REAL ADABOOST FOR CONTENT IDENTIFICATION

Rohit Naini and Pierre Moulin

University of Illinois
Beckman Inst., Coord. Sci. Lab., & ECE Dept.
405 N. Mathews Ave., Urbana, IL 61801, USA

ABSTRACT

This paper proposes a machine learning method based on Real Adaboost that jointly optimizes the content ID codes and the decoding metric. Significant performance gains over prior art are demonstrated for audio fingerprinting.

1. INTRODUCTION

Hash-based content identification (ID) is an emerging research area. Applications include broadcast monitoring, connected audio, content tracking, asset management, contextual advertisement, and last but not least, filtering for user-generated content websites [1, 2]. Content ID technologies are currently deployed on sites such as YouTube and Dailymotion and aims at identifying (automatically and in real time) copyrighted uploaded content (audio and video). Hash-based algorithms allow for real-time operation. Instead of matching the content itself, one matches short fingerprints extracted from it, using robust hashing methods.

An impressive variety of algorithms have already been developed for constructing signal processing primitives for robust hashes as well as efficient string matching algorithms. Recently there have been attempts to formulate a scientific framework for content ID, aiming at discovering the fundamental limits of content ID and ways to achieve them [3, 4]. On the algorithmic side, a promising hash design algorithm called *Symmetric Pairwise Boosting* (SPB) has been developed and applied to audio and video ID [6, 7]. Our recent paper [5] addressed the problem of optimizing the decoding metric for a given content ID code; significant improvements relative to [6, 7] were reported. This paper proposes a much improved boosting method that exploits confidence scores produced by weak learners, and explores joint design of the encoder and decoding metric.

2. STATEMENT OF THE CONTENT ID PROBLEM

A *content database* is a collection of M elements (content items) $\mathbf{x}(m) \in \mathcal{X}^N, \quad m = 1, 2, \cdots, M$, each of which is a sequence of N *frames* $\{x_1(m), x_2(m), \cdots, x_N(m)\}$. Here $\mathcal{X}$ is the set of possible values of a frame. A frame could be a short video segment, a short sequence of image blocks, or a short audio segment. Frames may be overlapping spatially, temporally, or both. For instance, the audio fingerprinting paper [6] uses overlapping time windows that are 2 sec long and start every 185 ms; the temporal overlap is 15/16. A 3-minute second song is represented by $N = 1000$ frames. It is desired that the audio be identifiable from a short segment, say 5 sec long, corresponding to $L = 16$ frames. This is called the granularity of the audio ID system [6]. Typically $L \ll N$.

The problem is to determine whether a given *probe* consisting of L frames, $\mathbf{y} \in \mathcal{X}^L$, is related to some element of the database, and if so, identify which one. To this end, an algorithm $\psi(\cdot)$ must be designed, returning the decision

$$\psi(\mathbf{y}) \in \{0, 1, 2, \cdots, M\}$$

where $\psi(\mathbf{y}) = 0$ indicates that $\mathbf{y}$ is unrelated to any of the database elements.

Algorithm performance is evaluated using several metrics [1], including execution time, probability of false positives, probability of false negatives, robustness, granularity (L), database size (linear in M), and storage requirements (linear in MN).

3. STRUCTURED CONTENT ID CODES

In this paper, we restrict our attention to the following fairly general class of content ID codes. The codes of [1, 6, 7], among others, fall in this category.

Definition 3.1 *A* (M, N, L) **content ID product code** *for a size-M database populated with $\mathcal{X}^N$-valued content items, and granularity L, is a pair consisting of a mapping $\phi : \mathcal{X} \to \mathcal{F}$ and a decoding function $\psi : \mathcal{F}^L \to \{0, 1, \cdots, M\}$, such that (i) a content item $\mathbf{x}$ is encoded into a fingerprint $\mathbf{f}$ with components $f_i = \phi(x_i)$, $1 \le i \le N$; (ii) a query $\mathbf{y}$ is encoded into a query fingerprint $\mathbf{g}$ with components $g_i = \phi(y_i)$, $1 \le i \le L$; (iii) the decoder returns $\hat{m} = \psi(\phi(y_1), \cdots, \phi(y_L))$.*

Hence the mapping ϕ is applied independently to each frame. It might be convenient to impose additional structure on the code. For instance, the mapping $\phi : \mathcal{X} \to \mathcal{F}$

in [6, 7] is obtained by applying a set of J optimized filters to each frame and quantizing each of the J real-valued filter outputs to four levels. Hence $\mathcal{F}$ takes the form $\mathcal{F} = \mathcal{A}^J$ with $\mathcal{A} = \{a, b, c, d\}$. In this case we view the fingerprint as an array $\mathbf{F} = \{F_{ij}, 1 \le i \le N, 1 \le j \le J\}$ and the probe fingerprint as an array $\mathbf{G} = \{G_{ij}, 1 \le i \le L, 1 \le j \le J\}$. We also write ϕ in vector form as $\phi = \{\phi_j, 1 \le j \le J\}$.

