Distortion Discriminant Analysis for Audio Fingerprinting

Christopher J.C. Burges, John C. Platt and Soumya Jana

***Abstract*— Mapping audio data to feature vectors for the classification, retrieval or identification tasks presents four principal challenges. The dimensionality of the input must be significantly reduced; the resulting features must be robust to likely distortions of the input; the features must be informative for the task at hand; and the feature extraction operation must be computationally efficient. In this paper, we propose Distortion Discriminant Analysis (DDA), which fulfills all four of these requirements. DDA constructs a linear, convolutional neural network out of layers, each of which performs an oriented PCA dimensional reduction. We demonstrate the effectiveness of DDA on two audio fingerprinting tasks: searching for 500 audio clips in 36 hours of audio test data; and playing over 10 days of audio against a database with approximately 240,000 fingerprints. We show that the system is robust to kinds of noise that are not present in the training procedure. In the large test, the system gives a false positive rate of 1.5×10^{-8} per audio clip, per fingerprint, at a false negative rate of 0.2% per clip.**

***Index Terms*— Audio fingerprinting, robust feature extraction, dimensional reduction**

I. Introduction

AUDIO feature extraction is a necessary step for the classification, retrieval, and identification tasks. To be effective, audio feature extraction must meet four challenging requirements. First, the dimensionality of the input audio signal must be significantly reduced: this paper presents a system that reduces the input dimensionality by a factor of approximately 8,000. Second, the resulting features must be robust to likely distortions of the input: for example, if the task is the identification of songs playing on the radio, the system must be robust to the kinds of nonlinear distortions that most stations introduce into the signal before broadcasting. Third, the resulting features must be informative: for audio identification, different audio clips should map to features that are distant, in some suitable metric. Fourth, the feature extraction operation must be computationally efficient: we require that it use a small fraction of the resources available on a typical PC.

Previous research has usually approached the problem of feature design by hand-crafting features that are hoped to be well-suited for a particular task. For example, current audio classification, segmentation and retrieval methods use heuristic features such as the mel cepstra, the zero crossing rate, energy measures, spectral component measures, and derivatives of these quantities [1], [2], [3]. However, a system designed with heuristic features may not be optimal: other features may give better performance, or may be more robust to noise.

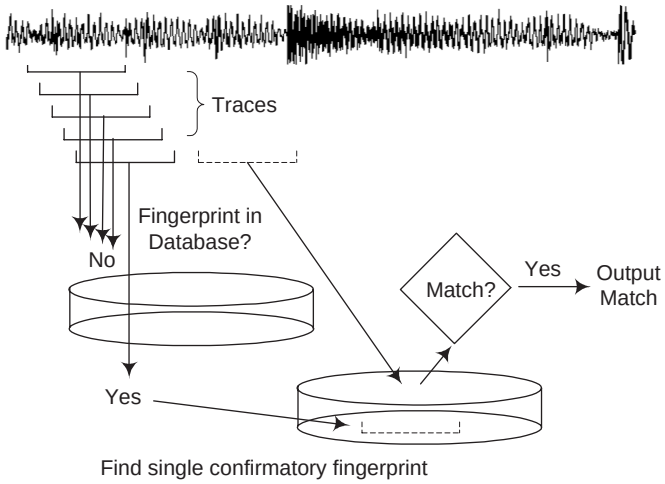

Fig. 1. Overall architecture of the stream audio fingerprinting system.

In this paper, we study stream audio fingerprinting (SAF) as a test bed for studying these issues. In SAF, the task is to identify audio segments in an audio stream, where the stream may have been corrupted by noise. Figure 1 shows the overall setup. A fixed-length segment of the incoming audio stream is first converted into a low-dimensional trace (a vector, shown as an interval in the Figure). This input trace is then compared against a large set of stored, pre-computed traces (fingerprints), where each stored fingerprint has previously been extracted from a particular audio segment (for example, a song). The input traces are computed at repeated intervals in the stream and are compared with the database. An input trace that is found in the database can then be confirmed, at negligible additional computational cost, by using a secondary fingerprint. Typical applications include identifying broadcast audio, for example for royalty assessment, or to confirm that commercials were aired as a service to the sponsor; enabling a software player to identify tracks on user-generated CDs; finding metadata for unlabeled audio; or automatically detecting duplicates in large audio databases.

Audio fingerprinting is also known as audio hashing; for some recent work, see [4], [5], [6]. However the features used there are also hand-designed, and it is interesting and useful to ask whether more robust features could be learned from the data. This paper describes a new algorithm called *Distortion Discriminant Analysis* (DDA) for automatically

C.J.C. Burges and J.C. Platt are with Microsoft Research, 1 Microsoft Way, Redmond WA 98052-6399.

S. Jana is with the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign; worked performed while at Microsoft Research.

extracting noise-robust features from audio. The feature extractors learned with DDA fulfill all four of the requirements listed above. DDA features are computed by a linear, convolutional neural network, where each layer performs a version of oriented Principal Components Analysis (OPCA) dimensional reduction [7]. DDA was originally introduced in [8]; this paper extends the work presented there to address the issue of generalization to kinds of distortion that are not present in the training set, to assess performance on much larger databases, and to provide supporting theoretical arguments for the approach.

In order to build robustness against distortions, DDA assumes that distorted versions of a set of training signals are available. Requiring samples of distorted signals is less stringent and more general than requiring that the real noise model is known. DDA does not assume that the distortion is additive: non-linear distortions are also handled. While it may be useful to be able to train for specific distortions that are expected in test phase, in Section IV DDA is shown to generalize, in that it is robust to distortions that are not used for training.

The pre-computed traces are called 'fingerprints', since they are used to uniquely identify the audio segment. In this paper we perform experiments with one or two fingerprints per audio clip, although the error rates could be further reduced by using more fingerprints.

A. Paper Structure and Notation

Section II gives a brief review of OPCA. In Section III, an experimental DDA system is presented for stream audio fingerprinting. A single DDA layer is tested for robustness to various distortions in Section IV, using 500 clips in 36 hours of audio test data. Results for a two-layer system are given in Section V. In the final tests, 3,444 audio clips (amounting to over 10 days of audio) are tested against fingerprints extracted from 239,369 audio clips. Finally, the relation of DDA to reducing the probability of error for the identification task is studied in the Appendix.

In this paper, vectors are denoted in bold font and their components in normal font, and prime denotes transpose.

II. Oriented PCA

In this Section, we review the method of OPCA[1] [7]. Here, we will use a slightly modified version of this algorithm. Suppose we are given a set of vectors $\mathbf{x}_i \in \mathcal{R}^d$, $i = 1, \ldots, m$, where each $\mathbf{x}_i$ represents a signal (here and below, undistorted data will be referred to as 'signal' data), and suppose that for each $\mathbf{x}_i$ one has a set of N distorted versions $\tilde{\mathbf{x}}_i^k$, $k = 1, \ldots, N$. Define the corresponding difference vectors $\mathbf{z}_i^k \equiv \tilde{\mathbf{x}}_i^k - \mathbf{x}_i$ (referred to as 'noise' vectors below). Roughly speaking, we wish to find linear projections which are as orthogonal as possible to the $\mathbf{z}_i^k$ for all k, but along which the variance of the original signal $\mathbf{x}_i$ is simultaneously maximized. Denote the unit vectors defining the desired projections by $\mathbf{n}_i$, $i = 1, \ldots, M$, where M will be chosen by the user. Let us simplify the discussion by choosing $M = 1$ for the moment.

