Regularized Adaboost Learning for Identification of Time-Varying Content

Honghai Yu, Student Member, IEEE and Pierre Moulin, Fellow, IEEE

Abstract

This paper proposes a regularized Adaboost algorithm to learn and extract binary fingerprints of time-varying content by filtering and quantizing perceptually significant features. The proposed algorithm extends the recent symmetric pairwise boosting (SPB) algorithm by taking feature sequence correlation into account. An information-theoretic analysis of the SPB algorithm is given, showing that each iteration of SPB maximizes a lower bound on the mutual information between matching fingerprint pairs. Based on the analysis, two practical regularizers are proposed to penalize those filters generating highly correlated filter responses. A learning-theoretic analysis of the regularized Adaboost algorithm is given. The proposed algorithm demonstrates significant performance gains over SPB for both audio and video content identification (ID) systems.

Index Terms

Content identification, fingerprinting, learning theory, mutual information, regularization

I. INTRODUCTION

Content identification (ID) has received considerable attention from both academia and industry. For instance, YouTube uses content ID to detect registered audio and video uploads in real time. Shazam and SoundHound use content ID for music identification on mobile devices. Other applications include advertisement tracking, broadcast monitoring, copyright control, and law enforcement [1]–[3]. In these applications, the content is encoded into a short fingerprint which allows real-time search. The fingerprint

H. Yu and P. Moulin are with the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign (UIUC), Urbana, IL 61801, USA, and Advanced Digital Sciences Center, Singapore (e-mail: yu75@uiuc.edu; moulin@ifp.uiuc.edu)

This work was supported by NSF grant CCF 12-19145 and by AUIP fellowship from the A*STAR Graduate Academy, Singapore.
must be robust to various content-preserving distortions, while being discriminative enough to distinguish perceptually different signals. The fingerprint is also known as a robust hash, or perceptual hash.

As illustrated in Fig. 1, a typical content ID system takes a snippet of a signal as a query and seeks a match in a fingerprint database. The system can be broken down into an offline part and an online part. The fingerprint database is built offline by extracting fingerprints from all database signals. When a query comes in, its fingerprint is extracted and used as a query in the fingerprint database.

![Fig. 1: Overview of a video content ID system.](image)

On the information-theoretic side, a framework for fingerprint-based content ID systems was presented in [4], and derived a fundamental relation between database size and query length under some statistical assumptions. Decoding of correlated fingerprints was studied in [5], [6]. The design of scalar quantizers inspired by the notion of biometric identification system capacity [7] was studied in [8]. The related problem of physical object identification was studied in [9].

On the algorithmic design side, various fingerprinting algorithms have been proposed based on hand-crafted signal features [1]–[3], [10]. Recently, there has been a surge of interest in learning-based similarity preserving hashing algorithms [11]–[22]. These hashing algorithms can be broadly divided into two categories: unsupervised and supervised. Locality-sensitive hashing (LSH) [11], kernelized LSH [12], [16], and spectral hashing [13] are among the most popular and well studied unsupervised algorithms. They learn hash codes to preserve certain distance metric (e.g., cosine similarity). Though unsupervised algorithms benefit from theoretical guarantees that the underlining metrics are well preserved, they require longer codes to work well and they are not applicable to general semantic similarity measures [18], [21], [22].

Semantic labels have been used in boosting [15], support vector machines (SVMs) [22], and deep neural networks [14] to learn compact binary codes in a supervised manner. Semi-supervised hashing [17] maximizes the empirical accuracy on the labeled data and uses unlabeled data as a regularizer.
Moreover, the algorithms in [18]–[21] apply under either metric or semantic similarities, and can be viewed as either unsupervised or supervised algorithms depending on the applications.

The aforementioned supervised learning algorithms have mainly been applied to static images. Except for boosting-based fingerprinting algorithms, their effectiveness has not been tested on content ID problems featuring temporally correlated feature sequences. Thus, we focus on boosting-based algorithms in this paper. In particular, fingerprinting algorithms that employ a variation of Adaboost to select filters and quantizers, such as Asymmetric Pairwise Boosting (APB) [23] and Symmetric Pairwise Boosting (SPB) [24], [25], have demonstrated excellent content ID performance and are used as the baseline methods.

As first recommended in [1], signals are segmented into overlapping slices \(^1\) in content ID systems so that in the worst-misalignment scenario, slice boundaries between the query signal and the database signal are off by only a small fraction. See Fig. 2 for an illustration. Note that overlapping slices can lead to highly correlated fingerprints, which can hurt content ID performance as we will show in later sections. Unfortunately SPB ignores statistical dependencies between signal slices.

This paper extends our previous work from [26] and presents two contributions. First, we provide an information-theoretic analysis of SPB and show that each iteration of SPB maximizes a lower bound on the mutual information between matching fingerprint pairs. Second, we propose a regularized Adaboost algorithm, which accounts for inter-slice dependencies, and provide a learning-theoretic analysis. The proposed algorithm is tested on both audio and video content ID systems, and outperforms SPB.

**Notation:** we follow the convention that uppercase letters represent random variables while lowercase letters represent particular realizations of these random variables. A vector is denoted by an underscore

\(^1\)In the literature, *frame* is sometimes used to refer to the basic building block for fingerprint extraction [1], [4]. Here, we use *slice* to avoid the confusion with the usual meaning of a video frame.
(e.g., $f$) and a temporal sequence by a boldface letter (e.g., $f$).

II. BACKGROUND

In this section, we provide a mathematical overview of the content ID problem with an emphasis on the design parameters. We focus on structured content ID codes, which encompass many current content ID algorithms, and review the SPB learning algorithm.