Frame overlap causes strong dependencies between successive fingerprint components. The following section summarizes material from [5].

4. FINGERPRINT MODELS

The simplest model for the original fingerprint is a memoryless model $p_{\mathbf{F}}(\mathbf{f}) = \prod_{j=1}^J \prod_{i=1}^N p_F(f_{ij})$ with marginal distribution p_F.

In the event the probe is related to some element of the database, we initially assume this relationship takes the following form. Let N_0 be an integer in $\{0, 1, 2, \cdots, N-L-1\}$ representing a time offset. We assume the degradation channel from $\mathbf{X}$ to $\mathbf{Y}$ is a stationary stochastic mapping. Hence so is the channel from $\mathbf{F}$ to $\mathbf{G}$.

The degradation channel is of the form $p_0(\mathbf{g}|\mathbf{f}, N_0) = \prod_{j=1}^J \prod_{i=1}^L W(g_{ij}|f_{i+N_0,j})$ where W is the conditional marginal of G_{ij} given F_{ij}. This model implies that the errors on the fingerprint symbols are conditionally iid given $\mathbf{F}$. We refer to this as the order-0 (or memoryless) degradation model.

4.1. Markov Model for Fingerprints

Assume that the process $\mathbf{F}$ is Markov, i.e.,

$$p(\mathbf{f}) = p(f_1) \prod_{i=1}^N p(f_{i+1}|f_i),$$

and the joint process $(\mathbf{F}, \mathbf{G})$ is homogeneous Markov, with

$$P_{\mathbf{G}|\mathbf{F}}(\mathbf{g}|\mathbf{f}) = W(g_1|f_1) \prod_{i=1}^{L-1} V(g_{i+1}|g_i, f_{i+1}, f_i)$$

where V is a conditional pmf on $\mathcal{F}$. This implies that the process $\mathbf{G}$ is a Hidden Markov Model, with hidden state $(F, G) \in \mathcal{F}^2$, transition kernel V, and deterministic observational model. However $\mathbf{G}$ is generally not Markov.

The metric matched to the channel is given by

$$d(\mathbf{f}, \mathbf{g}) = -\ln W(g_1|f_1) - \sum_{i=1}^{L-1} \ln V(g_{i+1}|g_i, f_{i+1}, f_i). \quad (1)$$

5. LEARNING-THEORETIC APPROACH

The paper [6] describes the following approach to audio fingerprinting. A frame X is a $N_s \times N_w$ image consisting of *normalized spectral subband centroids* (NSSC) computed from N_s subbands and N_w overlapping windows. Pixel $X(n, k)$ of this image represents the centroid of the k-th critical band in a short-term power spectrum of the temporal signal, at time instant n. Then a set of linear filters indexed by $j \in \mathcal{J}$ are applied to X, and the real-valued filter outputs are quantized to four levels. This produces the *quaternary fingerprint vector*

$$F_j = \phi_j(X) \triangleq Q_j \left[\sum_{(n,k) \in \mathcal{R}_j^+} X(n, k) - \sum_{(n,k) \in \mathcal{R}_j^-} X(n, k) \right]$$

for $j \in \mathcal{J}$, where $\mathcal{R}_j^+$ and $\mathcal{R}_j^-$ are disjoint subsets of the image domain $\Omega = \{1, \cdots, N_s\} \times \{1, \cdots, N_w\}$, and the scalar quantizer Q_j has four possible output values and is parameterized by three thresholds. Hence the alphabet for the fingerprint symbol $F \triangleq \{F_j, j \in \mathcal{J}\}$ is $\mathcal{F} = 4^{\mathcal{J}}$. The fingerprint mapping $F = \phi(X) = \{\phi_j(X), j \in \mathcal{J}\}$ is determined by the choice of filters and quantizers.

