995

HAMMING DISTANCE APPROXIMATION FOR A FAST LOG-LIKELIHOOD COMPUTATION FOR MIXTURE DENSITIES

Peter Beyerlein and Meinhard Ullrich
Philips GmbH Forschungslaboratorien Aachen, P.O.Box 1980, D-52021 Aachen, Germany

Abstract

A computationally very expensive task arising within speech recognition systems using continuous mixture density HMMs is the log-likelihood computation. In the Philips large-vocabulary continuous-speech recognition system it consumes 50% - 75% of the decoding time. In our system the log-likelihood computation amounts to a nearest-neighbor search, i.e. to a search for the component density of a mixture density whose mean vector has a minimal distance to the observed feature vector.
In this paper, we show that a Hamming Distance Approximation (HDA) of the angles between the vectors leads to a powerful nearest-neighbor search technique with negligible memory demands.
Thus the likelihood-computation was sped up by a factor of 10 without significant increase in the word error rate of our large vocabulary speech recognizer. Since the likelihood-computation in this system consumed 66% of the recognition runtime, the overall decoding runtime could be reduced by a factor of 2.5. We also report results on TI-digits and the WSJ task.

10.21437/Eurospeech.1995-210

1 Introduction

The acoustic modeling in the Philips large-vocabulary continuous-speech recognition system [2],[3] is based on HMMs with continuous mixture densities. The likelihood of the $(D-1)$-dimensional observation vector $\vec{x}$ is given by

$$p(\vec{x}) = \sum_{k=1}^{K} w(k) \cdot p(\vec{x}|k),$$

where $w(k)$ is the weight of the k-th Laplacian (or Gaussian) mixture density component. After scaling we arrive at component densities with a pooled, scalar variance. Replacing the sum of probabilities by the maximum and taking the logarithm we obtain the nearest-neighbor rule for the log-likelihood $\log(p(\vec{x}))$:

$$\log(p(\vec{x})) \approx A - \min_{k=1,\ldots,K} \{d(\vec{y}, \vec{a}(k)) - \log w(k)\},$$

where A is a certain constant and $d(\vec{y}, \vec{a}(k))$ is the city-block distance (L_1-norm) between scaled observation vector $\vec{y}$ and scaled location vector $\vec{a}(k)$. We now define

$$\begin{aligned} \vec{o}^T &= (\vec{y}^T, 0) \\ \vec{r}(k)^T &= (\vec{a}(k)^T, -\log w(k)), \end{aligned}$$

and we denote $\vec{r}(k)$ as prototype k. We further denote $\mathcal{P} = \{\vec{r}(1), \ldots, \vec{r}(K)\} \subset \mathbb{R}^D$ as the set of the K prototypes $\vec{r}(k)$. To compute the log-likelihood $log(p(\vec{x}))$ we have to find the value $l(\vec{o})$:

$$\begin{aligned} l(\vec{o}) &= \min\{d(\vec{o}, \vec{r}) : \vec{r} \in \mathcal{P}\} \\ d(\vec{o}, \vec{r}) &= \sum_{c=1}^{D} |o_c - r_c|. \end{aligned}$$

The task is now to find the prototype $\vec{r}^{opt}$ nearest to the observed feature vector $\vec{o} \,\epsilon\, \mathcal{O}$ ($\mathcal{O} = \mathbb{R}^D$, the set of possible observation vectors) in an efficient way:[1]

$$\vec{r}^{\,opt} := \operatorname*{argmin}_{\vec{r}\epsilon\mathcal{P}} d(\vec{r}, \vec{o}). \qquad (1)$$

2 Using Approximations

One approach to reduce the computational complexity of a speech recognizer is to reduce the set $\mathcal{P}$ to a subset of $\mathcal{P}$ which contains the true nearest neighbor $\vec{r}^{\,opt}$ (and of course to search for $\vec{r}^{\,opt}$ in that subset). This approach is characterized by the incorporation of additional knowledge about the positions of the prototype vectors and the observation vector, as e.g. provided by the triangle inequality [4],[1]. In [1] we theoretically and experimentally verified the well-known "curse of dimensionality" [5] for the nearest-neighbor search task: *The performance of a nearest-neighbor search decreases with increasing dimension and decreasing number of prototypes.*
Due to this fact we are forced to apply approximation techniques, as proposed in [6]. To introduce as low a distortion as possible we decided to reduce the set $\mathcal{P}$ down to a much smaller subset, based on a nearest-neighbor search with a computationally "cheaper" distance measure. This approach is described in the following sections.