A. Statement of the Content ID Problem

Following [4], a content database is defined as a collection of $M$ elements, $x(m) \in X^N, m = 1, 2, \ldots, M$, each of which is a sequence of $N$ slices $\{x_1(m), x_2(m), \ldots, x_N(m)\}$. A slice could be a short video clip, a short sequence of image blocks, or a short audio snippet. Slices may be overlapping spatially, temporally, or both, to prevent misalignment during identification [1]. For instance, the video fingerprinting paper [25] uses overlapping time windows that are 1 sec long and start every 100 ms; the temporal overlap is 9/10. A 3-minute video is represented by $N = 1791$ slices. It is desired that the video be identifiable from a short clip, say 10 sec long, corresponding to $L = 91$ slices. This is called the granularity of the video ID system [25]. Typically $L \ll N$.

The problem is to determine whether a given query consisting of $L < N$ slices, $y \in X^L$, 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 $\psi(y) \in \{0, 1, 2, \ldots, M\}$, where $\psi(y) = 0$ indicates that $y$ is unrelated to any of the database elements. This is a single-output decoder. Alternatively, a variable-size list decoder $L(y) \subseteq \{1, 2, \ldots, M\}$ might be used, returning 0, 1, 2 or more matches.

For applications where a unique output is desirable, such as the YouTube content ID system, the single-output decoder is used. The variable-size list decoder is useful for applications, such as the SoundHound music identification, that can tolerate a few incorrect items as long as the correct one is on the list.

To have a comprehensive analysis of the proposed algorithm, we will consider both decoders for our experiments in this paper.

B. Performance Metrics

Different performance metrics are considered depending on which decoder is used. Two types of error, namely false positive and false negative, are associated with the single-output decoder $\psi(y)$. In [4], the content ID problem is viewed as a hypothesis testing problem with $M + 1$ hypotheses $H_0, H_1, \ldots, H_M$. 
where the null hypothesis \( H_0 \) indicates the query is unrelated to any of the database item, the probability of false positive is

\[
P_{FP} \triangleq Pr[\psi(Y) > 0|H_0],
\]

and the probability of false negative is

\[
P_{FN} \triangleq \frac{1}{M} \sum_{m=1}^{M} Pr[\psi(Y) \neq m|H_m].
\]

There are two error events of interest for the variable-size list decoder of \( \mathcal{L}(Y) \):

- **Miss**: The correct \( m \) does not appear on the decoder’s list, \( m \not\in \mathcal{L}(Y) \).
- **Incorrect Decoding**: One or more incorrect \( m' \neq m \) appear on the decoder’s list, \( m' \in \mathcal{L}(Y) \). The number of incorrect items on the list is

\[
N_i(m) \triangleq \sum_{m' \neq m} \sum_{1 \leq m' \leq M} 1\{m' \in \mathcal{L}(Y)|H_m\}.
\]

Corresponding to these two events are the probability of miss:

\[
P_{miss} \triangleq \frac{1}{M} \sum_{m=1}^{M} Pr[m \not\in \mathcal{L}(Y)|H_m]
\]

and the expected number of incorrect items on the list:

\[
\mathbb{E}[N_i] \triangleq \frac{1}{M} \sum_{m=1}^{M} \mathbb{E}[N_i(m)]
\]

\[
= \frac{1}{M} \sum_{m=1}^{M} \sum_{m' \neq m} \sum_{1 \leq m' \leq M} Pr[m' \in \mathcal{L}(Y)|H_m].
\]

C. Structured Content ID Codes

In this paper, we restrict our attention to the following fairly general class of fingerprint-based content ID codes. The codes of [1], [23]–[25] among others, fall in this category.

**Definition 1**: A \((M, N, L)\) structured content ID encoder for a size-\( M \) database populated with \( \mathcal{X}^N \)-valued content items, and granularity \( L \), is a mapping \( \phi : \mathcal{X} \rightarrow \mathcal{F} \) generating an encoding function \( \Phi : \mathcal{X}^N \rightarrow \mathcal{F}^N \) that returns a fingerprint \( f = \Phi(x) \) with components \( f_i = \phi(x_i) \) for each \( 1 \leq i \leq N \).
Hence the mapping $\phi$ is applied independently to each slice. It might be convenient to impose additional structure on the code. For instance, the mapping $\phi : X \to F$ in [24], [25] is obtained by applying a set of $J$ optimized filters to each slice and quantizing each of the $J$ real-valued filter outputs to four levels. Hence $F$ takes the form $A^J$ with $A = \{a, b, c, d\}$. In this case we view the fingerprint as an array $f = \{f_{ij}, 1 \leq i \leq N, 1 \leq j \leq J\}$ and the query fingerprint as an array $g = \{g_{ij}, 1 \leq i \leq L, 1 \leq j \leq J\}$ where $i$ denotes time and $j$ filter index. We also use the notation $\_ = \{f_j, 1 \leq j \leq J\}$ for the subfingerprint associated with a given slice. We also write $\phi$ in vector form as $\phi = \{\phi_j, 1 \leq j \leq J\}$. The length of the binary subfingerprint $\_ = \{f_j, 1 \leq j \leq J\}$.

In most content ID systems, the decoding function is constructed from a distance measure between fingerprints [4], [6], [24], [25]. Define a decoding metric $d : F^2 \to \mathbb{R}$, extended additively to pairs of subfingerprints $\{f_{i+N_0}, g_i, 1 \leq i \leq L\}$ at time offset $N_0 \in \{0, 1, 2, \ldots, N - L\}$ as follows:

$$d(f, g|N_0) \triangleq \sum_{i=1}^{L} d(f_{i+N_0}, g_i).$$ (5)

In this paper, the Hamming metric is used to allow a fair comparison with the SPB algorithm of [24], [25]. Further define the distance between the query fingerprint $g$ and the database fingerprint $f$ as the minimum over $N_0$

$$d^*(f, g) \triangleq \min_{N_0} d(f, g|N_0).$$ (6)

Based on the decoding metric $d$, the two decoders are defined as follows for a decision threshold $\tau$.