By analogy with PCA, we could construct a feature extractor $\mathbf{n}$ which minimizes the mean squared reconstruction error $\frac{1}{mN}\sum_{i,k}(\mathbf{x}_i - \hat{\mathbf{x}}_i^k)^2$, where $\hat{\mathbf{x}}_i^k \equiv (\tilde{\mathbf{x}}_i^k \cdot \mathbf{n})\mathbf{n}$. It is straightforward to show that the $\mathbf{n}$ that solves this problem is that eigenvector of $R_1 - R_2$ with largest eigenvalue, where R_1, R_2 are the correlation matrices of the $\mathbf{x}_i$ and $\mathbf{z}_i$ respectively. However this feature extractor has the undesirable property that the direction $\mathbf{n}$ will change if the noise and signal vectors are globally scaled with two different scale factors.

Instead we use OPCA [7]. The OPCA directions are defined as those directions $\mathbf{n}$ that maximize the generalized Rayleigh quotient [9], [7]

$$q_0 = \frac{\mathbf{n}' C_1 \mathbf{n}}{\mathbf{n}' C_2 \mathbf{n}} \tag{1}$$

where C_1 is the covariance matrix of the signal and C_2 that of the noise. However in contrast to the original form of OPCA, we will use the correlation matrix of the noise rather than the covariance matrix, since we wish to penalize the mean noise signal as well as its variance[2]. Explicitly, we take

$$C \equiv \frac{1}{m}\sum_i (\mathbf{x}_i - E[\mathbf{x}])(\mathbf{x}_i - E[\mathbf{x}])' \tag{2}$$

$$R \equiv \frac{1}{mN}\sum_{i,k} \mathbf{z}_i^k (\mathbf{z}_i^k)' \tag{3}$$

and maximize the generalized Rayleigh quotient

$$q = \frac{\mathbf{n}' C \mathbf{n}}{\mathbf{n}' R \mathbf{n}} \tag{4}$$

The numerator in Eq. (4) is the variance of the projection of the signal data along the unit vector $\mathbf{n}$, and the denominator is the projected mean squared "error" (the mean squared modulus of all noise vectors $\mathbf{z}_i^k$ projected along $\mathbf{n}$).

We can find the directions $\mathbf{n}_j$ by setting $\nabla q = 0$, which gives the generalized eigenvalue problem

$$C\mathbf{n} = qR\mathbf{n} \tag{5}$$

It is straightforward to show that:

1) For positive semidefinite C, R (as is the case here), the generalized eigenvalues are positive. However if R is not of full rank, it must be regularized for the problem to be well-posed.
2) Scaling either the signal or the noise leaves the OPCA directions unchanged, although the eigenvalues will change.
3) The $\mathbf{n}_i$ are, or may be chosen to be, linearly independent.
4) Although the $\mathbf{n}_i$ are not necessarily orthogonal, they are conjugate with respect to both matrices C and R.
5) q is maximized by choosing $\mathbf{n}$ to be the highest weight generalized eigenvector.

The meaning of the subsequent eigenvectors is perhaps best understood in a coordinate system in which the noise is white; such a coordinate system is used in the Appendix to give a theoretical argument linking maximization of the generalized Rayleigh quotient to minimization of the expected error.

[1]The technique described here is also sometimes called linear discriminant analysis (LDA); we use the term OPCA since the latter more accurately describes the application.

[2]Consider, for example, noise that has zero variance but nonzero mean. We still wish to find directions that are orthogonal to the mean vector.

III. DDA for Audio Fingerprinting

For high dimensional data such as audio, OPCA can be applied in layers. Consider, for example, the extraction of a 64 dimensional fingerprint from 6 seconds of audio. If we first convert the audio signal to mono and downsample to 11025 Hz, the subsequent feature extraction must map a vector of dimension 66,150 to a vector of dimension 64. Directly solving the generalized eigenvalue problem in this case is infeasible. Instead, OPCA can be applied in two layers, where the first layer operates on a log spectrum computed over a small window and the second layer operates on a vector computed by aggregating vectors produced by the first layer. We call this approach "Distortion Discriminant Analysis" (DDA) [8]. DDA is a linear method; the projections that occur in a given layer may be viewed as a convolution. Thus DDA may be viewed as a linear, convolutional neural network, where the weights are chosen using OPCA.

In DDA, each subsequent layer sees a wider temporal window than the last: the eigen-directions found for that layer are ideally suited to that particular temporal scale. This is an important feature of DDA; for example, we will use it below to compensate for alignment noise, which is defined to be the noise resulting from the fact that a stored fingerprint can be temporally out of phase with the input traces. In the worst case, the fingerprint will have been computed from a frame which lies half way between the two frames used to compute two adjacent input traces. Compensation for such temporal distortions in a DDA system should be applied on the last layers, since they see the widest temporal windows.

DDA not only makes the test phase computationally efficient, and allows the compensation of distortions at different time scales; it is also efficient in the training phase. The required covariance and correlation matrices can be computed one vector at a time. These matrices can thus be estimated using an arbitrarily large amount of data. After the matrices are estimated, the generalized eigenvalues can be computed with standard numerical linear algebra packages.

A. The DDA Stream Audio Fingerprinting System

Techniques for audio processing, for example that of extracting features from speech, often use frame durations of order 20ms. However in order to reduce computational overhead for the fingerprinting application, it is desirable to generate traces from a stream at most a few times per second. For 20ms input frames, the step sizes used in the last DDA layer would have to sample at less than the initial sampling rate of 100Hz, and this can cause aliasing, which will act as a further source of distortion. The system shown in Figure 2 avoids this problem. There is no aliasing because there are no intermediate layers with reduced sampling rate. In fact this requirement, and the requirement that traces be generated at a time scale on the order of one half second, considerably constrains the possible durations of the first layer frame. Also, the temporally wide first layer allows DDA greater flexibility in choosing the important directions in frequency space.

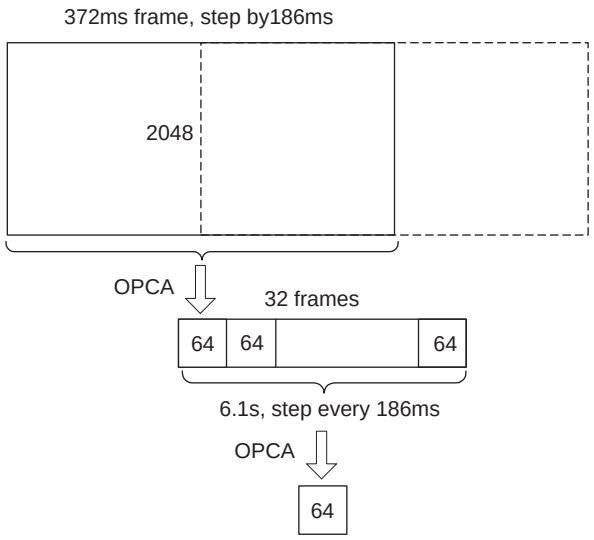

Fig. 2. Architecture of the DDA system. The wide arrows denote OPCA projections. 2048 MCLT log magnitudes are projected to a 64 dimensional space; 32 of the resulting frames are concatenated to form another 2048 dimensional vector, which is then projected using a second layer.