A learning method dubbed *symmetric pairwise boosting* (SPB) is used to select the filters and quantizers. First a training set $\mathcal{T} \triangleq \{(X^t, Y^t, Z^t) \in \mathcal{X}^2 \times \{\pm 1\}, t \in \mathcal{T}\}$ comprised of $|\mathcal{T}|/2$ *matching pairs* and $|\mathcal{T}|/2$ *nonmatching pairs* is built, where a pair $(X^t, Y^t) \in \mathcal{X}^2$ is said to be matching if the second audio signal is a distorted version of the first, and a pair $(X^t, Y^t) \in \mathcal{X}^2$ is said to be nonmatching if the two audio signals are independent. The binary variable Z^t is equal to 1 (resp. -1) if (X^t, Y^t) is matching (resp. nonmatching). Define the classifier $h_j : \mathcal{X}^2 \to \{\pm 1\}$ as

$$h_j(X, Y) = 2\,\mathbb{1}\{\phi_j(X) = \phi_j(Y)\} - 1 \quad (2)$$

and $\mathcal{H}$ as the class of feasible classifiers (indexed by the choice of filter and quantizer). Note that $h_j(X, Y)$ depends on $(X, Y) \in \mathcal{X}^2$ only via $(\phi_j(X), \phi_j(Y)) \in \mathcal{A}^2$.

The SPB algorithm is an adaptation of the well-known *Discrete Adaboost* classification algorithm of Freund and Schapire [8] and is given in Table 1.

Upon completion of the algorithm, Adaboost would output the *boosted classifier*

$$h(X, Y) \triangleq \mathrm{sgn} \left[\sum_{j \in \mathcal{J}} \alpha_j h_j(X, Y) \right]. \quad (3)$$

However note [6] does not use the boosted classifier, only the filters and quantizers associated with each h_j are used.

Given an audio signal $\mathbf{X} = \{X_1, \cdots, X_N\} \in \mathcal{X}^N$ consisting of N frames, the fingerprint sequence is obtained as the sequence of N fingerprint vectors $\mathbf{F} = \{F_1, \cdots, F_N\} \in \mathcal{F}^N$ where $F_n = \phi(X_n)$ for each $1 \le n \le N$.

6. EXPONENTIAL LOSS FUNCTION

The Discrete Adaboost algorithm of Table 1 (with predictor variable (X, Y) and binary response variable Z) admits

Initialization: define equal weights $w_j^t = 1/|\mathcal{T}|$ for all $j \in \mathcal{J}$ and $t \in \mathcal{T}$.
Iterations: for all $j \in \mathcal{J}$, do

1. Choose the binary classifier $h_j \in \mathcal{H}$ that minimizes the weighted classification error
$$\begin{aligned} e_j &= \mathbb{E}_{\mathcal{T},w}[\mathbb{1}\{h_j(X,Y) \neq Z\}] \\ &\triangleq \sum_{t\in\mathcal{T}} w_j^t \mathbb{1}\{h_j(X^t,Y^t) \neq Z^t\}. \end{aligned}$$
2. Compute $\alpha_j = \log \frac{1-e_j}{e_j}$.
3. Update the weights
$$w_{j+1}^t = w_j^t \exp\{-\alpha_j Z^t h_j(X^t,Y^t)\}$$
4. Normalize the weights so that $\sum_{t\in\mathcal{T}} w_j^t = 1$.

Table 1. Discrete Adaboost algorithm for optimizing ϕ

a known interpretation as an *iterative procedure* for fitting an additive logistic regression model [9]

$$f(x,y) = \sum_{j\in\mathcal{J}} \alpha_j h_j(x,y) \tag{4}$$

under the *exponential loss function*

$$L(z, f(x,y)) = \exp\{-zf(x,y)\}. \tag{5}$$

Interestingly, this particular loss function is closely related to the exponential bounds on probabilities of false positives and false negatives developed in [3] which justifies its use in our coding framework.

Unconstrained minimum. The minimum of $\mathbb{E}[e^{-Zf(X,Y)}]$ over all real-valued functions $f : \mathcal{X}^2 \to \mathbb{R}$ is obtained by simple differentiation of the cost function and is half the log posterior odds [9]

$$f^*(X,Y) = \frac{1}{2} \ln \frac{P[Z=1|X,Y]}{P[Z=-1|X,Y]}.$$

It is easily verified that $\mathbb{E}[e^{-Zf^*(X,Y)}] = 1$.

7. JOINT OPTIMIZATION OF D AND ϕ

The unconstrained minimizer of the exponential loss function $\mathbb{E}e^{-Zf(X,Y)}$ is half the log posterior odds. If f is constrained to be of the form $f(X,Y) = \tilde{f}(\phi(X), \phi(Y))$, one might ask how the functions $\phi : \mathcal{X} \to \mathcal{F}$ and $\tilde{f} : \mathcal{F}^2 \to \mathbb{R}$ should be designed. Thus, as an alternative to (4), we consider a richer class of classification functions where (i) the range of the elementary functions h_j, $j \in \mathcal{J}$ is no longer constrained to be $\{\pm 1\}$, and (ii) these functions are combined in a fairly general way. We therefore consider a broader class $\mathcal{H}$ of elementary classifiers mapping $\mathcal{X}^2$ to $\mathbb{R}$.