**Definition 2:** The single-output decoder is

$$\psi(g) \triangleq \begin{cases} m & \text{if } d^*(f(m), g) < \tau \text{ and } d^*(f(m), g) < d^*(f(m'), g), \forall m' \neq m \\ 0 & \text{if no such } m \text{ exists,} \end{cases}$$ (7)

When the minimizer of $d^*(f(m), g)$ is not unique, the single-output decoder declares no match ($\psi(g) = 0$). Ties could also be broken at random, but returning a single incorrect match is often more costly than returning no match.

**Definition 3:** The variable-size list decoder is

$$L(g) \triangleq \{m \in \{1, 2, \ldots, M\} : d^*(f(m), g) < \tau\}.$$ (8)

Note that $\mathbb{E}[N_i]$ can be greater than one, whereas $P_{FP} \leq 1$. Moreover, we have $P_{miss} \leq P_{FN}$ as shown in Appendix A. The decision threshold $\tau$ is associated with a point on the receiver operating characteristic (ROC) curve, and is chosen according to the desired false positive / false negative error.
probability tradeoff.

D. SPB for Filter and Quantizer Selection

Primitive signal processing features such as filters and quantizers had been heuristically chosen until learning-based methods such as Adaboost emerged [23]–[25]. Adaboost-selected filters and quantizers outperform heuristic designs [23]–[25].

The symmetric pairwise boosting (SPB) algorithm [24], [25] operates as follows. A training set $\mathcal{T} \triangleq \{(x_t, y_t, z_t) \in X^2 \times \{\pm 1\}, t \in \mathcal{T}\}$ is comprised of a subset $\mathcal{T}_+$ of $|\mathcal{T}|/2$ matching pairs and a subset $\mathcal{T}_-$ of $|\mathcal{T}|/2$ nonmatching pairs, where a pair $(x_t, y_t) \in X^2$ is said to be matching if the second signal is a distorted version of the first, and nonmatching if the two signals are independent. The binary variable (label) $z_t$ is equal to 1 (resp. -1) if $(x_t, y_t)$ is matching (resp. nonmatching). Define a set of $J$ weak classifiers $h_j : X^2 \rightarrow \{\pm 1\}, 1 \leq j \leq J$, as

$$h_j(x, y) = \begin{cases} 
+1 & \text{if } \phi_j(x) = \phi_j(y) \\
-1 & \text{otherwise}
\end{cases}$$

(9)

where $\phi_j$ is parameterized by a filter $\lambda_j : X \rightarrow \mathbb{R}$ and a quantizer $Q_j : \mathbb{R} \rightarrow \mathcal{A}$,

$$\phi_j(x) = Q_j(\lambda_j(x)).$$

(10)

Denote by $\mathcal{H}$ the class of feasible classifiers (indexed by the choice of filters and quantizers).

Fig. 3: 3-D Haar-like filters [25]: (a) spatio-temporal average, (b) temporal difference, (c,d) spatial difference, and (e,f) spatio-temporal difference. The x-coordinate is video frame index.

A popular family of filters is the Haar-like Viola-Jones filters used in [23]–[25] which are easy to compute and rich enough to describe perceptually significant visual features. The filter outputs for the 3-D Haar-like filters in [25] are the average difference between values in light and dark regions shown in Fig. 3.

To reduce the computational complexity of the training, a limited number of candidate quantizers are evaluated. In [24], 19 candidate thresholds that minimize the mean squared quantization error of the filter
responses of the training data are considered. In [25], 17 logarithmically spaced candidate thresholds are considered. For 4-level quantization, \(|A| = 4\), 969 and 680 candidate quantizers are evaluated for each filter for 19 and 17 candidate thresholds respectively.

The SPB algorithm is an adaptation of the well-known Adaboost classification algorithm given in Table I. Upon completion of the algorithm, Adaboost would output the boosted classifier

$$h_B(x, y) \triangleq \text{sgn} \left[ \sum_{1 \leq j \leq J} \alpha_j h_j(x, y) \right].$$

However the algorithms of [24], [25] do not use the boosted classifier. Only the filter \(\lambda_j\) and quantizer \(Q_j\) associated with each \(h_j\) are used to produce the fingerprints. The weights \(\{\alpha_j\}\) could be used to compute a weighted Hamming distance \(D(f, g) = \sum_{j=1}^J \alpha_j d_H(f_j, g_j)\), where \(d_H\) denotes the Hamming distance. However, in our content ID experiments, decoders based on Hamming and weighted Hamming distances yield similar results, whereas computing weighted Hamming distance is considerably slower. Thus, we simply report results for Hamming distance.

| Input: | training set \(\mathcal{T} \triangleq \{(x_t, y_t, z_t) \in \mathcal{X}^2 \times \{\pm 1\}, t \in \mathcal{T}\} \|
| Initialization: | define equal weights \(w_t^{(1)} = 1/|\mathcal{T}|, \forall t \in \mathcal{T}\) |
| Do for \(j = 1, \ldots, J\) | 
| 1) | Choose the classifier \(h_j\) that minimizes the weighted error over \(h \in \mathcal{H}\) \(e_j = \sum_{t \in \mathcal{T}} w_t^{(j)} \mathbb{I}\{h(x_t, y_t) \neq z_t\}\). \(\quad (11)\) |
| 2) | Compute \(\alpha_j = \frac{1}{2} \log \frac{1 - e_j}{e_j}\). |
| 3) | Update the weights \(w_t^{(j+1)} = w_t^{(j)} \exp\{-\alpha_j z_t h_j(x_t, y_t)\}\). |
| 4) | Normalize the weights so that \(\sum_{t \in \mathcal{T}} w_t^{(j+1)} = 1\). |
| Output: | \(J\) pairs of filter and quantizer \(\{ (\lambda_j, Q_j) \}_{j=1}^J\) parameterizing the chosen \(J\) classifiers \(\{h_j\}_{j=1}^J\). |

TABLE I: Adaboost for filter and quantizer selection.