The choice of 64 output dimensions for the first layer is guided by the measured generalized eigenspectra on the training data, shown in Figure 3. Most of the useful information from the first layer is captured in the first 100 projections. The spectrum on the second layer drops off less rapidly. However, to speed up the database lookup, we only consider the top 64 projections on the second layer, also. The speed of the database lookup could be further increased by a factor of two by only sampling the output every 372 ms rather than every 186 ms; this will be investigated below.

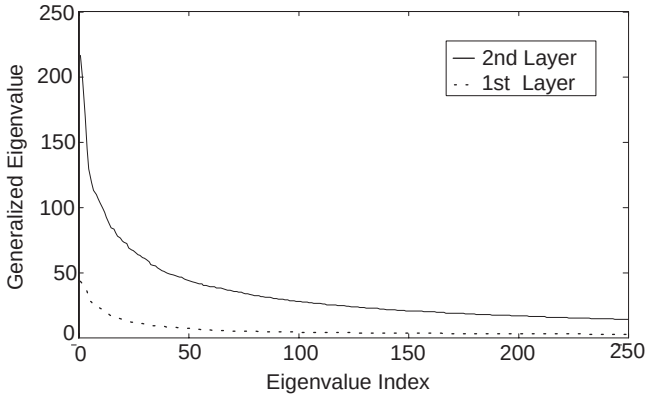

Fig. 3. Generalized eigenvalues of first and second layer projections, ordered by decreasing size. Only the first 250 of the 2048 values are shown.

B. Preprocessing

Our stream audio fingerprinting system first converts a stereo audio signal to mono and then downsamples to 11025 Hz. The signal is split into fixed-length, 372 ms frames which overlap by half. An MCLT (an overlapping windowed Fourier transform) [10] is then applied to each frame. A log

spectrum is generated by taking the log modulus of each MCLT coefficient.

The stream audio fingerprinting system performs two per-frame preprocessing steps that suppress specific, easy-to-identify distortions.

The first preprocessing step removes distortions caused by frequency equalization and volume adjustment. This 'de-equalization thresholding' step applies a low-pass filter to the log spectrum by taking the DCT of the log spectrum, multiplying each DCT coefficient by a weight which ramps linearly from 1 for the first component to 0 for the sixth and higher components, and then performing an inverse DCT. This results in a smooth approximation $\mathcal{A}$ to the log spectrum. $\mathcal{A}$ is then uniformly lowered by 6dB and clipped at -70dB. The output vector of the first preprocessing step is then the component-wise difference between the log spectrum and $\mathcal{A}$ if that difference is positive, else zero.

The second preprocessing step removes distortions in the signal that cannot be heard by a human listener. This step exponentiates the log spectrum from the first step, then generates a frequency-dependent perceptual threshold by an algorithm described in [11] . The final preprocessed signal is then the difference in dB between the log spectrum and the log perceptual threshold, if that difference is positive, and zero otherwise. The final preprocessed data consists of 2048 real coefficients (and thus 2048 bands) per frame.

In the next two Sections, results of four sets of experiments are presented. Section IV gives results on robustness to input signal distortion using only the first layer. Three methods are compared: PCA, OPCA, and features extracted using Bark averaging. Results are given for 9 distortion types that were used for training, and also for 20 distortion types that were not used for training. Section V contains results for the full two-layer DDA system. First, the system is tested for robustness to time misalignment between the input trace and the stored fingerprint. Next, the system is tested for robustness to distortion using a test set composed of 36 hours of audio, again with two sets of distortions, one that was used for training, and one that was not. Finally, results are given for a large scale test using over 10 days of audio, and using a database containing 239,369 audio clips. In all the tests, the stored fingerprints are computed from 6 seconds of audio approximately 30 seconds from the beginning of the clip, but where a random duration of up to 1 second has been deleted from the 30 seconds, in order to emulate the alignment distortion that will be present in any stream application. The confirmation fingerprints used in the final tests are computed from the 6 seconds of audio following the first 6 seconds used to generate the first fingerprint.

IV. Experimental Results: Single Layer

In order to compare the robustness to noise of the three different methods (PCA, OPCA and Bark averaging), in this Section only, results are reported for a single layer, since one of the methods (Bark averaging) only applies to the first computed set of frames.

The training data used for all experiments comprises 200 20s segments, each taken from the middle portion of randomly chosen clips, giving a total of 66.7 minutes of audio. Nine distortions are then applied, using the CoolEdit software tool [12]: a 3/1 compressor above 30dB, a compander, a spline boost between 1.2KHz and 5KHz, a spline notch filter between 430Hz and 3400Hz, a filter emulating poor quality AM radio, two non-linear amplitude distortions, a 1% pitch increase, and a 1% pitch decrease.

A. Robustness to Distortions

In the first of the three compared methods, PCA is performed on the preprocessed data, and the ten largest weight eigenvectors used as projection directions, giving features in a 10 dimensional space. In the second method, similar projections are computed, but using OPCA. For the OPCA generalized eigenvector problem, the 2048 coefficients for each signal frame are subtracted from those of the corresponding distorted frame to generate the 'noise' vectors used to compute the denominator in Eq. (4) (the numerator is computed from the signal vectors, in the same way as for PCA). Finally, projections corresponding to averaging over 10 Bark bands were used. The choice of 10 dimensions was used as a result of previous experiments which showed that the 10 chosen Bark bands (from 510 Hz to 2.7KHz) were the most robust to MP3 recoding and resampling distortions.

The test data consists of 15 clips that are not in the training set, concatenated into a single audio file, giving approximately 1 hour 11 minutes of audio. The following protocol is applied for all three methods. First, the signal data is preprocessed as described above: this generates 21,529 frames altogether. For each distortion, the same is done to generate the same number of 'distorted frames'. In the case of the time compression distortion, a smaller number of frames is generated, and so to compare with the signal frames, a simple form of time warping (on the frame data) is used. Also in some cases (e.g. MP3 recoding) the number of samples changes by approximately 0.002%; this is corrected by removing samples, or adding samples by interpolating between adjacent samples, uniformly across the raw audio data. In all cases, the three methods are compared using the same data.