Initialization: define equal weights $w_j^t = 1/|\mathcal{T}|$ for all $j \in \mathcal{J}$ and $t \in \mathcal{T}$.
Iterations: for all $j \in \mathcal{J}$, do

1. Choose the real-valued function $h_j \in \mathcal{H}$ that minimizes the cost function
$$\mathbb{E}_{\mathcal{T},w}[e^{-Zh_j(X,Y)}] \triangleq \sum_{t\in\mathcal{T}} w_j^t \, e^{-Z^t h_j(X^t,Y^t)}.$$
2. Update the weights
$$w_{j+1}^t = w_j^t \exp\{-Z^t h_j(X^t,Y^t)\}$$
3. Normalize the weights so that $\sum_{t\in\mathcal{T}} w_j^t = 1$.

Table 2. Real Adaboost algorithm for optimizing ϕ

The following mathematical structure will be computationally convenient. First, each elementary classifier outputs a real-valued *confidence score* determined by the choice of a filter mask and quantizer, as in [6], but also by a function $\omega : \mathcal{A}^2 \to \mathbb{R}$ to be determined. Then

$$h_j(x,y) = \omega(\phi_j(x), \phi_j(y)), \quad \forall j \in \mathcal{J}.$$

For instance, the SPB algorithm of [6] assumes the binary function $\omega(a,b) = 2\,\mathbb{1}\{a=b\} - 1$ given in (2).

We then learn the classification function

$$f(x,y) = \sum_{j\in\mathcal{J}} h_j(x,y) = \sum_{j\in\mathcal{J}} \omega(\phi_j(x), \phi_j(y)) \tag{6}$$

from training data. The goal is to minimize the exponential loss function $\mathbb{E}[e^{-Zf(X,Y)}]$ over such f. The proposed minimization algorithm is a variant of *Real Adaboost* [9] and is given in Table 2.

Note that, upon completion of the algorithm, Real Adaboost would output the *boosted classifier*

$$h(X,Y) \triangleq \operatorname{sgn}\left[\sum_{j\in\mathcal{J}} h_j(X,Y)\right] \tag{7}$$

but this classifier will not be needed here.

The function h_j can be interpreted as a *logit* transformation of a conditional probability:

$$h_j(x,y) = \frac{1}{2} \ln \frac{p_j(x,y)}{1-p_j(x,y)}, \quad x,y \in \mathcal{X}$$

$$\text{where} \quad p_j(x,y) = \hat{P}_w(Z=1|x,y) = \hat{\mathbb{E}}_w[\mathbb{1}\{Z=1\}|x,y]$$

is the weak learner's class probability estimate, $\operatorname{sgn}(h_j(x,y))$ is the weak learner's classification, and $|h_j(x,y)|$ is a measure of confidence in that classification.

This algorithm again performs a forward stagewise minimization of the expected loss, where now $f = \sum_{j\in\mathcal{J}} h_j$.

To gain further insights into the problem of optimally designing ω, we first consider the fingerprint degradation channel $W(g|f)$. Then the optimal decoding metric is the negative loglikelihood. The optimal ω is given by

$$\omega(f,g) = \tau + \ln W(g|f), \quad f,g \in \mathcal{A}. \tag{8}$$

Since the optimal ω is a function of the channel W which itself depends on the fingerprint mapping ϕ, we propose the following iterative design. First intialize ω; a reasonable choice would be (2), corresponding to the Hamming decoding metric. Then alternate between the following two steps:

1. Given ϕ, estimate W, V from the training data and obtain ω from (8).

2. Given ω, optimize ϕ using the Real Adaboost algorithm of Table 2.

8. EXPERIMENTAL EVALUATION

We consider an audio fingerprinting scenario with a training dataset $\mathcal{T}$ of size $100,000$ of which half are matched pairs ($Z = +1$) and half are non-matched pairs ($Z = -1$). The fingerprint pairs are generated at random using snippets ($T = 3$ sec) from a database of $M = 500$ songs from various musical genres.

Matched pairs are generated by applying one of the following distortions: a) Octave equalization; b) Windows Media Audio (WMA) 64kps encoding; c) Sample rate change; d) Bandpass filtering; e) Echo distortion or applying pairs of the above distortions. Non-matched fingerprint pairs are generated at random from different songs in the database.