[1] A similar simplification of the likelihood computation holds for Gaussian densities. The derived algorithm is the same for Laplacians (d is the L_1-norm) and Gaussians (d is the L_2-norm), hence we formulated its derivation for any L_r - norm.

3 General Approach

Our aim is to define a subset function s

$$\begin{aligned} s &: \mathcal{O} \rightarrow \wp(\mathcal{P}) \\ \wp(\mathcal{P}) &= \text{power set (set of all subsets) of set P ,} \end{aligned}$$

which maps the observation vector $\vec{o}$ on a subset P_s of "near" prototype vectors $\vec{r}$. We then arrive at the reduced task:

$$\begin{aligned} \vec{r}^{\,*} &= \underset{\vec{r}\epsilon\mathcal{P}_s}{\text{argmin}}\, d(\vec{r},\vec{o}) \qquad (2) \\ \mathcal{P}_s &\subset \mathcal{P} \\ \mathcal{P}_s &= s(\vec{o}). \end{aligned}$$

P_s does not necessarily contain the true nearest neighbor $\vec{r}^{\,opt}$, thus our goal is to design the subset function s such that $\vec{r}^{\,opt}$ will very likely be included in $\mathcal{P}_s$.

We construct the map s as follows: Let $\mathbb{B}$ be a set of discrete values and

$$q \;:\; \mathbb{R}^D \rightarrow \mathbb{B}^D \times \mathbb{R}^F$$

a "quantization" function. The subscript q denoting "quantization", this can be written as $q : \mathcal{O} \rightarrow \mathcal{O}_q$. So $\vec{o}_q := q(\vec{o})$ is a vector with D discrete components and F additional continuous components, containing information about the original vector $\vec{o}$. q induces a map $\mathcal{P} \rightarrow \mathcal{P}_q$, where $\mathcal{P}_q = q(\mathcal{P})$, as well as a map $q^* : \wp(\mathcal{P}_q) \rightarrow \wp(\mathcal{P})$, such that

$$q^*(M) = \{\vec{r} \in \mathcal{P} : q(\vec{r}) \in M\} \qquad (3)$$

Now we define a subset function s_q in the $\mathcal{O}_q$-space.

$$\begin{aligned} s_q &: \mathcal{O}_q \rightarrow \wp(\mathcal{P}_q) \\ s_q(\vec{o}_q) &= \{\vec{r}_q \in \mathcal{P}_q : h(\vec{o}_q,\vec{r}_q) < t(\vec{o},\mathcal{P})\}. \end{aligned}$$

Here the function h is a distance function in the space $\mathcal{O}_q$, with low computational complexity, t is any real valued thresholding function. s is now defined via the following diagram:

$$\begin{array}{ccccc} \text{s} : & \mathcal{O} & \xrightarrow{\;s\;} & \wp(\mathcal{P}) & \\ & \downarrow q & & \uparrow q^* & \\ & \mathcal{O}_q & \xrightarrow[s_q]{} & \wp(\mathcal{P}_q) & , \end{array}$$

or more explicitly

$$s(\vec{o}) = \{\vec{r} \in \mathcal{P} : h(q(\vec{o}),q(\vec{r})) < t(\vec{o},\mathcal{P})\}.$$

Once the subset $\mathcal{P}_s$ is given, we apply an optimal nearest-neighbor search within $\mathcal{P}_s$ to find $\vec{r}^{\,*}$.

4 The HDA Algorithm

For the proposed Hamming Distance Approximation (HDA) algorithm the functions q, h, t are defined in the following way[2] :

$$\begin{aligned} F &= 1 \\ \mathbb{B} &= \{-1,1\} \\ q &: \mathbb{R}^D \rightarrow \{-1,1\}^D \times \mathbb{R} \\ q(\vec{z}) &= [sign(z_1),...,sign(z_D),\|\vec{z}\|]^T \\ \vec{x} &= q(\vec{o}) \\ \vec{y} &= q(\vec{r}) \\ h(\vec{x},\vec{y}) &= b_{\vec{x}\vec{y}}\left|x_{D+1}-y_{D+1}\right|^r \\ &\quad +(1-b_{\vec{x}\vec{y}})\left|x_{D+1}+y_{D+1}\right|^r \\ b_{\vec{x}\vec{y}} &= \sum_{c:x_c=y_c}\frac{1}{D}. \end{aligned}$$