Since the end goal is to identify audio using a signal-to-distorted-signal distance measure, the quality of the method is measured by computing the average Euclidean distance between signal and distorted frames. For each method, all the Euclidean distances are scaled such that the mean distance between a signal frame and a different signal frame is scaled to unity. The results are shown in Figures 4 and 5. In each plot, the mean and upper and lower quartile distances are shown. Figure 4 shows the results for the training distortions applied to the test data. (Using the training data together with the distortions used for training gave very similar results.) Taking means over the results for the training distortions applied to test data gives overall mean weighted Euclidean distances of 0.229 for PCA, 0.095 for OPCA, and 0.234 for Bark averaging. Figure 5 shows results for test data using twenty distortions that were *not* used during training. Taking means over those twenty distortions gives overall mean weighted Euclidean distances of 0.233 for PCA, 0.171 for OPCA,

and 0.239 for Bark averaging. We conclude that overall, OPCA outperforms both PCA and Bark averaging on training distortions applied to test data, and that the same holds true for distortions that were not used during training, again applied to test data.

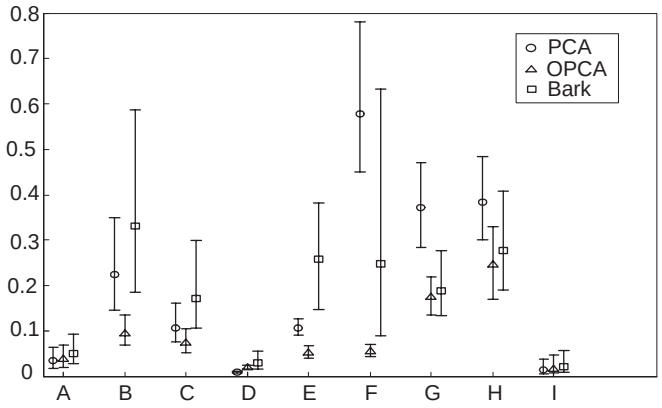

Fig. 4. PCA, OPCA and Bark mean Euclidean distances, and upper and lower quartiles (21,529 points) for all nine training distortions, applied to test data. The distortions are: A, 3:1 Compression above 30dB; B, Nonlinear amplitude distortion; C, Nonlinear bass distortion; D, Midrange frequency boost; E, Notch Filter, 750-1800Hz; F, Notch Filter, 430-3400 Hz; G, Raise Pitch 1%; H, Lower Pitch 1%; I, Companding.

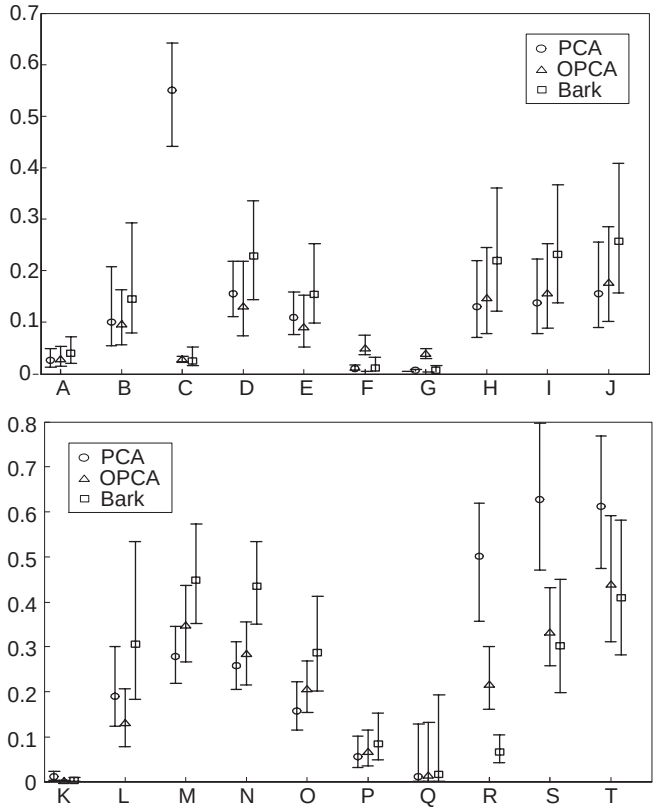

Fig. 5. PCA, OPCA and Bark mean Euclidean distances, and upper and lower quartiles (21,529 points) for twenty test distortions. The distortions are: A, 3:1 compression below 10dB; B, 3:1 expander below 10dB; C, 8KHz resampling; D, 48Kbps MP3 recoding; E, 64Kbps MP3 recoding; F, bass boost; G, bass cut; H, time compress by 2%; I, time compress by 4%; J, time compress by 6%; K, de-esser; L, 'super-loud' amplitude distortion; M, metal room echo chamber; N, small room echo chamber; O, light hall echo; P, 20:1 limiter at 9dB; Q, noise gate at 20dB; R, telephone bandpass, 135 - 3700Hz; S, raise pitch 2%; T, lower pitch 2%.

V. Experimental Results: The Full System

The full two-layer DDA system is constructed as follows. The preprocessing and computation of the signal covariance matrix C, and of the noise correlation matrix R, is performed as described in Section IV. A value v is then added to the diagonal of R, to emulate additional Gaussian noise (and to regularize the computation of the generalized eigenvectors). We chose $v = 0.001\lambda_{\max}$, where $\lambda_{\max}$ is the largest eigenvalue of R. Next, 64 projections are computed, using the highest weight generalized eigenvectors, for the signal and for the distorted training data. Each projection vector is then scaled, and an offset added, so that the training data has zero mean along that projection, and so that the noise vectors have unit variance along that projection. This is done to remove bias and to make each feature equally sensitive to noise. 32 windows of the resulting 64 features are concatenated. The training data is now supplemented with two additional distortions computed by time shifting (see Section V-A below). The training data is then passed through the system, and the second layer C_1^2 and C_2^2 matrices are computed from the results (here the superscript denotes the layer index). Again v is added to the diagonal elements of C_2^2, and the generalized eigenvectors for the second layer are computed. The projections are again normalized so that the signal data has zero mean and the noise has unit variance. These normalized projections are the final outputs of the system for a given input; these are used as both the traces computed by the system and the fingerprints stored in the database. However, in lookup phase, a further, per-fingerprint normalization is applied. Here and below, we define the Euclidean distance between a fingerprint and a clip to be the minimum Euclidean distance between the fingerprint and all traces generated by that clip. For each fingerprint, then, the mean Euclidean distance to a set of 100 clips is computed, and the result used to scale the Euclidean distance for that fingerprint to those clips to unity (the clips are chosen so as to not include the clip from which the fingerprint was computed). In this way the average 'fingerprint to different clip' distance is normalized to one; this enables us to use a single accept threshold for all fingerprints.

Training a DDA system thus amounts to finding the generalized eigenvectors for each layer, together with the offset and scaling factors described above. Once the training is done, the resulting directions are fixed and used for all tests. In test phase, a trace is generated every 186 ms, and compared against the database of fingerprints.

To simplify the exposition, the following definition is used: suppose that a fingerprint *F* has been extracted from a clip *A*. A 'target/target' distance is the distance between *F* and a possibly noisy version of clip *A*; and a 'target/non-target' distance is the distance between *F* and a different clip *B*.

A. Training for Misalignment

A stored fingerprint may not align exactly with an input trace, since a trace is generated only every 186 (or 372) ms.