### III. Mutual Information between Fingerprints

In this section we first review the relevance of mutual information for fingerprint code design, then establish a connection to Adaboost, and finally set up a framework for handling temporal dependencies within fingerprint sequences.
A. Mutual Information and Content ID capacity

A content ID system, like any other communication system, is subject to a fundamental capacity limit that upper bounds the rate at which information can be decoded with arbitrarily low probability of error. The content ID capacity is the supremum of all fingerprint code rates such that both $E[N_i]$ and $P_{\text{miss}}$ vanish as $L \to \infty$ [4]. For an iid signal process $X$, memoryless degradation channel, and fixed structured content ID code (Def. 1 and Def. 3) with mapping $\phi : X \to F$, the content ID capacity is given by $C = I(F; G)$, where $G$ is a distorted version of fingerprint $F$ and is stochastically related to $F$ via the conditional probability distribution $P_{G|F}$. If $\phi$ is a code design parameter, then $C = \max_{\phi} I(F; G)$.

Roughly speaking, the largest database that can be handled is $M \approx 2^{LC}$. When the signal $X$ is an ergodic stationary process and the degradation channel from $X$ to $Y$ is stationary ergodic, we propose to use the closely related design criterion $C_L = \max_{\phi} \frac{1}{L} I(F; G)$. The normalized mutual information $\frac{1}{L} I(F; G)$, is a nondecreasing function of the number of classifiers $J$, which is fixed here. Furthermore, in the analysis of Adaboost-based fingerprinting methods, we make the reasonable assumption that the mutual information is approximately additive over filters, i.e., $I(F; G) \approx \sum_{j=1}^{J} I(F_j; G_j)$. This assumption is justified by the near-independence between learned filters for both SPB and regularized Adaboost.

B. Information-Theoretic Analysis of SPB

We now show that at each iteration $1 \leq j \leq J$, SPB maximizes a lower bound on the mutual information $I(F_j; G_j) = H(F_j) - H(F_j|G_j)$ associated with the joint probability distribution $P_{F_j,G_j}$ induced by the choice (10) of $\phi_j$. Indeed we may rewrite (11) as follows. SPB selects the weak classifier that minimizes the weighted error

$$h_j = \arg \min_{h \in \mathcal{H}} \left[ \sum_{t \in T_+} w_t^{(j)} I\{h(x_t, y_t) = -1\} + \sum_{t \in T_-} w_t^{(j)} I\{h(x_t, y_t) = 1\} \right],$$

(12)

where the two error terms are the empirical weighted false-negative and false-positive error probabilities, respectively. For a given classifier $h \in \mathcal{H}$, the empirical version of the false-negative error probability for matching fingerprints, $P_{e,j} = P_{F_j,G_j}(F_j \neq G_j)$, is given by

$$P_{e,j} = \Pr(F_j \neq G_j|T_+, h) = \sum_{t \in T_+} w_t^{(j)} I\{h(x_t, y_t) = -1\},$$

(13)
and the empirical false-positive error probability, $P_{Fj}P_{G_j}(F_j = G_j)$, is

$$\hat{Pr}(F_j = G_j|\mathcal{T}_-,h) = \sum_{t \in \mathcal{T}_-} w^{(j)}_t \mathbb{1}\{h(x_t,y_t) = 1\}. \quad (14)$$

First, we derive a link between $\hat{Pr}(F_j \neq G_j|\mathcal{T}_+,h)$ and $\hat{H}(F_j|G_j)$. By Fano’s inequality [27]

$$H(F_j|G_j) \leq h_2(P_{e,j}) + P_{e,j} \log(|A| - 1), \quad (15)$$

where $P_{e,j} \triangleq P_{F_j,G_j}(F_j \neq G_j)$, $A$ is the alphabet for $F_j$, and $h_2(P_{e,j})$ is the binary entropy function. One may expect that a similar inequality holds using the empirical version of $H(F_j|G_j)$ and $P_{e,j}$:

$$\hat{H}(F_j|G_j) \lesssim h_2(\hat{P}_{e,j}) + \hat{P}_{e,j} \log(|A| - 1). \quad (16)$$

We have observed empirically that inequality (16) is tight. Fig. 4a shows the empirical equivocation $\hat{H}(F_j|G_j)$ and Fano’s upper bound $h_2(\hat{P}_{e,j}) + \hat{P}_{e,j} \log(|A| - 1)$ evaluated from 16,000 matching pairs and 16 classifiers. In view of (13) and (16), minimizing $\hat{Pr}(F_j \neq G_j|\mathcal{T}_+,h)$ is tantamount to minimizing a tight upper bound on the empirical conditional entropy $\hat{H}(F_j|G_j)$.

![Fig. 4: Simulation results of (16) and (18). The x-coordinate is the classifier index j.](image)

Next, we derive a link between $\hat{Pr}(F_j = G_j|\mathcal{T}_-,h)$ of (14) and $\hat{H}(F_j)$. When $F_j$ and $G_j$ are generated from nonmatching pairs, we model them by a product distribution with identical marginals. From Lemma 2.10.1 in [27], we have $P_{F_j,G_j}(F_j = G_j) \geq 2^{-H(F_j)}$, for two iid random variables $F_j$ and $G_j$. Hence $H(F_j)$ is lower bounded by

$$H(F_j) \geq -\log P_{F_j,G_j}(F_j = G_j). \quad (17)$$

March 19, 2016 DRAFT
The empirical version of (17) would be

\[ \hat{H}(F_j) \gtrsim -\log \hat{\Pr}(F_j = G_j|T_-, h). \]  

(18)

Again, we have observed empirically that (18) holds and is tight from nonmatching pairs, as shown in Fig. 4b. Thus, minimizing \( \hat{\Pr}(F_j = G_j|T_-, h) \) is tantamount to maximizing a tight lower bound on \( \hat{H}(F_j) \).