A motivation for these definitions is given below: Let $\vec{o} \in \mathcal{O}$ and $\vec{r} \in \mathcal{P}$. Their L_r distance is:

$$\|\vec{o}-\vec{r}\| = (\sum_{c=1}^{D}|o_c-r_c|^r)^{\frac{1}{r}}. \qquad (4)$$

For the position of a vector $\vec{o}$ in the feature space we can write:

$$\vec{o} = \vec{S}(\vec{o})\frac{\|\vec{o}\|}{D^{\frac{1}{r}}} + \Delta\vec{o}, \qquad (5)$$

where $\vec{S}(\vec{o}) = [sign(o_1),...,sign(o_D)]^T$. Reformulating (4) we arrive at:

$$\begin{aligned} \|\vec{o}-\vec{r}\| &= \overline{d}_{\vec{o}\vec{r}} + \varepsilon \\ \overline{d}_{\vec{o}\vec{r}} &= \left\|\vec{S}(\vec{o})\frac{\|o\|}{D^{\frac{1}{r}}} - \vec{S}(\vec{r})\frac{\|r\|}{D^{\frac{1}{r}}}\right\| \qquad (6) \\ |\varepsilon| &\leq \|\Delta\vec{o}\| + \|\Delta\vec{r}\|. \end{aligned}$$

After some computations we get for $\overline{d}_{\vec{o}\vec{r}}$:

$$\overline{d}^{\,r}_{\vec{o}\vec{r}} = q_{\vec{o}\vec{r}}\left|\|\vec{o}\|-\|\vec{r}\|\right|^r + (1-q_{\vec{o}\vec{r}})\left|\|\vec{o}\|+\|\vec{r}\|\right|^r \qquad (7)$$

where

$$q_{\vec{o}\vec{r}} = \sum_{c:o_c r_c>0}\frac{1}{D} \quad , \quad 0 \leq q_{\vec{o}\vec{r}} \leq 1.$$

Note that the term $D \cdot q_{\vec{o}\vec{r}}$ can be derived simply by computing the Hamming distance of the two code vectors $\vec{S}(\vec{o})$ and $\vec{S}(\vec{r})$. In speech recognition we often use Laplacian or Gaussian probability density functions. To show how the approximation (7) works, we derive the corresponding equations for $r = 1$ and $r = 2$. For Laplacians ($r = 1$) we approximate:

$$\|\vec{o}-\vec{r}\|_1^1 \approx \|\vec{o}\|_1^1 + \|\vec{r}\|_1^1 - 2q_{\vec{o}\vec{r}}min(\|\vec{o}\|_1^1,\|\vec{r}\|_1^1).$$

[2] $sign(x) = -1$ if $x < 0$, $sign(x) = 1$ if $x \geq 0$

For Gaussians ($r = 2$) we use:

$$||\vec{o} - \vec{r}||_2^2 \approx ||\vec{o}||_2^2 + ||\vec{r}||_2^2 - 2(2q_{\vec{o}\vec{r}} - 1)||\vec{o}||_2||\vec{r}||_2.$$

Thus using only the norms $||\vec{o}||, ||\vec{r}||$ of the two vectors together with the value $q_{\vec{o}\vec{r}}$ we get an estimate for the true distance $||\vec{o} - \vec{r}||$. Note that the above equations contain nothing but the special distance measures in the $\mathcal{O}_q$ space, defined in the previous section.

The estimation (7) can be used in several ways to derive fast and inexpensive nearest-neighbor search algorithms.

5 Combining the HDA Idea with the RJE Algorithm

In [1] we published the RJE algorithm — a fast triangle-inequality based nearest-neighbor search algorithm. It was derived from the AESA [4] and adapted to our setup, with 32 to 120 component densities per mixture density and a feature space dimension of D= 30,...,60. Unfortunately, this algorithm requires a memory of $\frac{N(N-1)}{2}$ distances. Thus we are concerned with the problem of reducing these memory requirements, while keeping up the performance of the nearest neighbor search.

Let $\vec{x}, \vec{y}, \vec{z}$ be D-dimensional vectors in a metrical feature space with distance function $d(\vec{x}, \vec{y})$. The triangle inequality can be written as $|d(\vec{x}, \vec{y}) - d(\vec{y}, \vec{z})| \leq d(\vec{x}, \vec{z})$. Let $d(\vec{o}, \vec{s})$ be the distance between the observation $\vec{o}$ and prototype $\vec{s}$, $d(\vec{r}, \vec{s})$ the distance between prototypes $\vec{s}$ and $\vec{r}$, and d_{min} the current nearest-neighbor distance.