Misalignment may cause the temporally closest input trace to a stored fingerprint to be rejected, or to be identified incorrectly. However, we can train the DDA system to compensate for misalignment by adding an extra distortion to the training of the last layer, computed as follows: shift the audio input window forward and back by a quarter frame (i.e. half a step size) and treat the resulting signal in the same way as any other distortion. In all the experiments described below, distortion due to misalignment is emulated by computing fingerprints on audio that has been randomly temporally shifted by up to 1 second. (Using a larger shift than the minimum needed for 186ms step sizes enables us to perform experiments at a lower sampling rate; such an experiment is also described below.)

The experiment that measures sensitivity to misalignment uses the same test audio data as in the single layer experiment. For each clip, a matrix of traces is computed; the clip is then randomly shifted and a fingerprint is computed. The smallest squared distance from a given stored fingerprint to all of the input traces from its corresponding target clip is denoted here by d_t, and the smallest squared distance from a given stored fingerprint to all other, non-target clips is denoted by d_n. Figure 6 shows the results for two different DDA systems, where the only difference between the two systems is that one has extra time-shift training on the last layer and the other does not. In both systems, the step size on the output layer was chosen to be 372 ms (we shall see below that this has little adverse effect on performance), and for this experiment, the alignment shift used for training was chosen to be 125 ms. In Figure 6, the y axis is the ratio d_t/d_n, and the clips are ordered by the alignment shift used to generate the fingerprint, shown on the x axis. Figure 6 shows that DDA is effective at reducing noise arising from misalignment of input trace to the stored fingerprint. Note that this kind of "noise" will be present in any system whose input is taken from a stream, and since the noise can be trained on, DDA is ideally suited for dealing with it. Also note that in Figure 6, neither graph is monotonic; the amount of alignment distortion depends on the audio signal as well as on the size of the temporal misalignment.

In all the experiments described below, step sizes of 186 ms were used, and the alignment shift used for training was chosen to be half of this.

B. 36 Hours of Test Data, 500 Fingerprint Database

The next two Sections give results of the full system applied to 500 audio clips, which amounts to approximately 36 hours of audio. The 500 clips are also used to construct the fingerprint database (one fingerprint per clip). As in all tests, each stored fingerprint is randomly shifted by up to one second to simulate alignment noise. Each fingerprint is then compared to, on average, approximately $700,000$ input traces (each clip generates one trace every 186 ms), the vast majority of which should not match[3]. Since we have 500 stored fingerprints, there are roughly 3.5×10^8 opportunities for a false positive to occur in a given experiment.

[3]Correct matches to traces other than the target can occur for audio that is temporally very close to the target, or for example if the music repeats within a given clip.

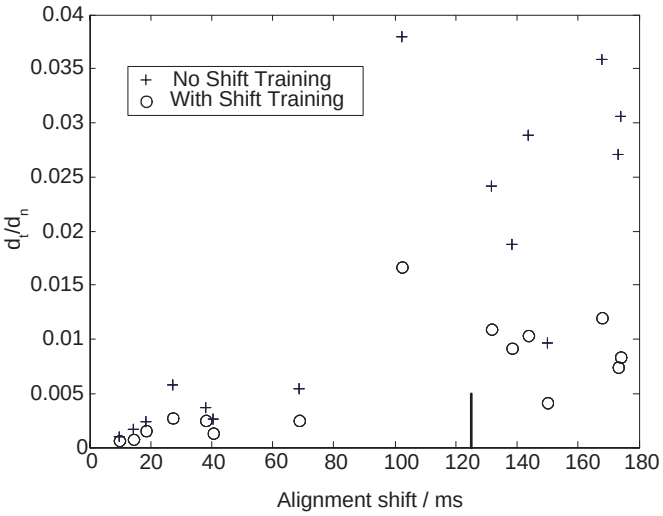

Fig. 6. Stream audio fingerprinting performance for systems trained with and without alignment robustness, ordered in increasing alignment shift. The amount of alignment shift used in training was 125 ms, indicated by the vertical solid line.

1) Alignment Robustness: In order to describe the results, we construct a 500 by 500 weighted Euclidean distance matrix whose rows are indexed by fingerprint and columns by test audio clip. Each element in the matrix is the distance from a fingerprint to a clip. The columns are ordered so that diagonal elements correspond to target/target distances. Thus ideally a threshold θ can be chosen such that all diagonal elements are smaller than θ and all off-diagonal elements are larger than θ. Recall that the mean target/non-target distance (for clips in the validation set) has been separately scaled to unity for each fingerprint. The left curves of Figure 7 show the 500 target / target distances (the diagonal of the distance matrix), sorted in increasing order. The right curve shows the smallest 250 (out of a possible 249,500) target / non-target distances. Note that in this experiment, although the data is test data, the only distortion present is due to misalignment. The largest score for a positive example was 0.22, and the smallest score for a negative example was 0.36, so any threshold chosen between these two numbers would result in zero false positives and zero false negatives.

Figure 7 also shows the effect of halving the output sampling rate of the system, from once every 186ms to once every 372ms. This may be desirable, since it halves the computation required for lookup. This gives little adverse effect on the results, in that almost the same range of thresholds still results in zero false positives and zero false negatives.

2) Robustness to Further Distortions: To test robustness to distortions beyond those resulting from misalignment, two further sets of experiments were performed - again, all distortions are in addition to the alignment distortion. Figure 8 shows histograms of fingerprint / clip distances on the test data. The top, baseline histogram is for alignment noise only, and uses the same data used to construct Figure 7. The second histogram shows the result of adding the distortions used for training. To do this, the 500 clips were split into 10 sets of 50

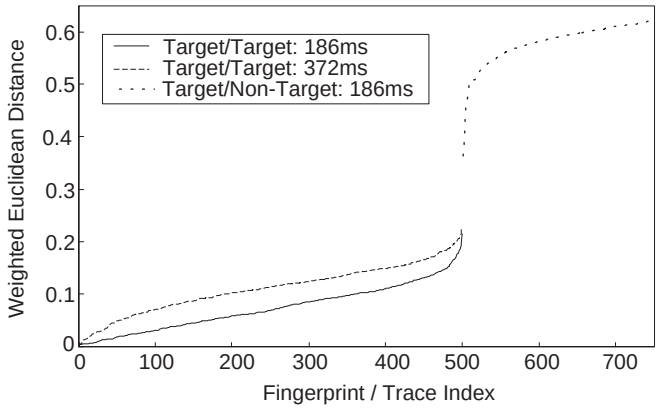

Fig. 7. Minimum normalized squared distances from stored fingerprints to all traces from target clips, and to all traces from non-target clips. The values are ordered in increasing size. Times given are step sizes in milliseconds.

clips, and one of the training distortions (see Section IV) was applied to each group (one group was left with only alignment distortion). The third histogram shows the result of adding the ten test distortions (A) through (J) as described in Figure 5. This set of distortions, although not in the training set, are typical of what might be encountered in the field - resampling, converting to different MP3 bit rates, time compression, etc.[4] The Figure shows that the DDA system is robust to types of noise that are not in the training set, applied to audio clips that were also not in the training set. Figure 9 shows the number of errors, for train and test distortions, as the threshold varies from 0.3 to 0.5; for example, choosing a threshold of 0.4 gives false positive rates of 8×10^{-6} per clip, per fingerprint for both train and test distortions, and false negative rates of 0.2% per clip for training distortions, and 0.8% per clip for test distortions.