From the above argument, we conclude that each iteration \( 1 \leq j \leq J \) of SPB simultaneously minimizes an upper bound on \( H(F_j|G_j) \) and maximizes a lower bound on \( H(F_j) \), thus maximizes a lower bound on \( I(F_j; G_j) = H(F_j) - H(F_j|G_j) \).

C. Temporal Dependencies

In content ID systems, slices are temporally overlapped to overcome misalignment during identification, which results in temporally correlated fingerprints \( \mathbf{F}_j = \{ F_{1j}, F_{2j}, \ldots, F_{Lj} \} \) for each chosen classifier \( h_j \). In a memoryless channel where each \( G_{ij} \) only depends on \( F_j \) only via \( F_{ij} \), we have [27]

\[ I(\mathbf{F}_j; \mathbf{G}_j) \leq \sum_{i=1}^{L} I(F_{ij}; G_{ij}). \]  

(19)

Equality holds when the input components \( \{ F_{1j}, F_{2j}, \ldots, F_{Lj} \} \) are independent. Conversely,

\[ I(\mathbf{F}_j; \mathbf{G}_j) \ll \sum_{i=1}^{L} I(F_{ij}; G_{ij}), \]  

(20)

when \( \{ F_{1j}, F_{2j}, \ldots F_{Lj} \} \) are highly correlated. Thus we can increase the mutual information by decorrelating temporal fingerprints. Many slice-wise distortions can be modeled as memoryless channels, including resizing, cropping and rotation.

In the next section, we show that the classifiers’ ability to decorrelate slices differs dramatically across different types of filters. In order to increase mutual information by decorrelating temporal fingerprints, we propose to use a regularizer to effectively eliminate from the candidate pool \( \mathcal{H} \) those filters that generate highly correlated fingerprints. Experiments demonstrate the effectiveness of this regularizer.

IV. REGULARIZED ADABOOST

A shortcoming of Adaboost for filter selection is the implicit assumption that slices are drawn independently from some unknown distribution. In practice, slice overlapping is necessary to overcome misalignment during identification. For instance, the papers [24] and [25] use overlapping factors of
10/11 and 9/10 respectively. Then slices are significantly correlated. In this section, we propose two regularizers to improve the content ID performance of the Adaboost algorithm in Table I.

**Input:** training set \( T \triangleq \{(x_t, y_t, z_t) \in X^2 \times \{\pm 1\}, t \in T \} \)

**Initialization:** define equal weights \( w_t^{(1)} = 1/|T|, \forall t \in T \)

**Do for** \( j = 1, \ldots, J \)

1) Choose the classifier \( h_j \) that minimizes the weighted error over \( h \in \mathcal{H} \)

\[
\varepsilon_{j}^{\text{REG}} = \sum_{t \in T} w_{t}^{(j)} \mathbb{1}\{h(x_t, y_t) \neq z_t\} + \gamma R(h). \tag{21}
\]

2) Compute \( \alpha_j = \frac{1}{2} \log \frac{1 - \varepsilon_{j}^{\text{REG}}}{\varepsilon_{j}^{\text{REG}}} \).

3) Update the weights

\[
w_{t}^{(j+1)} = w_{t}^{(j)} \exp\{-\alpha_j z_t h_j(x_t, y_t)\}.
\]

4) Normalize the weights so that \( \sum_{t \in T} w_{t}^{(j+1)} = 1 \).

**Output:** \( J \) pairs of filter and quantizer \( \{(\lambda_j, Q_j)\}_{j=1}^{J} \) parameterizing the chosen \( J \) classifiers \( \{h_j\}_{j=1}^{J} \).

**TABLE II:** Regularized Adaboost for filter and quantizer selection. \( R(h) \) is a generic regularizer. We use \( R(h) = I(h) \) for MIR Adaboost and \( R(h) = \overline{I}(h) \) for ACCR Adaboost.

\[A. \textbf{Mutual Information of Gauss-Markov Process as a Regularizer}\]

The first regularizer we propose is based on a first-order stationary Gauss-Markov process model for the filter response \( \lambda(X) \). The statistical structure of the centered process is completely determined by the correlation coefficient between two consecutive samples \( \lambda(X_i) \) and \( \lambda(X_{i+1}) \). Equivalently, the process is characterized by the mutual information between \( \lambda(X_i) \) and \( \lambda(X_{i+1}) \) [27]

\[
I(\lambda) = -\frac{1}{2} \log(1 - \rho^2), \tag{22}
\]

where \( \rho \in [-1, 1] \) is the correlation coefficient between \( \lambda(X_i) \) and \( \lambda(X_{i+1}) \). The functional \( I(\lambda) \) captures the filter’s ability to decorrelate consecutive slices and can be easily estimated from the training dataset. In Fig. 5, we show the estimated \( I(\lambda) \) for the family of Haar-like filters of Fig. 3 applied to video data. Within the family, type (b), (e) and (f) filters compute temporal differences and decorrelate temporal overlapping slices extremely well. Type (a) filters compute the average pixel intensity across the 3-D volume and produce highly correlated responses due to high temporal overlapping. For the spatial difference filters, type (d) filters produce less correlated responses than type (c) filters because horizontal
camera movement is normally more frequent than vertical movement resulting in more spatial difference in the horizontal direction.

Fig. 5: Mutual information \( I(\lambda) \) for the family of Haar-like filters on video slices.