Suppose all $d(\vec{r}, \vec{s})$ are known.

If $|d(\vec{o}, \vec{s}) - d(\vec{r}, \vec{s})| > d_{min}$ holds, than $d(\vec{o}, \vec{r}) > d_{min}$, and prototype $\vec{r}$ cannot be the nearest neighbor, i.e. it can be eliminated without distance computation.

To further increase the number of eliminated prototypes the triangle inequality can be tightened by a so-called 'looseness' parameter L, [7],[1]:

If $|d(\vec{o}, \vec{s}) - d(\vec{r}, \vec{s})| + L > d_{min}$ holds and L is close to zero, $\vec{r}$ is probably not the nearest neighbor, i.e. it will be eliminated without distance computation.

This led to a suboptimal but well-performing version of the algorithm.

To reduce the memory demands of this technique, we discarded the mutual distances of all the prototypes, but used the distances of the prototypes to the origin $\vec{0}$. Suppose all $d(\vec{r}, \vec{0}) = ||\vec{r}||$ are known. For each observation vector compute $d(\vec{o}, \vec{0}) = ||\vec{o}||$.

If $|d(\vec{o}, \vec{0}) - d(\vec{r}, \vec{0})| = \big|\, ||\vec{o}|| - ||\vec{r}|| \,\big| > d_{min}$ holds, $|d(\vec{o}, \vec{r})| > d_{min}$, and prototype $\vec{r}$ cannot be the nearest neighbor, i.e. it can be eliminated without distance computation.

Of course an elimination technique only based on the norm of the compared vectors is too weak and cannot work effectively enough. Hence we again have to tighten this inequality by a looseness value L.

If $\delta(\vec{o}, \vec{r}) = \big|\, ||\vec{o}|| - ||\vec{r}|| \,\big| + L > d_{min}$ holds and L is suitably chosen , $\vec{r}$ is probably not the nearest neighbor, i.e. it will be eliminated without distance computation.

At this point we have to discuss the choice for the value L. We know from geometrical considerations that $0 < L < 2\min(||\vec{o}||, ||\vec{r}||)$ is the only useful choice for L. Thus we can write with $0 < \gamma < 1$

$$\begin{aligned} L &= 2\min(||\vec{o}||, ||\vec{r}||)\gamma \\ &= \gamma(||\vec{o}|| + ||\vec{r}|| - \big|\, ||\vec{o}|| - ||\vec{r}|| \,\big|) \\ \delta(\vec{o}, \vec{r}) &= (1-\gamma)\big|\, ||\vec{o}|| - ||\vec{r}|| \,\big| + \gamma(||\vec{o}|| + ||\vec{r}||). \end{aligned}$$

Now we have to determine the value γ, which actually depends on the unknown angle between the observation vector $\vec{o}$ and the prototype vector $\vec{r}$. Obviously $\gamma = 0$, if this angle vanishes and $\gamma = 1$, if $\vec{r}$ points into the opposite direction of $\vec{o}$. We approximate γ by the value $\gamma \approx 1 - q_{\vec{o}\vec{r}}$, with $q_{\vec{o}\vec{r}}$ defined in (7). Thus we arrive at:

$$\begin{aligned} \delta(\vec{o}, \vec{r}) &= q_{\vec{o}\vec{r}}\big|\, ||\vec{o}|| - ||\vec{r}|| \,\big| + \\ &\quad (1 - q_{\vec{o}\vec{r}})(||\vec{o}|| + ||\vec{r}||). \end{aligned} \qquad (8)$$

The estimated distance $\delta(\vec{o}, \vec{r})$ is composed of the norms of the vectors $\vec{o}$ and $\vec{r}$ and the sign difference between these vectors, which can be computed very efficiently. This estimated distance $\delta(\vec{o}, \vec{r})$ can now be used to eliminate the prototype $\vec{r}$, based on the RJE techniques. Note that for an L_1 norm, i.e. for Laplacian densities, the above approximation (8) is equal to the HDA (7).

6 Experimental Results

In our system the likelihood computation is organized in two different ways. One way is to compute the likelihoods of all HMM states for all observations, regardless wether they are needed during the search or not. This is efficient if most of the HMM states are involved in the search, which mainly depends on the number of different states. The second way is to compute the likelihood of an HMM state for an observation only, if it is needed during the search for the sentence hypothesis. We call this approach "likelihood computation on demand". It is particularly efficient for triphone-based large vocabulary speech recognition systems.