Finally, we emphasize that all of the above tests were done using a single fingerprint for lookup. The error rates of the system can be further significantly reduced by using more than one stored fingerprint for a given clip. Once a clip has been tentatively identified, the extra fingerprint can be used to confirm the decision at negligible computational cost. Assuming that the clips generating the false positives are uncorrelated with the true clip, the probability of error can thus be made very low. The next Section uses this idea.

C. 10 Days of Audio Data and 239,369 Fingerprints

This Section gives results for both a larger test set, and for a large database of fingerprints. This is necessary to obtain accurate estimates of the false positive rate, as opposed to using less data and extrapolating using a model. The test set consists of 3,444 songs, amounting to over 10 days, 9 hours of music. The fingerprint database was constructed from 239,369 different pieces of music. The database was constructed independently from the 3,444 song test set, using different sources for all audio. Again, random alignment shifts were added in computing all fingerprints, to emulate alignment noise. In this Section, no further distortions were added to the test data, since the intent is to obtain baseline results for a realistically large system; however a further source of variability results from the fact that the audio in the database can be a remixed version of that used as test data, even though both may have the same label. For these tests, confirmation fingerprints, computed from the 6 seconds immediately after the end of the original fingerprint, are also used. A false positive was counted as an error even if the same clip also generated, elsewhere, a higher scoring match to the correct fingerprint.

The number of false positives was found to be 12, or a

[4]Radio stations often time-compress broadcast music.

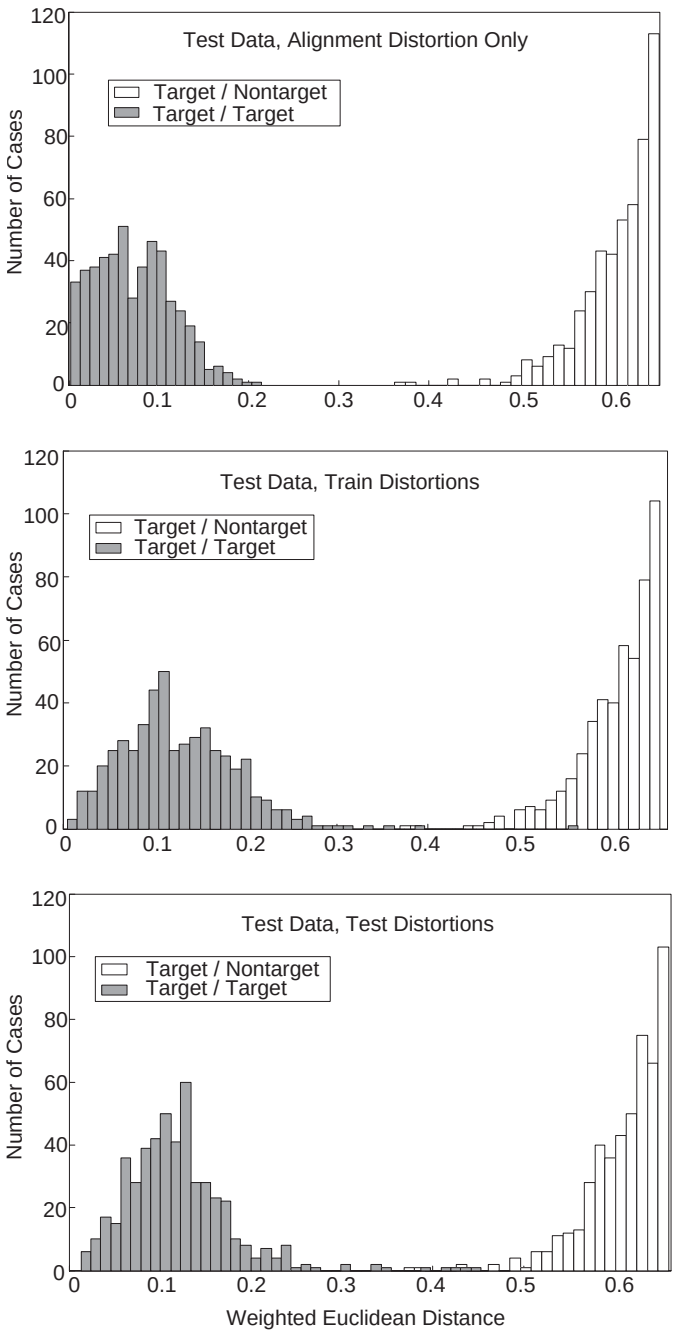

Fig. 8. Weighted Euclidean distances for three sets of distortions. Only the smallest 500 of the 249,500 target/nontarget distances are shown.

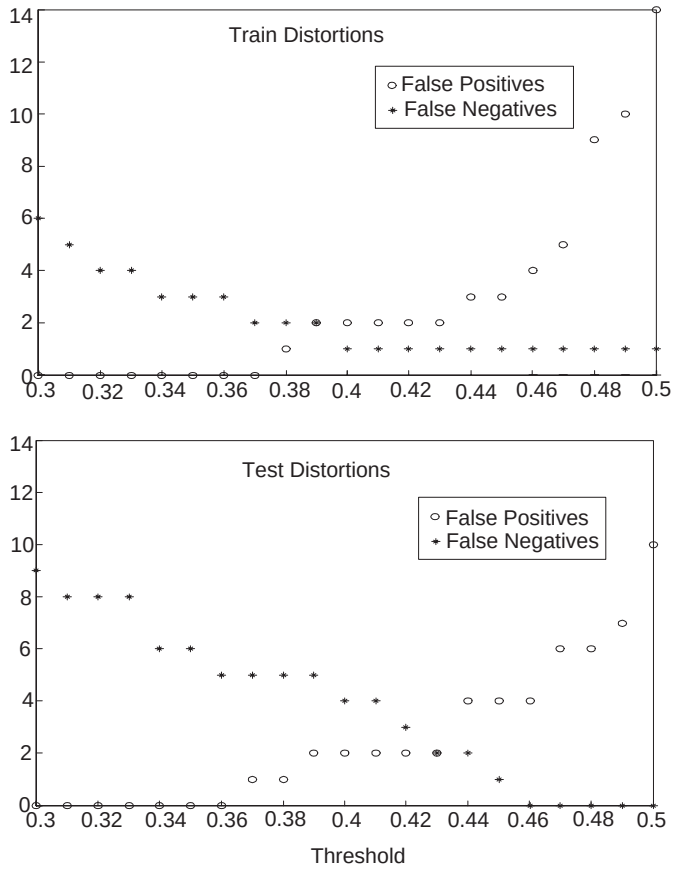

Fig. 9. Numbers of false positives and false negatives, for distortions used during training and distortions not used during training, for the 500 clip tests.

false positive rate of 1.5×10^{-8} per test clip, per database fingerprint. The false positive cases were instructive. Two were cases of the same music, the same artist and the same song, but sung in a different language. In all other instances, the fingerprint had been extracted from a part of the clip with very low musical variance (for example, a single repeated note). The false positive rate could therefore be reduced by ensuring that fingerprints are (automatically) chosen from part of the clip with sufficient musical variance.