From (22), we see that for small \( |\rho| \), \( I(\lambda) \) can be approximated by a linear function of \( |\rho| \), while for large \( |\rho| \), \( I(\lambda) \) increases much faster than linearly. We penalize filters with large mutual information between consecutive output samples using the new objective function

\[
e_{j}^{\text{REG}} = \sum_{t \in T} w_{t}^{(j)} 1\{h(x_t, y_t) \neq z_t\} + \gamma I(h),
\]

where \( I(h) = I(\lambda) \) indicates the weak classifier \( h \) is parameterized by the filter \( \lambda \), and \( \gamma \geq 0 \) is the regularization parameter which can be chosen by cross validation. If \( \gamma = 0 \), filters are selected by their weighted error on the training dataset without considering their ability to decorrelate overlapping slices. If \( \gamma \to \infty \), filters are chosen solely based on \( I(h) \). By varying the parameter \( \gamma \), one can explore the tradeoff between a filter’s ability to classify a single slice and to decorrelate overlapping slices. The mutual information regularized (MIR) Adaboost is given in Table II by replacing \( R(h) \) with \( I(h) \).

B. Average Correlation Coefficient as a Regularizer

Our second regularizer makes no Markovian or stationarity assumption about the filter response process. For a given filter \( \lambda \) (such as those from Fig. 3), the response \( \lambda(X) = \{\lambda(X_i), 1 \leq i \leq L\} \) is a \( L \)-dimensional random vector. Denote by \( \rho(s, t) \in [-1, +1] \) the correlation coefficient between two random variables \( \lambda(X_s) \) and \( \lambda(X_t) \). Define the average correlation coefficient (ACC) of \( \lambda(X) \) as

\[
\overline{\rho}(\lambda) \triangleq \frac{1}{L^2 - L} \sum_{s \neq t} |\rho(s, t)|.
\]

March 19, 2016
The functional $\overline{\rho}(\lambda)$ captures the filter’s ability to decorrelate overlapping slices and can be easily estimated from the training dataset. Similarly to $I(\lambda)$ in Fig. 5, we observe a similar contrast pattern across different types of filters, as shown in Fig. 6. But $I(\lambda)$ displays a larger dynamic range than $\overline{\rho}(\lambda)$ due to its heavier penalty for large $\rho$.

![Fig. 6: Average correlation coefficient $\overline{\rho}(\lambda)$ for the family of Haar-like filters on video slices.](image)

We penalize filters with large ACC using the new objective function

$$e_j^{\text{REG}} = \sum_{t \in T} w_{i, t}^{(j)} \mathbf{1}\{h(x_t, y_t) \neq z_t\} + \gamma \overline{\rho}(h), \quad (25)$$

where $\overline{\rho}(h) = \overline{\rho}(\lambda)$ indicates the weak classifier $h$ is parameterized by the filter $\lambda$. The ACC regularized (ACCR) Adaboost is given in Table II by replacing $R(h)$ with $\overline{\rho}(h)$.

C. Learning-Theoretic of the Regularized Adaboost Algorithm

In this section, we show that the regularized Adaboost in Table II fits an additive logistic regression model

$$\delta(x, y) = \sum_{1 \leq j \leq J} \alpha_j h_j(x, y),$$

under the regularized exponential loss function

$$L(z, \delta(x, y)) = \exp\{-z\delta(x, y)\} + \gamma \sum_{1 \leq j < J} 2 \sinh(\alpha_j) R(h_j). \quad (26)$$

The analysis is inspired by [28], [29] and does not depend on the specific form of the regularizer. As long as the regularizer is a functional of $h$, it can be plugged into the regularized Adaboost algorithm and the same analysis applies, which makes this approach fairly general. Hence, we show the derivation for a generic regularizer $R(h)$.
Using the regularized exponential loss function (26), at iteration \( j \), one must solve

\[
(\alpha_j, h_j) = \arg \min_{\alpha \in \mathbb{R}, h \in \mathcal{H}} \left[ \sum_{t \in \mathcal{T}} w_t^{(j)} \exp\{-\alpha z_t h(x_t, y_t)\} + 2 \sinh(\alpha) \gamma R(h) \right],
\]

where \( w_t^{(j)} = \exp\{-z_t \delta_{j-1}(x_t, y_t)\} \) and

\[
\delta_i(x, y) \triangleq \sum_{1 \leq j \leq i} \alpha_j h_j(x, y).
\]

Using the fact that \( h(x, y) \in \{-1, 1\} \), the objective function of (27) can be expressed as

\[
\left[ e^{-\alpha} \sum_{h(x_t, y_t) = z_t} w_t^{(j)} + e^\alpha \sum_{h(x_t, y_t) \neq z_t} w_t^{(j)} \right] + (e^\alpha - e^{-\alpha}) \gamma R(h),
\]

which in turn can be written as

\[
\left[ (e^\alpha - e^{-\alpha}) \sum_{t \in \mathcal{T}} w_t^{(j)} \mathbb{1}\{h(x_t, y_t) \neq z_t\} + e^{-\alpha} \sum_{t \in \mathcal{T}} w_t^{(j)} \right] + (e^\alpha - e^{-\alpha}) \gamma R(h).
\]

Since \( \sum_{t \in \mathcal{T}} w_t^{(j)} = 1 \), the objective function becomes

\[
2 \sinh(\alpha) \left[ \sum_{t \in \mathcal{T}} w_t^{(j)} \mathbb{1}\{h(x_t, y_t) \neq z_t\} + \gamma R(h) \right] + e^{-\alpha}.
\]

The minimum over \( h \in \mathcal{H} \) is given by

\[
h_j = \arg \min_{h \in \mathcal{H}} \sum_{t \in \mathcal{T}} w_t^{(j)} \mathbb{1}\{h(x_t, y_t) \neq z_t\} + \gamma R(h).
\]

Plugging \( h_j \) into (28) and solving for \( \alpha \), we obtain

\[
\alpha_j = \frac{1}{2} \log \frac{1 - e_j^{\text{REG}}}{e_j^{\text{REG}}},
\]

where \( e_j^{\text{REG}} \) is given by (21). Equation (29) and (30) are equivalent to Step 1 and 2 of the regularized Adaboost algorithm in Table II.