Table 1 shows the performance of the HDA algorithm on the adult speakers' portion of the TI/NIST Connected Digits Recognition Task. To train the system, LDA [8] and clustering techniques [9] were applied. The recognizer is composed of 20k densities (32 densities per HMM state) and 30k density weights. The table shows that the overall search time could be reduced by a factor of 2.5 without any increase in the

string error rate (SER). The HDA was applied to our

TI-digits	K_{eff}	recognition time [%]	SER [%]
FULL+OD	32	100	0.81
HDA+OD	8	38	0.81

Table 1: Results for $K = 32$ with

K number of mixture components
K_{eff} number of computed distances
OD (distance computation on demand)
FULL (exhaustive search),
PD (Partial-Distance Algorithm [1]) and
RJE (Recall Jump Eliminate Algorithm [1])
HDA (Hamming distance approximation)

German speaker-dependent CD-HMM system with a vocabulary of about 18k words. For details on the system, see [1]. Table 2 shows that the likelihood computation runtime could be sped up by a factor of up to 10 without any significant performance degradation. The overall search time could be reduced by a factor of 2.4. We applied the HDA approach to

German 18k words	K_{eff}	likelihood comp. runtime [%]	word errors [%]
FULL	120	100	14.7
PD	96	75	14.7
RJE	-	49	14.7
HDA	6	12	14.7
HDA	5	10	14.9

Table 2: Results for $K = 120$

our WSJ-system. For details on this system, see [10]. On the male portion of the 5k WSJ task, first tests confirmed a speed-up of the overall recognition by a factor of at least 2 (Table 3).

WSJ si_evl5 bigram	recognition time [%]	word errors [%]
FULL+OD	100	5.13
HDA+OD	50	5.20

Table 3: Preliminary results for $K = 32$

7 Summary

A fast Hamming-distance based approximation procedure has been introduced for the log-likelihood computation of CD-HMM's. The log-likelihood computation runtime is accelerated by a factor of 10 without any significant increase in the word error rate of our German large-vocabulary system. This is a notable improvement, in particular when compared with the PDBP and RJE algorithms, studied in [1]. The performance of the approximation procedure was validated on the adult's part of the TI digits database, where the overall recognition runtime could be reduced by a factor of 2.5. Thus the HDA is a powerful approximation technique for fast log-likelihood computation and sentence decoding in CD-HMM-based speech recognizers.

8 References

[1] P. Beyerlein, "Fast Log-Likelihood Computation for Mixture Densities in a High-Dimensional Feature Space", Proc. ICSLP Yokohama, 1994, pp. 271-274.

[2] H. Ney, "Acoustic Modelling of Phoneme Units for Continuous Speech Recognition", Signal Processing V: Theory and Applications, 1990.

[3] H. Ney, V. Steinbiss, R. Häb-Umbach, B.-H. Tran, U. Essen, "An Overview of the Philips Research System for Large-Vocabulary Continuous Speech Recognition", International Journal of Pattern Recognition and Artificial Intelligence, 1994, Vol. 8, No. 1, pp. 33-70.

[4] E. Vidal, "An Algorithm For Finding Nearest Neighbors in (Approximately) Constant Average Time", Pattern Recognition Letters 4, 1986.

[5] R.O. Duda, P.E. Hart, "Pattern Classification and Scene Analysis", John Wiley & Sons, Inc., 1973

[6] E. Bocchieri. "Vector Quantization for the Efficient Computation of Continuous Density Likelihoods", Proc. ICASSP Minneapolis, 1993, pp. 692-695.

[7] P. Aibar, A. Juan, E. Vidal, "Extensions to the Approximating and Eliminating Search Algorithm (AESA) for Finding K-Nearest-Neighbours", NATO-ASI, New Advances in Speech Recognition and Coding, Bubion, 1993.

[8] R. Häb-Umbach, D. Geller, H. Ney, "Improvements in Connected Digit Recognition Using Linear Discriminant Analysis and Mixture Densities", Proc. ICASSP Minneapolis, 1993, pp 239-292.

[9] C. Dugast, P. Beyerlein, R. Häb–Umbach, "Application of Clustering Techniques to Mixture Density Modelling for Continuous-Speech Recognition", Proc. ICASSP'95 Detroit, 1995, Vol. I. pp 524-527.

[10] X. Aubert, C. Dugast, H. Ney, V. Steinbiss, "Large Vocabulary Continuous Speech Recognition of Wall Street Journal Corpus", Proc. ICASSP Adelaide, 1994, Vol. II. pp 129-132.