False positive scores tended to be low and the incorrect assignments were often obscure. This suggests that combining the score with prior probabilities for the popularity of each song could be used to further reduce the false positive rate. Interestingly, this test uncovered several errors in both databases; thus, given two audio databases, a fingerprinting system can be used to greatly reduce the human effort required to 'clean' the labels for audio clips which occur in both databases. (Similarly, a fingerprinting system could be used to automatically identify duplicates in a single database). The false positive rate could also be further reduced by choosing confirmation fingerprints that are further away from the original fingerprints, since this would on average reduce the correlation between fingerprint and confirmation fingerprint.

The number of false negatives was found to be 7 of the 3,444, giving a false negative rate, per clip, of 0.2%. These cases sound to the human ear like re-mixes of the same song. Again, examining the false negatives uncovered more errors in the database (note that a mislabeled record in the database can generate both a false positive and a false negative).

VI. Conclusions

We have explored a new method, Distortion Discriminant Analysis (DDA), for extracting low-dimensional noise-robust features from audio data. Each DDA layer applies oriented PCA to maximize the SNR of its output. Multiple layers are aggregated in order to enforce shift invariance, to reduce computation time, and to build in robustness at different time scales.

We have shown that DDA generates features with the desired properties for the stream audio fingerprinting task. DDA is a low computational burden for current desktop computers: our current fingerprinting system runs in real time, using approximately 5% CPU on a 1.2 GHz Pentium 3 laptop, when checking an incoming audio stream against a database of size approximately 240,000. This system incorporates a new indexing scheme for fast approximate matching that will be described elsewhere.

We have also shown that DDA can handle types of noise that are not used during training, and that the training at different time scales is ideally suited to compensate for alignment noise. The error rates on a large database were found to be a false positive rate of 1.5×10^{-8} per test clip, per database fingerprint, at a false negative rate of 0.2% per clip.

DDA can be viewed as a linear convolutional neural network, where the weights are trained using OPCA rather than by back-propagation. It will be interesting to extend DDA to non-linear layers, to further reduce the false positive and false negative rates.

Acknowledgments

We thank H.S. Malvar for suggesting the de-equalization algorithm and for supplying the MCLT and perceptual thresholding code. Thanks also to E. Renshaw for helping us parse the large database test results, and to M. Seeger for useful discussions.

VII. Appendix: The Probability of Identification Error

Instead of viewing OPCA as maximizing a signal to noise ratio, we can view it as maximizing the signal variance in that coordinate system in which the noise has unit covariance matrix [7] (here, we simplify the discussion by assuming that the noise is zero mean). Let E be a matrix whose columns are the normalized eigenvectors of R and Λ the corresponding diagonal matrix of eigenvalues, so that $R = E\Lambda E'$. Rotating to a coordinate system in which the noise is white is accomplished by replacing every vector $\mathbf{x}$ by $\Lambda^{-1/2}E'\mathbf{x}$: we have

$$\mathring{R} \equiv \Lambda^{-1/2}E'RE\Lambda^{-1/2} = I, \tag{6}$$

where I is the unit matrix. In this coordinate system, choosing also $\|\mathbf{n}\| = 1$, the generalized Rayleigh quotient becomes

$$\mathring{q} \equiv \mathbf{n}'\mathring{C}\mathbf{n} \tag{7}$$

(where also $\mathring{C} \equiv \Lambda^{-1/2} E' C E \Lambda^{-1/2}$). Thus in this coordinate system, by maximizing $\mathring{q}$ we are just maximizing the signal variance along the unit vector $\mathbf{n}$.

We can characterize the relation between the maximization of $\mathring{q}$ and the expected error rate for the identification task as follows. For the purposes of this discussion we consider a single direction $\mathbf{n}$. Assume that a fixed threshold θ is used in all cases, that is, given some unit direction $\mathbf{n}$, then a distorted test vector $\tilde{\mathbf{s}}$ is identified with the i'th signal point $\mathbf{s}_i$ if both of the following two conditions hold:

$$i = \text{argmin}_j \|\mathbf{n} \cdot \tilde{\mathbf{s}} - \mathbf{n} \cdot \mathbf{s}_j\|^2 \quad (8)$$

$$\theta > \|\mathbf{n} \cdot \tilde{\mathbf{s}} - \mathbf{n} \cdot \mathbf{s}_i\|^2 \quad (9)$$

Note that overall, the noise has fixed, unit variance along $\mathbf{n}$, for any $\mathbf{n}$; however in general the noise associated with any particular signal point $\mathbf{s}$ may not have unit variance, and the density for the noise for a given $\mathbf{s}$ can differ from that for $\mathbf{s}'$, even if $\mathbf{s}$ and $\mathbf{s}'$ have the same projection. To simplify the discussion we therefore assume that, for each signal point, the noise distribution is the same, and is zero mean and unimodal, with mode at zero[5]. It then follows from the above construction that the noise distribution around any given $\mathbf{s}$ must have unit variance.

Now consider the situation where $\mathbf{s}$ is in the database, and where an incoming noisy version $\tilde{\mathbf{s}}$ is to be compared against the database. Suppose that the database is constructed from $\mathbf{s}$ and from k additional points, and that all points are IID with density $p(\mathbf{s})$. Let the k additional points have projections $y_i,\ i = 1, \cdots, k$. First note that, given the above assumptions, for a given threshold θ, *the probability of a false negative does not depend on the direction* $\mathbf{n}$. Let the associated density for the projections along $\mathbf{n}$ be $p_{\mathbf{n}}(y)$, where $y \equiv \mathbf{n} \cdot \mathbf{s}$, and let the cumulative distribution function (cdf) for $p_{\mathbf{n}}(y)$ be $F_{\mathbf{n}}(y)$. Suppose that, for a given $\mathbf{n}$, the target $\mathbf{s}$ with projected value y_t is drawn according to $p_{\mathbf{n}}(y)$. Define the random variable $\rho \equiv \min_i |y_t - y_i|,\ i = 1, \cdots, k$. The cdf for ρ is then

$$F_{\mathbf{n},y_t}(x) \equiv P_{\mathbf{n},y_t}(\rho \leq x) = 1 - (1 - q_{\mathbf{n}}(y_t, x))^k \quad (10)$$

where $q_{\mathbf{n}}(y_t, x) \equiv F_{\mathbf{n}}(y_t + x) - F_{\mathbf{n}}(y_t - x)$ (and where x is positive). Maximizing $\mathring{q}$ amounts to finding that $\mathbf{n}$ that maximizes the empirical variance of $p_{\mathbf{n}}(y)$. As the training sample size becomes sufficiently large, consistency implies that this direction will converge to that direction that maximizes the variance of the density $p_{\mathbf{n}}(y)$. Now $P_{\mathbf{n},y_t}(\rho \leq x)$, averaged over all choices of signal point $\mathbf{s}$, is

$$P_{\mathbf{n}}(\rho \leq x) = \int_{-\infty}^{\infty} (1 - (1 - q_{\mathbf{n}}(y, x))^k) p_{\mathbf{n}}(y) dy \quad (11)$$