**V. EXPERIMENTAL RESULTS**

In this section, we test the proposed MIR and ACCR Adaboost algorithms for both video and audio content ID systems. The results are compared with SPB in [25] and [24]. We examine the content ID performance based on the two decoders defined in Section II-C.
A. Video Fingerprinting

1) Experimental Setup: The video dataset we use contains 1,700 randomly selected videos from the publicly available Internet Archive videos (IACC.1.C) [30]. The archive covers a variety of genres including news, politics, animation, education, animals, vehicles, music and sports. We randomly divide the 1,700 videos into training, validation, and testing subsets consisting of 100, 100, and 1,500 videos respectively. From the training videos, we generate 20,000 matching and 20,000 nonmatching pairs (|T| = 40,000) of 1 sec video sequences. The training pairs are generated from the following video distortions:

1) Cropping of 25%;
2) Resizing to CIF (352 × 288);
3) Frame rate change to 15 fps;
4) WMV lossy compression at 256 kb/s;
5) Rotation at 10 degrees;
6) Shifting downward and left by 20 pixels.

We adopt the same video normalization as in [25]. Videos are resampled at 10 fps, converted to grayscale, and resized to QVGA (320 × 240). These preprocessing steps aim to make the fingerprinting algorithm robust to frame rate change, color variation, and image resizing. After preprocessing, we extract intermediate features from each image before applying filters. The intermediate feature used in our experiments and in [25] is block mean luminance (BML), which is perceptually significant and reduces computational complexity. The BML is extracted on 36 (4 × 9) blocks per frame. One second of intermediate features (4 × 9 × 10 blocks) becomes the basic building block (a slice) for fingerprint extraction.

The training dataset contains an equal number of pairs from each distortion, and 𝑍 = 16 filters and quantizers are selected by SPB or regularized Adaboost. Each filter output is quantized into 4 levels (𝐴 = {𝑎, 𝑏, 𝑐, 𝑑}) and converted to binary fingerprint by gray code. As noted in Section II-D, SPB chooses each quantizer from 680 and 969 candidate quantizers for the video and audio fingerprinting systems respectively. Besides the high computation cost, these candidate quantizers are not chosen based on any optimality criterion for content ID performance. In the paper [23], where |𝐴| = 2, the authors noted that all thresholds learned by APB were approximately at the median of the filter response distribution. Putting a threshold at the median maximizes the bit entropy. Similarly, for |𝐴|-level quantization, we propose to use the |𝐴| quantiles of the filter response distribution as the thresholds for a given filter. This
quantization scheme produce bins with equal probabilities \(1/|A|\) and therefore maximizes bit entropy. It also makes the training process much faster (nearly three orders of magnitude faster than evaluating 19 candidate thresholds). Note that achieving maximum entropy for each bit is a desirable property for many hashing algorithms [13], [18], [20], [31]. However, uniform distribution does not necessarily lead to maximization of the mutual information (except for some simple channels, e.g., symmetric channels), which is the proposed objective function for selecting hash functions in Section III-A. Here, we choose such a quantization rule mainly due to the simpler training requirement.

We choose the regularization parameter \(\lambda\) by validation on a few candidate choices. The validation set contains 100 videos independent from both training and testing. For the single-output decoder, we select the \(\lambda\) that generates the smallest \(P_{FN}\) at a fixed \(P_{FP}\) of interest. While for the variable-size list decoder, we seek the smallest \(P_{miss}\) at a fixed \(E[N_i]\) of interest.

The training time for regularized Adaboost is the same as SPB. To select 16 filters, the training time for both SPB and regularized Adaboost is 596 s on a desktop with Intel Xeon W3530 @ 2.80GHz processor and 6GB RAM.

2) Selected Filters: In all our experiments, SPB selects more filters from the high-correlation group (type (a), (c) and (d) filters), whereas filters chosen by regularized Adaboost are dominated by the low-correlation group (type (b), (e) and (f) filters). As Adaboost reweighs training examples after each iteration, to correctly classify those higher weighted examples (incorrectly classified in previous iterations) may require a different type of filters. Thus, SPB selects different types of filters to best fit the training examples. However, in regularized Adaboost, reducing weighted classification error is not the only objective at each iteration. The ability to decorrelate overlapping slices in order to increase mutual information is also considered. The regularizers effectively demote filters of type (a), (c) and (d) which generate highly correlated responses on overlapping slices. The superiority of the low-correlation filters is demonstrated next in a comparative test.

3) Comparative Test: To compare the content ID performance of SPB and regularized Adaboost, we generate 25,200 queries of 10-second intermediate feature sequences by applying the six distortions to the 1,500 testing videos, where the original 1,500 testing videos serve as the database in estimating \(P_{FN}\), \(P_{miss}\) and \(E[N_i]\). To estimate \(P_{FP}\) of the single-output decoder, we use the leave-one-subject-out (LOSO) scheme. In each run, we choose the samples from one testing video as the queries, and the remaining testing videos serve as the database.

Fig. 7 shows the content ID performance under the single-output decoder and the list decoder. Irrespective of the regularizer and decoder used, regularized Adaboost outperforms SPB. Fig. 7 reports the
overall performance under different video distortions. The performance under each individual distortion follows the same trend. Note that $E[N_i]$ can be much larger than one, but we only show $E[N_i] < 1$ as they represent the most relevant region for a practical content ID system.

![Graph](image1)

(a) Single-output decoder.

![Graph](image2)

(b) List decoder.

Fig. 7: Video content ID performance.