Each audio clip is divided into $N = 8$ frames with an overlap factor of $15/16$. The feature vector $\mathbf{x} \in \mathcal{X}$ is derived from the NSSC as described in Sec. 5. Subfingerprints are obtained by applying J Viola-Jones filters to each NSSC image and quantizing each filter output to 2 bits; we chose $J = 8$ in our experiments. We use the Discrete Adaboost system of [6] as a baseline against which our algorithm performance can be compared. Performance gain of Real Adaboost is due to better selection of weak learners.

First we use the filters and quantizers of [6] and apply Step 1 of the alternating optimization algorithm of Sec. 7 to estimate the channel law (W, V) and infer the decoding metric ω. Then we apply Step 2 of the algorithm and optimize the filters and quantizers using our proposed Real Adaboost algorithm.

Training a new set of filters using Real Adaboost incurs additional computational cost. Real Adaboost increases the computation time by only a small fraction: The boosting took 12 hours for Discrete Adaboost and 14.5 hours for Real Adaboost using MATLAB on a Pentium IV processor. The incremental cost is for accessing a look-up table for W at each iteration. The decoding cost for a novel query which remains unchanged from Discrete to Real Adaboost.

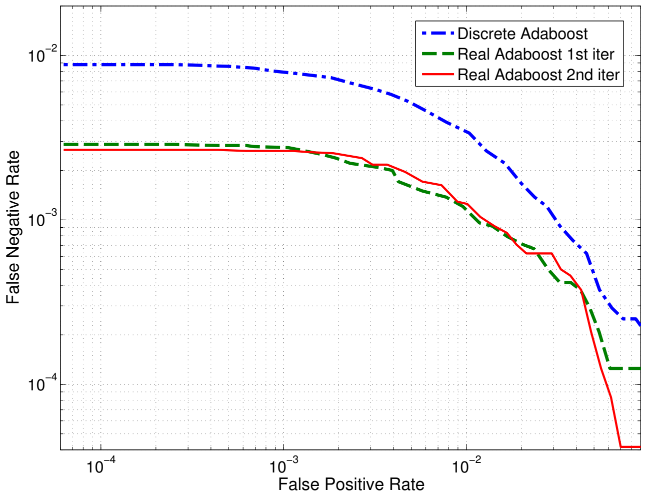

Fig. 1. Performance Comparison for Discrete vs. Real Adaboost for WMA filtering + Echo + MP3 distortion.

Performance is evaluated using an independent test dataset of $150,000$ matched fingerprint pairs and $750,000$ non-matched pairs using the same distortion set. Fingerprint decoding uses the first-order metric (1) derived from the joint-Markov assumption of the fingerprint-distortion process. A ROC performance curve is shown in Fig. 1. The first two steps of the optimization produce large gains relative to prior art, namely, a five-fold reduction of the false negative rate. Repeating these two steps produces scant improvement except at low false-positive rates.

9. REFERENCES

[1] J. Haitsma and T. Kalker, "A Highly Robust Audio Fingerprinting System," *Proc. Int. Conf. Music Information Retrieval*, 2002.

[2] S. Baluja and M. Covell, "Audio Fingerprinting: Combining Computer Vision & Data Stream Processing," *Proc. ICASSP*, Honolulu, HI, 2007.

[3] P. Moulin,"Statistical Modeling and Analysis of Content Identification," *Proc. IEEE Workshop on Information Theory and Applications (ITA)*, San Diego, CA, Jan-Feb. 2010.

[4] A. L. Varna and M. Wu, "Modeling and Analysis of Correlated Binary Fingerprints for Content Identification," *IEEE T-IFS*, Vol. 6, No. 3, pp. 1146—1159, Sep. 2011.

[5] R. Naini and P. Moulin ,"Model-Based Decoding Metrics for Content Identification," *ICASSP*, Kyoto, Japan, Mar. 2012.

[6] D. Jang, C. D. Yoo, S. Lee, S. Kim, and T. Kalker, "Pairwise Boosted Audio Fingerprint," *IEEE T-IFS*, Vol. 4, No. 4, pp. 995—1004, Dec. 2009.

[7] S. Lee, C. D. Yoo, and T. Kalker, "Robust Video Fingerprinting Based on Symmetric Pairwise Boosting," *IEEE T-CSVT*, Vol. 19, No. 9, pp. 1379—1388, Sep. 2009.

[8] Y. Freund and R. Schapire, "Experiments with a New Boosting Algorithm," *Proc. 13th Int. Conf. on Machine Learning*, pp. 148—156, 1996.

[9] J. Friedman, T. Hastie, and R. Tibshirani, "Additive Logistic Regression: A Statistical View of Boosting," *Ann. Stat*, Vol. 28, No. 2, pp. 337—407, 2000.