Let $j \equiv \text{argmin}_i |y_t - y_i|,\ i = 1, \cdots, k$. We make one final assumption, that $p(\mathbf{s})$ has the property that, for given x, if $\mathbf{n}$ is changed so as to increase $P_{\mathbf{n}}(\rho \leq x)$, then $P_{\mathbf{n}}(\rho' \leq x)$ also does not decrease, where ρ' is the shortest projected distance from $\mathbf{s}$ to that point which lies on the side of $\mathbf{s}$ opposite to point j. Consider now the probability that $\tilde{\mathbf{s}}$ is misidentified - that is, that $\tilde{\mathbf{s}}$ falls closer to a different point in the database, and the corresponding distance is less than θ. Call the expected probability of this type of false positive P_e. Now fix x and view $P_{\mathbf{n}}$ as a function of $\mathbf{n}$. Then given the above assumptions, P_e will be a strictly increasing function of $P_{\mathbf{n}}$. We can gain some insight into how $P_{\mathbf{n}}$ varies with the variance of $p_{\mathbf{n}}(y)$ by taking x sufficiently small so that

$$q_{\mathbf{n}}(y, x) = F_{\mathbf{n}}(y + x) - F_{\mathbf{n}}(y - x) \approx 2x \frac{\partial F(y)}{\partial y} \quad (12)$$

The integral is then approximated by

$$P_{\mathbf{n}} \approx 2kx \int_{-\infty}^{\infty} p_{\mathbf{n}}(y)^2 dy \quad (13)$$

which itself is a monotonically strictly increasing function of

$$S \equiv 2kx \log \int_{-\infty}^{\infty} p_{\mathbf{n}}(y)^2 dy \quad (14)$$

Hence P_e is a monotonically increasing function of S; but S is $-2kx$ times the second order Renyi entropy [13] for the density $p_{\mathbf{n}}(y)$. Thus viewing the Renyi entropy as a function of the variance, for any signal distribution for which increasing the variance increases the second order Renyi entropy[6], increasing the variance will decrease P_e. In fact for the situation discussed in this Section, this will be true for any distribution which can be parameterized by only its mean and variance, and for which either the mean is zero, or for which the mean can be removed from Eq. (13) by a change of variables which leaves the limits unchanged[5]. In that case, since y in (13) can be interpreted as a distance, by a dimensional argument, the integral in (13) must be proportional to the inverse of the standard deviation[7].

Finally, note that even though, for fixed threshold θ, the probability of a false negative does not depend on $\mathbf{n}$, choosing $\mathbf{n}$ that maximizes the variance of $p_{\mathbf{n}}(y)$ allows us to choose a larger θ to get a given false positive rate, so both error rates can effectively be reduced by choosing $\mathbf{n}$ to be that direction with maximum variance.

[5]This still leaves a large class of possible distributions: for example, the uniform, Laplacian, Gaussian, and generalized Gaussian distributions.

[6]For example, the Gaussian distribution.

[7]This is not true for dimensionless y. For example, a zero mean distribution which can be parameterized by only its variance and for which $\int_{-\infty}^{\infty} p_{\mathbf{n}}(y)^2 dy$ *increases* as the variance increases is the following: for any $L > (48)^{-1/4}$, $p(y) = L : -L - \frac{1}{4L} \leq x \leq -L + \frac{1}{4L}$; $p(y) = L : L - \frac{1}{4L} \leq x \leq L + \frac{1}{4L}$; $p(y) = 0$ otherwise.

REFERENCES

[1] T. Zhang and C.-C. J. Kuo, "Heirarchical classiciation of audio data for archiving and retrieving," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 6, 1999, pp. 3001–3004.

[2] L. Lu, H. Jiang, and H. Zhang, "A robust audio classification and segmentation method," Microsoft Research, Tech. Rep., 2001.

[3] J. Foote, "Content-based retrieval of music and audio," in *Multimedia Storage and Archiving Systems II, Proceedings of SPIE*, 1997, pp. 138–147.

[4] J. Haitsma, T. Kalker, and J. Oostveen, "Robust audio hashing for content identification," in *Second International Workshop on Content Based Multimedia and Indexing*. Brescia, Italy: CBMI, September 19-21 2001.

[5] S. Sukittanon and L. Atlas, "Modulation frequency features for audio fingerprinting," in *ICASSP*, vol. 2, 2002, pp. 1773–1776.

[6] J. Herre, E. Allamanche, and O. Hellmuth, "Robust matching of audio signals using spectral flatness features," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2001, pp. 127–130.
[7] K. Diamantaras and S. Kung, *Principal Component Neural Networks*. John Wiley, 1996.
[8] C. Burges, J. Platt, and S. Jana, "Extracting noise robust features from audio data," in *Proceedings of ICASSP 2002*, 2002, pp. 1021–1024.
[9] R. Duda and P. Hart, *Pattern Classification and Scene Analysis*. John Wiley, 1973.
[10] H. S. Malvar, "A modulated complex lapped transform and its applications to audio processing," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, Phoenix, 1999.
[11] H. Malvar, "Auditory masking in audio compression," in *Audio Anecdotes*, K. Greenebaum, Ed. A. K. Peters Ltd., 2001.
[12] http://www.syntrillium.com/cooledit.
[13] T. Cover and J. Thomas, *Elements of Information Theory*. Wiley Interscience, 1991.

PLACE PHOTO HERE

Christopher J.C. Burges received a B.A. (First Class) in Physics from Oxford University in 1977. He entered the PhD program at Brandeis University in 1979, graduating in 1984. After a postdoctoral position at MIT in theoretical particle physics, he joined AT&T Bell Labs in 1986, working on transmission performance and routing algorithms for the CCS7 signaling network. He then worked on applying neural networks to handwriting recognition, for zip codes and bank checks. While at AT&T Bell Labs, and later at Lucent Technologies, he worked on support vector machines, concentrating on training methods, pattern recognition, computational efficiency, theory, and applications such as OCR and speaker identification. He joined Microsoft Research in May, 2000. His current research interests include audio fingerprinting, speaker clustering, and image compression; new optimization algorithms; and kernel methods for machine learning.

PLACE PHOTO HERE

John C. Platt is a Senior Researcher in the Communication, Collaboration, and Signal Processing group at Microsoft Research. His current research interests include machine learning, kernel machines, text categorization, ClearType, multimedia indexing and user interfaces to digital media. Previously, John was Director of Research at Synaptics, Inc. He received his Ph.D. in Computer Science from Caltech in 1989.

PLACE PHOTO HERE

Soumya Jana received the B. Tech. (honors) degree in Electronics and Electrical Communication Engineering from the Indian Institute of Technology, Kharagpur, India, in 1995, and the M. E. degree in Electrical Communication Engineering from the Indian Institute of Science, Bangalore, India, in 1997. He worked for the Silicon Automation Systems (India) Limited during 1997-1998, and joined the University of Illinois at Urbana in 1999, where he is currently pursuing the Ph. D. degree. His research interests are in the area of signal (audio/image/video) compression and detection.