B. Audio Fingerprinting

1) Experimental Setup: The audio dataset is a collection of 1,700 songs spanning a variety of music genres including classical, vocal, rock and pop. We randomly divide the 1,700 songs into training, validation, and testing subsets consisting of 100, 100, and 1,500 songs respectively. From the training songs, we generate 22,400 matching and 22,400 nonmatching SSC image pairs. The audio distortions are created by the software GoldWave [32] and the audio distortions considered in this paper are as follows:

1) Bandpass filtering (BPF): 400 Hz to 4 kHz bandpass filtering.
2) Echo (E): Tunnel reverberation.
3) Equalization 1 (EQ1): Boost bass.
4) Equalization 2 (EQ2): Recording industry association of America (RIAA).
5) Audio slice misalignment (ASM): 92.9 ms shift.
6) Sampling rate change (SR): Down-sampling to 16 kHz.
7) Volume change (V): Attack-Decay-Sustain-Release (ADSR) envelop.
8) WMA encoding (WMA): 64 kb/s WMA encoding.
On top of the above distortions, each audio signal is encoded by 96 kb/s MP3 encoding.

We follow the same experimental setup as in [24] for audio fingerprinting. An audio signal is first normalized to mono with 11,025 Hz sampling rate, and then converted into overlapping segments by a window with size 371.52 ms and shift 185.76 ms. For every segment, an $M$-dimensional spectral subband centroid (SSC) vector is computed [33] from $M = 16$ critical subband linearly spaced in mel scale from 300 Hz to 5300 Hz. A SSC image, built from $N = 10$ consecutive SSC vectors, is the basic building block (a slice) for fingerprint extraction. For every shift of 185.76 ms, an SSC image is obtained from an audio slice of length 2.04 s.

Different from the 3-D Haar-like filters for video (see Fig. 3), the candidate filters for audio are 2-D Haar-like filters (see Fig. 8) applied on $M \times N$ SSC images.

\[ \text{Fig. 8: 2-D Haar-like filters [24]: The filter outputs are the average difference between values in light and dark regions.} \]

The training set contains an equal number of SSC image pairs from each distortion. Similar to video fingerprinting, $J = 16$ filters are selected. Each filter response is quantized into $A = 4$ levels and converted to binary fingerprint by gray code. The regularization parameter $\gamma$ is chosen by validation on a few candidate choices. To select 16 filters, the training time for both SPB and regularized Adaboost is 152 s on a desktop with Intel Xeon W3530 @ 2.80GHz processor and 6GB RAM.

2) Comparative Test: To compare the content ID performance of SPB and regularized Adaboost, we generate 112,000 queries of 10-second SSC images equally divided by the eight considered distortions. As in [24], the overlapping factor between two consecutive slices is 10/11 and a 10-second query corresponds to $L = 44$ slices.

Fig. 9 shows the audio content ID performance under both decoders. Irrespective of the regularizer and decoder used, regularized Adaboost outperforms SPB. Moreover, ACCR Adaboost and MIR Adaboost performs comparably.

3) Audio Slice Misalignment: Most content ID systems use a high overlapping factor for fingerprint extraction. Though slice overlapping increases system complexity, it is a practical compromise to overcome misalignment between query fingerprint and database fingerprint. For our audio content ID system, a 2.04 s subfingerprint is extracted for every 185.76 ms shift. So the worst misalignment we will encounter
To further examine the effect on misalignment for audio content ID systems, we add the worst slice misalignment to each distortion. As shown in Fig. 10, MIR Adaboost and ACCR Adaboost still outperform SPB under both decoders. Comparing Fig. 9 with Fig. 10, both SPB and regularized Adaboost perform worse under slice misalignment. However, the misalignment has a stronger impact on MIR Adaboost than ACCR Adaboost because MIR penalizes filters generating highly correlated responses on two consecutive slices, where correlation between two consecutive slices helps alleviate slice misalignment. Note that the misalignment problem does not exist for our video query fingerprints because the shift is a single frame.

VI. CONCLUSION

Audio and video content ID systems use substantial overlapping of slices to mitigate misalignment during identification [1]–[3]. Hence fingerprinting algorithms such as SPB [24], [25] produce highly correlated fingerprints. While some correlation in fingerprints is useful to combat misalignment, information-theoretic analysis and real world experiments show that too much correlation is undesirable. Our proposed fingerprinting algorithm is based on the boosting framework and uses a regularization term to control the amount of fingerprint correlation and improve content ID performance. We have proposed a mutual information regularizer (MIR) and an average correlation coefficient regularizer (ACCR), both of which are easy to compute and can capture the filter’s ability to decorrelate overlapping slices. Significant
performance gains over SPB have been demonstrated for both video and audio content ID systems.

APPENDIX A

RELATION BETWEEN $P_{FN}$ AND $P_{miss}$

To derive the link between $P_{FN}$ and $P_{miss}$ for the two decoders in Def. 2 and Def. 3, we express $P_{FN}$ and $P_{miss}$ in terms of the decision regions.

For the single-output decoder $\psi$, the decision region $R_m^\psi$ for $m \in \{1, 2, \ldots, M\}$ is given by

$$R_m^\psi = \{ g : d^* (f(m), g) < \tau \text{ and } d^* (f(m'), g) < d^* (f(m''), g), \forall m' \neq m \}.$$  \hfill (31)

For the variable-size list decoder $L$, the decision region $R_m^L$ for $m \in \{1, 2, \ldots, M\}$ is given by

$$R_m^L = \{ g : d^* (f(m), g) < \tau \}.$$  \hfill (32)

While $\{R_m^\psi\}$ are disjoint sets, $\{R_m^L\}$ are generally overlapping. It follows from (2) and (3) that

$$P_{FN} = \frac{1}{M} \sum_{m=1}^{M} \sum_{g \notin R_m^\psi} Pr[g|H_m],$$  \hfill (33)

$$P_{miss} = \frac{1}{M} \sum_{m=1}^{M} \sum_{g \notin R_m^L} Pr[g|H_m].$$  \hfill (34)
Clearly, we have $R_{\psi m} \subseteq R_{\ell m}$. Therefore,

$$P_{\text{miss}} \leq P_{\text{FN}}.$$  \hspace{1cm} (35)

**ACKNOWLEDGMENT**

The authors thank the reviewers for the careful reading of the manuscript and valuable suggestions.

**REFERENCES**


